AN ISOGEOMETRIC ANALYSIS APPROACH FOR MORPHOELASTIC MODELS

APPLICATION TO SKIN CONTRACTURE

by

Antonio Barion

in partial fulfillment of the requirements for the degree of

Master of Science
in Applied Mathematics

at the Delft University of Technology,
to be defended publicly on Thursday November 5, 2020 at 14:00.

Supervisor:	Prof. dr. ir. F. Vermolen	
Thesis committee:	Prof. dr. ir. F. Vermolen,	UHasselt
	Dr. M. Möller,	TU Delft
	Dr. H. M. Schuttelaars,	TU Delft

Delft
University of
Technology

PREFACE

This work marks the conclusion of a journey started two years ago. Two years filled with people, experiences and challenges I probably would never have encountered, if I had stayed home. However, this work is far from being purely the result of my individual efforts, as I was given constant support by multiple people.

I would like to express my gratitude to my supervisor Fred Vermolen. His passion to this project was clear from the very first moment and contributed to my decision to choose this topic. Even though it was not possible to have meetings in person, your kindness always reached me. Thank you also Matthias Möller for the precious insights you have given me throughout this months. Although we only collaborated towards the end of my work, I wanted to thank you Hugo Verhelst. Your assistance and constant helpfulness have been of great aid.

Lastly, I want to thank my parents and relatives for always supporting me and believing in my capabilities. Grazie Gianluca, for being a great friend and housemate and thank you to all my friends which have been close to me, even if some were physically distant. Grazie Giulia, per aver intrapreso questo viaggio con me.

Antonio Barion
Delft, November 2020

ABSTRACT

Skin grafting is a common technique employed to treat patients after burn injuries. In contrast to the frequency and gravity of contractures following from skin grafts, the phenomenon itself is still poorly understood and subject of studies. The development of an accurate model of skin contractures will allow medical researchers to better understand the healing process compared to current in vitro experiments and thus enable the design of efficient and patient-specific treatments. In this work we provide a mathematical model able to capture both the mechanical and biological processes involved in skin graft healing. Two-dimensional morphoelastic equations are used to model the mechanics of contraction of skin during the healing process. A system of equations describing the cell components (myo-, fibroblasts, collagen density and signalling molecule) is solved separately to derive the forcing terms for the mechanical system. The finite element method with bilinear quadrilateral elements is used to solve the differential equations on a moving time-dependent domain. A flux corrected transport (FCT) algorithm is used to stabilize the biological equations, thus enforcing positivity of the constituents in the dermal layer and nonlinearities are treated using Picard iterations. Time stepping is performed by applying the Euler backward scheme for the mechanical system and the implicit midpoint rule for the biological system. The results are replicated using the IGA (Isogeometric Analysis) library G+Smo in C++. The results of the IGA implementation provide a solid basis for future algorithms using IGA.

CONTENTS

I

PART I

1

INTRODUCTION

The focus of this work is on burn wounds and their healing process. Annually in the UK, 13,000 hospitalizations are because of burn injuries. Globally, more than 6 million patients require skin grafts because of burns each year [1]. In general, patients who suffer from chronic wounds experience physical, psychological and social problems [2]. The constant pain, which can derive from a chronic wound, continuously affects the quality of life of the patient. Burn victims are often subject to this type of wounds, as the scarring process of third-degree burn wounds (or higher) is complex and takes a considerably long time. Not only does the patient suffer from these conditions, but he/she also has to deal with a possibly impaired mobility, if the burn wound is big enough or in proximity of joints. The most frequent treatment of dermal wounds caused by burns is skin grafting. The burned skin is excised and a 'healthy' skin graft is placed to cover the open wound area. A study shows that more than 30% of patients, who were subject to skin graft treatment, suffer from hypertrophic scars and contractures [3]. Furthermore, given the complexity of the healing process, alterations of the latter occur frequently, which further encourages the development of hypertrophic scars and contractures. A study performed in the USA between 1993 and 2002 reported that almost 40% of the burn patients had been discharged from the hospital with at least one contracture. The mean number of contractures was three and the most frequently affected joint was the shoulder, followed by elbow and knee [4]. In contrast to the frequency and gravity of contractures following from skin grafts, the phenomenon itself is still poorly understood and subject of studies [1].

1.1. MOTIVATION

Most techniques applied to prevent skin graft contractures involve counteracting the contracting mechanical forces arising in the wound. Two common approaches are splinting joints and wearing pressure garments. In [5] it is highlighted how general guidelines for the counteraction of skin graft contractures are still missing. This further aggravates the problem at hand. It is widely accepted that splinting and pressure treatment are most effective in the early stages of the healing process and are performed for 6 months to a year post-operatively [1]. As we see, these types of treatments only prolong the period of medical treatment, thus further postponing the patient's return to a normal life. In the case of already developed contractures, additional excision and grafting of the area is performed, which only reinitializes the whole problem. In [1] it is stressed that a deeper understanding of the cellular and molecular mechanisms of graft contraction is necessary in order to advance in both treatment and prevention.

While in vitro experiments are certainly possible and are being performed, e.g. [1], the complex interactions occurring are difficult to follow experimentally. As often done in design engineering, one can gain a better insight into physical/chemical dynamics through the use of mathematical modelling. In [6] it has been shown how mathematical models can be used to describe the healing process of skin grafts. Using the morphoelastic models developed in [7], it is possible to track the deformation of skin and the evolution of its strains and stresses. We think that the main cause of a patient's pain or discomfort, when dealing with contractures, is due to residual stresses in the skin tissue after the healing process. Aim of this work is to devise a reliable algorithm to solve the mathematical model which describes, both mechanically and (bio)chemically, the development of skin graft contractures during wound healing. We will also investigate the possibility of using new discretization techniques, such as isogeometric analysis (IGA), to solve our equations. The flexibility of IGA in the representation of geometries would make it an ideal tool to approximate the geometrically complex boundaries which could arise during the contracture of skin grafts. Hence, we would be able to accurately follow the healing process. Furthermore, with the development of such an algorithm, medical experts would have a much more versatile device at their disposal to investigate the dynamics of skin graft contractures, opposed to in vitro experiments. With the help of more accurate, adaptable methods to study the problem at hand, we believe it possible, through close collaboration with doctors, medical researchers and staff, to advance and meliorate the current situation in regard to the treatment of skin graft contractures. Furthermore, the time needed to perform simulations could be reduced substantially with the aid of neural networks, see [8]. All this could lead to real time predictions of the evolution of burn injuries, such that patient specific treatments can become a widespread reality in our hospitals.

1.2. STRUCTURE

This work is divided in two parts. In the first part all concepts needed for a finite element analysis of the problem are described and the results deriving from it are shown. The application of isogeometric analysis and its results are presented in the second part.

In Chapter 2 we begin our work with a brief description of the biological processes involved in skin wound healing. The composition of skin and the different healing phases are described. The fundamental mathematical concepts utilized are explained in Chapter 3. We then present the mathematical equations which model the mechanical and biological dynamics of skin wounds in Chapter 4. The necessary discretizations and the numerical methods applied are described in Chapter 5. The considered equations give rise to undesired properties of the solution. The flux correcting algorithm, employed to rectify this issue, is presented in Chapter 6. The results of the simulations are then shown in Chapter 7.

In Chapter 8 we give a general introduction to the fundamental concepts of IGA. The results of the implementation of this novel method are shown in Chapter 9.

We finish our work with Chapter 10 by presenting our conclusions and some indications for future work on the topics that have been touched upon.

2

BIOLOGICAL BACKGROUND

The skin is our biggest organ and performs various crucial tasks. First and foremost, the skin provides protection to our body from potentially lethal factors from the outside world. It also serves as one of our main feedback mechanisms when interacting with objects, as it gives us information about pressure, heat, texture, etc. Being arguably the most exposed of our organs, it is constantly subject to (minor) injuries and needs to be replaced and/or repaired constantly. The process of tissue healing is complex and involves multiple mechanisms. We will now briefly introduce the main contributors in wound healing to have a better overview of the work to come. We will base our descriptions on [2, 6, 9].

2.1. SKIN LAYERS

The skin can be divided in three layers: the epidermis, the dermis and the hypodermis, see Figure 2.1.1.

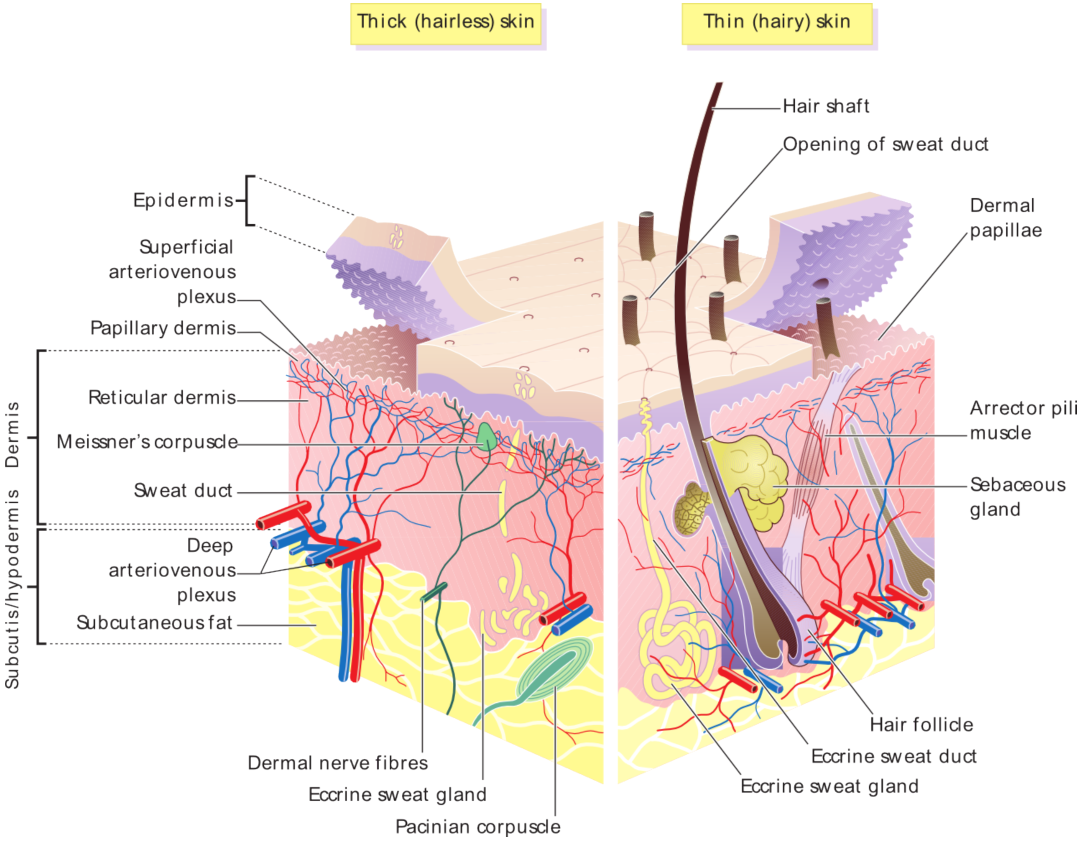


Figure 2.1.1: Representation of skin layers. Taken from [6].

The outermost layer is the epidermis. It is responsible for the production of keratin, which makes up the peripheral physical barrier, and melanin, which protects us from ultraviolet (UV) radiation. The epidermis is completely replaced by new cells over a period of approximately one month.

The dermis is the intermediate layer and it is the thickest of the three. In the dermis we find the papillary and reticular dermis, hair follicles, sebaceous glands and sweat glands necessary for tissue regeneration. Blood vessels permeate this skin layer. The extracellular matrix (ECM) is also to be found in the dermis. It is composed by the basement membrane, the fibers and the extrafibrillar matrix components. As it occupies most of the dermal layer, it is responsible for the regulation of intercellular communication and resisting mechanical forces. Fibroblasts, being the most common type of cell in the dermis, are responsible for the secretion of precursors of the components of the ECM. Fibroblasts are not fully specialized cells, which means they can further differentiate into different types. In our work, we will pay considerable attention to the differentiation of fibroblasts into myofibroblasts.

Lastly, the innermost layer is the hypodermis. Its main purpose is to attach the dermis to the muscles and bones. It houses blood vessels and nerves, and it plays a role in the regulation of body temperature.

2.2. WOUND HEALING

After injury, the process of wound healing starts almost immediately and involves multiple complex cellular and biochemical events. Wounds can be classified into epidermal and dermal wounds. Epidermal wounds are superficial and do not affect the dermis. Their healing process usually leads to the recovery of the wound surface and no scars [2]. We will focus on dermal wounds, where the dermal matrix is destroyed. The four phases of dermal wound healing are: haemostasis, inflammation, proliferation, and remodelling and scar maturation [9]. These phases, while being distinct, do overlap during the healing process as seen in Figure 2.2.1.

Haemostasis Haemostasis is the first reaction of our body to a dermal wound and it kicks in almost instantaneously after the injury. As soon as the blood vessels are damaged and blood flows into the wound, our organism tries to limit further blood loss through constriction of vessels and platelet aggregation. In this phase, different growth proteins are secreted which attract and activate fibroblasts, endothelial cells and macrophages. Clot formation also occurs during haemostasis. The clot serves as provisional matrix for cell migration in the absence of the ECM.

Inflammation After initial constriction, the vessels dilate to favour the transport of the leukocytes. Among the different types of leukocytes we can find polymorphonuclear leukocytes, monocytes and T lymphocytes. Bacteria and other foreign particles are phagocytosed and in general bacterial contamination is combated. Everything occurs in order to prevent infections. Macrophages come into play in the later stages of the inflammatory phase. They are key regulatory cells for repair, as they are the primary producers of growth factors for proliferation of smooth muscle cells, of endothelial cells resulting in angiogenesis and production of the extracellular matrix by fibroblasts.

Proliferation Shortly after inflammation, the proliferative phase commences. This phase is characterized by fibroblast migration, deposition of the extracellular matrix and formation of granulation tissue. The dermis, which is the main physical barrier, is gradually restored through reepithelialization. Angiogenesis helps to deliver nutrients and to transport new cells through the wound area. In the first days, fibroblasts mainly migrate and proliferate, while later they are crucial factors in the reconstruction of the collagen matrix. Fibroblasts contribute to the structural integrity of connective tissue by continuously secreting precursors of the components of the extracellular matrix [2]. A provisional matrix is formed to facilitate the interaction between cells and it will later be substituted by the normal ECM which contains more collagen. Multiple factors influence the migration and proliferation of fibroblasts. One agent, which serves both as chemokines for fibroblasts chemotaxis and as proliferation signal, is the transforming growth factor β (TGF-β). Furthermore, fibroblasts can differentiate into myofibroblasts. Like the fibroblasts, myofibroblasts are responsible for the production of constituents of the ECM and matrix metalloproteinase (MMP), which helps regulate the collagen deposition dynamics. Wound contraction takes place during the proliferation phase and is a natural response to the body to hinder further intrusion of bacteria and other external particles into the wound by reducing the exposed area. Production of resistant scar tissue takes time, hence a wound area reduction is beneficial. Myofibroblasts attach to the ECM and pull it when they contract, thus reducing the wound size.

Remodelling Phase and Scar Maturation Conversely to the relatively short time window in which the previous phases act, the remodelling of the ECM takes place over much longer periods of time, lasting one year or more. The type III collagen present in the ECM is substituted by the stronger type I collagen. Collagen bundles are being rearranged, cross-linked, and aligned along tension lines, thus more parallel to the surface of the skin compared to the collagen bundles of uninjured dermal tissue [6]. Gradually, the concentration levels of the different cells decrease, leaving a relatively acellular scar tissue of slowly increasing strength. However, it will never regain the strength of unwounded skin, only reaching a maximum of 80% of the original strength.

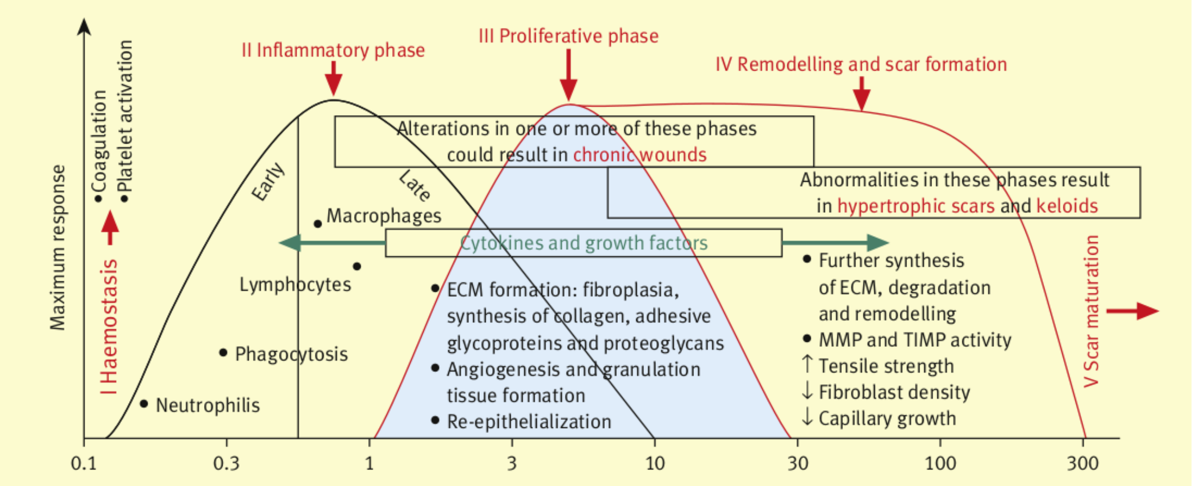


Figure 2.2.1: Phases of wound healing [9].

2.3. BURNS AND SKIN GRAFTS

In this section a short overview of various dermal burn wounds is given based on the information found in [10]. This topic will connect burns, skin grafts and contractures to the mathematical simulations we performed.

Burns at a dermal level are divided into three categories. First-degree burns only affect the epidermis and cause minimal damage and blisters. If the burns penetrate the dermis, they are considered a partial-thickness burn and are categorized as second-degree. They are further differentiated into partial-thickness and deep partial-thickness burns. The former case only reaches the papillary dermis and can take 10 to 14 days to heal with little to no scar formation. If the burn reaches the reticular dermis, it is referred to as a deep partial-thickness burn. In this case, healing will take over three weeks and lead to hypertrophic scarring (raised scar). Burns which fully penetrate the dermis and hypodermis are called third-degree burns. This type of wound is slow to heal and should be excised, which causes extensive scar formation. To replace the missing patch of skin, a layer of healthy skin is usually harvested from the patient and placed on the open wound. Sometimes skin substitutes are used instead. While wound area contraction is a common biological response, contractures are reduction in size of the scar surface. Therefore, contractures constrict the adjacent tissues resulting in physical disability [3]. Hypertrophic scar formation in burn patients is common, as they are hypermetabolic and in a pro-inflammatory state and thus experience prolonged wound-healing. Contractures also contribute to the formation of residual stresses in the skin tissue which, in addition to the aforementioned motory impairment, cause pain for the patient. In this work we aim to model the contraction and contracture of skin grafts via a morphoelastic mechanical model.

3

NUMERICAL FRAMEWORK

In this chapter we will briefly touch upon the most important ingredients which are necessary for our work. A concise description of the setup of bilinear quadrilateral elements is given. Additionally, we introduce the concept of Lagrangian framework, which will be pivotal to our work.

3.1. BILINEAR QUADRILATERAL FINITE ELEMENTS

Keeping in mind the future extension to Isogeometric Analysis (IGA), we have chosen to use bilinear quadrilateral elements to discretize our computational domain. In fact, IGA does rely on quadrilateral elements, thus making a comparison further on more reliable. Indeed, for first order basis functions IGA will reveal itself to be equivalent to traditional linear FEM.

The main difference, between the more commonly used linear triangular and the bilinear quadrilateral elements, is the existence of exact integration formulas, e.g. Holand-Bell's integration rules [11], for the former opposed to the lack thereof for the latter. Furthermore, there is no general expression for the description of continuous basis functions across (quadrilateral) elements of the domain [12, Ch. 8.3]. An immediate consequence is the increase in computational time, since the element matrices will have to be computed for each element opposed to having a fixed representation applicable to any element of our mesh, see [13]. To circumvent any possible discontinuities arising from a bilinear basis, we take advantage of the isoparametric transformation from the standard element in the reference domain ξ, η. Let $\boldsymbol{x}_i$ be the vertices of the quadrilateral, then each vertex is associated to a corner of the unit square $\hat{\boldsymbol{x}}_i$ as follows:

$$\begin{aligned} \boldsymbol{x}_1 &\to (0,0) = \hat{\boldsymbol{x}}_1, \quad & \boldsymbol{x}_2 &\to (1,0) = \hat{\boldsymbol{x}}_2, \\ \boldsymbol{x}_3 &\to (1,1) = \hat{\boldsymbol{x}}_3, \quad & \boldsymbol{x}_4 &\to (0,1) = \hat{\boldsymbol{x}}_4. \end{aligned}$$

The basis functions in the reference domain are:

$$\hat{\varphi}_1 = (1-\xi)(1-\eta), \quad \hat{\varphi}_2 = \xi(1-\eta), \quad \hat{\varphi}_3 = \xi\eta, \quad \hat{\varphi}_4 = (1-\xi)\eta,$$

where $\hat{\varphi}_i(\hat{\boldsymbol{x}}_j) = \delta_{ij}$. The isoparametric transformation $\psi : (\xi, \eta) \to (x, y)$, from the reference element to the general element, utilizes the aforementioned basis functions to parametrize from reference to global coordinates: $\boldsymbol{x}(\xi,\eta) = \sum_{i=1}^{4} \boldsymbol{x}_i \hat{\varphi}_i(\xi,\eta)$. It can be shown, that the gradients of the basis functions in the computational domain are related to the gradients of the reference basis functions through:

$$\nabla_{(x,y)} \varphi_i = J^{-T} \nabla_{(\xi,\eta)} \hat{\varphi}_i,$$

where J is the Jacobian of the isoparametric transformation ψ, defined by

$$J(\xi,\eta) = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix}.$$

Abusing notation we consider

$$J(x,y) = \begin{bmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \\ \frac{\partial \eta}{\partial x} & \frac{\partial \eta}{\partial y} \end{bmatrix},$$

where $J(x,y)$ is the Jacobian of the inverse of ψ. It is immediate to see that $J(x,y) = J^{-1}(\xi,\eta)$. As such we will refer to $J(x,y)$ as J^{-1} from now on. Through the isoparametric transformation ψ and its inverse ψ^{-1}, we have that every global basis function $\varphi_i(x,y)$ can be represented as $\varphi_i(x,y) = \hat{\varphi}_i(\psi^{-1}(x,y))$; note that we are again abusing notation as the indices of the global basis functions are $i = 1,...,N$, where N is the number of nodes in our mesh, while the indices of the reference basis functions are $i = 1,...,4$. We now apply the chain rule of differentiation:

$$\frac{\partial\varphi_i}{\partial x} = \frac{\partial\hat{\varphi}_i}{\partial x} = \frac{\partial\hat{\varphi}_i}{\partial\xi}\frac{\partial\xi}{\partial x} + \frac{\partial\hat{\varphi}_i}{\partial\eta}\frac{\partial\eta}{\partial x},$$
$$\frac{\partial\varphi_i}{\partial y} = \frac{\partial\hat{\varphi}_i}{\partial y} = \frac{\partial\hat{\varphi}_i}{\partial\xi}\frac{\partial\xi}{\partial y} + \frac{\partial\hat{\varphi}_i}{\partial\eta}\frac{\partial\eta}{\partial y},$$

and thus

$$\nabla_{(x,y)}\varphi_i = \begin{bmatrix} \frac{\partial\xi}{\partial x} & \frac{\partial\eta}{\partial x} \\ \frac{\partial\xi}{\partial y} & \frac{\partial\eta}{\partial y} \end{bmatrix} \nabla_{(\xi,\eta)}\hat{\varphi}_i.$$

This shows exactly how $\nabla_{(x,y)}\varphi_i = J^{-T}\nabla_{(\xi,\eta)}\hat{\varphi}_i$, and we are able to evaluate the Jacobian of ψ^{-1} without knowing an explicit form for it.

As a result, it is now possible to calculate integrals of basis functions over a general element with domain Ω_{el} as follows:

$$\begin{aligned} s_{ij} &= \int_{\Omega_{el}} \nabla\varphi_i \cdot \nabla\varphi_j \, d\Omega \\ &= \int_{\hat{\Omega}_{el}} \left(J^{-T}\nabla_{(\xi,\eta)}\hat{\varphi}_i\right)\cdot\left(J^{-T}\nabla_{(\xi,\eta)}\hat{\varphi}_j\right) |\det J| \, d\hat{\Omega}, \end{aligned}$$

where $\hat{\Omega}_{el}$ is the domain of the reference element.

As for the computation of integrals which arise during the discretization, we applied the Gaussian quadrature product rule. Given the 1D Gaussian weights $(\omega_m)_{m=1}^N$ and quadrature points $(\xi_m)_{m=1}^N, (\eta_m)_{m=1}^N$, a double integral is calculated as

$$\begin{aligned} \int\int f(\xi,\eta)\, d\xi d\eta &\approx \int \sum_{k=1}^{N} \omega_k f(\xi_k,\eta)\, d\eta \\ &\approx \sum_{j=1}^{N}\sum_{k=1}^{N} \omega_j\omega_k f(\xi_k,\eta_j). \end{aligned}$$

In literature it is most common to find the Gaussian weights and points for the interval $[-1,1]$, while our reference square is $[0,1]^2$. It is straightforward to show that the Gaussian weights ω_m and points x_m on $[-1,1]$ relate to their counterpart $\widetilde{\omega}_m$ and $\widetilde{x}_m$ on $[0,1]$. We consider the parametrization $g_1 : [-1,1] \to [a,b]$ of the interval $[a,b]$ of the form $g_1(x) = \frac{b-a}{2}x + \frac{b+a}{2}$. If we want to integrate a general function f over the interval $[a,b]$ using standard Gaussian integration, we proceed as follows:

$$\int_a^b f(s)ds = \int_{-1}^{1} f(g_1(x))\frac{b-a}{2}\,dx \approx \frac{b-a}{2}\sum_{m=1}^{N}\omega_m f(g_1(x_m)).$$

Let $g_2 : [0,1] \to [a,b]$ be a different parametrization of the same interval $[a,b]$ expressed as $g_2(\widetilde{x}) = (b-a)\widetilde{x}+a$. Again we would like to evaluate the integral of f over $[a,b]$. However, we will now use the Gaussian weights and points $\widetilde{\omega}_m$ and $\widetilde{x}_m$ on $[0,1]$.

$$\int_a^b f(s)ds = \int_0^1 f(g_2(x))(b-a)\,dx \approx (b-a)\sum_{m=1}^{N}\widetilde{\omega}_m f(g_2(\widetilde{x}_m)).$$

Since we wish the two approximations to be equivalent, we have that

$$\frac{b-a}{2}\sum_{m=1}^{N}\omega_m f(g_1(x_m)) = (b-a)\sum_{m=1}^{N}\widetilde{\omega}_m f(g_2(\widetilde{x}_m)).$$

It is apparent how, by imposing $\widetilde{\omega}_m = \frac{\omega_m}{2}$ and $(b-a)\widetilde{x}_m + a = \frac{b-a}{2}x_m + \frac{b+a}{2}$, the above equation holds true. These conditions result in:

$$\begin{cases} \widetilde{\omega}_m = \frac{\omega_m}{2}, \\ \widetilde{x}_m = \frac{x_m}{2} + \frac{1}{2}. \end{cases}$$

We can now translate the Gaussian points and weights from the interval $[-1,1]$ to $[0,1]$.

3.2. LAGRANGIAN FRAMEWORK

We are interested in observing the contraction of skin during the healing process of burn injuries. As such, if we model the skin layer with our mesh, we will end up working with a moving mesh simulation. To keep track of the nodal points, the mesh is updated at each time step with the velocity field of the previous time step. Thus, the mesh velocity field is supposed to be a stepwise constant function in time. Let $\boldsymbol{X}(t)$ be the vector/matrix with the mesh points information at time t and $\boldsymbol{V}(t)$ the respective velocity field, then the mesh points at time $t+\Delta t$ are calculated as follows:

$$\boldsymbol{X}(t+\Delta t) = \boldsymbol{X}(t) + \Delta t \boldsymbol{V}(t).$$

To summarize, we could say that we are interested in individual particles, nodes in our case, as they move through space in time. This is a common description used in solid mechanics [14] and is referred to as the Lagrangian description. The motion of particles is described by a function $x = h(X,t)$, which maps the domain from its original state, i.e. particles in position X at time t_0, to its current state at time t, i.e. particles in position x. The velocity V of a particle X at time t is:

$$V(X,t) = \frac{\partial x}{\partial t} = \frac{\partial h(X,t)}{\partial t}.$$

Conversely, the Eulerian description fixates on one point in space and describes the property of particles passing through it. Let v represent the velocity in the Eulerian frame, then the two specifications are related through:

$$v(x,t) = v(h(X,t),t) = \frac{\partial h(X,t)}{\partial t}.$$

Both V and v describe the velocity of a particle in position x at time t that was in position X at time t_0.

We will now consider the property $\boldsymbol{F}$. We are interested in the total rate of change of $\boldsymbol{F}$ experienced by a particle, hence $\boldsymbol{F} := \boldsymbol{F}(x(X,t),t)$ (where $x(X,t) := h(X,t)$). Thus, the rate of change in time of our property $\boldsymbol{F}$ in one (fixed) point will be described by $\partial\boldsymbol{F}/\partial t$. However, to monitor the rate of change in time of $\boldsymbol{F}$ in a particle along its path, we will consider the so called material/Lagrangian derivative $D\boldsymbol{F}/Dt$,

$$\frac{D\boldsymbol{F}}{Dt} = \frac{\partial \boldsymbol{F}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{F},$$

where $\boldsymbol{v}$ is the flow velocity field. The material derivative can be interpreted as the total derivative in time of the property $\boldsymbol{F}$ following the flow $x(X,t)$ of the particle with initial position X and thus relates the Lagrangian to the Eulerian description:

$$\begin{aligned} \frac{d\boldsymbol{F}(x(X,t),t)}{dt} &= \frac{\partial \boldsymbol{F}}{\partial t} + \frac{d(x(X,t))}{dt}\nabla\boldsymbol{F} \\ &= \frac{\partial \boldsymbol{F}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{F}. \end{aligned}$$

4

MATHEMATICAL MODELS

A main part of the initial work done consisted in replicating existing results with the addition of quadrilateral elements in the discretization. In order to arrive at the results given in [13] and [6], several intermediate steps have been taken.

We constructed a 1D morphoelastic model. Successively, we implemented a 2D viscoelastic model and lastly we tackled 2D morphoelasticity. This concluded the mechanical aspects of this work.

It is important to grasp the concepts behind the mechanical equations we are going to solve. This will allow us to develop a sort of intuition on what to expect, thus increasing our familiarity with the problem at hand. While viscoelasticity is more commonly known, the concept of morphoelasticity (coined by Goriely and Ben Amar [15]) is a more recent addition to the variety of mechanical equations of materials. The first description and introduction of a zero stress state in biomechanics was done by Rodriguez et al. [16] in 1994.

To conclude, we introduce the biochemical equations described in [6].

4.1. STRAINS AND STRESSES

In this section we will loosely follow the contents explained in [14]. Before we describe the definitions of stress and strain, we need to introduce the concept of displacement. Let $\boldsymbol{X}$ be the original position of a particle which moved to the coordinates $\boldsymbol{x}$, then the Eulerian displacement or deformation vector $\boldsymbol{u}(\boldsymbol{x})$ is defined as: $\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{X}$. Analogously, the Lagrangian deformation vector $\boldsymbol{U}(\boldsymbol{X})$ is defined as: $\boldsymbol{U}(\boldsymbol{X}) = \boldsymbol{x}(\boldsymbol{X}) - \boldsymbol{X}$, where we suppose that the deformed position $\boldsymbol{x}$ is a known function of $\boldsymbol{X}$.

