Title
Computation of in vivo myocardial deformation from planar imaging

Author
Rodriguez, Miguel Angel

Publication Date
2022

Peer reviewed|Thesis/dissertation

Computation of in vivo myocardial deformation from planar imaging

by

Miguel Angel Rodriguez

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering – Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Shawn C. Shadden, Chair
Professor David J. Steigmann
Associate Professor Moriel H. Vandsburger

Summer 2022

Computation of in vivo myocardial deformation from planar imaging

Abstract

Computation of in vivo myocardial deformation from planar imaging

by

Miguel Angel Rodriguez

Doctor of Philosophy in Engineering – Mechanical Engineering

University of California, Berkeley

Professor Shawn C. Shadden, Chair

Cardiovascular disease has remained the leading cause of death worldwide. In addition, the 2015 Annual Data Report from the US Renal Data System shows that sudden cardiac death and arrhythmias are the leading cause of mortality for patients on routine hemodialysis for end-stage renal disease (ESRD). In this study, we aim to augment cardiac magnetic resonance (CMR) images obtained for such patients through a registration framework developed for analyzing two-dimensional (2D) CMR image sequences. The framework consolidates existing tools, such as segmentation and the large deformation diffeomorphic metric mapping (LDDMM) algorithm, and provides a systematic process to define the circumferential and radial directions in general left ventricle geometries, to quantify its myocardial strain throughout the cardiac cycle. While CMR image sequences are used in this work, the framework is designed to be agnostic to the imaging modality. In conjunction with this framework, validation tests were developed using synthetic deformation data generated through solid mechanics simulations. This was done for three analytical shapes: a box, thick-walled cylinder, and a thick-walled ellipsoid, all in three dimensions. This synthetic data was then used to provide a 2D input to the registration framework, which allowed a direct comparison of the deformation computed from the solid mechanics simulation and the registration framework. All of this was developed to use data that is generally readily accessible in clinical settings, as well as to provide a template for designing validation tests for myocardial strain computations.

A mis padres que arriesgaron todo para darnos más de lo que tuvieron.

Contents

List of Figures

List of Tables

Acknowledgments

My journey before and during graduate school is indebted to many wonderful people. First and foremost, I would like to thank my PhD advisor, Dr. Shawn Shadden, whose support has granted me with a breadth of knowledge and confidence in effectively communicating the work we do—even after showing up unannounced to his research group. The courses provided by Dr. David Steigmann have also been a great source of knowledge provided with a level of rigor that is unparalleled by many. I will forever be grateful for our continuum mechanics discussions and his expert guidance. Additionally, the work presented in this dissertation would not have been possible without Dr. Moriel Vandsburger, as well as his support and assistance. The image data provided by Dr. Vandsburger and his colleagues was a true savior to my research; as well as the assistance in processing this image data by Cindy Ayala and Paul Dennig. I would be remiss if I didn't thank Meltem for always making sure I had what I needed to succeed.

I would also like to thank my undergraduate professors and mentors. The confidence that Dr. Hubertus von Bremen placed in me throughout my undergraduate studies kept me motivated to learn as much as I possibly could, both from an engineering and mathematical perspective. I owe a great deal of gratitude to Dr. Marco Quadrelli—another phenomenal expert in continuum mechanics—for giving me the opportunity and support to participate in the internship program at the Jet Propulsion Laboratory; it is here that I was introduced to Python and Emacs. Dr. Arlo Caine also provided a breadth of mathematical knowledge that satiated my desire for rigor often lacking in undergraduate engineering curriculums. Not only was he able to answer many of my questions on the spot, he is the person who, knowing my history with congenital heart disease, introduced me to the field of cardiovascular biomechanics. That was the moment I decided to contribute to this field.

I can confidently say that I would not have made it this far without the care provided by my physicians, specifically Dr. Richard Wittner and the cardiology team at UCLA. Dr. Wittner's incredibly high standards towards medical colleagues served as an example of how one should stay honest to themselves and their work. I owe my life to him.

My graduate school experience would not have been complete without the community of friends and colleagues. The student organization LAGSES provided the perfect setting for building life-long friendships. I owe much appreciation to Benson, Roberto, and Jeff who were always there when needed and willingly spent many days sharing experiences, stories, and recipes. To Jasmine, thank you for the love and support you continue to provide throughout our journey together; also for enduring the pandemic with me.

Last, but definitely not least, I would like to thank my family. They provided an immense amount of love and support throughout my entire life; particularly during my medical hardships. Mom and Dad, I would not have gotten this far nor would I be who I am without your love, support, and incredible guidance throughout my life. Thank you for everything that you have risked and provided for us.

Stay honest and please help leave this world in better shape than when we arrived.

Chapter 1

Introduction

1.1 Motivation

Engineers and mathematicians have contributed more and more to the study of the human body over the past few decades. This is due to the rise in computational power and the fact that the physical principles used to design a vehicle that can safely transport hundreds of people in the air across the globe make no assumption about their application. Therefore, the human body and its complex characteristics—many of which are yet to be understood—should be describable through these same fundamental tools. In addition, surgeries have been one of trial-and-error at the detriment of unsuspecting test subjects for the better part of its existence as can be inferred from its history [1], further motivating the need to quantify healthy and pathological phenomena.

To this day, cardiovascular disease remains the leading cause of death worldwide [2]. Notably, the 2015 Annual Data Report from the US Renal Data System states that the leading causes of death for patients on routine hemodialysis for end-stage renal disease (ESRD) are sudden cardiac death and arrhythmias [3]. Though cardiac magnetic resonance (CMR) imaging combined with gadolinium-based contrast agents is the standard for diagnosing heart disease [4, 5, 6], these contrast agents are contraindicated in patients with ESRD due to their reduced kidney function [7]. Driven by this limitation, and the increased risk of adverse cardiac events [8], Stromp et al. developed a gadolinium-free imaging method to identify diseased myocardium [9] and subsequently found a strong correlation between initial fibrotic burden and subsequent loss of contractile function [10].

While these and other advancements in medical imaging have provided clinicians with the capability to obtain views of the cardiovascular system that were previously unfathomable, there are still many unknowns. Namely, while medical images capture the varying shapes the heart exhibits throughout a cardiac cycle, they are incapable of quantifying the mechanical strain that the tissue undergoes on their own. Herein we refer to this mechanical strain, i.e. the geometric measure of deformation, solely as deformation. To quantify such deformation requires that mathematical descriptions of deformable motion be augmented,

either through image analysis or mathematical modeling. Continuum mechanics provides a formal mathematical approach to describe such motion which, in conjunction with medical images, can expand our understanding of the differing kinematics of the heart between different patients. Additional challenges include accurately diagnosing patients due to the plethora of various heart diseases, heart anatomies, and intra- and inter-operator variations in acquiring and analyzing image data.

As Amzulescu et al. mention in their review [11], numerous approaches to compute myocardial strain have been developed to address the aforementioned challenges. Speckle tracking echocardiography, for example, uses feature tracking (FT) algorithms to compute deformation from images themselves. More specifically, the intensities of the images are used to track their speckle texture as in Notomi et al. [12], or through block matching as reviewed in [13]. Another approach is tagged CMR wherein different regions of tissue are magnetically labeled to provide the tagged pattern throughout the cardiac cycle [14]. Amzulescu et al. point out the drastic differences in spatial and temporal resolutions across various techniques. They also summarize the various causes for differences between these techniques, e.g. the quality of the acquisition, size of the search region in FT algorithms, down to the definition of strain itself (Lagrangian vs. Eulerian strain). This led to our goal of developing and applying a registration framework to planar images in order to compute myocardial strain, while simultaneously designing a validation process using synthetically generated data. Though this framework was applied to readily available CMR images, only the endo- and epicardium were extracted from them.

1.2 Dissertation Outline

This work consisted of developing a framework that would augment planar medical images by computing myocardial deformation of the left ventricle (LV). A validation procedure was also designed in conjunction to test the strain computing framework with higher rigor than is typically found in the literature. This analysis was performed using synthetic data generated through solid mechanics simulations of idealized geometries in order to allow a direct comparison of the framework's results with a true answer—something that is, at this moment, ethically untenable with patient data.

Some basics of continuum mechanics needed to formulate cardiovascular biomechanics problems are provided in Chapter 2, along with a summary of FEniCS Mechanics [15], which is a Python package for simulating a single-domain, steady or time-dependent, fluid or solid mechanics problem. Chapter 3 then describes the framework developed to compute LV deformation from two-dimensional (2D) CMR data, and presents results for 35 patients. The validation tests developed and performed on the strain computing framework using idealized geometries are presented in Chapter 4. It is here that the choice of parameters used when computing the strain for patient image data is determined. Lastly, an overall summary of the results, shortcomings, and potential future work are discussed in Chapter 5.

Chapter 2

Computational Solid Mechanics

In this chapter, we will cover the fundamentals necessary to solve computational mechanics problems. Specifically, we will cover continuum mechanics, the variational formulation of such mechanics problems, and discretization techniques used to obtain approximate solutions to these problems. Additionally, we will present the Python package FEniCS Mechanics, as well as some specific examples that will be of interest later on. While we will try to make this as extensive as necessary for chapters that follow, the interested reader is encouraged to read additional material such as Chadwick [16] and Hughes [17] for continuum mechanics and FEM, respectively.

2.1 Continuum Mechanics

Continuum mechanics is the foundation on which mathematical descriptions of a majority of physical phenomena around us is based. As the name suggests, the key assumption is that we are dealing with continuous media; that is to say that there are no voids present in any infinitesimal neighborhood of any interior point of the body of interest. While we now know that this assumption will be proved erroneous by sufficiently zooming into any material, it is still reasonable for a vast number of engineering applications. So much so that undergraduate engineering curricula are primarily based on the formulations provided by continuum mechanics.

A brief dive into the myriad of continuum mechanics textbooks will quickly show that the level of mathematical formatility varies significantly based on the author. Here, we will aim to be as formal as possible while assuming familiarity with certain results from calculus and tensor analysis.

2.1.1 Kinematics of Continua

The first step in our journey to describe the behavior of continuous media is to define some terminology. In our case, we aim to describe the motion of an arbitrary body that is assumed

to occupy a subregion of three-dimensional Euclidean space, $\mathcal{E}_3$, at some initial time t_0. We will denote this arbitrary region by $\mathcal{B}_0$ and refer to it as the reference configuration. The subregion of $\mathcal{E}_3$ occupied by the same body (same set of material points) at some later time, t, will be denoted by $\mathcal{B}$ and referred to as the current configuration. The description that we seek is in the form of a map that transforms the body from $\mathcal{B}_0$ to $\mathcal{B}$, which we will denote as $\boldsymbol{\varphi} : \mathcal{B}_0 \times \mathbb{R} \to \mathcal{B}$. In order to first develop some tools, we will assume that $\boldsymbol{\varphi}$ is known to be continuously differentiable with respect to spatial variables. I.e., there exists a unique second-order tensor $\mathbf{F}$ such that

$$\boldsymbol{\varphi}(\mathbf{X}_2, t) = \boldsymbol{\varphi}(\mathbf{X}_1, t) + \mathbf{F}(\mathbf{X}_1, t)\,(\mathbf{X}_2 - \mathbf{X}_1) + \mathbf{o}(d), \tag{2.1}$$

where $\mathbf{X}_1, \mathbf{X}_2 \in \mathcal{B}_0$ are position vectors to two material points in the reference configuration, and

$$\lim_{d \to 0} \frac{\mathbf{o}(d)}{d} = 0,$$

where $d = \|\mathbf{X}_2 - \mathbf{X}_1\|$. The tensor $\mathbf{F}$ is what is known as the deformation gradient. A common convention in continuum mechanics is to define this deformation gradient such that

$$d\mathbf{x} = \mathbf{F}\; d\mathbf{X}, \tag{2.2}$$

where $\mathbf{x} = \boldsymbol{\varphi}(\mathbf{X}, t)$, and hence $d\mathbf{x}$ and $d\mathbf{X}$ are vectors between two infinitesimally close material points at the current and reference configuration, respectively. Since $\mathbf{F}$ is really just a derivative of the map $\boldsymbol{\varphi}$, it may also be denoted as

$$\mathbf{F} = \frac{\partial \boldsymbol{\varphi}}{\partial \mathbf{X}}. \tag{2.3}$$

Furthermore, note that the map $\boldsymbol{\varphi}$ can be written in terms of a displacement, i.e.

$$\mathbf{x} = \boldsymbol{\varphi}(\mathbf{X}, t) = \mathbf{X} + \mathbf{u}(\mathbf{X}, t).$$

Thus, the deformation gradient can also be written as

$$\mathbf{F} = \mathbf{I} + \frac{\partial \mathbf{u}}{\partial \mathbf{X}}. \tag{2.4}$$

If the map is also assumed to be differentiable with respect to time t, its velocity field is given by

$$\mathbf{v} = \frac{\partial \mathbf{x}}{\partial t} = \frac{\partial \boldsymbol{\varphi}(\mathbf{X}, t)}{\partial t}. \tag{2.5}$$

2.1.1.1 Key results from tensor algebra and calculus

In order to ensure that the deformation described by a map $\boldsymbol{\varphi}$ is in accordance with our experience of the world, we must impose some mathematical restrictions on it. To do so, however, we first look at quantities associated with a second-order tensor that are invariant to a frame of reference. The values we will discuss are often simply referred to as the invariants of a tensor. First though, let

$$[\mathbf{u}, \mathbf{v}, \mathbf{w}] = \mathbf{u} \cdot (\mathbf{v} \times \mathbf{w}), \tag{2.6}$$

known as the box product of three vectors. The three invariants of a second-order tensor, say $\mathbf{A}$, in $\mathcal{E}_3$ that we will consider are $I_1(\mathbf{A}), I_2(\mathbf{A}), I_3(\mathbf{A})$, such that

$$I_1(\mathbf{A})[\mathbf{u}, \mathbf{v}, \mathbf{w}] = [\mathbf{A}\mathbf{u}, \mathbf{v}, \mathbf{w}] + [\mathbf{u}, \mathbf{A}\mathbf{v}, \mathbf{w}] + [\mathbf{u}, \mathbf{v}, \mathbf{A}\mathbf{w}], \tag{2.7}$$

$$I_2(\mathbf{A})[\mathbf{u}, \mathbf{v}, \mathbf{w}] = [\mathbf{A}\mathbf{u}, \mathbf{A}\mathbf{v}, \mathbf{w}] + [\mathbf{A}\mathbf{u}, \mathbf{v}, \mathbf{A}\mathbf{w}] + [\mathbf{u}, \mathbf{A}\mathbf{v}, \mathbf{A}\mathbf{w}], \tag{2.8}$$

$$I_3(\mathbf{A})[\mathbf{u}, \mathbf{v}, \mathbf{w}] = [\mathbf{A}\mathbf{u}, \mathbf{A}\mathbf{v}, \mathbf{A}\mathbf{w}], \tag{2.9}$$

for any $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathcal{E}_3$. It can be shown that

$$I_1(\mathbf{A}) = \operatorname{tr} \mathbf{A}, \tag{2.10}$$

$$I_2(\mathbf{A}) = \frac{1}{2}\left[(I_1(\mathbf{A}))^2 - I_1(\mathbf{A}^2)\right], \tag{2.11}$$

$$I_3(\mathbf{A}) = \det \mathbf{A}, \tag{2.12}$$

where tr denotes the trace of a tensor, and det denotes its determinant. The third invariant is of key importance for the mathematical description that we seek since, from calculus, we know that

$$\int_{\mathcal{B}} f \, dv = \int_{\mathcal{B}_0} \hat{f} J \, dV, \tag{2.13}$$

when $f : \mathcal{B} \to \mathbb{R}$ undergoes a change of coordinates, or configuration, defined by $\boldsymbol{\varphi} : \mathcal{B}_0 \to \mathcal{B}$, and $J = \det \mathbf{F}$ is the determinant of its deformation gradient. Here, $\hat{f} = f \circ \boldsymbol{\varphi}$. Note that setting $f = 1$ for all $\mathbf{x} \in \mathcal{B}$ results in

$$\int_{\mathcal{B}} dv = \int_{\mathcal{B}_0} J \, dV,$$

which indicates that J quantifies the amount by which an infinitesimal volume element in the reference configuration, dV, is altered by the deformation resulting from $\boldsymbol{\varphi}$. Thus,

$$\det \mathbf{F} = J > 0 \tag{2.14}$$

must hold in order for $\boldsymbol{\varphi}$ to make physical sense.

If f is extended such that it is time dependent, i.e. $f : \mathcal{B} \times \mathbb{R} \to \mathbb{R}$, its rate of change with respect to time as the material particle at which it is evaluated is held constant is given by

$$\dot{f} = \frac{d}{dt} f(\mathbf{x}, t) = \frac{d}{dt} f(\boldsymbol{\varphi}(\mathbf{X}, t), t) = \frac{\partial f}{\partial t} + \mathbf{v} \cdot \operatorname{grad} f, \tag{2.15}$$

where "grad" is the gradient differential operator with respect to $\mathbf{x} \in \mathcal{B}$, and $\mathbf{v}$ is the velocity field of $\boldsymbol{\varphi}$. Upper-case versions of differential operators, e.g. "Grad" indicate that the differential operator is instead taken with respect to $\mathbf{X} \in \mathcal{B}_0$. Note that this result is obtained by applying the chain rule from multivariable calculus, and is referred to as the material derivative. This definition can be extended to vector-valued functions; consider a vector-valued function $\mathbf{g}$ which also depends on $\mathbf{x}$ and t. The material derivative of $\mathbf{g}$ is

$$\dot{\mathbf{g}} = \frac{\partial \mathbf{g}}{\partial t} + (\operatorname{grad} \mathbf{g})\, \mathbf{v}. \tag{2.16}$$

While the material derivative is often denoted by

$$\frac{D(\cdot)}{Dt} = \frac{\partial(\cdot)}{\partial t} + \mathbf{v} \cdot \nabla(\cdot)$$

in the literature, the overhead dot will be preserved for this explicit purpose here. Note that the material derivative of a function reparametrized to be defined in terms of $\mathbf{X}$ rather than $\mathbf{x}$ mantains the same physical meaning while eliminating the second term in Equations (2.15) and (2.16).

With this result and definition of the third invariant, it can be shown that

$$\dot{J} = J \operatorname{tr} \mathbf{L}, \tag{2.17}$$

where $\mathbf{L} = \operatorname{grad} \mathbf{v} = \operatorname{grad} \dot{\boldsymbol{\varphi}}$ is the gradient of the velocity vector field.

Other important results are the transport relations for scalar- and vector-valued functions. The transport relations state that for an arbitrary scalar-valued function, f,

$$\frac{d}{dt} \int_{\mathcal{B}} f \, dv = \int_{\mathcal{B}} \left(\dot{f} + f \operatorname{tr} \mathbf{L} \right) dv, \tag{2.18}$$

It can also be shown that $\operatorname{tr} \mathbf{L} = \operatorname{div} \mathbf{v}$. Similarly, for a vector-valued function $\mathbf{u}$,

$$\frac{d}{dt} \int_{\mathcal{B}} \mathbf{u} \, dv = \int_{\mathcal{B}} (\dot{\mathbf{u}} + \mathbf{u} \operatorname{tr} \mathbf{L}) \, dv. \tag{2.19}$$

Lastly, we would like to recall the divergence theorem, which allows us to transform an integral over a volumetric domain, e.g. $\mathcal{B}$, to an integral over its boundary, $\partial\mathcal{B}$. We take the general form of this theorem presented by Chadwick [16], which is

$$\int_{\partial\mathcal{B}} \mathbf{u} \otimes \left(\mathbf{T}^T \mathbf{n} \right) da = \int_{\mathcal{B}} \left(\mathbf{u} \otimes \operatorname{div} \mathbf{T} + (\operatorname{grad} \mathbf{u})\, \mathbf{T} \right) dv, \tag{2.20}$$

where $\otimes$ denotes the tensor product, $\mathbf{T}$ is an arbitrary second-order tensor, and $\mathbf{u}$ is an arbitrary vector. The reader is encouraged to arrive at familiar forms of the divergence theorem by exploring special cases of $\mathbf{T}$ and $\mathbf{u}$.

All of the aforementioned results will be useful for the development of the balance laws, which in turn will give us a set of equations to solve.

2.1.1.2 Strain components along a curve

In preparation for later chapters, let us consider a continuously differentiable simple curve, call it $\mathcal{C}_0$, embedded in the body of interest. That is to say that $\mathcal{C}_0 \subset \mathcal{B}_0$. Let $\mathbf{M}$ be a unit vector tangent to $\mathcal{C}_0$ at $\mathbf{X}$. Furthermore, an infinitesimal arc of the curve at $\mathbf{X}$ is denoted by $d\mathbf{X} = \mathbf{M}dS$, where $\mathbf{M}$ is a unit vector tangent to the curve at $\mathbf{X}$, and dS is the length of the infinitesimal arc. This material curve is then mapped to $\mathcal{C} = \boldsymbol{\varphi}(\mathcal{C}_0)$, and thus $d\mathbf{X}$ is mapped to $d\mathbf{x}$ in the current configuration, where $\mathbf{x} = \boldsymbol{\varphi}(\mathbf{X}, t)$. Making use of Equation (2.2), we can write

$$\begin{aligned} ds^2 = d\mathbf{x} \cdot d\mathbf{x} &= (\mathbf{F}\, d\mathbf{X}) \cdot (\mathbf{F}\, d\mathbf{X}) = d\mathbf{X} \cdot \left(\mathbf{F}^T\mathbf{F}\right) d\mathbf{X} = (\mathbf{M}\, dS) \cdot \left(\mathbf{F}^T\mathbf{F}\right) (\mathbf{M}\, dS) \\ &= \left(\mathbf{M} \cdot \left(\mathbf{F}^T\mathbf{F}\right) \mathbf{M}\right) dS^2 = (\mathbf{M} \cdot \mathbf{C}\mathbf{M})\, dS^2, \end{aligned} \tag{2.21}$$

where $dS^2 = d\mathbf{X} \cdot d\mathbf{X}$, $ds^2 = d\mathbf{x} \cdot d\mathbf{x}$, and $\mathbf{C} = \mathbf{F}^T\mathbf{F}$ is the right Cauchy-Green tensor. Note that $\mathbf{M} \cdot \mathbf{C}\mathbf{M}$ can be interpreted as the "M-M" component of $\mathbf{C}$ since $\mathbf{M}$ is a unit vector. Thus, the diagonal components of the right Cauchy-Green tensor is the square of the stretch experienced by the curve at the material point $\mathbf{X}$ under the deformation described by $\boldsymbol{\varphi}$.

Now, let us consider a second curve that intersects orthogonally with $\mathcal{C}_0$ at $\mathbf{X}$, call it $\hat{\mathcal{C}}_0$. Let $\hat{\mathbf{M}}$ be the unit tangent vector to this second curve, while $d\hat{\mathbf{X}}$ and $d\hat{\mathbf{x}}$ are the corresponding infinitesimal tangent vectors in the reference and current configurations, respectively. Following the same strategy as in Equation (2.21), we see that

$$\begin{aligned} d\hat{\mathbf{x}} \cdot d\mathbf{x} &= \left(\mathbf{F}\, d\hat{\mathbf{X}}\right) \cdot (\mathbf{F}\, d\mathbf{X}) = d\hat{\mathbf{X}} \cdot \mathbf{C}\, d\mathbf{X} \\ &= \left(\hat{\mathbf{M}}\, d\hat{S}\right) \cdot \mathbf{C}\, (\mathbf{M}\, dS) = \left(\hat{\mathbf{M}} \cdot \mathbf{C}\mathbf{M}\right) d\hat{S}\, dS. \end{aligned}$$

Note that

$$d\hat{\mathbf{x}} \cdot d\mathbf{x} = (\hat{\mathbf{m}}\, d\hat{s}) \cdot (\mathbf{m}\, ds) = (\hat{\mathbf{m}} \cdot \mathbf{m})\, d\hat{s}\, ds = \cos\theta\, d\hat{s}\, ds,$$

where $\hat{\mathbf{m}}$ and $\mathbf{m}$ are the unit vectors tangent to $\hat{\mathcal{C}}$ and $\mathcal{C}$, respectively, and θ is the angle between them. Thus,

$$\cos\theta\, d\hat{s}\, ds = \left(\hat{\mathbf{M}} \cdot \mathbf{C}\mathbf{M}\right) d\hat{S}\, dS. \tag{2.22}$$

This shows that the off-diagonal components of $\mathbf{C}$ with respect to an orthonormal basis are the product of the stretch undergone by two infinitesimal material curve elements along the two directions and the cosine of the resulting angle in the current configuration. We will be making further use of the right Cauchy-Green strain tensor when discussing constitutive equations, and Equations (2.21) and (2.22) will be applied in Section 3.3.

2.1.2 Balance Laws

In the previous section, we discussed the map, $\boldsymbol{\varphi}$, that transforms the reference configuration of a body, $\mathcal{B}_0$, to the current configuration, $\mathbf{B}$. The observant reader will have already noticed that we made no comment on how $\boldsymbol{\varphi}$ is determined. While leaving this discussion general opens the door to numerous applications, we are specifically interested in describing the motion of continuous media constrained by the laws of physics. In order to abide by these non-negotiable laws, we must ensure that our description of motion guarantees that:

1. mass is neither created or destroyed,
2. the rate of change in linear (angular) momentum of the body is balanced by the sum of forces (moments) exerted on the body.
3. energy is neither created or destroyed,

These three seemingly simple postulates (at first glance), we are able to arrive at a general mathematical description for the motion of a myriad of continuous media. First, we describe the conservation of mass.

2.1.2.1 Conservation of mass

At this point, we will depend on our intuition of what mass is. The total mass in a sub-region of a body in the current configuration, call it $\mathcal{D} \subset \mathcal{B}$, is given by

$$m(\mathcal{D}, t) = \int_{\mathcal{D}} \rho(\mathbf{x}, t)\, dv,$$

where we assume that there exists a non-negative mass density function $\rho : \mathcal{B} \times \mathbb{R} \to \mathbb{R}$. By the first postulate, we must have that the mass in a fixed subregion, such as $\mathcal{D}$, must remain constant with respect to time. I.e.,

$$0 = \frac{dm(\mathcal{D}, t)}{dt} = \frac{d}{dt} \int_{\mathcal{D}} \rho\, dv = \int_{\mathcal{D}} \left(\dot{\rho} + \rho \operatorname{tr} \mathbf{L}\right)\, dv,$$

where we made use of the transport relation given in Equation (2.18), and we can conclude that

$$\frac{\partial \rho}{\partial t} + \operatorname{div}(\rho \mathbf{v}) = \dot{\rho} + \rho \operatorname{div} \mathbf{v} = 0 \tag{2.23}$$

by shrinking the region $\mathcal{D}$ such that its volume tends to zero, which is known as the continuity equation.

While ρ was defined as the density function in the current configuration, we can let $\rho_0 : \mathcal{B}_0 \to \mathbb{R}$ be the referential density function. Note that

$$0 = \dot{\rho} + \rho \operatorname{tr} \mathbf{L} = \dot{\rho} + \rho \frac{\dot{J}}{J} = \frac{1}{J}\left(\dot{\rho} J + \rho \dot{J}\right) = \frac{1}{J}(\rho J)^{\cdot},$$

which shows that ρJ is constant for all time, and hence

$$\rho_0 = \rho J \tag{2.24}$$

is the referential form of the continuity equation.

The reader will also find other mathematical forms of this postulate in the literature. Namely,

$$\frac{d}{dt}\int_{\mathcal{W}} \rho \, dv + \int_{\partial\mathcal{W}} \rho \mathbf{v} \cdot \mathbf{n} \, da = 0, \tag{2.25}$$

where $\mathcal{W}$ is a sub-region of $\mathcal{B}$ that is not necessarily made up of the same material points, $\partial\mathcal{W}$ is its surface boundary, and $\mathbf{n}$ is the unit outward normal to this surface.

2.1.2.2 Balance of momentum

Next, we consider the balance of linear momentum. First, we consider two types of forces which are recognized in continuum mechanics. Namely, body forces and traction (or surface) forces. Body forces whose magnitude depends on the amount of matter on which it is acting on. Let $\mathbf{b}(\mathbf{x}, t)$ be the net sum of this body force per unit mass at $\mathbf{x}$. Then the total body force exerted on the region of interest is

$$\mathbf{b}_{\text{total}}(\mathcal{B}, t) = \int_{\mathcal{B}} \rho \mathbf{b} \, dv. \tag{2.26}$$

On the other hand, traction forces act on surface regions and are thus proportional in magnitude to surface area. For this category of forces, let $\mathbf{t}(\mathbf{x}, t; \mathbf{n})$ denote the net sum of traction forces per unit area acting at $\mathbf{x} \in \partial\mathcal{B}$. Note that this traction force density depends on the position, time, and the vector normal to the surface at $\mathbf{x}$. The total traction force exerted on $\mathbf{B}$ is then

$$\mathbf{t}_{\text{total}}(\partial\mathcal{B}, t) = \int_{\partial\mathcal{B}} \mathbf{t} \, da. \tag{2.27}$$

Now, recall that the linear momentum of a particle is defined as the product of its mass and velocity. Thus, for the case of a continuous media, the total linear momentum is given by

$$\mathbf{G}(\mathcal{B}, t) = \int_{\mathcal{B}} \rho \mathbf{v} \, dv. \tag{2.28}$$

Then, by the second postulate, we must have that

$$\frac{d}{dt}\mathbf{G}(\mathcal{B}, t) = \mathbf{b}_{\text{total}}(\mathcal{B}, t) + \mathbf{t}_{\text{total}}(\partial\mathcal{B}, t).$$

In other words,

$$\frac{d}{dt}\int_{\mathcal{B}} \rho\mathbf{v}\, dv = \int_{\mathcal{B}} \rho\dot{\mathbf{v}}\, dv = \int_{\mathcal{B}} \rho\mathbf{b}\, dv + \int_{\partial\mathcal{B}} \mathbf{t}\, da, \tag{2.29}$$

where we made use of the transport Equation (2.18) and the Continuity Equation (2.23) in order to convert the time derivative outside of the integral into a material derivative of the velocity field. It can be shown that there exists a unique second-order tensor $\mathbf{T}(\mathbf{x}, t)$

$$\mathbf{t}(\mathbf{x}, t; \mathbf{n}) = \mathbf{T}\mathbf{n}. \tag{2.30}$$

This tensor is called the Cauchy stress tensor. Substituting this tensor into Equation (2.29) and making use of the divergence theorem, we arrive at the governing partial differential equation for continuous media, which is

$$\rho\dot{\mathbf{v}} = \operatorname{div}\mathbf{T} + \rho\mathbf{b}. \tag{2.31}$$

This partial differential equation (PDE), along with a set of initial and boundary conditions (BCs) defines an Initial Boundary Value Problem (IBVP) for which analytical solutions can only be found for specialized cases.

While one can continue to dive deeper into the development of the Cauchy stress tensor, we will simply state that, for our purposes, the balance of angular momentum simply provides us with the necessary condition that

$$\mathbf{T}^T = \mathbf{T} \tag{2.32}$$

must hold for all $\mathbf{x} \in \mathcal{B}$ and $t \in \mathbb{R}$.

As we initially did with the conservation of mass, our developments to this point have all been in the current configuration. Some massaging of the equations will lead us to conclude that

$$\rho_0\dot{\mathbf{v}} = \operatorname{Div}\mathbf{P} + \rho_0\mathbf{b}, \tag{2.33}$$

where Div is the divergence operator with respect to the referential coordinates, and both $\mathbf{v}$ and $\mathbf{b}$ are assumed to have been re-parameterized with respect to these same referential coordinates. Also,

$$\mathbf{P} = J\mathbf{T}\mathbf{F}^{-T} \tag{2.34}$$

is known as the first Piola-Kirchhoff stress tensor. With this tensor, we can define the referential traction force

$$\mathbf{p} = \mathbf{P}\mathbf{N}, \tag{2.35}$$

where $\mathbf{N}$ is the unit vector that is normal to the surface in the reference configuration, $\partial\mathcal{B}_0$. We will make extensive use of these governing equations in the reference configuration.

2.1.2.3 Conservation of energy

For the last postulate, we will simply state the general form of the resulting governing equation. This is due to the fact that the conservation of energy is inherently satisfied for the applications that we will consider.

$$\frac{d}{dt}\int_{\mathcal{B}} \rho \left(\frac{1}{2}\mathbf{v}\cdot\mathbf{v} + e\right) dv = \int_{\mathcal{B}} \rho\left(\mathbf{v}\cdot\mathbf{b} + r\right) dv + \int_{\partial\mathcal{B}} \left(\mathbf{v}\cdot\mathbf{T}\mathbf{n} + h\right) da \tag{2.36}$$

where e is the internal energy per unit mass, r is the rate at which heat is transferred to the material per unit mass, h is the heat flux per unit area through the boundary $\partial\mathcal{B}$, and all other variables are as before.

