UNIVERSITY OF CAPE TOWN

Proper Orthogonal Decomposition with Interpolation-based Real-time Modelling of the Heart

Author:
Ritesh Rao RAMA

Supervisors:
Dr. Sebastian SKATULLA
Prof. Daya REDDY

Thesis presented for the degree of
Doctor of Philosophy
in the department of Civil Engineering,
University of Cape Town

August 2017

Centre for Research in Computational
and Applied Mathematics
&
Computational Continuum Mechanics Research Group

Plagiarism form

I, **Ritesh Rao Rama**, hereby declare that the work on which this thesis is based is my original work (except where acknowledgements indicate otherwise) and that neither the whole work nor any part of it has been, is being, or is to be submitted for another degree in this or any other university. I authorise the University to reproduce for the purpose of research either the whole or any portion of the contents in any manner whatsoever.

Signature: Ritesh Rao Rama Date: August 2017

Abstract

Several studies have been carried out recently with the aim of achieving cardiac modelling of the whole heart for a full heartbeat. However, within the context of the Galerkin method, those simulations require high computational demand, ranging from 16 – 200 CPUs, and long calculation time, lasting from 1 h – 50 h. To solve this problem, this research proposes to make use of a Reduced Order Method (ROM) called the Proper Orthogonal Decomposition with Interpolation method (PODI) [1] to achieve real-time modelling with an adequate level of solution accuracy. The idea behind this method is to first construct a database of pre-computed full-scale solutions using the Element-free Galerkin method (EFG) and then project a selected subset of these solutions to a low dimensional space. Using the Moving Least Square method (MLS), an interpolation is carried out for the problem-at-hand, before the resulting coefficients are projected back to the original high dimensional solution space. The aim of this project is to tackle real-time modelling of a patient-specific heart for a full heartbeat in different stages, namely: modelling (i) the diastolic filling with variations of material properties, (ii) the isovolumetric contraction (IVC), ejection and isovolumetric relation (IVR) with arbitrary time evolutions, and (iii) variations in heart anatomy.

For the diastolic filling, computations are carried out on a bi-ventricle model (BV) to investigate the performance and accuracy for varying the material parameters. The PODI calculations of the LV are completed within 14 s on a normal desktop machine with a relative L_2 error norm of 6×10^{-3}. These calculations are about 2050 times faster than EFG, with each displacement step generated at a calculation frequency of 1074 Hz. An error sensitivity analysis is consequently carried out to find the most sensitive parameter and optimum dataset to be selected for the PODI calculation.

In the second phase of the research, a so-called "*time standardisation scheme*" is adopted to model a full heartbeat cycle. This is due to the simulation of the IVC, ejection, and IVR phases being carried out using a displacement-driven calculation method [2] which does not use uniform simulation steps across datasets. Generated results are accurate, with the PODI calculations being 2200 faster than EFG.

The PODI method is, in the third phase of this work, extended to deal with arbitrary heart meshes by developing a method called "*Degrees of freedom standardisation*" (DOFS). DOFS consists of using a template mesh over which all dataset result fields are projected. Once the result fields are standardised, they are consequently used for the PODI calculation, before the PODI solution is projected back to the mesh of the problem-at-hand. The first template mesh to be considered is a cube mesh. However, it is found to produce results with high errors and non-physical behaviour. The second template mesh used is a heart template. In this case, a preprocessing step is required where a non-rigid transformation based on the coherent point drift method is used to transform all dataset hearts onto the heart template. The heart template approach generated a PODI solution of higher accuracy at a relatively low computational time.

Following these encouraging results, a final investigation is carried out where the PODI method is coupled with a computationally expensive gradient-based optimisation method called the Levenberg-Marquardt (PODI-LVM) method. It is then compared against the full-scale simulation one where the EFG is used with the Levenberg-Marquardt method (EFG-LVM). In this case, the PODI-LVM simulations are 1025 times faster than the EFG-LVM, while its error is less than 1 %. It is also observed that since the PODI database is built using EFG simulations, the PODI-LVM behaves similarly to the EFG-LVM one.

Acknowledgements

First and foremost, I would like to give my heartfelt thanks to my supervisor, Dr Sebastian Skatulla. His guidance and continuous support have helped me in my research. Moreover, he has always made himself available when I needed him most and has been kind enough to patiently attend to all my queries. Since I joined the University of Cape Town, it has been an immense pleasure working with him.

In 2011, Prof Daya Reddy generously offered me a place in CERECAM. The office space became a safe haven where I could spend countless hours on my studies. There, I was welcomed by a warm and friendly academic community, who shared my research and social interests and I also met Olivia Goodhind - an amazing person who looked after me when Cape Town was very new.

Additionally, I want to express my gratitude to everyone who has helped me directly or indirectly in completing this project, and for their encouragement and advice. Among them, I would like to give a special mention to:

- Roshan Bhurtha: for all our programming-related discussions and your friendship over the years.
- Andie de Villiers: for all the jokes we shared in the CERECAM tea room and our nice camping trip!
- Winston Guess and Paul Taylor: for our endless football chats and fun football games we've played together.
- Alta du Plooy: for showing me the beautiful side of South Africa.

And finally, my family's endless supply of moral support and love have been instrumental during the most difficult times of my studies. For this, I will always be grateful.

This research has been supported by the Centre for High Performance Computing South Africa and the National Research Foundation of South Africa (Grant Numbers 90528, 93111, 104839 and 105858). Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the NRF.

Contents

List of Figures

List of Tables

Nomenclature

$\bar{\mathbf{U}}_i$	zero-mean ensemble matrix for a particular time step, i., see equation (5.62), page 67
β	transverse/sheet angle, see equation (3.43), page 33
$\boldsymbol{\gamma}_M$	right singular vector, see equation (5.19), page 52
$\boldsymbol{\nu}$	normal vector in the reference configuration, see equation (3.8), page 23
$\boldsymbol{\phi}$	a particular Proper Orthogonal Mode, see equation (5.1), page 49
$\boldsymbol{\phi}_M$	left singular vector or Proper orthogonal modes, see equation (5.19), page 52
$\boldsymbol{\Psi}$	left singular matrix, see equation (5.15), page 51
$\boldsymbol{\Sigma}$	Singular values diagonal matrix, see equation (5.15), page 51
$\boldsymbol{\sigma}$	Cauchy stress tensor, see equation (3.13), page 24
$\tilde{\boldsymbol{\Phi}}$	Reduced Proper Orthogonal Modes matrix, see equation (5.41), page 58
$\tilde{\boldsymbol{f}}$	rotated fibre direction vector, see equation (3.43), page 33
$\tilde{\boldsymbol{n}}_s$	rotated sheet-normal direction vector, see equation (3.43), page 33
$\tilde{\boldsymbol{s}}$	rotated sheet direction vector, see equation (3.43), page 33
$\boldsymbol{\vartheta}$	right singular matrix, see equation (5.15), page 51
$\delta\mathcal{W}_{\text{int}}$	internal virtual work, see equation (3.24), page 26
$\delta\mathcal{W}_{\text{ext}}$	external virtual work, see equation (3.24), page 26
$\emptyset$	Empty set, see equation (3.18), page 25
λ	a particular Proper Orthogonal Value, see equation (5.5), page 49
λ^{CPD}	constant regulating the contribution of the CPD regularisation term, see equation (5.68), page 76
λ^{LVM}	Lagrange multiplier constant in LVM to transition from the steepest descent method to the Newton method, see equation (7.2), page 138
$\tilde{\mathbf{Z}}$	coefficients determined using reduced Proper Orthogonal Modes matrix, $\tilde{\boldsymbol{\Phi}}$, see equation (5.41), page 58
$\mathbf{b}_{\mathcal{B}}$	Body force acting on body, $\mathcal{B}$, see equation (3.11), page 24
$\mathbf{C}$	Damping matrix of the FEM equation of motion, see equation (5.39), page 58
$\mathbf{C}$	right Cauchy-Green tensor, see equation (3.9), page 23
$\mathbf{E}$	Green strain tensor, see equation (3.10), page 24
$\mathbf{K}$	stiffness matrix, see equation (4.3), page 41
$\mathbf{M}$	Mass matrix of the FEM equation of motion, see equation (5.39), page 58
$\mathbf{N}$	shape functions or interpolants, see equation (4.2), page 41
$\mathbf{n}$	normal vector in the current configuration, see equation (3.8), page 23
$\mathbf{t}^{(\nu)}$	Traction force acting on body, $\mathcal{B}$, see equation (3.11), page 24
$\mathbf{u}$	displacement field vector, see equation (3.3), page 23

$\mathbf{X}$	point in the reference/undeformed configuration, see equation (3.1), page 23
$\mathbf{x}$	point in the current/deformed configuration, see equation (3.1), page 23
$\mathcal{B}$	body in the reference/undeformed configuration, see equation (3.0), page 22
$\mathcal{B}_t$	body in the current/deformed configuration, see equation (3.0), page 22
ω	function of time to calculate C_t, see equation (3.47), page 35
ϕ^{wall}	Orientation index for defining surface fibre, see equation (3.35), page 32
$\mathbf{a}_j$	eigen vector of covariance matrix, see equation (5.12), page 50
$\mathbf{c}$	Vector defining the circumferential, c-axis, see equation (3.35), page 32
$\mathbf{D}$	Scaling or damping matrix in LVM, see equation (7.4), page 139
$\mathbf{f}$	fibre direction vector, see equation (3.39), page 32
$\mathbf{f}_{\text{ext}}$	external force vector, see equation (4.3), page 41
$\mathbf{f}_{\text{int}}$	internal force vector, see equation (4.3), page 41
$\mathbf{L}$	autocorrelation matrix, see equation (5.7), page 49
$\mathbf{M}_i$	Structural tensor constructed from fibre directions, see equation (3.33), page 27
$\mathbf{n}_s$	Sheet-normal direction vector, see equation (3.40), page 32
$\mathbf{r}$	Vector defining the radial, r-axis, see equation (3.35), page 32
$\mathbf{r}_{\text{cz}}$	vector perpendicular to P_{cz}, see equation (3.36), page 32
$\mathbf{R}_{\text{int}}$	Residual vector for solving discrete equation system, see equation (4.6), page 41
$\mathbf{s}$	Sheet direction vector, see equation (3.37), page 32
$\mathbf{S}^{\text{Active}}$	Active stress, see equation (3.44), page 34
$\mathbf{S}^{\text{Passive}}$	Passive stress, see equation (3.44), page 34
$\mathbf{S}^{\text{Total}}$	Total stress at any point in the myocardium, see equation (3.44), page 34
$\mathbf{U}_i$	ensemble matrix for a particular time step, i., see equation (5.62), page 67
$\mathbf{Y}$	covariance matrix, see equation (5.12), page 50
$\mathbf{z}$	Vector defining the longitudinal, z-axis, see equation (3.35), page 32
E_{ff}	strain along the fibre direction, see equation (3.51), page 35
T^{active}	active tension force, see equation (3.45), page 34
θ	fibre angle, see equation (3.38), page 32
$\tilde{\mathbf{p}}$	fibre projected vector, see equation (3.38), page 32
ϑ	singular values, see equation (5.17), page 51
ζ	Neumann loading factor use to adjust ventricular pressure, see equation (4.11), page 43
A	Stress scaling coefficient, see equation (3.31), page 27
A_{compr}	Compressibility coefficient, see equation (3.31), page 27
a_i	constants of $\Upsilon_{\text{passive}}$ with $i \in (1, 2, \ldots, 6)$, see equation (3.32), page 27
B	constant that is characterised from an active tension - sarcomere length relationship, see equation (3.50), page 35
b	y-intercept of equation to compute t_r, see equation (3.49), page 35
C	elastic blood vessels compliance, see equation (3.53), page 37
C_t	variable that defines how the active tension should change with time, see equation (3.46), page 35
Ca_0	peak intracellular calcium ion concentration, see equation (3.46), page 35
dA	unit area in the reference configuration, see equation (3.8), page 23
da	unit area in the current configuration, see equation (3.8), page 23

dV unit volume in the reference configuration, see equation (3.8), page 23

dv unit volume in the current configuration, see equation (3.8), page 23

E_{fraction} End-diastole volume, see equation (2.2), page 21

E_{PV} Elastance of the pressure-volume loop, see equation (2.1), page 20

ECa_{50} extracellular calcium ion concentration when 50 % of peak active contraction force is achieved, see equation (3.46), page 35

J Jacobian of the deformation gradient, see equation (3.2), page 23

J_{KLD} Jacobian of the Karhunen-Loève decomposition problem, see equation (5.5), page 49

L inductance, see equation (3.58), page 39

$l^{\text{sarcomere}}$ sarcomere length, see equation (3.48), page 35

$l_0^{\text{sarcomere}}$ sarcomere length at rest state, see equation (3.50), page 35

$l_{\text{R}}^{\text{sarcomere}}$ resting sarcomere subjected to zero stress, see equation (3.51), page 35

L_2 L_2-error norm, see equation (5.45), page 60

m gradient of equation to compute t_r, see equation (3.49), page 35

P Ventricular pressure, see equation (2.1), page 20

P_o^{aorta} initial aorta pressure, see equation (3.56), page 38

P^{ED} Ventricular pressure at end diastole, see equation (4.11), page 43

P_{cr} Plane defined by the c and r-axis, see equation (3.34), page 31

P_{cz} Plane defined by the c and z-axis, see equation (3.34), page 31

P_{zr} Plane defined by the z and r-axis, see equation (3.34), page 31

Q Function defined by constants, a_{ij} and the Green-strain components, $\hat{E}^{ij}$, see equation (3.32), page 27

R_a resistance due to the aortic valve, see equation (3.57), page 38

R_p resistance of the vein - Windkessel, see equation (3.52), page 37

t specific point in time, see equation (3.0), page 22

t_0 time at which maximum tension is reached, see equation (3.48), page 35

t_r time taken for the tension force to dissipate and reach zero, see equation (3.48), page 35

T_{max} isometric tension force under maximal activation, see equation (3.46), page 35

V Ventricular volume, see equation (2.1), page 20

V_{ED} End-diastole volume, see equation (2.2), page 21

V_{stroke} Stroke volume, see equation (2.2), page 21

$V_{P=0}$ Ventricular volume when pressure is zero, see equation (2.1), page 20

W_{compr} Near incompressibility part of the Passive strain energy function, see equation (3.31), page 27

$W_{\text{exp-ortho}}$ Exponential and orthotropic part of the Passive strain energy function, see equation (3.31), page 27

W_{passive} Passive strain energy function, see equation (3.31), page 27

$\mathbf{F}$ deformation gradient, see equation (3.2), page 23

I Blood flow rate, see equation (3.55), page 38

Part I

Introduction

Chapter 1

Introduction

1.1 Thesis introduction

Computational cardiac mechanics is emerging as a rapidly expanding area of research that brings together multidisciplinary research centred on understanding the physiological and mechanical behaviour of the heart at scales ranging from cell to tissue and organ levels. Principles of continuum mechanics are key in creating a realistic multi-scale model of the heart. They allow one to describe the directly observable behaviour of the heart by incorporating its complex heterogeneous and anisotropic structure as well as the coupling mechanisms between mechanical fields on the one hand, and chemical and electrical fields on the other. Computational models therefore help to quantify the bio-mechanical environment in healthy, injured, and diseased hearts which, in turn, leads to advances in diagnostic and therapeutic procedures.

In cardiac modelling, many efforts have been made towards simulating a full heartbeat in order to evaluate the heart's performance. In the past 30 years, the field of cardiac modelling has undergone major developments that have made it possible to become close to a full working mathematical model that can mimic the heart's behaviour in a realistic way that is needed to obtain medical performance indicators [19]. This has been achieved by employing complex non-linear partial and ordinary differential equations, which are solved using staggered iterative schemes. This work is based on previous cardiac mathematical models [20] implemented within the Element-free Galerkin (EFG) [21] framework and solved using a Newton method, which is either pressure- or displacement-driven [2]. However, it has been found that within the context of the Galerkin method, solving of the discrete system of equations is one of the most computationally expensive processes [22]. Observations based on these cardiac simulations have identified that one of the biggest problems encountered is the high demand in computational time and resources, due to EFG itself.

Across the literature, several researchers have encountered the same issue during cardiac modelling, particularly when using 3D and realistic heart geometries. Those computational demands range from the usage of 16 – 200 CPUs, while the simulation times range from 1 h – 50 h [19, 23, 24, 25, 26, 27]. Hence, it can be deduced that cardiac modelling is very complex and requires much effort in terms of computational resources and time. Due to these issues, practical applications such as predicting the behaviour and performance of a specific heart under different physiological conditions concerning hemodynamical loads, progressing disease, therapeutic measures or medication, etc. have been limited in

the medical field. Moreover, cardiac modelling cannot be used as a training tool to help medical doctors since, as stated by Taylor et al. [28], interactive surgical tools require fast solution speed in order to have a realistic feedback from the model which will mimic surgical operations. These devices operate at a frequency of 300 – 1000 Hz [28, 29, 30], constraining modelling implementations to solve for discrete systems of equations under a very limited time frame. Current advances in computational power do not yet allow such a speed-up for the EFG computations. Hence, in literature, many researchers have tried alternative ways. One of the popular approaches consists of the use of a mass-spring model [31, 32]. According to Meier et al. [32], a spring-mass model defines a geometrical mesh in terms of discrete mass points which are interconnected by springs. Using the Newtonian law of motion and a time discretisation scheme, the deformation of the mesh can be solved for when subjected to external forces. This method is considered simple and computationally efficient [33]. Consequently, its application has been broad. For example, Nedel and Thalmann [34] simulated the brachialis muscle of the upper arm. Liu et al. [35] considered a non-biological example where a hanging cloth was the problem to be solved. Using the mass-spring approach, they achieved a calculation frequency range varying from 0.2 Hz to 185 Hz, with the frequency level being high for large errors. In [36], Luboz et al. modelled the deployment of a stent in the femoral artery using an inflated balloon. Their simulations were carried out at a frequency range of 26 Hz to 53 Hz. Even though high computational speed is achieved using the mass-spring method, Nealen et al. [33] reported that the models had low accuracy, and according to [32], unrealistic behaviour, such as delays in deformation propagation and un-natural oscillations were observed. Another common approach for more realistic real-time simulation is through the use of the linear Finite Element method (FEM) [31, 37, 38, 39, 40, 41]. The latter does not require an update of the stiffness matrix to solve for the mechanical fields such as displacement, stress and strain. Some examples of its application in the literature are as follows: Cotin et al. [31] adopted a modified FEM elastic model which is solved using static equations. In order to account for the non-linearity of their liver model, an "enhanced linear model" was adopted. Berkley et al. [38] also employed a linear model. In their case, they deemed the dynamic effects of biological tissue to be unnecessary for their suturing procedure. As such, a calculation frequency of 30 Hz was achieved. In another example, Lim and De [39] tried an approach of coupling a non-linear, and linear material model for their surgery tool-tip penetration of a kidney. Their idea consisted of defining the surrounding region of a kidney, where contacts are made between the surgery tool tip and the organ, to be non-linear, while the rest of the organ is defined as linear. Additionally, for the solving procedure, they made use of a mesh-free technique called the *Point collocation-based method of finite sphere*. Hence, they managed to achieve a calculation frequency of 1000 Hz. Finally, Courtecuisse et al. [40] tried to use Graphical Processing Units (GPU) for their linear finite element calculation to model the cutting of a liver. With their implementation, a simulation frequency of 29 Hz was achieved. Even though linear elastic models provide a fast computation time, their solution accuracy of large deformation is usually poor with, for example, the error being higher than 30 % in the work of Lim and De [39].

In this project, an alternative approach is considered, so that real-time simulation of the heart is achieved within reasonable range of accuracy. Consequently, this work consists of studying its performance and applying it to another computationally expensive problem: parameter optimisation. The proposed solution is to make use of the Proper Orthogonal Decomposition with interpolation [1], also known as PODI. The latter is employed as a reduced order technique which captures the behaviour of a pumping

heart and creates a low dimensional space where interpolation of different sets of data will take place. This consequently lowers the computational and time resources requirement to run cardiac mechanic models. PODI is considered to be part of the Reduced Order Method (ROM) family and is an extension of the POD method. Generally, POD provides the means to extract information from a predefined set of data obtained from a set of experiments or computer simulations using statistical methods. In particular, it has the ability to capture important details or trends, and represents them in terms of a set of numerical vectors known by several names such as *POD basis, Proper Orthogonal Modes (POMs), empirical eigenfunctions or empirical orthogonal functions*. The POD method has gained great popularity due to its simplicity, flexibility, ease of implementation, ability to reduce the order of a system of equations, and its application to a broad range of field problems such as image processing [42], heat flow problems [43], computational fluid dynamics (CFD) [44, 45, 46] and solid mechanics [47, 48, 49, 50, 51, 52, 53, 54].