Strain is the measure of the deformation of a body with respect to its initial state at rest. To formalize this concept, we start by considering two points P and Q in the original configuration with infinitesimal distance $d\boldsymbol{X}$ and their deformed counterparts P', Q' with distance $d\boldsymbol{x}$. Since we supposed $\boldsymbol{x}$ to be a function of $\boldsymbol{X}$ we can define the so called deformation gradient $A := d\boldsymbol{x}/d\boldsymbol{X}$. The Lagrangian finite strain tensor $\boldsymbol{E}$ is defined by

$$\boldsymbol{E} = \frac{1}{2}(A^T A - I),$$

where I is the identity matrix. We can immediately verify that the tensor is symmetric. If we recall the definition of $\boldsymbol{U}$ and A, we have that the deformation gradient is given by $\nabla \boldsymbol{U} = A - I$. Thus, since $A = \nabla \boldsymbol{U} + I$ we have that the relation between displacement and Lagrangian strain is described by

$$\boldsymbol{E} = \frac{1}{2}(\nabla \boldsymbol{U} + \nabla \boldsymbol{U}^T + \nabla \boldsymbol{U}^T \nabla \boldsymbol{U}).$$

The Eulerian finite strain tensor is defined by

$$\boldsymbol{e} = \frac{1}{2}(I - A^{-T} A^{-1}).$$

As its Lagrangian counterpart, also $\boldsymbol{e}$ is symmetric. We recall the definitions of $\boldsymbol{u}$ and A and derive that $\nabla \boldsymbol{u} = I - A^{-1} \implies A^{-1} = I - \nabla \boldsymbol{u}$. Thus, the relationship between Eulerian displacement and strain reads:

$$\boldsymbol{e} = \frac{1}{2}\left(\nabla \boldsymbol{u}^T + \nabla \boldsymbol{u} - \nabla \boldsymbol{u}^T \nabla \boldsymbol{u}\right).$$

Lastly, we note how for infinitesimal strains/deformations, the Eulerian strain tensor reduces to Cauchy's strain tensor $\boldsymbol{\varepsilon}$

$$\boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \boldsymbol{u}^T + \nabla \boldsymbol{u}\right).$$

The distinction between Eulerian and Lagrangian strain tensor disappears (i.e. $\boldsymbol{E} \approx \boldsymbol{e} \approx \boldsymbol{\varepsilon}$).

Stress is a reaction to external forces on the surface of a body. The stress vector is defined as

$$\boldsymbol{T}^{(\boldsymbol{n})} = \frac{d\boldsymbol{F}}{d\Gamma} = \lim_{\Delta\Gamma \to 0} \frac{\Delta \boldsymbol{F}}{\Delta\Gamma},$$

where $\boldsymbol{n}$ is the normal vector of the surface Γ and $\boldsymbol{F}$ represents the force acting on the surface. Let us imagine a cube and consider a force acting along the z-axis on its upper face. We will call the correspondent stress the normal stress σ_{zz} and it holds true that $T^{(\boldsymbol{e}_3)} = \sigma_{zz}$ ($\boldsymbol{e}_3$ is the unitary vector along the z-axis). If the force is not exactly aligned with the z-axis, we decompose it into three components along the axes σ_{zx}, σ_{zy}, σ_{zz}, where the notational convention is $\sigma_{\text{face, direction}}$. Analogously, we can proceed with the remaining two axes. The three components $\sigma_{xx}, \sigma_{yy}, \sigma_{zz}$ are called normal stresses, while the remaining six are called shear stresses. The Cauchy stress tensor is defined as

$$\boldsymbol{\sigma} = [T^{(\boldsymbol{e}_1)}, T^{(\boldsymbol{e}_2)}, T^{(\boldsymbol{e}_3)}] = \begin{bmatrix} \sigma_{xx} & \sigma_{yx} & \sigma_{zx} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{zy} \\ \sigma_{xz} & \sigma_{yz} & \sigma_{zz} \end{bmatrix}.$$

This expression allows for a different representation of $\boldsymbol{T}^{(\boldsymbol{n})}$. In fact, let $\boldsymbol{T}^{(\boldsymbol{n})}$ be the stress vector acting on the surface $d\Gamma$ with normal $\boldsymbol{n}$, then we have that $\boldsymbol{T}^{(\boldsymbol{n})} = \boldsymbol{\sigma}\boldsymbol{n}$.

From the next section on, $\overline{\overline{\boldsymbol{B}}}$ will denote a second order tensor.

4.2. Linear Viscoelasticity

In order to discuss viscoelasticity we must first touch upon the concept of elasticity. The physical property, for which a body deforms under forces and returns to its original shape once these cease to act upon it, is called elasticity. For elastic materials, the stress is only related to the strain at that moment and not to its history. The most prominent example of elastic behaviour is the spring. As for the spring, where the force is equal to the product of spring constant and displacement, so for the Hookean elastic model the stress (force per unit of area) relates to strain (displacement per unit of length) through

$$\sigma_{ij} = c_{ijkl}\varepsilon_{kl},$$

where c_{ijkl} is the elasticity tensor. If we consider an isotropic material, i.e. c_{ijkl} is invariant under rotation and reflection, then the stress can be described by

$$\sigma_{ij} = v\delta_{ij}\varepsilon_{kk} + 2\mu\varepsilon_{ij},$$

where v and μ are called Lamé parameters. In our case we will consider

$$\overline{\overline{\boldsymbol{\sigma}}} = \frac{E\sqrt{\rho}}{1+\nu}\left(\overline{\overline{\boldsymbol{\varepsilon}}} + \frac{\nu}{1-2\nu}\operatorname{Tr}(\overline{\overline{\boldsymbol{\varepsilon}}})\overline{\overline{\boldsymbol{I}}}\right),$$

where $E\sqrt{\rho}$ is a density-dependent Young's modulus, ν is Poisson's ratio, $\overline{\overline{\boldsymbol{I}}}$ is the second order identity tensor and Tr is the trace operator. The addition of the density term is introduced in [6], usually only E is used.

A material is described as viscoelastic, if it displays both viscous (dash-pot) and elastic (spring-like) properties when subject to deformation. In purely elastic materials the resistance to deformation (stresses) is only dependent on the difference of the configuration from its status at rest. In viscoelastic materials also the rate of deformation influences the material properties. To model these behaviours, multiple mechanical analogues of spring and dash-pot couplings have been used. We will follow the Kelvin-Voigt model, where a spring and a dash-pot are coupled in parallel, see Figure 4.2.1. In one dimension the spring and dash-pot strain models are respectively described by

$$\sigma = k\varepsilon \quad \text{and} \quad \sigma = \eta\frac{d\varepsilon}{dt},$$

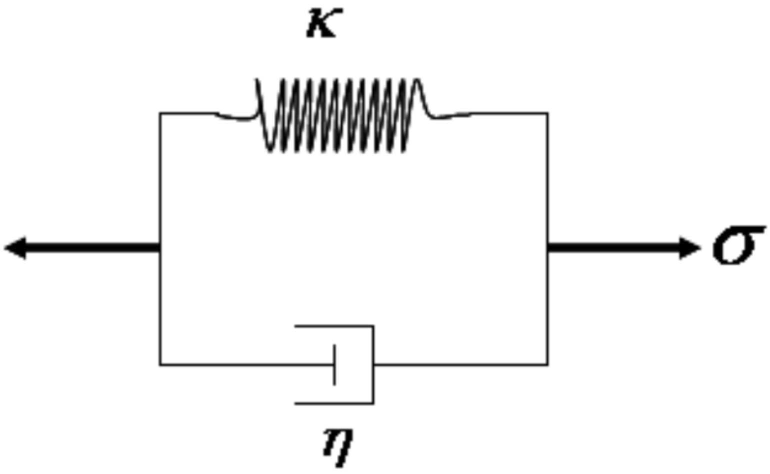


Figure 4.2.1: Schematic representation of the Kelvin-Voigt model. Taken from [14].

with k being the elastic modulus of the material and η its viscosity.

If we now recall the definition of Cauchy's stress tensor and reduce it to one dimension, then we have that $\frac{d\varepsilon}{dt} = \frac{dv}{dx}$ where v is the velocity. Thus the one dimensional viscoelastic law reads

$$\sigma = k\varepsilon + \eta\frac{dv}{dx}.$$

We will use the following three-dimensional expansion of the aforementioned viscoelastic constitutive law with the added modification for the Young's modulus:

$$\overline{\overline{\boldsymbol{\sigma}}} = \mu_1\,\mathrm{sym}(\nabla\boldsymbol{v}) + \mu_2(\nabla\cdot\boldsymbol{v})\overline{\overline{\boldsymbol{I}}} + \frac{E\sqrt{\rho}}{1+\nu}\left(\overline{\overline{\boldsymbol{\varepsilon}}} + \frac{\nu}{1-2\nu}\mathrm{Tr}(\overline{\overline{\boldsymbol{\varepsilon}}})\overline{\overline{\boldsymbol{I}}}\right),$$

where μ_1 is the shear viscosity and μ_2 the bulk viscosity. In [15] it is noted how in experiments the re-expansion of skin, after the completion of the healing process, is not instantaneous. Although it being much faster than the contraction process, the viscous relaxation was noticeable and hence a Kelvin-Voigt constitutive law was deemed appropriate to capture this phenomenon.

4.3. MORPHOELASTICITY

If we bend a metal spoon and it does not snap back to its original position, then the deformation is described as *plastic*. Once stress is exerted on the material beyond a certain limit (yield stress), the particles slip relatively to each other. This allows the strains to increase significantly, leaving the related stresses unchanged. As a result, there is a change in the constitutive law between stresses and strains. The aim of morphoelasticity is to be able to encompass these complicated relations within one mechanical description. The underlying idea is that there exists a relation between the original state, the zero stress state and the current state of a body. Let us introduce the concept of a zero stress state of the body, i.e. a configuration in which 'the body would like to be', as described in [7]. In our spoon example this would be the bended configuration after we stopped applying force on it. Imagine now, that we are still holding and actively bending the spoon. Morphoelasticity theory affirms the existence of mappings between the body in its original state (undeformed spoon), the body in its zero stress state (how the spoon would be after we release it) and the body in its current state (the spoon while we are still actively bending it). Hence, it would be possible to track the zero stress state, which reflects the absence of elastic stresses, and thus have a description of the residual stresses after deformation, see Figure 4.3.1.

We will now follow the one dimensional presentation in [15] to explain the derivation of the equations of strain evolution. In Section 4.1 we already encountered the deformation gradient A. As hinted at, we write A as the product of the elastic stretch α and the growth stretch γ:

$$A = \alpha\gamma. \tag{4.3.1}$$

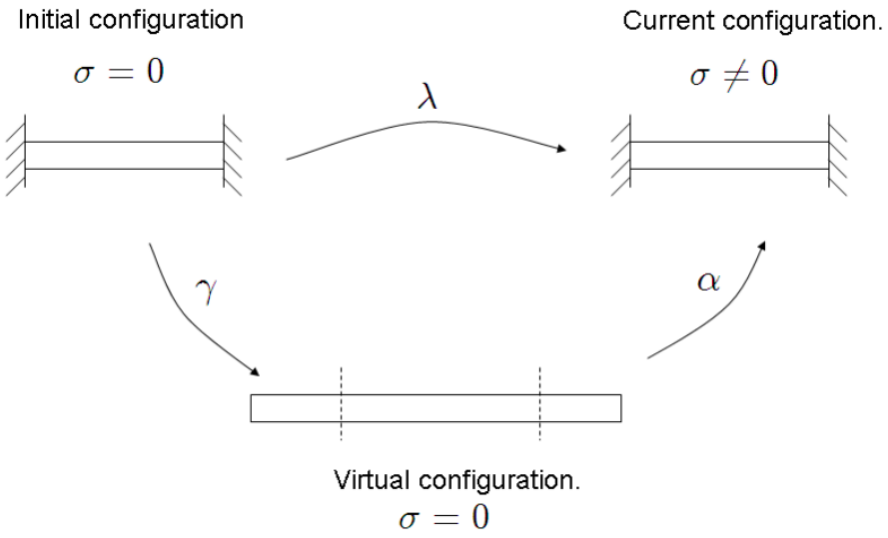


Figure 4.3.1: The initial configuration is shown to be stress free. In order to be at the zero stress state, the rod 'wants' to extend over the physical boundaries imposed on the system, which is impossible. In this figure λ represent the the overall deformation gradient (which we will define as A), γ is the Lagrangian zero stress deformation gradient, which represents the deformation from the original state to the zero stress state and α is the Eulerian zero stress deformation gradient, which describes the deformation from the zero stress state to the current configuration. Figure taken from [17].

Furthermore, in [15] it is stated that the material derivative of γ should be of the form

$$\frac{D\gamma}{Dt} = Ag(x,t), \tag{4.3.2}$$

where $g(x,t)$ measures the rate of change of the zero stress state. We now need to relate the stresses in the body to the elastic stretch α. For this purpose, we write

$$\sigma = E(1-\alpha^{-1}).$$

This expression is the analogous of Hooke's law for linear elastic materials in one dimension using the Eulerian measure of strain $\varepsilon = 1-\alpha^{-1}$. This effective strain is the link between the virtual zero stress state and the current configuration. Main reason for this choice is the implication this expression will have on the derivation of the strain rate equations.

We now substitute expression (4.3.1) into (4.3.2) and obtain

$$A\frac{D\alpha^{-1}}{Dt} + \alpha^{-1}\frac{DA}{Dt} = Ag \implies \frac{D\alpha^{-1}}{Dt} + \alpha^{-1}A^{-1}\frac{DA}{Dt} = g.$$

We note that $A^{-1}DA/Dt = \partial v/\partial x$.

$$\begin{aligned} A^{-1}\frac{DA}{Dt} &= \frac{D}{Dt}\left(\frac{\partial x}{\partial X}\right)\frac{\partial X}{\partial x}, \\ &= \frac{\partial}{\partial X}\left(\frac{Dx}{Dt}\right)\frac{\partial X}{\partial x}, \\ &= \frac{\partial v}{\partial X}\frac{\partial X}{\partial x}, \\ &= \frac{\partial v}{\partial x}. \end{aligned}$$

Given that $A^{-1}DA/Dt = \partial v/\partial x$ and considering the measure of strain $\varepsilon = 1-\alpha^{-1}$, we arrive at

$$\frac{D(1-\varepsilon)}{Dt} + (1-\varepsilon)\frac{\partial v}{\partial x} = g \implies \frac{D\varepsilon}{Dt} + (\varepsilon-1)\frac{\partial v}{\partial x} = -g.$$

Thus, we have an equation describing the rate of change of the effective strain. Coupled with the well known Cauchy momentum equation and the viscoelastic constitutive law, we have a system of equations which fully describe the behaviour of a morphoelastic material.

$$\begin{cases} \sigma = \mu \dfrac{\partial v}{\partial x} + E\varepsilon, \\ \dfrac{D(\rho v)}{Dt} + \rho v \dfrac{\partial v}{\partial x} = \dfrac{\partial \sigma}{\partial x} + f \quad \text{in } \Omega_t \times [0,T], \\ \dfrac{D\varepsilon}{Dt} + (\varepsilon - 1)\dfrac{\partial v}{\partial x} = -g \quad \text{in } \Omega_t \times [0,T], \\ v = \dfrac{Du}{Dt}, \end{cases}$$

where v is the velocity of the material and ε the effective Eulerian strain. The corresponding extension to three dimensions reads:

$$\begin{cases} \overline{\overline{\boldsymbol{\sigma}}} = \mu_1 \operatorname{sym}(\nabla \boldsymbol{v}) + \mu_2 (\nabla \cdot \boldsymbol{v}) \overline{\overline{\boldsymbol{I}}} + \dfrac{E\sqrt{\rho}}{1+\nu}\left(\overline{\overline{\boldsymbol{\varepsilon}}} + \dfrac{\nu}{1-2\nu}\operatorname{Tr}(\overline{\overline{\boldsymbol{\varepsilon}}})\overline{\overline{\boldsymbol{I}}}\right), \\ \dfrac{D(\rho \boldsymbol{v})}{Dt} + \rho \boldsymbol{v} (\nabla \cdot \boldsymbol{v}) = \nabla \cdot \overline{\overline{\boldsymbol{\sigma}}} + \boldsymbol{f} \quad \text{in } \Omega_t \times [0,T], \\ \dfrac{D\overline{\overline{\boldsymbol{\varepsilon}}}}{Dt} + \overline{\overline{\boldsymbol{\varepsilon}}} \operatorname{skw}(\nabla \boldsymbol{v}) - \operatorname{skw}(\nabla \boldsymbol{v}) \overline{\overline{\boldsymbol{\varepsilon}}} + \left(\operatorname{Tr}(\overline{\overline{\boldsymbol{\varepsilon}}}) - 1\right) \operatorname{sym}(\nabla \boldsymbol{v}) = -\overline{\overline{\boldsymbol{g}}} \quad \text{in } \Omega_t \times [0,T], \\ \dfrac{D\boldsymbol{u}}{Dt} = \boldsymbol{v}. \end{cases}$$

4.4. EXTENDED BIOCHEMICAL MODEL

To complete the model of burn contraction, we need to describe the biochemical process of wound healing. Our framework is based on the equations presented in [6] and we will follow its presentation of the model equations [6, Ch. 4].

In this regard, we consider four constituents to be the main contributors to the healing process: the fibroblasts N, the myofibroblasts M, a generic signalling molecule c and the density of collagen molecules ρ. We will investigate the concentrations z_i, $i \in \{N, M, c, \rho\}$ of these constituents in the dermal layer. The general equation describing the rate of change of our unknowns will be the following mass conservation equation as described in [6]:

$$\frac{Dz_i}{Dt} + z_i\,(\nabla \cdot \boldsymbol{v}) = -\nabla \cdot \boldsymbol{J}_i + R_i,$$

where $\boldsymbol{J}_i$ stands for the fluxes associated with the constituent i and R_i accounts for the chemical kinetics related to i. By abuse of notation we will denote the constituent and its concentration by the same symbol.

Fibroblasts and myofibroblasts move randomly across the considered domain. To capture this, a cell-density based Fickian diffusion model is used. Furthermore, the signalling molecule should direct the myo- and fibroblasts toward the area of interest, i.e. the wound. This directed movement is being modelled by chemotaxis as myo- and fibroblasts move up the gradient of the signalling molecule.

$$\boldsymbol{J}_N = -D_F F \nabla N + \chi_F N \nabla c,$$
$$\boldsymbol{J}_M = -D_F F \nabla M + \chi_F M \nabla c,$$

where $F = N + M$, D_F is the diffusion coefficient, which accounts for the random walk of cells, and χ_F is the chemotactic parameter. For instance, the signaling molecule transforming growth factor-β (TGF-β) has a strong attracting stimulus for (myo)fibroblasts. The proliferation of N and M has been modelled via logistic growth models. The main difference is that myofibroblasts only divide in presence of the signalling molecule, which is not the case for fibroblasts. Cell differentiation from fibroblasts to myofibroblasts under influence of the signalling molecule has also been taken into account. A simple sink term has been added to represent apoptosis (cell death). The combination of these factors results in:

$$R_N = r_F \left[1 + \frac{r_F^{\max} c}{a_c^I + c}\right] \left[1 - \kappa_F F\right] N^{1+q} - k_F c N - \delta_N N,$$
$$R_M = r_F \left\{ \frac{\left[1 + r_F^{\max}\right] c}{a_c^I + c} \right\} \left[1 - \kappa_F F\right] M^{1+q} + k_F c N - \delta_M M.$$

Here, r_F is the proliferation rate, $r_F^{\max}$ is the maximum factor with which the proliferation rate can be enhanced due to the presence of the signalling molecule, a_c^I is the concentration of the signalling molecule that causes the half-maximum enhancement of the proliferation rate, $\kappa_F F$ represents the reduction in the proliferation rate due to crowding, q is a fixed constant, k_F is the signalling molecule-dependent cell differentiation rate of fibroblasts into myofibroblasts, δ_N is the apoptosis rate of fibroblasts and δ_M is the apoptosis rate of myofibroblasts [6].

The signalling molecule c is assumed to move according to linear Fickian diffusion. Additionally, (myo)fibroblasts consume and secrete signalling molecules and proteolytic breakdown of c occurs in the dermal layer. The combination of these interactions results in the following expressions for $\boldsymbol{J}_c$ and R_c:

$$\boldsymbol{J}_c = -D_c \nabla c,$$
$$R_c = k_c \left[\frac{c}{a_c^{II} + c} \right] [N + \eta^I M] - \delta_c g(N, M, c, \rho) c,$$

where $g(N,M,c,\rho) \propto \dfrac{[N+\eta^{II}M]\,\rho}{1+a_c^{III}c}$. As described in [6], the parameter D_c represents the Fickian diffusion coefficient of the generic signalling molecule, k_c represents the maximum net secretion rate, η_I is the ratio of myofibroblasts to fibroblasts in the maximum net secretion rate of the signalling molecule, a_c^{II} is the concentration of the signalling molecule that causes the half-maximum net secretion rate and δ_c is the proteolytic breakdown rate of the signalling molecule. The concentration of the generic metalloproteinase (MMP) (i.e. $g(N,M,c,\rho)$), which is responsible for the breakdown of the signalling molecule, was assumed to be proportional to the cell density of the (myo)fibroblasts, and the concentration of both the collagen and the signalling molecule according to the following formula:

$$g(N, M, c, \rho) \propto \frac{[N + \eta^{II} M]\,\rho}{1 + a_c^{III} c}.$$

The parameter η^{II} is the ratio of myofibroblasts to fibroblasts in the secretion rate of the MMPs, and $1/[1 + a_c^{III}c]$ represents the inhibition of the secretion of the generic MMP due to the presence of the signaling molecule.

Lastly, we describe the equations for the density of collagen molecules ρ. Since collagen molecules are supposed to attach immediately to the extracellular matrix, no active transport takes place. At the same time, collagen is produced by both myofibroblasts and fibroblasts. The production is further enhanced by the presence of the signalling molecule. A proteolytic breakdown similar to the one presented for c is incorporated in the equations:

$$\boldsymbol{J}_\rho = \boldsymbol{0},$$
$$R_\rho = k_\rho \left\{ 1 + \left[\frac{k_\rho^{\max} c}{a_c^{IV} + c} \right] \right\} [N + \eta^I M] - \delta_\rho g(N, M, c, \rho) \rho.$$

Here, k_ρ is the collagen molecule secretion rate, $k_\rho^{\max}$ is the maximum factor with which this secretion rate can be enhanced due to the presence of the signaling molecule, a_c^{IV} is the concentration of the signaling molecules that causes the half-maximum enhancement of the secretion rate and δ_ρ is the proteolytic breakdown rate of the molecules.

The constituents interact with the mechanical description of the dermal layer through the following forcing terms $\boldsymbol{f}$ and $\overline{\overline{\boldsymbol{g}}}$, for the velocities and strains respectively. The mechanical pulling forces $\boldsymbol{f}$ model an isotropic stress caused by the pulling of the myofibroblasts. This pulling stress is proportional to the concentration of myofibroblasts times a simple function of the collagen density:

$$\boldsymbol{f} = \nabla \cdot \boldsymbol{\psi}$$
$$\boldsymbol{\psi} = \xi M \frac{\rho}{R^2 + \rho^2} \overline{\overline{\boldsymbol{I}}}.$$

Here, the parameter ψ represents the total generated stress by the myofibroblast population, ξ is the generated stress per unit cell density and the inverse of the unit collagen molecule concentration, and R is a fixed constant [6]. The rate of change of the effective strain is proportional to the product of the amount of effective strain, the local concentration of the MMPs, the local concentration of the signaling molecule and the inverse of the local concentration of the collagen molecules:

$$\overline{\overline{\boldsymbol{g}}} = \zeta \left[\frac{g(N,M,c,\rho)\,c}{\rho} \right] \overline{\overline{\boldsymbol{\varepsilon}}} = \zeta \left\{ \frac{\left[N+\eta^{II}M\right]c}{1+a_c^{III}c} \right\} \overline{\overline{\boldsymbol{\varepsilon}}},$$

where ζ is the rate of morphoelastic change.

5

NUMERICAL METHODS

In this chapter we will reintroduce the equations for each model and derive their weak forms. A finite element discretization will be presented the considered models. For the biochemical model additional considerations will be made, as it represents the core of this work and also the most complicated model of the following.

5.1. 1D MORPHOELASTICITY

The governing equations for the one-dimensional morphoelastic model are [13]

$$\begin{cases} \sigma = \mu \dfrac{\partial v}{\partial x} + E\varepsilon, \\ \dfrac{D(\rho v)}{Dt} + \rho v \dfrac{\partial v}{\partial x} = \dfrac{\partial \sigma}{\partial x} + F_b \quad \text{in } \Omega_t \times [0,T], \\ \dfrac{D\varepsilon}{Dt} + (\varepsilon - 1)\dfrac{\partial v}{\partial x} = -g \quad \text{in } \Omega_t \times [0,T], \\ v = \dfrac{Du}{Dt}. \end{cases} \tag{5.1.1}$$

The variables in (5.1.1) represent respectively σ the stress, μ dynamic viscosity, E Young's modulus, ε strain, ρ density, v velocity, F_b body forces, g forcing terms and u the displacement. The boundary conditions for (5.1.1) on the domain $\Omega_t = [0, L(t)]$ read

$$\begin{cases} v(0) = 0, \\ \sigma(L(t)) = 0 \quad \forall t, \end{cases}$$

i.e. fixed left edge and free right edge and we will use $g = \zeta\varepsilon$.

To derive the weak form of the momentum equation in (5.1.1) we multiply by a set of test functions $\{\varphi(x,t) \in H^1(\Omega_t)\,\forall t\}$ and integrate over the domain. The density ρ is considered to be constant. In the one-dimensional case we have $d\Omega = dx$.

$$\int_{\Omega_t} \left(\frac{D(\rho v)}{Dt} + \rho v \frac{\partial v}{\partial x} - \frac{\partial \sigma}{\partial x} \right) \varphi \, d\Omega = \int_{\Omega_t} F_b \varphi \, d\Omega$$

We include φ in the material derivative, with the required algebraic modifications, and expand it.

$$\int_{\Omega_t} \rho \frac{\partial (v\varphi)}{\partial t} + \rho v \frac{\partial (v\varphi)}{\partial x} + \rho v \varphi \frac{\partial v}{\partial x} - \frac{\partial \sigma}{\partial x}\varphi - \rho v \frac{D\varphi}{Dt} \, d\Omega = \int_{\Omega_t} F_b \varphi \, d\Omega$$

The second and third term can be grouped together under the derivation and integrated directly. Additionally, we apply Reynold's Transport Theorem on the first integration term:

$$\frac{d}{dt}\int_{\Omega_t} \rho v \varphi \, d\Omega - [\rho v^2 \varphi]_0^{L(t)} + [\rho v^2 \varphi]_0^{L(t)} + \int_{\Omega_t} -\frac{\partial(\sigma\varphi)}{\partial x} + \sigma \frac{\partial \varphi}{\partial x} - \rho v \frac{D\varphi}{Dt} \, d\Omega = \int_{\Omega_t} F_b \varphi \, d\Omega.$$

The weak form of the momentum equation in (5.1.1) reads:

$$\frac{d}{dt}\int_{\Omega_t} \rho v \varphi \, d\Omega = \int_{\Omega_t} -\sigma \frac{\partial \varphi}{\partial x} + F_b \varphi + \rho v \frac{D\varphi}{Dt} \, d\Omega + [\sigma\varphi]_0^{L(t)}$$

We will look now at the weak form of the strain equation. Again we multiply by the set of test functions $\{\varphi(x,t) \in L^2(\Omega_t)\,\forall t\}$ and integrate over the domain.

$$\int_{\Omega_t} \left(\frac{D\varepsilon}{Dt} + (\varepsilon - 1)\frac{\partial v}{\partial x}\right)\varphi \, d\Omega = -\int_{\Omega_t} g\varphi \, d\Omega$$

By performing similar steps as for the momentum equation we derive the weak form

$$\frac{d}{dt}\int_{\Omega_t} \varepsilon\varphi \, d\Omega - \int_{\Omega_t} \frac{\partial v}{\partial x}\varphi + \varepsilon\frac{D\varphi}{Dt} \, d\Omega = -\int_{\Omega_t} g\varphi \, d\Omega.$$

The next step is the discretization of the weak form. After dividing the domain in $N-1$ intervals $[0, L(t)] = \cup_{i=1}^{N-1}[x_i(t), x_{i+1}(t)]$, we consider the set of basis functions $\varphi_j(x,t)$ on Ω_t that approximate our model variables, i.e.

$$v(x,t) \approx v^h(x,t) := \sum_{j=1}^{N} v_j(t)\varphi_j(x,t), \quad \varepsilon(x,t) \approx \varepsilon^h(x,t) := \sum_{j=1}^{N} \varepsilon_j(t)\varphi_j(x,t). \tag{5.1.2}$$

Note that $x_i < x_{i+1}, \forall i$ and $\varphi_j(x_i,t) = \delta_{ij}, \forall t$. The chosen basis functions are piecewise linear polynomials. This condition implies that $D\varphi_j/Dt = 0\,\forall j$ on each interval [18].

The spatially discretized Galerkin equations read: Find $v^h(x,t) \in H^1(\Omega_t)\,\forall t$, $\varepsilon^h(x,t) \in L^2(\Omega_t)\,\forall t$, that satisfy

$$\frac{d}{dt}\int_{\Omega_t} \rho v^h \varphi_i \, d\Omega = \int_{\Omega_t} -\left(\mu\frac{\partial v^h}{\partial x} + E\varepsilon^h\right)\frac{\partial \varphi_i}{\partial x} + F_b\varphi_i \, d\Omega + [\sigma\varphi_i]_0^{L(t)}$$

$$\frac{d}{dt}\int_{\Omega_t} \varepsilon^h \varphi_i \, d\Omega = \int_{\Omega_t} \frac{\partial v^h}{\partial x}\varphi_i \, d\Omega - \int_{\Omega_t} \zeta\varepsilon^h\varphi_i \, d\Omega$$

for all basis functions φ_i. Note that given the boundary conditions in (5.1.1), the term $[\sigma\varphi_j]_0^{L(t)}$ is equal to zero.

A first order Euler backward method is used to discretize in time. Let $0 = t^1 < t^2 < \cdots < t^K = T$ be our equidistant meshing of the time interval, i.e. $t^{n+1} = t^n + \Delta t$ for $n = 1, \dots, K-1$, the superscript n will indicate the evaluation of the respective variable at time t^n, e.g. $x^n = x(t^n)$. Thus, we are looking for the vectors $\boldsymbol{v}^{n+1} := (v_j(t^{n+1}))_{j=1}^N$, $\boldsymbol{\varepsilon}^{n+1} := (\varepsilon_j(t^{n+1}))_{j=1}^N$ that are solution to the following linear system:

$$\rho M^{n+1}\boldsymbol{v}^{n+1} - \rho M^n \boldsymbol{v}^n = -\Delta t\left(\mu S^{n+1}\boldsymbol{v}^{n+1} + EC^{n+1}\boldsymbol{\varepsilon}^{n+1}\right) + \Delta t\boldsymbol{f}^{n+1} \tag{5.1.3}$$

$$M^{n+1}\boldsymbol{\varepsilon}^{n+1} - M^n\boldsymbol{\varepsilon}^n = \Delta t\left(C^T\right)^{n+1}\boldsymbol{v}^{n+1} - \Delta t\zeta M^{n+1}\boldsymbol{\varepsilon}^{n+1}. \tag{5.1.4}$$

The matrices M^n, S^n, C^n are assembled in a traditional finite element manner using the element matrices $M_{e,t}$, $S_{e,t}$, $C_{e,t}$ and $\boldsymbol{f}^n$ using the element vector $\boldsymbol{f}_{e,t}$, where

$$\begin{aligned}
M_{e,t} &= \int_{e,t} \varphi_i\varphi_j \, d\Omega \quad i,j = 1,2,\\
S_{e,t} &= \int_{e,t} \frac{\partial\varphi_i}{\partial x}\frac{\partial\varphi_j}{\partial x} \, d\Omega \quad i,j = 1,2,\\
C_{e,t} &= \int_{e,t} \frac{\partial\varphi_i}{\partial x}\varphi_j \, d\Omega \quad i,j = 1,2,\\
\boldsymbol{f}_{e,t} &= \int_{e,t} F_b\varphi_i \, d\Omega \quad i = 1,2,
\end{aligned}$$

and e,t stands for a general element e of the mesh at time t. The integrals can readily be computed using exact integration via the Holand-Bell's formulas [11]. Note that, since the mesh is moving, the matrices have to be re-calculated at each time step as the basis functions have the property: $\varphi_i(x,t) \neq \varphi_i(x, t+\Delta t)$.