2.1.3 Constitutive Equations

As mentioned before, the conservation of energy is inherently satisfied by our formulation. Thus, we only need to worry about the conservation of mass and the balance of momentum. I.e., our starting point from here on out will be

$$\begin{aligned} \rho_0 &= \rho J, \quad \mathbf{X}\in\mathcal{B}_0, \\ \rho_0\ddot{\mathbf{u}} &= \operatorname{Div}\mathbf{P} + \rho_0\mathbf{b}, \quad \mathbf{X}\in\mathcal{B}_0, \end{aligned} \tag{2.37a}$$

$$\begin{aligned} \dot{\rho} + \rho\operatorname{div}\mathbf{v} &= 0, \quad \mathbf{x}\in\mathcal{B}, \\ \rho\dot{\mathbf{v}} &= \operatorname{div}\mathbf{T} + \rho\mathbf{b}, \quad \mathbf{x}\in\mathcal{B}, \end{aligned} \tag{2.38a}$$

$$\begin{aligned} \mathbf{u}(\mathbf{X},t) &= \bar{\mathbf{u}}(\mathbf{X},t), \quad \mathbf{X}\in\Gamma_{u_0}, \\ \mathbf{P}(\mathbf{X},t)\mathbf{N}(\mathbf{X}) &= \bar{\mathbf{p}}(\mathbf{X},t), \quad \mathbf{X}\in\Gamma_{q_0}, \end{aligned} \tag{2.37b}$$

$$\begin{aligned} \mathbf{v}(\mathbf{x},t) &= \bar{\mathbf{v}}(\mathbf{x},t), \quad \mathbf{x}\in\Gamma_{v}, \\ \mathbf{T}(\mathbf{x},t)\mathbf{n}(\mathbf{x},t) &= \bar{\mathbf{t}}(\mathbf{x},t), \quad \mathbf{x}\in\Gamma_{q}, \end{aligned} \tag{2.38b}$$

$$\begin{aligned} \mathbf{u}(\mathbf{X},0) &= \mathbf{u}_0(\mathbf{X}), \quad \mathbf{X}\in\bar{\mathcal{B}}_0, \\ \dot{\mathbf{u}}(\mathbf{X},0) &= \dot{\mathbf{u}}_0(\mathbf{X}), \quad \mathbf{X}\in\bar{\mathcal{B}}_0, \end{aligned} \tag{2.37c}$$

$$\mathbf{v}(\mathbf{x},0) = \mathbf{v}_0(\mathbf{x}), \quad \mathbf{x}\in\bar{\mathcal{B}}, \tag{2.38c}$$

where the equations on the left column are for problems formulated with respect to the reference configuration, and those on the right with respect to the current configuration. Note that $\mathcal{B}$ is the interior of the body excluding its boundary $\partial\mathcal{B}$, the domains Γ_u and Γ_q are disjointed subsets of the boundary, and $\bar{\mathcal{B}}$ denotes the closure of $\mathcal{B}$. I.e., $\Gamma_u \cap \Gamma_p = \varnothing$ and $\bar{\mathcal{B}} = \mathcal{B}\cup\partial\mathcal{B}$. The same holds true for the corresponding domains in the reference configuration.

It is clear that there are too many unknowns for this set of equations in either case. Thus, it is imperitive to provide additional equations in order to solve the problem. These additional equations are the constitutive equations, which characterize the properties of specific materials of interest. While the elegance of this field is phenomenal, we refrain from stating more than just the necessary results for this current work. Specifically, the constitutive equations for linear, neo-Hookean [18], and Guccione [19] solid materials are given. The constitutive equations on Newtonian fluids will not be covered, though FEniCS Mechanics is suited for solving these problems.

Before proceeding, we remind the reader that the constitutive equations for many elastic materials are often given in terms of a strain energy function, which can be shown to be related to the first Piola-Kirchhoff stress tensor by

$$\mathbf{P} = \frac{\partial \hat{\psi}}{\partial \mathbf{F}} = 2\mathbf{F}\frac{\partial \psi}{\partial \mathbf{C}}, \tag{2.39}$$

where $\hat{\psi}$ is the strain energy function expressed in terms of $\mathbf{F}$, and ψ is the same quantity re-parameterized in terms of $\mathbf{C}$. Keep in mind that ψ is the more commonly used form for reasons not discussed here.

Additionally, we would like to mention that many materials are modeled as incompressible, which is accomplished through a constraint of the form

$$G(\mathbf{C}, p) = \phi(\mathbf{C}) - \frac{1}{\kappa}p = 0, \tag{2.40}$$

where p is the hydrostatic pressure, and κ is the bulk modulus of the material. In this work, either

$$\phi(\mathbf{C}) = \frac{1}{2}\left(J-1\right)^2 \qquad \text{or} \qquad \phi(\mathbf{C}) = \frac{1}{2}\left(\ln J\right)^2 \tag{2.41}$$

is used.

2.1.3.1 Linear Elastic Material

The simplest solid material that one can consider is a linearly elastic material. This material is such that there is no need to distinguish between reference and current configurations given that the deformation can be approximated in a linear fashion. The constitutive equation in this equation is

$$\mathbf{T}(\mathbf{x}, t) = \lambda(\operatorname{tr}\boldsymbol{\varepsilon})\mathbf{I} + 2\mu\boldsymbol{\varepsilon}, \tag{2.42}$$

where

$$\boldsymbol{\varepsilon} = \frac{1}{2}\left[\frac{\partial \mathbf{u}}{\partial \mathbf{X}} + \left(\frac{\partial \mathbf{u}}{\partial \mathbf{X}}\right)^T\right]$$

is the symmetric part of the displacement gradient, and λ, μ are known as the first and second Lamé parameters, respectively. The Lamé parameters are related to the more familiar Young's modulus, E, and Poisson's ratio, ν, by

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \quad \text{and} \quad \mu = \frac{E}{2(1+\nu)}. \tag{2.43}$$

2.1.3.2 Neo-Hookean Material

The next material we consider is one that behaves linearly under shear deformation. Unlike a linear elastic material, however, we must maintain the distinction between the reference and current configurations. The strain energy function for this material is given by

$$\psi\left(\mathbf{C}\right) = \frac{1}{2}\mu\left(I_1\left(\mathbf{C}\right) - 3\right) + \frac{1}{2}\kappa\left(J-1\right)^2, \tag{2.44}$$

where μ is the shear modulus (same as the second Lamé parameter), and κ is the bulk modulus of the material.

This results in the first Piola Kirchhoff stress tensor

$$\mathbf{P} = \mu\mathbf{F} + \kappa J(J-1)\mathbf{F}^{-T}. \tag{2.45}$$

For computational purposes, the multiplicative decomposition $\mathbf{F} = \mathbf{F}_{\text{vol}}\bar{\mathbf{F}}$, where $\mathbf{F}_{\text{vol}} = J^{1/3}\mathbf{I}$ and hence $\bar{\mathbf{F}} = J^{-1/3}\mathbf{F}$, is used in order to presume an additive decomposition of the strain energy function

$$\psi(\mathbf{C}) = U(J) + \bar{\psi}\left(\bar{\mathbf{C}}\right) \tag{2.46}$$

in which the first component quantifies strain energy due to changes in volume, while the second quantifies strain energy due to isochoric deformation. This decomposition of the strain energy function is used extensively, e.g. [20, 21, 22]. It was originally formulated to model polymers [23], but has been postulated for tissue behavior as well. The resulting first Piola-Kirchhoff stress tensor with this decomposition is

$$\mathbf{P} = J^{-1/3}\left[J(J-1) - \frac{1}{3}\mu\,\mathrm{tr}\,\bar{\mathbf{C}}\right]\bar{\mathbf{F}}^{-T} + \mu J^{-1/3}\bar{\mathbf{F}}, \tag{2.47}$$

or, in terms of $\mathbf{F}$ and $\mathbf{C}$,

$$\mathbf{P} = \left[J(J-1) - \frac{1}{3}\mu J^{-2/3}\,\mathrm{tr}\,\mathbf{C}\right]\mathbf{F}^{-T} + \mu J^{-2/3}\mathbf{F}. \tag{2.48}$$

One benefit of the decomposition that can be concluded from Equation (2.46) is the ability to debug by "turning off" the volumetric component and ensuring that a constant volume is maintained.

2.1.3.3 Guccione Material

Another constitutive equation that is used was developed by Guccione et al. [19]. This models a transversely isotropic material with

$$\psi = \frac{1}{2}C\left(e^{Q} - 1\right) \tag{2.49}$$

as its strain energy function, where

$$Q = b_f E_{11}^2 + b_t \left(E_{22}^2 + E_{33}^2 + E_{23}^2 + E_{32}^2\right) + b_{fs} \left(E_{12}^2 + E_{21}^2 + E_{13}^2 + E_{31}^2\right). \tag{2.50}$$

Note that these components of the Lagrangian strain tensor,

$$\mathbf{E} = \frac{1}{2}\left(\mathbf{C} - \mathbf{I}\right),$$

are with respect to an orthonormal basis, say $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$, where $\mathbf{b}_1$ is the fiber direction, $\mathbf{b}_2$ is the direction normal to the fiber *within* the fiber plane, and $\mathbf{b}_3$ is the out-of-plane direction. This basis can vary continuously throughout the domain. Thus, b_f quantifies the stiffness in the fiber, while b_t quantifies it within the plane perpendicular to the fiber direction, and b_{fs} quantifies the shear stiffness in the fiber direction. On the other hand, the scalar C scales the quantity to give the stress.

Similar to the neo-Hookean material, the additive decomposition of the strain energy function in Equation (2.46) is used. This leads to the first Piola-Kirchhoff stress tensor

$$\mathbf{P} = \mathbf{P}_{\text{vol}} + \mathbf{P}_{\text{iso}}. \tag{2.51}$$

The stress due to isochoric deformation is given by

$$\mathbf{P}_{\text{iso}} = J^{-2/3}\mathbf{F}\hat{\mathbf{S}} - \frac{1}{3}J^{-2/3}\,\text{tr}(\mathbf{C}\hat{\mathbf{S}})\mathbf{F}^{-T}, \tag{2.52}$$

where

$$\begin{aligned}\hat{\mathbf{S}} = Ce^{\bar{Q}} \big[&b_f \bar{E}_{11}\mathbf{b}_1 \otimes \mathbf{b}_1 + b_t \bar{E}_{22}\mathbf{b}_2 \otimes \mathbf{b}_2 + b_t \bar{E}_{33}\mathbf{b}_3 \otimes \mathbf{b}_3 \\ &+ b_{fs}\bar{E}_{12}\left(\mathbf{b}_1 \otimes \mathbf{b}_2 + \mathbf{b}_2 \otimes \mathbf{b}_1\right) + b_t \bar{E}_{23}\left(\mathbf{b}_2 \otimes \mathbf{b}_3 + \mathbf{b}_3 \otimes \mathbf{b}_2\right) \\ &+ b_{fs}\bar{E}_{13}\left(\mathbf{b}_1 \otimes \mathbf{b}_3 + \mathbf{b}_3 \otimes \mathbf{b}_1\right)\big].\end{aligned} \tag{2.53}$$

On the other hand, the stress due to changes in volume is given by

$$\mathbf{P}_{\text{vol}} = \kappa(\ln J)\mathbf{F}^{-T}, \tag{2.54}$$

for a compressible material where we used

$$U(J) = \frac{1}{2}\kappa\left(\ln J\right)^2$$

in this case. If the material is specified as incompressible, the volumetric portion of the stress is given by

$$\mathbf{P}_{\text{vol}} = -Jp\mathbf{F}^{-T}, \tag{2.55}$$

where p is the pressure and serves as the Lagrange multiplier for the incompressibility constraint.

2.2 Discretizing the governing equations

Now that the governing equations for the motion and deformation of solid materials has been developed, we must discretize them in order to obtain approximate solutions. This is necessary since analytical solutions to these equations are only possible in very specialized cases. Here, we will use a combination of the Finite Element Method (FEM) and Finite Differences (FD) for approximating solutions to the resulting IBVP. This will, however, only provide a brief overview of the method, and thus the reader is referred to [17] for a more detailed discussion.

2.2.1 Variational form of PDEs

The FEM begins with a variational form of the governing equations. Such a form is typically "derived" by integrating Equation (2.31) over the domain of the entire body, $\mathcal{B}$, which is then integrated by parts. Though there is a minor flaw in this narrative that ends up correcting itself. This process should be considered a *motivation* for the variational form rather than its *derivation*. This is due to the fact that a certain level of smoothness is necessary to integrate by parts, but is then taken away in the interest of *weakening* what is asked of the solution.

With this is in mind, consider the variational form

$$\int_{\mathcal{B}} \boldsymbol{\xi} \cdot \rho \dot{\mathbf{v}} \, dv + \int_{\mathcal{B}} \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{x}} \cdot \mathbf{T} \, dv - \int_{\mathcal{B}} \boldsymbol{\xi} \cdot \rho \mathbf{b} \, dv - \int_{\Gamma_q} \boldsymbol{\xi} \cdot \bar{\mathbf{t}} \, da = 0, \tag{2.56}$$

where $\boldsymbol{\xi}$ is what is commonly referred to as a vector-valued test function. In Calculus of Variations, this will be referred to as the first variation of all possible solutions. It can be shown that a smooth enough solution to this form, where $\boldsymbol{\xi} = \mathbf{0}$ for $\mathbf{x} \in \Gamma_u$, will also be a solution to Equation (2.31). Note that this variational form is given with respect to the current configuration of a body, but does not include the conservation of mass given by

$$\int_{\mathcal{B}} \zeta \left(\dot{\rho} + \rho \operatorname{div} \mathbf{v} \right) dv = 0, \tag{2.57}$$

where ζ is a scalar-valued test function. Different variational forms *motivated* by the above can be formulated to be better suited for boundary conditions. They can also be shown to be equivalent when sufficient smoothness of the solution and test functions is assumed.

However, we will make exclusive use of the governing equations formulated with respect to the reference configuration of the body. In this case, we have

$$\int_{\mathcal{B}_0} \boldsymbol{\xi} \cdot \rho_0 \ddot{\mathbf{u}} \, dV + \int_{\mathcal{B}_0} \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{X}} \cdot \mathbf{P} \, dV - \int_{\mathcal{B}_0} \boldsymbol{\xi} \cdot \rho_0 \mathbf{b} \, dV - \int_{\Gamma_{q_0}} \boldsymbol{\xi} \cdot \bar{\mathbf{p}} \, dA = 0, \tag{2.58}$$

where, once again, a smooth enough solution to this variational form will also be a solution to Equation (2.33). While the conservation of energy needs to be included in most cases formulated with respect to the current configuration, it can be shown that this is not the case

with respect to the reference configuration. This is due to the fact that a surface within $\mathcal{B}_0$ is also a material surface. Thus the mass flux across any such surface will be zero. Therefore, the conservation of mass is automatically satisfied by the fact that the variational form is expressed with respect to the reference configuration.

2.2.2 Finite element formulation for solid mechanics

Now that we have an equivalent variational form for the governing equations of solid mechanics, Equation (2.58), we can discuss a bit more of the specifics. Namely that the utility of the FEM is realized once the geometry is discretized. Furthermore, if the true solution belongs to a function space $\mathcal{F}$, the FEM uses a subspace of this such that it is spanned by a basis with finite cardinality, which we will denote by $\mathcal{F}^h$. We take $\{\eta_i\}_{i=1}^n$ to be the set of basis functions where $n \in \mathbb{N}$. In our work, we will make use of polynomial basis functions with order p.

Now, for solid mechanics problems formulated with respect to the reference configuration, the goal is to solve for the displacement field as a function of referential coordinates, and potentially time. We can then represent an approximate solution $\mathbf{u}^h \in \left(\mathcal{F}^h\right)^3$ as a linear combination of the basis functions, i.e.

$$\mathbf{u}^h(\mathbf{X}, t) = \sum_{i=1}^{N_n} \hat{\mathbf{u}}_i(t)\eta_i(\mathbf{X}), \tag{2.59}$$

where $\hat{\mathbf{u}}_i$ are the coefficients of the approximation, and N_n is the cardinality of $\mathcal{F}^h$. We can substitute this approximation of $\mathbf{u}$ into Equation (2.58) and obtain a system of ordinary differential equations (ODEs) in which we will solve for $\hat{\mathbf{u}}_i(t)$. We will write this system of ODEs as

$$\begin{aligned} M\ddot{u} + R(u, p) &= F_b(t) + F_q(u, t), \\ G(u, p) &= 0 \end{aligned} \tag{2.60}$$

where u and p (when incompressibility constraint is applied) are now column matrices storing the coefficient values $\hat{\mathbf{u}}_i$ and $\hat{p}_i$, respectively, for all $i = 1, \ldots, N_n$ at time t. I.e.,

$$\begin{aligned} u &= \left[\hat{u}_0, \hat{v}_0, \hat{w}_0, \ldots, \hat{u}_i, \hat{v}_i, \hat{w}_i, \ldots, \hat{u}_{N_n}, \hat{v}_{N_n}, \hat{w}_{N_n}\right]^T, \\ p &= \left[\hat{p}_0, \hat{p}_1, \ldots, \hat{p}_i, \ldots, \hat{p}_{N_n}\right]. \end{aligned} \tag{2.61}$$

Note that

$$\hat{\mathbf{u}}_i = \hat{u}_i\mathbf{E}_x + \hat{v}_i\mathbf{E}_y + \hat{w}_i\mathbf{E}_z,$$

where $\mathbf{E}_x, \mathbf{E}_y, \mathbf{E}_z$ are the orthonormal basis vectors in the reference configuration.

Furthermore, $R(u, p)$ is the column matrix that results from the stress tensor term in Equation (2.58). In general, this term is not linear. The terms $F_b(t)$ and $F_q(u, t)$ are the

column matrices resulting from the body and traction force terms, respectively. Note that F_q is dependent on u by its use of Nanson's formula,

$$\mathbf{n}\, da = J\mathbf{F}^{-T}\mathbf{N}\, dA, \tag{2.62}$$

which helps convert surface integrals between the reference and current configurations.

2.2.3 Finite difference methods

At this point, the derivatives with respect to spatial variables have been taken care of, and hence we need to approximate the derivatives with respect to time. For the system of second-order ODEs resulting from the FEM formulation of solid mechanics, we will be using the Newmark scheme [24], which is a finite difference method for second order ODEs. As can be seen in Equation (2.58), the only term that needs to be approximated using a finite difference is u. The Newmark scheme gives

$$\dot{u}_{n+1} = \dot{u}_n + \Delta t\left[(1-\gamma)\ddot{u}_n + \gamma\ddot{u}_{n+1}\right], \tag{2.63a}$$

$$u_{n+1} = u_n + \dot{u}_n\Delta t + \frac{1}{2}(\Delta t)^2\left[(1-2\beta)\,\ddot{u}_n + 2\beta\ddot{u}_{n+1}\right], \tag{2.63b}$$

or alternatively,

$$v_{n+1} = v_n + \Delta t\left[(1-\gamma)a_n + \gamma a_{n+1}\right], \tag{2.64a}$$

$$u_{n+1} = u_n + v_n\Delta t + \frac{1}{2}(\Delta t)^2\left[(1-2\beta)\,a_n + 2\beta a_{n+1}\right], \tag{2.64b}$$

where $v_n = \dot{u}_n$, and $a_n = \ddot{u}_n$, and γ, β are the Newmark scheme parameters. Typical values used are $\gamma = 1/2$ and $\beta = 1/4$. Now with this finite difference scheme, we can rewrite Equation (2.60) as

$$\begin{aligned} Ma_{n+1} = {} & \theta\left[F_b(t_{n+1}) + F_q(t_{n+1}) - R(u_{n+1}, t_{n+1})\right] \\ & + (1-\theta)\left[F_b(t_n) + F_q(t_n) - R(u_n, t_n)\right], \end{aligned} \tag{2.65}$$

where $\theta \in [0,1]$ specifies whether the problem is integrated in time explicitly, implicitly, or a combination of both. After this equation is solved for a_{n+1}, we can compute u_{n+1} and v_{n+1} with Equations (2.64b) and (2.64a), respectively.

2.3 FEniCS Mechanics

The formulation of the resulting IBVP from continuum mechanics and the computation of its numerical solution through the FEM was streamlined with FEniCS Mechanics (FM)—a Python package built on top of the tools provided by the FEniCS Project [25]. While there are many options for FEM modeling [26], FEniCS was chosen as the backend because it is open-source, widely-used, well-supported, and has broad capabilities that are leveraged by the FM package developed.

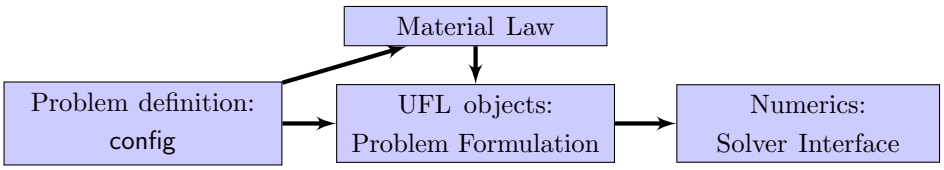


Figure 2.1: Diagram indicating the flow of information in FM from the user-defined `config` dictionary to the written output files.

2.3.1 Motivation

FEniCS, the backend to FM, was designed to solve problems that can be formulated in variational form. Thus, it is the perfect tool for problems that arise from continuum mechanics; specifically when solved using the FEM method. Using FEniCS to solve such IBVPs, one must specify the variational form of the PDEs, function space, finite element types, solver settings, etc., through scripts. It is, however, possible to formulate a range of continuum mechanics problems similarly, as is shown by Equations (2.56) and (2.58) where no mention of material behavior or properties was yet made. This is the basis for FM.

With FM, a variety of continuum mechanics problems can be formulated and solved through minor changes to a configuration file, or potentially "minimally-invasive" changes to scripts that define a generalized mechanics problem. This results in an efficient framework to consider a variety of mechanics problems, or testing of various modeling choices, e.g. types of materials or boundary conditions. Overall, this package increases accessibility to computational mechanics simulations for users with minimal programming knowledge, while still maintaining a powerful and extensive open-source framework that can access all of the capabilities provided through the FEniCS Project.

At the current state, FM supports both steady-state and time-dependent problems in a single domain, and the user can choose from a list of implemented material models, or provide their own, so long as the material is elastic (stress depends on the deformation gradient) or viscous (stress depends on the velocity gradient). The discretization in time is handled by single-step finite-difference schemes, including the θ-method for first order systems, and the Newmark scheme (covered in Section 2.2.3).

2.3.2 Code structure

The overall flow of information in FM is shown in Figure 2.1. The user defines the problem to be solved through nested Python dictionaries, which we will refer to as `config`. The sub-dictionaries are named `mesh`, `material`, and `formulation`. In these, the user specifies the material model, time integration parameters, domain, BCs, and the file where the mesh is stored. Specific examples will be shown in Section 2.4, though the full documentation can be found in [27]. This `config` dictionary is parsed as part of the instantiation of a problem class.

Abbrev.	Full Name	Abbrev.	Full Name
BMP	`BaseMechanicsProblem`	EM	`ElasticMaterial`
MP	`MechanicsProblem`	IM	`IsotropicMaterial`
FMP	`FluidMechanicsProblem`	LIM	`LinearIsoMaterial`
SMP	`SolidMechanicsProblem`	NHM	`NeoHookeMaterial` [18]
MBS	`MechanicsBlockSolver`	AM	`AnisotropicMaterial`
NVS	`NonlinearVariationalSolver`	FM	`FungMaterial` [28]
BMS	`BaseMechanicsSolver`	GM	`GuccioneMaterial` [19]
SMS	`SolidMechanicsSolver`	F	`Fluid`
FMS	`FluidMechanicsSolver`	NF	`NewtonianFluid`

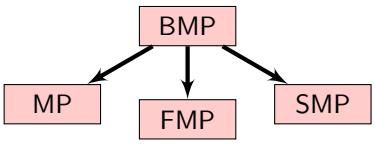
Table 2.1: Abbreviations of some of the classes defined within FEniCS Mechanics. Note that the `NonlinearVariationalSolver` class is defined in `dolfin` from the FEniCS project.

We will now use the abbreviations shown in Table 2.1. As shown in Figure 2.2, all problem classes are child classes of the BMP class. This base class provides methods common to all mechanics problems, including the parsing of the `config` dictionary. Key-value pairs provided in `config` and their combinations are checked by the parsing process to ensure that problem definitions are physically consistent.

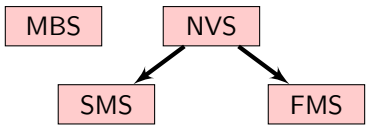
Once `config` is parsed, the respective problem class—currently MP, SMP, or FMP—defines the variational equations for the respective problem using the UFL. The MP class defines the variational problem with separate function spaces for vector- and scalar-valued field variables, whereas the SMP and FMP classes use the mixed function space functionality of `dolfin`. The information pertaining to material models in `config` is passed to separate classes defining constitutive equations, as can be seen in Figure 2.1.

With the variational equations defined through the UFL and stored as member data of a problem object, a solver object is next created. The three current solver classes are MBS, SMS, and FMS, and are to be used with MP, SMP, and FMP, respectively. These solver objects have methods that use the UFL forms from the problem objects to assemble the resulting linear algebraic system at each iteration of a nonlinear solve. This is repeated for each time step if the problem is time-dependent. Note that SMS and FMS are a subclass of the NVS class from `dolfin` through the BMS class, while `MBS` is a stand-alone block solver class based on the FEniCS Application, CBC-Block [29], as shown in Figure 2.2b.

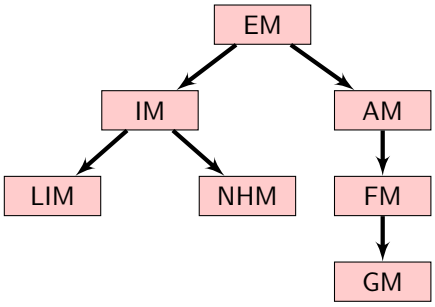
The user is expected to interact the most with the problem and solver classes mentioned above. However, in addition to these, various constitutive models have been implemented in a `materials` sub-module within FM. These constitutive models and their inheritance trees



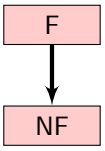
(a) Classes to define the variational problem using the UFL.



(b) Classes to assemble the variational forms defined by problem objects, and call the numerical solvers.



(c) Classes that define the constitutive equations for different elastic materials initiated in SMP or MP.



(d) Classes that define the constitutive equations for fluids initiated in FMP or MP.

Figure 2.2: Class inheritance tree for classes defined within FEniCS Mechanics.

are shown in Figures 2.2c and 2.2d. The design of FM also allows the user to define their own constitutive equation. In order to do this, the user must define a class with expressions of the stress tensor. Further details on this procedure can be found on [27].

In summary, FM facilitates the problem formulation and solving by providing three overarching functionalities:

1. Key-value pairs provided, and their combinations, are checked for validity. This increases accessibility by making sure that invalid, or inconsistent, values in `config` are not used.

2. FM uses the problem specification provided in `config` to define the variational form using the UFL.

3. The variational form defined through the UFL is used to assemble the resulting linear systems and obtain a numerical solution to the problem.

The next section will demonstrate all of these functionalities with four different examples.

2.4 Applications and examples

As previously mentioned, FM parses a Python dictionary, `config`, in which the user defines the computational mechanics problem they wish to solve. Various examples of solid mechanics problems solved with FM will be shown in this section.

2.4.1 Three-dimensional Box

For the first case, we formulate a solid mechanics problem on a 3D box domain subject to a time-dependent load. Consider a three-dimensional box domain denoted by

$$\mathcal{R}_0 = \left\{ \mathbf{X} \in \mathcal{E}_3 \;\middle|\; (X_1, X_2, X_3) \in [0, L_x] \times [0, L_y] \times [0, L_z] \right\}, \tag{2.66}$$

with $L_x = 10$ mm, $L_y = 2$ mm, $L_z = 4$ mm, and two boundary regions,

$$\Gamma_{u_0} = \left\{ \mathbf{X} \in \partial\mathcal{R}_0 \;\middle|\; X_1 = 0 \right\}, \tag{2.67a}$$

and

$$\Gamma_{q_0} = \left\{ \mathbf{X} \in \partial\mathcal{R}_0 \;\middle|\; X_1 = 10 \right\}. \tag{2.67b}$$

We model this as a nearly incompressible neo-Hookean material with a bulk modulus of $\kappa = 10^6$ [Pa or g/mm^2], a shear modulus of $\mu = 1333/9.242 = 144.23$ [Pa or g/mm^2], and a density of $\rho_0 = 1055$ [kg/m^3] $= 1.055 \times 10^{-3}$ [g/mm^3]. The bulk and shear moduli values were chosen to be as compatible as possible to Augustin et al. [20], though an exponential type of strain energy function is used there, while the density was taken from Gheorge et al. [30].

The time-dependent pressure

$$\bar{\mathbf{t}} = -\bar{p}(t)\mathbf{n} \quad \Rightarrow \quad \bar{\mathbf{p}} = -\bar{p}(t) J \mathbf{F}^{-T} \mathbf{N}, \tag{2.68a}$$

with

$$\bar{p}(t) = \frac{1}{2}\bar{p}_0 \left[\cos(2\pi t) - 1\right] \tag{2.68b}$$

is applied at $X_1 = 10$, where $\bar{p}_0 = 100$ [Pa or g/mm^2]. The free-body diagram as viewed from the XY-plane for this problem is shown in Figure 2.3, where the traction force, $\bar{\mathbf{p}}$, is distributed over the entire surface Γ_{q_0}. The resulting variational form of the governing equations is then

$$\int_{\mathcal{R}_0} \boldsymbol{\xi} \cdot \rho_0 \ddot{\mathbf{u}} \, dV + \int_{\mathcal{R}_0} \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{X}} \cdot \mathbf{P} \, dV - \int_{\Gamma_{q_0}} \boldsymbol{\xi} \cdot \bar{\mathbf{p}} \, dA - \int_{\mathcal{R}_0} \zeta \left(\frac{1}{\kappa} p - \phi(\mathbf{C}) \right) dV = 0, \tag{2.69}$$

such that $\mathbf{u} = \mathbf{0}$ for $\mathbf{X} \in \Gamma_{u_0}$.

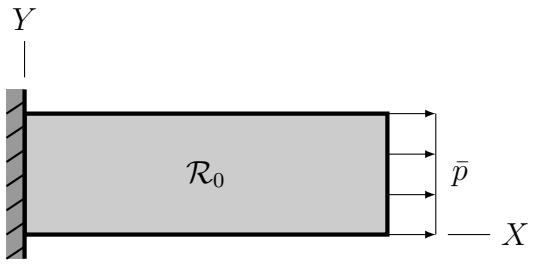


Figure 2.3: Free-body diagram of the steady-state elongation of a unit square.

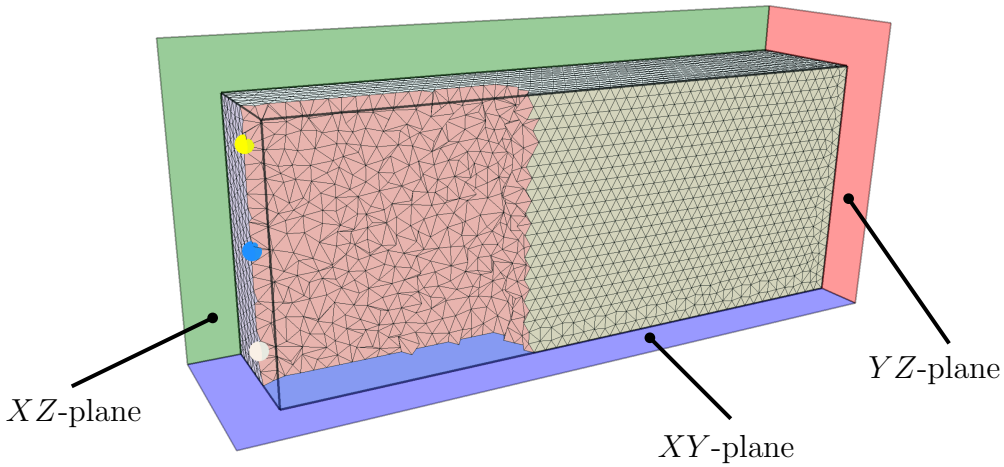


Figure 2.4: Tetrahedral mesh of the 3D box domain with 9,532 nodes and 46,448 tetrahedral cells.