As mentioned above, this research utilises the PODI method, which is a particular variant of POD. It was developed by Ly and Tran [1] and involves a collection of datasets of the structure under consideration, describing its mechanics for a range of variations in terms of geometry, material properties, loading conditions, etc. These datasets are transferred to a low-dimensional space via their associated POMs, where the unknown or not yet determined mechanical response of a structure of the same category is interpolated using the Moving Least Square Approximation (MLS). Note that this stands in contrast to methods which involve direct interpolation of Proper Orthogonal Modes [30] or interpolation of reduced order models in the tangent space [55]. In our case, the collection of datasets comprises full-scale simulation results of the human heart obtained using EFG for different cardiac tissue parameters (such as stiffness, fibre orientation, etc). Once these results have been obtained, they were stored off-line in a suitable database format for ease of access. Recent research carried out by Coelho et al., Ko et al. and Niroomandi et al. has found that such an approach facilitates sub-second calculation times, therefore making high frequency computation feasible [29, 56, 57].

Given the benefits of PODI to accelerate simulations, a numerical feasibility study of PODI applied to cardiac mechanics is carried out with the following objectives:

1.1.1 Real-time modelling of the heart to study its performance

The Literature showing the application of the PODI method with bio-mechanics problems is very limited. Only one paper [29] has been encountered so far, where the main purpose was to develop real-time virtual reality software to help clinicians in surgical training with haptic devices. In [29], it was observed that PODI is applicable to the palpation of a cornea. However, in this research, PODI was subjected to a more complex problem: that of a heart undergoing several phases of expansion, contraction, elongation and torsion during a heartbeat. Accordingly, the problem is highly non-linear, not only due to the compliant nature of cardiac tissue, but also due to the time-evolution of active contraction and haemodynamics. There are also other ways in which this research differs from the presented literature. Firstly, a multi-dimensional parameter domain was considered. It comprised parameters from the cardiac constitutive equations, which were varied to create a series of datasets in the form of a database. Secondly, due to the number of parameters involved in this research, a piecewise linear interpolation scheme was deemed inadequate. As such, the Moving Least Square method was favoured. MLS has the ability to scale up smoothly to several dimensions, is generally very fast and can consider different numbers of data points when multi-parametric simulations are carried out, while still generating accurate results. Studies,

such as [56, 57], have already coupled PODI with MLS, and are aimed at capturing localised phenomena in their respective solutions.

As indicated earlier, the time-evolution of active contraction and haemodynamics are highly non-linear. Consequently, simulation of the same cardiac problem with different material properties leads to a different timeline of a full heartbeat due to those simulation models depending on strain [2, 17]. As such, a procedure called the *time standardisation scheme* was introduced by the current work. The latter helps in accurately predicting the start and end points of each heart phase, which are the diastole filling, isovolumetric contraction, ejection and isovolumetric relaxation of the problem-at-hand. Additionally, it ensured that the steps of the datasets are synchronised such that each PODI calculation restricted the usage of simulation result fields belonging to the same phase across selected datasets. Thus, the time standardisation procedure maximised the PODI solution accuracy and also ensured that the timeline of the problem-at-hand is correctly predicted.

Even though the computational speed of cardiac modelling has the potential of being drastically improved, it still not ready for realistic application. A problem that may arise concerns variations in anatomy and the physiological properties across the hearts. Broadly, healthy hearts have in common the same general features such as two ventricles, two atria, ventricular and aortic valves and electro-physiological components, like the atrioventricular node, and the bundle of His and Purkinje fibres. Hence, any of these hearts will therefore go through the four phases of a heartbeat, that is, diastole filling, isovolumetric contraction, ejection and isovolumetric relaxation. The phases which constitute a heartbeat consequently lead the heart to expand, contract and twist on itself. Therefore, different healthy hearts have globally similar behaviours, which can be extracted using the Proper Orthogonal Decomposition (POD) as Proper Orthogonal Modes (POMs). However, those healthy hearts are geometrically unique with respect to each other. For these geometrical variations, every chamber of the heart comes in different sizes resulting in a range of values as shown in [18, 58, 59, 60, 61, 62, 63, 64, 65, 66]. Within the PODI framework proposed here, these differences are more difficult to integrate. This problem relies on the way the POMs are extracted. In PODI, a collection of solution fields, originating from simulations carried out from different material parameters, are assembled in a matrix. In order to build the matrix, the number of degrees of freedom of all the solution fields must be the same and also must have the same spatial location. Yet, hearts extracted from Magnetic Resonance Imaging (MRI) images, have geometries that are close to, if not, unique. Hence, the discretisation of these geometries with a mesh for the EFG simulations leads to the total number of nodes not being the same and additionally, not sharing the same spatial coordinates across the hearts. Consequently, assembling their solution fields into a matrix leads to incompatible data and as such, the extraction of POMs cannot proceed. Most implementations across the literature have been found to be aimed at a fixed mesh configuration, a trend also noted recently by Iuliano and Quagliarella [67], Amsallem et al. [68], González et al. [69], who also attempted to find a solution. However, their techniques were found to be complex and uncertain to achieve an acceptable level of accuracy. Therefore, for this research, an alternative approach, was proposed and is referred to as the *degrees of freedom standardisation method* (DOFS). The method consists of standardising the solution field of the datasets over a set of template mesh. Once the PODI has been run on the standardised datasets, the PODI solution is then projected on to the geometry of the problem-at-hand. As template mesh, two different geometries are considered. The first is a cube grid, while the second is a heart. In the second scenario, a non-rigid registration procedure should be carried out so that the dataset hearts are

mapped onto the template heart due to their geometrical differences. The non-rigid registration technique used in this research is based on the Coherent Point Drift (CPD) method introduced by Myronenko et al. [70]. Its application to cardiac modelling is demonstrated in [71], where two left ventricular point sets were successfully registered. The advantage with this method is that the two 3D point sets could possibly not share the same number of nodes, or have missing points, while still obtaining a good registration. The reason why this is achievable is due to the CPD employing a so-called Gaussian mixture model (GMM) and the Expected maximisation (EM) coupled with a regularisation term [71].

1.1.2 Real-time modelling of the heart for the purpose of parameter optimisation

Another usage of the PODI method within the cardiac mechanics framework was also investigated. ROM calculations can be used to speed up computationally expensive methods that require simulation results as inputs [72, 73]. One of these computationally expensive methods is the inverse parameter optimisation method. Within the context of cardiac modelling, parameter optimisation is useful when calibrating the constants of the cardiac constitutive equations such that the model behaves closely to observations made from experimental results [73, 74, 75, 76, 77]. One optimisation method that is known to suffer from expensive calculation time, is a gradient-based parameter optimisation technique called *Levenberg-Marquardt* (LVM) [78]. The latter is classified as a non-linear least square and requires the evaluation of $n + 1$ simulations at each iteration if n parameters are to be calibrated. As such, its computational need on resources and time is heavy when multiple parameters or iterations are required. With PODI, the aim of the research was to reduce the time and computational expense of the Levenberg-Marquardt algorithm. The PODI-LVM coupling was similar to the surrogate model optimisation problems [79] except that in this case, the calculation took place within a low dimensional space.

1.1.3 Limitations

This thesis aims at developing a numerical framework suitable to carry out a proper investigation on real-time modelling of the heart using the Proper orthogonal decomposition with interpolation. Yet, some limitations still exists. The first one was the usage of idealised geometries. MRI images of the heart were not available to create a real heart geometry. As such, idealised ones extracted from the literature were used. These geometries disregard the aorta and valves and consist only of the left and right ventricles. Moreover, this research focused on the solid mechanics of the heart. It did not consider the fluid mechanics relating to the blood movement across the heart's ventricles and its fluid-structure interaction.

Regarding the PODI calculation itself, this work has has been found to be limited with respect to the two following aspects. The first one concerns the linear properties of PODI, which originates from POD. An important aspect of the POD method is that it is considered a linear procedure, with the POMs corresponding to linear spaces [80]. Its application to non-linear problems has not been studied rigorously within a mathematical and PODI framework. This research does not intend to do so, as it falls outside the scope of the project's aim. However, it is believed that the numerical application presented here can still contribute in providing an insight on the suitability of POD, along with PODI, to highly non-linear problems. The second limited aspect of this research within the context of ROM is

the definition of calculation time. The calculation time is only of importance in this work to measure the speed of calculations. However, they do not account for the post-processing phase, which includes visualisation and rendering. These processes are deemed dependent on the hardware and software visualisation/rendering tools employed and are not part of the objectives of this study to be optimised to achieve real-time modelling. On that line, it should be stressed out that in this research, the aim is to reduce those calculation times through a ROM mathematical approach only, and not by influencing the hardware performances such as utilising faster processors or code parallelisation.

1.2 Publication and Presentations

Some of the work described in this thesis has been published and presented.

Publications:

- R.R. Rama, S. Skatulla, and C. Sansour. Towards real-time modelling of the heart using the proper orthogonal decomposition with interpolation approach. *Insights and Innovations in Structural Engineering, Mechanics and Computation.* 624–628, August 2016, doi:10.1201/9781315641645-103.

- Ritesh R. Rama, Sebastian Skatulla, and Carlo Sansour. Real-time modelling of diastolic filling of the heart using the Proper Orthogonal Decomposition with Interpolation. *International Journal of Solids and Structures*, 96(1):409–422, October 2016, doi:10.1016/j.ijsolstr.2016.04.003.

Presentations:

- Ritesh R. Rama and Sebastian Skatulla. Real time modeling of the heart *at the Commonwealth Science Conference*, Bangalore, India, 2014.

- Ritesh R. Rama, Sebastian Skatulla, and Carlo Sansour. Using the Proper Orthogonal Decomposition with Interpolation for Real-time modelling of the Heart *at the* 9^{th} *European Solid Mechanics Conference*, Madrid, Spain, 2015.

- Ritesh R. Rama, Sebastian Skatulla, and Carlo Sansour. A Proper Orthogonal Decomposition with Interpolation-based Real-time modelling of the Heart *at the VI International Conference on Computational Bioengineering*, Barcelona, Spain, 2015.

- Ritesh R. Rama, Sebastian Skatulla, and Carlo Sansour. Real-time Modelling of the Heart using a Proper Orthogonal Decomposition with Interpolation approach *at the* 6^{th} *International Conference on Structural Engineering, Mechanics and Computation*, Cape Town, South Africa, 2016.

1.3 Thesis Overview

The research carried out in this thesis was written in six parts, which are presented as follows:

Part I gives an overview of this work. The chapters of Part II deal with the literature of heart modelling and the Reduced Order Method. Chapter 2 presents the physiology of the heart and broadly touches on its function. An in-depth analysis of the heart's cellular and chemical composition is carried out, and the

heart's mechanical behaviour during a full heartbeat is elaborated upon. Chapter 3 consequently provides the different existing mathematical models that are used to reproduce these mechanical behaviours. It firstly introduces Continuum mechanics, a framework used to develop those models. A mathematical description of the heart fibres is given prior to the four models representing each phase of a heartbeat being stated. Chapter 4 presents the solving of those models within the context of the Element-free Galerkin method, which involves the variational principles. The computational non-linear schemes are given, and the different implemented mechanisms used to solve the previously introduced four mathematical models are explained. As such, by the end of the three chapters, the presented literature should provide a global picture of the full cardiac modelling framework used in this research.

Chapter 5 talks about the Reduced Order Method. More specifically, it gives the theory behind the Proper Orthogonal Decomposition and Proper Orthogonal Decomposition with Interpolation. The interpolation scheme used within the PODI method, which is the Moving Least Square Approximation (MLS) is depicted, before the implementation of PODI in this work is provided. The chapter also discusses the current main limitations of PODI and the proposed solution, which takes the form of the Degrees of freedom standardisation method. The latter considers two techniques, one involving a registration procedure, which employs the Coherent Point Drift method.

Chapter 6 of Part III discusses the application of some examples within the context of PODI. Section 6.1 includes a strut benchmark example, which provides an initial analysis of the PODI method. Section 6.2 introduces the cardiac example that was further used for the PODI calculation. Hence, the cardiac example's geometrical and material properties are discussed. In Section 6.3, the PODI calculations of the diastole filling of a bi-ventricle is carried out using a single parameteric and multiple parametric dimensional database. Section 6.4 then investigates the Degrees of freedom standardisation method through two examples, where one is a cubic template, the other is a heart template. The last section of this chapter, which is Section 6.5, discusses an attempt to perform a full heart cycle PODI calculation by making use of a time standardisation approach, involving temporal PODI calculations.

In Part IV, PODI is coupled with the LVM optimisation scheme. Consequently, Chapter 7 introduces the theory and implementation of the Levenberg-Marquardt algorithm. Chapter 8 looks at the examples of using PODI with LVM for cardiac problems. Two examples of different levels of complexity are considered. In the first example, the end-diastole volume of a left ventricle needs to be found using a one-dimensional parametric database. In the second example, the left ventricular end-diastole volume of a bi-ventricle model is looked at for using a multi-dimensional domain. For each example, the results obtained are then analysed.

One of the last parts of this work is Part V, which is split into three chapters. The first is Chapter 9, which gives a summary of all the results and analysis given in Part III and Part IV, while additionally providing a more coherent explanation across the different investigations carried out. Chapter 10 concludes this research by stating the objectives that were achieved, and some of the existing limitationd. Finally, Chapter 11 presents a few suggestions for future research that can be derived or expanded from this work.

The Appendix is the final part of this thesis and is given in Part VI. Appendix A briefly talks about the practical aspect of the work's EFG implementation. Appendix B discusses the Point-In-Polygon problem, which was found to be useful during the degrees of freedom standardisation procedure.

Part II

Theory of Cardiac modelling and Reduced Order Methods

Chapter 2

Cardiac Physiology

Cardiac Mechanics is the art of expressing the heart's physiological mechanism in terms of mathematical equations to realistically reproduce the organ's mechanical behaviour. In this chapter, the physiology of the human heart is introduced, based on the literature from [10, 16, 81].

2.1 The Heart

The heart is an organ located to the left of the mediastinum in the thoracic cavity of the chest. The heart wall is mostly made up of muscle fibres, known as the *myocardium*, which contracts periodically to allow receiving and pumping blood back into the circulatory system of the body. This blood flow allows the continuous feeding of minerals, nutrients and oxygen to different organs, while discharging generated toxic waste.

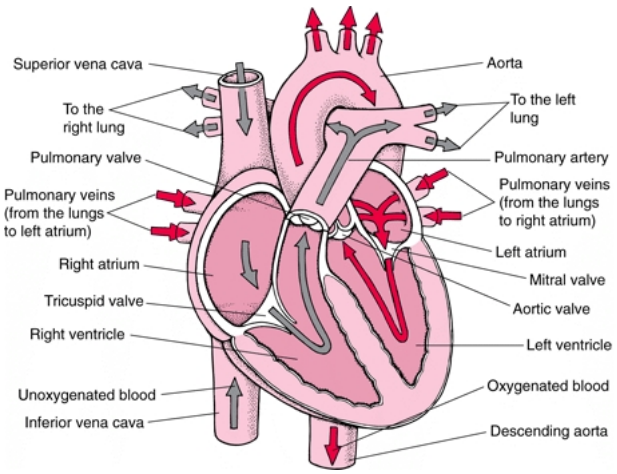


Figure 2.1: Heart cross-section [3].

To understand the pumping mechanism of the heart, its internal structure needs to be investigated. Fig. 2.1 shows that the heart is made up of four chambers. The smaller two chambers at the top half are called the left and right atria, while the larger, bottom two are referred to as the left and right ventricles.

Each atrium is connected to the ventricles and arteries with valves, which are present in order to regulate the inflow and outflow of blood to the chambers.

The blood circulation across the heart's chambers takes place in the following sequential manner: Based on Fig. 2.1, the de-oxygenated blood first travels from the body's venous system into the *superior vena cava* before reaching the right atrium. The right atrium then expels the fluid into the right ventricle as the *atrioventricular valve*, also known as the *tricuspid valve*, opens. At the end of this step, the tricuspid valve closes and the *aortic valves* open up, channelling blood to the left and right lung. Inside the lung, blood is re-oxygenated and transferred to the left atrium. With the *mitral valve* opened, blood is pushed into the left ventricle before being expelled back into the circulatory system through the aorta as the aortic valves opens.

For this research, the left and right ventricle were the primary focus, since they are the two main chambers of the heart and are surrounded by a larger portion of the myocardium, which dictates the overall behaviour of the heart. The left and right ventricles form the bottom half of the heart, as shown in Fig. 2.2.

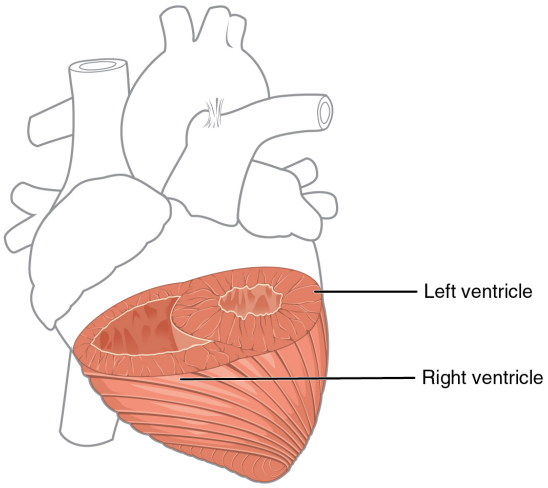


Figure 2.2: Bottom half of the heart [4].

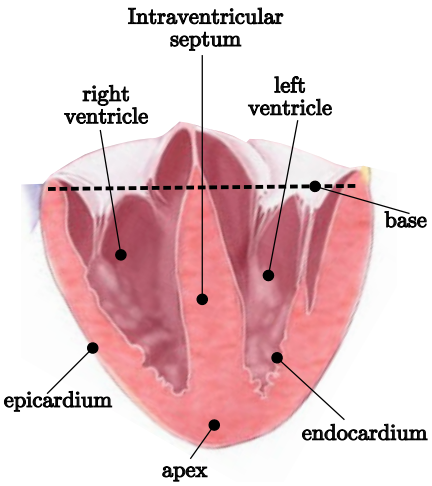


Figure 2.3: Cross-section of the right and left ventricle.

Fig. 2.3 shows a cross-sectional view of the left and right ventricle. The wall that separates the two ventricles is called the *intraventricular septum*, and is as thick as the left ventricular wall. In general, it has been observed from Magnetic Resonance Imaging (MRI) that the wall surrounding the left ventricle is considerably thicker than the right one. This is due to the left ventricle being responsible for developing enough pressure to ensure blood being ejected back into the circulatory system. The bottom tip of the heart is called the *apex*, while the surface along the dotted line represents the *base*. The base is usually defined by the interface wall between atria and ventricles, and houses the valves. Any surface corresponding to the outside layer of the myocardium is called the *epicardium*, while the inside layer is called the *endocardium*.

The heart's pumping activity is a process where the myocardium walls expand and contract. This process is usually referred as *heartbeats*. An isolated heartbeat consists of four primary phases that help the flow of blood. The phases are: *filling*, *isovolumetric contraction*, *ejection* and *isovolumetric relaxation*. During each of these phases, the heart's mechanical behaviour is different, leading to different

states of deformation as shown in Fig. 2.4. The different deformation shows that the heart undergoes what can broadly be classified as twisting, elongation, or expansion. As shown in [5], at each heart phase, these motions can happen simultaneously. During filling, the heart elongates and radially expands as blood applies pressure against the ventricular wall. In isovolumetric contraction, the heart muscle contracts through an anti-clockwise rotation, when looked at from the apex into the heart. This results in a shortening of the ventricle along the longitudinal direction. In the ejection phase, the twisting direction changes. At the bottom region of the ventricle, the apex rotates in an anti-clockwise direction, similar to the isovolumetric contraction phase. But at the base, which is the top region of the ventricle, the heart muscle rotates in a clockwise direction. Finally, during relaxation, the ventricles rotate clockwise to untwist, and elongate along the longitudinal direction in order to reach their original configuration for the next heartbeat.

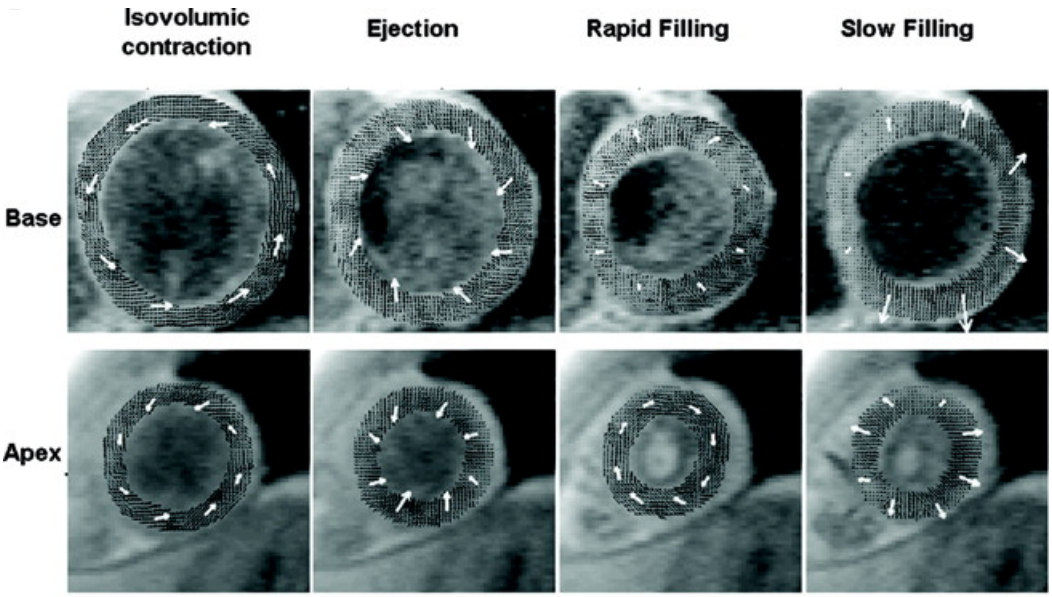


Figure 2.4: Cross-sectional deformation of the left ventricle during a heartbeat [5].