The momentum equations for the visco- and morphoelastic cases are identical. However, the strain rate equations are much simpler in the former model. As a consequence, we decided to approach first viscoelasticity as an intermediate step to reach morphoelasticity.

Before we start, it is necessary to clarify some assumptions made throughout our work. In order to adapt the three-dimensional equations to a two-dimensional domain, it was assumed that $w = 0$, $\partial u/\partial z = 0$, $\partial v/\partial z = 0$, $v_3 = 0$, $\partial v_1/\partial z = 0$, $\partial v_2/\partial z = 0$, $\varepsilon_{13} = \varepsilon_{23} = \varepsilon_{31} = \varepsilon_{32} = \varepsilon_{33} = 0$ and $\partial\varepsilon_{11}/\partial z = \partial\varepsilon_{22}/\partial z = 0$ for all $t \geq 0$, where $\boldsymbol{u} = (u, v, w)$ is the displacement, $\boldsymbol{v} = (v_1, v_2, v_3)$ is the velocity and

$$\overline{\overline{\boldsymbol{\varepsilon}}} = \begin{bmatrix} \varepsilon_{11} & \varepsilon_{12} & \varepsilon_{13} \\ \varepsilon_{21} & \varepsilon_{22} & \varepsilon_{23} \\ \varepsilon_{31} & \varepsilon_{32} & \varepsilon_{33} \end{bmatrix}$$

is the strain tensor. Let the xy-plane be the plane parallel to the tissue surface, then we can reduce our calculations to any infinitely thin slice of tissue running parallel to the surface [6].

5.2. 2D VISCOELASTICITY

The governing equations for the two-dimensional viscoelastic model read [13]:

$$\begin{cases} \overline{\overline{\boldsymbol{\sigma}}} = \mu_1 \operatorname{sym}(\nabla \boldsymbol{v}) + \mu_2(\nabla\cdot\boldsymbol{v})\overline{\overline{\boldsymbol{I}}} + \dfrac{E\sqrt{\rho}}{1+\nu}\left(\overline{\overline{\boldsymbol{\varepsilon}}} + \dfrac{\nu}{1-2\nu}\operatorname{Tr}(\overline{\overline{\boldsymbol{\varepsilon}}})\overline{\overline{\boldsymbol{I}}}\right), \\ \dfrac{D(\rho\boldsymbol{v})}{Dt} + \rho\boldsymbol{v}(\nabla\cdot\boldsymbol{v}) = \nabla\cdot\overline{\overline{\boldsymbol{\sigma}}} + \boldsymbol{f} \quad \text{in } \Omega_t \times [0,T], \\ \overline{\overline{\boldsymbol{\varepsilon}}} = \dfrac{1}{2}\left(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T\right) \quad \text{in } \Omega_t \times [0,T], \\ \dfrac{D\boldsymbol{u}}{Dt} = \boldsymbol{v}. \end{cases} \tag{5.2.1}$$

Here, $\overline{\overline{\boldsymbol{\sigma}}}$ is the stress tensor, $\operatorname{sym}(A)$ is the symmetric part of a matrix A, $\boldsymbol{v}$ is the velocity of the tissue, μ_1 is the shear viscosity, μ_2 is the bulk viscosity, ν is Poisson's ratio, $\overline{\overline{\boldsymbol{\varepsilon}}}$ is the strain tensor, $E\sqrt{\rho}$ is the Young's modulus, $\overline{\overline{\boldsymbol{I}}}$ is the identity tensor, ρ is the tissue density, $\boldsymbol{f}$ represents the body forces and $\boldsymbol{u}$ is the displacement. Note that the strain tensor is symmetric, i.e. $\varepsilon_{12} = \varepsilon_{21}$. We will work with homogeneous Dirichlet boundary conditions on $\partial\Omega_t$ for the velocity. The initial conditions read $\boldsymbol{v} = \boldsymbol{0}$, $\overline{\overline{\boldsymbol{\varepsilon}}} = \boldsymbol{0}$ on Ω_t for $t = 0$.

We now derive the weak formulation for the component v_1 by multiplying the first component of the momentum equation by a sufficiently smooth test function $\varphi \in H^1(\Omega_t)\,\forall t$ and integrating over the domain Ω_t.

$$\int_{\Omega_t}\left(\frac{D(\rho v_1)}{Dt} + \rho v_1(\nabla\cdot\boldsymbol{v}) - \nabla\cdot\boldsymbol{\sigma}_{.1}\right)\varphi\, d\Omega = \int_{\Omega_t} f_1\varphi\, d\Omega$$

With the needed algebraic modifications we can apply Gauss' Theorem on the third integrand. Additionally, we include the test function in the material derivative. Thus, we get

$$\int_{\Omega_t}\frac{D(\rho v_1\varphi)}{Dt} + \rho v_1(\nabla\cdot\boldsymbol{v})\varphi + \boldsymbol{\sigma}_{.1}\cdot\nabla\varphi - \rho v_1\frac{D\varphi}{Dt}\, d\Omega - \int_{\partial\Omega_t}(\boldsymbol{\sigma}_{.1}\varphi)\cdot\boldsymbol{n}\, d\Gamma = \int_{\Omega_t} f_1\varphi\, d\Omega$$

Using Reynold's Transport Theorem we derive the weak formulation:

$$\frac{d}{dt}\int_{\Omega_t}\rho v_1\varphi\, d\Omega = -\int_{\Omega_t}\boldsymbol{\sigma}_{.1}\cdot\nabla\varphi - \rho v_1\frac{D\varphi}{Dt}\, d\Omega + \int_{\partial\Omega_t}(\boldsymbol{\sigma}_{.1}\varphi)\cdot\boldsymbol{n}\, d\Gamma + \int_{\Omega_t} f_1\varphi\, d\Omega,$$

where $\boldsymbol{n}$ is the outward pointing vector on the boundary $\partial\Omega_t$. The weak form for v_2 is derived following the same steps, i.e. it is only needed to modify the indices.

The weak form of the strains does not necessitate any transformation and thus simply is:

$$\int_{\Omega_t}\varepsilon_{11}\varphi\, d\Omega = \int_{\Omega_t}\frac{\partial u_1}{\partial x}\varphi\, d\Omega,$$

$$\int_{\Omega_t}\varepsilon_{22}\varphi\, d\Omega = \int_{\Omega_t}\frac{\partial u_2}{\partial y}\varphi\, d\Omega,$$

$$\int_{\Omega_t}\varepsilon_{12}\varphi\, d\Omega = \frac{1}{2}\int_{\Omega_t}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right)\varphi\, d\Omega,$$

where $\varphi \in L^2(\Omega_t) \forall t$.

To discretize the weak form, we consider the quadrilateral mesh $\mathcal{Q}_{h,t}$ approximating our domain Ω_t. Each edge of the quadrilaterals is a straight line and we consider the set of piecewise bilinear basis functions $\varphi_i(\boldsymbol{x}, t)$ on the mesh with the property $\varphi_i(\boldsymbol{x}_j, t) = \delta_{ij} \, \forall t$, where $\boldsymbol{x}_j$ are the nodes of our mesh. Again we discretize our model variables as a linear combination of basis functions similarly to (5.1.2). Additionally, we explicitly write out the components of $\overline{\overline{\boldsymbol{\sigma}}}$:

$$
\begin{aligned}
\sigma_{11} &= \mu_1 \frac{\partial v_1}{\partial x} + \mu_2 \left(\frac{\partial v_1}{\partial x} + \frac{\partial v_2}{\partial y} \right) + \frac{E\sqrt{\rho}}{1+\nu} \left(\varepsilon_{11} + \frac{\nu}{1-2\nu} (\varepsilon_{11} + \varepsilon_{22}) \right), \\
\sigma_{22} &= \mu_1 \frac{\partial v_2}{\partial y} + \mu_2 \left(\frac{\partial v_1}{\partial x} + \frac{\partial v_2}{\partial y} \right) + \frac{E\sqrt{\rho}}{1+\nu} \left(\varepsilon_{22} + \frac{\nu}{1-2\nu} (\varepsilon_{11} + \varepsilon_{22}) \right), \\
\sigma_{12} &= \frac{1}{2} \mu_1 \left(\frac{\partial v_1}{\partial y} + \frac{\partial v_2}{\partial x} \right) + \frac{E\sqrt{\rho}}{1+\nu} \varepsilon_{12} = \sigma_{21}.
\end{aligned}
$$

Thus, the spatially discretized Galerkin equations read: Find $v_i^h(\boldsymbol{x}, t) \in H^1(\Omega_t) \, \forall t, \varepsilon_{jk}^h(\boldsymbol{x}, t) \in L^2(\Omega_t) \, \forall t$ that satisfy for all basis functions φ_i

$$
\begin{aligned}
\frac{d}{dt} \int_{\Omega_t} \rho v_1^h \varphi_i \, d\Omega = \\
&- \int_{\Omega_t} \left[\mu_1 \frac{\partial v_1^h}{\partial x} + \mu_2 \left(\frac{\partial v_1^h}{\partial x} + \frac{\partial v_2^h}{\partial y} \right) + \frac{E\sqrt{\rho}}{1+\nu} \left(\varepsilon_{11}^h + \frac{\nu}{1-2\nu} (\varepsilon_{11}^h + \varepsilon_{22}^h) \right) \right] \frac{\partial \varphi_i}{\partial x} \, d\Omega \\
&- \int_{\Omega_t} \left[\frac{1}{2} \mu_1 \left(\frac{\partial v_1^h}{\partial y} + \frac{\partial v_2^h}{\partial x} \right) + \frac{E\sqrt{\rho}}{1+\nu} \varepsilon_{12}^h \right] \frac{\partial \varphi_i}{\partial y} \, d\Omega \\
&- \int_{\Omega_t} \rho v_1^h \frac{D\varphi_i}{Dt} \, d\Omega \\
&+ \int_{\partial\Omega_t} n_1 \left[\mu_1 \frac{\partial v_1^h}{\partial x} + \mu_2 \left(\frac{\partial v_1^h}{\partial x} + \frac{\partial v_2^h}{\partial y} \right) + \frac{E\sqrt{\rho}}{1+\nu} \left(\varepsilon_{11}^h + \frac{\nu}{1-2\nu} (\varepsilon_{11}^h + \varepsilon_{22}^h) \right) \right] \varphi_i \, d\Gamma \\
&+ \int_{\partial\Omega_t} n_2 \left[\frac{1}{2} \mu_1 \left(\frac{\partial v_1^h}{\partial y} + \frac{\partial v_2^h}{\partial x} \right) + \frac{E\sqrt{\rho}}{1+\nu} \varepsilon_{12}^h \right] \varphi_i \, d\Gamma \\
&+ \int_{\Omega_t} f_1 \varphi_i \, d\Omega,
\end{aligned}
$$

$$
\begin{aligned}
\frac{d}{dt} \int_{\Omega_t} \rho v_2^h \varphi_i \, d\Omega = \\
&- \int_{\Omega_t} \left[\frac{1}{2} \mu_1 \left(\frac{\partial v_1^h}{\partial y} + \frac{\partial v_2^h}{\partial x} \right) + \frac{E\sqrt{\rho}}{1+\nu} \varepsilon_{12}^h \right] \frac{\partial \varphi_i}{\partial x} \, d\Omega \\
&- \int_{\Omega_t} \left[\mu_1 \frac{\partial v_2^h}{\partial y} + \mu_2 \left(\frac{\partial v_1^h}{\partial x} + \frac{\partial v_2^h}{\partial y} \right) + \frac{E\sqrt{\rho}}{1+\nu} \left(\varepsilon_{22}^h + \frac{\nu}{1-2\nu} (\varepsilon_{11}^h + \varepsilon_{22}^h) \right) \right] \frac{\partial \varphi_i}{\partial y} \, d\Omega \\
&- \int_{\Omega_t} \rho v_2^h \frac{D\varphi_i}{Dt} \, d\Omega \\
&+ \int_{\partial\Omega_t} n_1 \left[\frac{1}{2} \mu_1 \left(\frac{\partial v_1^h}{\partial y} + \frac{\partial v_2^h}{\partial x} \right) + \frac{E\sqrt{\rho}}{1+\nu} \varepsilon_{12}^h \right] \varphi_i \, d\Gamma \\
&+ \int_{\partial\Omega_t} n_2 \left[\mu_1 \frac{\partial v_2^h}{\partial y} + \mu_2 \left(\frac{\partial v_1^h}{\partial x} + \frac{\partial v_2^h}{\partial y} \right) + \frac{E\sqrt{\rho}}{1+\nu} \left(\varepsilon_{22}^h + \frac{\nu}{1-2\nu} (\varepsilon_{11}^h + \varepsilon_{22}^h) \right) \right] \varphi_i \, d\Gamma \\
&+ \int_{\Omega_t} f_2 \varphi_i \, d\Omega.
\end{aligned}
$$

As we have done for the one-dimensional case, also on the quadrilateral mesh we want to get rid of the integral with $D\varphi_i/Dt$. In [18] only the proof for simplices is given. We therefore need to derive an equivalent statement for quadrilaterals. Our goal is to eliminate the integral involving the Lagrangian derivative of the test function. Since the union of all elements is equal to our computational domain, it is sufficient to prove that on each element the integral is zero.

It can be shown that the material derivative of the test functions is indeed zero on each element. This is due to the fact that in the reference domain they do not depend on time and neither does the reference domain itself [19]. As a consequence, test functions can be ‘included’ in a material derivative without generating any additional terms. Thus, we eliminate the term with $D\varphi_i/Dt$.

Similarly to $v_i^h, \varepsilon_{jk}^h$, we approximate the displacement $\boldsymbol{u}$ with a linear combination of basis functions which we will denote as u_1^h, u_2^h. The Galerkin equations for the strain are: Find $\varepsilon_{jk}^h \in L^2(\Omega_t)\,\forall t$ that satisfy for all basis functions φ_i

$$\begin{aligned}
\int_{\Omega_t} \varepsilon_{11}^h \varphi_i \, d\Omega &= \int_{\Omega_t} \frac{\partial u_1^h}{\partial x} \varphi_i \, d\Omega, \\
\int_{\Omega_t} \varepsilon_{22}^h \varphi_i \, d\Omega &= \int_{\Omega_t} \frac{\partial u_2^h}{\partial y} \varphi_i \, d\Omega, \\
\int_{\Omega_t} \varepsilon_{12}^h \varphi_i \, d\Omega &= \int_{\Omega_t} \frac{1}{2}\left(\frac{\partial u_1^h}{\partial y} + \frac{\partial u_2^h}{\partial x}\right) \varphi_i \, d\Omega.
\end{aligned}$$

To discretize in time, Euler’s backward differentiation has been used. As mentioned in Chapter 3, the mesh is being updated supposing constant velocity at each time step. This implies that $u_i^h(\boldsymbol{x}, t+\Delta t) = u_i^h(\boldsymbol{x}, t) + \Delta t\, v_i^h(\boldsymbol{x}, t)$. The following linear system is being solved for all model variables simultaneously:

$$\begin{aligned}
\rho M^{n+1} \boldsymbol{v}_1^{n+1} = \rho M^n \boldsymbol{v}_1^n & \\
& + \Delta t \frac{E\sqrt{\rho}}{1+\nu} \frac{1-\nu}{1-2\nu} \left(\partial M_1^{n+1} - C_x^{n+1}\right) \boldsymbol{\varepsilon}_{11}^{n+1} \\
& + \Delta t \frac{E\sqrt{\rho}}{1+\nu} \frac{\nu}{1-2\nu} \left(\partial M_1^{n+1} - C_x^{n+1}\right) \boldsymbol{\varepsilon}_{22}^{n+1} \\
& + \Delta t \frac{E\sqrt{\rho}}{1+\nu} \left(\partial M_2^{n+1} - C_y^{n+1}\right) \boldsymbol{\varepsilon}_{12}^{n+1} \\
& + \Delta t \left[(\mu_1+\mu_2)\left(\partial \left(C_{x1}^T\right)^{n+1} - S_{xx}^{n+1}\right) + \frac{\mu_1}{2}\left(\partial \left(C_{y2}^T\right)^{n+1} - S_{yy}^{n+1}\right)\right] \boldsymbol{v}_1^{n+1} \\
& + \Delta t \left[\mu_2 \left(\partial \left(C_{y1}^T\right)^{n+1} - S_{xy}^{n+1}\right) + \frac{\mu_1}{2}\left(\partial \left(C_{x2}^T\right)^{n+1} - S_{yx}^{n+1}\right)\right] \boldsymbol{v}_2^{n+1} \\
& + \Delta t \boldsymbol{f}_1^{n+1}
\end{aligned}$$

$$\begin{aligned}
\rho M^{n+1} \boldsymbol{v}_2^{n+1} = \rho M^n \boldsymbol{v}_2^n & \\
& + \Delta t \frac{E\sqrt{\rho}}{1+\nu} \frac{\nu}{1-2\nu} \left(\partial M_2^{n+1} - C_y^{n+1}\right) \boldsymbol{\varepsilon}_{11}^{n+1} \\
& + \Delta t \frac{E\sqrt{\rho}}{1+\nu} \frac{1-\nu}{1-2\nu} \left(\partial M_2^{n+1} - C_y^{n+1}\right) \boldsymbol{\varepsilon}_{22}^{n+1} \\
& + \Delta t \frac{E\sqrt{\rho}}{1+\nu} \left(\partial M_1^{n+1} - C_x^{n+1}\right) \boldsymbol{\varepsilon}_{12}^{n+1} \\
& + \Delta t \left[\frac{\mu_1}{2}\left(\partial \left(C_{y1}^T\right)^{n+1} - S_{xy}^{n+1}\right) + \mu_2 \left(\partial \left(C_{x2}^T\right)^{n+1} - S_{yx}^{n+1}\right)\right] \boldsymbol{v}_1^{n+1} \\
& + \Delta t \left[(\mu_1+\mu_2)\left(\partial \left(C_{y2}^T\right)^{n+1} - S_{yy}^{n+1}\right) + \frac{\mu_1}{2}\left(\partial \left(C_{x1}^T\right)^{n+1} - S_{xx}^{n+1}\right)\right] \boldsymbol{v}_2^{n+1} \\
& + \Delta t \boldsymbol{f}_2^{n+1}
\end{aligned}$$

$$\begin{aligned}
M^{n+1} \boldsymbol{\varepsilon}_{11}^{n+1} &= \left(C_x^T\right)^{n+1} \boldsymbol{u}_1^{n+1} \\
M^{n+1} \boldsymbol{\varepsilon}_{22}^{n+1} &= \left(C_y^T\right)^{n+1} \boldsymbol{u}_2^{n+1} \\
M^{n+1} \boldsymbol{\varepsilon}_{12}^{n+1} &= \frac{1}{2}\left(\left(C_y^T\right)^{n+1} \boldsymbol{u}_1^{n+1} + \left(C_x^T\right)^{n+1} \boldsymbol{u}_2^{n+1}\right)
\end{aligned}$$

In our notation $\partial\cdot$ represents a matrix corresponding to a boundary contribution, i.e. an integration over $\partial\Omega_t$ and M stands for the standard mass matrix. The remaining matrices are build with the following element matrices:

$$
\begin{aligned}
C_{x_{e,t}} &= \int_{e,t} \frac{\partial \varphi_i}{\partial x} \varphi_j \, d\Omega \quad i,j = 1,\dots,4 \\
S_{xy_{e,t}} &= \int_{e,t} \frac{\partial \varphi_i}{\partial x} \frac{\partial \varphi_j}{\partial y} \, d\Omega \quad i,j = 1,\dots,4.
\end{aligned}
$$

If there are any numerical subscripts, this indicates that the integrand is multiplied by the respective component of the outer normal vector along the boundary. The transposed symbol T, e.g. in C_x^T, represents that the differentiation is applied on φ_j, opposed to being applied on φ_i. The superscripts n represent again the time t^n at which the domain of integration is considered, where $t^{n+1} = t^n + \Delta t$. Integrations have been performed using the Gaussian product rule with 2 × 2 integration points, as previously described in Chapter 3. We will give an example using the element matrix of C_x.

$$
\begin{aligned}
C_{x_{e,t}} &= \int_{e,t} \frac{\partial \varphi_i}{\partial x} \varphi_j \, d\Omega \\
&= \int_{\hat{\Omega}_{el}} \left(J_{1\cdot}^{-T} \cdot \nabla_{(\xi,\eta)} \hat{\varphi}_i\right) \hat{\varphi}_j \, |\det J| \, d\hat{\Omega} \\
&\approx \sum_{k=1}^{2} \sum_{l=1}^{2} \omega_k \omega_l \left(J_{1\cdot}^{-T}(\xi_k,\eta_l) \cdot \nabla \hat{\varphi}_i(\xi_k,\eta_l)\right) \hat{\varphi}_j(\xi_k,\eta_l) |\det J(\xi_k,\eta_l)| \\
&\quad i,j = 1,\dots,4,
\end{aligned}
$$

where $\hat{\varphi}$ are basis functions on the reference element and J is the Jacobian of the isoparametric transformation ψ_t. For the integrals along the boundaries a two-point 1D Gaussian quadrature rule has been employed.

5.3. 2D MORPHOELASTICITY

In order to solve the two-dimensional morphoelastic model, we add strain evolution equations to (5.2.1). The complete model now reads:

$$
\begin{cases}
\bar{\bar{\boldsymbol{\sigma}}} = \mu_1 \,\mathrm{sym}(\nabla \boldsymbol{v}) + \mu_2 (\nabla \cdot \boldsymbol{v}) \bar{\bar{\boldsymbol{I}}} + \dfrac{E\sqrt{\rho}}{1+\nu}\left(\bar{\bar{\boldsymbol{\varepsilon}}} + \dfrac{\nu}{1-2\nu}\mathrm{Tr}(\bar{\bar{\boldsymbol{\varepsilon}}})\bar{\bar{\boldsymbol{I}}}\right), \\
\dfrac{D(\rho \boldsymbol{v})}{Dt} + \rho \boldsymbol{v}(\nabla \cdot \boldsymbol{v}) = \nabla \cdot \bar{\bar{\boldsymbol{\sigma}}} + \boldsymbol{f} \quad \text{in } \Omega_t \times [0,T], \\
\dfrac{D\bar{\bar{\boldsymbol{\varepsilon}}}}{Dt} + \bar{\bar{\boldsymbol{\varepsilon}}}\,\mathrm{skw}(\nabla \boldsymbol{v}) - \mathrm{skw}(\nabla \boldsymbol{v})\bar{\bar{\boldsymbol{\varepsilon}}} + \left(\mathrm{Tr}(\bar{\bar{\boldsymbol{\varepsilon}}}) - 1\right)\mathrm{sym}(\nabla \boldsymbol{v}) = -\bar{\bar{\boldsymbol{g}}} \quad \text{in } \Omega_t \times [0,T], \\
\dfrac{D\boldsymbol{u}}{Dt} = \boldsymbol{v}.
\end{cases}
\tag{5.3.1}
$$

Here, $\mathrm{skw}(A)$ represents the skew-symmetric part of a matrix A. We will work with homogeneous Dirichlet boundary conditions on $\partial\Omega_t$ for the velocity. The initial conditions read $\boldsymbol{v} = \boldsymbol{0}$, $\bar{\bar{\boldsymbol{\varepsilon}}} = \boldsymbol{0}$ on Ω_t for $t = 0$.

It can be proved that if $\bar{\bar{\boldsymbol{g}}}$ is symmetric for $t \geq 0$, then also $\bar{\bar{\boldsymbol{\varepsilon}}}$ is symmetric for $t \geq 0$ [13]. We will therefore consider $\varepsilon_{12} = \varepsilon_{21}$.

Lemma 1. *Let the evolution of the strain tensor be defined as in* (5.3.1)*. Let* $\bar{\bar{\boldsymbol{g}}}$ *be symmetric for* $t \geq 0$ *and* $\bar{\bar{\boldsymbol{\varepsilon}}}$ *symmetric at* $t = 0$*. Then* $\bar{\bar{\boldsymbol{\varepsilon}}}$ *is symmetric for* $t \geq 0$*. The same holds true for* $\bar{\bar{\boldsymbol{g}}} = \zeta\bar{\bar{\boldsymbol{\varepsilon}}}$*.*

Proof. Tranposing the strain evolution equation yields:

$$
\frac{D\bar{\bar{\boldsymbol{\varepsilon}}}^T}{Dt} + \mathrm{skw}(\nabla \boldsymbol{v})^T \bar{\bar{\boldsymbol{\varepsilon}}}^T - \bar{\bar{\boldsymbol{\varepsilon}}}^T \mathrm{skw}(\nabla \boldsymbol{v})^T + \left(\mathrm{Tr}(\bar{\bar{\boldsymbol{\varepsilon}}}) - 1\right)\mathrm{sym}(\nabla \boldsymbol{v})^T = -\bar{\bar{\boldsymbol{g}}}^T \tag{5.3.2}
$$

Per construction both $\mathrm{skw}(A)$ and $\mathrm{sym}(A)$ are symmetric. Hence we can drop the transposed symbol. Additionally, the tensor $\bar{\bar{\boldsymbol{g}}}$ is symmetric by assumption. If we subtract (5.3.2) from the strain evolution equation in (5.3.1) we get

$$
\frac{D\left(\bar{\bar{\boldsymbol{\varepsilon}}} - \bar{\bar{\boldsymbol{\varepsilon}}}^T\right)}{Dt} - \mathrm{skw}(\nabla \boldsymbol{v})\left(\bar{\bar{\boldsymbol{\varepsilon}}} - \bar{\bar{\boldsymbol{\varepsilon}}}^T\right) + \left(\bar{\bar{\boldsymbol{\varepsilon}}} - \bar{\bar{\boldsymbol{\varepsilon}}}^T\right)\mathrm{skw}(\nabla \boldsymbol{v}) = 0 \tag{5.3.3}
$$

Since we supposed the strain to be symmetric at time zero, the above equation has as solution $\overline{\overline{\boldsymbol{\varepsilon}}} - \overline{\overline{\boldsymbol{\varepsilon}}}^T \equiv 0$ for $t \geq 0$. In our model we will use $\overline{\overline{\boldsymbol{g}}} = \zeta \overline{\overline{\boldsymbol{\varepsilon}}}$. On the RHS of (5.3.3) we will get $\zeta\left(\overline{\overline{\boldsymbol{\varepsilon}}} - \overline{\overline{\boldsymbol{\varepsilon}}}^T\right)$. Since we suppose the strain to be symmetric at $t = 0$, we conclude that $\overline{\overline{\boldsymbol{\varepsilon}}} - \overline{\overline{\boldsymbol{\varepsilon}}}^T \equiv 0$ is the solution for $t \geq 0$. □

Before we derive the weak form for the strain equations, we modify them by adding $\varepsilon_{ij}(\nabla \cdot \boldsymbol{v})$ on the left-hand and right-hand side of the equation for the respective variable. Thus, we get in $\Omega_t \times [0, T]$:

$$\begin{cases} \dfrac{D\varepsilon_{11}}{Dt} + \varepsilon_{11}(\nabla \cdot \boldsymbol{v}) = (1-\varepsilon_{22})\dfrac{\partial v_1}{\partial x} + \varepsilon_{11}\dfrac{\partial v_2}{\partial y} - \varepsilon_{12}\left(\dfrac{\partial v_2}{\partial x} - \dfrac{\partial v_1}{\partial y}\right) - g_{11}, \\ \dfrac{D\varepsilon_{22}}{Dt} + \varepsilon_{22}(\nabla \cdot \boldsymbol{v}) = (1-\varepsilon_{11})\dfrac{\partial v_2}{\partial y} + \varepsilon_{22}\dfrac{\partial v_1}{\partial x} - \varepsilon_{12}\left(\dfrac{\partial v_1}{\partial y} - \dfrac{\partial v_2}{\partial x}\right) - g_{22}, \\ \dfrac{D\varepsilon_{12}}{Dt} + \varepsilon_{12}(\nabla \cdot \boldsymbol{v}) = \varepsilon_{12}(\nabla \cdot \boldsymbol{v}) + \dfrac{1}{2}\left(\dfrac{\partial v_1}{\partial y}(1-2\varepsilon_{11}) + \dfrac{\partial v_2}{\partial x}(1-2\varepsilon_{22})\right) - g_{12}. \end{cases} \tag{5.3.4}$$

The weak forms of (5.3.4) can be readily derived through multiplication by the test functions $\varphi \in L^2(\Omega_t)\ \forall t$ and integrating over the domain Ω_t. Using Reynold's Transport Theorem yields:

$$\frac{d}{dt}\int_{\Omega_t} \varepsilon_{11}\varphi\, d\Omega = \int_{\Omega_t} \left[(1-\varepsilon_{22})\frac{\partial v_1}{\partial x} + \varepsilon_{11}\frac{\partial v_2}{\partial y} - \varepsilon_{12}\left(\frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial y}\right) - g_{11}\right]\varphi + \varepsilon_{11}\frac{D\varphi}{Dt}\, d\Omega,$$

$$\frac{d}{dt}\int_{\Omega_t} \varepsilon_{22}\varphi\, d\Omega = \int_{\Omega_t} \left[(1-\varepsilon_{11})\frac{\partial v_2}{\partial y} + \varepsilon_{22}\frac{\partial v_1}{\partial x} - \varepsilon_{12}\left(\frac{\partial v_1}{\partial y} - \frac{\partial v_2}{\partial x}\right) - g_{22}\right]\varphi + \varepsilon_{12}\frac{D\varphi}{Dt}\, d\Omega,$$

$$\frac{d}{dt}\int_{\Omega_t} \varepsilon_{12}\varphi\, d\Omega = \int_{\Omega_t} \left[\varepsilon_{12}(\nabla \cdot \boldsymbol{v}) + \frac{1}{2}\left(\frac{\partial v_1}{\partial y}(1-2\varepsilon_{11}) + \frac{\partial v_2}{\partial x}(1-2\varepsilon_{22})\right)\right]\varphi - g_{12}\varphi + \varepsilon_{12}\frac{D\varphi}{Dt}\, d\Omega.$$