The mesh used to solve this problem is shown in Figure 2.4, along with three observation points, namely

$$\mathbf{r}_0 = 10\mathbf{E}_x + \frac{11}{10}\mathbf{E}_y + \frac{1}{2}\mathbf{E}_z, \tag{2.70a}$$

$$\mathbf{r}_1 = 10\mathbf{E}_x + \frac{11}{10}\mathbf{E}_y + 2\mathbf{E}_z, \tag{2.70b}$$

and

$$\mathbf{r}_0 = 10\mathbf{E}_x + \frac{11}{10}\mathbf{E}_y + \frac{7}{2}\mathbf{E}_z, \tag{2.70c}$$

where all components are given in millimeters. The Python script to formulate and solve this problem is shown below:

Python code

```python
import fenicsmechanics as fm

# Region IDS
ALL_ELSE = 0
LEFT = 1
RIGHT = 2
BOTTOM = 3
TOP = 4

# Limit the amount of printed messages to make logs more readable.
rank = fm.MPI.rank(fm.MPI.comm_world)
if rank > 0:
    dlf.set_log_level(fm.LogLevel.ERROR)

# Density from:
#
#     Cardiac left ventricular myocardial tissue density, evaluated by
    computed
#     tomography and autopsy, by A.G. Gheorge et al. (2019)
#

material_dict = {
    "type": "elastic",
    "const_eqn": "neo_hookean",
    "incompressible": True,
    "density": 1.055e-3, # g/mm^3
    "mu": 1333.0/9.242,  # g/mm*s^2
    "kappa": 1e6,        # g/mm*s^2
}

mesh_dict = {
    "mesh_file": "mesh/box-domain.h5",
    "boundaries": "mesh/box-domain.h5"
}

# 2e3 g/(mm*s^2) is approximately 15 mmHg
p_max = 100.0 # g/mm*s^2
a = p_max/2.0

p_inner = "{a}*(cos(2.0*pi*t) - 1.0)".format(a=a)

formulation_dict = {
    "time": {
        "dt": 0.01,
        "interval": [0.0, 4.0]
    },
    "element": "p2-p1",
    "domain": "lagrangian",
    "bcs": {
```

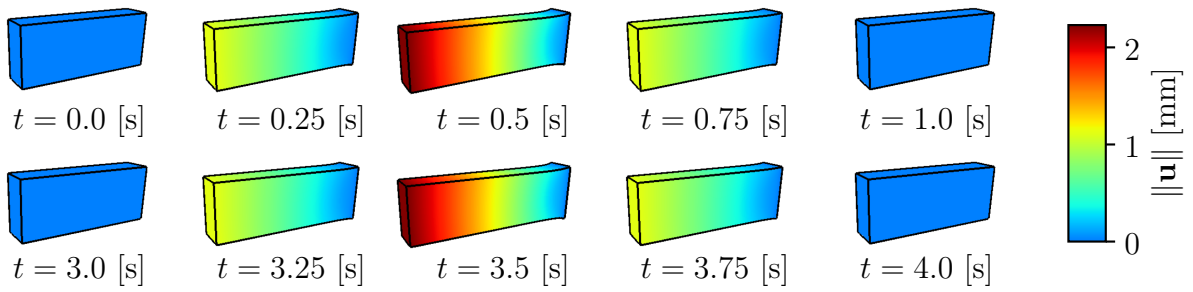
```
        "dirichlet": {
            "displacement": [[0.0, 0.0, 0.0]],
            "regions": [LEFT],
            "components": ["all"]
        },
        "neumann": {
            "regions": [RIGHT],
            "types": ["pressure"],
            "values": [p_inner]
        }
    }
}

config = {
    "mesh": mesh_dict,
    "material": material_dict,
    "formulation": formulation_dict
}

fname_hdf5 = "results/hdf/displacement.h5"
fname_disp = "results/vtk/displacement.pvd"
fname_pressure = "results/vtk/pressure.pvd"
fname_config = "results/vtk/config.log"

problem = fm.SolidMechanicsProblem(config)
problem.write_config(fname_config)

solver = fm.SolidMechanicsSolver(problem, fname_disp=fname_disp,
                                 fname_pressure=fname_pressure,
                                 fname_hdf5=fname_hdf5)

solver.full_solve()
```

As can be seen from the problem configuration, the deformation is simulated for four full cycles of the loading function. The resulting deformation of the full domain during the first and last loading cycle are shown in Figure 2.5a. Additionally, the components of displacement at the three observation points given in Equation (2.70) are shown in Figures 2.5b-2.5c. Given the problem formulation, including the geometry and boundary conditions, the vector-valued displacement field exhibits symmetry with respect to the plane $Z = l_z/2$. This can be seen by the fact that the u_3 component evaluated at $\mathbf{r}_2$ is the negative of the u_3 component at $\mathbf{r}_0$.

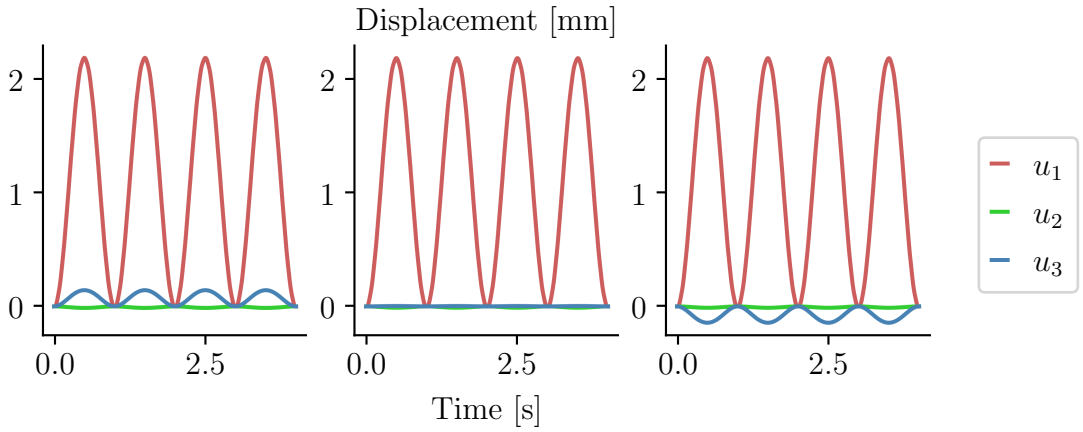
2.4.2 Thick-walled Cylinder

For the next example, consider a thick-walled cylinder

$$\mathcal{D}_0 = \left\{ \mathbf{X} \in \mathcal{E}_3 \;\middle|\; R_1^2 < X_1^2 + X_2^2 < R_2^2,\ -L/2 < X_3 < L/2 \right\}, \tag{2.71}$$



(a) Deformation of the box domain during the first and last loading cycle.



(b) Evaluted at $\mathbf{r}_0$ (c) Evaluated at $\mathbf{r}_1$ (d) Evaluated at $\mathbf{r}_2$

Figure 2.5: Displacement of the box-domain.

where $R_1 = 0.65$, $R_2 = 1.1$, and $L = 1$. Additionally, the surfaces

$$\mathcal{S}^{\mathcal{D}}_{\text{in}} = \left\{ \mathbf{X} \in \partial\mathcal{D}_0 \;\middle|\; X_3 = -L/2 \right\}, \tag{2.72a}$$

$$\mathcal{S}^{\mathcal{D}}_{\text{out}} = \left\{ \mathbf{X} \in \partial\mathcal{D}_0 \;\middle|\; X_3 = L/2 \right\}, \tag{2.72b}$$

$$\mathcal{S}^{\mathcal{D}}_{\text{inner}} = \left\{ \mathbf{X} \in \partial\mathcal{D}_0 \;\middle|\; X_1^2 + X_2^2 = R_1^2 \right\}, \tag{2.72c}$$

$$\mathcal{S}^{\mathcal{D}}_{YZ} = \left\{ \mathbf{X} \in \bar{\mathcal{D}}_0 \;\middle|\; X_1 = 0 \right\}, \tag{2.72d}$$

and

$$\mathcal{S}^{\mathcal{D}}_{XZ} = \left\{ \mathbf{X} \in \bar{\mathcal{D}}_0 \;\middle|\; X_2 = 0 \right\}, \tag{2.72e}$$

are regions on which BCs or constraints will be imposed. Note that the superscript $\mathcal{D}$ denotes the fact that these are sub-domains of $\mathcal{D}_0$, while $\bar{\mathcal{D}}_0$ refers to the closure of $\mathcal{D}_0$. The BCs

imposed are

$$\mathbf{u}(\mathbf{X},t)\cdot\mathbf{E}_z = u_Z(\mathbf{X},t) = 0 \quad \text{for } \mathbf{X}\in\mathcal{S}_{\text{in}}\cup\mathcal{S}_{\text{out}}, \tag{2.73a}$$
$$\mathbf{u}(\mathbf{X},t)\cdot\mathbf{E}_x = u_X(\mathbf{X},t) = 0 \quad \text{for } \mathbf{X}\in\mathcal{S}_{YZ}, \tag{2.73b}$$
$$\mathbf{u}(\mathbf{X},t)\cdot\mathbf{E}_y = u_Y(\mathbf{X},t) = 0 \quad \text{for } \mathbf{X}\in\mathcal{S}_{XZ}, \tag{2.73c}$$

and

$$\bar{\mathbf{p}}(\mathbf{X},t) = -\bar{p}(t)J\mathbf{F}^{-T}\mathbf{N} \quad \text{for } \mathbf{X}\in\mathcal{S}_{\text{inner}}, \tag{2.73d}$$

where

$$\bar{p}(t) = \frac{1}{2}\bar{p}_0\left(1-\cos(2\pi t)\right), \tag{2.73e}$$

with $\bar{p}_0 = 700$ kPa. The surfaces $\mathcal{S}_{YZ}$ and $\mathcal{S}_{XZ}$, though internal surfaces, are defined in order to constrain the solution, $\mathbf{u}$, such that it is axisymmetric. The mesh was generated so that the finite elements conform across these surfaces, i.e. the facets of the nearby tetrahedral cells coincide with the surfaces, and is shown in Figure 2.6. Unlike the others, the last BC shown in Equation (2.73d) is a traction BC.

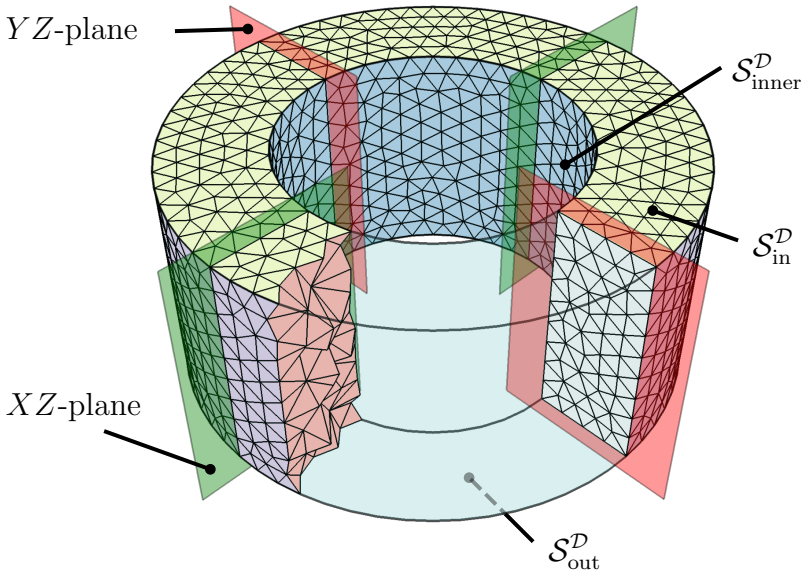


Figure 2.6: Tetrahedral mesh of the thick-walled cylinder, $\mathcal{D}_0$, with cells conforming to the YZ and XZ planes. This mesh contains 2,483 nodes and 9,654 tetrahedral cells.

Though we subject this body to a time-dependent loading function, we assume a quasi-

static deformation. Thus, the resulting variational form for this problem is

$$\int_{\mathcal{D}_0} \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{X}} \cdot \mathbf{P}\, dV + \bar{p}(t) \int_{\mathcal{S}^{\mathcal{D}}_{\text{inner}}} \boldsymbol{\xi} \cdot \left(J\mathbf{F}^{-T}\mathbf{N}\right) dA, - \int_{\mathcal{D}_0} \zeta \left(\frac{1}{\kappa} p - \phi(\mathbf{C})\right) dV = 0, \tag{2.74}$$

where we made use of the fact that $\bar{p}(t)$ is not dependent on spatial variables and hence can be taken out of the integral. Here, we use the model for a nearly incompressible neo-Hookean material with $\kappa = 10$ GPa, and $\mu = 1.5$ MPa. Note that the density of the material is not used here due to the fact that the inertial term has been neglected.

The Python script to solve this problem is shown below:

Python code

```
import fenicsmechanics as fm

material = {
    "type": "elastic",
    "const_eqn": "neo_hookean",
    "incompressible": True,
    "density": 0.0,
    "kappa": 10e9, # Pa
    "mu": 1.5e6 # Pa
}

mesh = {
    "mesh_file":  "mesh/thick-walled-cylinder.h5",
    "boundaries": "mesh/thick-walled-cylinder.h5"
}

p_max = 7e5
p_expr = "0.5*{p_max}*(1.0 - cos(2.0*pi*t))".format(p_max=p_max)

# Region IDs defined during mesh generation
INNER_WALL = 1
INLET = 3
OUTLET = 4
YZ_PLANE = 5
XZ_PLANE = 6

formulation = {
    "element": "p2-p1",
    "domain": "lagrangian",
    "time": {
        "dt": 0.01,
        "interval": [0.0, 1.5]
    },
    "bcs": {
        "dirichlet": {
            "displacement": [0.0, 0.0, 0.0, 0.0],
            "regions": [INLET, OUTLET, YZ_PLANE, XZ_PLANE],
```

```
            "components": ["z", "z", "x", "y"]
        },
        "neumann": {
            "values": [p_expr],
            "regions": [INNER_WALL],
            "types": ["pressure"]
        }
      }
    }

config = {
    "material": material,
    "mesh": mesh,
    "formulation": formulation
}

fname_config = "results/config.log"
fname_hdf = "results/hdf/displacement.h5"
fname_disp = "results/vtk/displacement.pvd"

problem = fm.SolidMechanicsProblem(config)
problem.write_config(fname_config)

solver = fm.SolidMechanicsSolver(problem, fname_disp=fname_disp)
solver.set_parameters(newton_abstol=1e-6, newton_reltol=1e-8)
solver.full_solve()
```

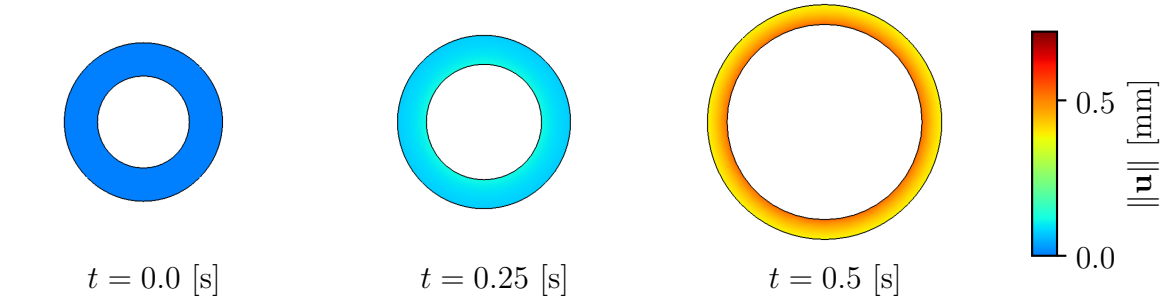
This deformation is simulated for 1.5 cycles of the loading function, which we will make use of in Chapter 4. The resulting deformations at $t = 0, 0.25, 0.5$ s are shown in Figure 2.7a. Here, we can see the axial symmetry in the magnitude of the displacement. The displacement vector was also evaluated at

$$\mathbf{r}_0 = \left(\frac{R_1 + R_2}{2}\right)\left[\cos\left(\frac{\pi}{4}\right) + \sin\left(\frac{\pi}{4}\right)\right], \tag{2.75}$$

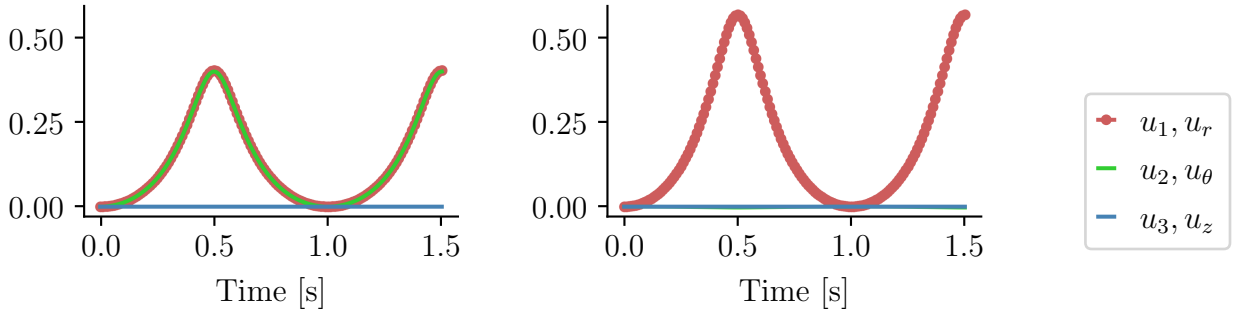
and the cartesian and cylindrical components are shown in Figure 2.7b and 2.7c, respectively. This further confirms that the restrictions we specified in Equations (2.73b) and (2.73c) helped maintain the axial symmetry by the fact that the u_r component is the only one of the cylindrical components that is non-zero for all $t \in [0, 1.5]$.

2.4.3 Thick-walled Ellipsoids

Last but not least, we will consider two thick-walled ellipsoids, which are often considered to be an idealized geometry of the human left ventricle [31]. As in the case of the thick-walled cylinder, we will subject the body to a time-dependent loading function while assuming a quasi-static formulation to avoid unwanted wave reflections. Thus, the variational form for these two examples is the same as Equation (2.74) when $\mathcal{D}_0$ is replaced with the ellipsoidal volume and $\mathcal{S}^{\mathcal{D}}_{\text{inner}}$ is replaced by the inner surface of the thick-walled ellipsoid.



(a) Deformation of the thick-walled cylinder at $z = 0$



(b) Cartesian components (u_1, u_2, u_3) (c) Cylindrical components (u_r, u_θ, u_z)

Figure 2.7: Displacement of the thick-walled cylinder.

The geometries for both cases that we consider are constructed in a similar fashion. Thus, we will go through a general construction where the (i, j) pair will help us distinguish between different entities used. Specifically, we will use $i = 1$ to identify an axisymmetric thick-walled ellipsoid, and $i = 2$ to identify an eccentrically aligned one. Furthermore, $j = 1$ will identify the inner surface, while $j = 2$ will identify the outer surface, in both cases (axisymmetric and eccentric).

To begin, let us define three sub-domains of $\mathcal{E}_3$. Namely,

$$\mathcal{K}_j^i = \left\{ \mathbf{X} \in \mathcal{E}_3 \;\middle|\; \mathbf{X} \cdot \mathbf{A}_j^i \mathbf{X} < 1 \right\}, \tag{2.76a}$$

where $\mathbf{A}_j^i = \mathbf{Q}_j^i \mathbf{D}_j^i \left(\mathbf{Q}_j^i\right)^T$, and

$$\mathbf{D}_j^i = \left(\frac{1}{a_j^i}\right)^2 \left(\mathbf{l}_j^i \otimes \mathbf{l}_j^i\right) + \left(\frac{1}{b_j^i}\right)^2 \left(\mathbf{m}_j^i \otimes \mathbf{m}_j^i\right) + \left(\frac{1}{c_j^i}\right)^2 \left(\mathbf{E}_z \otimes \mathbf{E}_z\right) \tag{2.76b}$$

		Inner ($j = 1$)				Outer (j = 2)			
Type	i	a_j^i	b_j^i	c_j^i	θ_j^i (deg)	a_j^i	b_j^i	c_j^i	θ_j^i (deg)
Axisymmetric	1	7	7	17	0	10	10	20	0
Eccentric	2	6	6.4	14	20	10	11	20	0

Table 2.2: Axes lengths and rotations used to construct the geometries for the axisymmetric and eccentric cases.

is the tensor defining the quadric surface with respect to its principal directions, $\left\{\mathbf{l}_j^i, \mathbf{m}_j^i, \mathbf{E}_z\right\}$, and

$$\mathbf{Q}_j^i = \mathbf{E}_z \otimes \mathbf{E}_z + (\mathbf{E}_x \otimes \mathbf{E}_x + \mathbf{E}_y \otimes \mathbf{E}_y) \cos\theta_j^i - (\mathbf{E}_x \otimes \mathbf{E}_y - \mathbf{E}_y \otimes \mathbf{E}_x) \sin\theta_j^i \tag{2.76c}$$

is a rotation of θ_j^i about the Z-axis. Additionally,

$$\mathcal{Z}^i = \left\{ \mathbf{X} \in \mathcal{E}_3 \;\middle|\; \mathbf{X} \cdot \mathbf{E}_z = X_3 < Z^i \right\}. \tag{2.76d}$$

Note that $\mathcal{K}_j^i$ is a volume encompassed by an ellipsoidal surface, while $\mathcal{Z}^i$ is anything *below* the plane $X_3 = Z^i$. The volumetric domain of interest is then defined by

$$\mathcal{V}^i = \mathcal{Z}^i \cap \left(\mathcal{K}_2^i - \mathcal{K}_1^i\right). \tag{2.77}$$

The boundary surfaces are given by

$$\mathcal{S}_{\text{inner}}^i = \partial\mathcal{K}_1^i \cap \bar{\mathcal{Z}}^i, \tag{2.78a}$$

and

$$\mathcal{S}_{\text{base}}^i = \partial\mathcal{Z}^i \cap \left(\bar{\mathcal{K}}_2^i - \bar{\mathcal{K}}_1^i\right), \tag{2.78b}$$

where $\bar{\mathcal{Z}}^i$, $\bar{\mathcal{K}}_1^i$, and $\bar{\mathcal{K}}_2^i$ refer to the closure of the sets $\mathcal{Z}^i$, $\mathcal{K}_1^i$, and $\mathcal{K}_2^i$, respectively. The values of the constants for each case are given in Table 2.2.

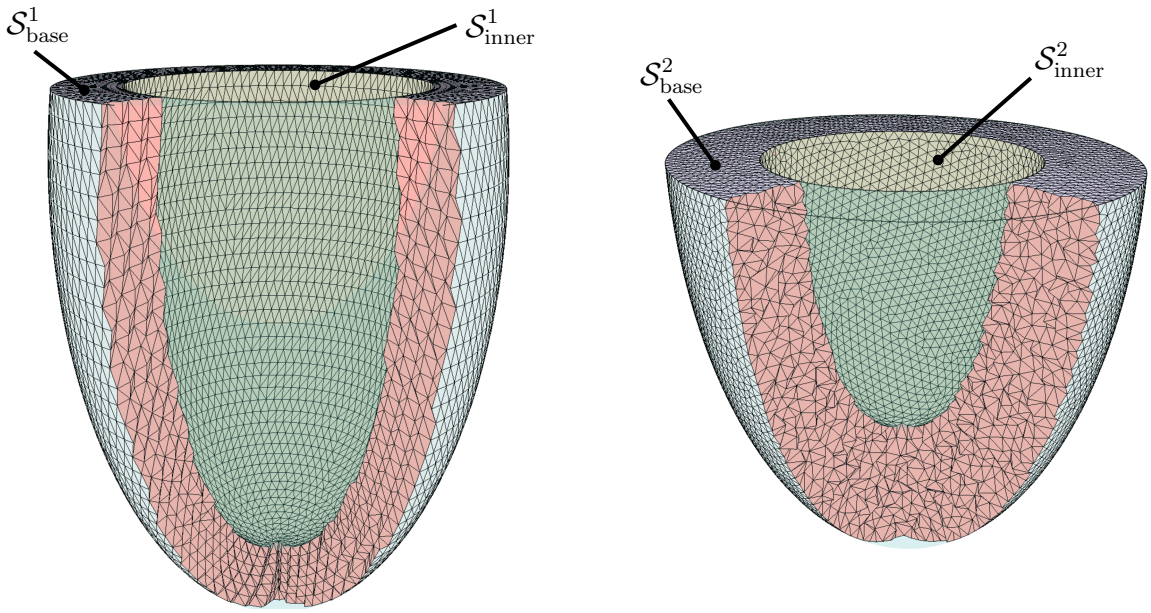
Furthermore, the BCs for both cases are

$$\mathbf{u}(\mathbf{X}, t) = \mathbf{0} \quad \text{for } \mathbf{X} \in \mathcal{S}_{\text{base}}^i, \tag{2.79a}$$

and

$$\bar{\mathbf{p}}(\mathbf{X}, t) = -\bar{p}(t) J \mathbf{F}^{-T} \mathbf{N} \quad \text{for } \mathbf{X} \in \mathcal{S}_{\text{inner}}, \tag{2.79b}$$

where we will define $\bar{p}(t)$ for each case later.



(a) Axisymmetric (b) Eccentric

Figure 2.8: Tetrahedral meshes used for the thick-walled ellipsoids. The axisymmetric mesh contains 21,504 nodes and 108,738 tetrahedral cells, while the eccentric mesh contains 23,317 nodes and 119,360 tetrahedral cells.

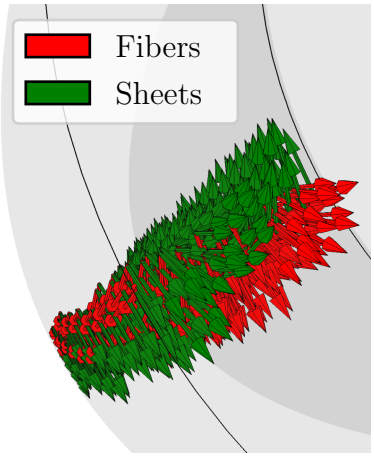


Figure 2.9: Fibers and sheets

The meshes used for the axisymmetric and eccentric ellipsoids are both shown in Figure 2.8. The material is modeled using the Guccione model [19] for the axisymmetric case, and as a neo-Hookean material for the eccentric case. Each case is also subject to different traction BCs as well.

2.4.3.1 Axisymmetric Thick-walled Ellipsoid

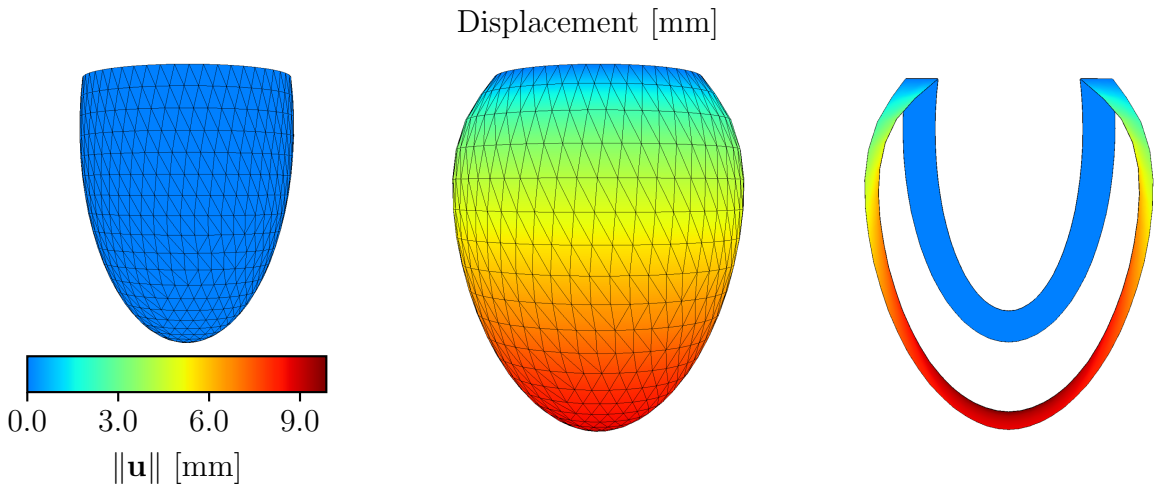
Given that the model provided by Guccione et al. is a transversely isotropic constitutive equation, we need fiber and sheet directions. Both of these are shown in Figure 2.9. The loading function for this case was set to

$$\bar{p}(t) = 10t. \tag{2.80}$$

Additionally, the material parameters were chosen to be $C = 10$ kPa, $b_f = b_t = b_{fs} = 1$, and the deformation was simulated for $t \in [0, 1]$, as can be seen in the Python script below:

Python code

```
import fenicsmechanics as fm

# Material model and parameters
mat_dict = {
    "const_eqn": "guccione",
    "type": "elastic",
    "incompressible": True,
    "density": 0.0,
    "bt": 1.0, "bf": 1.0, "bfs": 1.0, "C": 10.0,
    "fibers": {
        "fiber_files": ["n1-p0.xml.gz",
                        "n2-p0.xml.gz"],
        "fiber_names": ["n1", "n2"],
        "element-wise": True}
}

# Mesh file names
mesh_dict = {
    "mesh_file": "mesh.xml.gz",
    "boundaries": "boundaries.xml.gz"
}

# Time integration parameters, BCs, and polynomial degree.
formulation_dict = {
    "time": {
        "dt": 0.01,
        "interval": [0., 1.]
    },
    "element": "p2-p1",
    "domain": "lagrangian",
    "bcs":{
        "dirichlet": {
"displacement": [[0., 0., 0.]],
            "regions": [10], # Integer ID for base plane
        },
        "neumann": {
            "regions": [20], # Integer ID for inner surface
            "types": ["pressure"],
            "values": ["10.0*t"]
        }
    }
}

# Combine above dictionaries into one.
config = {"material": mat_dict, "mesh": mesh_dict,
    "formulation": formulation_dict}

# Create problem and solver objects.
fname_disp="results/displacement_output.pvd"
```



(a) Reference configuration (b) Current configuration (c) Sliced view

Figure 2.10: Deformation of thick-walled axisymmetric ellipsoid.

```
problem = fm.SolidMechanicsProblem(config)
solver = fm.SolidMechanicsSolver(problem, fname_disp=fname_disp)

# Numerically solve the problem.
solver.full_solve()
```

The initial configuration, and the configuration at $t = 1$ s are both shown in Figure 2.10. Additionally, a convergence study was performed and the locations of the apical endocardium and epicardium were compared. These displacements for the different mesh refinements are shown in Table 2.3, and can be directly compared to the results in Figure 6 in Land et al. [31].

2.4.3.2 Eccentric Thick-walled Ellipsoid

Similar to the axisymmetric case, the pressure BC applied to the inner wall was linear for $t \in [0, 1)$ s. However, this case is simulated for $t \in [0, 3]$ s, where the pressure applied from $t = 1$ to $t = 3$ is given by a piecewise function with a period of $T = 1$ s, and depends both on time, t, and on the relative position along its axis. Let us define

$$\hat{Z} = \frac{Z - Z_{\min}}{Z_{\max} - Z_{\min}}, \tag{2.81a}$$

and

$$\tau = t - \lfloor t \rfloor. \tag{2.81b}$$

Note that $\hat{Z}$ is the ratio of the distance between the apex of the inner surface and its base, whereas τ is the decimal portion of the time; both of these variables will always be between

			Final Apex Location (z, mm)	
Mesh Size (μm)	**DOF (u)**	**DOF (p)**	**Endocardium**	**Epicardium**
2000	6264	362	-26.85	-28.09
1000	63630	3084	-26.68	-28.34
500	475176	21504	-26.67	-28.33
300	2156805	94622	-26.67	-28.33

Table 2.3: Convergence study results for the ellipsoid problem. P2-P1 elements were used for all simulations. Endocardial and epicardial apex locations are in agreement with Figure 6 in Land et al. [31].

zero and one. Also, let

$$\hat{t}_1 = \hat{t}_1(\hat{Z}) = t_{10} + \hat{Z}\,(t_{11} - t_{10})\,, \tag{2.81c}$$

and

$$\hat{t}_2 = \hat{t}_2(\hat{Z}) = t_{20} + \hat{Z}\,(t_{21} - t_{20})\,. \tag{2.81d}$$

Now, let us define the piecewise function

$$g(\tau, \hat{Z}) = \begin{cases} \frac{1}{2}\left(1 - \cos\left(\frac{\pi\tau}{\hat{t}_1}\right)\right) & \tau \in [0, \hat{t}_1) \\ 1 & \tau \in [\hat{t}_1, \hat{t}_2)\,. \\ \frac{1}{2}\left(1 + \cos\left(\frac{\pi(\tau - \hat{t}_2)}{1 - \hat{t}_2}\right)\right) & \tau \in [\hat{t}_2, 1] \end{cases} \tag{2.82}$$

Note that, depending on the specific values of t_{10}, t_{11}, t_{20}, and t_{21}, $g(\tau, \hat{Z})$ will resemble a smoothed step function, where the cosine functions are used to smooth the increase from zero to one.