2.2 Myocardium

The myocardium is a bundle of muscle fibres. Their arrangement defines the different chambers of the heart. Understanding how the myocardium operates during a heartbeat can help in predicting the heart's behaviour. As such, the macro and micro-structure of the myocardium was investigated. The macro-structure looks at the muscle layout around the heart, while the micro-structure closely explores the internal structure of the cardiac muscle.

2.2.1 Macro-structure

The cardiac muscle is made up of different components that consist of muscle fibres, blood vessels and nerves. However, the presence of muscle fibres is dominant, as they occupy 70 % to 90 % of the total volume. The idea that the heart is made up of a single continuous helical band that coils on itself to

form the different heart chambers [82], as shown in Fig. 2.5, has been gaining ground within the cardiac community, and this can therefore explain the fibre distribution across the heart.

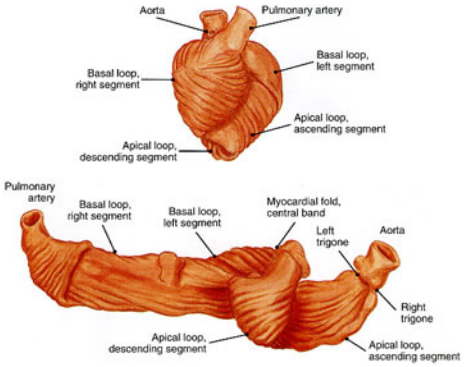


Figure 2.5: Unfolding of the heart into a band [6].

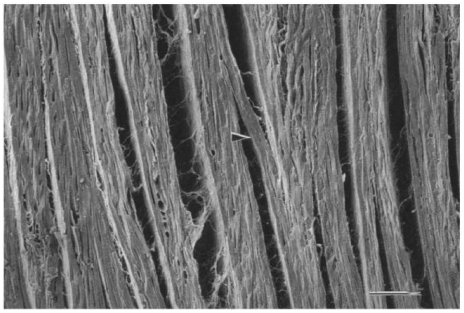


Figure 2.6: Sheet stack of fibres [7].

These fibres are usually stacked in discreet sheets forming cleavage planes. Cuttings of the myocardium, as shown in Fig. 2.6, reveal these sheets and their internal structure. Even though the cleavage planes are distinct from each other, it is common for branching to occur, leading to connectivity between the sheets. In between and around the sheets, several thin web-like filaments, known as *collagen fibres*, are present. They form a complex network, which also enhances the connectivity between the sheets. Due to the twisting of the myocardium band to define the ventricles, the orientation of fibres and sheets has been found to change across the heart and within the ventricular walls. In the literature, many authors such as LeGrice et al. [7] and Lunkenheimer et al. [8], have made such physiological observations by looking at slices of ventricular walls as shown in Fig. 2.7. Naked eye observations reveal the sheet orientation, while microscopic observations extract the fibre directions, which have been found to vary inside the sheets [14, 83].

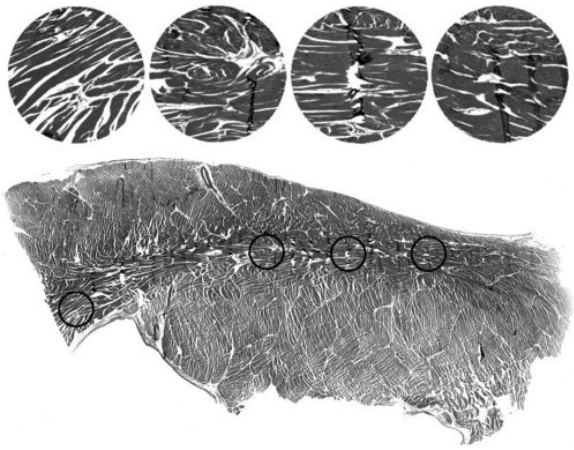


Figure 2.7: Slice of left ventricular wall [8].

Along the longitudinal direction of the ventricles, the sheets are stacked on top of each other and are parallel to the radial direction. However, they are not horizontal. They are concave in nature, with their

angle changing across the ventricular wall thickness; that is, from endocardium to epicardium, as shown in Fig. 2.8. Fig. 2.9 shows the sheet layout across a left ventricle. The epi-endo angle difference is more pronounced at the apex than at the base [9]. Transmurally (along the thickness of the ventricular wall from the epicardium to the endocardium), the fibre direction changes from a negative angle to a positive one, if the angle is measured from the circumferential plane of the ventricle [9, 84].

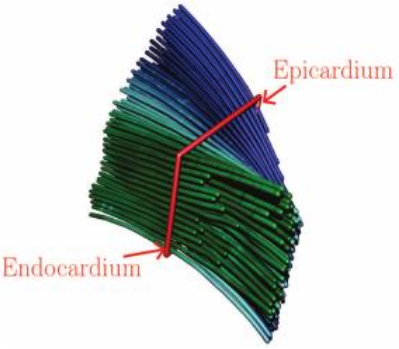


Figure 2.8: Fibre distribution from the endocardium to the epicardium surface [9].

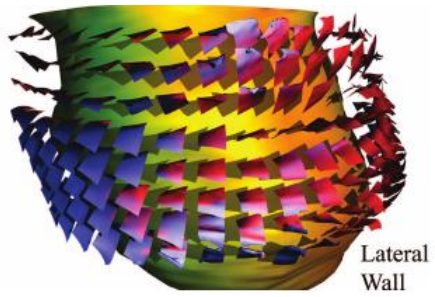


Figure 2.9: Sheet distribution across the left ventricle [9].

The fibre angle directions, as observed by Rohmer et al. [9] and Lombaert et al. [84], are of prime importance to understanding the twisting of the heart. As such, the work of Nakatani [85], which describes the fibre mechanics, is used to describe this relationship. Along the epicardium, the fibre angle is considered to be inclined downwards. That is, the fibre is running from the left top-end side of the ventricle to the right bottom-end side. At the endocardium, the fibres are oriented in the opposite direction, where they incline upwards. This upwardness is deduced from the fact that the fibres are aligned along a path going from the bottom-left corner to the top-right corner of the ventricle. In terms of measured quantities, the fibre angles are found to be within the range of $-15°$ to $-65°$ and $50°$ to $70°$ on the epicardium and endocardium respectively [9, 84].

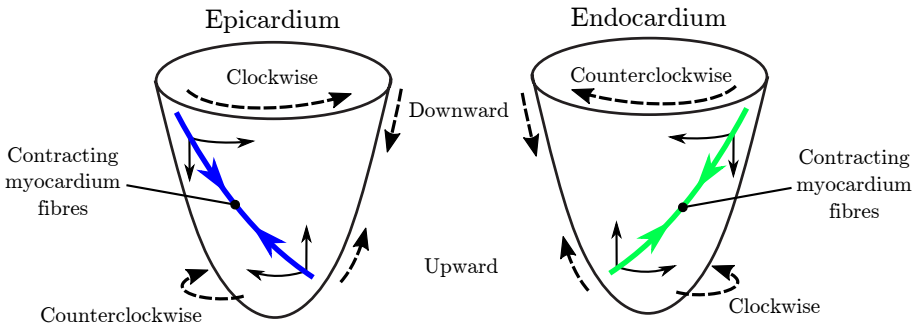


Figure 2.10: The mechanics caused by the fibre orientation on the epicardium and endocardium.

During isovolumetric contraction, the cardiac fibres contract along their direction with the centre of the net resultant contraction being at the middle of the fibre. Using this fact, and knowing the fibre orientation, the heart's movement can be globally predicted as shown in Fig. 2.10. On the epicardium, the downward-inclined fibre leads to the pulling of the top-left and bottom-right corners of the heart to the equatorial region of the ventricle. Due to the diagonal inclination of the fibres, a longitudinal and

circumferential movement is induced. At the top-left corner, the longitudinal movement is downward as the net resultant contraction of the fibre is located below. However, at the bottom-right corner, the longitudinal movement is upward, since the net resultant contraction is located above. Hence, during a heart contraction, the ventricles are expected to shorten.

For the circumferential rotation, the location of the net resultant contraction here is also important. Since it is located to the right of the top-left corner, the myocardium at the base tends to move from left to right, causing a clockwise rotation. At the bottom-right corner where the fibre ends, the net resultant contraction is on the left. Hence, the movement of the myocardium is right to left, translating in a counterclockwise rotation. Since the top-left and bottom-right corners are being pulled at the same time, the clockwise and anticlockwise rotation occurs at the same time. Such a model is found to agree with the isovolumetric phase observation given earlier in Fig. 2.4 from [5].

2.2.2 Micro-structure

The micro-structure of a heart is also an important aspect that provides insight into the behaviour of the heart fibres. Under a microscope, the heart's muscle fibres are found to be composed of *myocytes* and connective tissue. The latter is made up of a complex network of collagen fibres as introduced in Section 2.2.1, and exist in three forms: *endomysium*, *perimysium* and *epimysium*. The endomysium surrounds the myocytes and the perimysium encompasses a group of myocytes, while the epimysium lies on top of the entire muscle and are present at either the endocardium or epicardium [86]. On the other hand, the myocytes themselves are long and thin cells made up of vascular smooth tissues called *endothelial cells* and *fibroblasts*. The fibroblasts are helpful in maintaining the healthy state of the collagen, while the endothelial cells are a lining which promotes tissue growth and remodelling. The vascular smooth tissues are connected in a serial manner, with an *intercalated disc* at the connective interface. The purpose of the intercalated disc is to form a strong mechanical linkage between cells.

Even though there are several types of myocytes in the heart, Katz [16] states that there is only one category of myocytes that can contract. These are called *working myocytes*. The working myocytes are cross-striated in nature and contain a nucleus at their centre. In terms of composition, they are made up of myofibrils, mitochondria, sarcoplasmic reticulum, and other components, dominated mostly by cytosol. The arrangement of different parts of a myocyte is illustrated in Fig. 2.11.

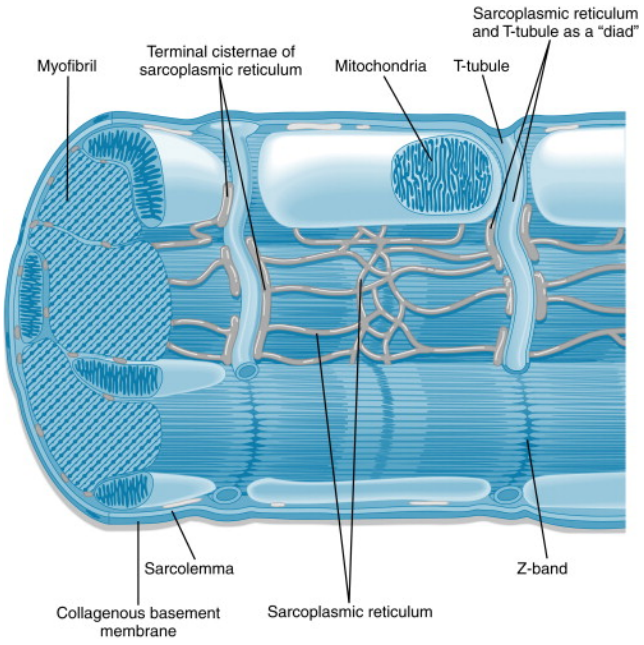


Figure 2.11: Structure of a myocyte [10].

The mitochondria occupy about 36 % of a myocyte and are responsible for the production of *adenosine triphosphate* (ATP), which participates in the contraction process. On the other hand, the sarcoplasmic reticulum, which mostly consists of a network of tubes, aims at transporting another contraction-dependent ion called *calcium*. The myofibril is also present in a myocyte and hosts the mechanism for contraction. It is commonly known as a "*unit of muscle*". Under a microscope, it is observed that the surface of myofibrils is composed of several subunits, delimited by zig-zag-like lines. These subunits are called sarcomeres and are responsible for inducing the myofibres to contract.

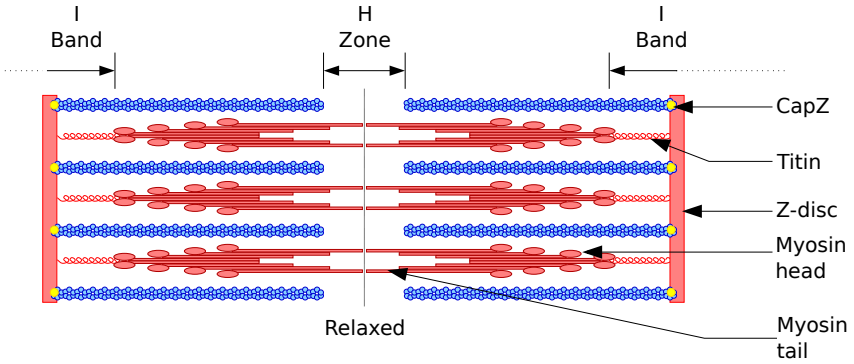


Figure 2.12: Sarcomere, elongated and passive.

The zig-zag-like lines are called *Z-line discs* and are shown in Fig. 2.12. Between those Z-line discs exist two bands: the I-band and the A-band [10]. The I-band is referred to as the isotropic band, and spans over two sarcomeres. The A-band, which corresponds to the anisotropic band, is always restricted within the sarcomere, i.e. between two z-line discs. The I-bands represent a thick filament called *Actin*, while the A-bands contain a thick filament known as *myosin*. The myosin is attached to a protein chain

known as *titin*, which is itself connected to the Z-line discs found on the left and right end sides of the sarcomere. For contraction to occur, myosin and actin need to interact. This occurs when the globular heads that form part of the myosin move along the actin filament. As the globular heads move, they pull along the actin filament towards the centre of the sarcomere, as show in Fig. 2.13. This behaviour leads to a shortening of the distance between the two Z-line discs and hence an overall shortening of the myofibrils, which translates to a contraction of the myofibres.

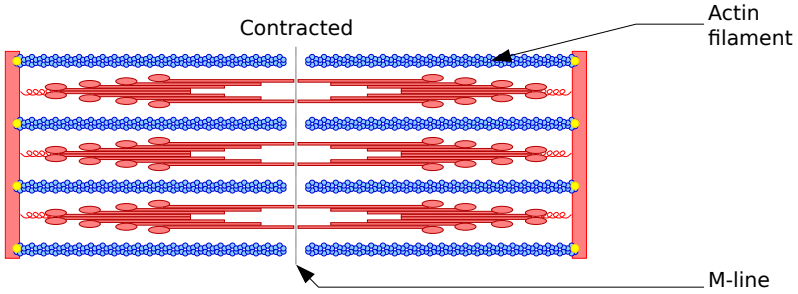


Figure 2.13: Sarcomere, active and contracted.

The force generated from the myocardium is dependent on the stretch of the sarcomere [16]. The force-stretch relationship usually follows the curve provided in Fig. 2.14. The latter can be broken down into three stages: A, B and C. In stage A, the sarcomere is only slightly pulled. The distance over which the actin filaments have been stretched away from the myosin's centreline is also minor. Hence during the active phase, a small contraction force is developed. In stage B, the stretch of the sarcomere leads the actin filaments to be pulled to such a point where the innermost myosin heads are close to falling out of contact. Any contraction occurring thereafter will induce a peak tensile force. In the last stage, the sarcomere is overstretched and some of the myosin heads are no longer in contact with the upper filament. Hence, fewer myosin heads are at work and a smaller force is recorded.

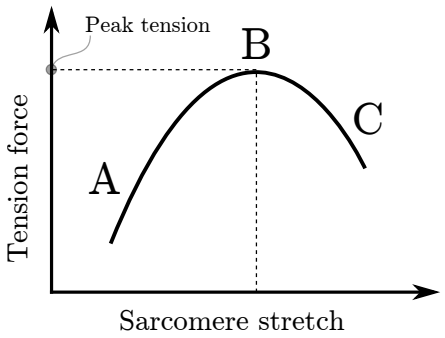


Figure 2.14: Change in tension as the sarcomere is stretched (based on plot from [11]).

The movement of the myosin head is not permanent. It occurs in a periodic way and as such, dictates the frequency of heart contractions. To understand the circumstance under which the myosin becomes active, the molecular level of the heart function was examined. More specifically, the electro-chemical changes occurring at a myocardium cellular level were investigated.

Following the release of an electric signal from the Purkinje fibres into the myocardium, the potential voltage across a heart cell is altered as shown in Fig. 2.15. These phases, labelled 1 to 4, mainly represent

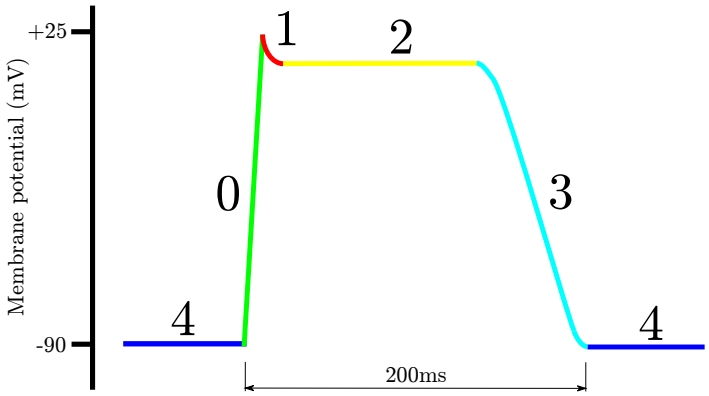


Figure 2.15: Action potential of fast-response cardiac fibres..

the permeability level of the cell. As the potential changes, several gates/channels along the cell's membrane are activated or deactivated to allow for the transfer of ions. According to Pappano and Wier [10], the ions that play an important role in the overall potential of the heart's cell are Potassium ions, K^+, Sodium ions, Na^+ and Calcium ions, Ca^{++}. The concentration of these ions usually varies depending on whether they are measured inside (intracellular ions) or outside (extracellular ions) of the myocardium cell. Additionally, the variation can be observed across the different phases of the action potential propagation as follows [10]:

- Phase 4: At resting phase, the net resultant potential of the cell is $-85\,\text{mV}$ to $-90\,\text{mV}$. The extracellular concentration of potassium ions, $[K^+]_e$ is less than the intracellular ones, $[K^+]_i$, while for Na^+ and Ca^{++}, their extracellular concentration is greater than the intracellular ones. During this phase, there is a slight efflux of K^+ from the cell.
- Phase 0: Depolarisation of the cell occurs due to the electrical propagation wave initiated at the end of the Purkinge fibres. A rapid influx of Na^+ takes place, causing the potential of the cell to switch from negative to positive ($\sim$25 mV).
- Phase 1: A small dip in potential is encountered, signifying that the cell has partially repolarised. This is due to the rapid Na^+ channels closing and the influx of negative charged Chlorine ions, Cl^-.
- Phase 2: A plateau phase is reached. Extracellular Ca^{++} ions, migrates inside the cell while an efflux of K^+ ions is noted. This leads to an increase in intracellular Ca^{++} ions, while the net charge of the cell remains the same.
- Phase 3: This marks the repolarisation phase of the myocardium cell. The influx of Ca^{++} ions stops as the Ca^{++} channels are closed. Only the K^+ gates are opened for an efflux of K^+ ions. This leads to a gradual decrease of the cell potential till it reaches $-85\,\text{mV}$ to $-90\,\text{mV}$ before the K^+ gates are finally closed to reach Phase 4 again.

According to [10], the start and peak of action potential and contraction does not occur at the same time. The initialisation of the latter happens after the start of depolarisation and before the end of depolarisation. The reason for the delay is because contraction is a calcium-ion-induced mechanism which only occurs during phase 2, as previously described.

During that phase, the Ca^{2+} ions are transferred into the myocardium cell through the L-type calcium channel and bind to a protein receptor called the *Ryanodine Receptor 2* (RyR2). These receptors are attached to a network of membranes, the *Sarcoplasmic reticulum* (SR), which store the so-called *cytoplasmic calcium ions*. Upon contact with $[K^+]_i$, RyR2 signal to the SR to release the cytoplasmic calcium ion, which then binds with Troponin-C protein. This induces the movement of tropomyosin complex off from the binding sites found in the actin filament. These binding sites are usually present to prohibit interaction between the myosin globular head and the actin filament. But once cleared from the tropomyosin complex, the myosin globular head can move and contraction occur.