As we have done in Section 5.2, we discretize our domain and the model variables using the basis functions φ_j. The spatially discretized Galerkin equations read: Find $\varepsilon_{jk}^h \in L^2(\Omega_t)\ \forall t$ that satisfy for all basis functions φ_i

$$\frac{d}{dt}\int_{\Omega_t} \varepsilon_{11}^h\varphi_i\, d\Omega = \int_{\Omega_t} \left[(1-\varepsilon_{22}^h)\frac{\partial v_1^h}{\partial x} + \varepsilon_{11}^h\frac{\partial v_2^h}{\partial y} - \varepsilon_{12}^h\left(\frac{\partial v_2^h}{\partial x} - \frac{\partial v_1^h}{\partial y}\right) - g_{11}\right]\varphi_i\, d\Omega,$$

$$\frac{d}{dt}\int_{\Omega_t} \varepsilon_{22}^h\varphi_i\, d\Omega = \int_{\Omega_t} \left[(1-\varepsilon_{11}^h)\frac{\partial v_2^h}{\partial y} + \varepsilon_{22}^h\frac{\partial v_1^h}{\partial x} - \varepsilon_{12}^h\left(\frac{\partial v_1^h}{\partial y} - \frac{\partial v_2^h}{\partial x}\right) - g_{22}\right]\varphi_i\, d\Omega,$$

$$\frac{d}{dt}\int_{\Omega_t} \varepsilon_{12}^h\varphi_i\, d\Omega = \int_{\Omega_t} \left[\varepsilon_{12}^h(\nabla \cdot \boldsymbol{v}^h) + \frac{1}{2}\left(\frac{\partial v_1^h}{\partial y}(1-2\varepsilon_{11}^h) + \frac{\partial v_2^h}{\partial x}(1-2\varepsilon_{22}^h)\right) - g_{12}\right]\varphi_i\, d\Omega,$$

where v_i^h are the solutions to the Galerkin equations treated in Section 5.2. The Lagrangian derivative of the basis function is zero as explained previously. Again, Euler backward is used to discretize in time. The new linear system for strain, solved simultaneously with the one for velocity of Section 5.2, reads:

$$\begin{aligned} M^{n+1}\boldsymbol{\varepsilon}_{11}^{n+1} = M^n\boldsymbol{\varepsilon}_{11}^n &+ \Delta t\left(\left(C_x^T\right)^{n+1}\boldsymbol{v}_1^{n+1} - N_x(\boldsymbol{v}_1^*)^{n+1}\boldsymbol{\varepsilon}_{22}^{n+1}\right) \\ &+ \Delta t N_y(\boldsymbol{v}_2^*)^{n+1}\boldsymbol{\varepsilon}_{11}^{n+1} \\ &+ \Delta t\left(N_y(\boldsymbol{v}_1^*)^{n+1} - N_x(\boldsymbol{v}_2^*)^{n+1}\right)\boldsymbol{\varepsilon}_{12}^{n+1} \\ &- \Delta t\,\boldsymbol{g}_{11}^{n+1}, \end{aligned}$$

$$\begin{aligned} M^{n+1}\boldsymbol{\varepsilon}_{22}^{n+1} = M^n\boldsymbol{\varepsilon}_{22}^n &+ \Delta t\left(\left(C_y^T\right)^{n+1}\boldsymbol{v}_2^{n+1} - N_y(\boldsymbol{v}_2^*)^{n+1}\boldsymbol{\varepsilon}_{11}^{n+1}\right) \\ &+ \Delta t N_x(\boldsymbol{v}_1^*)^{n+1}\boldsymbol{\varepsilon}_{22}^{n+1} \\ &+ \Delta t\left(N_x(\boldsymbol{v}_2^*)^{n+1} - N_y(\boldsymbol{v}_1^*)^{n+1}\right)\boldsymbol{\varepsilon}_{12}^{n+1} \\ &- \Delta t\,\boldsymbol{g}_{22}^{n+1}, \end{aligned}$$

$$
\begin{aligned}
M^{n+1}\boldsymbol{\varepsilon}_{12}^{n+1} = M^{n}\boldsymbol{\varepsilon}_{12}^{n} &+ \Delta t\left(N_x(\boldsymbol{v}_1^*)^{n+1} + N_y(\boldsymbol{v}_2^*)^{n+1}\right)\boldsymbol{\varepsilon}_{12}^{n+1} \\
&+ \Delta t\left(\frac{1}{2}\left(C_y^T\right)^{n+1}\boldsymbol{v}_1^{n+1} + \frac{1}{2}\left(C_x^T\right)^{n+1}\boldsymbol{v}_2^{n+1}\right) \\
&- \Delta t\left(N_y(\boldsymbol{v}_1^*)^{n+1}\boldsymbol{\varepsilon}_{11}^{n+1} + N_x(\boldsymbol{v}_2^*)^{n+1}\boldsymbol{\varepsilon}_{22}^{n+1}\right) \\
&- \Delta t\,\boldsymbol{g}_{12}^{n+1}.
\end{aligned}
$$

Here, we have introduced the new matrix $N_\star(\boldsymbol{v}_k)$, which stands for the integral

$$
N_\star(\boldsymbol{v}_k)^n = \int_{\Omega_{t^n}} \varphi_i\varphi_j\frac{\partial v_k^h}{\partial\star}\,d\Omega.
$$

The term $N_\star(\boldsymbol{v}_k)\boldsymbol{\varepsilon}_{lm}$ clearly introduces a non-linearity in our system as both velocity and strain are being solved for. We choose to use the velocity of the previous time step to initiate a Picard iteration. This leads to the notation of $\boldsymbol{v}_k^*$, which represent the velocity solution at the previous Picard iteration. The integral is calculated on each element by representing the velocity as linear combination of the basis functions, i.e. on $\Omega_{t,el}$, $v_k^h = \sum_{i=1}^4(\boldsymbol{v}_k)_i\varphi_i$. We omitted the time indices for the sake of clarity. The implementation of the calculations for the element matrix reads:

$$
\begin{aligned}
N_x(\boldsymbol{v}_k)_{e,t} &= \int_{e,t}\varphi_i\varphi_j\frac{\partial v_k^h}{\partial x}\,d\Omega \\
&= \int_{e,t}\varphi_i\varphi_j\sum_{m=1}^{4}(\boldsymbol{v}_k)_m\frac{\partial\varphi_m}{\partial x}\,d\Omega \\
&= \int_{\hat{\Omega}_{el}}\hat{\varphi}_i\hat{\varphi}_j\sum_{m=1}^{4}(\boldsymbol{v}_k)_m(J_{1\cdot}^{-T}\cdot\nabla\hat{\varphi}_m)|\det J|\,d\hat{\Omega} \\
&\approx \sum_{s=1}^{2}\sum_{t=1}^{2}\omega_s\omega_t\hat{\varphi}_i(\xi_s,\eta_t)\hat{\varphi}_j(\xi_s,\eta_t)|\det J(\xi_s,\eta_t)|S(\xi_s,\eta_t), \\
&\quad i,j = 1,\dots,4,
\end{aligned}
$$

where $S(\xi_s,\eta_t) = \sum_{m=1}^4(\boldsymbol{v}_k)_m(J_{1\cdot}^{-T}(\xi_s,\eta_t)\cdot\nabla\hat{\varphi}_m(\xi_s,\eta_t))$.

5.4. EXTENDED BIOCHEMICAL MODEL

The complete biochemical-mechanical model consists of four conservation equations describing the variation of concentration of the cell variables N, M, c, ρ, see Section 4.4, and two equations describing the mechanical response of the tissue similar to the ones introduced in Section 4.3.

$$
\begin{cases}
\dfrac{Dz_i}{Dt} + z_i\,(\nabla\cdot\boldsymbol{v}) = -\nabla\cdot\boldsymbol{J}_i + R_i \quad \text{in } \Omega_t\times[0,T], \\
\dfrac{D(\rho_t\boldsymbol{v})}{Dt} + \rho_t\boldsymbol{v}(\nabla\cdot\boldsymbol{v}) = \nabla\cdot\overline{\overline{\boldsymbol{\sigma}}} + \boldsymbol{f} \quad \text{in } \Omega_t\times[0,T], \\
\dfrac{D\overline{\overline{\boldsymbol{\varepsilon}}}}{Dt} + \overline{\overline{\boldsymbol{\varepsilon}}}\,\mathrm{skw}(\nabla\boldsymbol{v}) - \mathrm{skw}(\nabla\boldsymbol{v})\overline{\overline{\boldsymbol{\varepsilon}}} + \left(\mathrm{Tr}(\overline{\overline{\boldsymbol{\varepsilon}}}) - 1\right)\mathrm{sym}(\nabla\boldsymbol{v}) = -\overline{\overline{\boldsymbol{g}}} \quad \text{in } \Omega_t\times[0,T],
\end{cases}
\tag{5.4.1}
$$

where z_i is the concentration of each cell constituent $i \in \{N, M, c, \rho\}$. Furhter, $\boldsymbol{J}_i$ stands for the fluxes associated with the constituent i and R_i accounts for the chemical kinetics related to i [6]. Here ρ_t represents the total mass density of the dermal tissue, $\boldsymbol{f}$ the exerted body forces and $\overline{\overline{\boldsymbol{g}}}$ the tensor describing the rate of change of strain. Furthermore, the considered stress-strain law is analogous to the one presented in equation (5.2.1). The Young's modulus is assumed to be largely determined by the collagen density ρ, which therefore also affects the mechanical properties of the tissue.

We recall that the mechanical pulling forces $\boldsymbol{f}$ are proportional to the concentration of myofibroblasts and collagen molecules:

$$
\begin{aligned}
\boldsymbol{f} &= \nabla\cdot\boldsymbol{\psi} \\
\boldsymbol{\psi} &= \xi M\frac{\rho}{R^2+\rho^2}\overline{\overline{\boldsymbol{I}}}.
\end{aligned}
$$

The rate of change of the strain tensor is given by:

$$\overline{\overline{\boldsymbol{g}}} = \zeta \left[\frac{g(N,M,c,\rho)c}{\rho} \right] \overline{\overline{\boldsymbol{\varepsilon}}} = \zeta \left\{ \frac{[N+\eta^{II}M]\,c}{1+a_c^{III}c} \right\} \overline{\overline{\boldsymbol{\varepsilon}}}.$$

Following the result of Lemma 1, we have again symmetry of the strain tensor for $t \geq 0$.

The boundary conditions for the velocity field are homogenous Dirichlet. For the cell constituents we have $N(\boldsymbol{x},t) = \overline{N}$, $M(\boldsymbol{x},t) = \overline{M}$ and $c(\boldsymbol{x},t) = \overline{c}$ for $\boldsymbol{x} \in \partial\Omega_t$. Let $\mathcal{G}$ be a subdomain of our computational domain and $I_{\mathcal{G}}$ its indicator function. Supposing that the domain $\mathcal{G}$ describes the area of the skin graft, the initial conditions read:

$$\begin{aligned} N(\boldsymbol{x},0) &= \overline{N} - (\overline{N} - N^w) I_{\mathcal{G}}, \\ M(\boldsymbol{x},0) &= 0, \\ c(\boldsymbol{x},0) &= c^w I_{\mathcal{G}}, \\ \rho(\boldsymbol{x},0) &= \overline{\rho} \\ \boldsymbol{v}(\boldsymbol{x},0) = \boldsymbol{0} &\text{ and } \overline{\overline{\boldsymbol{\varepsilon}}}(\boldsymbol{x},0) = \boldsymbol{0}. \end{aligned}$$

As we are not dealing any more with homogeneous initial conditions, it becomes relevant how these are enforced. We will be using a combination of tanh to approximate the indicator function related to $I_{\mathcal{G}}$. The function

$$f(x) = (0.5 + 0.5\tanh(k(x+1)))(0.5 - 0.5\tanh(k(x-1))), \tag{5.4.2}$$

with $k \geq 0$, becomes a indicator function for $\mathcal{G} = [-1,1]$ for $k \to \infty$, i.e. value equal to one on $x \in [-1,1]$ and zero everywhere else. The extension to two dimensions and any rectangular domain with edges parallel to the axes is trivial. This approximation of the indicator function is very useful, as we can keep the gradients of the initial condition under control by varying the variable k. We will see in the coming sections how problems arise when steep gradients come into play and we can thus first focus on solving the equations and successively turn our attention to the treatment of high gradients.

The weak formulation for the velocity and strain equations have been derived in Section 5.3. Hence, we will only investigate the weak form of the conservation law of the constituents.

As already shown before, we start with multiplying the equation by a test function $\varphi \in H^1(\Omega_t)$ and with integrating the product over the domain Ω_t. The application of Reynold's Transport Theorem and Gauss' Divergence Theorem yields:

$$\frac{d}{dt}\int_{\Omega_t} z_i \varphi \, d\Omega = \int_{\Omega_t} \boldsymbol{J}_i \cdot \nabla\varphi + R_i \varphi + z_i \frac{D\varphi}{Dt} \, d\Omega - \int_{\partial\Omega_t} \varphi \boldsymbol{J}_i \cdot \boldsymbol{n} \, d\Gamma$$

Analogously as in previous sections, we discretize our domain and the model variables with the basis functions φ_k. The spatially discretized Galerkin equations read: Find $z_N^h, z_M^h, z_c^h \in H^1(\Omega_t)$ and $z_\rho^h \in L^2(\Omega_t)$ that satisfy for all basis functions φ_k

$$\frac{d}{dt}\int_{\Omega_t} z_i^h \varphi_k \, d\Omega = \int_{\Omega_t} \boldsymbol{J}_i^h \cdot \nabla\varphi_k + R_i^h \varphi_k \, d\Omega - \int_{\partial\Omega_t} \varphi_k \boldsymbol{J}_i^h \cdot \boldsymbol{n} \, d\Gamma,$$

where again we used that the integral with the term $\dfrac{D\varphi_k}{Dt}$ is zero. To discretize in time, the Euler backward method has been used. We first solve the biological system and after that we proceed with the mechanical system. The mechanical forcing term reads $f_1 = \partial/\partial x \left(\xi M \frac{\rho}{R^2+\rho^2}\right)$ for v_1 and $f_2 = \partial/\partial y \left(\xi M \frac{\rho}{R^2+\rho^2}\right)$ for v_2. Since we do not have any unknowns (as the cell constituents are solved separately) we can either directly approximate the full derivative or first analytically use the product rule of differentiation and then approximate the derivatives of the constituents. To better explain we take f_1 as an example.

1. Consider $g = \xi M \frac{\rho}{R^2+\rho^2}$. The values $g_j = g(\boldsymbol{x}_j)$, i.e. g at the nodes of our discretization, are known. Thus we can approximate

$$\frac{\partial g}{\partial x} \approx \sum_{j=1}^{N} g_j \frac{\partial \varphi_j}{\partial x}$$

2. Analytically expand $\partial g/\partial x$:

$$\frac{\partial g}{\partial x} = \xi \left\{ \frac{\partial M}{\partial x} \frac{\rho}{R^2+\rho^2} + M \left[\frac{\frac{\partial \rho}{\partial x}\left(R^2+\rho^2\right) + 2\rho^2 \frac{\partial \rho}{\partial x}}{\left(R^2+\rho^2\right)^2} \right] \right\}.$$

   Subsequently, approximate the cell variables' derivates.

We chose to use the second method, as it should yield more accurate results.

The construction of the linear system is performed similarly to the previous sections. As the process of linearization requires more detailed analysis, we will focus on that instead.

In the following section we will present two separate approaches to the linearization of our nonlinear system of equations. While the treatment and manipulation of certain terms is different, one aspect needs to be clarified in both cases during the construction of the element matrices: the values of nonlinear functions at Gaussian integration points using the solution at a previous iteration step.

Let $f(N, M, c, \rho)$ be a nonlinear function evaluated using the solution of the previous Picard iteration of the constituents. We have two choices for its evaluation:

- We can interpolate the previous solutions at the Gaussian points and then evaluate the function using the interpolated solution values;

- We can evaluate the function at the nodes and then interpolate its values at the Gaussian points.

If we denote with a subscript G_p the interpolation at a Gaussian point, then the first choice will result in $f(N_{G_p}, M_{G_p}, c_{G_p}, \rho_{G_p})$ and the second in $f_{G_p}(N, M, c, \rho)$. Given our choice of bilinear quadrilateral elements, it is supposed that our solution is piecewise bilinear. The second approach reflects this idea and hence is used in our calculations. This will not hold true for the G+Smo code in the second part of this work. As we do not have any direct influence on the calculation of the element matrices, given our lack of experience with the library, G+Smo will in fact calculate the matrices using the first choice.

5.4.1. LINEARIZATION

Patankar Source/Sink Separation For the linearization we will employ the Picard iteration method. Thus, variables which generate a nonlinear behaviour will be considered at a previous iteration. This is equivalent to considering the solution for that variable at the precedent time step evaluated on the new grid [20]. We will denote with an asterisk when a solution is being considered at a previous Picard iteration step. For the linearization of the flux term $\boldsymbol{J}_N$ we proceed as follows:

$$\boldsymbol{J}_N = -D_F F \nabla N + \chi_F N \nabla c \approx -D_F F^* \nabla N + \chi_F N \nabla c^*.$$

The treatment of the flux for M is analogous and there is no linearization needed for $\boldsymbol{J}_c$. The terms R_i can be considered as source terms for the respective equations. As such, we base our linearization procedure on the guidelines of Patankar's source/sink separation [21]. To consistently linearize the source term S of an always-positive variable we consider its source $S_1 > 0$ and sink $S_2 > 0$ terms s.t. $S = S_1 - S_2$. The proposed technique sets

$$S = S_1 - \frac{S_2}{u} u \approx S_1 - \frac{S_2}{u^*} u. \tag{5.4.3}$$

Since for our set of equations the source term is a function of u, i.e. $S = S_1(u) - S_2(u)$, this steps would lead to nonlinearity. As such we first set $S = S(u) \approx S(u^*)$ and then proceed with (5.4.3). In our notation we have $S = R_i$. The steps to linearize the source terms are:

1. We eliminate the dependence respective to the variable whose equation we are considering by taking its solution at the previous iteration step;

2. Terms involving the considered variable are treated as described in Patankar's technique and we set S_1 as a constant since it is being treated as such in (5.4.3);

3. Remaining nonlinearities are again eliminated using Picard's method.

The source terms R_N, R_M, R_c, R_ρ have been linearized as described in the following steps:

$$R_N \approx R_N^* = \underbrace{r_F\left[1+\frac{r_F^{\max}c}{a_c^I+c}\right][1-\kappa_F F]\,N^{*1+q}}_{S_1}\underbrace{-\left(k_F c N^* + \delta_N N^*\right)}_{-S_2},$$

$$\text{Patankar: } r_F\left[1+\frac{r_F^{\max}c^*}{a_c^I+c^*}\right][1-\kappa_F F^*]\,N^{*1+q} - k_F c N^*\frac{N}{N^*} - \delta_N N^*\frac{N}{N^*},$$

$$\text{Picard: } r_F\left[1+\frac{r_F^{\max}c^*}{a_c^I+c^*}\right][1-\kappa_F F^*]\,N^{*1+q} - k_F c^* N - \delta_N N;$$

$$R_M \approx R_M^* = \underbrace{r_F\left\{\frac{[1+r_F^{\max}]\,c}{a_c^I+c}\right\}[1-\kappa_F F]\,M^{*1+q} + k_F c N}_{S_1}\underbrace{-\delta_M M^*}_{-S_2},$$

$$\text{Patankar: } r_F\left\{\frac{[1+r_F^{\max}]\,c^*}{a_c^I+c^*}\right\}[1-\kappa_F F^*]\,M^{*1+q} + k_F c^* N^* - \delta_M M^*\frac{M}{M^*},$$

$$\text{Picard: } r_F\left\{\frac{[1+r_F^{\max}]\,c^*}{a_c^I+c^*}\right\}[1-\kappa_F F^*]\,M^{*1+q} + k_F c^* N c^* - \delta_M M;$$

$$R_c \approx R_c^* = \underbrace{k_c\left[\frac{c^*}{a_c^{II}+c^*}\right][N+\eta^I M]}_{S_1}\underbrace{-\delta_c g(N,M,c^*,\rho)c^*}_{-S_2},$$

$$\text{Patankar: } k_c\left[\frac{c^*}{a_c^{II}+c^*}\right][N^*+\eta^I M^*] - \delta_c g(N,M,c^*,\rho)c^*\frac{c}{c^*},$$

$$\text{Picard: } k_c\left[\frac{c^*}{a_c^{II}+c^*}\right][N^*+\eta^I M^*] - \delta_c g(N^*,M^*,c^*,\rho^*)c;$$

$$R_\rho \approx R_\rho^* = \underbrace{k_\rho\left\{1+\left[\frac{k_\rho^{\max}c}{a_c^{IV}+c}\right]\right\}[N+\eta^I M]}_{S_1}\underbrace{-\delta_\rho g(N,M,c,\rho^*)\rho^*}_{-S_2},$$

$$\text{Patankar: } k_\rho\left\{1+\left[\frac{k_\rho^{\max}c^*}{a_c^{IV}+c^*}\right]\right\}[N^*+\eta^I M^*] - \delta_\rho g(N,M,c,\rho^*)\rho^*\frac{\rho}{\rho^*},$$

$$\text{Picard: } k_\rho\left\{1+\left[\frac{k_\rho^{\max}c^*}{a_c^{IV}+c^*}\right]\right\}[N^*+\eta^I M^*] - \delta_\rho g(N^*,M^*,c^*,\rho^*)\rho.$$

Naive Approach We will present a more naive linearization approach. Every term in the equation not being the respective variable is taken at the previous Picard iteration. The purpose is to see how the changes in the S_1 terms would affect the results. The asterisk again denotes that we are considering the values of the previous Picard iteration. The flux terms' manipulation remains unchanged. The nonlinear components of the reaction terms are treated as follows:

$$R_N \overset{Picard}{\approx} R_N^* = r_F\left[1+\frac{r_F^{\max}c^*}{a_c^I+c^*}\right][1-\kappa_F F^*]\,N^{*q}N - k_F c^* N - \delta_N N;$$

$$R_M \overset{Picard}{\approx} R_M^* = r_F\left\{\frac{[1+r_F^{\max}]\,c^*}{a_c^I+c^*}\right\}[1-\kappa_F F^*]\,M^{*q}M + k_F c^* N^* - \delta_M M;$$

$$R_c \overset{Picard}{\approx} R_c^* = k_c\left[\frac{c}{a_c^{II}+c^*}\right][N^*+\eta^I M^*] - \delta_c g(N^*,M^*,c^*,\rho^*)c;$$

$$R_\rho \overset{Picard}{\approx} R^*_\rho = k_\rho \left\{ 1 + \left[\frac{k_\rho^{\max} c^*}{a_c^{IV} + c^*} \right] \right\} [N^* + \eta^I M^*] - \delta_\rho g(N^*, M^*, c^*, \rho^*)\rho.$$

Given that q is negative, we will impose during the element matrix assembly, that if N or M are equal to zero, then N^q and M^q will be considered equal to zero.

5.4.2. COMPUTATIONAL IMPROVEMENTS

Since we are dealing with a symmetric domain with symmetric initial conditions, also the solution will inherit this useful property. In fact, this allows to consider only one fourth of the whole domain in our calculations, resulting in great benefits regarding the computational speed. To fully exploit this property, we need to define appropriate boundary conditions along our symmetry domains. For the cell constituents this results in simply imposing a zero flux condition along the symmetric boundary:

$$\boldsymbol{J}_i \cdot \boldsymbol{n} = 0,$$

where $\boldsymbol{n}$ is the outward pointing normal vector. Similar conditions are imposed on the mechanical equations:

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = 0,$$
$$\boldsymbol{v} \cdot \boldsymbol{n} = 0.$$

We are now able to perform the simulation on a quarter of the original domain.

Another improvement in the simulation is the introduction of a variable time step. It has been noted, that the part of the simulation, where the cell constituents have reached their peak concentration and are diminishing towards their respective equilibrium values, is the most ‘complicated’ part. This becomes apparent twice: once in the (non-)convergence of the mechanical Picard iteration (before maximum iterations are reached) and the second time in the negative values arising in the flux corrected transport (FCT) algorithm, which will later be described in Chapter 6. We suppose that in this phase the Picard operator does cease to be a contraction for the chosen value of Δt. To combat this and to improve overall the efficiency of the simulation, we implemented a convergence verification step. If the chosen time step does lead to convergence of the Picard iteration, we accept the computed solution and advance in time. If the same time step ‘succeeds’ ten times, then the time step is incremented by a factor of 1.1. However, if convergence is not reached within the predetermined number of Picard iterations, then the computed solution is discarded and the algorithm starts over with a time step reduced by a factor of 0.9.

5.4.3. PRELIMINARY RESULTS

We are now able to set up the linear system to be solved for the cell variables and can combine it with the mechanical equations of morphoelasticity. We noticed that the widely used backslash operator of MATLAB '\' introduced noise in the mechanically stationary solution as iterations went on. This occurred with Patankar’s linearization technique, but not when the naive approach was used. To avoid this, we opted to use the BiCGSTAB iterative solver with a modified incomplete LU decomposition as preconditioning as recommended in [22] for non-symmetric systems and in [23]. Possible negative values are cut off and set to be equal to zero.

The parameters used for the computations in Section 5.4 are listed in Table 5.4.1. In Figure 5.4.2, we can see the solution to our set of equations at $T = 27$. The initial domain $\Omega_0 = [-4,4]^2$ and the wound domain is $\mathcal{G} = [-1,1]^2$. These results are close to the ones found in [6]. However, for larger values of k it becomes apparent that the steep gradients cause large oscillations in the solution. As a consequence, we have negative values in our solutions of the cell constituents, see Figure 5.4.1. If we consider the fact that the unknowns we are solving for are in reality the concentrations of certain cell components, we understand that negative values are not physically acceptable. Cutting off the solution and setting it to zero where needed introduces errors and it would be ideal to avoid relying on this rudimentary practice. To make up for this, we intend to briefly discuss the commonly used stabilization techniques in FEM. These are the SUPG and the algebraic flux correction method, which has been shown to be effective in similar cases [6, 23–25].

5.4.4. STABILIZATION

Our set of constituents equations in (5.4.1) can be interpreted as coupled convection-diffusion-reaction equations. To solve this type of nonlinearly coupled sets of equations, sharp interfaces need to be resolved accurately and spurious oscillations must be avoided; especially in the case of always-positive variables where oscillations could give rise to non-physical negative values in the solved variables.

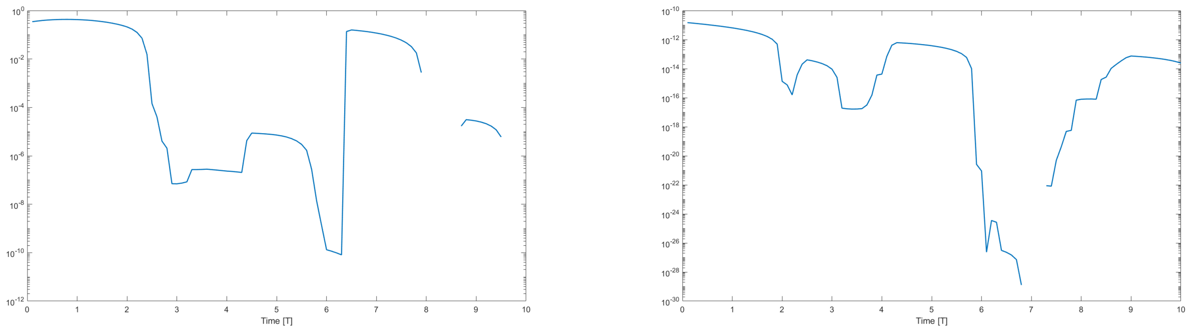


Figure 5.4.1: Absolute values of negative terms in the solution of M and c for $k = 50$ using the naive linearization. Discontinuities indicate that the absolute value was zero.

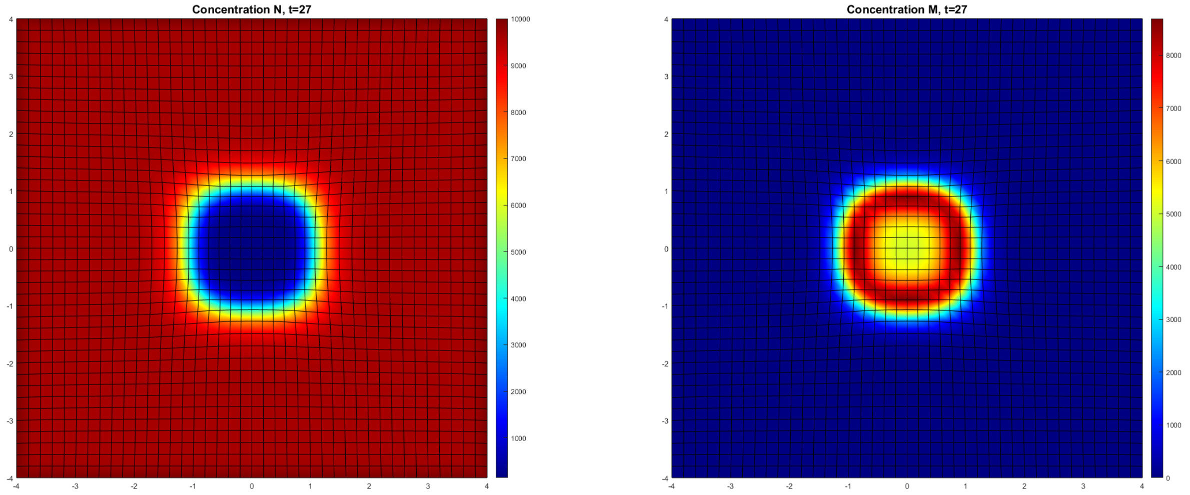
When increasing the value of k in our initial condition, and thus its gradient, it becomes apparent how the current numerical scheme is not appropriate for our set of equations. The numerical oscillations are non-physical and can adversely affect the results introducing errors. It is well known, that for diffusion-convection equations the standard Galerkin method does suffer from this oscillatory solution behaviour, when they are convection dominated. It is common to combat this unwanted effect by introducing numerical diffusion. To limit its effects, artificial diffusion is added in the streamline direction only. A commonly used method, which follows this reasoning, is the Streamline Upwind Petrov Galerkin (SUPG) method, e.g. [23, 24, 26].

The SUPG method can be seen as the standard Galerkin method, where instead of multiplying by a set of test functions φ, the equations are multiplied by a modified set of test functions of the type $\varphi + \tau \boldsymbol{v} \cdot \nabla\varphi$, where $\boldsymbol{v}$ is the convective velocity and τ a problem-dependent parameter. However, as stated in [23] and shown in [24] this method can still produce oscillatory results. In [24] different stabilization techniques are compared including SUPG and algebraic flux correction methods. The latter methods are shown to yield overall better results.

Opposed to SUPG, algebraic flux correction methods do not modify the Galerkin equations, but rather act on a discrete *algebraic* level. They manipulate the constructed matrices, thus adding artificial diffusion and aim for a low order solution with no negative values. Successively, anti-diffusive terms are added to recover as much as possible of the original high order solution. Furthermore, no ‘arbitrary’ parameter, as τ for the SUPG method, is needed, thus making it a more robust option. In our work, we will implement a flux correcting method, as also done in [6], and will investigate the effects it has on the solution of our model.

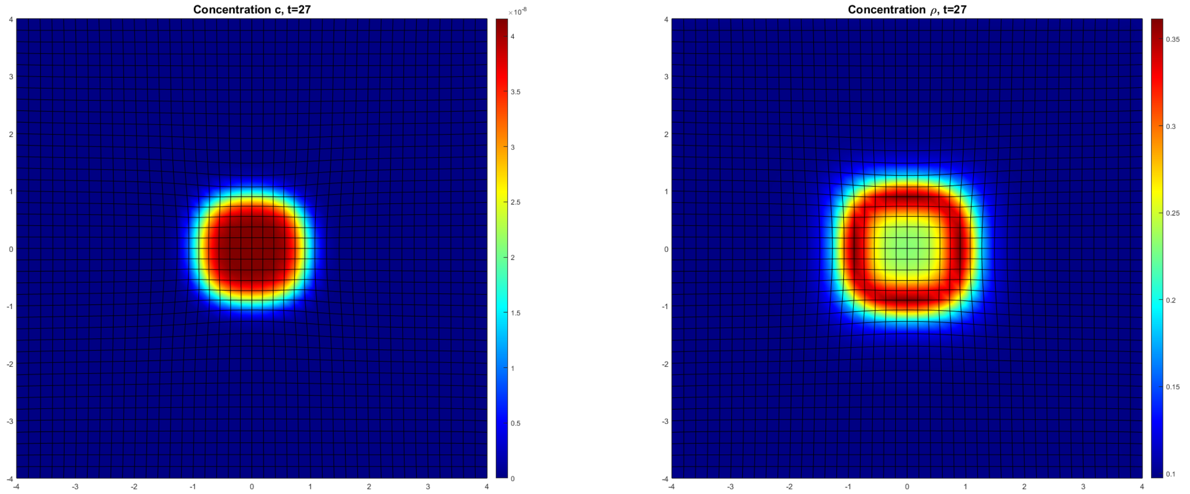
Parameters	Values	Dimensions
D_F	10^{-7}	$cm^5/(cells\, day)$
χ_F	2×10^{-3}	$cm^5/(g\, day)$
q	-4.2×10^{-1}	-
r_F	9.24×10^{-1}	$cm^{3q}/(cells^q\, day)$
r_F^{max}	2	-
a_c^I	10^{-8}	g/cm^3
κ_F	10^{-6}	$cm^3/cells$
k_F	1.08×10^7	$cm^3/(g\, day)$
δ_N	2×10^{-2}	$/day$
δ_M	6×10^{-2}	$/day$
D_c	2.9×10^{-3}	cm^2/day
k_c	4×10^{-13}	$g/(cells\, day)$
η^I	2	-
a_c^{II}	10^{-8}	g/cm^3
δ_c	5×10^{-4}	$cm^6/(cells\, g\, day)$
η^{II}	5×10^{-1}	-
a_c^{III}	2×10^8	g/cm^3
k_ρ	6×10^{-8}	$g/(cells\, day)$
k_ρ^{max}	10	-
a_c^{IV}	10^{-9}	g/cm^3
δ_ρ	6×10^{-6}	$cm^6/(cells\, g\, day)$
ρ_t	1.02	g/cm^3
μ_1	10^2	$(N\, day)/cm^2$
μ_2	10^2	$(N\, day)/cm^2$
E	3.2×10	$N/(g\, cm)^{1/2}$
ν	4.9×10^{-1}	-
ξ	5×10^{-2}	$(N\, g)/(cells\, cm^2)$
R	9.95×10^{-1}	g/cm^3
ζ	9×10^2	$cm^6/(cells\, g\, day)$
$\overline{N}$	10^4	$cells/cm^3$
$\overline{M}$	0	$cells/cm^3$
$\overline{c}$	0	g/cm^3
$\overline{\rho}$	10^{-1}	g/cm^3
N^w	2×10^3	$cells/cm^3$
c^w	10^{-8}	g/cm^3

Table 5.4.1: Parameters used in Section 5.4.3 [6]



(a) Concentration of N at $T = 27$.