Now with all of the preliminary variables and functions in place, let us define the pressure applied on the inner surface, $\mathcal{S}^2_{\text{inner}}$, as

$$\bar{p}(\hat{Z}, t) = \begin{cases} \bar{p}_0 t & t \in [0, 1) \\ \bar{p}_0\left(1 - g(\hat{Z}, t)\right) & t \in [1, 3] \end{cases}. \tag{2.83}$$

All of this combined results in a step-like increase of the pressure applied to the inner wall, while ensuring that the reduction of pressure occurs faster at the base than at the apex. This is shown in Figure 2.11. Note that the parameters were set to $z_{\min} = -15$ mm, $z_{\max} = -1$ mm, $t_{10} = 0.4$, $t_{20} = 0.4$, $t_{11} = 0.3$, $t_{21} = 0.5$, and $p_0 = 100$ Pa.

The Python script to formulate and solve the eccentric case is shown below:

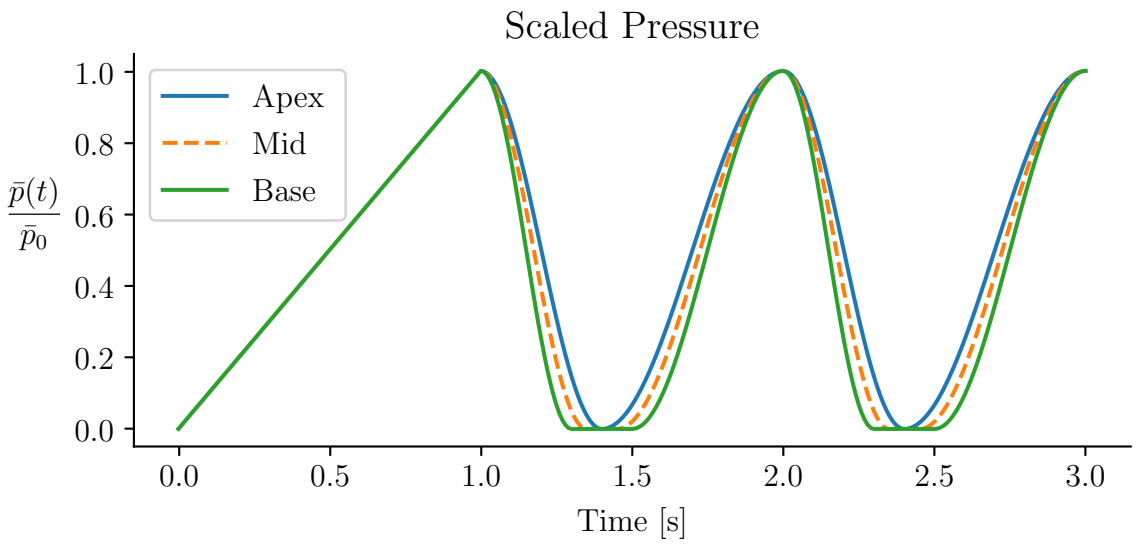


Figure 2.11: Pressure wave form applied at $\mathcal{S}^2_{\text{inner}}$.

Python code

```
import fenicsmechanics as fm
from pressure import create_pressure_expression

BASE = 10
ENDO = 20

mat_dict = {
    "type": "elastic",
    "incompressible": True,
    "const_eqn": "neo_hookean",
    "density": 0.0, # g/mm^3
    "mu": 1333.0/9.242,  # g/mm*s^2
    "kappa": 1e6,        # g/mm*s^2
}

mesh_dict = {
  "mesh_file": "mesh/ellipsoid-eccentric.h5",
  "boundaries": "mesh/ellipsoid-eccentric.h5"
}

zmin = -15.0
zmax = -1.0
p0 = 50.0
p_inner = create_pressure_expression(p0, zmin, zmax)

formulation_dict = {
```

```
    'time': {
        'dt': 0.01,
        'interval': [0.0, 3.0]
    },
    'element': element,
    'domain': 'lagrangian',
    'bcs': {
        'dirichlet': {
            'displacement': [[0.0, 0.0, 0.0]],
            'regions': [BASE]
        },
        'neumann': {
            'regions': [ENDO],
            'types': ['pressure'],
            'values': [p_inner]
        }
    }
}
```

Note that there is additional Python and C++ code where the pressure wave form shown in Figure 2.11 that is not shown here. This code can be found in Appendix A. The resulting deformation is shown in Figure 2.12.

This chapter includes substantial portions (including but not limited to text, figures, and simulation results) from [15] in collaboration with Christoph M. Augustin, and Shawn C. Shadden.

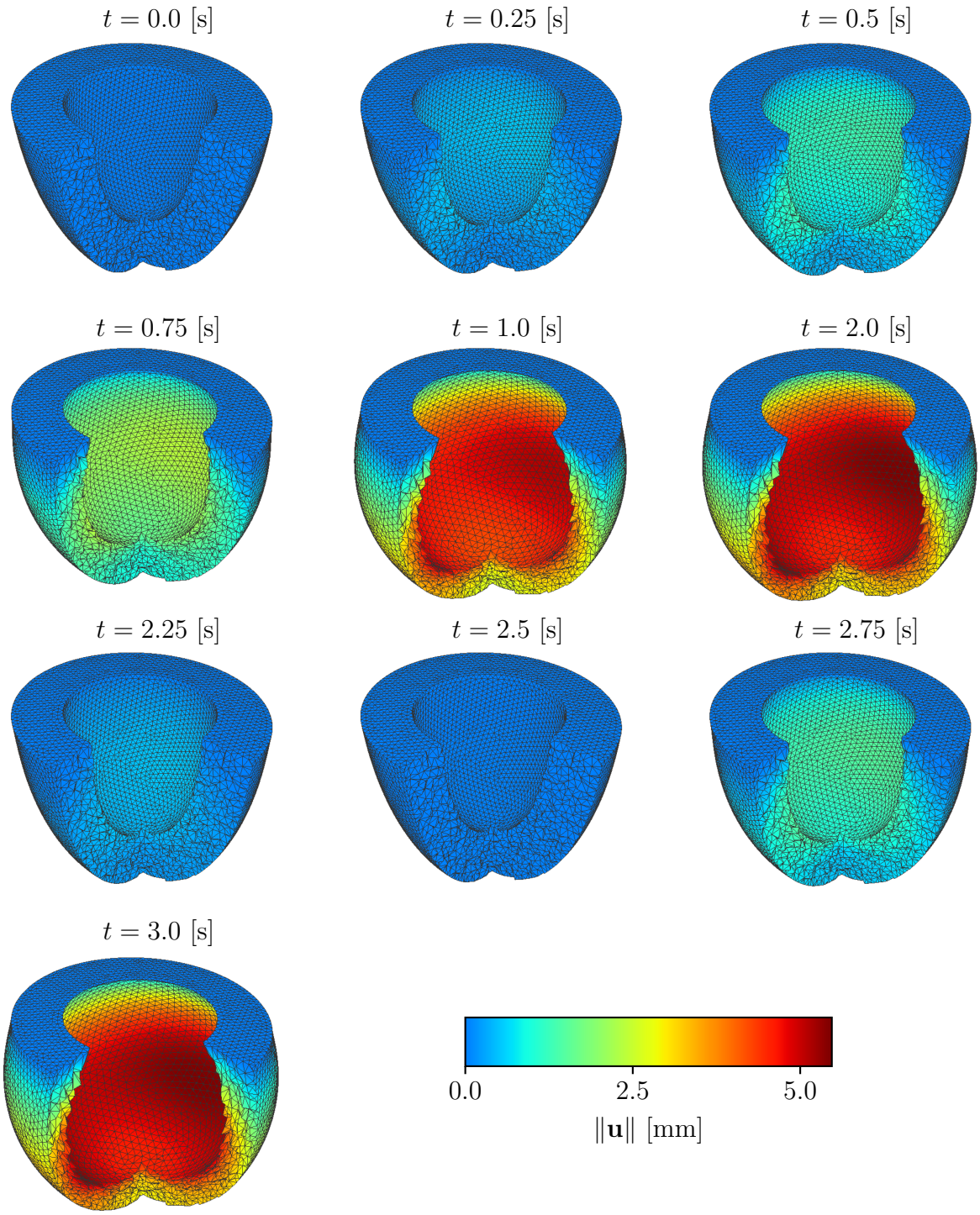


Figure 2.12: Deformation of the thick-walled eccentrically aligned ellipsoid for $t \in [0, 1] \cup [2, 3]$.

Chapter 3

In vivo Myocardial Strain

As mentioned in Section 1.1, the overarching goal of this work is to enhance the diagnosing of ESRD patients with heart disease using sequences of two-dimensional medical images obtained through cardiac magnetic resonance (CMR) imaging. A description of the framework developed in order to compute the planar deformation of the LV and its results when applied to numerous subjects will follow.

3.1 Motivation

Images used for this study were acquired by M.H. Vandsburger and colleagues as a part of earlier studies [9, 10]. The two-dimensional CMR images were acquired along a short-axis plane of the heart, as can be seen in Figure 3.1. Multiple parallel sets of images were obtained for each patient at different locations along the long axis. Due to the imaging protocol and the varying anaerobic endurance between patients, the number of frames taken is not guaranteed to be uniform at different short-axis planes for each patient. Furthermore, the number of locations used along the long axis also varies from patient to patient.

Additionally, the number of frames per cardiac cycle reduces as the heart rate of the subject increases. Because of this, there is no guarantee that the number of frames per cycle at different short-axis planes will remain constant for a single patient, let alone across multiple patients. This is both due to the effect of the varying fitness levels across patients and the need for expedient results in some cases. While spatial and temporal interpolation can be performed to reconstruct the time-resolved 3D geometry of the left ventricle, we are interested in using data that is generally accessible in clinical settings to determine the amount of strain undergone by the heart. Thus, we aim to study the deformation confined to each image plane which we will denote $\mathcal{P}_m$ for $m = 1, 2, \ldots, N_p$, where p denotes a unique patient identifier.

The human LV undergoes complex deformations which consists of contracting, twisting, and translating motions [14]. Thus, there is no guarantee whatsoever that the part of the myocardial LV tissue observed on one of the imaging planes, $\mathcal{P}_m$, at some time t_i will be

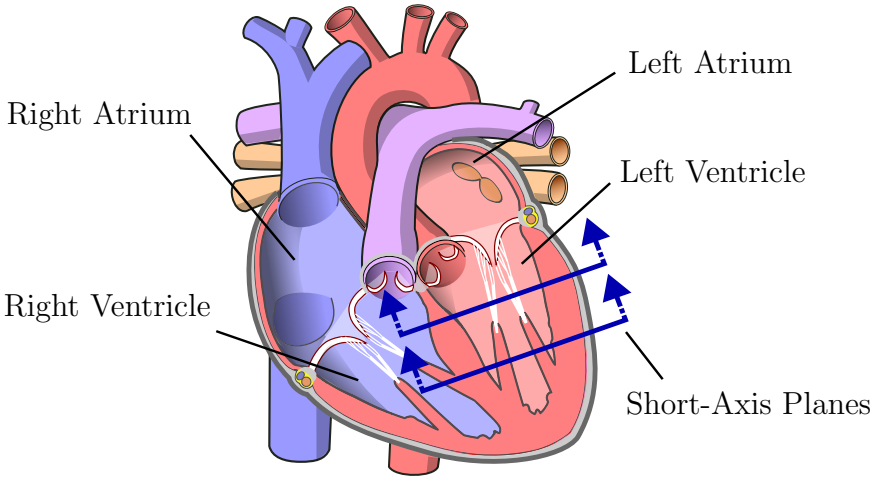


Figure 3.1: Cross section schematic of the four-chamber plane through the human heart showing the cutting plane line of two short-axis planes. This schematic was derived from [32].

the exact same part of the tissue at some other time t_j, where $i \neq j$. Nevertheless, we can identify the intersection of the inner and outer walls of the LV, named the endocardium and epicardium respectively, and $\mathcal{P}_m$ with a higher degree of certainty. As a result, we perform a segmentation of the LV for all of the available frames of the cardiac cycle. Our goal is then to compute some metric of deformation for the region enclosed by the endocardium and epicardium within said image plane.

While there has been extensive research in computing strain from medical images, many approaches do not guarantee a smooth and differentiable map, and some are only applicable to small deformations as mentioned by Beg et al. [33]. Hence, we turn to the large deformation diffeomorphic metric mapping (LDDMM) algorithm that they propose as a result. This algorithm requires that the solution to the shape matching problem simultaneously be a solution to a differential equation, which leads to a diffeomorphism that transforms a template shape into a target one. Having a diffeomorphism is specifically of interest here because the map must represent a realistic and physiological deformation. For an extensive discussion of the LDDMM algorithm, the reader is referred to *Shapes and Diffeomorphisms* by Laurent Younes [34].

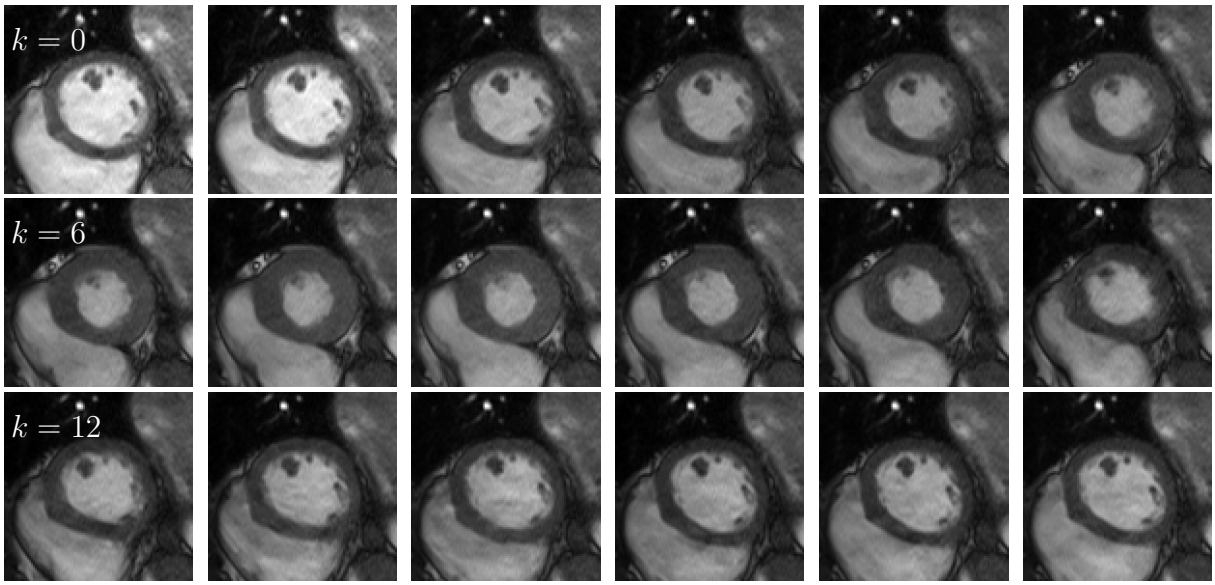


Figure 3.2: Full image sequence at one location along the long axis for one patient.

3.2 Planar Deformation Framework

As previously mentioned, the full pipeline of computing LV deformation begins with a sequence of 2D medical images, which we denote by

$$(I_k)_{k=0}^{N_{ps}-1}, \tag{3.1}$$

where N_{ps} is the total number of images in the sequence for a given patient p, at a slice or image plane s. An example of such an image sequence is shown in Figure 3.2. While it is possible to apply registration techniques to the images themselves, we opted for applying it to triangulated surfaces representing the left ventricular myocardium due to the previously mentioned fact that there is no guarantee that the set of material points in one frame of the image sequence will be present in the remaining frames. Thus, the only geometric entities of the LV that can be identified with confidence are the endocardium, epicardium, and the myocardium within the two; with the exception of intra- and inter-operator variations. Furthermore, reducing the input to the framework to only the segmentations of the endocardium and epicardium makes the framework invariant to the imaging modality used so long as segmentations can be obtained.

Each full sequence of images within a particular image plane, $\mathcal{P}_m$, was used to identify and segment the LV region. These segmentations were obtained with the segmentation capabilities of the free academic version of the Medviso Segment software [35, 36]. A full image sequence, along with the resulting curve segmentations of the endocardium and epicardium, denoted by $\mathcal{C}_{\text{endo}}(t_k)$ and $\mathcal{C}_{\text{epi}}(t_k)$ respectively for $k = 0, 1, \ldots, N_{ps} - 1$, are shown in Figure

3.3a. We have highlighted the frame $k = 8$ due to the fact that this segmentation has the largest Hausdorff distance between the endocardium and epicardium over the entire cardiac cycle imaged. The Hausdorff distance at time t_k is given by

$$b_{\max}(t_k) = \max\left\{ \sup_{x \in \mathcal{C}_{\text{endo}}(t_k)} d(x, \mathcal{C}_{\text{epi}}(t_k)), \sup_{y \in \mathcal{C}_{\text{epi}}(t_k)} d(\mathcal{C}_{\text{endo}}(t_k), y) \right\}, \tag{3.2}$$

where

$$d(x, \mathcal{C}) = \inf_{y \in \mathcal{C}} d(x, y) \tag{3.3}$$

is the shortest distance between a point x and a curve $\mathcal{C}$. Note that the distance between two points, $d(x, y)$, is taken to be the Euclidean norm in our case. The maximum Hausdorff distance for a sequence on a given image plane is given by

$$\hat{b}_{\max} = \max\left\{b_{\max}(t_k)\right\}_{k=0}^{N_{ps}-1}. \tag{3.4}$$

Once all of the frames are segmented, a Delaunay triangulation of the interior is generated to represent the myocardial region via the MeshPy package [37], which provides a Python interface to the open source libraries Triangle [38, 39], Tetgen [40], and Gmsh [41]. Given that we are using 2D triangular meshes, only the interface to Triangle was used. In order to specify a mesh element size we define the unitless parameter

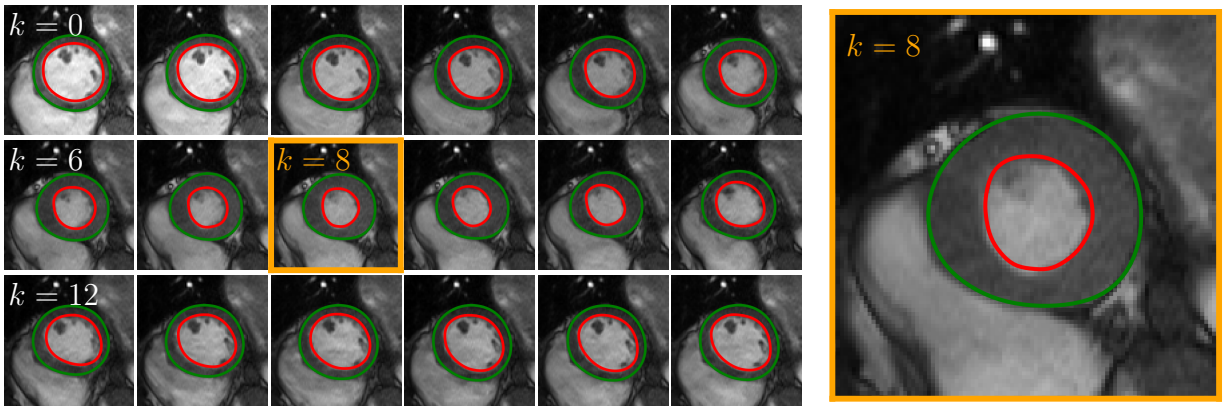
$$\sigma_{\max} = \frac{\hat{b}_{\max}}{h_{\max}}, \tag{3.5}$$

where

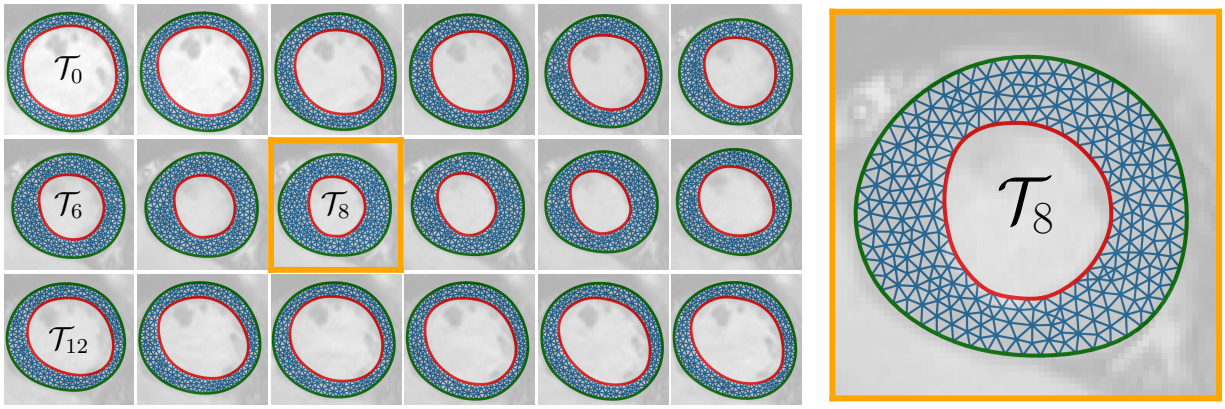
$$A_{\max} = \frac{1}{2} h_{\max}^2 \tag{3.6}$$

is the maximum area provided to MeshPy as the element size. The utility of the unitless parameter $\sigma_{\max}$, which will be referred to as the mesh resolution, will be clear in Chapter 4. Once a mesh element size is specified, a sequence of meshes is generated. Figure 3.3b shows this sequence for the images shown in Figure 3.2 with $\sigma_{\max} = 5$ for illustration purposes only. Note that the segmentation and mesh for the frame with the maximum Hausdorff distance, t_8, are again highlighted in Figure 3.3b.

These triangular meshes are now target surfaces for the registration algorithm, which we refer to as $\mathcal{T}_k$, for $k = 0, 1, \ldots, N_{ps} - 1$. Recall that a mesh $\mathcal{T}$ is uniquely identified by a vertex set and a cell connectivity set, or edge set if preferred. All meshes in the sequence are generated independently, and thus there is no guarantee that either the vertex or cell connectvity sets will be the same and/or have a one-to-one correspondence with each other for any given target pair. Thus, there is no straightforward way to compute displacements at this stage. This is where the LDDMM algorithm comes in.



(a) Full image sequence on the left with segmentations. The thickest segmentation for this case is shown again on the right.



(b) The target meshes, $\mathcal{T}_k$, is shown overlayed on the images and segmentations.

Figure 3.3: Schematic of the diffeomorphic registration framework used to compute planar deformation.

We must designate one of the target meshes as the template mesh, which will be transformed to fit the remaining targets. In order to prevent the mesh quality from deteriorating too much, we choose the mesh corresponding to the segmentation with the largest Hausdorff distance between the endo- and epicardium. This, in combination with the mesh element size defined through Equations (3.5) and (3.6), also increases the number of elements across the thickness on which the resulting deformation will be distributed. The larger number of triangular elements for t_8 can be seen in Figure 3.3b. We will refer to the template mesh as $\mathcal{T}_{k_0}$ where $k_0 \in [0, N_{ps} - 1]$ is now a fixed index dependent on p and s.

With the target meshes and a designated template mesh, we are now prepared to apply the LDDMM algorithm to a sequence of template-target surface pairs through the Python code made available by Laurent Younes [42]. The formulation used in this code base makes extensive use of kernel functions to represent the solution to the LDDMM optimization

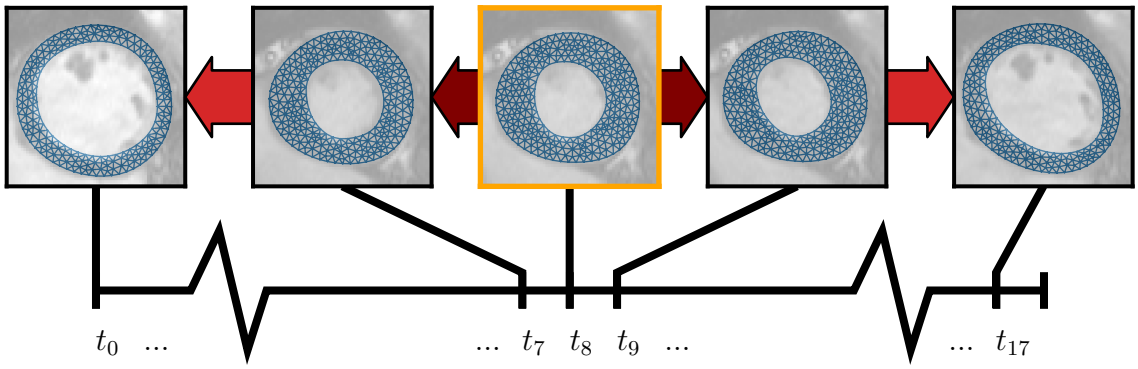


Figure 3.4: The template mesh surface and the target meshes to which it is mapped to. Note that the maroon arrows indicate registration between two consecutive images, while red indicates multiple registrations with intermediate images that are not shown.

problem. The user is allowed to specify the radii of kernel functions used. For more detailed information, the reader is again referred to [34]. In light of our definition of the mesh resolution, $\sigma_{\max}$, we similarly define a kernel radius ratio in terms of the largest thickness, namely

$$\gamma_{\max} = \frac{\hat{b}_{\max}}{\sigma_k}, \tag{3.7}$$

where σ_k is the kernel radius itself.

Since the template is determined by the segmentation with the largest thickness ($k = k_0$), we will need to apply the algorithm to pairs of surfaces both backward and forward along the ordered sequence. I.e., the algorithm will be applied to the surface pairs $(\mathcal{M}_k, \mathcal{T}_{k+1})$ for $k = k_0, k_0 + 1, \ldots, N_{ps} - 2$, and to the surface pairs $(\mathcal{M}_k, \mathcal{T}_{k-1})$ for $k = k_0, k_0 - 1, \ldots, 1$, where $\mathcal{M}_k$ denotes the template deformed to match the shape of the target $\mathcal{T}_k$. I.e.,

$$\mathcal{M}_k = \hat{\boldsymbol{\varphi}}_{k_0,k}(\mathcal{T}_{k_0}) = \begin{cases} \hat{\boldsymbol{\varphi}}_{k-1,k}(\mathcal{M}_{k-1}) = \hat{\boldsymbol{\varphi}}_{k-1,k} \circ \cdots \circ \hat{\boldsymbol{\varphi}}_{k_0,k_0+1}(\mathcal{T}_{k_0}) & k \geq k_0 \\ \hat{\boldsymbol{\varphi}}_{k+1,k}(\mathcal{M}_{k+1}) = \hat{\boldsymbol{\varphi}}_{k+1,k} \circ \cdots \circ \hat{\boldsymbol{\varphi}}_{k_0,k_0-1}(\mathcal{T}_{k_0}) & k < k_0 \end{cases}, \tag{3.8}$$

where we define $\mathcal{M}_{k_0} = \hat{\boldsymbol{\varphi}}_{k_0,k_0}(\mathcal{T}_{k_0}) = \mathcal{T}_{k_0}$ since $\hat{\boldsymbol{\varphi}}_{k_0,k_0}$ is the identity map. Note that $\hat{\boldsymbol{\varphi}}_{j,k}$ will be a composition of solutions to the LDDMM algorithm for $|j-k|> 1$. Figure 3.4 shows a flow chart of the forward and backward directions of registration from the template mesh. Three sequences result from this process:

1. $\left(\hat{\boldsymbol{\varphi}}_{k,k+1}\right)_{k=k_0}^{N_{ps}-2}$: the sequence of forward mappings,
2. $\left(\hat{\boldsymbol{\varphi}}_{k,k-1}\right)_{k=1}^{k_0}$: the sequence of backward mappings, and
3. $(\mathcal{M}_k)_{k=0}^{N_{ps}-1}$: the sequence of the deformed template mesh shown in Figure 3.5.

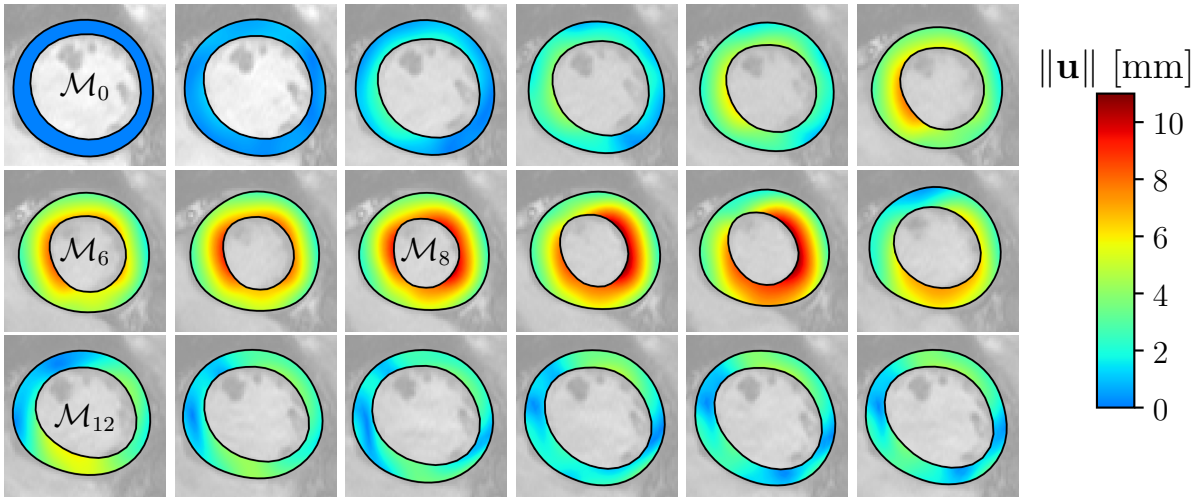


Figure 3.5: The deformed template mesh and the resulting displacement field with respect to $\mathcal{M}_0$.

With these three sequences at our disposal, we are well posed to compute displacements at the vertices of the template mesh, $\mathcal{T}_{k_0}$, since now all $\mathcal{M}_k$ have the same mesh topology. Furthermore, we can designate any of the deformed meshes to be our reference. Thus, we will take $\mathcal{M}_0$ to be our reference configuration to compute the displacement field, as well as any subsequent strain measures. Specifically, we will use a linear finite element representation of the displacement, $\hat{\mathbf{u}}(t_k)$, as discussed in Section 2.2 to represent the displacement field. The resulting displacement fields for the example case are shown Figure 3.5. The computation of the strain in order to quantify the myocardial tissue deformation will be discussed in the following section.

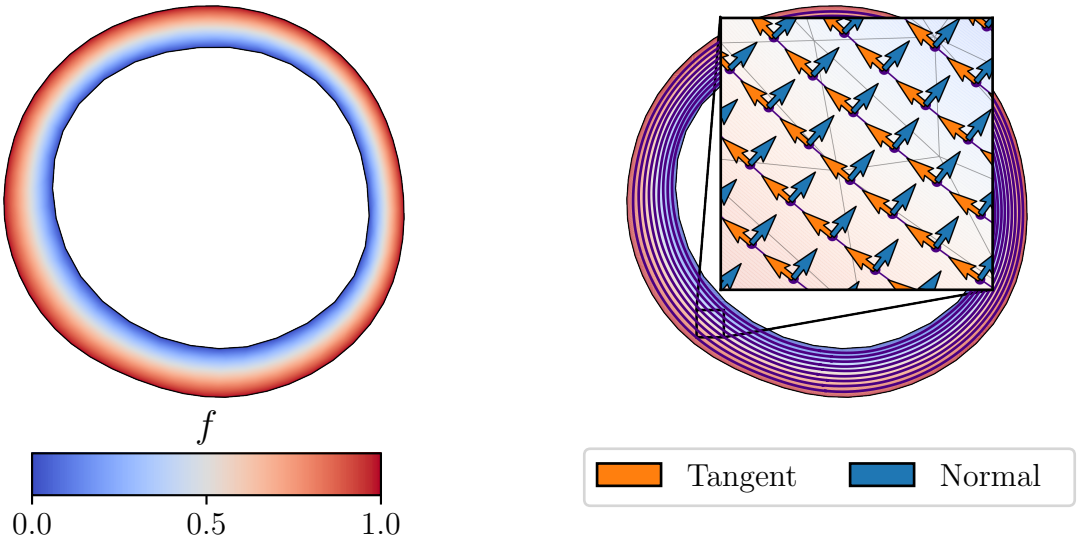
3.3 Strain Computations in non-Circular Domains

As mentioned in Section 2.1.1.2, the square of the strain along a curve with tangent vector $\mathbf{M}$ is given by

$$C_{MM} = \mathbf{M} \cdot \mathbf{CM},$$

where $\mathbf{C} = \mathbf{F}^T\mathbf{F}$ is the right Cauchy-Green strain tensor. But first, we must specify the curves that are of interest.