2.3 Cardiac cycle

The opening and closing of the heart valves may be used to define the starting and ending phase of the heart chambers during a heartbeat. However, a proper definition can be obtained when monitoring the chambers; pressure and volume evolution through a heartbeat. More specifically, the ventricular pressure and volume evolution is analysed since they are the most dominant part of the heart. A *cardiac cycle*, also known as a heartbeat, can be separated into two major phases, the *diastole* and *systole*.

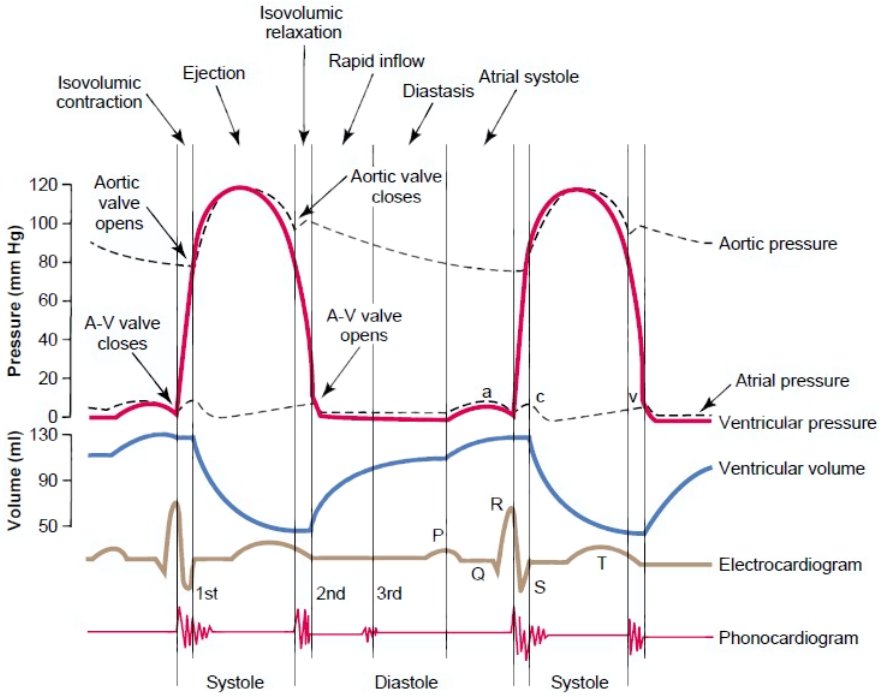


Figure 2.16: Pressure and volume change across time during one heartbeat [12].

Fig. 2.16 shows the starting and ending of the diastole and systole phase when the pressure and volume of a ventricle is recorded. Each phase can be subsequently broken down into periods. The diastole encompasses the *isovolumetric relaxation* and the *filling* period, while the systole includes the *isovolumetric contraction* and *ejection* period of the ventricle. The average phase durations are given in Tab.2.1.

Phase	Duration (s)
Filling	0.41
Isovolumetric contraction	0.05
Ejection	0.26
Isovolumetric relaxation	0.08
Total	0.8

Table 2.1: Phase duration of a heart beating at a rate of 75 min^{-1} [16].

An alternative way of representing the graph in Fig. 2.16 is by plotting a pressure-volume graph, as shown in Fig. 2.17.

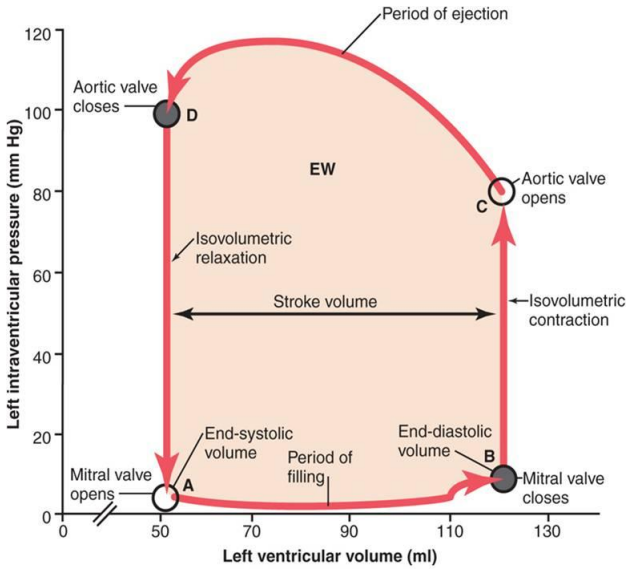


Figure 2.17: Ventricular pressure and volume loop of one heartbeat [12].

In this plot, the different phases of the heart can easily be distinguished by finding the corners of the loop, labeled as A, B, C and D. AB, BC, CD and DA represent filling, isovolumetric contraction, ejection and isovolumetric relaxation respectively. Additional information that can be extracted from the loop is the *end-systolic pressure-volume relationship*, ESPVR, and the *stroke volume*, V_S. The end-systolic pressure-volume relationship represents a line starting from the end-systole phase, joining the volume axis where the pressure is zero. In order to obtain it, the end-systole points of several heartbeats are used to fit a straight line. The gradient of the line is called the elastance, E_{PV}, and can be computed as [87]:

$$E_{PV}(t) = \frac{P(t)}{V(t) - V_{P=0}(t)}, \tag{2.1}$$

where V_0 is the ventricular volume when pressure is zero. P and V are the pressure and volume at a specific point in time, t, along the ESPVR line. The stroke volume is defined as the difference in volume between isovolumetric contraction and the isovolumetric relaxation line. The value obtained represents the amount of blood ejected from the ventricle, or more precisely, the *cardiac output*. Using stroke

volume, another cardiac performance measurement that can be extracted from the PV loop is the *ejection fraction*, E_{fraction}. It is defined by calculating the ratio of V_{stroke} and the end-diastole volume, V_{ED}:

$$E_{\text{fraction}} = \frac{V_{\text{stroke}}}{V_{\text{ED}}}. \tag{2.2}$$

The stroke volume and the end-diastole volume have an interesting relationship which is guided by the *Frank-Starling law*. This law specifically states that with different end-diastole volumes, the heart will adjust its contraction performance in order to achieve higher cardiac output. This therefore induces a higher ejection fraction and a larger stroke volume. The mechanism is possible as the myocardium stretches more with an increase in ventricular volume leading to a higher stretch ratio of sarcomere and larger contraction force, as seen in Section 2.2.2. It also holds true that with too large a stretch, the contraction force decreases as does the cardiac output, causing a smaller stroke volume.

Chapter 3

Cardiac Mechanics

3.1 Introduction

In this chapter, the mathematical models used to mimic the physiological details of the heart are introduced Each phase of a heartbeat is defined by a specific model, which will be elaborated. However, since the models have been derived from a classical continuum mechanics framework, that framework is first introduced.

3.2 Continuum Mechanics

Continuum mechanics consists of solid and fluid mechanics, which relies on the fundamental assumption where, if one continuously magnifies a piece of a material, the underlying molecular structure is generalised by the overlying material's properties. In this research, the basic principles of continuum mechanics are used to describe the biomechanics of the heart. For more detailed descriptions and derivations of Continuum mechanics, the reader may find the following books helpful: Mase et al. [88] and Lai et al. [89]

3.2.1 Kinematics

Consider a body, $\mathcal{B}$, in a three-dimensional Euclidean vector space, parameterised by the Cartesian co-ordinates and characterised as the reference state (i.e. the undeformed configuration). After being subjected to certain boundary conditions for a time, t, the body is now in a deformed state (i.e. the current configuration) and is referred as $\mathcal{B}_t$. To describe this transition of state, a non-linear deformation mapping $\mathcal{M} : \mathcal{B} \rightarrow \mathcal{B}_t$ is introduced, enabling us to define the relation of the two different states at any material point for $\mathbf{X} \in \mathcal{B}$ and $\mathbf{x} \in \mathcal{B}_t$ in the form of:

$$\mathbf{x} = \mathcal{M}(\mathbf{X}, t). \tag{3.1}$$

With $\mathcal{M}$ being known, a corresponding invertible linear tangent map, $\mathbf{F} = \text{Grad}\ \mathcal{M}$, known as the *deformation gradient*, is established. The determinant of $\mathbf{F}$ corresponds to the *Jacobian*, J, through

$$J = \det \mathbf{F} = \det\left(\text{Grad}\ \mathcal{M}\right) \quad \text{with } J > 0, \tag{3.2}$$

whereby the operator $\text{Grad}\,(\cdot) = \boldsymbol{\nabla}\,(\cdot) = \frac{\partial}{\partial \mathbf{X}}$ is defined with respect to the reference configuration and its corresponding current configuration operator is given by $\text{grad}\,(\cdot) = \boldsymbol{\nabla}_t\,(\cdot) = \frac{\partial}{\partial \mathbf{x}}$. Since the displacement field is expressed as $\mathbf{u}(\mathbf{X}, t) = \mathbf{x}(\mathbf{X}, t) - \mathbf{X}$, $\mathbf{F}$ can be restated as:

$$\mathbf{F} = 1 + \text{Grad}\ \mathbf{u}. \tag{3.3}$$

Any change in time is coined as the *local rate of change* and expressed as:

$$\frac{\partial}{\partial t}\,(\bullet) \tag{3.4}$$

and the overall time rate of change of a group of particles is given by,

$$\frac{D}{Dt}(\bullet) = \frac{\partial}{\partial t}(\bullet) + \frac{\partial}{\partial x_i}(\bullet)\frac{dx_i}{dt}. \tag{3.5}$$

In that sense, the Jacobian, which defines the change of volumes from different configuration states, can also vary with respect to time:

$$\dot{J} = J\ \text{div}\ \dot{\mathbf{x}}, \tag{3.6}$$

where $\text{div}\,(\cdot)$ is the *divergence* operation which is defined in the current configuration and $\text{Div}\,(\bullet) = \frac{\partial}{\partial X_i}\,(\bullet)\,.e_i$ in the reference configuration. Two important equations used in the study:

$$dv = \det(\mathbf{F})\ dV = J\ dV \tag{3.7}$$

$$\boldsymbol{\nu} da = \det(\mathbf{F})\mathbf{F}^{-T}\mathbf{n}\ dA. \tag{3.8}$$

Eq. (3.7) relates to the change in configuration of a volume, V, and Eq. (3.8) to the change in configuration of an area, a with $\mathbf{n}$ being the normal reference configuration and $\boldsymbol{\nu}$, the normal in the current configuration. Using the deformation gradient, the symmetric *right Cauchy-Green* tensor, $\mathbf{C}$, is defined as:

$$\mathbf{C} = \mathbf{F}^T\mathbf{F}. \tag{3.9}$$

In turn, the *Green strain* tensor, $\mathbf{E}$, which is symmetric in nature, can be introduced as a function of $\mathbf{C}$ using

$$\mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{1}) = \frac{1}{2}\left(\mathbf{F}^T\mathbf{F} - 1\right). \tag{3.10}$$

The stretch tensor and the left or right Cauchy-Green strain tensor are closely related through the following definition:

$$\mathbf{C} = \mathbf{U}^2. \tag{3.11}$$

3.2.2 Stress

In continuum mechanics, stress measures have been used in order to establish the magnitude and direction of these forces inside a body. Consider again the body, $\mathcal{B}$, subjected to a body force, $\mathbf{b}_{\mathcal{B}}$, and a surface force, $\mathbf{t}^{(\nu)}$, known as the traction vector, acting on a surface with a unit normal, $\boldsymbol{\nu}$. Applying the Cauchy stress principle to the resulting force, $\triangle\mathbf{f}$, acting on a surface element, da, inside $\mathcal{B}_t$, then:

$$\frac{d\mathbf{f}}{da} = \mathbf{t}^{(\nu)}. \tag{3.12}$$

Consequently, any point $\mathbf{x}$, forming part of the deformed body, $\mathcal{B}_t$, will have a stress state corresponding to the following equation:

$$\mathbf{t}^{(\nu)}(\mathbf{x}, t) = \boldsymbol{\sigma}^T(\mathbf{x}, t)\, \boldsymbol{\nu}(\mathbf{x}, t), \tag{3.13}$$

with $\boldsymbol{\sigma}$ representing the Cauchy stress tensor, defined in the deformed configuration, at a particular point $\mathbf{x}$. If the stress, at the same particular point, is defined in the current configuration but acts on an element surface, dA, with a unit normal of $\mathbf{n}$, which resides in the reference configuration, then Eq. (3.13) is reformulated as:

$$\mathbf{t}^{(\boldsymbol{n})}(\mathbf{X}, t) = \mathbf{P}(\mathbf{X}, t)\, \mathbf{n}(\mathbf{X}, t), \tag{3.14}$$

with $\mathbf{P}$ being the stress in the current configuration, also known as the first Piola-Kirchhoff stress tensor. $\mathbf{P}$ can be computed using the deformation gradient and the stress in the reference configuration, also known as the second Piola-Kirchhoff stress, $\mathbf{S}$, through:

$$\mathbf{P} = \mathbf{F}\,\mathbf{S}. \tag{3.15}$$

$\mathbf{P}$ can also be defined by using Eq. (3.8) and $\boldsymbol{\sigma}$ as follows:

$$d\mathbf{f} = \mathbf{P}\mathbf{n}\, dA = \boldsymbol{\sigma}^T \boldsymbol{\nu}\, da = \boldsymbol{\sigma}^T \det(\mathbf{F})\mathbf{F}^{-T}\mathbf{n}\, dA. \tag{3.16}$$

Therefore, the first Piola-Kirchhoff stress can also be calculated from:

$$\mathbf{P} = \det(\mathbf{F})\boldsymbol{\sigma}^T\mathbf{F}^{-T}. \tag{3.17}$$

It should be noted that while the Cauchy stress tensor, $\boldsymbol{\sigma}$, is symmetric, the first Piola-Kirchhoff stress tensor, $\mathbf{P}$, is not.

3.3 Variational principles

Following the introduction of the Continuum mechanics framework, the variational principles, used to solve the cardiac mechanics models, are provided in this section. The work of Holzapfel [90] is used to briefly introduce the theory of the variational principles. For a more in-depth review, the reader is referred to Hughes [91] and Zienkiewicz and Taylor [92].

The variation principles define the basis of the Galerkin method, which is itself the foundation of the Finite Element Method (FEM). For this work, since the solid mechanics aspect of the heart was investigated, a *single-field variational principle* was applied where the only unknown field was the displacement field, $\mathbf{u}$. The study of the variational principles starts with the law of conservation of linear momentum, which is recalled along with the Lagrangian equilibrium equation. At this point, any continuum body, $\mathcal{B}$ defined within such context can only experience a body force through the term $\mathbf{b}_{\mathcal{B}}(\mathbf{X}, t)$. To cater for external forces, the Lagrangian equilibrium equation is transformed into a so-called *boundary-value problem*. This leads to the introduction of *boundary conditions*, which are defined along the surface region, $\partial\mathcal{B}$, of the body surface, $\mathcal{B}$. Only two classes of boundary conditions are specified: *Dirichlet boundary conditions*, $\partial\mathcal{B}_{\mathrm{D}}$, and *von Neumann boundary conditions*, $\partial\mathcal{B}_{\mathrm{N}}$. Those surface conditions share the following characteristics:

$$\partial\mathcal{B} = \partial\mathcal{B}_{\mathrm{D}} \cup \partial\mathcal{B}_{\mathrm{N}} \quad \text{and} \quad \partial\mathcal{B}_{\mathrm{D}} \cap \partial\mathcal{B}_{\mathrm{N}} = \emptyset. \tag{3.18}$$

The boundary conditions also have physical meanings. Usually $\partial\mathcal{B}_{\mathrm{D}}$ are prescribed displacements, to, for example, represent support reactions. On the other end, $\partial\mathcal{B}_{\mathrm{N}}$ are traditionally imposed traction forces. With the boundary conditions defined, the *strong form* of the boundary value problem is stated as:

$$\mathrm{Div}\mathbf{FS} + \mathbf{b}_{\mathcal{B}} = \mathbf{0} \quad \text{in} \quad \mathcal{B} \tag{3.19}$$

$$\mathbf{u} = \bar{\mathbf{u}} \quad \text{on} \quad \partial\mathcal{B}_{\mathrm{D}} \tag{3.20}$$

$$\bar{\mathbf{t}}^{(\mathbf{n})} = \mathbf{t}^{(\mathbf{n})} = \mathbf{FSn} \quad \text{on} \quad \partial\mathcal{B}_{\mathrm{N}}. \tag{3.21}$$

$(\bar{\cdot})$ represents prescribed functions on the boundary surface, $\mathbf{t}$, the traction force vector and $\mathbf{n}$, the surface outward pointing norm vector. Applying the weighted residual method, i.e. casting the governing equation in integral form, multiplying by an arbitrary function, $\delta\mathbf{u}$, and applying subsequently the Gauss divergence theory, the weak form is obtained:

$$\int_{\mathcal{B}} \mathbf{S} : \delta\mathbf{E}\,\mathrm{dV} - \int_{\mathcal{B}} \mathbf{b}_{\mathcal{B}} \cdot \delta\mathbf{u}\,\mathrm{dV} - \int_{\partial\mathcal{B}_{\mathrm{D}}} \bar{\mathbf{t}}^{(\mathbf{n})} \cdot \delta\mathbf{u}\,\mathrm{dA} = 0. \tag{3.22}$$

The principle of virtual work is usually associated to Eq. (3.22) as it uses the concept of work done to explain the distribution of energy in the equation system. Hence, it can also be broken into two parts: *internal virtual work*, $\delta\mathcal{W}_{\mathrm{int}}$, and *external virtual work*, $\delta\mathcal{W}_{\mathrm{ext}}$, where

$$\delta\mathcal{W}_{\mathrm{int}} = \int_{\mathcal{B}} \mathbf{S} : \delta\mathbf{E}\,\mathrm{dV} \tag{3.23}$$

$$\delta\mathcal{W}_{\mathrm{ext}} = \int_{\mathcal{B}} \mathbf{b}_{\mathcal{B}} \cdot \delta\mathbf{u}\,\mathrm{dV} + \int_{\delta\mathcal{B}_{\mathrm{D}}} \bar{\mathbf{t}}^{(\mathbf{n})} \cdot \delta\mathbf{u}\,\mathrm{dA}. \tag{3.24}$$

For the system to be in equilibrium,

$$\delta\mathcal{W} = \delta\mathcal{W}_{\text{int}} - \delta\mathcal{W}_{\text{ext}} = 0. \tag{3.25}$$

In cardiac mechanics, body force, $\mathbf{b}_{\mathcal{B}}$, is the weight, and its effects on the mechanics of the heart are neglected, leading to $\mathbf{b}_{\mathcal{B}} = \mathbf{0}$. However, surface pressure boundary conditions are considered in order to represent blood pressure, P^{blood}, on the ventricular wall, redefining the traction term in Eq. (3.24) as:

$$\bar{\mathbf{t}}^{(\mathbf{n})} = P^{\text{blood}}\,\mathbf{n} \quad \text{on} \quad \partial\mathcal{B}_{\text{D},t}, \tag{3.26}$$

and consequently, the external virtual work done as:

$$\delta\mathcal{W}_{\text{ext}} = \int_{\partial\mathcal{B}_{\text{D},t}} P^{\text{blood}}\,\mathbf{n}\cdot\delta\mathbf{u}\;\text{da}. \tag{3.27}$$

The boundary surface $\partial\mathcal{B}_{D,t}$ is used here as the pressure is applied in the current configuration. Since the weak form is formulated in the reference configuration, the Nanson's formula is used to project the integral of $\partial\mathcal{B}_{D,t}$ to $\partial\mathcal{B}_D$ by modifying Eq. (3.27) into:

$$\delta\mathcal{W}_{\text{ext}} = \int_{\partial\mathcal{B}_{\text{D}}} P^{\text{blood}}\,J\mathbf{F}^{-\text{T}}\mathbf{n}\cdot\delta\mathbf{u}\;\text{dA}. \tag{3.28}$$

Subsequently, the final state of the weak form for cardiac mechanics is expressed as

$$\delta\mathcal{W} = \int_{\mathcal{B}} \mathbf{S}:\,\delta\mathbf{E}\text{dV} - \int_{\partial\mathcal{B}_{\text{D}}} P^{\text{blood}}\,J\mathbf{F}^{-\text{T}}\mathbf{n}\cdot\delta\mathbf{u}\;\text{dA} = 0. \tag{3.29}$$

3.4 Cardiac Models

The next step was to look at the mathematical models that reproduce the biomechanics, physiology and hemodynamics during the four phases of a heartbeat: filling, isovolumetric contraction, ejection and isovolumetric relaxation.

3.4.1 Filling stage

The filling stage was considered first. During this phase, the heart's myocardium is said to be acting passively since it does not trigger any contraction.