(b) Concentration of M at $T = 27$.



(c) Concentration of c at $T = 27$.

(d) Concentration of ρ at $T = 27$.

Figure 5.4.2: Cell variable concentrations at $T = 27$, with $k = 3$ in (5.4.2).

6

ALGEBRAIC FLUX CORRECTION

We already have stressed the importance of a positivity preserving scheme that can handle well steep gradients. While cutting off the solutions at zero is possible, it clearly is not an ideal conclusion to this issue. We need an algorithm that guarantees positivity without altering the solution with ‘brute force’ as we did in Section 5.4. At the same time, it is best not to rely on parameter-dependent methods like the SUPG method. In order to address this issue, we will implement the flux-corrected transport (FCT) algorithm presented in [27]. In the article, a convection-diffusion equation is solved on both fixed and moving domains. We will attempt to adapt the method to our convection-diffusion-reaction equations accordingly.

6.1. THE FCT-ALGORITHM

In [27] the considered convection-diffusion equation leads to the following system of differential algebraic equations for the unknown $c(t)$:

$$\frac{d}{dt}[M_C(t)c(t)] = [K(t)+S(t)]c(t)+q(t).$$

Here, $M_C(t)$ is the consistent mass matrix, $K(t)$ includes the convection terms, $S(t)$ is the diffusion matrix and $q(t)$ the contribution of known terms. The artificial diffusion operator D is added to the convection matrix in order to construct a nonoscillatory low-order counterpart to the latter:

$$L = K + D,$$

where $d_{ij} = \max\{-k_{ij}, 0, -k_{ji}\}$ for $j \neq i$ and $d_{ii} = -\sum_{j\neq i} d_{ij}$. The property of zero row and column sums ensures mass conservation on a discrete level [27]. A low-order solution c^L of $c(t)$ is calculated by solving the equation:

$$\frac{d}{dt}[M_L(t)c_L] = [L(t)+S(t)]c(t)+q(t) \quad t\in(t^n, t^{n+1}), \tag{6.1.1}$$

where $M_L(t)$ is the lumped mass matrix, i.e. $M_L = \text{diag}\{m_i\}$, where $m_i = \sum_j m_{ij}$.

The algebraic ODE (6.1.1) is solved using the implicit midpoint rule:

$$\begin{aligned} Ac^L &= Bc^n + \Delta t q^{n+1/2}, \\ A &= M_L^{n+1} - \frac{\Delta t}{2}[L^{n+1/2}+S^{n+1/2}], \\ B &= M_L^n + \frac{\Delta t}{2}[L^{n+1/2}+S^{n+1/2}], \end{aligned}$$

where the superscripts refer to the time evaluation of the matrices and terms, where $t^{n+1/2} = (t^n + t^{n+1})/2$. The aim of the FCT algorithm is to produce a matrix A which is an M-matrix and to have a matrix B with only non-negative components. It can be proved that, under such conditions, the low order scheme is positivity preserving [23]. It is now necessary to evaluate the approximation of the nodal derivatives using

$$M_C^{n+1}\dot{c}^L = \left[K^{n+1}+S^{n+1}\right]c^L + q^{n+1}. \tag{6.1.2}$$

With these we proceed to calculate the anti-diffusive flux terms f_{ij} as

$$f_{ij} = m_{ij}^{n+1}\left(\dot{c}_i^L - \dot{c}_j^L\right) + d_{ij}^{n+1}\left(c_i^L - c_j^L\right), \quad f_{ji} = -f_{ij}.$$

Finally, we recover the high order solution c^{n+1} by applying the anti-diffusive correction

$$M_L^{n+1} c^{n+1} = M_L^{n+1} c^L + \Delta t \bar{f}, \quad \bar{f}_i = \sum_{j \neq i} \alpha_{ij} f_{ij}. \tag{6.1.3}$$

The correction factors in (6.1.3) are obtained with Zalesak's limiter. In [27] the following preliminary step is recommended:

$$f_{ij} := 0, \quad \text{if} \quad f_{ij}\left(c_j^L - c_i^L\right) > 0.$$

The motivation given is that if the flux f_{ij} has the same sign as $\left(c_j^L - c_i^L\right)$, then it tends to be diffusive and to flatten the solution instead of steepening it. The steps to determine the factors α_{ij}, as described in [27], are:

1. Compute the sums of positive/negative antidiffusive fluxes into node i

$$P_i^+ = \sum_{j \neq i} \max\{0, f_{ij}\}, \quad P_i^- = \sum_{j \neq i} \min\{0, f_{ij}\}$$

2. Compute the distance to a local extremum of the auxiliary solution c^L

$$\begin{aligned} Q_i^+ &= \max\left\{0, \max_{j \neq i}\left(c_j^L - c_i^L\right)\right\}, & Q_i^- &= \min\left\{0, \min_{j \neq i}\left(c_j^L - c_i^L\right)\right\} \\ &= \max_j\left\{c_j^L - c_i^L\right\} & &= \min_j\left\{c_j^L - c_i^L\right\} \end{aligned}$$

3. Compute the nodal correction factors for the net increment to node i

$$R_i^+ = \min\left\{1, \frac{m_i Q_i^+}{\Delta t P_i^+}\right\}, \quad R_i^- = \min\left\{1, \frac{m_i Q_i^-}{\Delta t P_i^-}\right\}$$

4. Define α_{ij} so as to satisfy the positivity constraint for nodes i and j

$$\alpha_{ij} = \begin{cases} \min\left\{R_i^+, R_j^-\right\}, & \text{if } f_{ij} > 0 \\ \min\left\{R_i^-, R_j^+\right\}, & \text{otherwise} \end{cases}$$

In [24] it is said that any time integration method would be accepted “at least under certain time step restrictions”.

6.2. ADAPTATION TO OUR EQUATIONS

A few adaptations have been made in order to apply the described algorithm to our case. First and foremost we are working with a system of equations opposed to a single one. In general, this should not influence the algorithm itself, other than the fact that for instance the mass matrix M_C will be a block diagonal matrix with the ‘original’ mass matrix $\widetilde{M}_C$ as diagonal entries. The block structures of the considered matrices for the linearization approach described as Patankar in 5.4.1 are as follows:

$$M_C = \begin{bmatrix} \widetilde{M}_C & \emptyset & \emptyset & \emptyset \\ \emptyset & \widetilde{M}_C & \emptyset & \emptyset \\ \emptyset & \emptyset & \widetilde{M}_C & \emptyset \\ \emptyset & \emptyset & \emptyset & \widetilde{M}_C \end{bmatrix}$$

$$K = \begin{bmatrix} K_N & \emptyset & \emptyset & \emptyset \\ \emptyset & K_M & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{bmatrix}$$

$$S = \begin{bmatrix} S_N & \emptyset & \emptyset & \emptyset \\ \emptyset & S_M & \emptyset & \emptyset \\ \emptyset & \emptyset & S_c & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{bmatrix} \tag{6.2.1}$$

$$R = \begin{bmatrix} R_N & \emptyset & \emptyset & \emptyset \\ \emptyset & R_M & \emptyset & \emptyset \\ \emptyset & \emptyset & R_c & \emptyset \\ \emptyset & \emptyset & \emptyset & R_\rho \end{bmatrix}$$

$$q = \begin{bmatrix} rhs_N \\ rhs_M \\ rhs_c \\ rhs_\rho \end{bmatrix},$$

where we omitted time dependence for simplicity. The steps described in Section 6.1 have mostly been kept the same, except for the calculations of $Q^{\pm}$. In this case, $Q_i^{\pm}$ is first calculated for the low-order solutions of our cell constituents separately and then merged into one $Q^{\pm} = \left[Q_N^{\pm}, Q_M^{\pm}, Q_c^{\pm}, Q_\rho^{\pm}\right]$. Another small change employed is that whenever $P_i^{\pm}$ is equal to zero, then $R_i^{\pm}$ is equal to zero as indicated in [28]. Furthermore, whenever in Section 6.1 the diffusion matrix S is considered in the algorithm, we also add the reaction matrix R. Lastly, to enforce the boundary conditions at each solution step of the algorithm, Dirichlet boundary conditions have been enforced. This means that for each cycle of the algorithm we applied the prescribed boundary conditions for the low-order solution and homogeneous boundary conditions for the calculation of the nodal derivatives. However, for the anti-diffusive correction step no boundary conditions are applied, as we are not solving for our unknowns, but simply trying to recover the high-order solution by adding anti-diffusive terms.

If we consider the linearization approach described as naive in 5.4.1, then the block structure of the matrices will be the same as in (6.2.1). Compared to (6.2.1) only q changes:

$$q = \begin{bmatrix} \emptyset \\ rhs_M \\ \emptyset \\ rhs_\rho \end{bmatrix}.$$

We have shown the construction and structure of our block matrices. In Section 6.1, most of the matrices are evaluated at $t^{n+1/2} = t^n + \Delta t/2$ for the calculation of the low order solution. As such, the considered mesh points are $\boldsymbol{X}^{n+1/2} = \boldsymbol{X}^n + \Delta t \boldsymbol{V}^n/2$. It is now important to consider the linearization process of our discretized equations: we need solution values at a previous iteration step. Earlier, when evaluating matrices at the time step t^{n+1}, we used the previous Picard iteration solution as information, since that was the candidate solution for the time step $n+1$. If we do the same now, we would map the candidate solution for t^{n+1} onto the mesh at time step $n+1/2$. Conversely, if we simply use the solution for t^n for the linearization, we would suppose that the constituents are piece-wise constant in time. However, this would result in no real Picard iteration, since for each loop after the first one we would perform the same calculations. We therefore suppose piecewise linear solutions in time for the constituents and will use $\boldsymbol{u}^{n+1/2} = (\boldsymbol{u}_{pic}^{(m)} + \boldsymbol{u}^n)/2$. Here, $\boldsymbol{u}_{pic}^{(m)}$ represents the

Picard solution at the previous, m-th, iteration and $\boldsymbol{u}^n$ is the solution at time step t^n. To initiate the Picard loop we consider $u_{pic}^{(0)} = u^n$.

Changes to the Original FCT Algorithm Even though the chosen FCT algorithm greatly meliorates the problem of negative values, it does not completely solve it. The first modification was to include the matrices S and R in the calculation of D. Hence, the artificial diffusion in the low order solution is not only trying to dampen the possibly negative effects of K, but of the sum $K+S+R$ instead. Using this expedient, as it was also done in [24], we managed to have non-negative low order solutions c^L also in the case of an adaptively refined mesh, see Figure 6.3.1. The final, high order solution however still has negative values. These are introduced by the anti-diffusive fluxes. We therefore decided to perform the calculation of the nodal derivatives using a low order approximation of the equations. This entails, that we will use the lumped matrix M_L instead of the consistent one M_C and that again we will calculate D with respect to $K+S+R$ and add it to the calculations. Hence, equation (6.1.2) becomes:

$$M_L^{n+1}\dot{c}^L = \left[K^{n+1} + S^{n+1} + R^{n+1} + D^{n+1}\right] c^L + q^{n+1}.$$

6.3. RESULTS OF THE FCT ALGORITHM

The described algorithm does preserve positivity, when applied to a regular quadrilateral grid, as for instance the one shown in Figure 5.4.2. The peaks arising at the wound boundary are still present as they also are in [6], but it is not necessary any more to correct negative terms at each time step.

In order to limit the mentioned peaks, we implemented the adaptive meshing algorithm described in [29] (`https://github.com/aschmidtuulm/ameshref`).This allowed us to have a finer mesh in our region of interest, while keeping the number of elements at a manageable quantity for our hardware, i.e. a home desktop. Indeed, the magnitude of the peaks diminishes the finer the mesh is across the wound boundary. However, it is again necessary to correct for negative values in the low order solution of our constituents using the changes mentioned in the previous section. The diffusion matrix is likely the reason for this. As stated in [23], the discrete diffusion operator does not necessarily satisfy the discrete maximum principle, if the mesh in not *nonnarrow*, i.e. if the ratio between longest and shortest edges of each rectangle is greater than $\sqrt{2}$. With the adaptive mesh algorithm we used, the ratio between longest and shortest edges is in fact equal to two for some elements. Usually the convective terms are the 'bad' factor, but as just described also the diffusive terms can lead to undesired effects in our solution. It is therefore needed to counteract this negative influence.

For the uniform mesh case, the standard FCT algorithm described in [27] was sufficient to ensure a positive low-order solution. The modifications applied to the calculation of the nodal derivatives, i.e. the use of the lumped mass matrix M_L and of the correction matrix D when only dependent on K, were sufficient to ensure a positive high order reconstruction using the uniform mesh. By applying the matrix manipulations on all matrices K, S, R, as in [24], we eliminated the appearance of negative terms in the low order solution also for the adaptive mesh case. In fact, in doing this the whole term $K+S+R$ is seen as a possible cause for problems and the FCT algorithm strives to counteract any unfavourable effects. The sole negative values have a magnitude in the order of 10^{-26} (less than machine precision) at most and are introduced by the anti-diffusive correction step and are explicitly set to zero in the code. These terms occur during the skin relaxation phase of our simulation. To note, this is exactly the same phase which was causing convergence problems in the Picard iterations of the mechanical part. We therefore have additional proof of the computational difficulty to model this phenomenon correctly and accurately. Given the magnitude of these terms, we don't consider this practice to be detrimental to the accuracy of the proposed method.

The FCT algorithm also helped us solve an underlying issue of the proposed linearization techniques described for the biochemical model: which of the two approaches should be used? When applying the modified FCT algorithm where D is calculated only depending on K, we could see how the naive approach did maintain the desired property of positivity of the low order solution with the uniform mesh, while Patankar's approach resulted in small negative values after $T \approx 70$. We have therefore decided to only consider the results deriving from the naive approach.

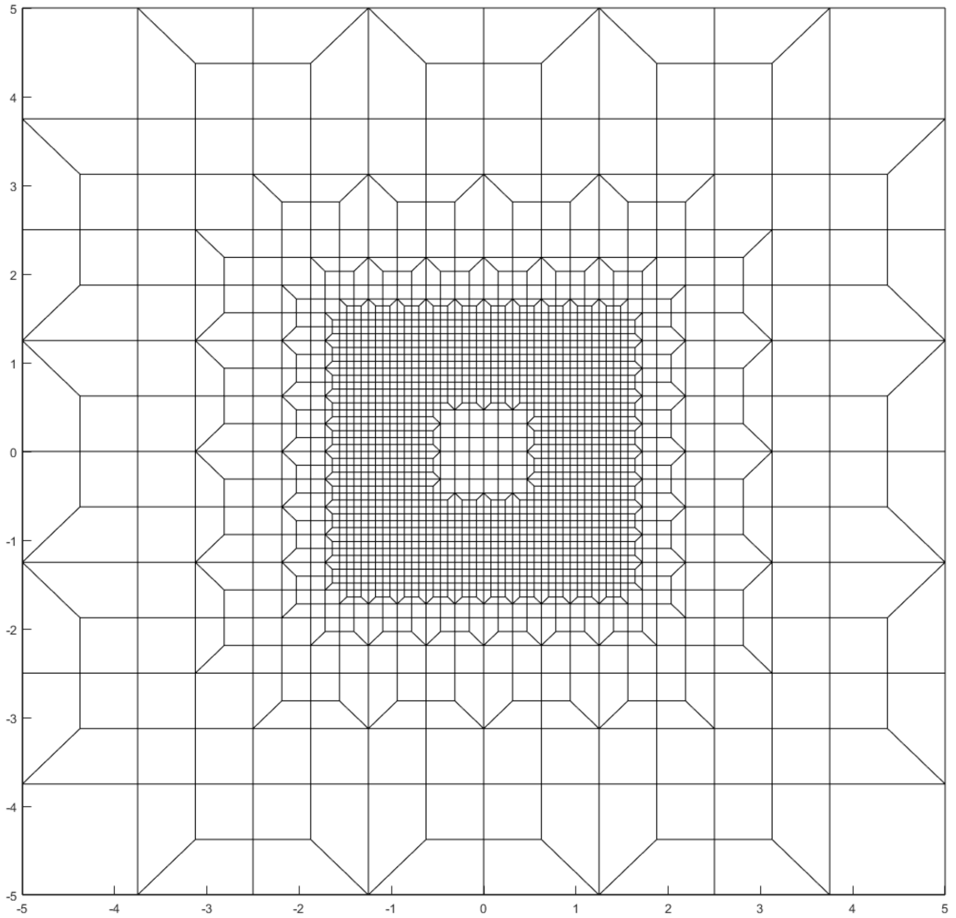


Figure 6.3.1: Adaptive mesh with 2200 elements.

7

Results

We will now present our results from the algorithms described so far. The first sections about the sole mechanical part are useful to develop a sort of intuition and further our understanding of the dynamics of the skin tissue. Thus, we can better predict how the wound area will evolve in time for a burn wound. Lastly, we will investigate the quality of our results and the accuracy of our discretizations. All calculations in this chapter have been performed using MATLAB.

7.1. 1D Morphoelasticity

With the one dimensional morphoelastic model we want to be able to see the plastic deformation compared to a purely viscoelastic one. We therefore apply increasing contracting body forces on the whole domain, which will then become smaller after the hypothetical contraction process is concluded at t_{ap}. We expect to see the tissue shrink and gradually re-expand after t_{ap}, but it should not return to its original length.

In Figure 7.1.1 we can see the total length variation of a tissue with $L(0) = 10$, when subject to the following body forces:

$$F_b(x,t) = \begin{cases} 0, & t < t_{f0} \\ \overline{F}(x)\left(1 - \exp\left(-c_{fd}\frac{t-t_{f0}}{t_{fm}-t_{f0}}\right)\right), & t_{f0} \le t < t_{\mathrm{ap}} \\ \overline{F}(x)\left(1 - \exp\left(-c_{fd}\frac{t-t_{f0}}{t_{fm}-t_{f0}}\right)\right)\exp\left(-\left(t - t_{ap}\right)\right), & t \ge t_{ap} \end{cases} \tag{7.1.1}$$

The parameter ζ represents the rate of morphoelastic change, "i.e. the rate at which the effective strain changes actively over time" [6]. If ζ is set to zero, the effective strain is not evolving thus modelling a viscoelastic deformation. This change in the mechanical response of the system can be observed in Figure 7.1.1. The results reflect our suppositions and we can clearly see the contraction of skin and its relaxation even though it does not return to its original disposition.

The parameters for the simulation can be found in Table 7.1.1.

Parameters	Values	Dimensions
E	31	N
μ	100	$N day$
ρ	1.02	g/cm
ζ	0.05	-
c_{fd}	4	-
t_{f0}	0.1	day
t_{fm}	20	day
t_{ap}	22	day
$\overline{F}$	-4.2	N/cm

Table 7.1.1: Parameters used in Section 5.1 [13]

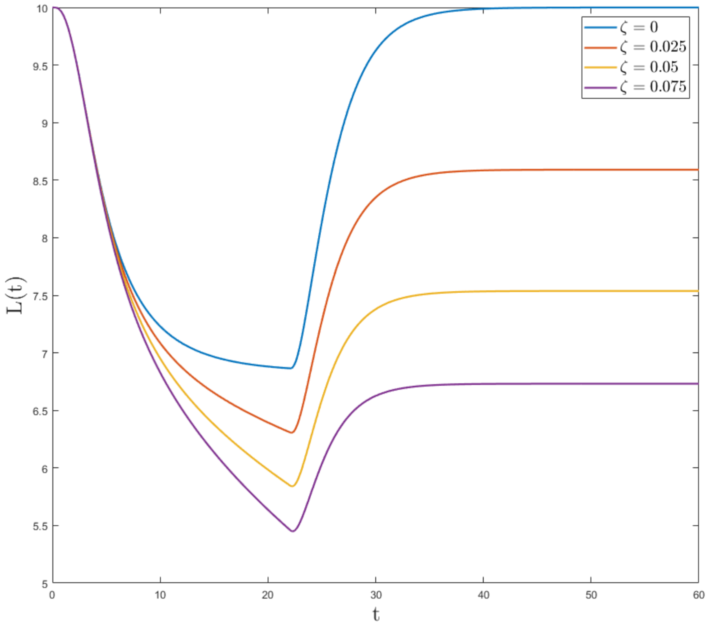


Figure 7.1.1: Length of the tissue described by (5.1.1) in time.

7.2. 2D VISCOELASTICITY

The addition of one spatial dimension greatly complicates the mechanical equations. The main new dynamical feature of a two dimensional body compared to a one dimensional one is the ability to rotate. Additionally, two dimensional bodies display the so called Poisson effect under compression/elongation. Under compression the object becomes shorter in the direction of the compressive load and wider laterally (conversely for elongation).

In Figures 7.2.1a, 7.2.1b, 7.2.2a we can see the strains of a tissue, $\Omega_0 = [-6,6]^2$, subject to the following forces:

$$\boldsymbol{f}(x,y,t) = \begin{cases} (F_b(t)\,\mathrm{sign}(y), 0) & \text{on } [-1.5, 1.5]^2 \\ (0,0) & \text{else ,} \end{cases} \tag{7.2.1}$$

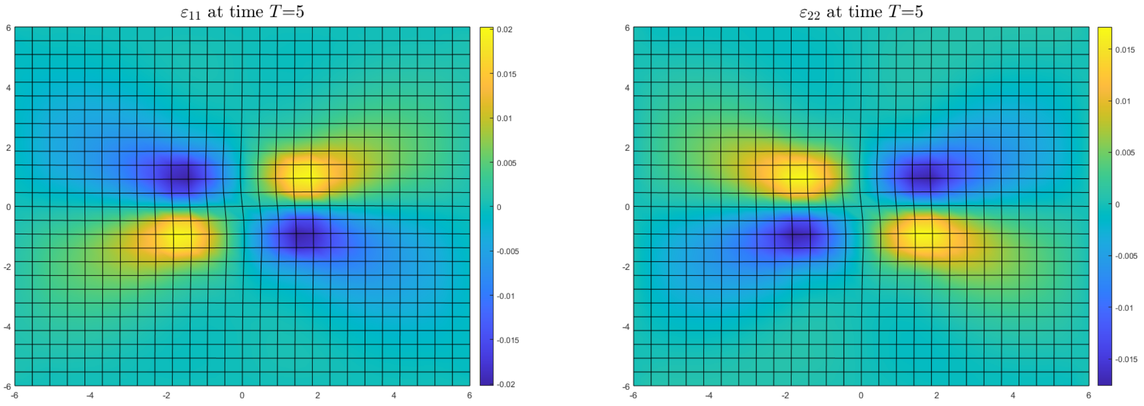
where F_b is defined in (7.1.1). In Figure 7.2.2b we can observe how a subdomain is deformed under these forces. The parameters used are listed in Table 7.2.1.

Parameters	Values	Dimensions
E	31	$N/(g\,cm)^{1/2}$
μ_1	100	$N\,day/cm^2$
μ_2	100	$N\,day/cm^2$
ρ	1.02	g/cm^3
ν	0.48	-
c_{fd}	4	-
t_{f0}	0.1	day
t_{fm}	20	day
t_{ap}	22	day
$\overline{F}$	-4.2	N/cm^3

Table 7.2.1: Parameters used in section 5.2 [13]

In Figure 7.2.3a we can see the strain of a tissue, $\Omega_0 = [-6,6]^2$, subject to the following forces:

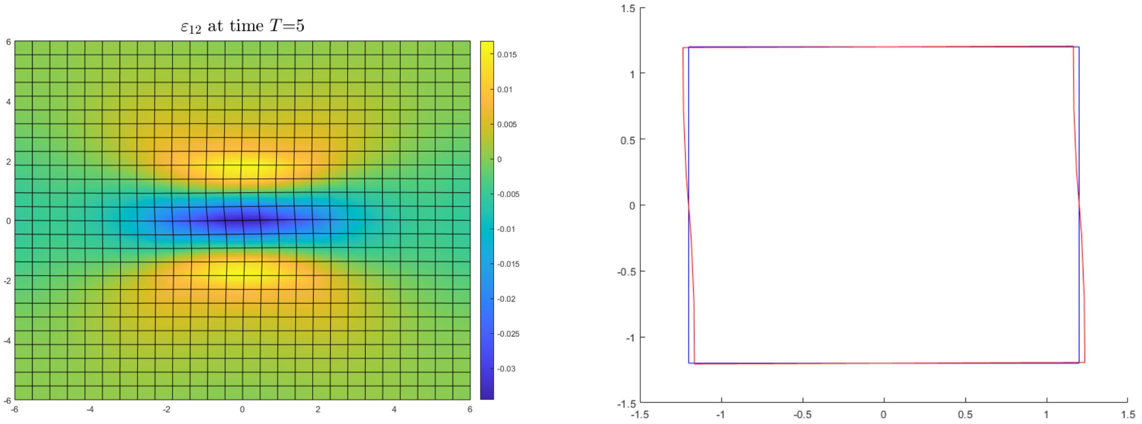
$$\boldsymbol{f}(x,y,t) = \begin{cases} (F_b(t)\,\mathrm{sign}(x), 0) & \text{on } [-1.5, 1.5]^2 \\ (0,0) & \text{else ,} \end{cases} \tag{7.2.2}$$



(a) ε_{11} at $T = 5$ of the tissue described in (5.2.1)

(b) ε_{22} at $T = 5$ of the tissue described in (5.2.1)

Figure 7.2.1: Tissue subject to body forces (7.2.1)



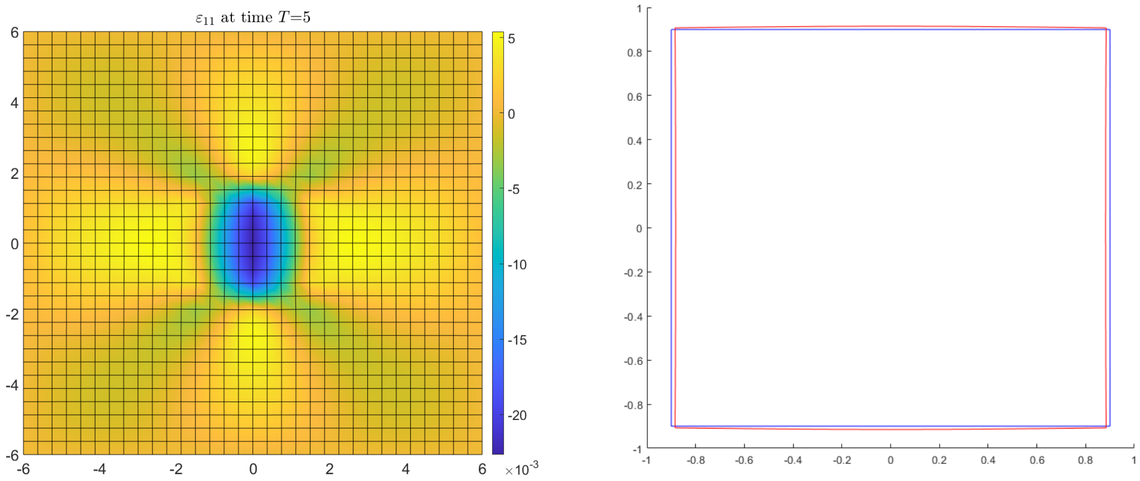
(a) ε_{12} at $T = 5$ of the tissue described in (5.2.1).

(b) Deformation of subdomain (red) at $T = 5$ of the tissue described in (5.2.1). In blue the original shape at $T = 0$.

Figure 7.2.2: Tissue subject to body forces (7.2.1)

where F_b is defined in (7.1.1). In Figure 7.2.3b we can observe how a subdomain is deformed under these forces.

In Figure 7.2.2b we observe the rotation of the considered subdomain. In Figure 7.2.3b we can see the Poisson effect in our tissue as it widens along the y-axis while under compressing forces along the x-axis. Both mechanical features mentioned at the beginning of the section have been reproduced with success.



(a) ε_{11} at $T = 5$ of the tissue described in (5.2.1).

(b) Deformation of subdomain (red) at $T = 5$ of the tissue described in (5.2.1). In blue the original shape at $T = 0$.

Figure 7.2.3: Tissue subject to body forces (7.2.2)

7.3. 2D MORPHOELASTICITY

The morphoelastic equations in one dimension are linear. The same does not hold true in the two dimensional case. We therefore need to iteratively solve our system of equations. We expect to notice similar behaviours of compression and relaxation as in the one dimensional case. To do so, we compare the original mesh at $t = 0$ to the mesh at the observed time step. Under both contracting and rotating forces we see that the mesh does deform with decreasing velocity (dampening effect of viscoelastic law) and once the forces cease to act the tissue returns towards its original state without attaining it completely.

We used the initial domain $\Omega_0 = [-1,1]^2$ and applied the following symmetric contracting body forces:

$$\begin{cases} f_1 = \tau_c((x<0)-(x>0))(t<t_{ap}) \\ f_2 = \tau_c((y<0)-(y>0))(t<t_{ap}), \end{cases} \tag{7.3.1}$$

where the logical expressions have value one if true and zero if false. The parameters used can be found in Table 7.3.1.

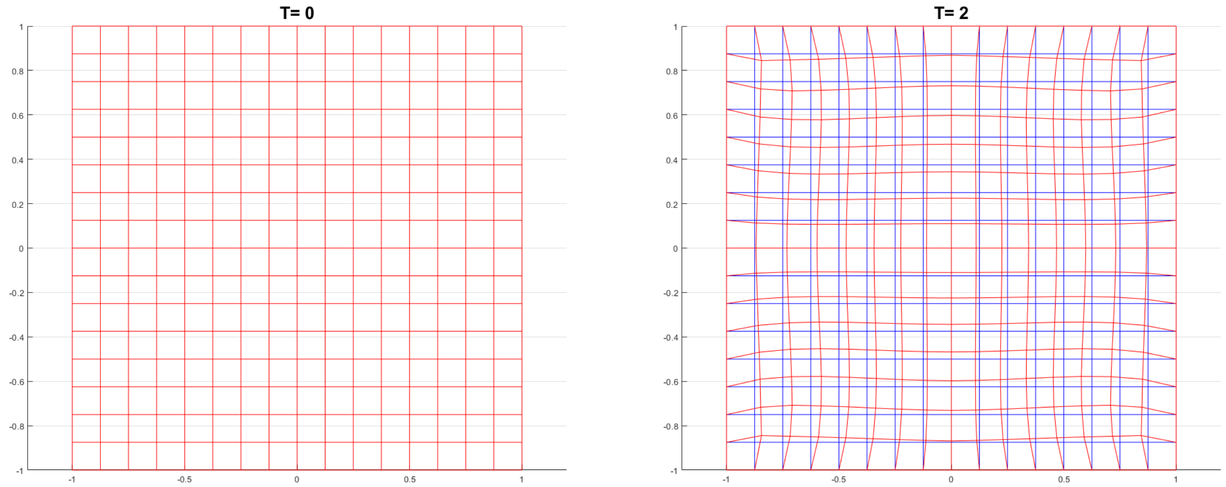
Parameters	Values	Dimensions
E	10	$N/(g\,cm)^{1/2}$
μ_1	10	$N day/cm^2$
μ_2	10	$N day/cm^2$
ρ	1.02	g/cm^3
ν	0.48	-
ζ	0.3	-
τ_c	20	N/cm^3
τ_r	2	N/cm^3
t_{ap}	5	day

Table 7.3.1: Parameters used in section 5.3

In Figures 7.3.1a, 7.3.1b, 7.3.1c, 7.3.1d we can see how the mesh deforms under the body forces acting until $T = 5$ and then gradually relaxes. However, it does not return to its initial state. The symmetry of the body forces is reflected in the symmetry of the solution.