Given that the short-axis view of the LV is a near-circular region, strain measures of the LV are often given in terms of *radial* and *circumferential* components, such as in [43, 44, 45]. While this is easy to do in an exactly circular domain by using the cylindrical orthonormal basis $(\mathbf{e}_r, \mathbf{e}_\theta, \mathbf{e}_z)$, it is less clear what is meant by the *radial* and *circumferential* directions



(a) (b)

Figure 3.6: Solution to the BVP given in Equation 3.9, and the resulting unit tangent and normal vectors. Note that the vectors have been scaled for improved visualization.

in an arbitrary LV geometry. For our purposes, we will define the *circumferential* directions to be along the level sets of the solution to Laplace's equation with constant Dirichlet BCs applied at the endo- and epicardium. I.e., the level sets of f such that

$$\Delta f = 0, \quad \text{for } \mathbf{X} \in \mathcal{M}_0, \tag{3.9a}$$

$$f = 0, \quad \text{for } \mathbf{X} \in \mathcal{C}_{\text{endo}}(t_0), \tag{3.9b}$$

$$f = 1, \quad \text{for } \mathbf{X} \in \mathcal{C}_{\text{epi}}(t_0), \tag{3.9c}$$

where Δ is the Laplacian differential operator here. This BVP is solved on the reference mesh $\mathcal{M}_0$, and not the template mesh $\mathcal{M}_{k_0} = \mathcal{T}_{k_0}$, using first-order continuous Galerkin finite elements. While the solution is not linear, it still provides a systematic way of identifying curves of interest.

The solution to the BVP given in Equation (3.9) for the deformed mesh $\mathcal{M}_0$ shown in Figure 3.5 is shown in Figure 3.6a, and the resulting level sets,

$$\mathcal{C}_j = \left\{ \mathbf{X} \in \mathcal{M}_0 \;\middle|\; f(\mathbf{X}) = 0.1j \right\} \tag{3.10}$$

for $j = 1, 2, \ldots, 9$, extracted via the contour filter provided by VTK [46], are shown in Figure 3.6b. We will refer to the $\mathcal{C}_j$ as the interior curves of the domain. Once the curves have been extracted, we can compute the Frenet-Serret apparatus while keeping in mind numerical

errors caused by interpolation or floating point arithmetic. Given a parameterization of a curve, say $\mathbf{x}(s)$, the equations for the Frenet-Serret apparatus are

$$\mathbf{t}(s) = \frac{\mathbf{x}'(s)}{\|\mathbf{x}'(s)\|}, \quad \mathbf{b}(s) = \frac{\mathbf{y}(s)}{\|\mathbf{y}(s)\|}, \quad \mathbf{n}(s) = \mathbf{b}(s) \times \mathbf{t}(s), \tag{3.11a}$$

$$\kappa(s) = \frac{\|\mathbf{y}(s)\|}{\|\mathbf{x}'(s)\|^3}, \quad \tau(s) = \mathbf{x}'''(s) \cdot \mathbf{y}(s), \tag{3.11b}$$

where $\mathbf{y}(s) = \mathbf{x}'(s) \times \mathbf{x}''(s)$, and

$$\mathbf{x}'(s) = \frac{d\mathbf{x}}{ds}, \quad \mathbf{x}''(s) = \frac{d^2\mathbf{x}}{ds^2}, \quad \mathbf{x}'''(s) = \frac{d^3\mathbf{x}}{ds^3}. \tag{3.12}$$

Note that the parameterization does not need to be with respect to arc length for these equations.

Given that we are treating all deformations as planar within $\mathcal{P}_m$, the binormal vector and torsion are given by $\mathbf{b}(s) = \mathbf{e}_z$, and $\tau(s) = 0$ for every interior curve across all cases, respectively. Figure 3.6b shows the unit vectors that are tangent (orange) and normal (blue) to the interior curves (indigo). With FE representations of the Frenet-Serret apparatus, we have everything we need to compute the strain components along the interior curves.

First, we compute the left Cauchy-Green strain tensor, $\mathbf{C}$, on the full triangular mesh $\mathcal{M}_0$ for each $\hat{\mathbf{u}}(t_k)$, $k = 0, 1, \ldots, N_{ps} - 1$. Then, we evaluate the tensor on the curve meshes and compute its three components (the tensor is symmetric and 2D)

$$C_{tt} = \mathbf{t} \cdot \mathbf{C}\mathbf{t}, \quad C_{nn} = \mathbf{n} \cdot \mathbf{C}\mathbf{n}, \quad C_{nt} = \mathbf{n} \cdot \mathbf{C}\mathbf{t}, \tag{3.13}$$

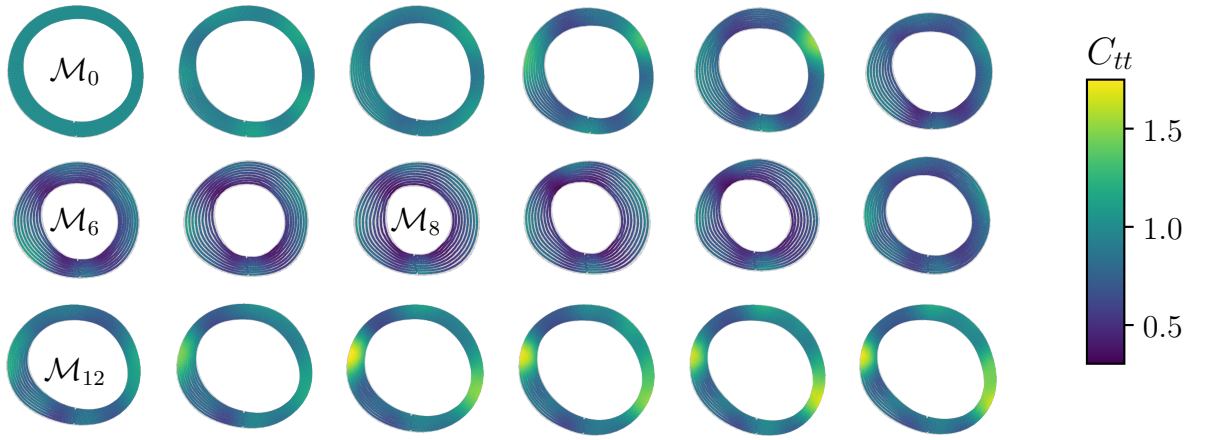
which we will refer to as the circumferential, radial, and shear components along the curves, respectively. Note that

$$C_{bt} = C_{tb} = C_{bn} = C_{nb} = 0, \text{ and } C_{bb} = 1 \tag{3.14}$$

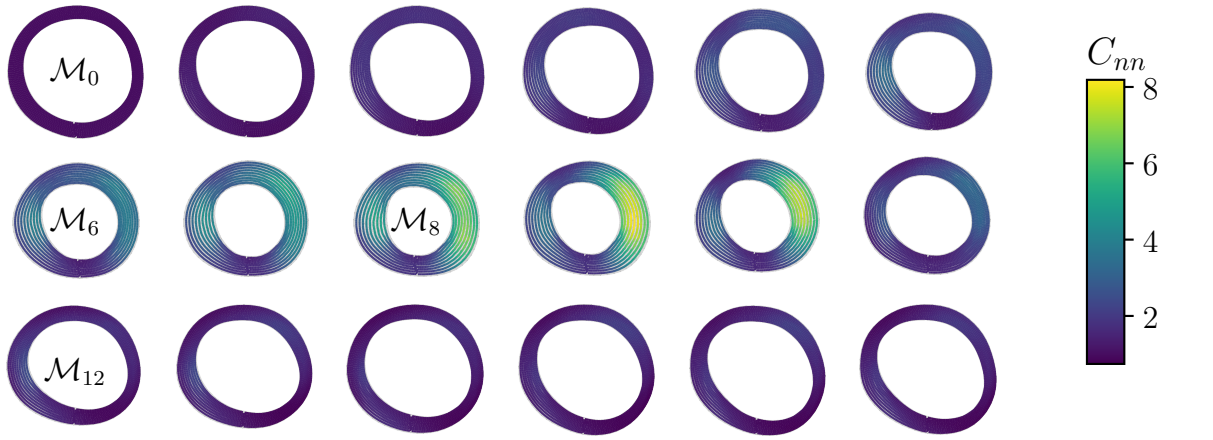
for all cases due to our assumption of planar motion.

The curve strain components for the example case are shown in Figures 3.7a-3.7c. We can see that as the LV constricts, the circumferential component of strain, C_{tt}, is less than 1. This indicates that the material curve is decreasing in length (at least locally), which makes physical sense since the overall circumference of each curve decreases during this phase of the cardiac cycle. However, the radial component, C_{nn}, increases by a factor of 8. This is due to the fact that the distance between the endocardium and epicardium increases as the LV constricts, thus elongating (or thickening) the tissue in the radial direction. Lastly, the shear component of strain gives a measure of the amount of twist produced by the LDDMM solution.

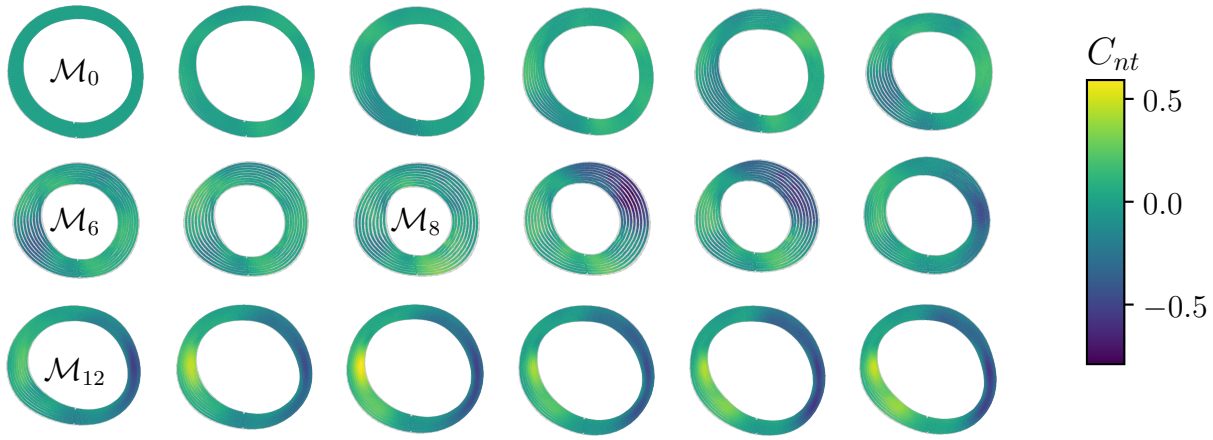
The convergence and accuracy of the displacement, $\mathbf{u}_{\text{reg}}$, and the strain $\mathbf{C}$ resulting from the framework is studied in Chapter 4. Specifically, the variation in the error of the resulting displacement and strain fields are examined as a function of $\sigma_{\max}$ and $\gamma_{\max}$. It was concluded that $\sigma_{\max} = 10$ and $\gamma_{\max} = 2$ will be used for studying patient data.



(a) The circumferential component of the right Cauchy-Green strain, $C_{tt} = \mathbf{t} \cdot \mathbf{Ct}$.



(b) The radial component of the right Cauchy-Green strain, $C_{nn} = \mathbf{n} \cdot \mathbf{Cn}$.



(c) The off-diagonal component of the right Cauchy-Green strain, $C_{nt} = \mathbf{n} \cdot \mathbf{Ct}$.

Figure 3.7: The three components of the 2D right Cauchy-Green tensor.

Sex	No. Patients	Range	Mean	Median
All	35	32-79	54	54
Female	18	37-79	53.9	51.5
Male	17	32-78	54.1	56

Table 3.1: Age demographics of patients used for statistical testing.

3.4 Population Study

The CMR image data sets that we used for this study are a subset of those used in [10]. The age demographics of the subpopulation used for this study are shown in Table 3.1. Given the reduced accuracy of the framework when applied closer to the apex of the ellipsoid, discussed in Sections 4.3.1.3 and 4.3.2.3, image sequences closer to the middle of the heart were chosen for all patients. Image planes too close to the base and showing too much of the aortic arch were neglected due to the inability to confidently identify the myocardium throughout the entire cardiac cycle. This led to a reduction in useable image sequences for some patients. For consistency, the image sequence closest to the middle was used. The image slices are numbered from 1-8, for example, with 1 being the closest to the base and 8 being the closest to the apex. Thus, the slice chosen for each patient was between 3-5, depending on the specific patient and the usability of the image sequences.

The registration framework was applied to each image sequence; beginning with the segmentation and ending with the computation of curve strains. The end goal here is to identify if there are statistically significant differences in the strain components obtained through this framework between the control patients and those with CKD.

3.4.1 Statistical analysis

Here, a rudimentary statistical analysis is performed. First, a t-test is applied to check if there is a significant difference between the subjects known to be control patients and those with CKD. Second, a k-means cluster analysis is performed using various quantities to test the viability of this framework as a diagnostic tool.

3.4.1.1 Student t-test of Control vs. CKD

Given that it is known which are the control patients and which have CKD, a t-test was performed to determine which parameters, if any, are significantly different between the two groups of patients. There are $n_1 = 20$ control patients, and $n_2 = 15$ patients with CKD. Further, let $\bar{X}_1, \bar{X}_2$ denote the sample means, and s_1, s_2 the unbiased sample variances. SciPy

provides a function to perform a t-test with equal or unequal sample sizes, as well as *equal* or *unequal* sample variances [47].

The t-statistic for unequal sample sizes with $\frac{1}{2} < s_1/s_2 < 2$ is given by

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}, \tag{3.15a}$$

where

$$s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}} \tag{3.15b}$$

is the pooled standard deviation. On the other hand, the t-statistic is given by

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s_{\bar{\Delta}}}, \tag{3.16a}$$

with

$$s_{\bar{\Delta}} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}} \tag{3.16b}$$

when $s_1 > 2s_2$ or $s_2 > 2s_1$. The choice between Equations (3.15) and (3.16) were made by comparing the sample unbiased variances at runtime.

The full framework has afforded an extensive list of values that can be subject to the t-test; the choice of spatial and temporal metrics used also extend this list. One anecdotal observation made during visual inspection of the registration results was the increased dispersion of the strain component values for a control patient vs. one with CKD. This dispersion of values at any given time can be quantified by computing the second moment of the strain with respect to the spatial variables or, similarly, the standard deviation of the strain values throughout the spatial domain at any given time step.

As will be discussed in Section 4.3, a metric was decided for both the spatial and temporal dimensions. Specifically, the mean and standard deviations throughout the spatial domain of the strain components were considered as a function of time. Then, three metrics over the temporal domain were used: (1) the mean value over time, (2) the maximum value over time, and (3) the value at the time point with the thickest segmentation (denoted by k_0 above) which is referred to as the n-max value. The results of the Student t-test for the values mentioned above are shown in Table 3.2. In addition to categorizing the spatial and temporal evaluations for each component of strain, it is reported whether the population unbiased standard deviation was found to be similar as discussed above. If this entry is True, Equation (3.15) was used, while Equation (3.16) was used when it is False. In addition to the t-statistic and p values shown in Table 3.2, box plots for the spatial means and spatial standard deviations of the strain components are shown in Figures 3.8 and 3.9, respectively.

Spatial Evaluation	**Component**	**Temporal Evaluation**	**Similar Variance**	t-**statistic**	p-**val**
Mean	**n · Cn**	mean	True	0.3868	0.7014
		max	False	-0.4991	0.6211
		n-max	False	-0.4894	0.6278
	n · Ct	mean	False	-1.8361	0.0798
		max	True	-0.0445	0.9648
		n-max	False	-0.0771	0.9393
	t · Ct	mean	True	-1.1958	0.2403
		max	True	0.5102	0.6133
		n-max	False	-1.1003	0.2825
Standard Deviation	**n · Cn**	mean	False	0.2801	0.7819
		max	True	0.0519	0.9589
		n-max	True	0.4615	0.6475
	n · Ct	mean	True	0.5237	0.6040
		max	False	-0.3068	0.7609
		n-max	False	-1.1258	0.2687
	t · Ct	mean	False	2.9882	0.0079
		max	False	3.0113	0.0069
		n-max	True	3.1301	0.0036

Table 3.2: Student t-test performed on the mean and standard deviation over spatial variable of the three strain components.

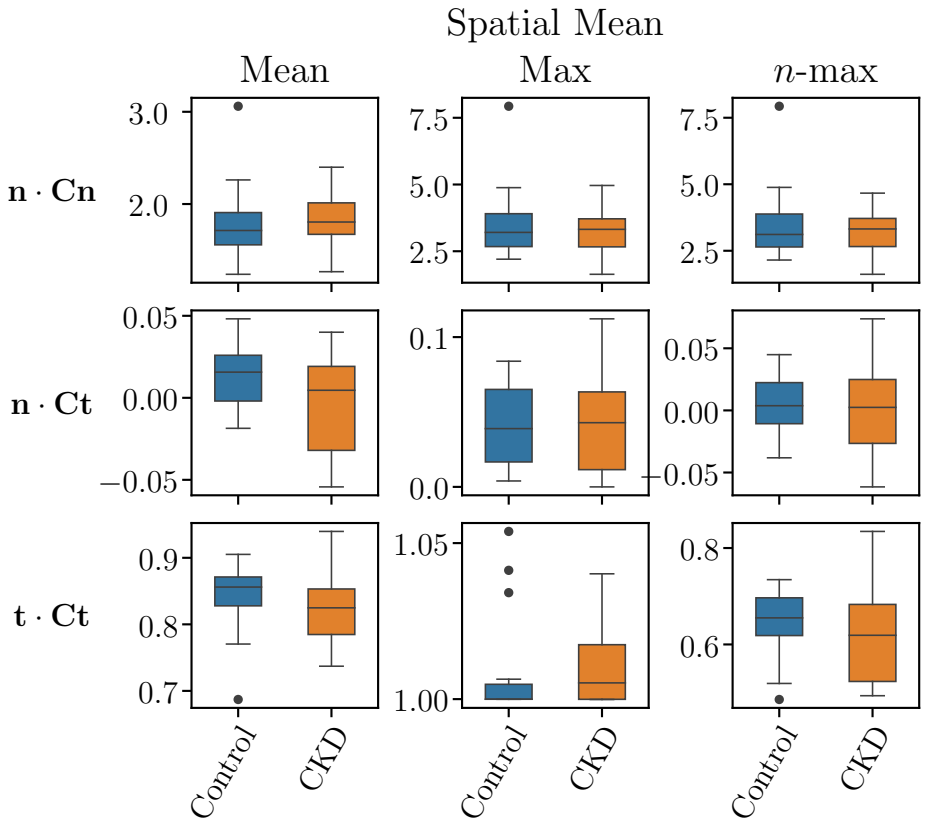


Figure 3.8: Box plots for the spatial mean of the strain components for control and CKD patients.

The results summarized in Table 3.2 clearly indicate that the null hypothesis—differences in two groups are due to chance alone—cannot be rejected for any of the strain components when the spatial mean is used. Furthermore, the null hypothesis can only be rejected for the circumferential component, $\mathbf{t} \cdot \mathbf{Ct}$, using the spatial standard deviation. This can also be seen in the box plots. The range of strain component values for the control vs. CKD patients exhibit significant overlap for all except the spatial standard deviation of the circumferential component; specifically evaluated at n-max. Thus, it can be said that with this framework, the circumferential component of strain is the most reliable when differentiating between control and CKD patients based on these subjects.

3.4.1.2 Cluster Analysis

In addition to the Student t-test, a k-means algorithm was applied to the resulting strain evaluations discussed above. The k-means algorithm is a clustering algorithm designed to group an arbitrary set of data points into k subgroups, or clusters, by iteratively computing the means of the subgroups and updating the clusters until convergence is reached [48].

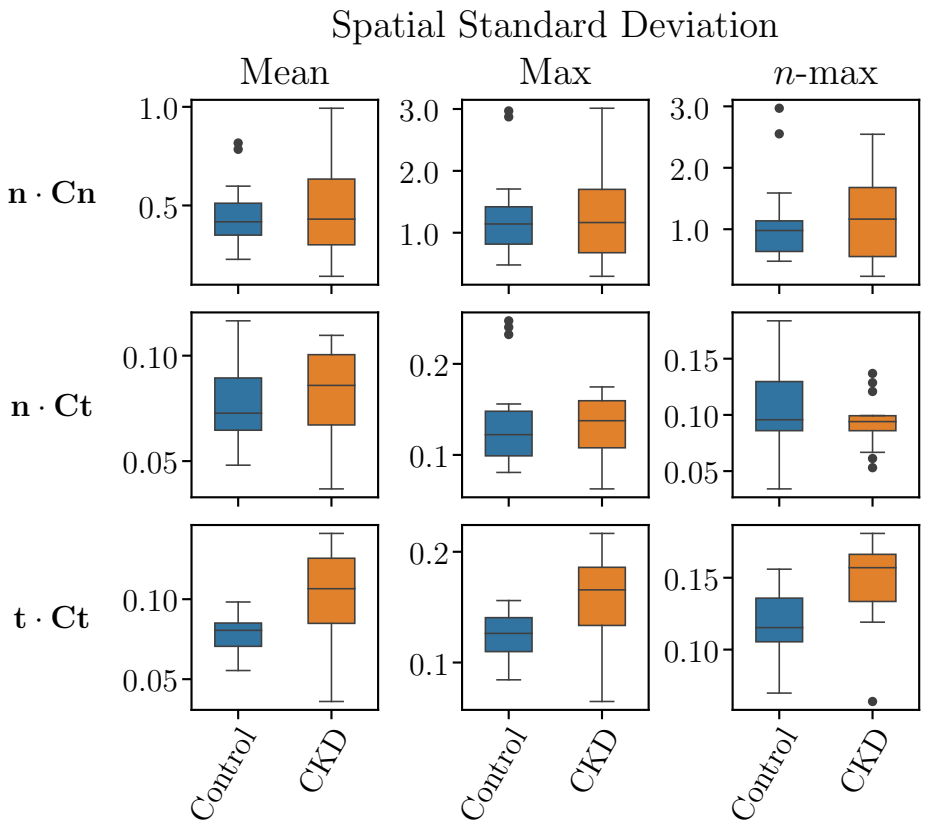


Figure 3.9: Box plots for the spatial standard deviation of the strain components for control and CKD patients.

Convergence is reached when the members of the subgroups no longer change. Thus, an initial guess of the mean values for each cluster is required in order to initially assign data points to clusters.

Recall that the circumferential strain component was shown to have the largest statistically significant difference between control patients and those with CKD when using the n-max of the spatial standard deviation. Nevertheless, the k-means algorithm was applied to 1-3 components at a time in order to provide a complete comparison.

The patient classifications resulting from the cluster analysis when using different components of the strain are shown in Table 3.3. The largest number of correct classifications—true positive and true negative—for any of the combinations is 24. However, it can be argued that the best results in a clinical scenario are those that produce the least number of false negative classifications. This is due to the fact that when a test results in a negative classification when it is truly positive, there is more potential for the patient to be neglected when they are in need of treatment. On the other hand, a false positive will tag a patient as needing treatment, likely subjecting them to further studies to confirm or reject the initial

	Component Used			True		False	
Spatial Evaluation	**n · Cn**	**n · Ct**	**t · Ct**	**Positive**	**Negative**	**Positive**	**Negative**
Mean	Yes	No	No	0	18	2	15
	No	Yes	No	8	9	10	8
	No	No	Yes	2	12	13	8
	Yes	Yes	No	0	18	2	15
	Yes	No	Yes	0	18	2	15
	No	Yes	Yes	7	17	3	8
	Yes	Yes	Yes	0	18	2	15
Standard Deviation	Yes	No	No	7	17	3	8
	No	Yes	No	14	0	9	12
	No	No	Yes	12	12	8	3
	Yes	Yes	No	7	17	3	8
	Yes	No	Yes	7	17	3	8
	No	Yes	Yes	14	0	9	12
	Yes	Yes	Yes	7	17	3	8

Table 3.3: Classification of patients using the k-means algorithm on the n-max temporal evaluation of the strain components, with the most ideal scenario highlighted.

diagnosis. With this in mind, the cluster analysis performed on the n-max spatial standard deviation of the circumferential component of strain, $\mathbf{t} \cdot \mathbf{Ct}$, is the most ideal out of all of the combinations presented and is highlighted in Table 3.3.

The framework presented in this chapter allowed us to compute strains from planar images by applying a registration framework to triangular surfaces used to represent myocardial regions; all while being agnostic to the imaging modality. Moreover, the components of strain were extracted through a systematic procedure applicable to generic LV geometries. It was found that the *circumferential* component of strain resulted in the largest statistically significant difference between control patients and those with CKD. While the population size considered here is relatively low, it serves as a provisional study ready for further development.

Chapter 4

Validation of Planar Strain Computations

As mentioned in Section 3.2, the values for the mesh resolution and the kernel sigma ratio were chosen to be $\sigma_{\max} = 10$ and $\gamma_{\max} = 2$ for the patient data. In this chapter, the analysis that led to this choice of parameters is presented.

Note that it is impossible to know what the *true* deformation of the myocardium in any patient is. Thus, it is impossible for us to perform a conclusive analysis of the accuracy and precision of the registration framework developed in Chapter 3 using patient data. Therefore, we will use the simulation results presented in Section 2.4 as the ground truth, and run this data through our registration framework to validate its utility. Specifically, we will use the results from (1) the 3D box, (2) the thick-walled cylinder, and (3) the eccentric thick-walled ellipsoid. All of these three examples use the same material model, but differ in geometry and BCs. The three combined will give us insight into how the accuracy of the registration framework depends on the mesh resolution and the kernel sigma ratio, as well as the effect that out-of-plane motion has on the results.

Recall that the framework consists of the following steps:

1. Segment image data (Figure 3.3a)
2. Generate triangulations of the template surfaces (Figure 3.3b)
3. Apply LDDMM to the target-template pairs in the forward and backward directions (Figure 3.4)
4. Compute displacement field (Figure 3.5)
5. Compute curve strains (Figure 3.7)

Now, the framework was developed in order for it to effectively be invariant to the imaging modality used to identify the LV myocardium. I.e., the only step that really depends on the imaging modality used is step 1. Thus, the biggest change that we will have to make

for the synthetically generated data is the way in which segmentations are extracted from the geometry. Once this is done, the rest of the framework remains the same. The same procedure for computing curve strains will be followed to examine both the displacement and the resulting strain computed by the framework.

After discussing how the framework is adapted to these synthetic cases, we will study the convergence and accuracy of the results when compared to the solid mechanics simulation results. We will refer to this as the refinement study.

4.1 Planar Deformation Framework Applied to Synthetic Data

Given that the BCs for all of the cases mentioned above are periodic with respect to time (after the initial ramping of pressure in the eccentric ellipsoid case), we will only be using a subset of the time interval. Specifically, we will use $t \in [0, 1]$ for the box, $t \in [1/2, 3/2]$ for the cylinder, and $t \in [2, 3]$ for the ellipsoid.

Also, recall that the images obtained from patient data are those of a LV contracting rather than expanding. Thus, we want to make sure that the synthetic data provides us with a scenario as close to that observed in patients whenever possible. Therefore, the reference configuration was changed for the thick-walled cylinder and eccentric ellipsoid cases. The reference configuration was taken to be the configuration of the body at $t = 1/2$ [s] for the cylinder, and at $t = 2$ [s] for the ellipsoid. We will denote these new reference configurations by $\hat{\mathcal{D}}_0$ and $\hat{\mathcal{V}}_0^2$ for the cylinder and ellipsoid, respectively. Likewise, the current configurations with respect to these new references will be denoted by $\hat{\mathcal{D}}$ and $\hat{\mathcal{V}}^2$ for the cylinder and ellipsoid, respectively.

This approach was taken in order to achieve an expansion of the geometry while avoiding buckling phenomena that can occur when negative pressures are applied to inner walls of thick-walled bodies without the presence of residual stress. Also, note that the framework developed in Chapter 3 is only concerned with the kinematics of the deformation and is thus oblivious to the existence of non-zero stress at this new reference configuration.

At this point, we will discuss the different steps of the framework as they pertain to the synthetic cases.

4.1.1 Synthetic Segmentation

As previously mentioned, the starting point for the synthetic cases differs due to the fact that there are no images. However, we must still mimic the segmentation process as best as we can. For this, we will need a set of imaging planes, $\mathcal{P}_m$ for $m = 1, 2, \ldots, N_p$, where p is also a unique *patient* identifier, which we will refer to as a case identifier for synthetic cases to avoid confusion with actual patient data. Table 4.1 lists the domain variables, description of the domains, and 6 or 7 imaging planes defined in terms of an origin and a unit normal vector.

Domain	**Description**	**Plane ID**	**Origin** (X, Y, Z)	**Normal**
$\hat{\mathcal{R}}_0$	3D Box	1	(5,1,2)	$\mathbf{N} = \mathbf{E}_z$
		2		$\mathbf{N} = \mathbf{E}_y$
		3	$(2, 1, 2)$	$\mathbf{N} = \mathbf{E}_x$
		4	$(4, 1, 2)$	
		5	$(6, 1, 2)$	
		6	$(8, 1, 2)$	
		7	$(9.99, 1, 2)$	
$\hat{\mathcal{D}}_0$	Thick-walled cylinder	1	$(0,\ 0,\ 0)$	$\mathbf{N} = \mathbf{E}_z$
		2		$\mathbf{N} = -\sin 2^\circ\ \mathbf{E}_y + \cos 2^\circ\ \mathbf{E}_z$
		3		$\mathbf{N} = -\sin 4^\circ\ \mathbf{E}_y + \cos 4^\circ\ \mathbf{E}_z$
		4		$\mathbf{N} = -\sin 6^\circ\ \mathbf{E}_y + \cos 6^\circ\ \mathbf{E}_z$
		5		$\mathbf{N} = -\sin 8^\circ\ \mathbf{E}_y + \cos 8^\circ\ \mathbf{E}_z$
		6		$\mathbf{N} = -\sin 10^\circ\ \mathbf{E}_y + \cos 10^\circ\ \mathbf{E}_z$
$\hat{\mathcal{V}}^2$	Eccentric thick-walled ellipsoid	1	$(0, 0, -2.8)$	$\mathbf{N} = \mathbf{E}_z$
		2	$(0, 0, -4.6)$	
		3	$(0, 0, -6.4)$	
		4	$(0, 0, -8.2)$	
		5	$(0, 0, -10)$	
		6	$(0, 0, -11.8)$	
		7	$(0, 0, -13.6)$	

Table 4.1: The unique identifiers for synthetic cases, as well as their imaging planes given in terms of an origin and unit normal vector.

Note that the segmentations shown in Figure 3.3a are sequences of curves delineating the boundaries of the LV geometry within $\mathcal{P}_m$. Thus, we use VTK [46] to extract the boundary curves within each image plane. Specifically, we extract the surface of the 3D tetrahedral mesh, and then slice it using each of the planes given in Table 4.1 for each case. The tetrahedral meshes along with the image planes for each case are shown on the left panel of Figure 4.1, while the resulting segmentations are shown on the right panel. Note that the configuration shown both on the right and left is the configuration used as reference in the solid mechanics problem, instead of the new references mentioned above.

The segmentations were extracted at every $\Delta t = 0.1$ [s], which corresponds to every 10 time steps in the solid mechanics simulation. Furthermore, the LDDMM algorithm was set to use 10 time steps when solving for the map between the template and the target. This not only causes the number of segmentations per cycle to be on the same order of magnitude as those for patient data, but it also allows the direct comparison of the LDDMM results at every single stage with solid mechanics simulation results if necessary.

In addition, each set of image planes was chosen to analyze different contributions to the error in the results, as will now be described.