3.4.1.1 Constitutive equation

The constitutive equation of a material is based on its behaviour during experiments. Different tests carried out have found that the stress-strain relationship of passive myocardium is exponential in nature [14, 83]. That is, its strain increases quickly when subjected to an initial force, but very slowly when the force subsequently gets larger. Even though this exponential behaviour is observed across the heart, researchers [14, 83] have observed that its evolution differs, even though the same specimen is analysed in different directions.

To incorporate the fibres along with the exponential behaviour of the heart, into a constitutive law, several approaches can be considered. As stated by Usyk et al. [93], three forms of constitutive equation can be set to model the myocardium: isotropic, transversely isotropic and orthotropic. The difference in those three cases is the way the fibres are defined. In the isotropic form, there is no preferred direction. However, in the transversely isotropic case, the fibre direction is considered the preferred direction, whereas in the cross-fibre direction, the behaviour is still isotropic. For the orthotropic scenario, material properties differ across the three fibre directions. In this research, the myocardium was considered orthotropic in order to conform with observations made by [7] and [9].

Another important characteristic of the myocardium to be considered in the constitutive equation is its level of compressibility i.e. how the volume of the myocardium changes when subjected to a force. According to Yin et al. [94], the myocardium is slightly compressible, and, based on the research of Usyk et al. [93], near incompressibility was considered in their cardiac material model. Thus, accounting for nearly incompressible material involves an additional term to the strain energy function.

Taking into account the energy of the exponential and orthotropic behavior of the heart, $W_{\text{exp-ortho}}$ and the near incompressibility of the myocardium, W_{compr}, the total passive strain energy function, W_{passive}, is given as [20]:

$$W_{\text{passive}} = W_{\text{exp-ortho}} + W_{\text{compr}} \tag{3.30}$$

$$= A\frac{\left(e^{Q}-1\right)}{2} + A_{\text{compr}}\left[\det\mathbf{U}\ \ln\left(\det\mathbf{U}\right) - \det\mathbf{U} + 1\right], \tag{3.31}$$

where

$$\begin{aligned} Q = {} & a_1\left(\text{tr}\left(\mathbf{M}_1\mathbf{E}\right)\right)^2 + a_2\left(\text{tr}\left(\mathbf{M}_2\mathbf{E}\right)\right)^2 + a_3\left(\text{tr}\left(\mathbf{M}_3\mathbf{E}\right)\right)^2 \\ & + a_4\,\text{tr}\left(\mathbf{M}_1\mathbf{E}^2\right) + a_5\,\text{tr}\left(\mathbf{M}_2\mathbf{E}^2\right) + a_6\,\text{tr}\left(\mathbf{M}_3\mathbf{E}^2\right). \end{aligned} \tag{3.32}$$

In Eq. (3.31), A stands for the stress scaling coefficient while A_{compr} for the compressibility coefficient. The determinant operator is defined by $\det(\cdot)$ and the logarithmic operator by $\ln(\cdot)$. $\mathbf{U}$ represents the right stretch tensor. Q, stated by Eq. (3.32), is a function of the constants, a_j, the *Green-strain* tensor, $\mathbf{E}$ and a structural tensor $\mathbf{M}_i$ where $i = 1, 2, 3$ are the local Cartesian basis. $\mathbf{M}_i$ is constructed from the fibre orientation, which are $\tilde{\boldsymbol{f}}$, the fibre direction, $\tilde{\boldsymbol{s}}$, the sheet direction, and $\tilde{\boldsymbol{n}}_s$, the sheet normal direction, as follows:

$$\mathbf{M}_1 = \tilde{\boldsymbol{f}} \otimes \tilde{\boldsymbol{f}}, \quad \mathbf{M}_2 = \tilde{\boldsymbol{s}} \otimes \tilde{\boldsymbol{s}}, \quad \mathbf{M}_3 = \tilde{\boldsymbol{n}}_s \otimes \tilde{\boldsymbol{n}}_s. \tag{3.33}$$

Consequently, the constitutive equation, which relates the passive stress with the strain energy, is given as:

$$\mathbf{S}^{\text{Passive}} = \frac{\partial W_{\text{passive}}}{\partial \mathbf{E}}. \tag{3.34}$$

3.4.1.2 Tissue testing

The equation used in defining the passive constitutive equation is made up of eight parameters, i.e. A, A_{compr}, $a_1 \ldots a_6$. In order to assign quantitative values to them, experimental data are used. The two

tests from which that data can be obtained are *bi-axial* and *tri-axial* experiments.

For the bi-axial test, a thin square slice of the heart tissue is first extracted. The major sides of the square span longitudinally and transversely, while the thickness span radially across the heart's ventricle. The thickness is usually kept around 2 mm to mitigate the effect of out of plane shear [13, 95]. Along the top surface of the specimen, the mean fibre direction is recorded and nine black markers are placed as close as possible to the centre. Hooks are then inserted along the sides of the rectangular slice with regular spacing. In total, 20 hooks are used, with five on each side. Surgical sutures are used to connect the hooks to a perpendicularly aligned rod, mounted on a clamping device, that is moved backwards in order to induce stretch in the specimen. An overall view of the setup is given in Fig. 3.1.

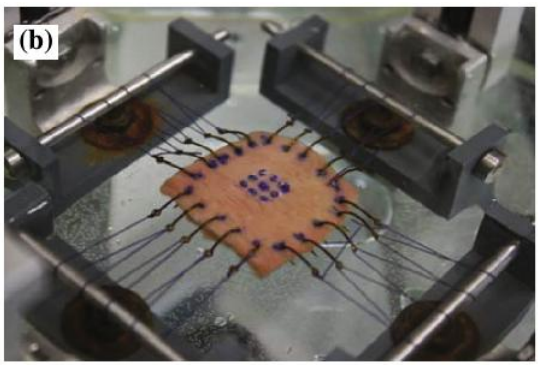


Figure 3.1: Biaxial test setup [13].

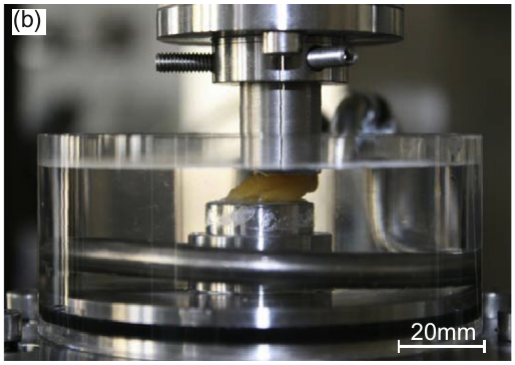


Figure 3.2: Shearing of tissue specimen during a triaxial shear experiment [14].

A force sensor is used to record the load transferred to the specimen, while a mounted camera records the movement of the black markers. Using the load and movement data, a stress and stretch curve is computed and plotted. To obtain values for the constants in the passive constitutive equation, its stress-strain plot is fitted to the biaxial one, using a parameter-fitting algorithm [96]. However, as explained by Holzapfel and Ogden [96], biaxial testing does not provide enough information to observe the stress variation along the thickness direction, and is not suitable for calibrating orthogonal tissue material. Therefore, for this research, triaxial test data was used.

The triaxial shear test is a way of obtaining material response through the shearing of cubic specimens. The setup of the experiment consists of an upper and lower platform to which the myocardium is glued and bathed in a solution to prevent dehydration, while keeping the temperature of the specimen constant. The lower platform is then moved with respect to the upper one, as shown in Fig. 3.2, in order to have shearing, while sensors measure the force generated. The location where the myocardium is extracted from the heart is important. In order to minimise the impact of an anisotropy effect on the shear experiment, the cubic specimen is kept small, with an edge length of about 4 mm, and is extracted from the basal midwall myocardium [14]. At this location, the fibre and sheet orientation is almost perpendicular to the cutting planes, as shown in Fig. 3.3.

With this fibre configuration, specific anisotropic types can be activated during shearing. To do so, both Dokos et al. [83] and Sommer et al. [14] used six modes of shearing, as shown in Fig. 3.4. The six modes are grouped into three families: FS and FN for the F-family, SF and SN for the S-family and NS and NF for the N-family. In the F-family, the shear modes work against the stiffness provided by the

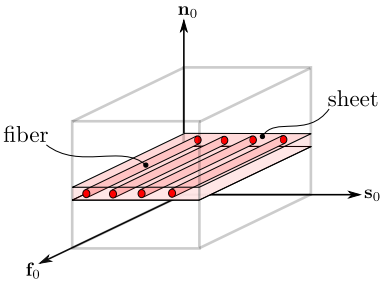


Figure 3.3: Fibre and sheet arrangement in the cubic myocardium.

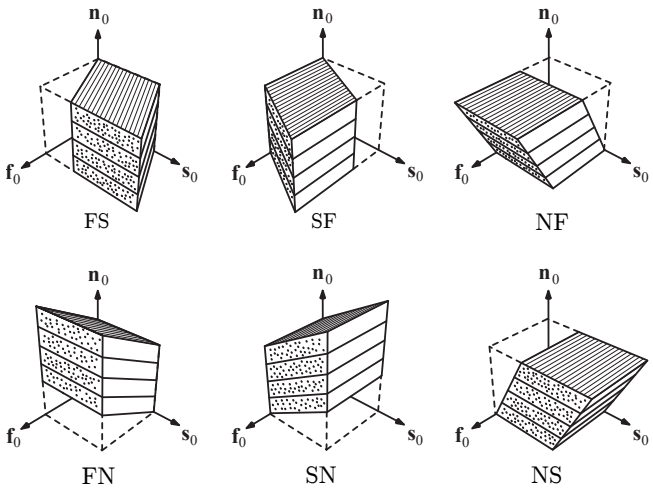
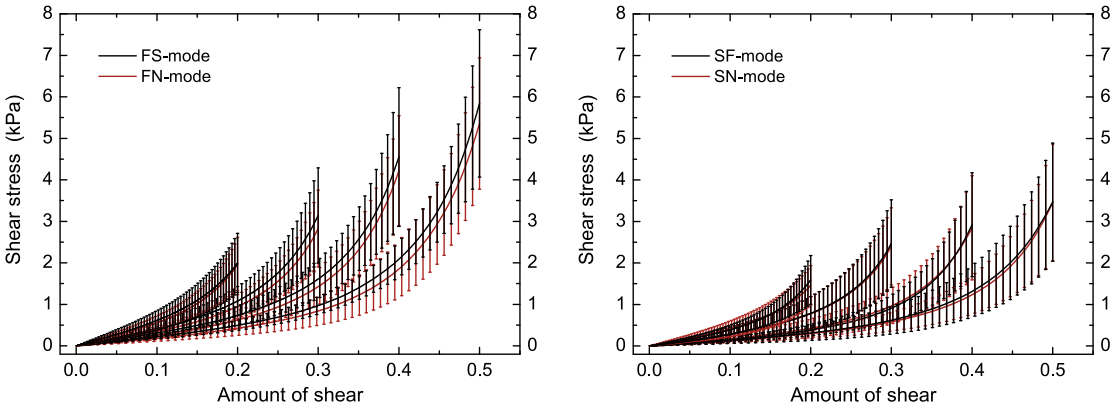


Figure 3.4: Six triaxial shearing modes [14].

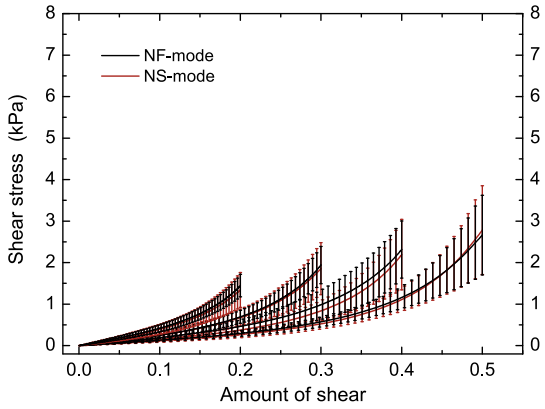
cross-sectional plane of the fibre. For the S-family, work is done mainly against the sheet plane, while for the N-family, the shear force is opposed by the resistance provided, purely due to the collagen matrix between the sheets.

From the stress-strain relationship of all the shear modes shown in Fig. 3.5, Sommer et al. [14] observed that the F-family was more stiff than the others, irrespective of the targeted level of stretch (0.1, 0.2, 0.3, 0.4 and 0.5). This observation confirms those made earlier by Dokos et al. [83]. Both authors were then able to conclude that the stiffness level could be sequenced as follows: F-family > S-family > N-family. The reason why FS and FN exhibited the highest stiffness was due to the elongation working parallel to the fibre direction.

By using the plot given in Fig. 3.5, the constants of the passive constitutive equation could be found. Using an initial set of assumed values, a stress-strain curve was generated from the constitutive equation and compared to the experimental one. If the curves did not fit, then the parameters were modified in an iterative process. According to the literature, this technique is commonly used [96, 97]. It was applied in this research, but through an automated process. The process, which falls in the category of parameter identification algorithms, is known as the *Levenberg-Marquardt Algorithm*, and is presented in Chapter 7.



(a) F-family. (b) S-family.



(c) N-family.

Figure 3.5: Stress-stretch plot of different shear families [14].

3.4.1.3 Myocardial fibre distribution

As explained in the previous chapter, the heart fibre orientation distribution dictates the overall behaviour of the heart. To have the model behaving in the correct way, the fibres located around the ventricles were incorporated in our formulation. The typical way to set them up is to first establish a spatial coordinate system which is defined by three directions/axes: *fibre axis*, *sheet axis* and *sheet-normal axis*. Using fibre angles extracted from human scans, those axes are rotated accordingly, while still maintaining their orthogonal characteristics.

The extraction of fibre directions can be done either on *in vivo* or *ex vivo* hearts. Previously, ex vivo heart fibre extraction methods were popular, as there was no other way to analyse the heart myocardium. In that case, the heart was first extracted and different techniques were used to record the angle of the fibre directions. For example, Nielsen et al. [98] used a measurement rig to record the angles while the ventricle was rotated by an axle introduced along the longitudinal axis of the heart. In other examples, the ventricles were cut in different locations, either in slices [8] or in blocks [99, 100]. From there, the fibre angles were read. With current technological advances, in vivo heart fibre direction can be obtained. This is done by using a method called *Diffusion tensor magnetic resonance imaging*, DTMRI, which consists of monitoring water molecules in the fibres of the heart. Data, obtained from a magnetic

resonance instrument, are represented by tensors at every point in the heart myocardium. By decomposing those tensors into eigenvectors, the fibre directions can be obtained when sorting them according to the decreasing order of their eigenvalues. The fibre-axis is defined by the eigenvector of the highest eigenvalue, while the sheet-normal is defined by the lowest eigenvalue [9, 84, 101].

In this work, due to the unavailability of experimental fibre angle data of ventricles, values reported in the literature were used instead. To setup the fibre vectors throughout a heart model, an algorithm developed by Wong and Kuhl [102] was implemented. It provided a fast and computationally inexpensive means of setting up the vectors, while considering the non-uniformity of the ventricular surface within the finite element framework. Its applicability to single or bi-ventricle models does not require any extra attention. Based on previous research [19, 103], this algorithm has proven to be successfully used in capturing the internal structural behaviour of the myocardium during a heartbeat. For explanation

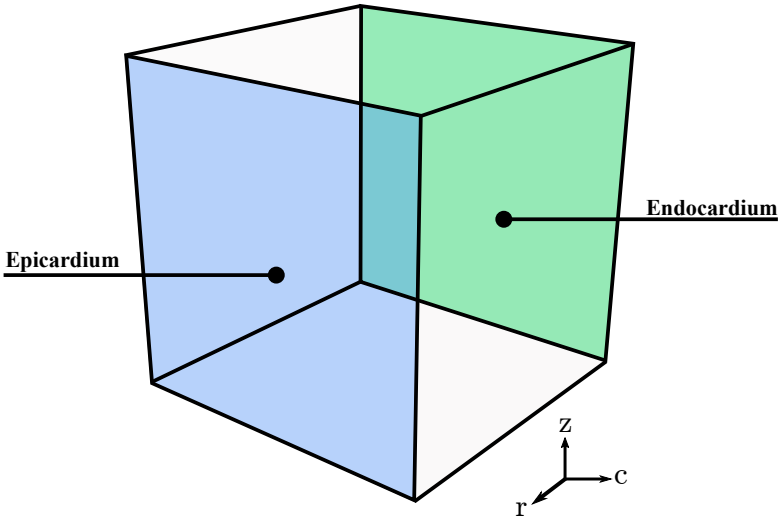


Figure 3.6: Surface definition of cube.

purposes, a cube, made up of 216 nodes and 125 hexahedral elements is set up within a coordinate system shown in Fig. 3.6. The cube's axes are then made to coincide with those defined for heart geometry. The longitudinal axis, denoted by z, runs from the base to the apex. The radial-axis, r, traverses the heart transmurally, from the epicardium to the endocardium, while the circumferential-axis, c, moves transversely, i.e. from either left to right, or vice versa. The axis planes are consequently defined as: cz-plane,P_{cz}, zr-plane P_{zr} and cr-plane, P_{cr}.

The first step of the Wong and Kuhl [102] method is to determine the local fibre-axis at every point on the defined surface of the cube. The r-axis acted along the same direction of the outward surface normals, $\mathbf{r}$, of a point on the surface of the cube. To obtain $\mathbf{r}$, the neighbouring surface to which the point is connected is found, and their normals, calculated. Then, an averaging operation is carried out over all neighbouring surface normals, as shown in Fig. 3.7. With the z-axis vector $\mathbf{z}$, defined globally to be the same for all nodes and surfaces, the only other direction that need to be found is the circumferential vector, $\mathbf{c}$ through:

$$\mathbf{c} = \phi^{\text{wall}} \left(\mathbf{z} \times \mathbf{r}\right) . \tag{3.35}$$

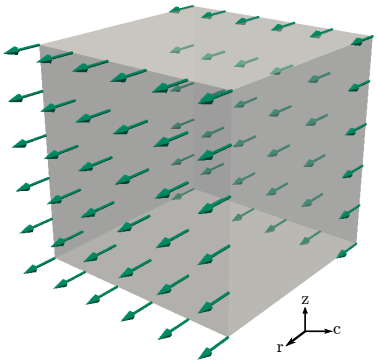


Figure 3.7: Averaged normal direction defined at epicardium and endocardium surface nodes.

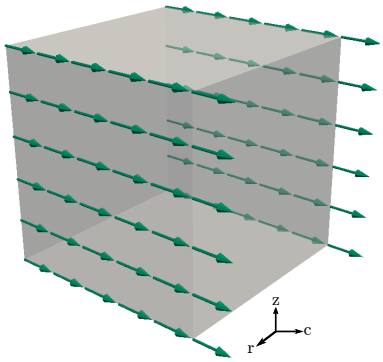


Figure 3.8: Circumferential direction defined at epicardium and endocardium surface nodes.

In Eq. (3.35), ϕ^{wall} is an orientation index which is defined as $+1$ on the epicardium and -1 on the endocardium, and leads to a circumferential vector as shown in Fig. 3.8. With the inversion of the circumferential vectors on the endocardium surfaces, the surface normal, defined point-wise, is re-computed:

$$\mathbf{r}_{\text{cz}} = \mathbf{c} \times \mathbf{z}. \tag{3.36}$$

With c, z and r locally defined on the surface points, the goal now is to rotate them in such a way that they are aligned with the natural fibre, sheet and sheet-normal orientation. Having $\mathbf{r}$ already aligned approximately in the same direction as the sheets, the sheet direction is calculated as:

$$\mathbf{s} = \text{sign}\,(\mathbf{r}_{\text{cz}} \cdot \mathbf{r})\,\mathbf{r}. \tag{3.37}$$

Next, the fibre vector is determined. First, a projection vector, $\mathbf{p}$, is created along the plane P_{cz} using the fibre angle, θ, which has been measured with respect to the circumferential axis:

$$\tilde{\mathbf{p}} = \text{proj}\,(\mathbf{f}) = \cos(\theta)\mathbf{c} + \sin(\theta)\mathbf{z}, \tag{3.38}$$

before being rotated tangential to the sheet plane, so as to conserve the orthogonal condition of the fibres, and form the fibre direction vector, $\mathbf{f}$:

$$\mathbf{f} = (-\tilde{\mathbf{p}} \cdot \mathbf{s})\,\mathbf{r}_{\text{cz}} + (\mathbf{r}_{\text{cz}} \cdot \mathbf{s})\,\tilde{\mathbf{p}}. \tag{3.39}$$

With $\mathbf{f}$ and $\mathbf{s}$ computed, the third direction, which is the sheet-normal $\mathbf{n}_s$, is obtained from:

$$\mathbf{n}_s = \mathbf{f} \times \mathbf{s}. \tag{3.40}$$

Up to this point, only the surfaces of the cube were assigned fibre directions. In order to have a distribution throughout the domain, Wong and Kuhl [102] used a Poisson-based interpolation scheme, implemented

within the Finite element framework. Each component of **f** and **s** was applied as boundary conditions and solved for. In total, six consecutive simulations had to be carried out in order to obtain the full fibre distribution.