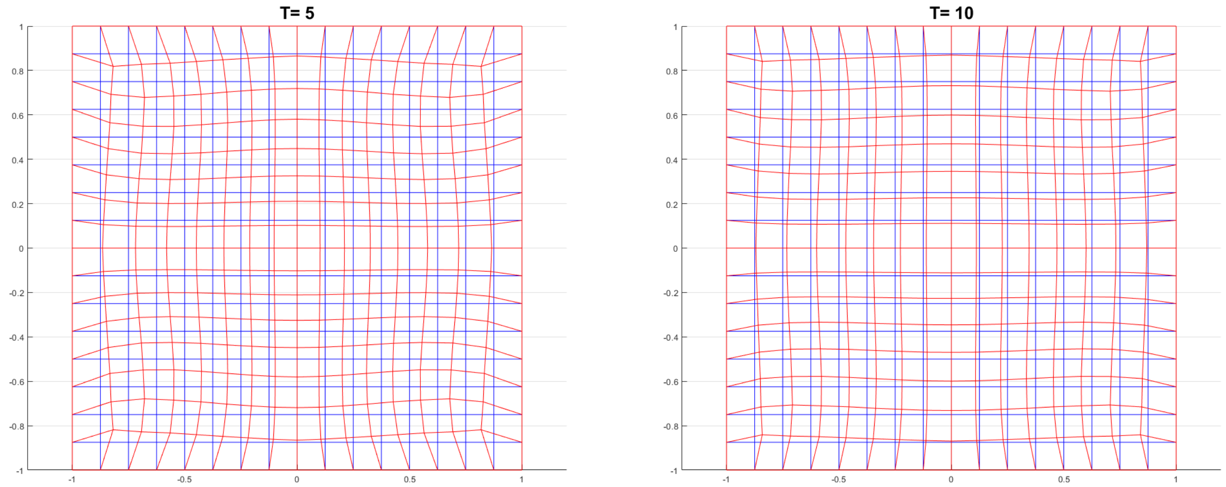
In Figures 7.3.2a, 7.3.2b, 7.3.2c, 7.3.2d we can see the tissue deforming under rotating body forces of the form:

$$\begin{cases} f_1 = \tau_r((y>0)-(y<0))(t<t_{ap}) \\ f_2 = \tau_r((x<0)-(x>0))(t<t_{ap}). \end{cases} \tag{7.3.2}$$



(a) Tissue described in (5.3.1) at time $T = 0$ (red).

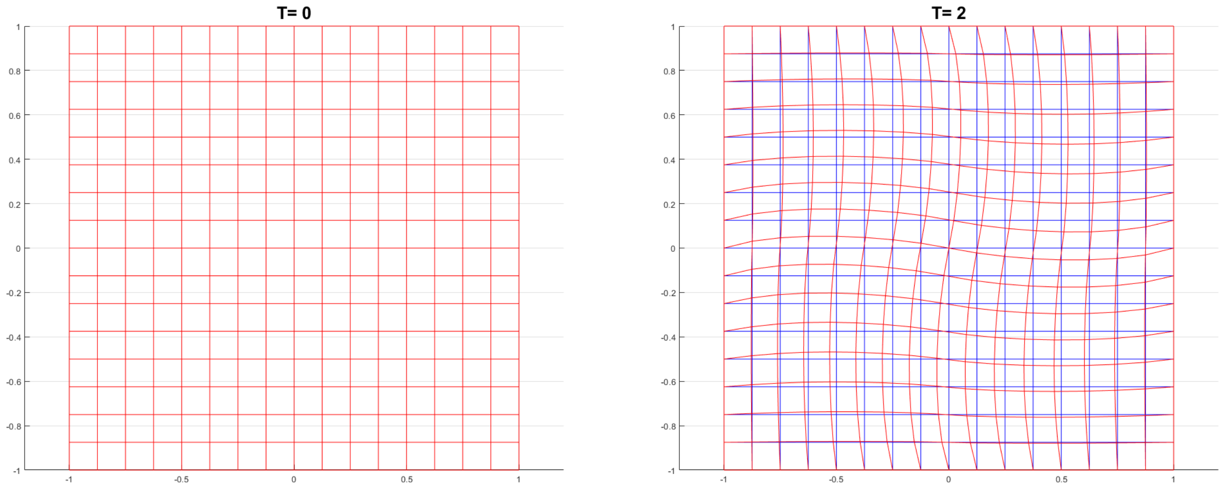
(b) Tissue described in (5.3.1) at time $T = 2$ (red).



(c) Tissue described in (5.3.1) at time $T = 5$ (red).

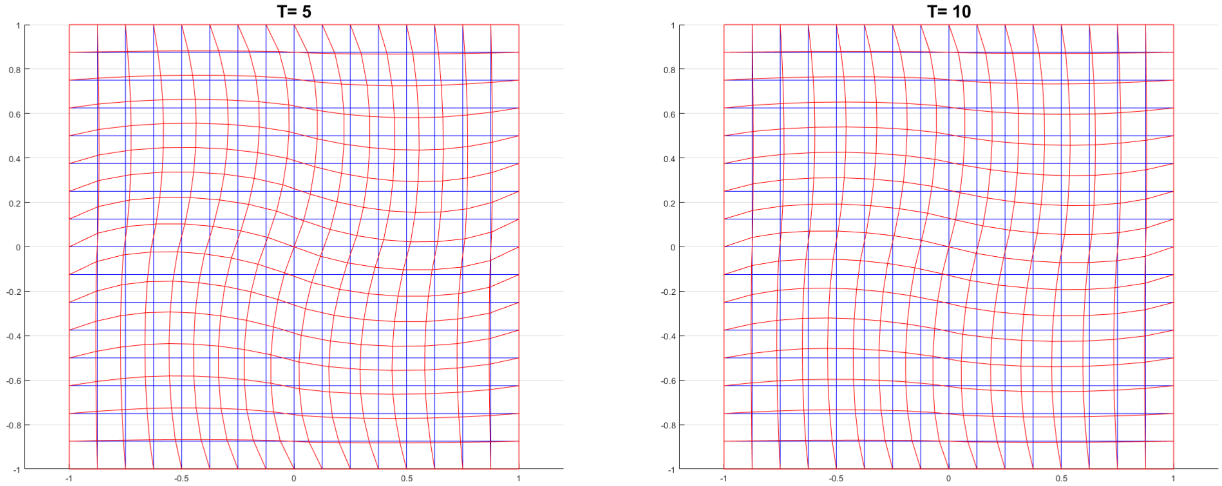
(d) Tissue described in (5.3.1) at time $T = 10$ (red).

Figure 7.3.1: Tissue subject to body forces (7.3.1). Initial state in blue.



(a) Tissue described in (5.3.1) at time $T = 0$ (red).

(b) Tissue described in (5.3.1) at time $T = 2$ (red).



(c) Tissue described in (5.3.1) at time $T = 5$ (red).

(d) Tissue described in (5.3.1) at time $T = 10$ (red).

Figure 7.3.2: Tissue subject to body forces (7.3.2). Initial state in blue.

7.4. EXTENDED MODEL

Aim of this work was to simulate the biochemical interactions within skin wounds after grafting. We will now show the results of the work done so far. The following simulations are performed on the symmetric case, with the modified FCT algorithm as described in Chapter 6, considering a variable time step and with an initial time step of $\Delta t = 0.1$, if not stated otherwise. The parameters used can be found in Table 5.4.1. The considered wound domain is $\mathscr{G} = [-1, 1]^2$. The parameter k of the initial condition was taken $k = 50$.

To gain a better insight in the simulation results we will look at Figures 7.4.3, 7.4.4, 7.4.5, 7.4.6, where all the constituents and mechanical components are displayed for different times for both the uniform mesh with symmetry and for the adaptively refined mesh. In Figures 7.4.4, 7.4.6 we observe how, after skin grafting, the concentration of fibroblasts diminishes, while all the other concentrations increase within the skin graft. This trend continues until 50-80 days after the surgery. Subsequently the concentrations of collagen, myofibroblasts and the signalling molecule steadily decrease until they reach their equilibrium value. The concentration of fibroblasts in the area of the skin graft rises again until it reaches its equilibrium state. For the mechanical components, we can observe how, after an initially rapid increase in the contraction velocity, the wound boundaries slow down. Eventually, the contraction process is reversed and the wound area starts to expand back towards its initial state. The initial state however is never fully reached as can be observed in Figure 7.4.1. It is interesting to note how, after the cell variables have almost reached their equilibrium state and the skin layer is no longer moving, we can see residual strains in the tissue. This implies that the skin layer is subject to residual stresses.

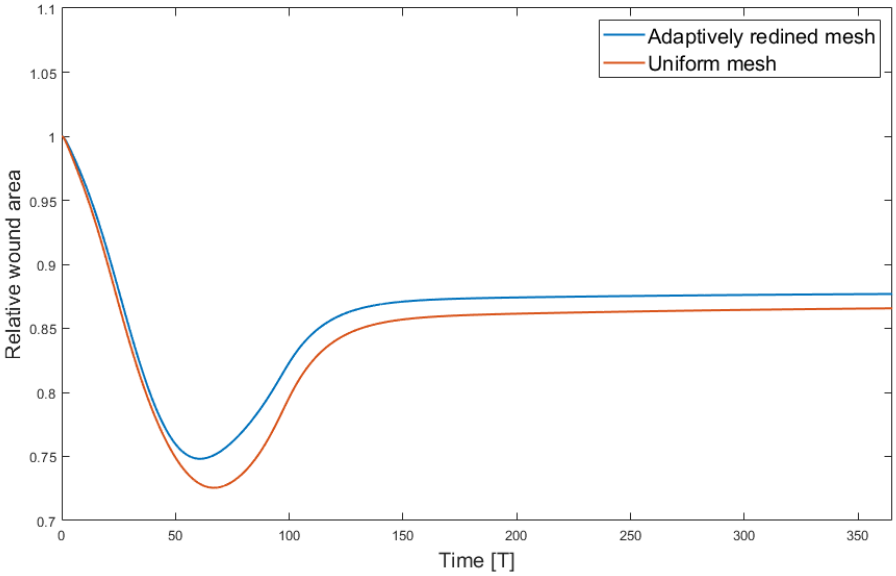


Figure 7.4.1: Evolution in time of the relative wound area using the naive linearization approach.

In Figure 7.4.2 we observe how the time steps changed during the course of the simulation. The plot shows the time step values for both the uniform case with symmetry and for the adaptively refined mesh. We imposed a maximum time step of $\Delta_{max} t = 0.15$ if $T < 150$ and $\Delta_{max} t = 1$ if $T \geq 150$, but it is clear that a higher bound could have been chosen and used successfully. The history of the time steps further confirms our supposition, that the dynamics involved, immediately after peak concentrations are reached for the constituents, are the mechanically most difficult to reproduce (for our algorithm). We see a second decrease in the time step around $T = 140$, which coincides with the vanishing of the signalling molecule c and the myofibroblasts M. Interestingly, only the adaptive mesh suffers from this second time step reduction.

In Chapter 6 we showed an adaptively refined mesh, whose purpose was to better resolve the boundary between wound area and healthy skin. We expect lower peaks and sharper interfaces. We will now list the peak values for each cell constituent for both the refined mesh and the uniform mesh in Table 7.4.1. We see a reduction in the peaks for the concentration of myofibroblasts. In Figure 7.4.1 we see how the wound area evolves in time for both the meshes. We can observe that the effects on the relative wound area are of about 1% in the final time step. It is important to note how better quality of the solution can eventually lead to more

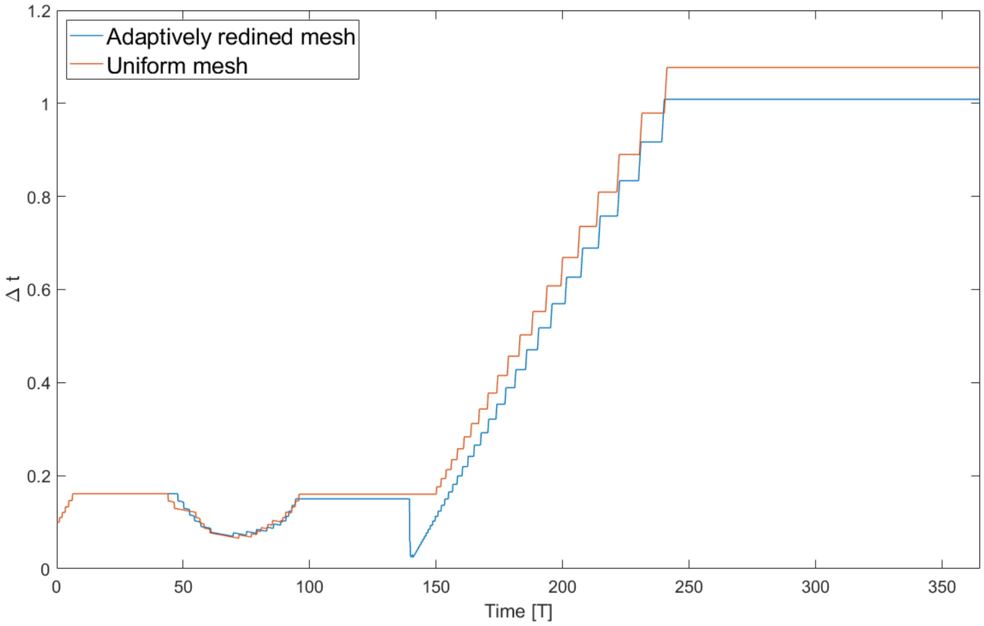


Figure 7.4.2: Time steps used for the simulations.

accurate results and hence a more reliable tool for further medical/experimental applications. The lower peak values for myofibroblasts shown in Table 7.4.1 are likely responsible for this difference in the wound area contraction. The adaptively refined mesh has not been used with the symmetry conditions, as it is not as straightforward to separate the correct (non-)symmetry elements and boundaries as it is for the uniform case. Lastly, the domain of the adaptively refined mesh is $\widetilde{\Omega}_0 = [-5,5]^2$ opposed to $\Omega_0 = [-4,4]^2$ of the uniform case.

Variables	Adaptively refined mesh	Uniform mesh
N	10000	10000
M	8744.2	9218.9
c	8.7362×10^{-8}	8.6601×10^{-8}
ρ	0.8665	0.8555

Table 7.4.1: Comparison of maximum values for the biological variables.

7.5. RICHARDSON'S EXTRAPOLATION

It is of interest to investigate what order of convergence we are able to attain in our calculations. We know that our time step integrations are of second order for the biochemical unknowns and of first order for the mechanical terms. Hence, we expect at least first order convergence in time. Spatial convergence is expected to be of second order in an ideal case. Different techniques can be employed in order to verify our anticipated results.

We define the discretization error E and consider its series expansion in terms of the element size h_k at refinement k:

$$E_k = \| u_{exact} - u_k \| = C_p h_k^p + h.o.t. \quad , \tag{7.5.1}$$

where $\|\cdot\|$ is an appropriate norm (we will use the 2-norm), u_k is the computed solution at refinement k, C_p is a constant and p is the observed order of accuracy. The main assumption is that $h \ll 1$ and hence the higher order terms are negligible.

For simple and commonly known problems it is possible to find an analytical solution. This is of great benefit as it is possible to directly compare the computed results with their analytical counterpart. Hence, it is sufficient to plot the differences and to observe the convergence order from the slope. Similarly, we can calculate two solutions u_1, u_2 with $h_1 < h_2$ and directly compute the observed order of accuracy p using (7.5.1):

$$p = \frac{\log\left(\frac{E_2}{E_1}\right)}{\log\left(\frac{h_2}{h_1}\right)}. \tag{7.5.2}$$

In [30], it is mentioned how the discretization of nonlinearities can adversely affect the order of accuracy. This mainly derives from early termination of the linearization iterations due to computational costs. In order to assure, that the iterations do not influence the above calculation it is recommended to set the iterative error at less than 1% of the discretization error.

When analytical solutions are not present, it is common to resort to the method of manufactured solutions. The idea is to decide a priori what the expression for the solution is and then to apply the considered differential operator on it. This will result in an expression for the source function. The solution to the considered equation with the newly calculated source function is indeed the a priori decided solution. We then can retrace the same steps as with the analytical solution and calculate the order of accuracy through (7.5.2). Whether the analytical solution be present in literature or manufactured, two computed solutions are sufficient to calculate p.

In our case we are not able to use either of the two proposed approaches. We therefore resort to using Richardon's extrapolation. This technique is usually utilized to provide a correction to the fine grid solution, by extrapolating the solutions from two different mesh grids onto an ideal mesh with infinitesimal mesh size. However, it is possible to adapt this concept to the calculation of the observed order of accuracy. Let us suppose a constant refinement factor r of our mesh, i.e. $r = h_{k+1}/h_k$ with $r > 1, \forall k$. Let us consider three meshes (fine mesh h_1, medium mesh h_2, coarse mesh h_3) and the calculated solutions on them with a p-th order accurate discretization:

$$\begin{aligned} u_1 &= u_{\text{exact}} + C_p h^p + \mathrm{O}\left(h^{p+1}\right) \\ u_2 &= u_{\text{exact}} + C_p (rh)^p + \mathrm{O}\left([rh]^{p+1}\right) \\ u_3 &= u_{\text{exact}} + C_p \left(r^2 h\right)^p + \mathrm{O}\left(\left[r^2 h\right]^{p+1}\right). \end{aligned}$$

Neglecting the high order terms, we can use the above three expression to solve for the order of accuracy p by

$$p = \frac{\log\left(\frac{\|u_3 - u_2\|}{\|u_2 - u_1\|}\right)}{\log(r)}.$$

We are thus able to calculate the observed order of accuracy p of both the time and space discretization by refining one and keeping the other constant.

The advantage of this approach is that we can compute an order of accuracy without any theoretical background and it is applicable to potentially any discretization scheme. Furthermore, it does not depend on the quantity of interest or on the choice of the norm. On the other hand, it requires at least three simulation runs at varying (spatial or temporal) step size, which can be quite expensive computationally. Furthermore, this extrapolation technique can potentially exacerbate negative effects from the iterative convergence error and is meant to be used for smooth solutions [30].

When comparing solutions on different meshes we interpolate the coarse solution using cubic functions onto the finer mesh. For this purpose we take advantage of the MATLAB function *griddata*. We are using bilinear elements, hence no additional error (which influences the order of accuracy) should be introduced by this. The rigorous procedure would require either a standard Lagrangian interpolation or an L^2 projection.

To better describe the standard Lagrangian interpolation and the L^2 projection, let us first consider two finite element partitions $\mathcal{Q}_{h,1}$, $\mathcal{Q}_{h,2}$ of the same domain Ω. The corresponding function spaces are $V_{h,1}$ and $V_{h,2}$. If we denote with $u_{h,1} \in V_{h,1}$ the discrete solution on $\mathcal{Q}_{h,1}$, then our objective is to construct an approximation $u_{h,2} \in V_{h,2}$ to $u_{h,1}$. Standard Lagrangian interpolation consists in obtaining the nodal values of $u_{h,2}$ through simple interpolation. Let U_n be the vector of nodal values of $u_{h,n}$ and U_n^a its value for the node a, then we have that

$$U_2^a = \sum_b U_1^b N_1^b(\boldsymbol{x}_2^a),$$

where N_1^b is the shape function corresponding to the node b in $\mathcal{Q}_{h,1}$ and $\boldsymbol{x}_2^a$ are the coordinates of a. To perform this computation one must first find the element $K_1 \in \mathcal{Q}_{h,1}$ where $\boldsymbol{x}_2^a$ lies for each a. This node search can result in expensive computations. The L^2 projection is equivalent to the minimization of the functional $J(v_{h,2}) = \|v_{h,2} - u_{h,1}\|^2_{L^2(\Omega)}$ with $v_{h,2} \in V_{h,2}$. This results in finding $u_{h,2} \in V_{h,2}$ such that

$$\int_\Omega u_{h,2} v_{h,2}\, d\Omega = \int_\Omega u_{h,1} v_{h,2}\, d\Omega \quad \forall v_{h,2} \in V_{h,2}.$$

When constructing the elements matrices for this system, we encounter again the difficulty of evaluating the basis functions of $V_{h,1}$ on nodes of $\mathcal{Q}_{h,2}$. We refer the interested reader to [31] for a more detailed discussion about the topic and its expansion to interpolations with restrictions.

We compute the different solutions using the symmetric model, as it allows to refine our mesh (two times!) while still maintaining reasonable computational costs. In fact, the standard number of elements we used in our uniform mesh is 20 by 20, which, if taken as coarsest baseline, would result in 80 by 80 for the finest stage when considering a refinement ratio of two. Instead we will start with a baseline of 20 by 20 elements and refine up to 45 by 45 elements with a ratio of 1.5, while using $\Delta t = 0.05$ as a time step to keep the time dependence of the error small. Regarding the accuracy with respect to time we will always consider a 20 by 20 mesh and a baseline time step equal to 0.2 and refine using a ratio of 2. The values for the time step refinement are shown in Table 7.5.1. When refining in space, we were initially not successful in the application of Richardson's extrapolation. The values for p fluctuated a lot between different solution variables. We therefore changed the parameter k of the initial condition to $k = 3$ in order to have a more smooth solution. In Table 7.5.2 we can observe the calculated values of p. For the variables N and c we notice a discordant value of around 3. This is probably due to the fact that their profile is very smooth and thus they benefit of the cubic interpolation.

Variables	p (for space)
N	3.1044
M	1.8897
c	3.2445
ρ	1.8585
ε_{11}	1.854
ε_{22}	1.854
ε_{12}	1.7894
v_1	1.9264
v_2	1.9264

Table 7.5.1: Values resulting from Richardson extrapolation in space.

Variables	p (for time)
N	0.9975
M	0.944
c	0.9897
ρ	0.9874
ε_{11}	0.9536
ε_{22}	0.9536
ε_{12}	0.9433
v_1	1.0043
v_2	1.0043

Table 7.5.2: Values resulting from Richardson extrapolation in time.

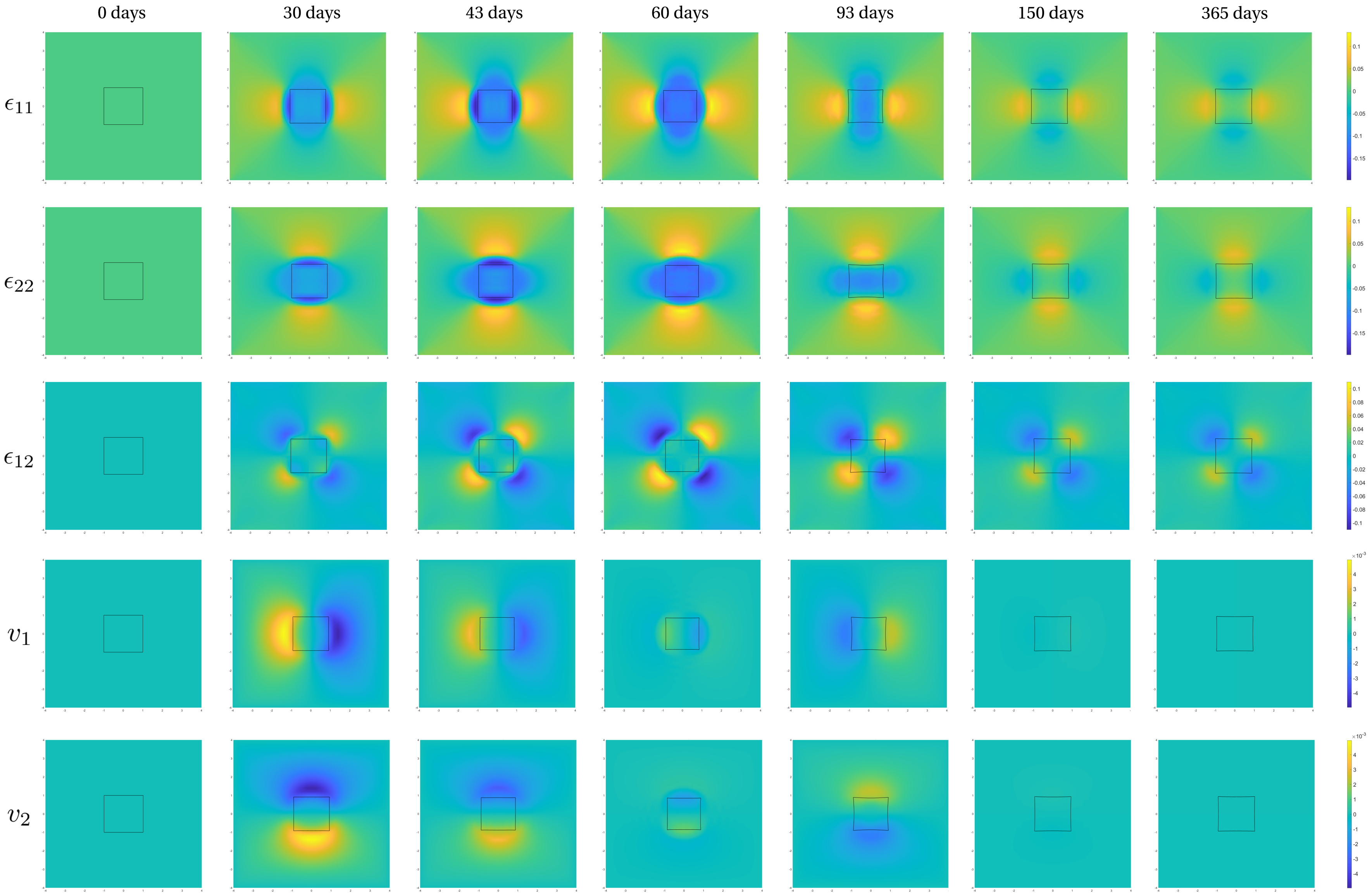


Figure 7.4.3: Plots of the mechanical variables at different time steps for the uniform mesh. In the first three rows we can observe the evolution in time of the strains. We omit ε_{21} since $\varepsilon_{21} = \varepsilon_{12}$. The last two rows show the evolution of the velocities. The parameters used can be found in Table 5.4.1.

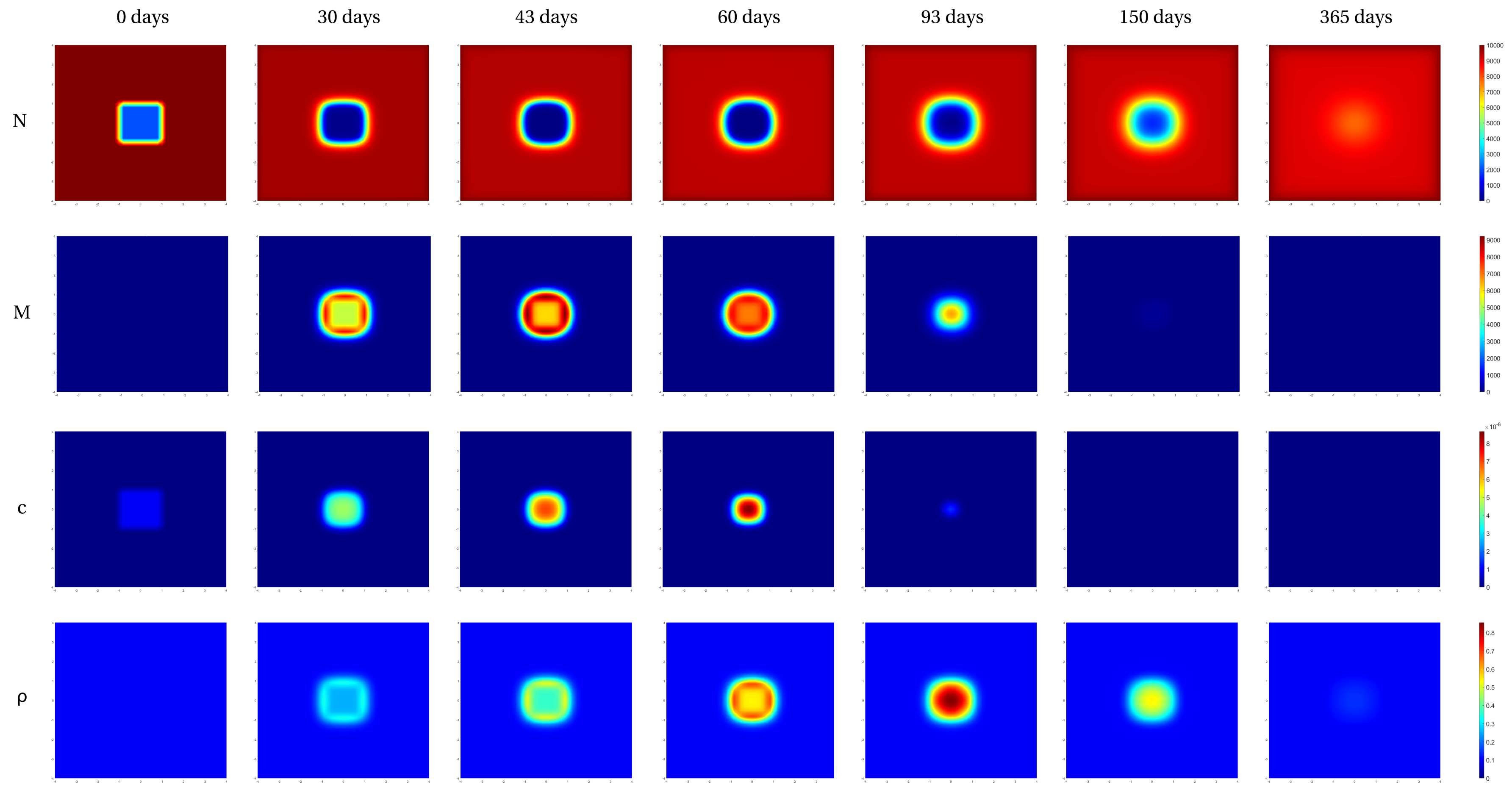


Figure 7.4.4: Plots of the biological variables at different time steps for the uniform mesh. The parameters used can be found in Table 5.4.1.