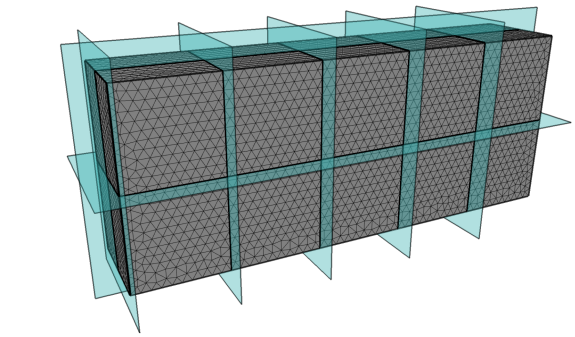
4.1.1.1 Three-dimensional Box

As can be seen from Table 4.1 and in Figure 4.1a, the image planes $\mathcal{P}_1$ and $\mathcal{P}_2$ are both planes of symmetry that are parallel to the XY and XZ planes, respectively. Note that the symmetry applies both to the geometry and the BCs. Therefore, the deformation evaluated on $\mathcal{P}_1$ and $\mathcal{P}_2$ should be fully contained within each plane, within numerical errors. Thus, these two planes will allow us to examine the accuracy of the registration framework when our assumption of planar motion coincides with reality.

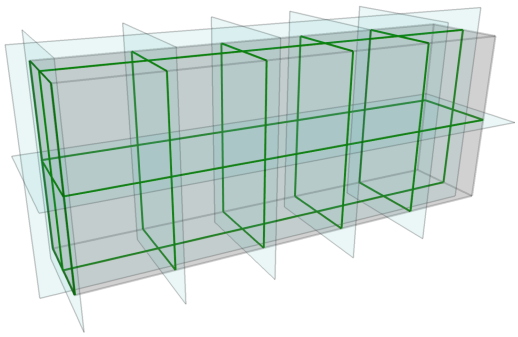
On the other hand, the image planes $\mathcal{P}_m$ for $m = 3, \ldots, 7$ are all parallel to the YZ plane, with $\mathcal{P}_3$ being the closest to the surface on which a zero Dirichlet BC is imposed, and $\mathcal{P}_7$ being the farthest. Note that the X coordinate of the latter plane was set to 9.99 instead of 10 in order to prevent the slicing of the mesh surface from producing an empty segmentation due to numerical errors that might stem from floating point arithmetic. The deformation evaluated on $\mathcal{P}_m$ for $m = 3, \ldots, 7$ will primarily be out of plane. This allows us to quantify how well the algorithm does at capturing the in-plane components of deformation when it is primarilyo ut-of-plane. If the in-plane deformation calculated by the LDDMM algorithm is satisfactory within some user-specified tolerance, we may be able to infer the nature of the out-of-plane deformation through other means. An example of this will be discussed later as a potential for future work.

4.1.1.2 Thick-walled Cylinder

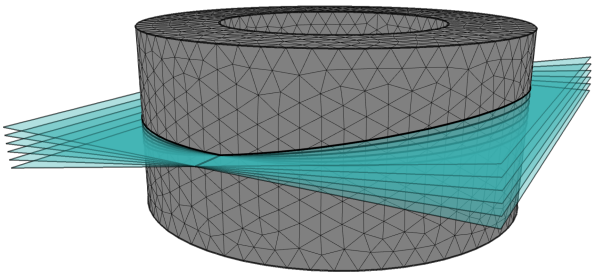
Recall that the mesh of the thick-walled cylinder was designed to achieve a solution that is as symmetric with respect to its axis as possible. Thus, deformation on the image plane $\mathcal{P}_1$ for the thick-walled cylinder will be planar within numerical errors, as was true for the



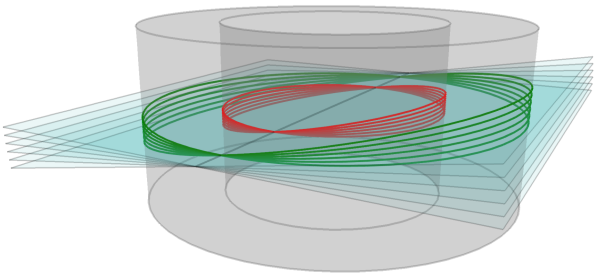
(a) Box mesh ($\mathcal{R}_0$)



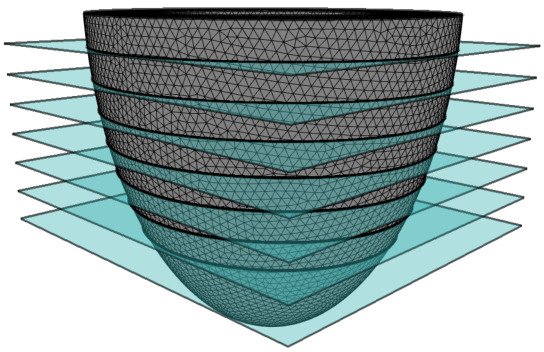
(b) Box segmentations



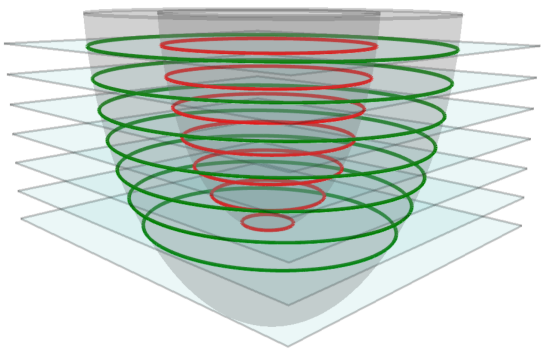
(c) Cylinder mesh ($\mathcal{D}_0$)



(d) Cylinder segmentations



(e) Ellipsoid mesh ($\mathcal{V}^2$)



(f) Ellipsoid segmentations

Figure 4.1: Imaging planes shown alongside of 3D tetrahedral meshes, as well as the resulting segmentations within each plane.

first two image planes in the previous case; again allowing us to examine the accuracy of the registration when the deformation is truly planar, but now in a thick-walled geometry. Unlike the previous case, we decided to incrementally rotate the image plane with respect to the X-axis in order to study how misalignments between the image plane and the object of interest will affect the results. Hence the components of the unit normals shown in Table 4.1 vary while the origin of the plane remains the same.

4.1.1.3 Eccentric Thick-walled Ellipsoid

Last but not least, the image planes for the ellipsoid are considered. In this case, all image planes are parallel to the XY-plane, just like the base on which a zero Dirichlet BC is imposed. The difference between the image planes is their location along the long axis of the ellipsoid; similar to the long axis of the LV. Each image plane will see varying amounts of out-of-plane deformation based on its proximity to the base. Given the nature of the geometry and the BCs applied, this is the closest to knowing the true deformation of an LV-like analytical geometry possible from the examples considered in Section 2.4.

Now that the segmentations have been extracted for all of the synthetic cases, it is time to generate triangular meshes representing the region enclosed by them.

4.1.2 Triangular Mesh Generation

Unlike the segmentations generated from patient images, the synthetic segmentations generated in the previous section are not restricted to the XY-plane, with the exception of the $\mathcal{P}_1$ plane for the cylinder. As a result, we must transform their coordinates with rotations and translations in order to have them all lie within the XY-plane. This is due to the fact that only the X and Y components of points along a 2D boundary must be passed to MeshPy when generating a 2D triangular mesh [37]. Inspecting the unit normals for all image planes in all cases, one can see that the necessary rotations and translations can be determined relatively easily, e.g. using simple orthogonal tensors to represent the rotation and its inverse.

Once the segmentations have been transformed to the XY-plane, they are passed to MeshPy to generate a triangular mesh. Then, the inverse transformation is applied in order to bring the mesh back to the image plane that the segmentations originally came from. This gives us a set of target meshes on which the LDDMM algorithm is then applied in order to solve for the $\boldsymbol{\varphi}_{i,j}$ maps, as discussed in Section 3.2.

4.1.3 Surface Matching

The surface matching algorithm applied to the target meshes generated in the previous section is the same as was discussed for the patient data. However, the mesh resolution, $\sigma_{\max}$, and the kernel sigma ratio, $\gamma_{\max}$, were varied in order to study their effects

on the convergence and accuracy of the resulting deformation. Namely, the combinations of mesh resolutions and kernel sigma ratios tested were all possible combinations of $\sigma_{\max} = 3.5, 4, 4.5, 5, 6, 7, 8, 10, 15, 20$, and $\gamma_{\max} = 1, 2, 5, 10, 20$. We will show the error of the displacement and curve strain fields with these various combinations to validate our choice of $\sigma_{\max}$ and $\gamma_{\max}$ used in Chapter 3. However, we must first discuss how the error is computed.

4.2 Computing Errors

Another challenging aspect in this problem is deciding the best way to compare the displacement field obtained through the LDDMM algorithm, $\mathbf{u}_{\text{reg}}$, with the field from the original solid mechanics simulation, $\mathbf{u}_{\mathcal{W}}$, where $\mathcal{W}$ is a subset of the body of interest. Specifically, the challenge lies in the fact that the subdomain seen within an image plane at any given time t, is in fact a subset of the *current* configuration at time t. We consider two options for comparing $\mathbf{u}_{\text{reg}}$ and $\mathbf{u}_{\mathcal{W}}$:

1. Lagrangian error: here, we evaluate $\mathbf{u}_{\mathcal{W}}$ at

$$\mathcal{W} = \mathcal{P}_m \cap \mathcal{B}_0. \tag{4.1}$$

   I.e., we evaluate the displacement at the set of material points that initially coincided with the image plane. We will use the subscript l (instead of $\mathcal{W}$) to denote this form of evaluation.

2. Eulerian error: here, we evaluate $\mathbf{u}_{\mathcal{W}}$ at

$$\mathcal{W} = \mathcal{P}_m \cap \boldsymbol{\varphi}\left(\mathcal{B}_0, t\right) = \mathcal{P}_m \cap \mathcal{B}. \tag{4.2}$$

   I.e., we evaluate the displacement at the set of material points that lie in the image plane at time t, which is not guaranteed to be a constant set of points. We will use the subscript e (instead of $\mathcal{W}$) to denote this form of evaluation.

Note that the Eulerian point of view is a closer representation of reality when inferring displacement from medical images. Though the Lagrangian point of view is the more ideal of the two since this is the true displacement, rather than an impression of what the displacement might be. We will present both interpretations for the displacement side by side only to illustrate how these error computations differ from one another in different contexts.

From here on out, we will denote the difference of the two displacement fields and the magnitude of the difference by

$$\mathbf{d}_a(\mathbf{u}) = \mathbf{u}_{\text{reg}} - \mathbf{u}_a, \tag{4.3}$$

and

$$d_a(\mathbf{u}) = \|\mathbf{d}_a(\mathbf{u})\|, \tag{4.4}$$

respectively, where a is either e or l, and the specific norm used will be given later. Furthermore, we can decompose the difference of the vectors, referred to as the error vector from here on out, into the two components

$$\mathbf{d}_a(\mathbf{u}) = \mathbf{d}_{ap}(\mathbf{u}) + \mathbf{d}_{an}(\mathbf{u}), \tag{4.5a}$$

where

$$\mathbf{d}_{an} = (\mathbf{N} \cdot \mathbf{d})\,\mathbf{N}, \tag{4.5b}$$

and

$$\mathbf{d}_{ap} = \mathbf{d} - \mathbf{d}_{an} = \mathbf{d} - (\mathbf{N} \cdot \mathbf{d})\,\mathbf{N} = (\mathbf{I} - \mathbf{N} \otimes \mathbf{N})\,\mathbf{d} \tag{4.5c}$$

are the components normal and parallel to the plane with normal vector $\mathbf{N}$, respectively. Keep in mind that

$$\|\mathbf{d}_a\|^2 = \|\mathbf{d}_{ap}\|^2 + \|\mathbf{d}_{an}\|^2, \tag{4.6}$$

but

$$\|\mathbf{d}_a\| \neq \|\mathbf{d}_{ap}\| + \|\mathbf{d}_{an}\| \tag{4.7}$$

as we investigate the components in the error for all three synthetic cases.

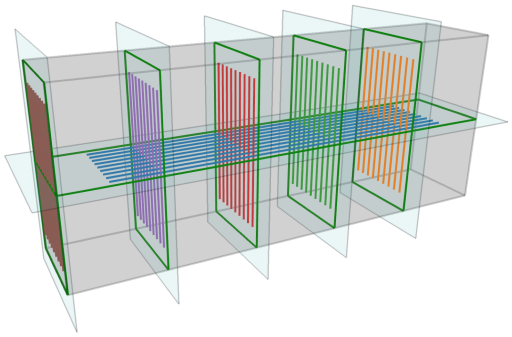
While the time steps used for the registration framework are identical to those in the solid mechanics simulation, we only use the time steps at which the segmentations were extracted ($\Delta t = 0.1$ [s]). We have noticed that, in some cases, the mapped template mesh does not match the true shape of the domain within the image plane in between these sampled time steps, and thus seems like an unfair comparison.

In addition to the error in displacement, we are interested in the error of strain measures. The right Cauchy-Green strain tensor, $\mathbf{C}$, was used to study the myocardial deformation in patients. Thus, we perform a similar analysis in which we evaluate $\mathbf{C}$ in the Eulerian manner, discussed in Equation (4.2), and compare the strains obtained from the solid mechanics displacement, as well as registration, and compute their difference. Note that all derivatives with respect to spatial variables must be taken with respect to the reference configuration. This is due to the fact that

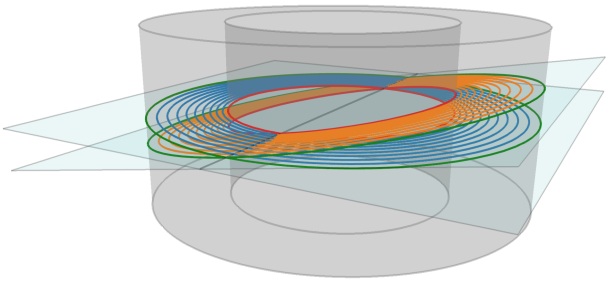
$$\mathbf{C} = \mathbf{F}^T\mathbf{F} = \left[\mathbf{I} + \mathrm{Grad}(\mathbf{u})\right]^T \left[\mathbf{I} + \mathrm{Grad}(\mathbf{u})\right], \tag{4.8}$$

where Grad is the gradient with respect to the reference configuration. Deforming the body and then computing the gradient is akin to computing the gradient of the displacement with respect to the spatial frame. Once $\mathbf{C}$ has been computed for all $\mathbf{X} \in \mathcal{B}_0$ and $t \in [t_0, t_f]$, the evaluation domain, $\mathcal{W}$, can be determined.

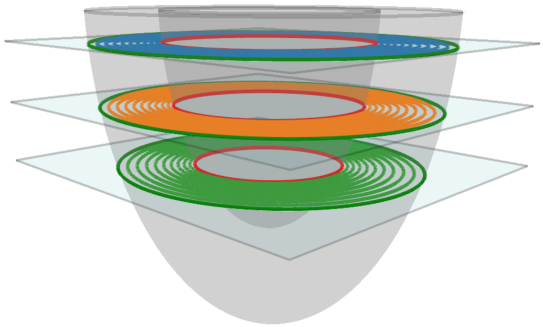
We will only consider the *radial* and *circumferential* components as defined in Section 3.3. In the case of the box domain, $\mathcal{R}_0$, curves parallel to coordinate planes are considered as the interior curves. Selected interior curves for the three domains are shown in Figure 4.2.



(a) Interior curves for the box domain ($\mathcal{R}_0$)



(b) Interior curves for the cylindrical domain ($\mathcal{D}_0$)



(c) Interior curves for the ellipsoidal domain ($\mathcal{V}^2$)

Figure 4.2: Selected interior curves used for computing the strain tensor components for the synthetic cases.

Similar to Equation (4.3), we will consider the error of the tensor components given by

$$D_{nn,a}(\mathbf{C}_{\text{reg}}) = |\mathbf{n} \cdot (\mathbf{C}_{\text{reg}} - \mathbf{C}_a)\,\mathbf{n}|\,, \tag{4.9a}$$

$$D_{nt,a}(\mathbf{C}_{\text{reg}}) = |\mathbf{n} \cdot (\mathbf{C}_{\text{reg}} - \mathbf{C}_a)\,\mathbf{t}|\,, \tag{4.9b}$$

and

$$D_{tt,a}(\mathbf{C}_{\text{reg}}) = |\mathbf{t} \cdot (\mathbf{C}_{\text{reg}} - \mathbf{C}_a)\,\mathbf{t}|\,. \tag{4.9c}$$

In practice, we first compute the components of $\mathbf{C}_{\text{reg}}$ and $\mathbf{C}_a$ separately, and then their difference.

4.3 Accuracy of Framework in Synthetic Cases

The error reported for all cases is the maximum magnitude of the error vector and tensor components throughout space and sampled time. I.e.,

$$\|\mathbf{d}_a\| = \max_{k\in\mathcal{I}} \left(\max_{\mathbf{x}\in\mathcal{W}} \sqrt{\mathbf{d}_a(t_k)\cdot\mathbf{d}_a(t_k)} \right), \tag{4.10}$$

and

$$\|D_{ij,a}\| = \max_{k\in\mathcal{I}} \left(\max_{\mathbf{x}\in\mathcal{W}} D_{ij,a}(t_k) \right), \tag{4.11}$$

where ij denotes the tensor components, a is either e or l, and $\mathcal{I} = \{0, 10, \ldots, 100\}$. In addition to this norm, the reader is also encouraged to keep the image planes for each case as described in Section 4.1.1, and summarized in Table 4.1, in mind.

4.3.1 Displacement Errors

4.3.1.1 Three-dimensional Box

First, we consider the error for the box domain, $\mathcal{R}_0$. It will be helpful to keep in mind that the maximum displacement obtained from the 3D solid mechanics simulation for all of t is 2.2 mm, as can be seen in Figure 2.5. Additionally, recall that the first two planes, $\mathcal{P}_1$ and $\mathcal{P}_2$, are planes of symmetry, while $\mathcal{P}_m$ for $m = 3, \ldots, 7$ are planes perpendicular to the loading vector and are thus expected to only capture a small component of the true displacement. As can be seen in Figure 4.3, the Lagrangian error will be shown on the left, and the Eulerian error on the right. In this case, we can see that, overall, the Eulerian error has a smaller magnitude than the Lagrangian error. Furthermore, we see that the displacement error—both Lagrangian and Eulerian—increases as the value of $\gamma_{\max}$ increases (including the zoomed-in portion for $m = 5$ on the last row). However, the distance from the Dirichlet boundary seems to play a more prominent role on the error for $m = 3, \ldots, 7$ than $\gamma_{\max}$.

The near-invariance to the choice of $\gamma_{\max}$ for $m = 3, \ldots, 7$ is also apparent when inspecting the perpendicular component of the error vector, $\mathbf{d}_{an}$ for $a = e, l$, as shown in Figure 4.4. We see that all of the curves are essentially the same for any given slice, except for the Eulerian error computed on $\mathcal{P}_1$ and $\mathcal{P}_2$. Though, even then, the variation with respect to $\gamma_{\max}$ is small in comparison to the magnitude of the full error vector shown in Figure 4.3. This is not too surprising since the template-target pairs throughout all of t remain planar for any given image plane, and thus the LDDMM algorithm does not result in any out-of-plane component to the displacement $\mathbf{u}_{\text{reg}}$. Thus, it is unreasonable to expect a good estimation of out-of-plane motion when our framework is not provided with even the slightest hint of what might be happening outside of the image plane.

A more fair comparison is the parallel component of the error vector, $\mathbf{d}_{ap}$ for $a = e, l$, as shown in Figure 4.5. Here, we see that the parallel component of the error, $\mathbf{d}_{ap}$, is the main

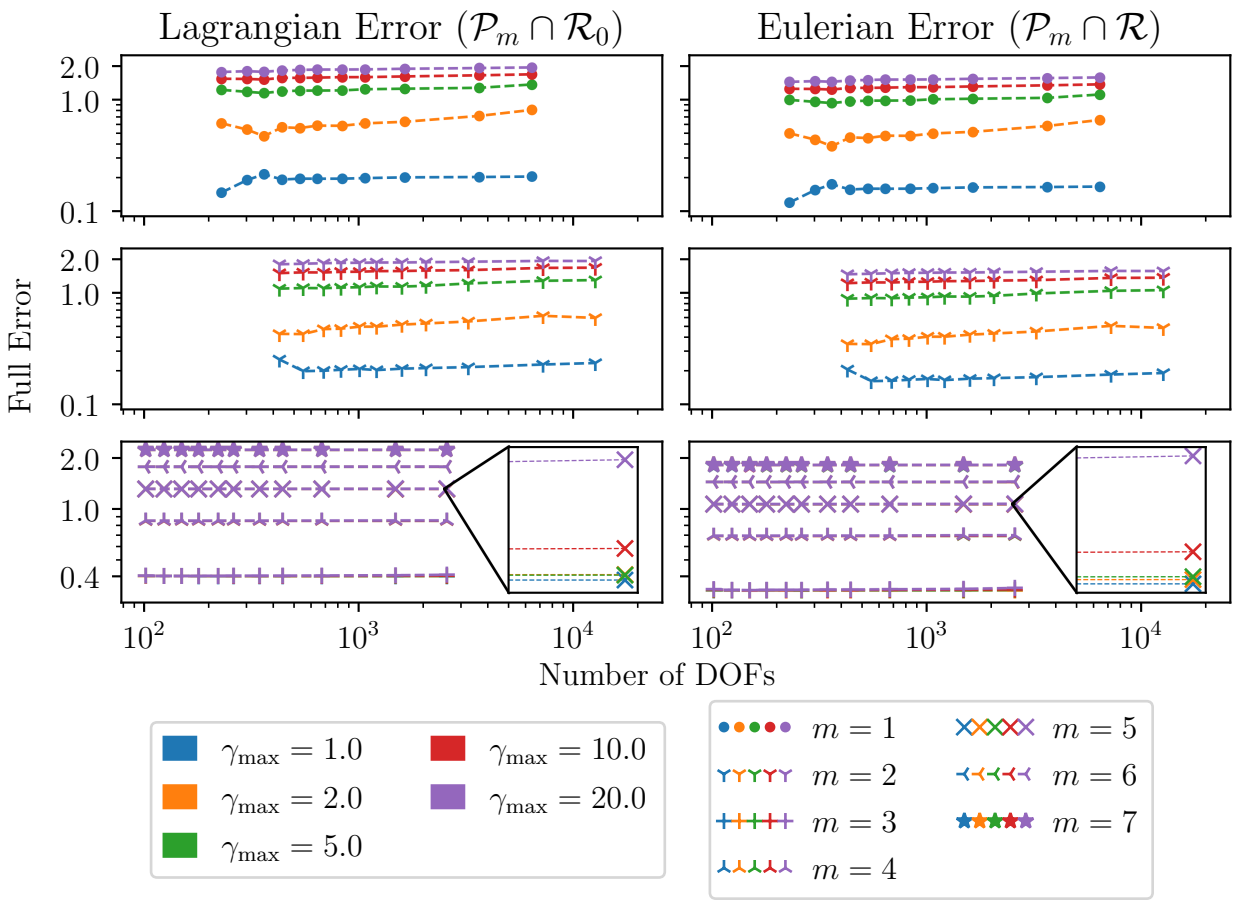


Figure 4.3: Full Lagrangian and Eulerian displacement error for the box domain, $\mathcal{R}_0$.

contributor to the full error for $m = 1, 2$. The opposite is true for $m = 3, \ldots, 7$. Furthermore, we can see that, for the most part, the error also increases with increasing $\gamma_{\max}$ for the latter set of image planes.

Overall, the displacement error does not change drastically when holding the image plane and $\gamma_{\max}$ constant while changing $\sigma_{\max}$, i.e. only changing the degrees of freedom (DOF). It is also clear that, for this domain, choosing $\gamma_{\max} = 1$ provides the most accurate results for all image planes.

4.3.1.2 Thick-walled Cylinder

Next, we consider the thick-walled cylinder, $\hat{\mathcal{D}}_0$, which was loaded with a time-varying internal pressure. While a different configuration was chosen for testing the registration framework, the maximum displacement magnitude remains the same. This is due to the fact the new reference was chosen to be the configuration in which the maximum displacement with respect to $\mathcal{D}_0$ occurs. Thus, the maximum displacement with respect to the new configura-

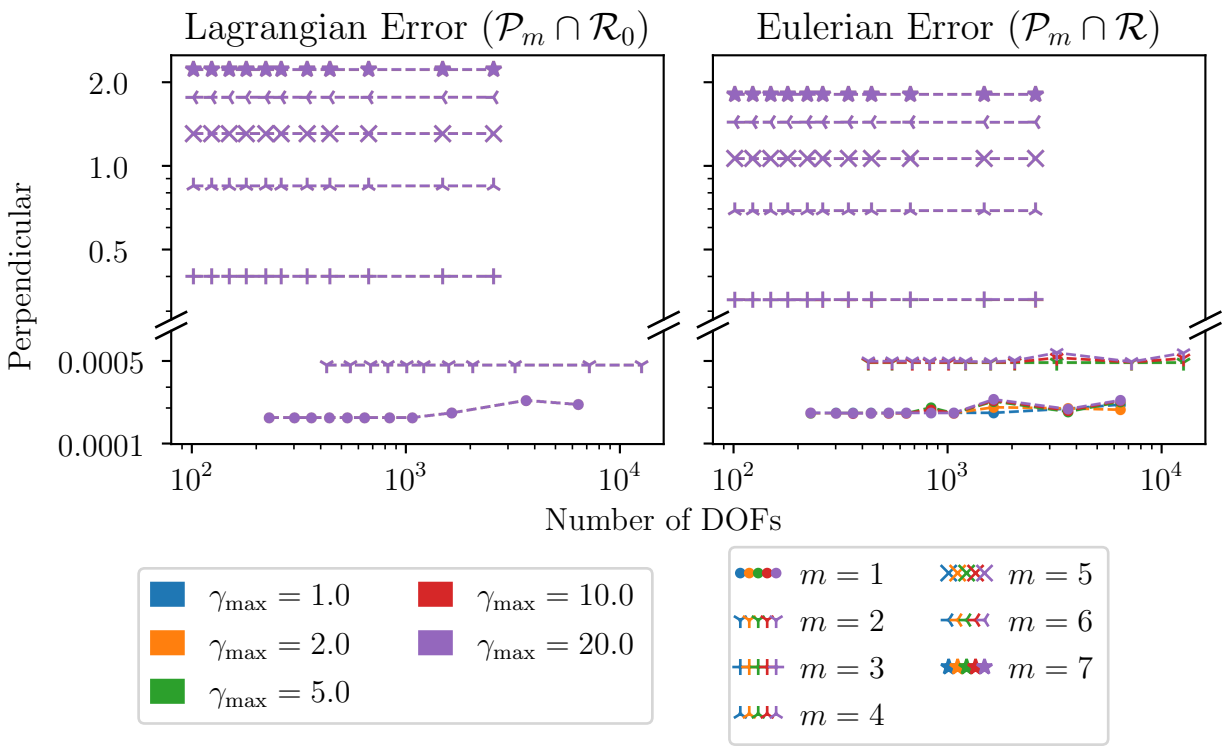


Figure 4.4: The perpendicular component of the Lagrangian and Eulerian error for the box domain, $\mathcal{R}_0$.

tion, $\hat{\mathcal{D}}_0$, is the one that takes us back to $\mathcal{D}_0$. The maximum displacement obtained from this 3D deformation is 0.72 mm, as shown in Figure 2.7.

As with the box domain, we see that overall, the Eulerian error is smaller in magnitude than the Lagrangian error for all cases, as can be seen in Figure 4.6. Likewise, we see that the error increases when $\gamma_{\max}$ increases. Cases for which the registration results produced negative Jacobian values at any point in the cycle were marked with a hollow dimmed circular marker, as well as dimmed connecting line segments.

The perpendicular component of the error, shown in Figure 4.7, exhibits a similar pattern to what we saw for the box domain. However, recall that the image planes are incrementally rotated with respect to the x-axis to mimic a misalignment of the imaging device with a plane of symmetry. Nevertheless, we expect the out-of-plane component of motion to increase with increasing m as before. This manifests in a larger error as m increases. Furthermore, the contribution of the perpendicular error for $m = 1$ is about three orders of magnitude smaller, which is expected due to the fact that it is a plane of symmetry.

The parallel component of the error is shown in Figure 4.8. We can see that the error curves remain relatively closer to each other across image planes when the perpendicular component of error is excluded. Thus, our registration framework seems to do a fair job of

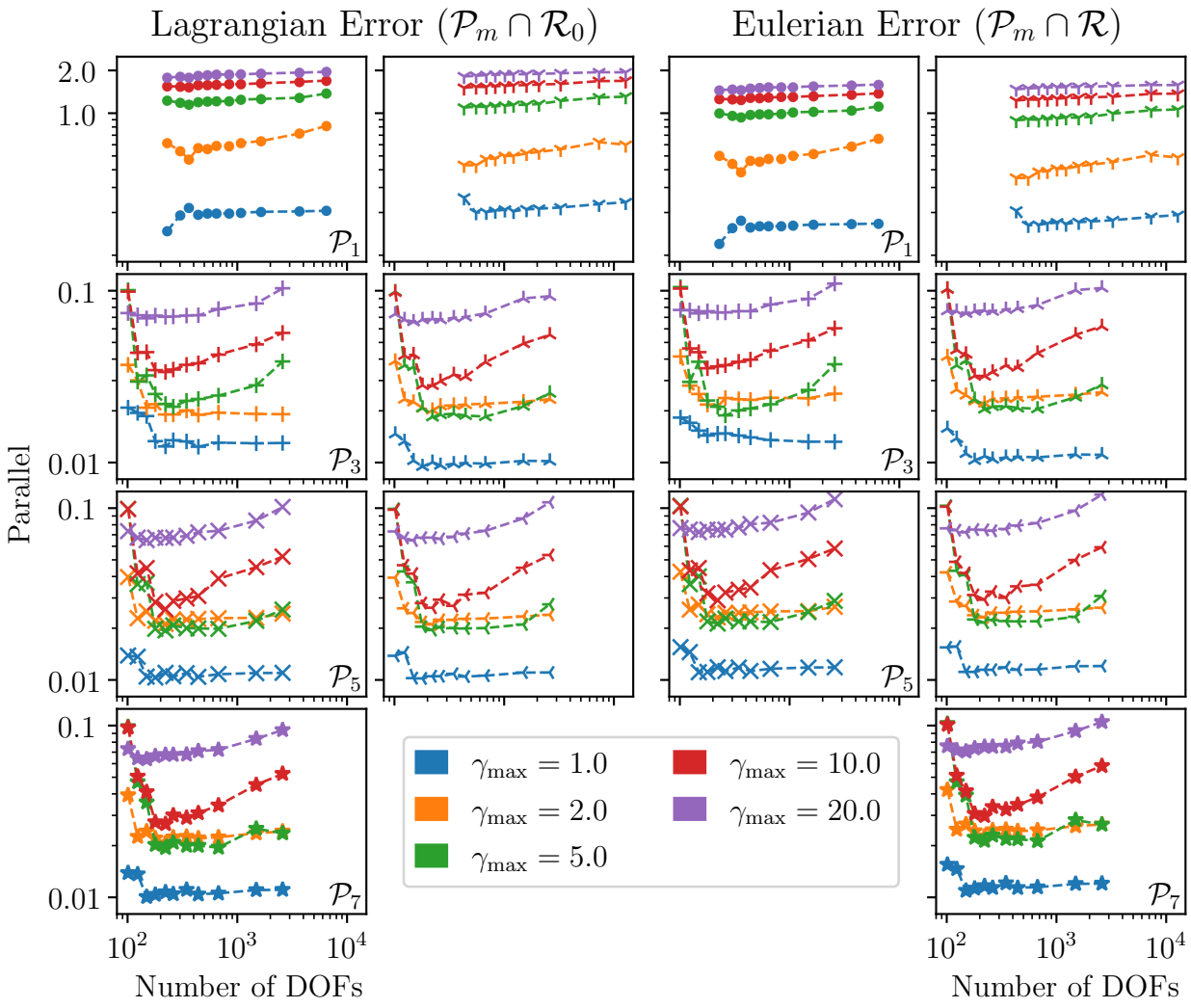


Figure 4.5: Parallel component of the Lagrangian and Eulerian error for the box domain, $\mathcal{R}_0$.