From the current implementation, two major drawbacks were observed. Firstly, the generated sheet direction seemed to be uniform transmurally, due to usage of the pointwise surface normal,$\mathbf{r}_{cz}$ as given in Eq. (3.36). Secondly, the interpolation procedure was deemed to be computationally expensive in terms of time and resources, due to the solving of six finite element problems separately, using a fine mesh. In this case, a solution was proposed for each problem.

In the first scenario, a rotation was applied to the sheet direction with respect to the circumferential direction. The angle β, known as the sheet angle, was used to denote the rotation angle measured from the radial direction, **r**, as given in [9]. Using Rodrigues' rotation formula, Eq. (3.41), β was used to obtain the new sheet-direction. In order to preserve the orthogonal condition within the fibre directions, Eq. (3.41) was additionally applied to **f** and $\mathbf{n}_s$.

$$\tilde{\boldsymbol{f}} = \mathbf{f}\cos\beta + (\mathbf{c}\times\mathbf{f})\sin\beta + \mathbf{c}\,(\mathbf{c}\cdot\mathbf{f})\,(1-\cos\beta) \tag{3.41}$$

$$\tilde{\boldsymbol{s}} = \mathbf{s}\cos\beta + (\mathbf{c}\times\mathbf{s})\sin\beta + \mathbf{c}\,(\mathbf{c}\cdot\mathbf{s})\,(1-\cos\beta) \tag{3.42}$$

$$\tilde{\boldsymbol{n}}_s = \mathbf{n}_s\cos\beta + (\mathbf{c}\times\mathbf{n}_s)\sin\beta + \mathbf{c}\,(\mathbf{c}\cdot\mathbf{n}_s)\,(1-\cos\beta)\,. \tag{3.43}$$

For the second scenario, where a less computationally demanding method was looked for, the Moving Least Square approximation scheme(MLS) was chosen. MLS is based on setting up pointwise interpolants from nodal spatial coordinates defined within the Cartesian coordinate system. From these interpolants, an interpolation could be carried out from known data points to the unknown points, in order to obtain a full fibre distribution across the domain. For more information about MLS, the reader is referred to Section 5.3.2, where a more detailed explanation is given.

Combining the first and second scenario, the new algorithm was then used with the previously set up cube. On the epicardium, $\theta = -30°$ and $\beta = 25°$ was specified, while on the endocardium, $\theta = 30°$ and $\beta = -25°$ was imposed. Following this, the fibre orientations were generated, and are given in Fig. 3.9, Fig. 3.10, and Fig. 3.11.

3.4.2 Isovolumetric contraction and relaxation

Isovolumetric contraction and relaxation marks the start and end of the contractile behaviour of the myocardium. Due to this physiological similarity, along with the fact that the heart valves are closed and no change in volume is registered, the same mathematical model is used during those two phases.

In this work, a simple contraction mechanism known as the *active stress model* [17] was used. In using this model, the electrical wave propagation that was observed was simplified, while the whole myocardium of the heart was assumed to contract simultaneously. The active stress model, set up by Guccione et al. [17], is based on an additive decomposition of the total second Piola-Kirchhoff stress, $\mathbf{S}^{\text{Total}}$, at any point inside the myocardium into a passive stress, $\mathbf{S}^{\text{Passive}}$, and an active stress component, $\mathbf{S}^{\text{Active}}$,

$$\mathbf{S}^{\text{Total}} = \mathbf{S}^{\text{Passive}} + \mathbf{S}^{\text{Active}}. \tag{3.44}$$

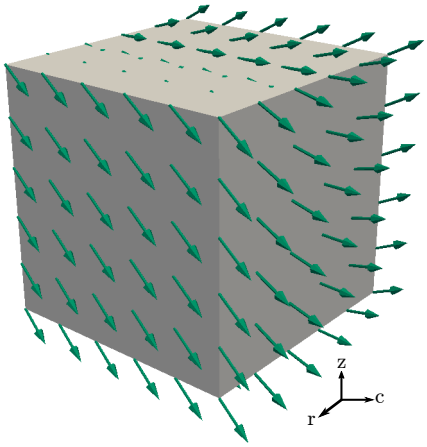


Figure 3.9: Fibre direction distribution across cube.

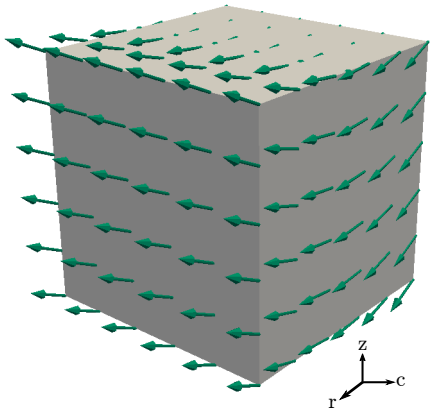


Figure 3.10: Sheet direction distribution across cube.

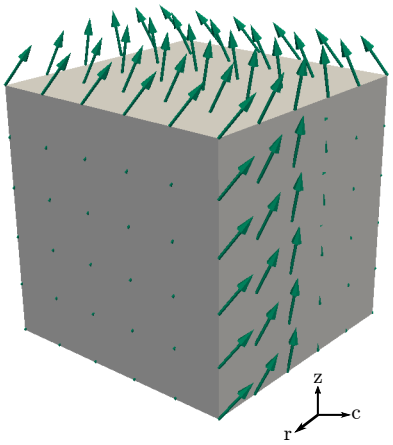


Figure 3.11: Sheet normal direction distribution across cube.

The passive stress is obtained from Eq. (3.34). However, the active stress has one component, T^{active}, in the fibre direction, $\tilde{\boldsymbol{f}}$, and is computed from:

$$\mathbf{S}^{active} = T^{active}\, \tilde{\boldsymbol{f}} \otimes \tilde{\boldsymbol{f}}. \tag{3.45}$$

Following several studies made on the cardiac tissue of rats, Guccione et al. isolated the sarcomere length and the calcium ion concentration as the principal features that dictate a contraction mechanism, and introduced the *Hill and Elastance* models. The computation of the active contraction force, T^{active}, along

the fibres of the heart was carried out using the following equation,

$$T^{\text{active}} = T_{\max} \frac{Ca_0^2}{Ca_0^2 + ECa_{50}^2} C_t, \tag{3.46}$$

where Ca_0 is the peak intracellular calcium ion concentration, ECa_{50} the extracellular calcium ion concentration when 50 % of peak active contraction force is achieved and $T_{\max}$, the *isometric* tension force under maximal activation. C_t, is the variable that defines how the active tension should change with time, and hence is defined as:

$$C_t = \frac{1}{2}\left(1 - \cos\omega\right). \tag{3.47}$$

ω is a function of time and is defined over three phases during a single one heart beat: (1) time to linearly reach maximal activation from rest (2) time to reach zero contraction from the maximum one, and (3), when no contraction is present. Each of them can be expressed in terms of equations such as:

$$\omega = \begin{cases} \pi \frac{t}{t_0} & \text{when } 0 \leq t < t_0, \\ \pi \frac{t - t_0 + t_r}{t_r} & \text{when } t_0 \leq t \leq t_0 + t_r, \\ 0 & \text{when } t_0 + t_r \leq t, \end{cases} \tag{3.48}$$

where t is the current time, t_0, the time at which maximum tension is reached and t_r, the time taken for the tension force to dissipate and reach zero. In order to include the dependency on mechanical deformation i.e. fibre stretch, the deformed sarcomere length, $l^{\text{sarcomere}}$ is accounted for in this model. Hence, t_r is calculated through a linear function involving two constants m and b along with $l^{\text{sarcomere}}$:

$$t_r = m l^{\text{sarcomere}} + b, \tag{3.49}$$

and ECa_{50}, from Eq. (3.46), was defined as:

$$ECa_{50} = \frac{(Ca_0)_{\max}}{\sqrt{e^{B\left(l^{\text{sarcomere}} - l_0^{\text{sarcomere}}\right)} - 1}}. \tag{3.50}$$

$(Ca_0)_{\max}$ represented the highest intracellular calcium ion concentration, $l_0^{\text{sarcomere}}$, the sarcomere length at rest state (i.e. absence of active tension force) and B is a constant that is characterised from an active tension - sarcomere length relationship. To allow for a variation of active tension force throughout the heart, the sarcomere length is made a function of the strain $\mathrm{E_{ff}}$, acting along the fibres' principal axis:

$$l^{\text{sarcomere}} = l_{\mathrm{R}}^{\text{sarcomere}} \sqrt{2\mathrm{E_{ff}} + 1}, \tag{3.51}$$

where $l_{\mathrm{R}}^{\text{sarcomere}}$ corresponds to resting sarcomere in the heart's unloaded configuration accounting for the residual stresses found in cardiac tissue. In [17], Guccione et al. used a set of parameters, given in Tab. 3.1, that was based on studies of the rat's myocardium. Those parameters have also been found to be adequately applicable to model the heart of other species. Such examples are: dogs [104], sheep [25, 105, 106] and humans [74, 107]. Given the fact that these sets of parameters are proven to be very flexible and stable for a wide range of applications, they were used in this research.

T_{max} (kPa)	$Ca_0(\mu M)$	$(Ca_0)_{max}(\mu M)$	$B(\mu M^{-1})$	$l_0(\mu M)$	t_0(s)	$m(\mu M^{-1})$	b(s)
137.5	4.35	4.35	4.75	1.58	0.1	1.0489	-1.429

Table 3.1: Active tension parameters of Guccione et al. [17].

3.4.3 Ejection

At the end of isovolumetric contraction, the ventricular pressure exceeds the aortic pressure and leads to the opening of the semi-lunar aortic valves. The ventricular blood flushes into the aorta while the ventricular walls continue contracting. This whole process takes place through two commonly known distinct phases: rapid ejection and reduced ejection.

In rapid ejection, the pressure of the ventricles and the aorta continuously increases. As the pressure is higher inside the ventricular chamber than in the aorta, the flow rate also increases continuously, while the ventricular volume encounters a sudden sharp decrease. About 70% of the ejected volume is expelled during a third of the ejection phase. Because the flow-rate from the aorta to the arteries is lower, the aortic pressure also builds up, leading to storing of potential energy in the aortic surface through the form of stretching. This ventricular-aortic flow-rate gradient then switches around after the peak flow occurs. This happens as the ventricular pressure and the aortic pressure reaches the same level. Given that the arterial wall cannot take in more blood as it is already overstretched, the flow rate starts to decrease.

At a certain point, the ventricle-to-aorta flow rate is equal to the flow of blood from the aorta to the rest of the body's arteries and veins, which is usually referred to as the *periphery*. This point is therefore defined as the end of the rapid ejection phase. With the flow rate still decreasing, the aorta-to-periphery flow is now higher, marking the start of the reduced phase and a decrease in aortic pressure. At the end of ejection, only about 40-50% of the blood volume that was present at the end of isovolumetric contraction remains, and is known as the *residual volume*. This point also demarcates the end of systolic phase and the start of diastolic phase.

To model the ejection phase, a mathematical model needs to primarily take into account the behaviour and characteristics of the ejecting ventricle and the arterial system in order to predict other quantities such as pressure, volume and flow rate. Additionally, this model will also need to reproduce the duration of the ejection period, while incorporating details such as rapid ejection and slow ejection phases.

One mathematical model that allows this, is the *Windkessel model*. In cardiac-related literature, its occurrence is common. For example, it was used by Creigen et al. [108] to model an artificial heart pump, and Molino et al. [109] to understand the arterial mechanical characteristic. Ottesen and Danielsen [110] made use of the Windkessel model to simulate a left ventricle with an arbitrary heart rate. In the context of cardiac modelling, many applications can also be found, such as in the work of Usyk et al. [93], Sainte-Marie et al. [111], Kerckhoffs et al. [112], Bovendeerd et al. [113] and Kerckhoffs et al. [114], where a single ventricular and bi-ventricular heart model successfully reproduced the ejection phase. Due to its popularity and validated results, the method was chosen to represent the current phase.

The Windkessel model is made up of a partial differential equation that is commonly used in three forms: 2-element Windkessel model, 3-element Windkessel model and 4-element Windkessel model. In order to explain each of them, an analogous concept where an electrical circuit consisting of capacitance, resistance and magnetic inductance, was used to describe the interaction between the left ventricle, the aorta and arteries, or the right ventricle and pulmonary arteries. Such analogy is commonly used in

explaining the Windkessel model, and the following Windkessel explanation is based on Creigen et al. [108].

- **Two-element Windkessel model**:
  The two-element Windkessel model is the most basic of the different Windkessel models and depends on only two physiological mechanisms of ejection: resistance and compliance. As explained earlier, while blood moves through the veins, especially when changing from bigger to smaller blood vessels, a resistance is encountered, slowing down the amount of blood being transferred from one point to the another. This resistance can be measured from a generalised equation:

$$R_p = \frac{P}{I}, \tag{3.52}$$

  where R_p is the resistance of the blood vessels of the entire circulatory system, P is the pressure difference and I the blood flow. For the elastic dilation of the blood vessels, the compliance can be calculated from:

$$I = C\,\frac{dP}{dt}, \tag{3.53}$$

  where the change of flow and pressure is considered as blood enters and leaves the blood vessels. A visual representation of resistance and compliance interacting together can be expressed through the following circuit:

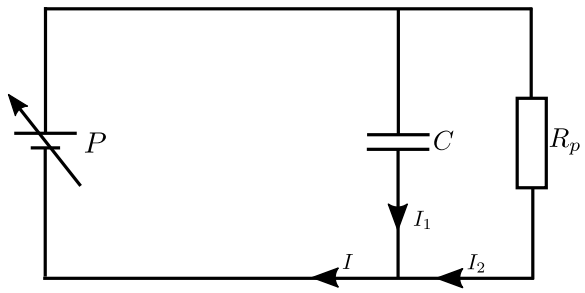


Figure 3.12: A two-element Windkessel analogous circuit.

In Fig. 3.12, the resistor, R_p is equivalent to the blood vessels resistance and acts as the capacitor to the elastic blood vessels' compliance C. P is the current source and equivalently, the pressure in the reservoir of the heart. The current flowing in the circuit is analogous to the blood flow in a vessel. Hence, in order to get the total current moving in the circuit, the current flowing through each component is added up:

$$I = I_1 + I_2, \tag{3.54}$$

which, replaced by Eq. (3.52) and Eq. (3.53) leads to the general two-elements Windkessel model:

$$I = \frac{P}{R_p} + C\,\frac{dP}{dt}. \tag{3.55}$$

In order to solve Eq. (3.55), an iterative scheme has to be employed, where the blood flow rate history is imposed for the pressure to be solved exactly. However, in the case of no blood flow ($I(t) = 0$), then the exact analytical solution can be found:

$$P(t) = P_o^{\text{aorta}}\, e^{-\frac{t}{R_P C}}, \tag{3.56}$$

with P_o^{aorta} being the initial aortic pressure.

- **Three-element Windkessel model**:
  In the 3-element Windkessel model, the resistance due to the aortic valve, R_a, is additionally considered. The aortic valve is made up three leaflets attached to a circular ring at the entrance of the aorta. It regulates the passage of blood coming from the ventricle. Aortic valves can have different opening diameters/areas which causes additional resistance during blood flow [115]. This therefore leads to a new resistor component in the 2-element Windkessel circuit: With this new

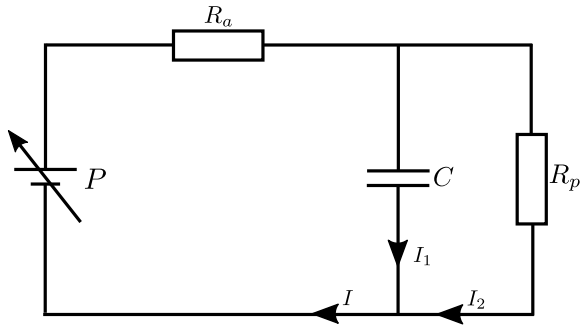


Figure 3.13: A three-element Windkessel analogous circuit.

configuration, the following differential equation can be formed:

$$\left(1 + \frac{R_a}{R_p}\right) I(t) + C R_a \frac{dI(t)}{dt} = \frac{P(t)}{R_p} + C\frac{dP(t)}{dt}. \tag{3.57}$$

If the blood flood is set to zero, then $I(t) = 0$ and the rate of change of blood flow also drops out as $\frac{dI(t)}{dt} = 0$. Hence, the exact analytical solution is unaffected by the aortic valve and is the same as Eq. (3.56).

- **Four-element Windkessel model**:
  The four-element Windkessel model introduces a new constant into the three-element Windkessel model called *inductance*, L. It represents inertia of the blood flow, or an increase/decrease of pressure due to changes in flow [116] from the ventricle to the arteries. This leads to a more accurate solution at low frequency of heartbeats [117]. The corresponding circuit diagram of the four-element Windkessel model is given as follows, with L being a magnetic inductance:

  Along with the additional inductance constant, the second-order rate of change of the blood flow is considered to account for the pressure change:

$$\left(1 + \frac{R_a}{R_p}\right) I(t) + \left(R_a C + \frac{L}{R_p}\right) \frac{dI(t)}{dt} + LC\frac{d^2 I(t)}{dt^2} = \frac{P(t)}{R_p} + C\frac{dP(t)}{dt}. \tag{3.58}$$

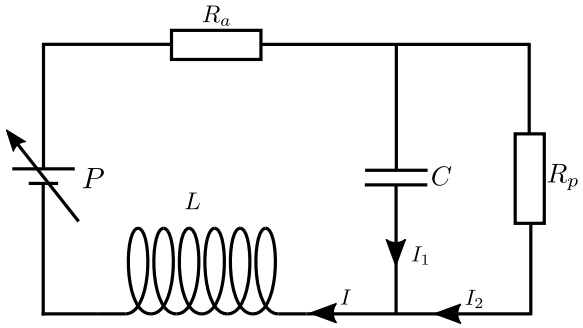


Figure 3.14: A four-element Windkessel analogous circuit.

Moving from the 2- to the 4-element Windkessel model increases the predictive accuracy of the model. Based on the work of Parragh et al. [116] and Westerhof et al. [117], it was found that the gain in accuracy is higher from the 2-element Windkessel to the 3-element Windkessel but less from the 3-element Windkessel to the 4-element one.

Even though the 4-element Windkessel model looks more attractive from a physiological point of view as it includes most of the physiological mechanism of ejection and is more accurate, it has however, two major drawbacks. Firstly, it's formulation consists of a second-order flow rate derivative. Such a term adds more complexity and instability in a solving scheme. Secondly, the inductance constant is difficult to quantify from experimental data [117]. Due to these reasons, the 3-element was preferred over the 4-element Windkessel in this research. The latter has the ability to provide a balanced solution between accuracy and complexity.

Throughout the literature, the usage of 3-element Windkessel model in cardiac mechanics has been found to be feasible and promising. In [111], Sainte-Marie et al. modelled a generic bi-ventricle model over six cycles. Kerckhoffs et al. [112] and Bovendeerd et al. [113] respectively studied different activation mechanisms and the influence of fibre orientation on the heart while using the Windkessel model to obtain a heart beat of a truncated prolate spheroidal ellipsoid model to represent a left ventricle. And finally, Sermesant et al. [118] used the Windkessel with a real bi-ventricle heart extracted from Magnetic Resonance Images (MRI).

Chapter 4

Computational Cardiac Mechanics

4.1 Introduction

This chapter discusses the implementation of the cardiac mechanic models defined in the previous chapter within the Galerkin method framework. The different approaches used to assemble and solve systems of equations are provided for each phase of a heartbeat.

4.2 Element-free Galerkin method

Eq. (3.29) needed to be solved over a heart domain in order to obtain mechanical quantities such as displacement, stress and strain. To do so, the *Element-free Galerkin method* (EFG) [119] was used in this research. It is very similar to the Galerkin method, the main differences being approximating the displacement field and its spatial derivatives using EFG shape functions, and employing a numerical integration scheme to replace the analytical integrals.

Being classified as a meshfree method, EFG's shape function calculations are based on a nodal discretised domain instead of a mesh one. Due to the non-connectivity feature, a neighbourhood-based approximation method was used, and thus, the *Moving Least Square approximation* method (MLS) [120] was employed. MLS is not mathematically presented in this section as it is introduced in Section 5.3.2. However, its properties are briefly given in order to understand their influence on EFG. MLS uses an influence zone to define the connectivity of nodes. Any neighbouring node found inside the influence zone is considered a supporting particle. The number of supporting particles can be regulated by changing the type and size of the influence zone. The approximants are built up from the spatial coordinates of the particles and because MLS is a partition of unity, the sum of the approximants computed from an influence zone should be equal to 1. Even though the MLS is a point-wise calculation, its approximants, along with their derivatives, are smooth. The MLS does not satisfy the *Kronecker delta* properties since it does not ensure that the approximant of the node of interest is one, a critical condition for the direct application of the Dirichlet boundary condition. Consequently, special methods need to be used such as the Lagrange multipliers and modified variational principles, or the modified boundary collocation method [121].