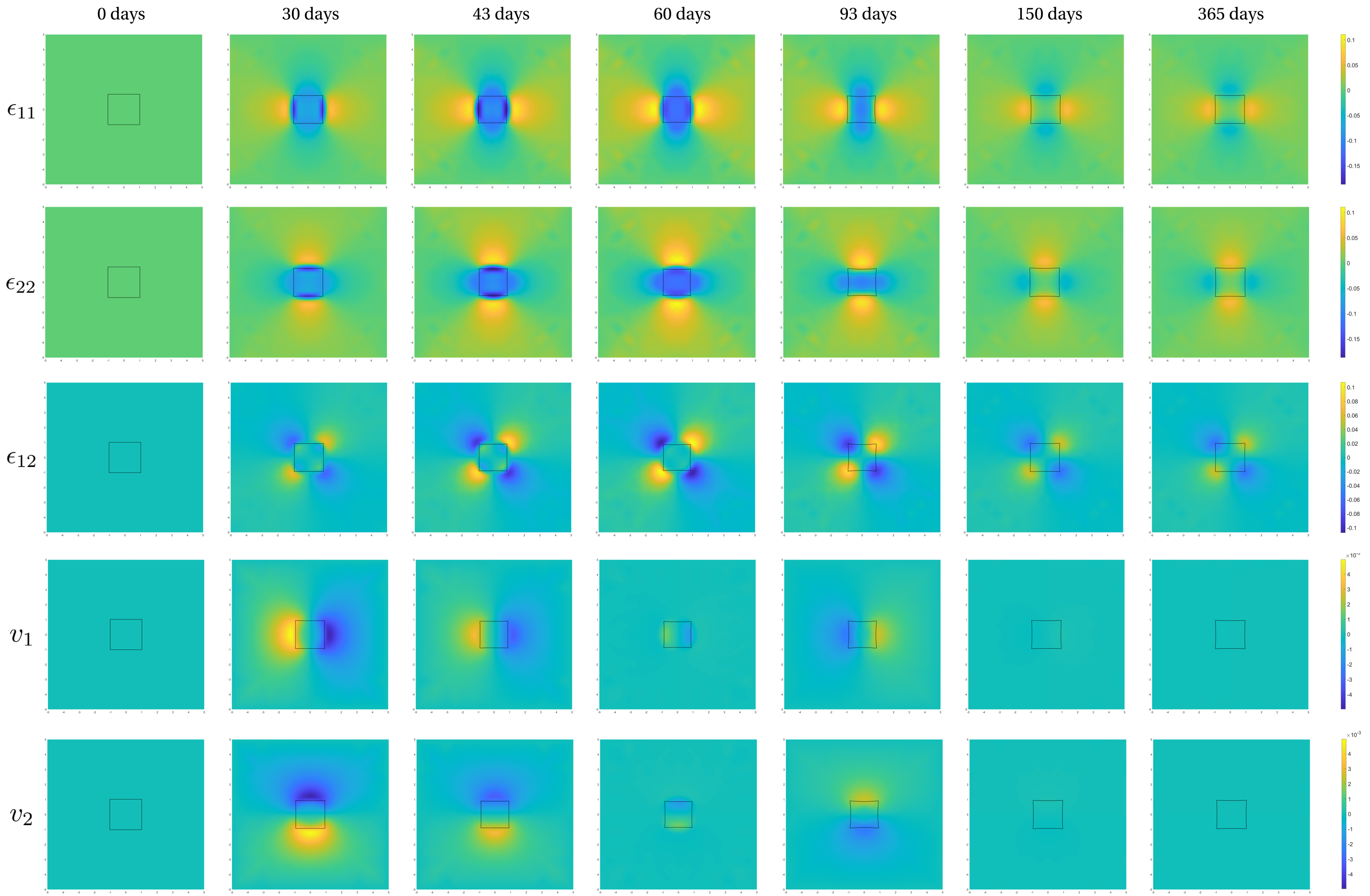


Figure 7.4.5: Plots of the mechanical variables at different time steps for the adaptively refined mesh. In the first three rows we can observe the evolution in time of the strains. We omit ε_{21} since $\varepsilon_{21} = \varepsilon_{12}$. The parameters used can be found in Table 5.4.1.

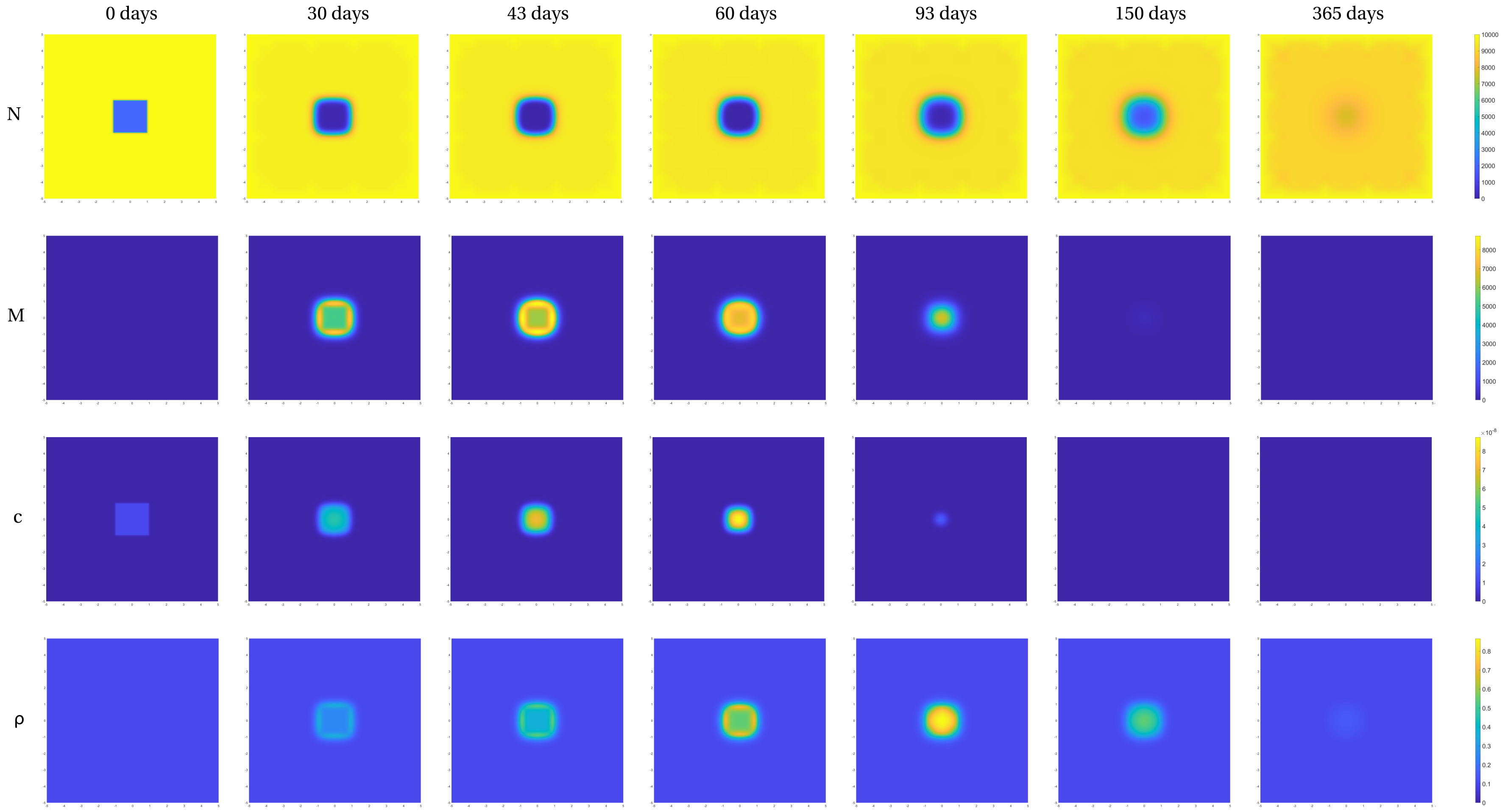


Figure 7.4.6: Plots of the biological variables at different time steps for the adaptively refined mesh. The parameters used can be found in Table 5.4.1.

II

PART II

8

INTRODUCTION TO IGA

In the second part of this work we will look at the implementation of the mechanical equations within the framework of isogeometric analysis (IGA). IGA is a relatively new addition to the ranks of the Galerkin methods and it has many things in common with traditional FE. It is strongly connected to the concept of computer aided design (CAD) as it takes non-uniform rational B-splines (NURBS) as basis functions. NURBS are widely used in CAD to accurately describe the geometry boundaries of the object in question.

One of the big barriers between engineering design and analysis today is the incompatibility between the geometric model derived from CAD and the analysis model used to perform the simulations. This hinders the development of automated processes which involve both design and analysis, as it is always necessary to translate the models, see Figure 8.0.1. Isogeometric analysis takes it upon itself to overcome this issue by utilizing the same functions for both the geometric modelling and the analysis, see Figure 8.0.2. Opposed to traditional FE, where the analysis dictates the functions used for the geometry approximation, in IGA the geometry functions are used for the analysis, thus preserving an accurate description of the computational domain. The most widely used geometrical technology for engineering design are non-uniform rational B-splines (NURBS). We can thus bypass the accuracy problems inherited from the geometry approximation. For instance, in computational fluid dynamics compressible Euler solvers were suffering from spurious oscillations in the 1980s and 1990s. The root of the cause was the piecewise linear approximation of the geometry, while a smooth geometry completely eliminated the entropy layers [32]. We will now proceed with a brief description of the concepts of IGA following one of its most relevant pieces of literature [33].

The basic construction unit of NURBS are B-splines. In contrast to traditional FE basis functions, the parameter space of B-splines is related to patches rather than elements. Patches describe (sub)domains where element types and material models are the same. As a result, simple domains can often be described by a single patch. The parameter space of traditional FE, also called the reference element, has a unique mapping onto each element in the domain. On the other hand, in IGA each patch of multiple elements has a global mapping onto the physical space. Figure 8.0.3 illustrates this difference between traditional FE and IGA. We will proceed with a brief description of B-splines and their implications in IGA.

8.1. KNOT VECTORS AND BASIS FUNCTIONS

The parameter space of B-splines is divided in elements and such partition is defined with the help of knot vectors. A knot vector Ξ is a sequence of non-decreasing values in the parameter space:

$$\Xi = \left[\xi_1, \xi_2, \cdots, \xi_{n+p+1}\right],$$

where $\xi_i \in \mathbb{R}$ is the i-th knot, the knot index $i = 1, ..., n+p+1$, n is the number of basis functions used to construct the B-spline and p is their polynomial order. The knots can be equally distributed in the parameter space, i.e. a uniform knot vector, or unequally spaced, i.e. a non-uniform knot vector. Knot values can be repeated and their multiplicity has important implications regarding the properties of the basis. An knot vector is called open, if the first and last knot are repeated $p+1$ times. Open knot vectors in one dimension generate interpolatory basis functions at the boundaries of the parameter space. Hence, in multiple dimensions they

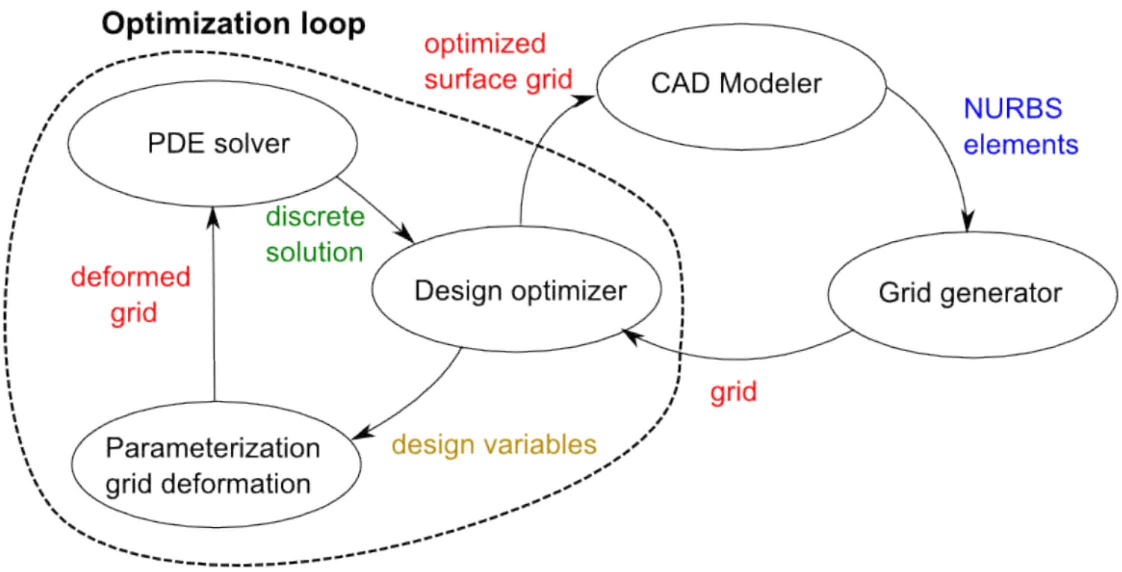


Figure 8.0.1: Design cycle based on traditional finite elements. Taken from [34].

will be interpolatory on the corners but not on the edges. This is a distinct difference between the concept of knots in IGA and nodes in traditional finite element analysis.

Knot vectors are the 'ingredients' of B-splines. We will now look at their 'recipe'. With a knot vector Ξ we define a B-spline's basis functions recursively. The zeroth order basis function ($p = 0$) reads:

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases}$$

For $p = 1, 2, \ldots$ we have

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi).$$

The above expression is referred to as the Cox-de Boor recursion formula. Note that, the ratio of $0/0$ is defined to be zero in these steps. In Figure 8.1.1 we can see how for $p = 0$ and $p = 1$ the constructed basis functions are exactly the standard piecewise constant and linear finite element functions. However, this does not hold true for the quadratic case where the constructed basis functions are all shifted versions of the same curve. In FEA the shape of the quadratic basis function depends on whether it is on an internal node or not.

Some particular features of B-spline basis functions are particularly noteworthy. First, the basis forms a partition of unity:

$$\sum_{i=1}^{n} N_{i,p}(\xi) = 1 \quad \forall \xi.$$

Second, each basis function is non-negative on the whole domain. Hence, all entries of the mass matrix will also be non-negative, which has implications for developing lumped mass schemes [33]. Furthermore, p-th order basis functions have $p-1$ continuous derivatives across the border of the element, i.e. on the knots. In general, p-th order basis functions have $p - m_i$ continuous derivatives over the knot ξ_i, where m_i is the multiplicity of ξ_i. As a result, if the multiplicity of the knot is equal to p then the basis function is interpolatory at that knot. In Figure 8.1.2 we can see examples for B-spline basis functions and traditional Lagrangian basis functions.

Another convenient property of the basis functions is the computation of their derivatives. In fact, as one can guess from their construction, also the derivatives can be defined recursively. We have that

$$\frac{d}{d\xi} N_{i,p}(\xi) = \frac{p}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) - \frac{p}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi).$$

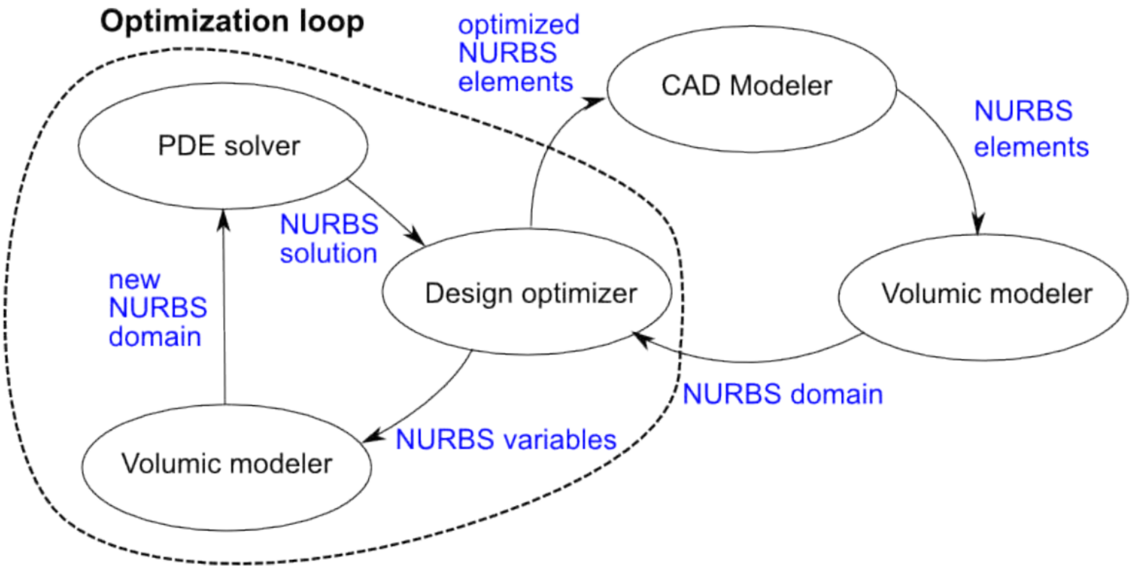
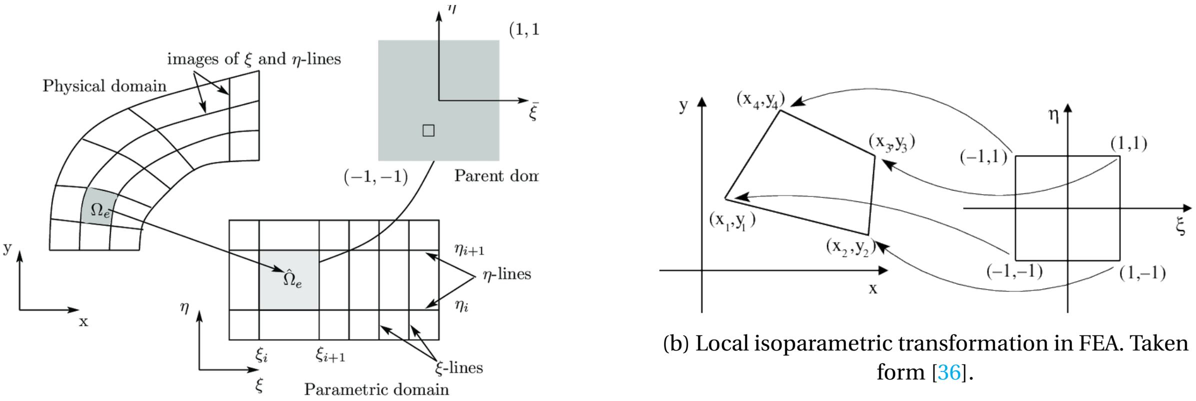


Figure 8.0.2: Design cycle based on IGA. Taken from [34].



(a) Global and local transformations in IGA. Taken form [35].

(b) Local isoparametric transformation in FEA. Taken form [36].

Figure 8.0.3: Comparison of mappings.

This can be extended to represent the k-th derivative of $N_{i,p}$ purely in terms of $N_{i,p-k}, \ldots, N_{i+k,p-k}$:

$$\frac{d^k}{d^k\xi} N_{i,p}(\xi) = \frac{p!}{(p-k)!} \sum_{j=0}^{k} \alpha_{k,j} N_{i+j,p-k}(\xi)$$

with

$$\begin{aligned}
\alpha_{0,0} &= 1 \\
\alpha_{k,0} &= \frac{\alpha_{k-1,0}}{\xi_{i+p-k+1} - \xi_i} \\
\alpha_{k,j} &= \frac{\alpha_{k-1,j} - \alpha_{k-1,j-1}}{\xi_{i+p+j-k+1} - \xi_{i+j}} \qquad j = 1, \ldots, k-1, \\
\alpha_{k,k} &= \frac{-\alpha_{k-1,k-1}}{\xi_{i+p+1} - \xi_{i+k}}.
\end{aligned}$$

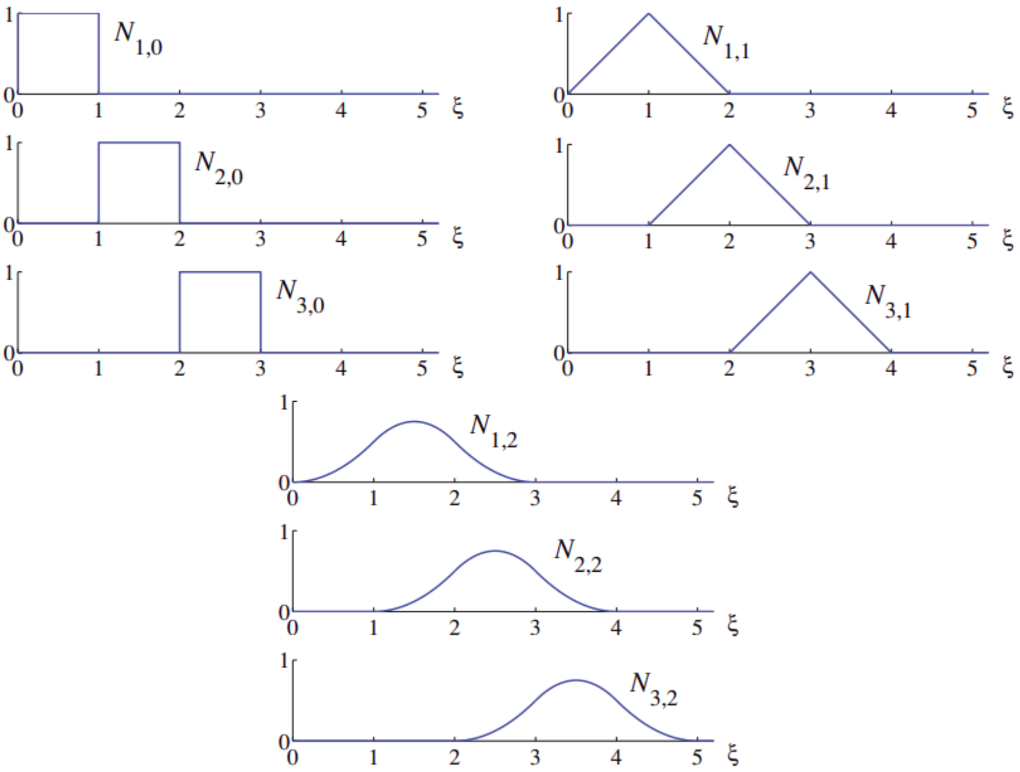


Figure 8.1.1: Basis functions of order $p = 0, 1, 2$ for knot vector $\Xi = \{0, 1, 2, 3, 4, 5, \ldots\}$. Taken from [33].

8.2. B-SPLINES

To construct a B-spline curve in $\mathbb{R}^d$ we take, just as in finite elements analysis, a linear combination of B-spline basis functions. Instead of the nodal values in finite element analysis, we call the vector-valued coefficients control points. In fact, the non-interpolatory nature of the basis makes it difficult to have a clear geometrical interpretation of the control points. Given n basis functions $N_{i,p}$ of order p and the corresponding control points $\boldsymbol{B}_i \in \mathbb{R}^d$ a piecewise-polynomial B-spline curve is given by

$$\boldsymbol{C}(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\boldsymbol{B}_i.$$

The control polygon is the polygon resulting from linear interpolation of the control points. Many properties of B-spline curves derive directly from the properties of their basis function. An example for this is that B-splines of order p have $p-1$ continuous derivatives in absence of repeated knots or control points. Furthermore, given the local support of the basis functions, moving one control point will affect the B-spline curve in at most $p+1$ elements of the curve. Another interesting property is that B-spline curves of order p are contained inside the union of convex hulls of $p+1$ successive control points, see Figure 8.2.1. B-spline curves also possess a variational diminishing property, which makes them much more robust, i.e. less prone to oscillations, than Lagrange polynomials. This is a reason why Lagrange polynomials are so uncommon in geometrical design software. Another interesting property of B-spline curves is referred to as affine covariance. This property consists in the fact that in order to apply an affine transformation to the curve, it is sufficient to apply this transformation to the control points.

For the construction of B-spline surfaces we consider a control net $\{\boldsymbol{B}_{i,j}\}$, $i = 1, \ldots, n$, $j = 1, \ldots, m$, polynomial orders p, q and knot vectors $\Xi = \{\xi_1, \xi_2, \ldots, \xi_n\}$ and $\mathcal{H} = \{\eta_1, \eta_2, \ldots, \eta_m\}$. Then, a tensor product B-spline surface is defined by

$$\boldsymbol{S}(\xi, \eta) = \sum_{i=1}^{n}\sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta)\boldsymbol{B}_{i,j},$$

where $N_{i,p}(\xi), M_{j,q}(\eta)$ are univariate B-spline basis functions of order p and q corresponding to the knot vectors Ξ and $\mathscr{H}$, respectively. The properties of affine covariance and inclusion in the convex hull hold true also for B-spline surfaces.

8.3. REFINEMENT

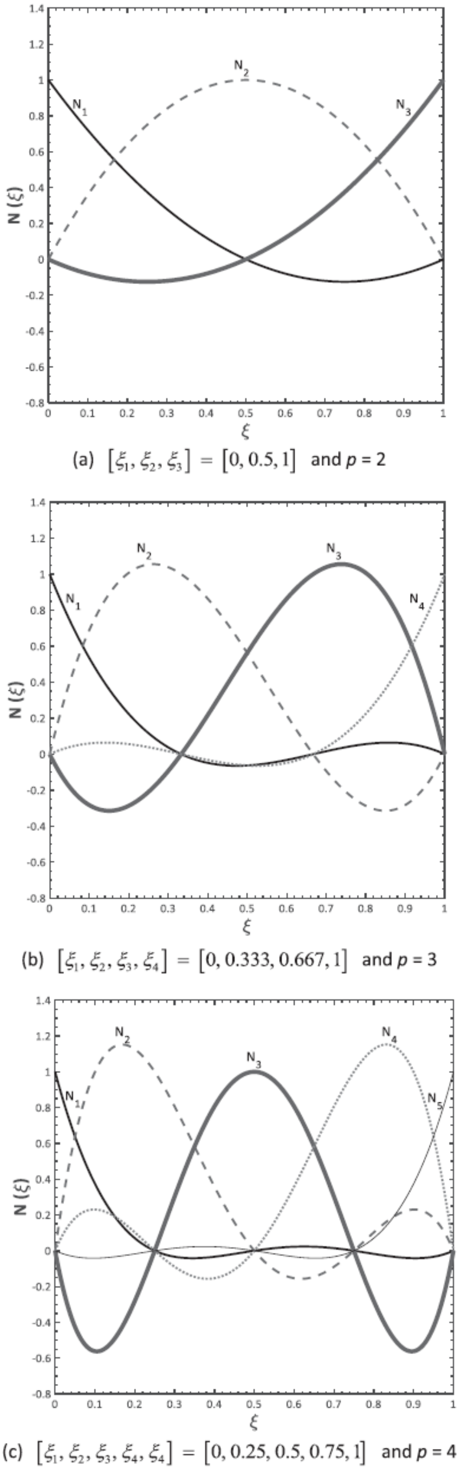
In FEA the traditional refinement techniques are: element size reduction and order elevation of the basis functions. B-splines enable the use of continuity control as a tool for refinement, both in the interiors as across the edges of the elements. We will now describe the three refinement approaches used in IGA.

First, we will talk about knot insertion. This technique allows us to modify the basis functions without modifying the geometry or parametrisation. Given the original knot vector Ξ, we consider a new knot vector $\tilde{\Xi}$ such that $\Xi \subset \tilde{\Xi}$. The new control points can be retrieved by linear combination of the original ones. Knot insertion can result in continuity reduction across borders, if an existing knot is repeated, or it can add a new basis function, if a new knot is inserted. While the former case does not have any counterpart in traditional FE, the latter behaves similarly to classical h-refinement, or element size refinement, in finite element analysis (if each new knot value is added with multiplicity $m_i = p$ it would be equivalent to h-refinement [33]). However, with the addition of one new knot only one new element and one new basis function are created, which is not true for the addition of one new node in traditional FE.

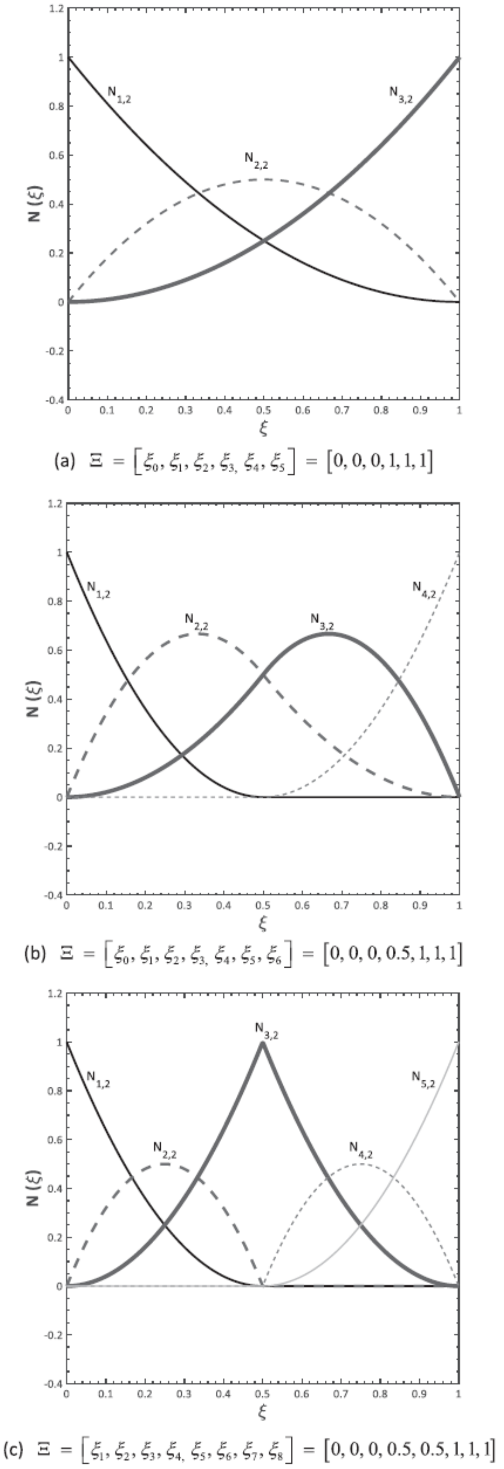
Second, a simple means to basis enrichment is order elevation. In IGA it is immediate to elevate all existing basis functions by one order: simply increase the multiplicity of the each knot by one. By doing this, the original continuity across borders is preserved. In fact, we recall that at each element boundary the basis will have $p - m_i$ continuous derivatives, where p is the order of the basis function and m_i its multiplicity. If we increase both by one, we preserve the discontinuities. No new knots are inserted during order elevation.

Last, a combination of the above can be used in order to elevate both order and continuity. This type of approach is often called k-refinement. It stems from the property that knot insertion and order elevation do not commute. If we add a unique knot $\hat{\xi}$ between two distinct knot values in a curve of order p, this will imply that all basis functions have $p-1$ continuous derivatives across $\hat{\xi}$. Subsequently, we elevate the order, which will preserve the existing discontinuities. This results in the basis functions still having $p-1$ continuous derivatives across $\hat{\xi}$, while the order has been increased to q. On the contrary, if we first elevate the original curve to order q and then insert $\hat{\xi}$, then the basis functions will have $q-1$ continuous derivatives across $\hat{\xi}$. This second process is referred to as k-refinement.

We have now the tools to refine purely in space, raise the order of our basis functions, while maintaining the minimum continuity, and increase both order and minimum continuity. The combination of h-, p- and k-refinement allows for a multitude of different refinement strategies, which are unheard of in traditional FE. All this comes as an addition to a set of basis functions, which already has improved continuity properties compared to traditional Lagrangian basis functions.



(a) Lagrange basis functions. Taken form [37].



(b) Quadratic B-spline basis functions. Taken form [37].

Figure 8.1.2: Comparison of basis functions.

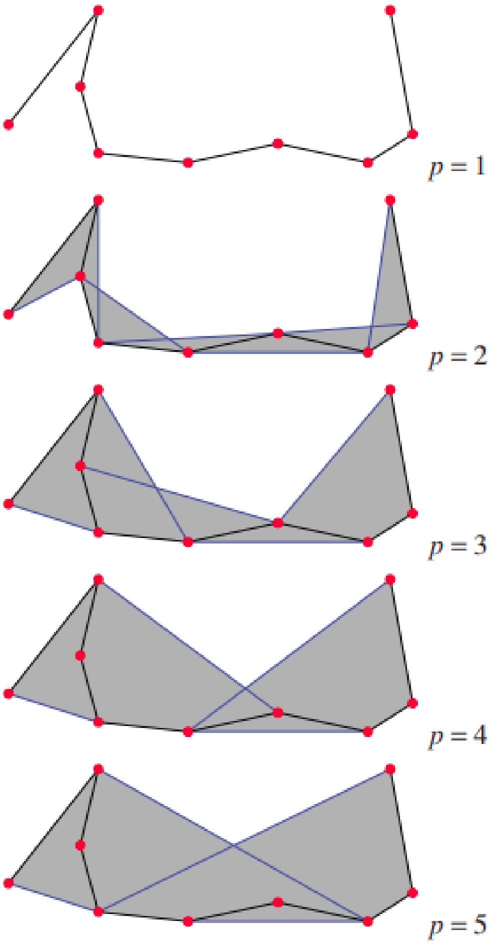


Figure 8.2.1: Convex hulls for $p = 1$ through $p = 5$. Taken from [33].