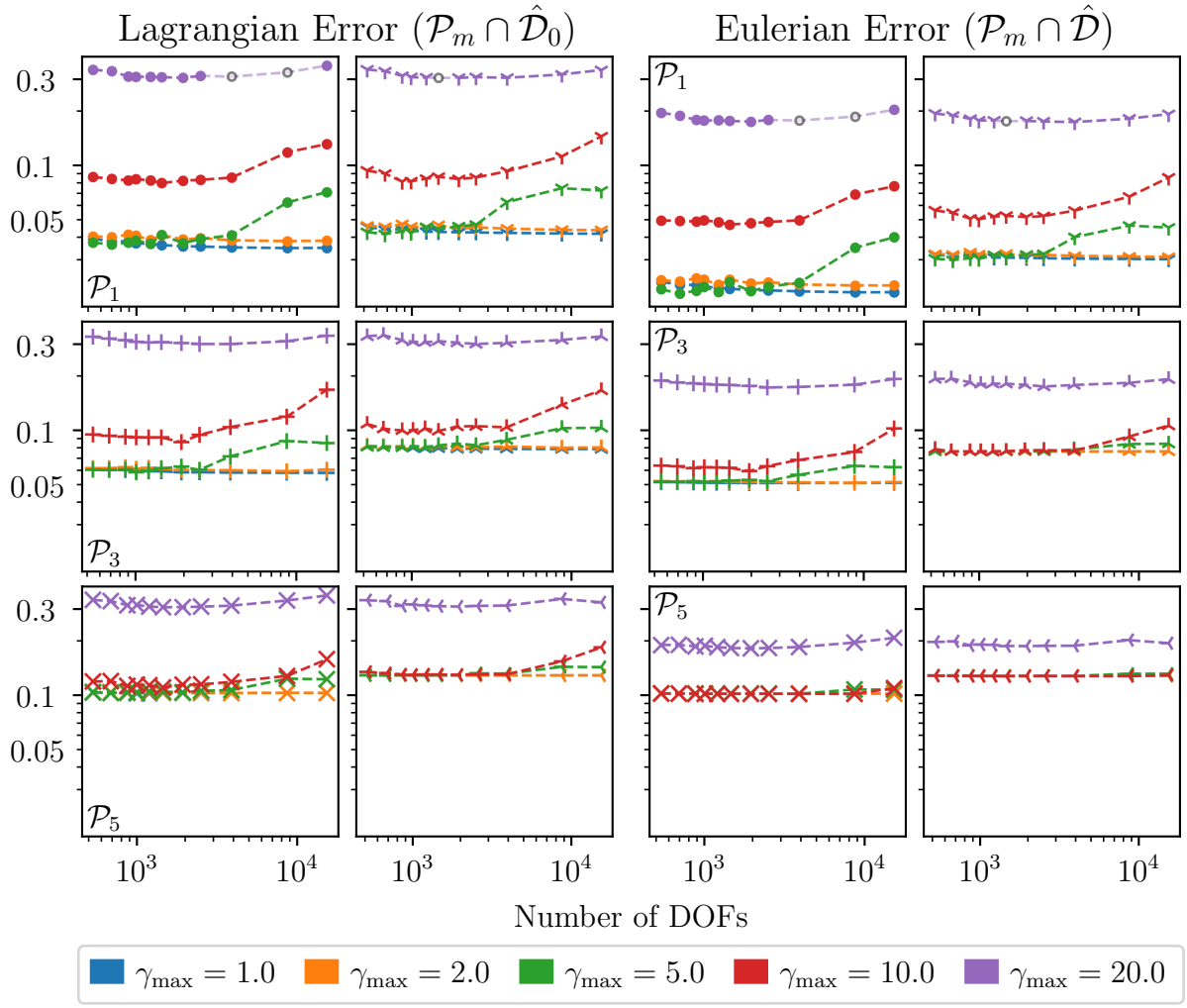


Figure 4.6: Full Lagrangian and Eulerian error for the cylinder domain, $\hat{\mathcal{D}}_0$.

capturing the in-plane motion in this case. Note that we also see an increase in parallel error when m increases for $m = 1, \ldots, 4$, and for high enough number DOFs when $\gamma_{\max} = 5$ and 10. The error is most consistent for $\gamma_{\max} = 1$ and $\gamma_{\max} = 2$.

Overall, the error does not change drastically when only varying the number of DOFs for $\gamma_{\max} = 1, 2$. However, we do see fluctuations in the parallel component of the error with respect to the number of DOFs for $\gamma_{\max} = 5, 10$. Thus, the top contenders in this case are $\gamma_{\max} = 1, 2$.

4.3.1.3 Eccentric Thick-walled Ellipsoid

Similar to the cylinder, the new configuration for the ellipsoid, $\hat{\mathcal{V}}_0^2$, was chosen such that the maximum displacement remains the same. The maximum displacement obtained from this

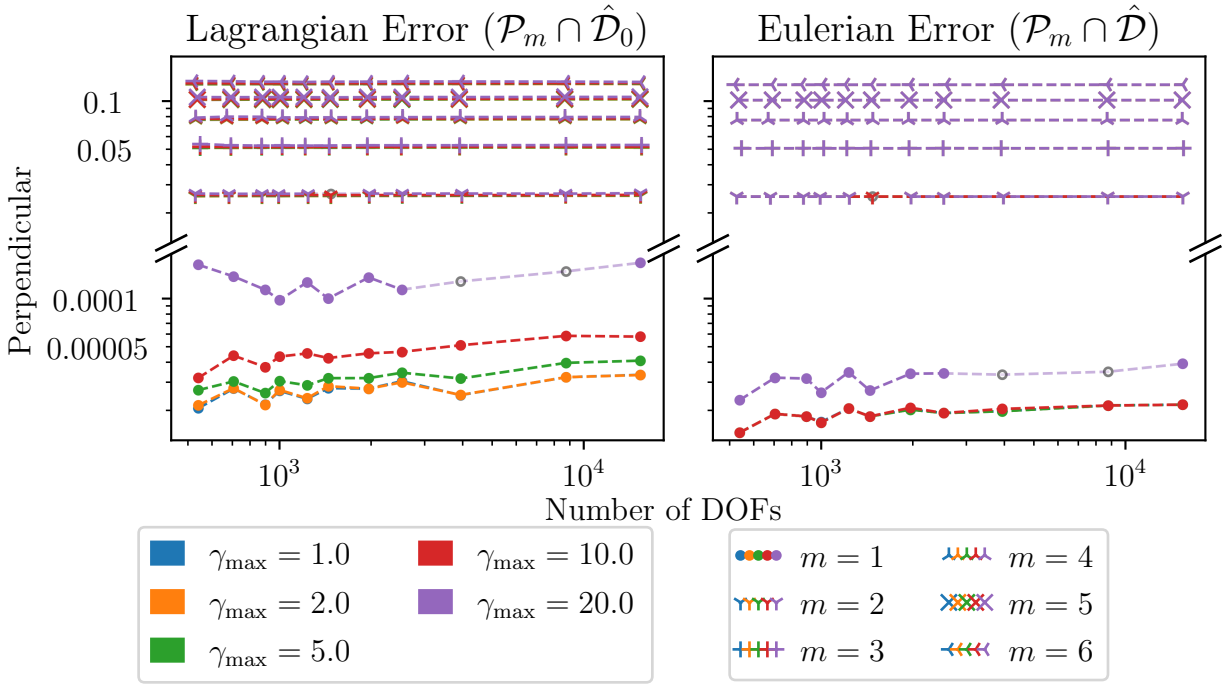


Figure 4.7: The perpendicular component of the Lagrangian and Eulerian error for the cylinder domain, $\hat{\mathcal{D}}_0$.

3D deformation is 5.5 mm, as can be seen in Figure 2.12.

The magnitude of the full error vector for the ellipsoid is shown in Figure 4.9. Markers for cases whose results produced negative Jacobian values were omitted and the connecting line segments were dimmed. The markers were omitted in this case due to the fact that a total of 88 out of the 385 cases exhibited a negative Jacobian value at some point in the cycle. Unlike the previous two cases, the Eulerian error is larger in magnitude than the Lagrangian error for all cases. It is also clear from this figure that the error does not vary as much when $\gamma_{\max}$ is varied for a given image plane once a certain number of DOFs is reached. Previously, we only saw this in cases where the perpendicular component of the error dominated over the parallel one. In fact, the higher values of $\gamma_{\max}$ seem to thrive over their counterparts for lower values of $\sigma_{\max}$ in some cases. More astounding is the magnitude in error seen for $m = 5, 6, 7$—the error for $m = 7$ with $\sigma_{\max} = 3.5$ and $\gamma_{\max} = 1$ exceeds 20 mm, which is more than 360% of the maximum displacement.

Figure 4.10 shows the perpendicular components of the error. The pattern is the same as the two previous cases. To elaborate, the perpendicular component of error depends more on the location of the image plane rather than $\sigma_{\max}$ or $\gamma_{\max}$; except for the two outliers shown in the plot of the Lagrangian error. The reader should note that these results show that the perpendicular component of the error is not solely responsible for the excessively large magnitude in error seen for $m = 5, 6, 7$.

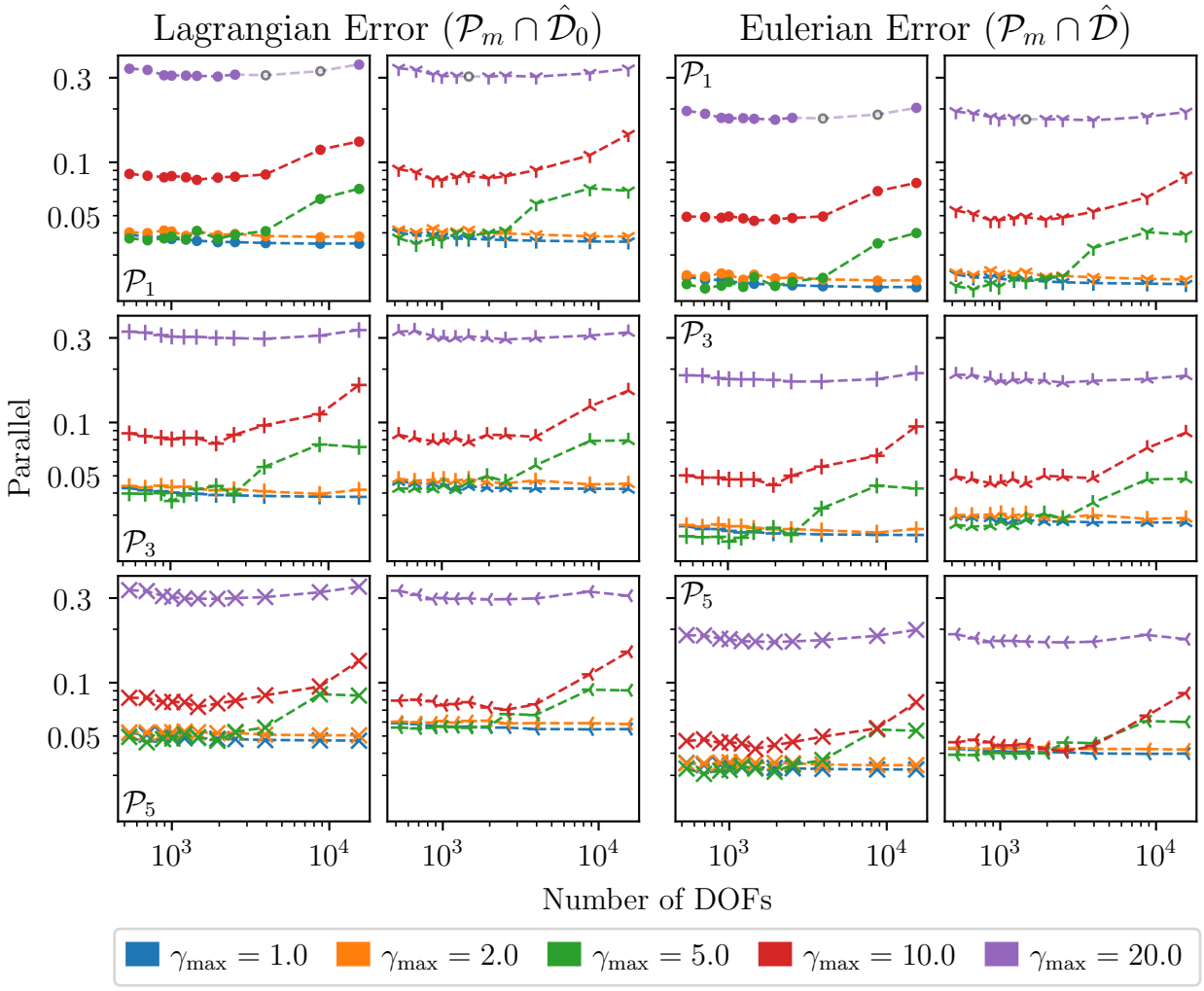


Figure 4.8: The parallel component of the Lagrangian and Eulerian error for the cylinder domain, $\hat{\mathcal{D}}_0$.

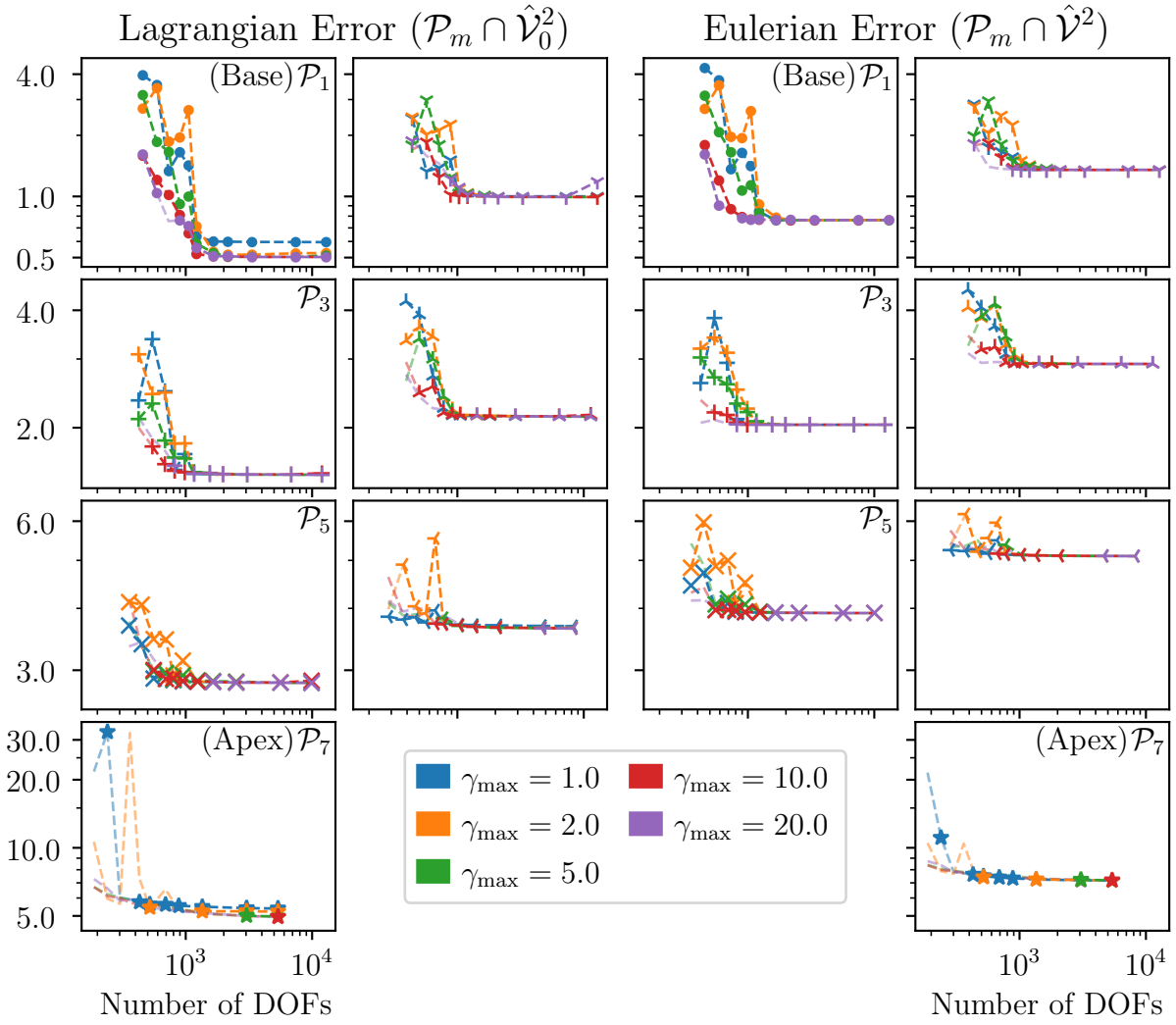


Figure 4.9: Full Lagrangian and Eulerian error for the ellipsoid domain, $\hat{\mathcal{V}}_0^2$.

The contribution from the parallel component to the full error can be seen in Figure 4.11. We see that, while still more than desired, the parallel error mostly remains below 4 mm for $m = 1, \ldots, 4$. However, the error grows dramatically the further the image plane is from the base and closer to the apex of the geometry. This is crucial to keep in mind when deciding which parameters and image planes to use when applying the registration framework to patient data.

While the results obtained with $\gamma_{\max} = 1$ excelled for the box and cylinder domains, that was not the case for the ellipsoid. Nor were the error magnitudes near-invariant to the change in number of DOFs within each image plane as before. Looking at the parallel component specifically, results obtained with $\gamma_{\max} = 10$ surpassed the rest in accuracy for $m = 1, \ldots, 4$.

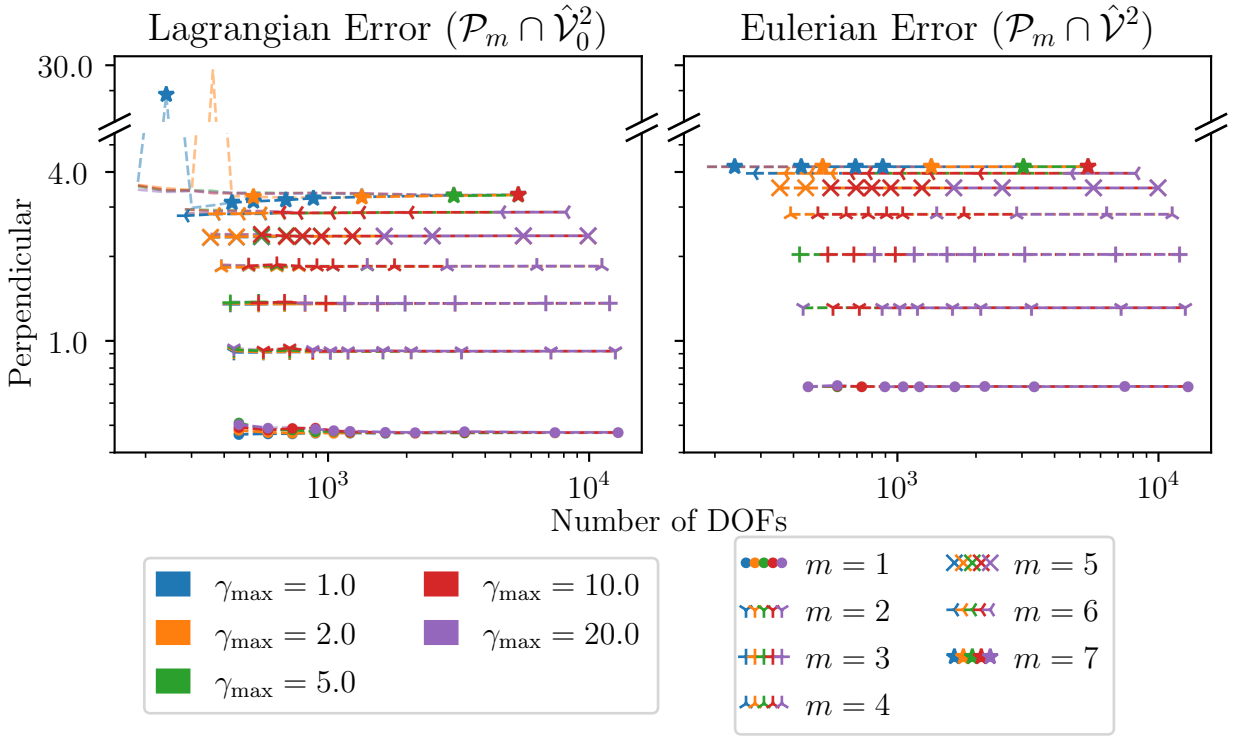


Figure 4.10: The perpendicular component of the Lagrangian and Eulerian error for the ellipsoid domain, $\hat{\mathcal{V}}_0^2$.

Overall, the lowest full Lagrangian error was obtained on the image plane closest to the base of the geometry, $\mathcal{P}_1$, with $\sigma_{\max} = 20$ and $\gamma_{\max} = 20$; a maximum of 0.50 mm was obtained. On the other hand, the lower parallel component of the Lagrangian error was obtained on the same image plane, but with $\sigma_{\max} = 10$ and $\gamma_{\max} = 10$; a maximum error of 0.29 mm was obtained. The lowest full Eulerian error, 0.76 mm, was obtained on $\mathcal{P}_1$, with $\sigma_{\max} = 8$ and $\gamma_{\max} = 1$. This combination of $\sigma_{\max}$ and $\gamma_{\max}$ also gave the lowest parallel component of the Eulerian error, coming in at 0.34 mm. This makes the combination of $\sigma_{\max} = 8$ and $\gamma_{\max} = 1$ the top contenders for this synthetic case—when looking at the displacement error—since the Eulerian error is of higher interest.

Overall, the performance of the registration framework when selecting $\gamma_{\max} = 1, 2$ seem to be the most favorable when considering the Eulerian error of the parallel component of displacement. A summary of the minimum values of the norm defined in Equation (4.10) is given in Table 4.2. This summary will be discussed in more depth later. The error of the curve strain components will be discussed next.

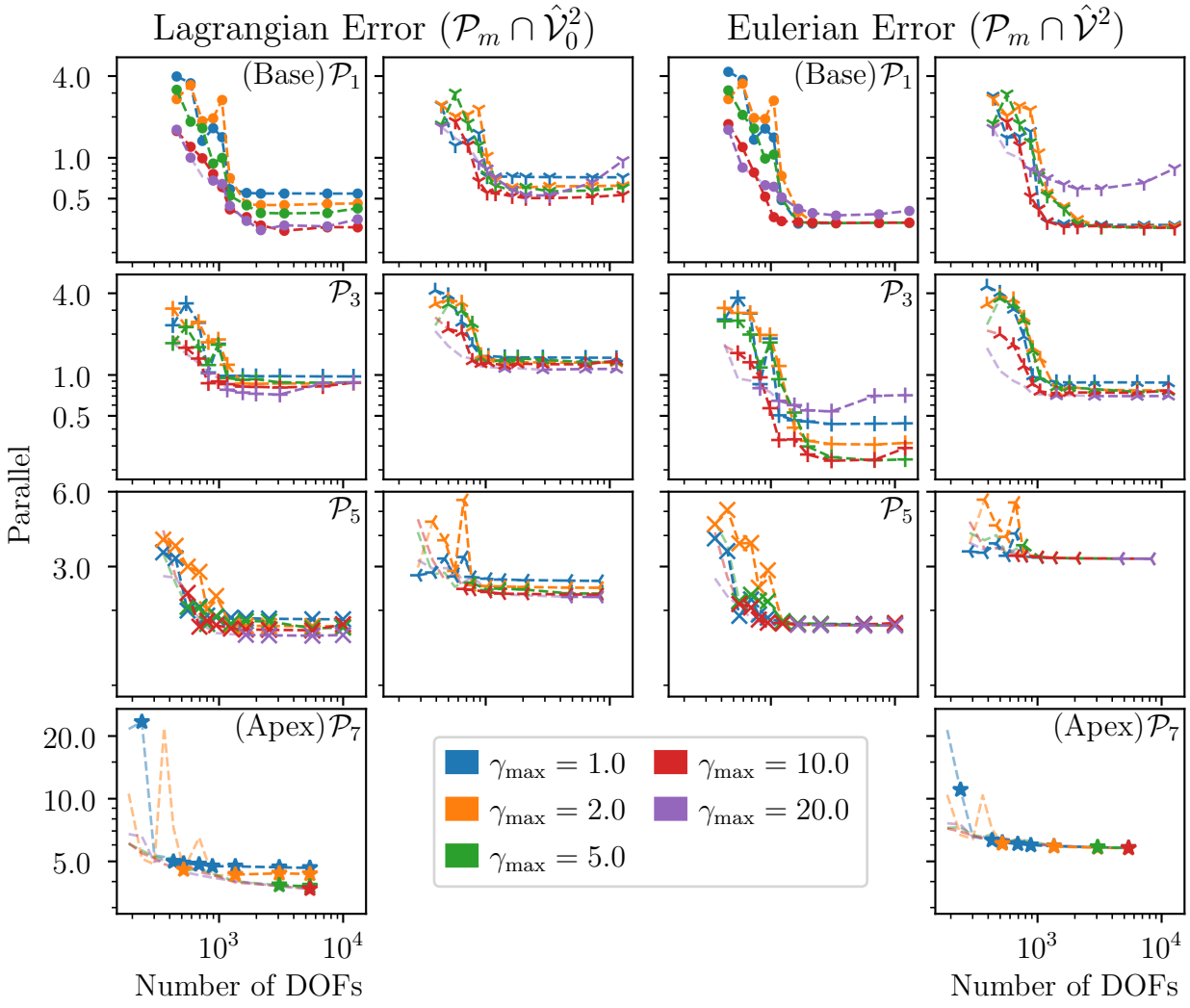


Figure 4.11: The parallel component of the Lagrangian and Eulerian error for the ellipsoid domain, $\hat{\mathcal{V}}_0^2$.

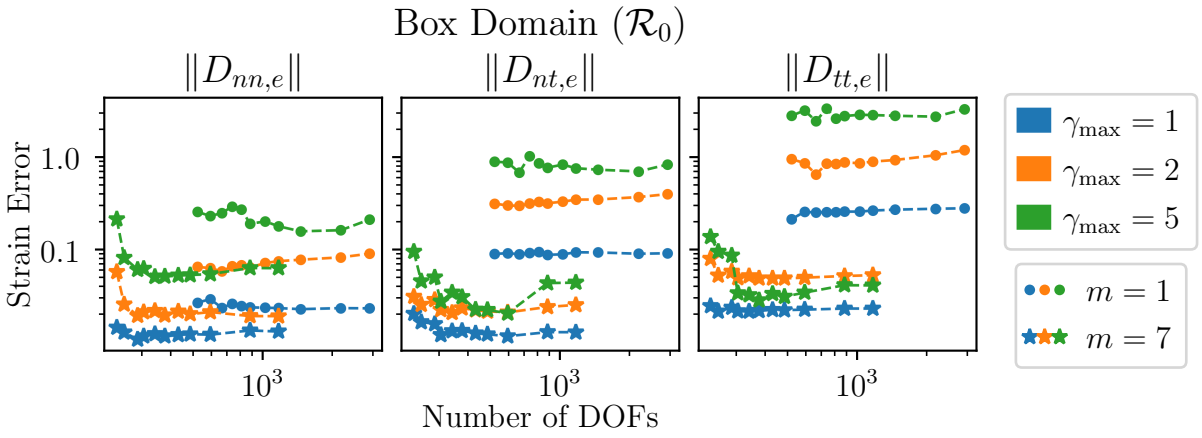


Figure 4.12: In-plane Lagrangian and Eulerian strain error for the box domain, $\mathcal{R}_0$.

4.3.2 Strain Error

In order to compute the error of the strain tensor components, we must first determine the interior curves for each image plane as was done in Section 3.3 for the patient data. The same procedure is followed for each case. Some resulting interior curves for all three cases with $\sigma_{\max} = 10$ and $\gamma_{\max} = 1$ are shown in Figure 4.2. Unlike the presentation of the displacement error, only the Eulerian error of the curve strain components will be presented here. This is due to the fact that the Lagrangian and Eulerian errors exhibits the same qualitative features.

4.3.2.1 Three-dimensional Box

Here, only the errors on the image planes $\mathcal{P}_1$ and $\mathcal{P}_7$ since $\mathcal{P}_1$ is a plane of symmetry like $\mathcal{P}_2$, and the displacement errors evaluated on $\mathcal{P}_7$ provide an upper bound for those on planes $m = 3, \ldots, 6$, as seen in Figure 4.4. Furthermore, only the $\gamma_{\max} = 1, 2, 5$ are presented since it is clear from Section 4.3.1.1 that these choices outperform $\gamma_{\max} = 10, 20$. The three components of the strain error, radial (nn), shear (nt), and circumferential (tt) can be seen in Figure 4.12.

The errors of all three components evaluated on $\mathcal{P}_1$ are larger than their counterparts evaluated on $\mathcal{P}_7$. This is partly due to the fact that the deformation on $\mathcal{P}_1$ is more substantial than that on $\mathcal{P}_7$. Within each plane, the radial components of the right Cauchy-Green tensor produce a smaller error than the other two components when the same $\sigma_{\max}$ and $\gamma_{\max}$ values are used. In addition, setting $\gamma_{\max} = 1$ produces the lowest and most consistent error with respect to $\sigma_{\max}$ as was the case with the displacement error.

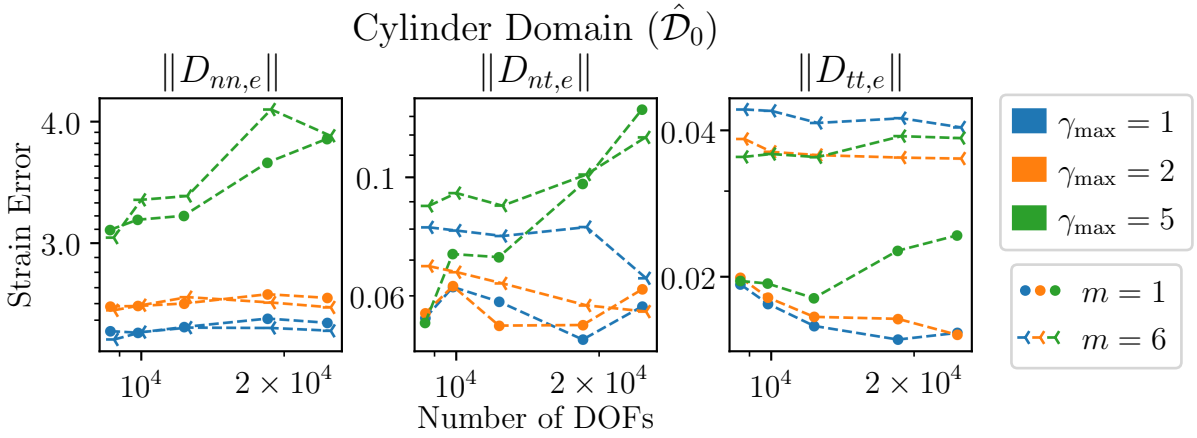


Figure 4.13: In-plane Lagrangian and Eulerian strain error for the cylindrical domain, $\hat{\mathcal{D}}_0$.

4.3.2.2 Thick-walled Cylinder

Similar to above, the curve strain errors presented here are limited to those evaluated on $\mathcal{P}_1$, a plane of symmetry, and $\mathcal{P}_6$, which is the plane most misaligned with $\mathcal{P}_1$. Thus the curve strain errors for the remaining planes are bounded by the results shown in Figure 4.13. Additionally, only $\sigma_{\max} = 7, 8, 10, 15, 20$, and $\gamma_{\max} = 1, 2, 5$ are shown. The mesh resolution was limited to the higher values since it is where a larger variation of error was observed for the displacement error in Figure 4.6. The higher values of $\gamma_{\max}$ were omitted due to their poor performance, including producing negative values of the Jacobian in some cases.

Unlike the results for the box domain, these are not as uniform for all three components of the curve strain. Most notable is the fact that the errors of the radial component (nn) are clustered based on the value of $\gamma_{\max}$. Furthermore, the errors of the shear component (nt) appear completely unstructured—the value of $\gamma_{\max}$ that produces the lowest error in this component changes with the choice of $\sigma_{\max}$. On the other hand, it is clear that $\gamma_{\max} = 1$ and $\gamma_{\max} = 2$ give the lowest error for the circumferential component (tt) when evaluated on $\mathcal{P}_1$ and $\mathcal{P}_6$, respectively. For the case of the thick-walled cylinder, the better choice is between $\gamma_{\max} = 1$ or 2.