In the EFG procedure, the first step is to discretise the heart domain. A trial function and test function

are set up to approximate the displacement field solution, $\mathbf{u}$, using the MLS shape function:

$$\mathbf{u}(\mathbf{x}) = \mathbf{N}_s^{\mathrm{T}}\, \mathbf{u}_s \tag{4.1}$$

$$\delta\mathbf{u}(\mathbf{x}) = \mathbf{N}_s^{\mathrm{T}}\, \delta\mathbf{u}_s, \tag{4.2}$$

where $\mathbf{u}(\mathbf{x})$ is the interpolated displacement at point $\mathbf{x}$ and the point $\mathbf{x}$ belongs to the heart domain. $\mathbf{N}_s$ and $\mathbf{u}_s$ are the shape functions and displacements at the supporting nodes, s, for point $\mathbf{x}$. The trial function, test function and weak form of the Lagrangian equilibrium equation, Eq. (3.29), are then further used to obtain the *discrete equation system*:

$$\mathbf{K}\Delta\mathbf{u} - [\mathbf{f}_{\mathrm{ext}} - \mathbf{f}_{\mathrm{int}}] = \mathbf{0}. \tag{4.3}$$

$\Delta\mathbf{u}$ is known as the displacement field increment, $\mathbf{f}_{\mathrm{ext}}$, the external applied force and $\mathbf{f}_{\mathrm{int}}$, the internal reaction vector. $\mathbf{K}$ corresponds to the stiffness matrix of size $t_{\mathrm{dof}} \times t_{\mathrm{dof}}$, where t_{dof} is the total number of degrees of freedoms (dofs). For a more detailed derivation of the discrete equation system, the reader is referred to Dolbow and Belytschko [21], and additionally to Hughes [91], Zienkiewicz and Taylor [92] and Belytschko et al. [119].

4.3 Solving non-linear problems

Solving a system of equation for linear problems, as given in Eq. (4.3), is straightforward. But in the case of cardiac modelling, which forms part of non-linear problems, an iterative method is employed. The iterative scheme employed in this research was the *Newton method*. The Newton method is commonly used and simple to implement [90]. One of its advantageous characteristics is its quadratic convergence rate if the starting point is in the vicinity of the solution.

The first step of the Newton method consists in linearising the weak form equation with respect to the quantity that is being solved, i.e. the displacement field $\mathbf{u}$. Therefore, the discrete system of equation assembled for an iteration, i, and a time or loading step, n, from the weak form Eq. (3.29), can be linearised using a first order Taylor expansion:

$$\delta\mathcal{W}(\mathbf{u}) = \delta\mathcal{W}\left(\mathbf{u}_n^{i-1} + \Delta\mathbf{u}\right) = \delta\mathcal{W}\left(\mathbf{u}_n^{i-1}\right) + \left.\frac{\partial\delta\mathcal{W}(\mathbf{u})}{\partial\mathbf{u}}\right|_{\mathbf{u}_n^{i-1}} \cdot \Delta\mathbf{u}, \tag{4.4}$$

where $\Delta\mathbf{u}$ is the increment in displacement field. Since linearisation is a function approximation technique, then the total energy in Eq. (3.29) is no longer strictly equal to zero:

$$\delta\mathcal{W}\left(\mathbf{u}_n^{i-1}\right) + \left.\frac{\partial\delta\mathcal{W}(\mathbf{u})}{\partial\mathbf{u}}\right|_{\mathbf{u}_n^{i-1}} \cdot \Delta\mathbf{u} \approx 0. \tag{4.5}$$

With the weak form linearised, the discrete equation system can be now expressed as:

$$\mathbf{K}_n^i \Delta\mathbf{u}_n^i = [\mathbf{f}_{\mathrm{ext}}]_n^i - [\mathbf{f}_{\mathrm{int}}]_n^i = \mathbf{R}_n^i, \tag{4.6}$$

where $\mathbf{R}_n^i$ is the residual vector. Due to the non-linear problem, $\mathbf{R}_n^i$ and the tangent matrix, $\mathbf{K}_n^i$, has to be re-evaluated at every iteration. To do so, the previous iteration's displacement field $\mathbf{u}_n^{i-1}$ is used as

$\mathbf{R}_n^i = \mathbf{R}\left(\mathbf{u}_n^{i-1}\right)$ and $\mathbf{K}_n^i = \mathbf{K}\left(\mathbf{u}_n^{i-1}\right)$. From Eq. (4.6), the increment that needs to be solved for is $\Delta\mathbf{u}_n^i$:

$$\Delta\mathbf{u}_n^i = \left[\mathbf{K}_n^i\right]^{-1}\mathbf{R}_n^i. \tag{4.7}$$

With the change in displacement field obtained, $\mathbf{u}_n^i$ is updated using:

$$\mathbf{u}_n^i = \mathbf{u}_n^{i-1} + \Delta\mathbf{u}_n^i. \tag{4.8}$$

The updated displacement field can now be used in the next iteration, $i+1$, to update the tangent and residual matrix. For the case of $i = 0$, where a new calculation step, n, is started, the converged displacement field obtained from $n-1$ is instead used, leading to $\mathbf{u}_n^{i+1} = \mathbf{u}_{n-1}$. In the special case where $n = 0$ and $i = 0$, that is the start of the calculation, then $\mathbf{u}_0^0 = \mathbf{0}$.

As the Newton method converges on the solution, the residual, $\mathbf{R}_n^i$, decreases to zero. To define a stopping criterion for Newton's algorithm, the *Euclidean norm* of the residual is computed by:

$$\|\mathbf{R}_n^i\| = \sqrt{\left[\mathbf{R}_n^i\right]^{\mathrm{T}}\mathbf{R}_n^i}, \tag{4.9}$$

and compared to a specified tolerance value, γ_{tol}. When $\|\mathbf{R}_n^i\| \leq \gamma_{\mathrm{tol}}$, then the next step, $n+1$, is considered. Reaching this stage also means that an equilibrium state has been achieved between the internal reactions and the external applied forces. Hence, at the converged iteration step, i_c,

$$[\mathbf{f}_{\mathrm{ext}}]_n^{i_c} \approx [\mathbf{f}_{\mathrm{int}}]_n^{i_c}\ . \tag{4.10}$$

4.4 Solving cardiac mechanics during a heartbeat cycle

The solving of cardiac models as described in Chapter 3, can be done using two calculation mechanisms. These are *pressure-driven*, where a surface force is applied in an incremental fashion and *displacement-driven*, where a cavity volume is prescribed in the form of displacement. In this research, the diastole was controlled using the pressure-driven mechanism, while the isovolumetric contraction, the ejection and isovolumetric relaxation phase were displacement-driven.

Passive modelling

During the diastole filling process, the heart valves are opened and blood is allowed in. As the ventricles fill, the blood exerts hydrostatic pressure along the myocardium wall to cause an overall ventricular dilatation. Due to this research focusing only on the solid mechanics aspect of cardiac modelling, the blood fluid was replaced by surface pressure that was perpendicular across the ventricular wall. In the literature, this approach is found to be commonly used [26, 93, 113, 122, 123].

With the applied surface pressure, a pressure-driven algorithm is used. The aim of such algorithms is to apply a step-wise increasing pressure until the end-diastole pressure (P^{ED}) is reached. Applying the full P^{ED} all at once would lead to instabilities or incorrect deformation states, due to the non-linearity of the problem.

To obtain the step-wise pressure increment, a so-called *Neumann loading factor*, ζ_n is introduced. ζ_n

is specified to be positive and less than unity, leading to $0 < \zeta \leq 1$. Hence, at any Newton step n, the blood pressure, P, is defined as:

$$P = \zeta_n \, P^{\mathrm{ED}}. \tag{4.11}$$

In order to increase the pressure at every new step, $n + 1$, the Neumann loading factor is updated as follows:

$$\zeta_{n+1} = \zeta_n + \Delta\zeta_{n+1}, \tag{4.12}$$

where $\Delta\zeta_n$ is the factor increment which is calculated at step $n + 1$. Since the factor increment is updated at every step, it can either be set as constant for the whole calculation, or be allowed to change. As P is expressed as a function of ζ_{n}, the variational formulation is modified correspondingly by substituting Eq. (4.11) in Eq. (3.28),

$$\int_{\partial\mathcal{B}_{\mathrm{N}}} P J \mathbf{F}^{-T} \mathbf{n} \cdot \delta\mathbf{u} \, dA = \zeta_n \int_{\partial\mathcal{B}_{\mathrm{N}}} P^{\mathrm{ED}} J \mathbf{F}^{-T} \mathbf{n} \cdot \delta\mathbf{u} \, dA. \tag{4.13}$$

$[\mathbf{f}_{\mathrm{ext}}]^i_n$ is defined by the left-hand side of Eq. (4.13) and will therefore need to be restated by considering the right-hand side of Eq. (4.13) instead. With the end diastole force vector expressed as $\left[\mathbf{f}^{\mathrm{ED}}_{\mathrm{ext}}\right]$, then

$$[\mathbf{f}_{\mathrm{ext}}]^i_n = \zeta_{\mathrm{n}} \left[\mathbf{f}^{\mathrm{ED}}_{\mathrm{ext}}\right]. \tag{4.14}$$

As such, the discrete equation system Eq. (4.6), is re-stated as:

$$\mathbf{K}^i_n \Delta\mathbf{u}^i_n = \zeta_{\mathrm{n}} \left[\mathbf{f}^{\mathrm{ED}}_{\mathrm{ext}}\right] - [\mathbf{f}_{\mathrm{int}}]^i_n = \mathbf{R}^i_n. \tag{4.15}$$

Isovolumetric modelling

At the end of the diastole, the valves of each ventricle close and a cardiac action potential is released to the bottom of the heart through the high-speed fibres called *Purkinje fibres*, which are found along the ventricular free wall. The action potential propagates through the myocardium of the heart and induces a contractile behaviour in the myocardium. Since the valves are closed, the blood in the cavities is further compressed against the ventricular wall. However, no change in volume of the cavities was recorded while the pressure build-up was observed. The enforcement of no change in cavity volume and the subsequent rise in ventricular blood pressure could be related. Consider a Newton iteration, i, where the equilibrium state between applied blood pressure, active contraction and elastic stress in the heart wall is sought. If a specific displacement field, $\mathbf{u}$, is required to conserve negligible change in cavity volume, $\Delta V \approx 0$, then as per Eq. (4.15), the only variable that must be found is the required cavity pressure increase, that is the corresponding Neumann loading factor increment $\Delta\zeta$. After following the procedure given in the work of Skatulla and Sansour [2] to determine $\Delta\zeta$, the loading step factor is then updated:

$$\zeta^i_n = \zeta^{i-1}_n + \Delta\zeta^i_n, \tag{4.16}$$

where ζ_n^{i-1} is the loading step factor from the previous iteration. The end of the isovolumetric contraction phase is reached when the cavity pressure is equal to or greater than the aortic pressure, ζ_{aorta}. Hence, the stopping criterion for the isovolumetric contraction is $\zeta \geq \zeta_{\text{aorta}}$.

Ejection modelling

The modelling of the ejection phase is formulated from the same equation system used in the previous section. Consequently, the calculation procedure is quite similar, except for the condition where $\Delta V = 0$ is enforced and ΔV and ΔP are controlled by the Windkessel ODE and the mechanical equilibrium. As such, the solution of $\Delta P(\Delta\zeta)$ and the displacement field, $\mathbf{u}$, need to satisfy both, and is iteratively determined by the Newton Raphson method. Keeping the same numerical framework of i being the iteration step and n, the time step, then Eq. (3.57) is stated as follows:

$$CR_0 \left[\frac{dI_{\text{ao}}}{dt}\right]_n^i + \left(1 + \frac{R_0}{R}\right)[I_{\text{ao}}]_n^i = C\left[\frac{dP_{\text{ao}}}{dt}\right]_n^i + \frac{1}{R}[P_{\text{ao}}]_n^i, \tag{4.17}$$

where, C is the arterial compliance, R, is the peripheral resistance and R_0 is the flow resistance. $[P_{\text{ao}}]_n^i$ and $[I_{\text{ao}}]_n^i$ representing the aortic pressure and the aortic flow rate respectively, these are the only quantities updated at every step since they relate directly to $\Delta\zeta_n^i$ and ΔV_n^i. Applying the first physiological observation where the aortic flow rate is defined as the negative rate of change of the ventricular cavity volume:

$$[I_{\text{ao}}]_n^i = [I_{\text{cav}}]_n^i = -\left[\frac{dV_{\text{lv}}}{dt}\right]_n^i. \tag{4.18}$$

Similarly, as the heart valves are opened, the pressure inside the cavities is approximately equal to the pressure in the aorta at any given iteration step:

$$[P_{\text{ao}}]_n^i \approx [P_{\text{cav}}]_n^i. \tag{4.19}$$

Incorporating the two prior assumptions, then the new formulated 3-element Windkessel model is expressed as:

$$CR_0 \left[\frac{dI_{\text{cav}}}{dt}\right]_n^i + \left(1 + \frac{R_0}{R}\right)[I_{\text{cav}}]_n^i = C\left[\frac{dP_{\text{cav}}}{dt}\right]_n^i + \frac{1}{R}[P_{cav}]_n^i. \tag{4.20}$$

In the event of small time increment, then $dt \approx \Delta t$, $dV_{\text{cav}} \approx \Delta V_{\text{cav}}$ and $dP_{\text{cav}} \approx \Delta P_{\text{cav}}$, Hence, Eq. (4.20) is restated as:

$$CR_0 \left[\frac{\Delta I_{\text{cav}}}{\Delta t}\right]_n^i + \left(1 + \frac{R_0}{R}\right)[I_{\text{cav}}]_n^i = C\left[\frac{\Delta P_{\text{cav}}}{\Delta t}\right]_n^i + \frac{1}{R}[P_{cav}]_n^i. \tag{4.21}$$

The cavity pressure is set to behave in two different ways, depending on the stage of the Newton iteration process. When $i = 0$, i.e. the start of the iterative process, the loading factor, ζ_n^i is the same as the one at the time step, $n - 1$. Hence, $\zeta_n^i = \zeta_{n-1}$. But as $i > 0$, ζ^i is kept fixed. This therefore leads to the

following:

$$[P_{cav}]_n^i = \begin{cases} \zeta_{n-1} P_0 & \text{for } i = 0 \\ \zeta_n^i P_0 & \text{for } i > 0. \end{cases} \tag{4.22}$$

To evaluate the time increments of I_{cav} and P_{cav} in Eq. (4.21), the backward differences method is made use of. For the aortic flow, the equation takes the form of:

$$\left[\frac{\Delta I_{\text{cav}}}{\Delta t}\right]_n^i = \frac{[I_{\text{cav}}]_n^i - [I_{\text{cav}}]_{n-1}}{\Delta t}, \tag{4.23}$$

with $[\Delta I_{\text{cav}}]_{n-1}$ being the flow rate from the previously converged time step. For the cavity pressure, two different scenarios have to be again considered: For $i = 0$, $\Delta P_{\text{cav}}/\Delta t$ is equal to the previously converged time steps while $i > 0$, the time dependent change in loading factor is used:

$$\left[\frac{\Delta P_{\text{cav}}}{\Delta t}\right]_n^i = \begin{cases} [\Delta P_{\text{cav}}/\Delta t]_{n-1} & \text{for } i = 0 \\ P_0 \frac{\zeta_n^i - \zeta_{n-1}}{\Delta t} & \text{for } i > 0. \end{cases} \tag{4.24}$$

With the pressure, rate of change of pressure and rate of change of aorta flow rate defined, then Eq. (4.22), Eq. (4.23) and Eq. (4.24) can be replaced in Eq. (4.21) in order to find the aorta flow rate $[\Delta I_{\text{cav}}]_n^i$:

$$[\Delta I_{\text{cav}}]_n^i = \begin{cases} \left[\frac{1}{\frac{CR_0}{\Delta t} + \left(1 + \frac{R_0}{R}\right)}\right] \left(C\left[\frac{\Delta P_{\text{cav}}}{\Delta t}\right]_{n-1} + \frac{p_0 \zeta_{n-1}}{R} + \frac{CR_0}{\Delta t} I_{n-1}\right) & \text{for } i = 0 \\ \left[\frac{1}{\frac{CR_0}{\Delta t} + \left(1 + \frac{R_0}{R}\right)}\right] \left(CP_0\left[\frac{\zeta_n^i - \zeta_{n-1}}{\Delta t}\right] + \frac{p_0 \zeta_n^i}{R} + \frac{CR_0}{\Delta t} I_{n-1}\right) & \text{for } i > 0. \end{cases} \tag{4.25}$$

For $i = 0$, the first part of Eq. (4.25) was calculated using an explicit one-step Euler backward prediction, while for $i > 0$, an implicit one-step Euler Backward prediction was used. After obtaining $[I_{\text{cav}}]_n^i$, the volume change can now be computed:

$$\Delta V_n = -\Delta t \, [I_{\text{cav}}]_n^i \, . \tag{4.26}$$

A negative sign is introduced to represent a decrease in volume due to contraction and ejection. Once the volume change is obtained, the Neumann loading coefficient and ζ_n^i are determined as described in [2], and the displacement increment, $\Delta \mathbf{u}_n^i$, is consequently updated.

For solving the 3-element Windkessel model, a residual $|\text{R}_{\text{Windkessel}}|$, is also calculated in order to define a stopping criterion, δ_w, which will be activated once $|\text{R}_{\text{Windkessel}}| \leq \delta_w$. To compute $|\text{R}_{\text{Windkessel}}|$, the following equation is used:

$$|\text{R}_{\text{Windkessel}}| = \left[C\left[\frac{\Delta P_{\text{cav}}}{\Delta t}\right]_n^i + \frac{1}{R}[P_{cav}]_n^i\right] - \left[CR_0\left[\frac{\Delta I_{\text{cav}}}{\Delta t}\right]_n^i + \left(1 + \frac{R_0}{R}\right)[I_{\text{cav}}]_n^i\right]. \tag{4.27}$$

In Eq. 4.27, $[I_{\text{cav}}]_n^i$ and $[P_{cav}]_n^i$ are the un-upated flow rate and cavity pressure for the iteration, i. The target volume increment was updated after every iteration until the Windkessel residual fell below the

specified threshold. From there, the target volume increment was kept fixed for the remaining iterations. For a calculation step to be completed, both the Windkessel and the discrete equation residual norm should be lower than their respective thresholds.

At the start of the 3-element Windkessel calculation, the initial flow rate and rate of change of flow rate were all set to zero since the valves had just opened after isovolumetric contraction. On the other hand, the cavity pressure and rate of change of pressure were extracted from the end of the active contraction stage. In order to define the end of ejection, another stopping criterion was implemented whereby if the flow rate changed signs, the Windkessel algorithm was switched off.

Chapter 5

Reduced Order Method

5.1 Introduction

A *reduced order method*, or ROM, is a method of reducing the complexity of a well-defined problem. It allows the reduction of matrices or system of equations into smaller ones such that they are easier to solve, while still producing results of reasonable accuracy. In this research, the *Proper Orthogonal Decomposition with Interpolation*, or PODI, which forms part of the ROM family, was used. Since PODI is based on a well-known technique called the *Proper Orthogonal Decomposition*, POD, the latter is first introduced.

5.2 Proper Orthogonal Decomposition

Proper Orthogonal Decomposition consists of finding a finite number of subspaces, each corresponding to a dimension, that can be used for complexity reduction. Usually, those subspaces are formed from prior knowledge of the problem being solved [124]. This knowledge is represented by sets of data over which the ROM must be predictive [125].

The POD method was developed by several independent researchers. According to Holmes et al. [126], the first to have established the method were Kosambi [127], Loève [128], Karhunen [129], Pougachev [130] and Obukhov [131]. However, its first application was suggested by Lorenz, for weather prediction [132]. Since then, the POD method has gained great popularity due to its simplicity, flexibility, ease of implementation, ability to reduce the order of a system of equations, and its application to a broad range of field problems such as:

- **Image Processing**
  In [42], Everson and Sirovich exploited the POMs to recover missing data from the facial images of human beings. Moreover, they also attempted to use the POD basis to reconstruct the face of a monkey. By doing so, the authors managed to prove that the POMs are specific to the data set from which they have been computed, as the reconstituted face had a resemblance similar to human facial features.

- **Heat flow problem**

Falkiewicz and S. Cesnik showed in [43] that is possible to decouple their system of equation by making use of the dominant POMs when solving the heat transfer problem of a hypersonic vehicle through time.

- **Computational Fluid Dynamics(CFD)**
  In the CFD field, the POD method is well established, as it has been used quite extensively. For example, Amsallem et al. [44] set up a database of pre-computed POD-based reduced order information in order to achieve near-real-time simulation of aero-elastic prediction of an aircraft under certain operational conditions. In [45], Christensen et al. used two modified POD methods, called weighted POD (w-POD) and predefined POD (p-POD), to demonstrate how the choice of different POMs can affect the calculation of fluid flow with a different Reynolds number. Finally, Druault et al. employed the POD method to interpolate the time information between two consecutive particle image velocimetry measurements in a spark ignition combustion engine, to obtain a smooth change in their solution field [46].