9

RESULTS

In this chapter we want to investigate the possibility to reproduce the results from Chapter 7. To do this we used G+Smo, a C++ library for isogeometric analysis. The calculations and matrix assembly are completely taken over by the G+Smo library. We were able to reproduce the results for the mechanical equations and only partially for the complete model. For the discretization only B-splines have been used. To accurately replicate the results from our traditional finite element model, we used first order B-splines, which are in fact equivalent to bilinear elements. It is however trivial to increase the order of the basis without the need of rewriting any part of the code.

It is common in FEM to assign the initial condition by simply evaluating the function at the nodes in the mesh and then using that information to initialize the first time step. In Chapter 8 we mentioned the difficulty to geometrically interpret the meaning of control points. In fact, when initializing a solution at $t = 0$, it is not a viable option to simply assign the function value to the control points. We thus have to resort to the method of L^2 projection. Projecting in L^2 entails, that given the initial condition $f \in L^2$, we look for a function $u_{init} \in V_h \subset L^2$ which satisfies:

$$\int_\Omega u_{init}\varphi \, d\Omega = \int_\Omega f\varphi \, d\Omega \quad \forall \varphi \in L^2(\Omega),$$

where V_h is the function space spanned by our basis. Strictly speaking, we are looking for a function $u_h \in V_h$ that minimizes the the functional $J(u_h) = \|u_h - f\|^2_{L^2(\Omega)}$. This results in the system $M\boldsymbol{u}_{init} = \boldsymbol{b}$, where

$$M_e = \int_e \varphi_i \varphi_j \, d\Omega \quad i, j = 1, \dots, 4,$$
$$\boldsymbol{b}_e = \int_e f\varphi_i \, d\Omega \quad i = 1, \dots, 4,$$

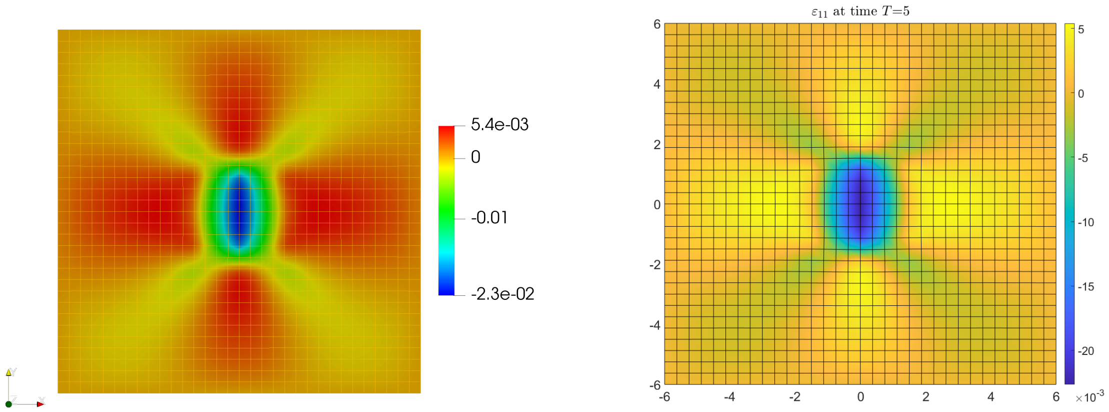
and $\boldsymbol{u}_{init}$ is the control point vector corresponding to the function u_{init}.

When constructing the necessary matrices for our linear system, G+Smo takes advantage of its ‘ExpressionAssembler’ routine. The main idea is to simply write the basis functions and other variables involved in the construction of the element matrix and the routine takes care automatically of the assembly of the global matrix. While being extremely intuitive, once you get to gist of it, it makes it quite complicated for the inexperienced user to have full control of the element matrix calculations. This will be of relevance when considering the nonlinear functions, as they will be directly evaluated on the Gaussian integration points. In the MATLAB code we first evaluated the nodal values of the functions instead and then interpolated onto the Gaussian points.

9.1. MECHANICAL MODELS

We will now show the results for the mechanical models reproduced with G+Smo.

In Figure 9.1.1 we can observe the solution of ε_{11} for the viscoelastic equations with the body forces described (7.2.2). The chosen time step was $\Delta t = 0.1$ and we used 32×32 elements. The same maximum and minimum values as in Figure 7.2.3a have been attained. Figure 9.1.2 shows the difference in the solution when using first bilinear and then quadratic B-splines with 16×16 elements.

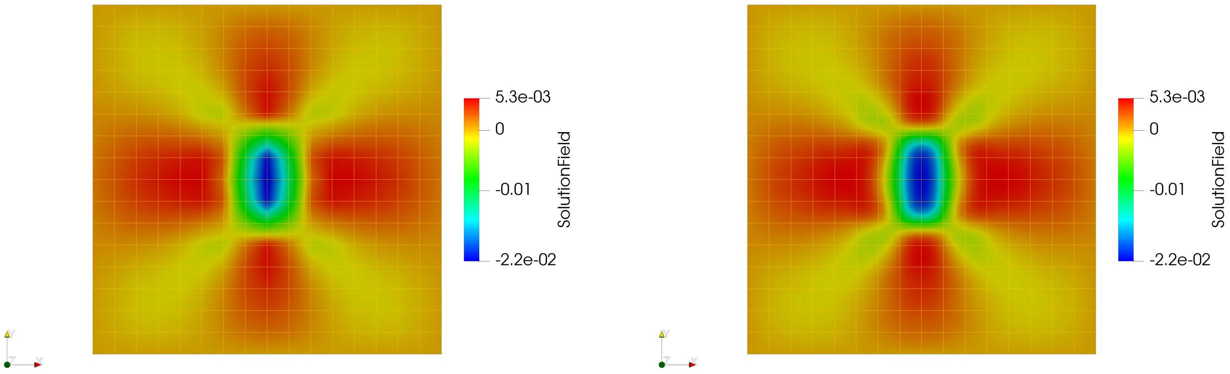


(a) Solution of viscoelastic model using G+Smo with bilinear basis functions.

(b) Solution of viscoelastic model using MATLAB with bilinear basis functions.

Figure 9.1.1: Comparison of results between G+Smo and MATLAB with bilinear basis functions. The difference in the colourmaps derives from the fact that we used Paraview for the G+Smo results.

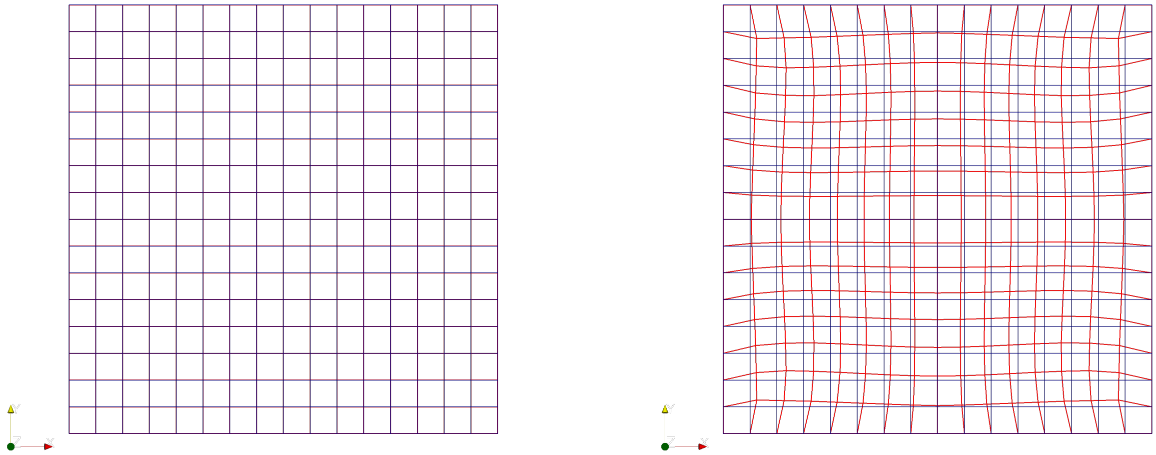
The solution to the morphoelastic equations is shown in Figure 9.1.3. The contracting forces (7.3.1) have been used. We performed the computations with the time step $\Delta t = 0.2$ and 16×16 elements.



(a) Solution of viscoelastic model using IGA with bilinear basis functions.

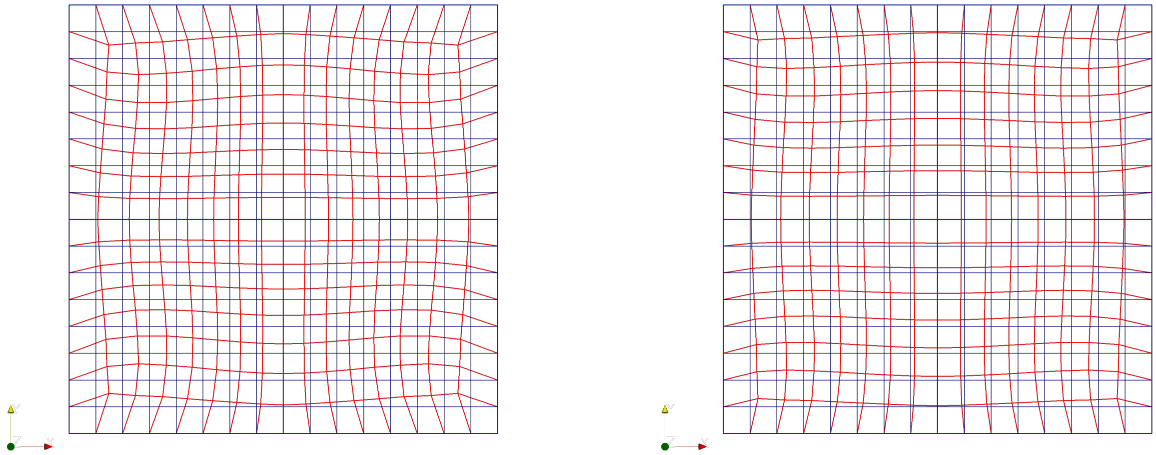
(b) Solution of viscoelastic model using IGA with quadratic basis functions.

Figure 9.1.2: Comparison of results with different basis functions (16 × 16 elements).



(a) Tissue described in (5.3.1) at time $T = 0$ (red).

(b) Tissue described in (5.3.1) at time $T = 2$ (red).



(c) Tissue described in (5.3.1) at time $T = 5$ (red).

(d) Tissue described in (5.3.1) at time $T = 10$ (red).

Figure 9.1.3: IGA solution of tissue subject to body forces (7.3.1). Initial state in blue.

9.2. EXTENDED MODEL

In order to linearize and impose proper boundary conditions in G+Smo, some modifications to the previously described algorithms have been made. Our inexperience with the G+Smo library is the culprit and we had to resort to some ‘tricks’.

The first modification pertains to the linearization of the reaction terms for N and M. We recall their linearization in the naive approach:

$$R_N \overset{Picard}{\approx} R_N^* = r_F \left[1 + \frac{r_F^{\max} c^*}{a_c^I + c^*}\right] \left[1 - \kappa_F F^*\right] N^{*q} N - k_F c^* N - \delta_N N;$$

$$R_M \overset{Picard}{\approx} R_M^* = r_F \left\{ \frac{\left[1 + r_F^{\max}\right] c^*}{a_c^I + c^*} \right\} \left[1 - \kappa_F F^*\right] M^{*q} M + k_F c^* N^* - \delta_M M.$$

As specified in Section 5.4.1, in the FE code we set N^q and M^q to zero if the value of the respective variable is zero. Since we were not able to do the same in G+Smo, we resorted to using a different linearization approach:

$$R_N \overset{Picard}{\approx} R_N^* = r_F \left[1 + \frac{r_F^{\max} c^*}{a_c^I + c^*}\right] \left[1 - \kappa_F (N + M^*)\right] N^{*1+q} - k_F c^* N - \delta_N N;$$

$$R_M \overset{Picard}{\approx} R_M^* = r_F \left\{ \frac{\left[1 + r_F^{\max}\right] c^*}{a_c^I + c^*} \right\} \left[1 - \kappa_F (N^* + M)\right] M^{*1+q} + k_F c^* N^* - \delta_M M,$$

where we used the fact that $F = N + M$.

The second difference between the IGA code and the traditional FE code consists in the enforcement of nonhomogeneous Dirichlet boundary conditions. While in the MATLAB code we computed the global matrix entries also for Dirichlet nodes, G+Smo reduces the matrix system by eliminating the Dirichlet boundary points. G+Smo ‘recognizes’ boundaries and their conditions via the ‘addCondition’ command. Again due to our inexperience, this resulted in the boundary conditions being indeed enforced on the boundary, but the solution hinted at the fact that the boundaries were considered to be homogeneous in the computations. We are however not able to verify this claim. To circumvent this problem, we resorted to not set the Dirichlet boundary conditions and enforce them weakly. In [33] we can find a brief explanation on how to proceed. The basic idea consists in adding terms which vanish if the solution has the correct boundary values. In doing so, we modify the PDE as if there were additional boundary terms. Adding ‘zero’ terms will not change our solution. Let u be our solution variable for which we want to impose the Dirichlet boundary condition $u = g$ on $\partial\Omega$, then we will add the following terms to our equation:

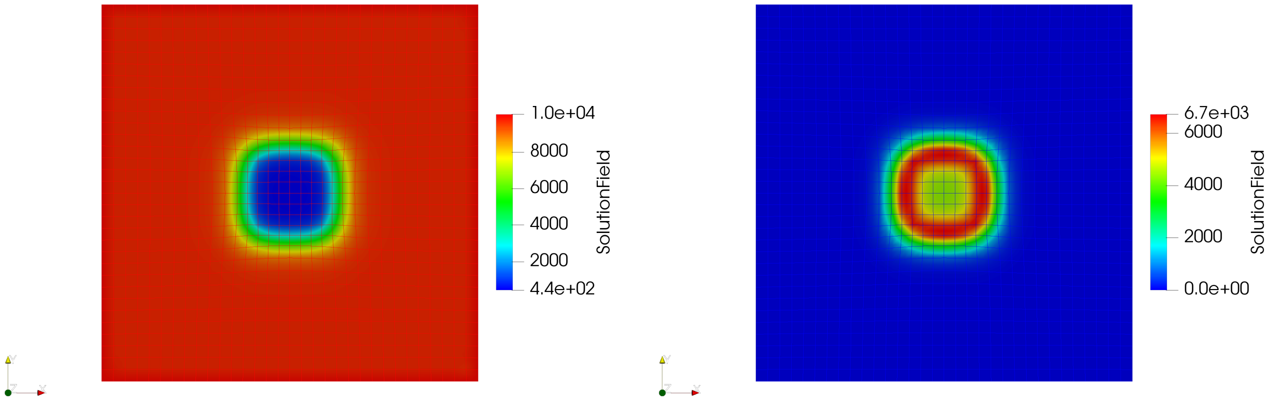
$$\int_{\partial\Omega} \gamma(\nabla w \cdot \mathbf{n})(u - g)\, d\Gamma + \int_{\partial\Omega} \frac{C}{h_e} w(u - g)\, d\Gamma,$$

were w is an element of our test space $H^1(\Omega)$, C is a constant, $\gamma = \pm 1$ and h_e an element length scale, see [38] for a more detailed discussion on the topic. As mentioned before, if our solution attains the desired boundary conditions these terms are equal to zero and the solution to the modified equations is the same as the solution for the original equations. We used $\gamma = -1$ and $\alpha = C/h_e = 1000$.

We have not been able, due to insufficient time, to implement the FCT algorithm in the IGA code. As such all results will be shown with a value of $k = 3$ for the parameter of the initial condition function. In Figure 9.2.1 we can see the plots of the solution variables N, M, c, ρ at $T = 20$. We can observe the same behaviour as in the finite element simulations. However, the peak values differ by approximately 5-10% in excess for the IGA simulation. This can be due to the different linearization applied. In fact, if we split the positive component of the linearized source term R_N in three parts we have

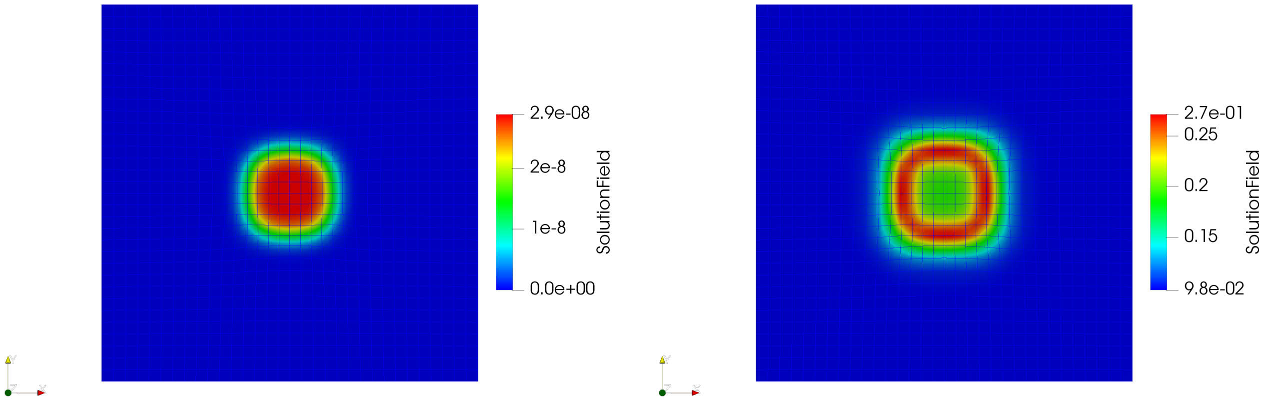
$$C(N^{*1+q} - \kappa_F N^{*1+q} N - \kappa_F M^* N^{*1+q}) \neq C(N^{*q} N - \kappa_F N^{*1+q} N - \kappa_F M^* N^{*q} N),$$

where $C = r_F \left[1 + \frac{r_F^{\max} c^*}{a_c^I + c^*}\right]$ and on the left side we see the split for the IGA linearization and on the right the split according to the naive approach of Section 5.4.1. The same applies for the linearized source term R_M.



(a) Variable N at $T = 20$.

(b) Variable M at $T = 20$.



(c) Variable c at $T = 20$.

(d) Variable ρ at $T = 20$.

Figure 9.2.1: IGA solution of biological variables at $T = 20$ with 32×32 bilinear elements, $\Omega_0 = [0, 8]^2$.

10

CONCLUSIONS AND FUTURE WORK

10.1. CONCLUSIONS

The aim of this work was to reproduce the healing process of skin burns after skin grafting. A morphoelastic model is used to describe the mechanical dynamics of the skin tissue and a biological model is described to track what are thought to be the main contributors to the healing process. Furthermore, we wanted to verify the viability of isogeometric analysis (IGA) in solving this set of equations. With this work we believe to have shown that the presented models do successfully describe the main characteristics and dynamics of skin graft contracture. It has also been shown that it is feasible to reproduce these results using IGA.

The developed algorithms are able to reproduce the general features of skin contracture after a skin graft placement. Qualitatively we can observe an increment of the myofibroblasts and of the collagen density together with a rise of the signalling molecules' concentration. This remains for a prolonged period of time as the collagen production rate is greater than its breakdown. The higher density of myofibroblasts and collagen in the wound area, compared to the uninjured surroundings, activates the mechanical process/response of skin contraction. The morphoelastic model captures the deformation of our domain and tracks the zero stress state in time. After reaching their respective peak values (at different times), the cell variables slowly return to their equilibrium state. Fibroblasts repopulate the wound area and myofibroblasts disappear, the collagen density returns to its normal levels and signalling molecules' concentration vanishes from the domain. Without the higher collagen density and myofibroblasts, the contracting forces decrease and the skin expands again until, after a year, the velocities of the modelled skin area are (close to) zero. We can observe residual strains in the domain, which indicates the development of residual stresses in the skin tissue, i.e. a contracture.

The developed model also highlights the importance of early treatment of skin grafts to prevent contractures. The mechanical equations show how it is possible to influence the evolution of the effective strain only when the concentration of signalling molecules is non-zero. As such, it appears crucial to start with the treatment as soon as possible after the skin graft surgery and to continue until the signalling molecule density has become negligible. This reasoning agrees with what found in [6].

Using the G+Smo library enabled us to implement first the mechanical and then the biological equations using an isogeometric analysis approach. The results obtained for the viscoelastic and morphoelastic models have been compared to the traditional finite element solutions and it can be seen that they coincide. For the extended model with biological constituents some modifications were necessary. A different linearization approach has been used and the boundary conditions have been enforced weakly. Qualitatively, the extended model solutions for IGA have the same dynamics as their traditional finite elements counterpart. The peak values of the constituents differ by approximately 5-10%, which we believe to be due to the different linearization approach. With the foundation laid in this work it is possible to further extend the IGA model and to fully exploit the potential of B-splines/NURBS. Their ability to accurately describe complex geometries can effectively be used to model real-life wounds, whose shapes can erratically differ from each other. Furthermore, it is still unclear how the increased continuity of the basis functions will affect the results of a fully developed model based on isogeometric analysis. The promising results from the application of IGA to mechanics and fluid dynamics [32, 33] make us believe that we can expect improvements in the modelling of the movement of the tissue.

10.2. RECOMMENDATIONS FOR FUTURE WORK

The work presented is a promising preliminary tool for skin graft contracture prediction. However, many aspects can be further improved and new additions can be made. In this section we want to give some suggestions for future research work on the topic of burn wound contraction. We believe that our current models and approaches can be further meliorated with the following ideas.

Boundary conditions In our results we could observe that the dimensions of the considered domain were too small for the modelled boundary conditions. In fact, the gradient of the concentration of fibroblasts close to the domain edges was not close to zero. This implies that imposing Dirichlet boundary conditions with equilibrium state values is not well suited for our case. Two possible approaches are to either consider a larger domain, which would imply an increase in computational costs, or to design and use a better suited boundary condition. This new boundary condition should simulate the desired far-field equilibrium value of fibroblasts, while allowing for exchange of fibroblasts across the boundaries of the computational domain. We also believe the description of the mechanical boundary conditions to be not versatile enough. Imposing that the boundaries of our domain do not move forces us again to consider a larger domain. A probably better suited mechanical boundary condition would be to consider spring-like forces applied along the edges of the domain.

Unify traditional FEM and IGA code To better compare the results between IGA and traditional FE, we recommend to apply the same linearization to both the approaches and to implement the FCT algorithm also for the IGA code. Our understanding of the methods applied in algebraic flux correction is limited and therefore it is better to first verify whether the concepts of finite element FCT translate directly onto the IGA framework. Algebraic flux correction enforced positivity of the nodal values, which would translate in positivity of the control points in IGA. Thanks to the convex hull property of B-spline surfaces this would suffice to ensure positivity of the solution.

Linearization Throughout this work we have discussed different linearization approaches and seen how small changes can influence the overall performance of the algorithms. It might be relevant to implement a Newton-Raphson linearization approach, which could possibly serve as a more reliable method. In fact, in [6] a Newton-Raphson linearization is used and it is presented as linearization method in [33]. An other possible linearization approach could be a modified L-scheme, as presented in [39] for nonlinear advection diffusion equations.

Time stepping schemes In [13] different time integration techniques for the morphoelastic equations are discussed. It is shown how the forward Euler integration in time is unstable for our modelling equations. The concepts of time integration on fixed meshes do not completely translate to time integration on moving meshes and it is therefore important to better understand these processes. For instance, the backward Euler method is not unconditionally stable in an ALE framework [26, 40]. In order to make this widely used scheme unconditionally stable, it must satisfy the discrete geometric conservation law. In [40] an analysis about these modifications can be found and an unconditionally stable second-order accurate scheme is presented.

Mesh refinement The adaptive mesh refinement tool used in this work is not well suited for our purpose. It would be beneficial for both increased accuracy and reduced computational costs to have a refinement tool which produces higher quality meshes. G+Smo allows for mesh refinement using THB-splines. Although it being unclear to us how non-conforming elements would affect the computations, the quality of the elements would probably be improved.

3D Extension An extension to a three dimensional model certainly allows for more detailed simulations. However, assumptions to reduce the model to a two dimensional surface are often made in order to reduce the computational costs. An interesting compromise would be to model a 2D surface in a 3D domain. It would thus be possible to better reproduce contractures of idealized two dimensional skin grafts in a three dimensional space, as the surface of an arm or any joint can be. This would allow us to more accurately determine the contracture behaviour and also track the influence that the location of the wound has on its healing process. Together with a detailed experimental validation it would almost certainly lead to a widely applicable and reliable model.

Neural networks Lastly, we want to stress the importance this research topic has for medical staff and professionals. Being able to predict the future contracture development of the skin grafts can allow for more efficient treatments. To allow for patient specific simulations, and thus treatments, it is crucial to reduce the computational time needed to have the results. In [8] we see how a neural network based computational framework is applied to the models described in our work. Promising results are shown and further research is needed. The interaction between numerical models and artificial intelligence is an emerging research topic which certainly can be investigated in our setting.

BIBLIOGRAPHY

[1] C. A. Harrison and S. MacNeil, *The mechanism of skin graft contraction: an update on current research and potential future therapies,* Burns **34**, 153 (2008).

[2] L. Yang, *Computational modeling and simulation study of dermal wound healing proliferative phase,* (2011).

[3] G.-Y. Hur, D.-K. Seo, and J.-W. Lee, *Contracture of skin graft in human burns: effect of artificial dermis,* Burns **40**, 1497 (2014).

[4] J. C. Schneider, R. Holavanahalli, P. Helm, R. Goldstein, and K. Kowalske, *Contractures in burn injury: defining the problem,* Journal of burn care & research **27**, 508 (2006).

[5] R. Richard and R. S. Ward, *Splinting strategies and controversies,* The Journal of burn care & rehabilitation **26**, 392 (2005).

[6] D. C. Koppenol and F. J. Vermolen, *Biomedical implications from a morphoelastic continuum model for the simulation of contracture formation in skin grafts that cover excised burns,* Biomechanics and modeling in mechanobiology **16**, 1187 (2017).

[7] C. L. Hall, *Modelling of some biological materials using continuum mechanics*, Ph.D. thesis, Queensland University of Technology (2008).

[8] M. W. Schaaphok, *A fast neural network based computational framework for the prediction of skin contraction,* (2020), to be defended Nov. 2020.

[9] S. Enoch and D. J. Leaper, *Basic science of wound healing,* Surgery (Oxford) **26**, 31 (2008).

[10] R. C. Jordan A. Browning, *Burn debridement, grafting, and reconstruction,* .

[11] I. Holand and K. Bell, *Finite element methods in stress analysis.* Tapir, Trondheim (1969).

[12] J. Van Kan, A. Segal, and F. Vermolen, *Numerical methods in scientific computing,* Department of Applied Mathematics, Delft University of Technology, The Netherlands (2008).

[13] D. Smits, *Morphoelastic models for burn contraction,* (2019).

[14] H. T. Banks, S. Hu, and Z. R. Kenz, *A brief review of elasticity and viscoelasticity for solids,* Advances in Applied Mathematics and Mechanics **3**, 1 (2011).

[15] S. N. Menon, C. L. Hall, S. W. McCue, and D. S. McElwain, *A model for one-dimensional morphoelasticity and its application to fibroblast-populated collagen lattices,* Biomechanics and Modeling in Mechanobiology **16**, 1743 (2017).

[16] E. K. Rodriguez, A. Hoger, and A. D. McCulloch, *Stress-dependent finite growth in soft elastic tissues,* Journal of biomechanics **27**, 455 (1994).

[17] A. Erlich, T. Lessinnes, D. Moulton, and A. Goriely, *A short introduction to morphoelasticity: the mechanics of growing elastic tissues,* in *Extremely Deformable Structures* (Springer, 2015) pp. 269–297.

[18] G. Dziuk and C. M. Elliott, *Finite element methods for surface pdes,* Acta Numerica **22**, 289 (2013).

[19] F. Nobile and L. Formaggia, *A stability analysis for the arbitrary lagrangian eulerian formulation with finite elements,* East-West Journal of Numerical Mathematics **7**, 105 (1999).

[20] A. Madzvamuse, A. J. Wathen, and P. K. Maini, *A moving grid finite element method applied to a model biological pattern generator,* Journal of computational physics **190**, 478 (2003).

[21] S. Patankar, *Numerical Heat Transfer and Fluid Flow* (McGraw-Hill, New York, 1980).

[22] L. G. de Pillis, *A comparison of iterative methods for solving nonsymmetric linear systems,* Acta Applicandae Mathematica **51**, 141 (1998).

[23] D. Kuzmin, *A guide to numerical methods for transport equations,* Univ. Nürnberg. URL http://www.mathematik.uni-dortmund.de/˜ kuzmin/Transport.pdf , 146 (2010).

[24] V. John and E. Schmeyer, *Finite element methods for time-dependent convection–diffusion–reaction equations with small diffusion,* Computer methods in applied mechanics and engineering **198**, 475 (2008).

[25] M. Möller, D. Kuzmin, and D. Kourounis, *Implicit fem-fct algorithms and discrete newton methods for transient convection problems,* International journal for numerical methods in fluids **57**, 761 (2008).

[26] S. Ganesan and S. Srivastava, *Ale-supg finite element method for convection–diffusion problems in time-dependent domains: Conservative form,* Applied Mathematics and Computation **303**, 128 (2017).

[27] O. Boiarkine, D. Kuzmin, S. Čanić, G. Guidoboni, and A. Mikelić, *A positivity-preserving ale finite element scheme for convection–diffusion equations in moving domains,* Journal of Computational Physics **230**, 2896 (2011).

[28] D. Kuzmin, M. Möller, and S. Turek, *Multidimensional fem-fct schemes for arbitrary time stepping,* International journal for numerical methods in fluids **42**, 265 (2003).

[29] S. A. Funken and A. Schmidt, *Adaptive mesh refinement in 2d–an efficient implementation in matlab,* Computational Methods in Applied Mathematics **1** (2019).

[30] C. J. Roy, *Review of code and solution verification procedures for computational simulation,* Journal of Computational Physics **205**, 131 (2005).

[31] A. Pont, R. Codina, and J. Baiges, *Interpolation with restrictions between finite element meshes for flow problems in an ale setting,* International Journal for Numerical Methods in Engineering **110**, 1203 (2017).

[32] T. J. Hughes, J. A. Cottrell, and Y. Bazilevs, *Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement,* Computer methods in applied mechanics and engineering **194**, 4135 (2005).

[33] J. A. Cottrell, T. J. Hughes, and Y. Bazilevs, *Isogeometric analysis: toward integration of CAD and FEA* (John Wiley & Sons, 2009).

[34] R. Duvigneau, *An introduction to isogeometric analysis with application to thermal conduction,* (2009).

[35] V. P. Nguyen, P. Kerfriden, S. P. Bordas, and T. Rabczuk, *Isogeometric analysis suitable trivariate nurbs representation of composite panels with a new offset algorithm,* Computer-Aided Design **55**, 49 (2014).

[36] H. Elman, D. Silvester, and A. Wathen, *Finite elements and fast iterative solvers: With applications in incompressible fluid dynamics,* (2006).

[37] K. Yanase, *A gentle introduction to isogeometric analysis part i: B-spline curve & 1d finite element analysis,* Fukuoka University Review of Technological Sciences **98**, 1 (2017).

[38] Y. Bazilevs and T. J. Hughes, *Weak imposition of dirichlet boundary conditions in fluid mechanics,* Computers & Fluids **36**, 12 (2007).

[39] K. Mitra and I. S. Pop, *A modified l-scheme to solve nonlinear diffusion problems,* Computers & Mathematics with Applications **77**, 1722 (2019).

[40] J. Mackenzie and W. Mekwi, *An unconditionally stable second-order accurate ale–fem scheme for two-dimensional convection–diffusion problems,* IMA Journal of Numerical Analysis **32**, 888 (2012).