4.3.2.3 Eccentric Thick-walled Ellipsoid

Similar to the cylinder case, only the errors pertaining to $\sigma_{\max} = 7, 8, 10, 15, 20$ are presented here. The values $\gamma_{\max} = 1, 2, 5, 10$ are shown, and the imaging planes used are $\mathcal{P}_1$ and $\mathcal{P}_4$. These errors are shown in Figure 4.14. The lower mesh resolutions were omitted along with $\gamma_{\max} = 20$ due to the extensive amount of cases that exhibited negative Jacobian values for this range, while $\gamma_{\max} = 10$ was added to the cases presented here due to its unexpectedly good performance shown in Figure 4.11.

It can be seen that $\gamma_{\max} = 10$ outperforms the other cases for all mesh resolutions when computing the radial component of error on $\mathcal{P}_1$. However, $\gamma_{\max} = 2$ outperforms the rest on

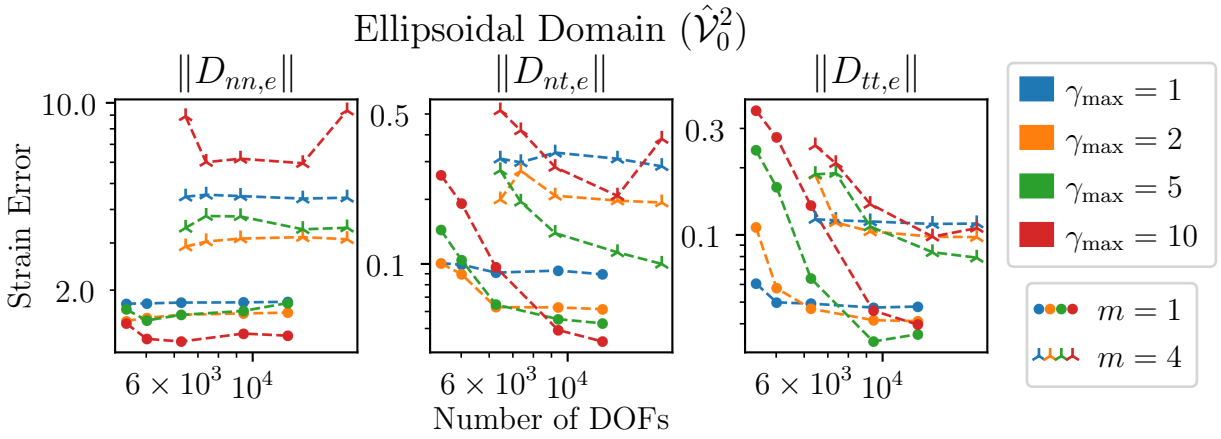


Figure 4.14: In-plane Lagrangian and Eulerian strain error for the ellipsoidal domain, $\hat{\mathcal{V}}_0^2$.

$\mathcal{P}_4$. For the shear component, $\gamma_{\max} = 10$ produces the lowest error for $\sigma_{\max} = 15$ and 20, but exhibits the highest error for all other values of $\sigma_{\max}$ on $\mathcal{P}_1$. On $\mathcal{P}_4$ $\gamma_{\max} = 5$ performs the best for all values of $\sigma_{\max}$ except $\sigma_{\max} = 7$, which $\gamma_{\max} = 2$ swapping places with it. Similar behavior is observed in the circumferential component of the error; the lowest error is produced either by $\gamma_{\max} = 2$ or 5.

It is important to keep in mind the the goal is to compute the myocardial strain in a clinical settings, and thus computational cost should be considered. A summary of the best performing parameters will now be provided with this consideration.

4.4 Discussion

In this chapter, results from solid mechanics simulations were used as synthetic data to test the registration framework developed in Chapter 3. This provided a means to directly compare the deformation computed through the registration framework with the deformation it was meant to reproduce. The main difference between the patient and synthetic cases is how the input to the framework was obtained—input from patient data requires segmentation of medical images, while computational geometric algorithms provided by VTK were used for the synthetic cases. After this, the framework is nearly identical, except for the rotations of segmentations that were required for some image planes. Throughout this chapter, errors of displacement and curve strain components as evaluated from a Lagrangian and Eulerian point of view were reported. This was done to understand how the choice of mesh resolution, $\sigma_{\max}$, the kernel radius ratio, $\gamma_{\max}$, and the location of the image planes, $\mathcal{P}_m$, affect the accuracy of results.

From this analysis, it is clear that a secret combination of $\sigma_{\max}$ and $\gamma_{\max}$ values that performs the best for all image planes in all cases does not exist. A summary of the minimum

			Plane		$\sigma_{\max}$		$\gamma_{\max}$		**Mininum**	
Domain	**Field**	**Component**	L	E	L	E	L	E	L	E
Box	Disp. ($\times 10^{-3}$ mm)	Full	1	1	3.5	3.5	1	1	148.3	120.7
		Perp.	1	1	3.5	4.5	1	10	0.1688	0.1835
		Parallel	5	5	8	8	1	1	10.44	11.26
	Curve Str. ($\times 10^{-2}$)	Radial	7	7	4.5	4.5	1	1	1.357	1.366
		Shear	3	3	5	8	1	1	1.038	0.9571
		Circum.	4	4	6	6	1	1	2.059	1.997
Cylinder	Disp. ($\times 10^{-4}$ mm)	Full	1	1	15	5.5	2	2	380.1	356.0
		Perp.	1	1	3.5	4	1	2	0.207	0.3346
		Parallel	1	3	15	5.5	2	2	380.1	344.3
	Curve Str. ($\times 1$)	Radial	5	6	4.5	5	1	1	2.296	2.299
		Shear	5	4	20	8	1	2	0.0432	0.0451
		Circum.	3	3	5.5	5.5	2	2	0.0500	0.498
Ellipsoid	Disp. ($\times 1$ mm)	Full	1	1	20	8	20	1	0.5024	0.7615
		Perp.	1	1	3.5	5.5	1	1	0.4660	0.6890
		Parallel	1	1	10	8	10	1	0.2920	0.3354
	Curve Str. ($\times 1$)	Radial	1	1	10	10	10	10	0.8765	1.296
		Shear	1	1	20	20	1	10	0.04828	0.04381
		Circum.	1	1	20	20	10	1	0.06781	0.04886

Table 4.2: Summary of the Lagrangian (L) and Eulerian (E) errors for all components of displacement and curve strain considered.

values obtained for the error norms defined in Equations (4.11) and (4.10) is given in Table 4.2. It is clear that $\gamma_{\max} = 1$ and 2 produce the lowest error most often, as observed in previous sections. Though $\gamma_{\max} = 10$ performs very well when considering the radial and shear components of the curve strain for the ellipsoid. A higher value of $\gamma_{\max}$ generally leads to a larger range of strain, while a lower value spreads the deformation throughout the domain in a more uniform manner. In order to obtain the best of both worlds, a value of $\gamma_{\max} = 2$ was chosen when studying patient data.

Additionally, the errors obtained for the lower values of $\gamma_{\max}$ was mostly agnostic to the choice of mesh resolution, $\sigma_{\max}$. Thus, the more important factors to consider when choosing $\sigma_{\max}$ are the computational cost and the level of spatial granularity desired. With these two factors in mind, $\sigma_{\max}$ was set to 10 in order to provide enough spatial detail when computing strain fields as well as to lower the computational cost required by higher values, $\sigma_{\max} = 15$ and 20. This rise in computational cost is caused by the increase in triangular elements used to represent the domain within the image planes.

Furthermore, the image planes used to study myocardial strain were limited to those closest to the base of the left ventricle. The thick-walled ellipsoid case clearly showed that the error increases unacceptably as the image plane used to compute strain moves away from the ventricular base. This is believed to be caused by the increase in the amount of out-of-plane deformation.

This analysis with synthetic cases permitted a more intelligent choice of parameters for the registration framework when applied to patient data. It can also serve as a template for studying algorithms designed to analyze patient data. While medical images provide a lot of information about the physiology of a patient, there is still no ethical way of determining the true deformation. Thus, algorithms, such as the framework discussed here, must be vetted with due diligence to ensure their efficacy before relying on them to determine treatment.

Chapter 5

Summary

The major contributions of this work include the development of (1) a registration framework for computing the LV myocardial strain from planar images, and (2) validation tests using synthetic data to which results can be directly compared. The full registration framework consisted of segmenting, or delineating, the endo- and epicardium from 2D CMR image sequences. These segmentations were then used to create a 2D triangular mesh to represent the LV myocardium, which were then used as target-template pairs for the LDDMM registration algorithm. The LDDMM algorithm provided diffeomorphic maps for the full sequence of planar triangular meshes for each patient, allowing the computation of the right Cauchy-Green strain tensor. The solution to the Laplace equation was used to programmatically define the circumferential and radial directions in general LV geometries. Throughout this framework, it was assumed that the deformation of the myocardium fully remained within each given image plane. This led to the development of validation tests to understand the effect of such an assumption on the accuracy of the results.

In order to validate the registration framework, synthetic displacement data was generated through three-dimensional solid mechanics simulations of a box, thick-walled cylinder, and a thick-walled eccentric ellipsoid. These results were then used to generate segmentations that served as input to the registration framework. Generating the segmentations from simulations allowed the direct comparison of the *true results*, so to speak, with the results from the registration algorithm while varying mesh resolution and the kernel function radii used to represent the deformation maps. Once the mesh resolution and kernel function radius that produced the most accurate registration results were determined, they were used to apply the registration framework to patient images. In this study, it was found that the circumferential components of the right Cauchy-Green strain tensor showed the most statistically significant difference between control patients and those with CKD.

As with all research, there are several shortcomings with the approaches presented here. For one, the assumption of planar motion is physically inaccurate and hence cannot be expected to capture all of the information. With that said, further studies on the effect of out-of-plane motion and whether any inference can be made on its nature from planar images can greatly improve the framework. Furthermore, it would be worthwhile to investigate

and compare the utility of other registration algorithms for this framework—it is possible that different algorithms will capture different modes of cardiac deformation. It is also important to keep in mind that the statistical analysis performed in Chapter 3 was rudimentary in nature, thus a larger population is necessary to obtain more conclusive clinical results. Additionally, the data per patient was limited to one image plane, though it is possible that using multiple may provide a more comprehensive quantification of myocardial strain for any given patient.

In terms of the validation tests, the non-linear neo-Hookean model used to generate synthetic data adheres more to the behavior of rubber as opposed to LV tissue for which various anisotropic models have been developed [19, 28]. The simpler model was used in order to reduce the amount of variables in the overall system, though future tests should also consider constitutive equations specifically designed to describe the LV. Other assumptions include the use of quasi-steady formulations to circumvent wave reflections with Dirichlet boundary conditions, as well as the pre-loading and reassigning of the reference configuration done to avoid buckling of solutions and the need to compute residual stresses.

Despite the shortcomings of this work, we believe it provides a foundation for further development of tools to compute myocardial strain along with validation tests. The validation tests can be further authenticated by accounting for the physics between the tissue motion and the acquiring of images; either through simulation of the data acquisition process, or the use of phantoms. Once satisfactory validation of such myocardial strain-computing tools has been performed, further investigation of the correlation between the 2D deformation captured and the true 3D deformation can be studied to potentially infer the full range of motion. E.g., through the use of incompressibility constraints. Such an accomplishment will provide a more complete understanding of LV kinematics and in turn improve diagnostics in clinical settings.

Bibliography

[1] *Surgery: Medicine*. July 2022. URL: https://www.britannica.com/science/surgery-medicine (visited on 07/24/2022).

[2] Jared Ortaliza Ortaliza, Krutika Amin Twitter, and Cynthia Cox Twitter. *Covid-19 leading cause of death ranking*. June 2022. URL: https://www.healthsystemtracker.org/brief/covid-19-leading-cause-of-death-ranking/.

[3] Rajiv Saran et al. "US Renal Data System 2016 Annual Data Report: Epidemiology of Kidney Disease in the United States". In: *Am. J. Kidney Dis.* 69.3 (Mar. 2017), Suppl. ISSN: 1523-6838. DOI: 10.1053/j.ajkd.2016.12.004. eprint: 28236831.

[4] Walter J. Rogers Jr. et al. "Early Contrast-Enhanced MRI Predicts Late Functional Recovery After Reperfused Myocardial Infarction". In: *Circulation* (Feb. 1999). DOI: 10.1161/01.CIR.99.6.744.

[5] David Bello et al. "Gadolinium Cardiovascular Magnetic Resonance Predicts Reversible Myocardial Dysfunction and Remodeling in Patients With Heart Failure Undergoing β-Blocker Therapy". In: *Circulation* (Oct. 2003). URL: https://www.ahajournals.org/doi/full/10.1161/01.CIR.0000095029.57483.60.

[6] David Bello et al. "Infarct morphology identifies patients with substrate for sustained ventricular tachycardia". In: *J. Am. Coll. Cardiol.* (Apr. 2005). URL: https://www.jacc.org/doi/abs/10.1016/j.jacc.2004.12.057.

[7] *FDA Drug Safety Communication: New warnings for using gadolinium-based contrast agents in patients with kidney dysfunction*. June 2018. URL: https://www.fda.gov/drugs/drug-safety-and-availability/fda-drug-safety-communication-new-warnings-using-gadolinium-based-contrast-agents-patients-kidney (visited on 07/24/2022).

[8] Katherine C. Wu et al. "Late Gadolinium Enhancement by Cardiovascular Magnetic Resonance Heralds an Adverse Prognosis in Nonischemic Cardiomyopathy". In: *J. Am. Coll. Cardiol.* (June 2008). URL: https://www.jacc.org/doi/abs/10.1016/j.jacc.2008.03.018.

[9] Tori A. Stromp et al. "Gadolinium free cardiovascular magnetic resonance with 2-point Cine balanced steady state free precession". In: *J. Cardiovasc. Magn. Reson.* 17.1 (Dec. 2015), pp. 1–11. ISSN: 1532-429X. DOI: 10.1186/s12968-015-0194-1.

[10] Tori A. Stromp et al. "Quantitative Gadolinium-Free Cardiac Fibrosis Imaging in End Stage Renal Disease Patients Reveals A Longitudinal Correlation with Structural and Functional Decline". In: *Sci. Rep.* 8.16972 (Nov. 2018), pp. 1–10. ISSN: 2045-2322. DOI: 10.1038/s41598-018-35394-4.

[11] M. S. Amzulescu et al. "Myocardial strain imaging: review of general principles, validation, and sources of discrepancies". In: *Eur. Heart J. Cardiovasc. Imaging* 20.6 (June 2019), pp. 605–619. ISSN: 2047-2404. DOI: 10.1093/ehjci/jez041.

[12] Yuichi Notomi et al. "Measurement of Ventricular Torsion by Two-Dimensional Ultrasound Speckle Tracking Imaging". In: *J. Am. Coll. Cardiol.* (June 2005). URL: https://www.jacc.org/doi/abs/10.1016/j.jacc.2005.02.082.

[13] Ruta Jasaityte, Brecht Heyde, and Jan D'hooge. "Current State of Three-Dimensional Myocardial Strain Estimation Using Echocardiography". In: *J. Am. Soc. Echocardiogr.* 26.1 (Jan. 2013), pp. 15–28. ISSN: 0894-7317. DOI: 10.1016/j.echo.2012.10.005.

[14] E. A. Zerhouni et al. "Human heart: tagging with MR imaging–a method for noninvasive assessment of myocardial motion." In: *Radiology* (Oct. 1988). DOI: 10.1148/radiology.169.1.3420283.

[15] Miguel A. Rodriguez, Christoph M. Augustin, and Shawn C. Shadden. "FEniCS mechanics: A package for continuum mechanics simulations". In: *SoftwareX* 9 (Jan. 2019), pp. 107–111. ISSN: 2352-7110. DOI: 10.1016/j.softx.2018.10.005.

[16] Peter Chadwick. *Continuum mechanics: concise theory and problems*. Dover Publications, 1999.

[17] Thomas J. R. Hughes. *The finite element method: linear static and dynamic finite element analysis*. Dover publication, Inc, 2007.

[18] L. R. G. Treloar. "The elasticity of a network of long-chain molecules—II". In: *Trans. Faraday Soc.* 39.0 (Jan. 1943), pp. 241–246. ISSN: 0014-7672. DOI: 10.1039/TF9433900241.

[19] Julius M. Guccione, Kevin D. Costa, and Andrew D. Mcculloch. "Finite element stress analysis of left ventricular mechanics in the beating dog heart". In: *Journal of Biomechanics* 28.10 (1995), pp. 1167–1177. DOI: 10.1016/0021-9290(94)00174-3.

[20] Christoph M. Augustin et al. "Anatomically accurate high resolution modeling of human whole heart electromechanics: A strongly scalable algebraic multigrid solver method for nonlinear deformation". In: *J. Comput. Phys.* 305 (Jan. 2016), pp. 622–646. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2015.10.045.

[21] Minliang Liu, Liang Liang, and Wei Sun. "A new inverse method for estimation of in vivo mechanical properties of the aortic wall". In: *J. Mech. Behav. Biomed. Mater.* 72 (Aug. 2017), pp. 148–158. ISSN: 1751-6161. DOI: 10.1016/j.jmbbm.2017.05.001.

[22] Kevin L. Sack et al. "Construction and Validation of Subject-Specific Biventricular Finite-Element Models of Healthy and Failing Swine Hearts From High-Resolution DT-MRI". In: *Front. Physiol.* 0 (May 2018). ISSN: 1664-042X. DOI: 10.3389/fphys.2018.00539.

[23] P. J. Flory. "Thermodynamic relations for high elastic materials". In: *Trans. Faraday Soc.* 57.0 (Jan. 1961), pp. 829–838. ISSN: 0014-7672. DOI: 10.1039/TF9615700829.

[24] Nathan M. Newmark. "A Method of Computation for Structural Dynamics". In: *Journal of the Engineering Mechanics Division* 85.3 (July 1959), pp. 67–94. DOI: 10.1061/JMCEA3.0000098.

[25] *FEniCS Project*. https://fenicsproject.org/. (Visited on 11/01/2016).

[26] Wikipedia. *List of finite element software packages*. URL: https://en.wikipedia.org/wiki/List_of_finite_element_software_packages.

[27] *FEniCS Mechanics Documentation*. Nov. 2020. URL: https://shaddenlab.gitlab.io/fenicsmechanics (visited on 01/01/2020).

[28] Jay D. Humphrey. "Mechanics Of The Arterial Wall: Review And Directions". In: *Critical Reviews in Biomedical Engineering* 23.1-2 (1995), pp. 1–162. DOI: 10.1615/critrevbiomedeng.v23.i1-2.10.

[29] *CBC Block*. July 2022. URL: https://bitbucket.org/fenics-apps/cbc.block/src/master (visited on 07/24/2022).

[30] Alexandra G. Gheorghe et al. "Cardiac left ventricular myocardial tissue density, evaluated by computed tomography and autopsy". In: *BMC Med. Imaging* 19.1 (Dec. 2019), pp. 1–9. ISSN: 1471-2342. DOI: 10.1186/s12880-019-0326-4.

[31] Sander Land et al. "Verification of cardiac mechanics software: benchmark problems and solutions for testing active and passive material behaviour". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science* 471.2184 (Aug. 2015), p. 20150641. DOI: 10.1098/rspa.2015.0641.

[32] Eric Pierce. *Diagram of the human heart*. June 2006. URL: https://commons.wikimedia.org/wiki/File:Diagram_of_the_human_heart_(cropped).svg (visited on 07/27/2022).

[33] M. Faisal Beg et al. "Computing Large Deformation Metric Mappings via Geodesic Flows of Diffeomorphisms". In: *Int. J. Comput. Vision* 61.2 (Feb. 2005), pp. 139–157. ISSN: 1573-1405. DOI: 10.1023/B:VISI.0000043755.93987.aa.

[34] Laurent Younes. *Shapes and Diffeomorphisms*. Berlin, Germany: Springer, 2010. ISBN: 978-3-642-12055-8. URL: https://link.springer.com/book/10.1007/978-3-642-12055-8.

[35] Einar Heiberg et al. "Design and validation of Segment - freely available software for cardiovascular image analysis". In: *BMC Med. Imaging* 10.1 (Dec. 2010), pp. 1–13. ISSN: 1471-2342. DOI: 10.1186/1471-2342-10-1.

[36] Jane Tufvesson et al. "Validation and Development of a New Automatic Algorithm for Time-Resolved Segmentation of the Left Ventricle in Magnetic Resonance Imaging". In: *Biomed Res. Int.* 2015 (June 2015), p. 970357. ISSN: 2314-6133. DOI: 10.1155/2015/970357.

[37] *MeshPy 2020.1 documentation*. Dec. 2021. URL: https://documen.tician.de/meshpy (visited on 01/15/2020).

[38] Jonathan Richard Shewchuk. "Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator". In: *Applied Computational Geometry: Towards Geometric Engineering*. Ed. by Ming C. Lin and Dinesh Manocha. Vol. 1148. Lecture Notes in Computer Science. From the First ACM Workshop on Applied Computational Geometry. Springer-Verlag, May 1996, pp. 203–222.

[39] *Triangle: A Two-Dimensional Quality Mesh Generator and Delaunay Triangulator*. July 2022. URL: http://www.cs.cmu.edu/~quake/triangle.html (visited on 07/24/2022).

[40] *TetGen - Tetraedrischen Netze generieren - Software Download*. July 2022. URL: https://www.berlios.de/software/tetgen (visited on 07/24/2022).

[41] Christophe Geuzaine and Jean-François Remacle. "Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities". In: *Int. J. Numer. Methods Eng.* 79.11 (Sept. 2009), pp. 1309–1331. ISSN: 0029-5981. DOI: 10.1002/nme.2579.

[42] Laurent Younes. *Registration*. https://bitbucket.org/laurent_younes/registration/src/master/. 2012. (Visited on 06/27/2019).

[43] Claudio de Lucia et al. "Echocardiographic Strain Analysis for the Early Detection of Left Ventricular Systolic/Diastolic Dysfunction and Dyssynchrony in a Mouse Model of Physiological Aging". In: *J. Gerontol. A Biol. Sci. Med. Sci.* 74.4 (Mar. 2019), pp. 455–461. ISSN: 1079-5006. DOI: 10.1093/gerona/gly139.

[44] Chun-mei Li et al. "Value of Three-Dimensional Speckle-Tracking in Detecting Left Ventricular Dysfunction in Patients with Aortic Valvular Diseases". In: *J. Am. Soc. Echocardiogr.* 26.11 (Nov. 2013), pp. 1245–1252. ISSN: 0894-7317. DOI: 10.1016/j.echo.2013.07.018.

[45] Yolanda Vives-Gilabert et al. "Left ventricular myocardial dysfunction in arrhythmogenic cardiomyopathy with left ventricular involvement: A door to improving diagnosis". In: *Int. J. Cardiol.* 274 (Jan. 2019), pp. 237–244. ISSN: 0167-5273. DOI: 10.1016/j.ijcard.2018.09.024.

[46] *VTK - The Visualization Toolkit*. July 2022. URL: www.vtk.org (visited on 07/22/2022).

[47] *scipy.stats.ttest_ind — SciPy v1.8.1 Manual*. May 2022. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_ind.html (visited on 07/02/2022).

[48] J. MacQueen. "Some methods for classification and analysis of multivariate observations". In: *Project Euclid* 5.1 (Jan. 1967), pp. 281–298.

Appendix A

Inner Wall Pressure

Two versions of C++ code were used due to differences in compatibility.

Version for FEniCS 2019.1 and later.

C++ code

```
#define _USE_MATH_DEFINES

#include<math.h>
#include<algorithm>
#include<iostream>
#include<stdexcept>

#include <pybind11/pybind11.h>
#include <pybind11/eigen.h>

#include <dolfin/function/Expression.h>
#include <dolfin/function/Constant.h>


namespace py = pybind11;


class InnerPressure : public dolfin::Expression
{
public:

  double t;
  double p0;
  double lz;
  double zmin;
  Eigen::Vector3d step_interval;
  Eigen::VectorXd jumps;

  // Create a sub-class of Expression (scalar-valued).
```

```
  InnerPressure() : dolfin::Expression() {}

  // Function to evaluate the pressure on the endocardium.
  void eval(Eigen::Ref<Eigen::VectorXd> values,
            Eigen::Ref<const Eigen::VectorXd> x) const override
  {
    // First ramp up to p0, then apply a "negative" pressure,
    // relative to p0.
    if (t < 1.0) {
      values[0] = p0*t;
    }
    else{
      double z_hat;
      double k_val;

      z_hat = (x[2] - zmin)/lz;

      double time_function = eval_time_function(t, z_hat);
      values[0] = p0*(1.0 - time_function);
    }
  }

  double eval_time_function(double a_t, double a_z_hat) const
  {
    // Compute decimal portion first since the period of the function
    // from [n, n+1] where n is an integer.
    double floor_val = floor(a_t);
    double tt = a_t - floor_val;
    double val = 0.0;

    double t0 = step_interval[0];
    double t1 = compute_t1(a_z_hat);
    double t2 = compute_t2(a_z_hat);
    double t3 = step_interval[1];

    if ((tt < 0.0) || (tt > 1.0)) {
      throw std::runtime_error("Shifted time value is outside of [0.0,
          1.0].");
    }

    if (tt >= t3) {
      val = 0.0;
    }
    else if (tt >= t2) {
      val = func2(tt, a_z_hat);
    }
    else if (tt >= t1) {
      val = 1.0;
    }
    else if (tt >= t0) {
```

```
      val = func1(tt, a_z_hat);
    }
    else {
      val = 0.0;
    }

    if (val <= 1E-10) {
      val = 0.0;
    }

    return val;
  }

private:

  double func1(double tt, double z_hat) const
  {
    // Extract the start and end for this specific function of the whole
    // piecewise function.
    double t0 = step_interval[0];
    double t1 = compute_t1(z_hat);

    double numerator = M_PI*(tt - t0);
    double denominator = t1 - t0;
    return 0.5*(1 - cos(numerator/denominator));
  }

  double func2(double tt, double z_hat) const
  {
    // Extract the start and end for this specific function of the whole
    // piecewise function.
    double t2 = compute_t2(z_hat);
    double t3 = step_interval[1];

    double numerator = M_PI*(tt - t2);
    double denominator = t3 - t2;
    return 0.5*(1 + cos(numerator/denominator));
  }

  double compute_t1(double z_hat) const
  {
    double t10 = jumps[0];
    double t11 = jumps[2];
    return t10 + z_hat*(t11 - t10);
  }

  double compute_t2(double z_hat) const
  {
    double t21 = jumps[1];
    double t20 = jumps[3];
```

```
    return t21 + z_hat*(t20 - t21);
  }
};


PYBIND11_MODULE(SIGNATURE, m)
{
  py::class_<InnerPressure, std::shared_ptr<InnerPressure>,
     dolfin::Expression>
    (m, "InnerPressure")
    .def(py::init<>())
    .def_readwrite("t", &InnerPressure::t)
    .def_readwrite("p0", &InnerPressure::p0)
    .def_readwrite("lz", &InnerPressure::lz)
    .def_readwrite("zmin", &InnerPressure::zmin)
    .def_readwrite("step_interval", &InnerPressure::step_interval)
    .def_readwrite("jumps", &InnerPressure::jumps);
}
```

Version for FEniCS before 2019.1.

C++ code

```
#define _USE_MATH_DEFINES

#include<math.h>
#include<algorithm>
#include<iostream>
#include<stdexcept>
#include<vector>

#include <dolfin/function/Expression.h>


class InnerPressure : public dolfin::Expression
{
public:

  double t;
  double p0;
  double lz;
  double zmin;

  // Create a sub-class of Expression (scalar-valued).
  InnerPressure() : dolfin::Expression() {}

  // Function to evaluate the pressure on the endocardium.
  void eval(Array<double>& values, const Array<double>& x) const
  {
    // First ramp up to p0, then apply a "negative" pressure,
```

```
    // relative to p0.
    if (t < 1.0) {
      values[0] = p0*t;
    }
    else{
      double z_hat;
      double k_val;

      z_hat = (x[2] - zmin)/lz;

      double time_function = eval_time_function(t, z_hat);
      values[0] = p0*(1.0 - time_function);
    }
  }

  double eval_time_function(double a_t, double a_z_hat) const
  {
    // Compute decimal portion first since the period of the function
    // from [n, n+1] where n is an integer.
    double floor_val = floor(a_t);
    double tt = a_t - floor_val;
    double val = 0.0;

    double t1 = compute_t1(a_z_hat);
    double t2 = compute_t2(a_z_hat);

    double t0 = 0.0;
    double t3 = 0.5;

    if ((tt < 0.0) || (tt > 1.0)) {
      throw std::runtime_error("Shifted time value is outside of [0.0,
          1.0].");
    }

    if (tt >= t3) {
      val = 0.0;
    }
    else if (tt >= t2) {
      val = func2(tt, a_z_hat);
    }
    else if (tt >= t1) {
      val = 1.0;
    }
    else if (tt >= t0) {
      val = func1(tt, a_z_hat);
    }
    else {
      val = 0.0;
    }
```

```
    if (val <= 1E-10) {
      val = 0.0;
    }

    return val;
  }

// private:

  double func1(double tt, double z_hat) const
  {
    // Extract the start and end for this specific function of the whole
    // piecewise function.
    double t1 = compute_t1(z_hat);
    double t0 = 0.0;

    double numerator = M_PI*(tt - t0);
    double denominator = t1 - t0;
    return 0.5*(1 - cos(numerator/denominator));
  }

  double func2(double tt, double z_hat) const
  {
    // Extract the start and end for this specific function of the whole
    // piecewise function.
    double t2 = compute_t2(z_hat);
    double t3 = 0.5;

    double numerator = M_PI*(tt - t2);
    double denominator = t3 - t2;
    return 0.5*(1 + cos(numerator/denominator));
  }

  double compute_t1(double z_hat) const
  {
    double t10 = 0.05;
    double t11 = 0.1;

    return t10 + z_hat*(t11 - t10);
  }

  double compute_t2(double z_hat) const
  {
    double t21 = 0.4;
    double t20 = 0.45;

    return t21 + z_hat*(t20 - t21);
  }
};
```

Python module with some testing code.

Python code

```
import os

import dolfin as dlf
import fenicsmechanics as fm
from fenicsmechanics.dolfincompat import DOLFIN_VERSION_INFO

ENDO = 20

# Discontinuities for step function.
step_interval = [0.0, 1.0, 1.0]

# Discontinuities for smoothing of step function (t10, t20, t11, t21).
jumps = [0.4, 0.4, 0.3, 0.5]

if DOLFIN_VERSION_INFO < (2019, 1):
    cfile = "./pressure_expression-v2.cc"
else:
    cfile = "./pressure_expression.cc"

with open(cfile, "r") as f:
    p_cpp = f.read()


def create_pressure_expression(p0, zmin, zmax):
    lz = zmax - zmin

    if DOLFIN_VERSION_INFO < (2019, 1):
        p_expr = dlf.Expression(p_cpp)
        p_expr.t = 0.0
        p_expr.p0 = p0
        p_expr.lz = lz
        p_expr.zmin = zmin
        p_expr.degree = 1

    else:
        cpp_module = dlf.compile_cpp_code(p_cpp)

        p_expr = dlf.CompiledExpression(cpp_module.InnerPressure(),
           t=0.0, p0=p0,
                                        lz=lz, zmin=zmin, degree=1,
                                           jumps=jumps,
                                        step_interval=step_interval)

    return p_expr


if __name__ == "__main__":
```

```
    import numpy as np
    import fenicsmechanics as fm

    p0 = 100.0
    zmin = -14.0
    zmax = -1.0

    meshfile = "meshfile.h5"
    mesh = fm.load_mesh(meshfile)
    facets = fm.load_mesh_function(meshfile, mesh)
    fcts_array = facets.array()

    bmesh = dlf.BoundaryMesh(mesh, "exterior")
    if DOLFIN_VERSION_INFO < (2019, 1):
        Vp = dlf.FunctionSpace(bmesh, "CG", 1)
    else:
        c2f = bmesh.entity_map(2).array()

        bcells = dlf.MeshFunction("size_t", bmesh,
           bmesh.topology().dim())
        bcells.set_values(fcts_array[c2f])

        endomesh = dlf.SubMesh(bmesh, bcells, ENDO)
        Vp = dlf.FunctionSpace(endomesh, "CG", 1)

    p_func = dlf.Function(Vp, name="p_inner")

    fname = os.path.join(dirname, "TEST", "ECCENTRIC",
                         "TEST-presure-wave.pvd")
    f = dlf.File(fname)

    tspan = np.linspace(0, 3.0, 401)
    p_expr = create_pressure_expression(p0, zmin, zmax)
    for t in tspan:
        print("Working on t = {0}".format(t))
        p_expr.t = t
        p_func.interpolate(p_expr)
        f << (p_func, t)
    fm.fmio.fmio.rewrite_vtk_files(fname)
```