- **Solid Mechanics**
  Several usages of the POD method in solid mechanics frameworks are found in the literature. Amongst the first is Georgiou and Sansour, who analysed the dynamics of non-linear in-plane rods [47]. Generally, employing of POD in vibration analysis of structures is quite common [48, 49, 50, 51]. It can help in understanding the different deformation modes that a structure undergoes before failing, or when under impact [52]. In [53], POD was employed to reduce the complexity of modelling a cylindrical shell structure by making use of a perturbation procedure. The same strategy was applied by Brigham and Aquino [54] to model the inverse viscoelastic material characterisation in acoustic-structure interaction.

Different strategies can be used to carry out the Proper orthogonal decomposition of sets of data. In this section, two of the most popular will be looked at [80]: the *Karhunen-Loève Decomposition*, KLD, and the *Singular Value Decomposition*, SVD.

5.2.1 Karhunen-Loève decomposition

The Karhunen-Loève decomposition, KLD, is one of the many ways of extracting linear subspaces from a given set of data. The advantage of using this method is its ability to define those subspaces *optimally* [133]. That is, they are the best subspace-representation of the sets of data and have mobilised the most energy of the system [125], from which they have been extracted, as opposed to other methods such as the Fourier series [80].

5.2.1.1 Mathematical derivation

The Karhunen-Loève decomposition methodology that is now introduced is adapted from [134] and [80]. Consider the presence of an ensemble set of displacement field vectors, $\mathbf{U} = \{\mathbf{u}_s\}$, obtained from Eq. (4.6) and defined over the domain Ω. Here, $\mathbf{u}_s$ is assumed to be the converged solution, e.g. for a certain time or loading step, s. The ensemble of N displacement fields can be stated over different quantity spaces. However, for this thesis, it is defined either over temporal or parametric spaces, since it is assumed that

only the material properties of cardiac tissue change across different hearts. Within the KLD context, any displacement solution field is defined in terms of a linear collection of vectorial subspaces, $\mathbf{\Phi} = \left\{\boldsymbol{\phi}_j\right\}$:

$$\mathbf{u}_s = \sum_{j=1}^{N} z_{js}\boldsymbol{\phi}_j, \tag{5.1}$$

where z_{js} are the coefficients which regulate the corresponding subspaces, $\boldsymbol{\phi}_j$. The subspaces, themselves, are chosen to be orthogonal:

$$(\boldsymbol{\phi}_a, \boldsymbol{\phi}_b) = \begin{cases} 1 & \text{for} \quad a = b \\ 0 & \text{for} \quad a \neq b, \end{cases} \tag{5.2}$$

where the inner product, $(\mathbf{f}, \mathbf{g}) = \int_0^1 \mathbf{f}^{\mathrm{T}}(x)\,\mathbf{g}^{\mathrm{T}}(x)\ dx$ is defined within the general Hilbert space, $\mathcal{H}$. In order to find $\mathbf{\Phi}$, an optimality scheme is formulated from Eq. (5.1) where a minimisation problem is obtained:

$$\min_{\boldsymbol{\phi}_j \in \mathcal{H}} \left\langle \left\| \mathbf{u}_s - \frac{\left(\mathbf{u}_s, \boldsymbol{\phi}_j\right)}{\left\|\boldsymbol{\phi}_j\right\|}\boldsymbol{\phi}_j \right\|^2 \right\rangle, \tag{5.3}$$

with $\|\mathbf{f}\| = (\mathbf{f}, \mathbf{f})^{\frac{1}{2}}$. The minimisation problem can also be re-cast into its equivalent maximisation problem which consists of an average projection of $\mathbf{u}_s$ onto $\boldsymbol{\phi}_j$ while imposing an orthogonality constraint [80]:

$$\begin{aligned} &\max_{\boldsymbol{\phi} \in \mathcal{H}} \left\langle \left| \left(\mathbf{u}_s, \boldsymbol{\phi}_j\right) \right|^2 \right\rangle \\ &\text{s.t.} \quad \|\boldsymbol{\phi}_j\|^2 = 1, \end{aligned} \tag{5.4}$$

with $|\cdot|$ being the absolute value and $\langle\cdot\rangle$, the averaging operation. To solve Eq. (5.4), the *Lagrange Multiplier method* is introduced and the Jacobian, J_{KLD} is obtained:

$$J_{\mathrm{KLD}} = \left\langle \left| \left(\mathbf{u}_s, \boldsymbol{\phi}_j\right) \right|^2 \right\rangle - \lambda_j \left(\|\boldsymbol{\phi}_j\|^2 - 1 \right), \tag{5.5}$$

where λ_j is the Lagrange multiplier. Following the variation of J_{KLD} as

$$\frac{d}{d\delta} J_{\mathrm{KLD}} \left[\boldsymbol{\phi}_j + \delta\boldsymbol{\psi}_j \right] \Big|_{\delta=0} = 0, \tag{5.6}$$

an eigenproblem can be consequently obtained:

$$\mathbf{L}\boldsymbol{\phi}_j = \lambda_j \boldsymbol{\phi}_j, \tag{5.7}$$

with

$$\mathbf{L} = \frac{1}{M} \sum_{s=1}^{M} \mathbf{u}_s \left(\mathbf{u}_s\right)^{\mathrm{T}} = \frac{1}{M}\mathbf{U}\mathbf{U}^{\mathrm{T}}. \tag{5.8}$$

In the literature, $\mathbf{L}$ is referred as the *autocorrelation matrix* [135] or *averaged two-points correlation function* [15]. In order to obtain $\mathbf{\Phi}$ and $\boldsymbol{\lambda}$, the eigenvectors and eigenvalues of the autocorrelation matrix need to be computed. Within the POD framework, the eigenvectors are called the *Proper Orthogonal Modes*, POMs, and the eigenvalues, the *Proper Orthogonal Values*, POVs. Since $\mathbf{L}$ is a compact, self-adjoint and non-negative definite matrix, a finite number of eigenvalues, λ_j and eigenvectors, $\boldsymbol{\phi}_j$ can be obtained [134]. The number of eigenvalues is equal to the rank of $\mathbf{L}$, while the number of displacement fields, M, dictates the number of POMs that can be obtained [51].

5.2.1.2 Snapshot Method

The eigen decomposition of the autocorrelation matrix, $\mathbf{L}$, can in certain cases be computationally expensive due to its size [136]. For example, consider that $\mathbf{U} \in \mathbb{R}^{M\times N}$ is defined by M-rows corresponding to degrees of freedom, $\mathbf{u}_s$, and N-columns, referring to a number of time steps. Therefore, through Eq. (5.8), $\mathbf{L}$ corresponds to a square matrix of size M, $\mathbf{L} \in \mathbb{R}^{M\times M}$. If M is large, then $\mathbf{L}$ results in a very large matrix and the computation of its eigenvalues and eigenvectors impose substantial computational resources demand. In most engineering modelling problems, the number of degrees of freedom is much greater than the number of time steps, $M \gg N$. As such, Sirovich and Kirby [137] proposed a more efficient way by redefining the construction of $\mathbf{L}$, known as the *snapshot method*.

The snapshot method derivation starts as follows. Let a POM be defined by a linear superposition of the ensemble displacement field $\{\mathbf{u}_s\}$ which is regulated by the coefficient basis, a_{js}, through:

$$\boldsymbol{\phi}_j = \sum_{s=1}^{M} a_{js}\mathbf{u}_s. \tag{5.9}$$

With Eqs. (5.9) and (5.8), Eq. (5.7) is rewritten as

$$\left(\frac{1}{M}\sum_{i=1}^{M}\mathbf{u}_i\,(\mathbf{u}_i)^{\mathrm{T}}\right)\sum_{s=1}^{M}a_{js}\mathbf{u}_s = \lambda_j\sum_{s=1}^{M}a_{js}\mathbf{u}_s. \tag{5.10}$$

After further simplifications, the following expression is obtained:

$$\sum_{s=1}^{M}\frac{1}{M}\,(\mathbf{u}_s,\mathbf{u}_i)\,a_{js} = \lambda_j a_{ji}, \tag{5.11}$$

which in vector-matrix notation is equivalent to

$$\mathbf{Y}\mathbf{a}_j = \lambda_j\mathbf{a}_j \quad \text{where} \quad \mathbf{Y} = \frac{1}{M}\mathbf{U}^{\mathrm{T}}\mathbf{U}. \tag{5.12}$$

From Eq. (5.12), $\mathbf{Y}$ is known as the *covariance matrix* [138] and is of size N, i.e. $\mathbf{Y} \in \mathbb{R}^{N\times N}$, representing a smaller problem, as opposed to $\mathbf{L}$, for an eigen decomposition. According to Sirovich and Kirby [137], the eigenvalues of $\mathbf{L}$ are the same as $\mathbf{Y}$, but not the eigenvectors. In order to recover the POMs, $\boldsymbol{\phi}_j$, the following equation was then suggested by Falkiewicz and S. Cesnik [43], Li et al. [139] and Burkardt

et al. [140]:

$$\boldsymbol{\phi}_j = \frac{1}{\sqrt{\lambda_j M}} \mathbf{U}\mathbf{a}_j. \tag{5.13}$$

5.2.2 Singular value decomposition

An alternative method to determine the subspaces or POMs and their respective POVs is by using the Singular Value Decomposition, SVD. The SVD is a method used in linear algebra to decompose a matrix into three different matrices, which are the left, right and singular values matrices. Since its introduction as a statistical tool, the method has found its usage being linked to different fields such as image processing and data compression [141]. The real breakthrough in the SVD method came in 1939 when Eckart and Young [142] managed to generalise its use for any non-symmetric matrix size (i.e. small, large, square, rectangular, singular, non-singular, sparse and dense [143]). In this sense, SVD has consequently been found appropriate to be applied to datasets given in the form of matrices.

5.2.2.1 Mathematical introduction

Consider the same displacement field ensemble, $\mathbf{U}$, defined in Section 5.2.1, where for a particular converged step, s, the displacement vector is $\mathbf{u}_s$, and

$$\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_s, ..., \mathbf{u}_N), \tag{5.14}$$

where N is the total number of time steps, $\mathbf{u}_s \in \mathbb{R}^M$, $\mathbf{U} \in \mathbb{R}^{M \times N}$ and M is the total number of degrees of freedom. The definition of the Singular Value decomposition can now be given as

$$\mathbf{U} = \boldsymbol{\Psi}\boldsymbol{\Sigma}\boldsymbol{\Gamma}^{\mathrm{T}}, \tag{5.15}$$

where $\boldsymbol{\Psi} \in \mathbb{R}^{M \times M}$ and $\boldsymbol{\Gamma} \in \mathbb{R}^{N \times N}$ represent an orthonormal matrix with a set of left singular vectors, which are the POMs, $\boldsymbol{\Psi} = (\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_M)$ and an orthonormal matrix containing the right singular vectors, $\boldsymbol{\Gamma} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_M)$, respectively. $\boldsymbol{\Sigma}$ is a rectangular matrix, of size $M \times N$, with the *singular values*, ϑ_i, defined as:

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\vartheta} & 0 \\ 0 & 0 \end{pmatrix}, \tag{5.16}$$

with $\boldsymbol{\vartheta} = \mathrm{diag}\,(\vartheta_1, \ldots, \vartheta_r)$ and r is the rank of $\mathbf{U}$. The singular values are computed from the eigenvalues of $\mathbf{U}\mathbf{U}^{\mathrm{T}}$, φ_i, through:

$$\vartheta_i = +\sqrt{\varphi_i}. \tag{5.17}$$

Since $\mathbf{U}\mathbf{U}^{\mathrm{T}} \in \mathbb{R}^{M \times M}$ is a positive semi-definite matrix, ϑ_i are always non-negative and greater than zero. The singular values obtained from $\boldsymbol{\Sigma}$ are always sorted in decreasing order of magnitude as:

$$\vartheta_1 \geqslant \vartheta_2 \geqslant \cdots \geqslant \vartheta_r > 0. \tag{5.18}$$

According to Kerschen et al. [144], the POVs can be found from the SVD through:

$$\lambda_i = \frac{(\vartheta_i)^2}{N}. \tag{5.19}$$

The POMs and POVs can be also represented geometrically [145] using the SVD. Understanding such an aspect can provide a deeper insight into the meaning of the POMs and POVs. Using the work of Phillips [146], consider a circle being transformed by matrix **P** to an ellipse, as shown in the diagram below.

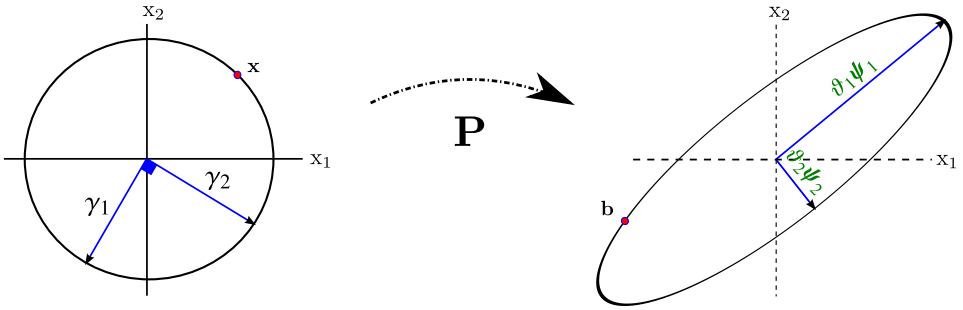


Figure 5.1: Direct transformation using **P** matrix.

For any arbitrary point **x** lying on the circle, it can be related to **b** as:

$$\mathbf{b} = \mathbf{P}\mathbf{x}. \tag{5.20}$$

If **P** is decomposed using the SVD method, Eq. (5.20) results in:

$$\mathbf{P} = \mathbf{\Psi}\mathbf{\Sigma}\mathbf{\Gamma}^{\mathrm{T}}. \tag{5.21}$$

The transformation of point **x** into point **b** described by **P** can be split into three steps:

- **Step 1**

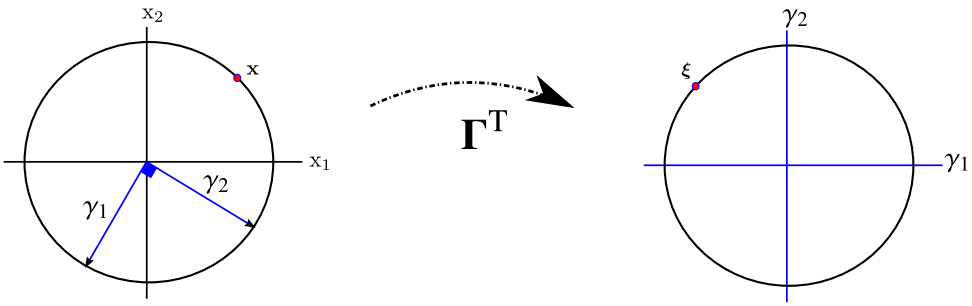


Figure 5.2: Transformation using $\mathbf{\Gamma}^{\mathrm{T}}$ matrix.

The application of matrix $\mathbf{\Gamma}$ results in an orthogonal coordinate transformation $\boldsymbol{\xi} = \mathbf{\Gamma}^{\mathrm{T}}\mathbf{x}$ where γ_i represents the corresponding coordinate frame.

- **Step 2**

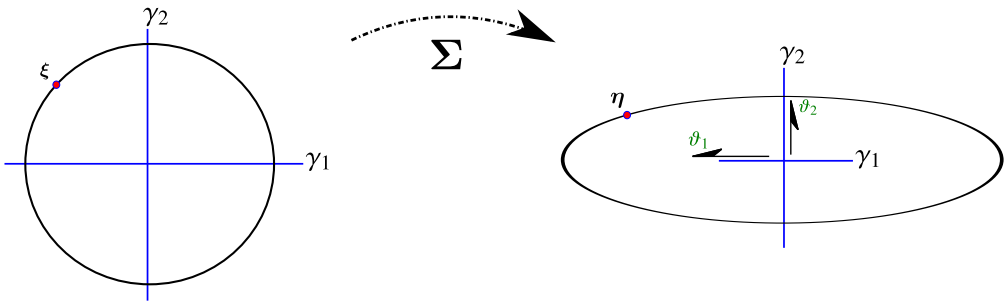


Figure 5.3: Transformation using $\boldsymbol{\Sigma}$ matrix.

The subsequent application of $\boldsymbol{\Sigma}$ scales each orthogonal coordinate axis, $\boldsymbol{\gamma}_i$, by the corresponding singular value, giving $\boldsymbol{\xi} = \boldsymbol{\Sigma}\boldsymbol{\eta}$.

- **Step 3**

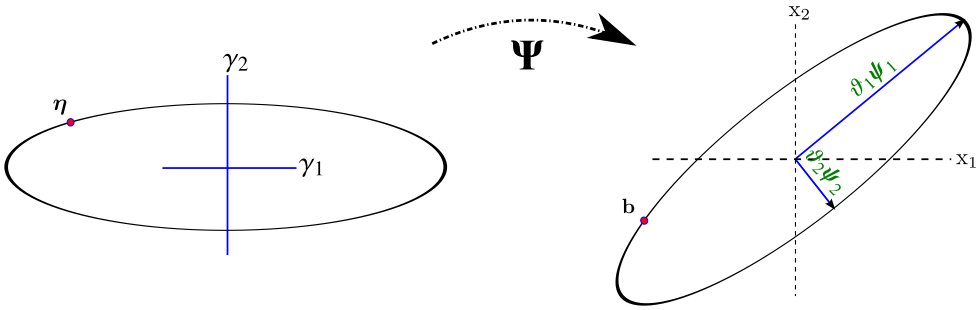


Figure 5.4: Transformation using $\mathbf{U}$ matrix.

Finally, the application of transformation $\boldsymbol{\Psi}$ causes either a rotatory or flip effect, thus providing one with $\mathbf{b} = \boldsymbol{\Psi}\boldsymbol{\eta}$.

The above geometrical interpretation visually defines the roles of the POMs and POVs. This can be elaborated on by looking at the analogy given in Shlens [147]. If the circle is replaced by a set of points, then the POMs will define the structural layout of those points in the form of a vectorial direction, which is referred to as a *component*. Across the set of points, a finite number of components can be obtained. To distinguish which direction is more dominant, the POVs are used to scale those structural directions around the dominant points distribution. The larger the value of the POV, the more dominant will be the component. Hence, as can be seen in Fig. 5.4, the component $\boldsymbol{\psi}_1$ is the most important, as it stretches over a larger distance due to its corresponding scaling value ϑ_1.

5.2.2.2 SVD calculation

The calculation of the singular value decomposition consists of multiplying $\mathbf{U}^{\mathrm{T}}\mathbf{U}$ by $\mathbf{\Gamma}^{\mathrm{T}}$ before expressing the resulting equation as follows [148, 149]:

$$\mathbf{U}^{\mathrm{T}}\mathbf{U}\mathbf{\Gamma}^{\mathrm{T}} = \mathbf{\Lambda}\mathbf{\Gamma}^{\mathrm{T}}, \tag{5.22}$$

where using Eq. (5.16), $\mathbf{\Lambda}$ is can be present in the form of

$$\mathbf{\Lambda} = \begin{pmatrix} \boldsymbol{\vartheta}^2 & 0 \\ 0 & 0 \end{pmatrix}, \tag{5.23}$$

with $\boldsymbol{\vartheta}^2$ being a diagonal matrix with entries $\vartheta_1^2 \geqslant \vartheta_2^2 \geqslant \cdots \geqslant \vartheta_r^2 > 0$. From Eq. (5.22), $\mathbf{\Gamma}$ represents the eigenvectors of $\mathbf{U}^{\mathrm{T}}\mathbf{U}$ and the diagonals of $\mathbf{\Lambda}$ are its corresponding eigenvalues. In order to find the $\mathbf{U}$ matrix, i.e. the left singular vectors, the following has to be carried out. Let $\mathbf{\Psi} = \left[\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_j\right]$ and $\mathbf{\Gamma} = \left[\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_j\right]$, then

$$\mathbf{U} = \mathbf{\Psi}\mathbf{\Sigma}\mathbf{\Gamma}^{\mathrm{T}} \tag{5.24}$$

$$= \begin{bmatrix} \boldsymbol{\psi}_1 & \boldsymbol{\psi}_2 & \cdots & \boldsymbol{\psi}_j \end{bmatrix} \begin{bmatrix} \vartheta_1 & & & \\ & \vartheta_2 & & \\ & & \ddots & \\ & & & \vartheta_j \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma}_1^{\mathrm{T}} \\ \boldsymbol{\gamma}_2^{\mathrm{T}} \\ \vdots \\ \boldsymbol{\gamma}_j^{\mathrm{T}} \end{bmatrix} \tag{5.25}$$

$$\mathbf{U} = \sum_j \boldsymbol{\psi}_j \vartheta_j \boldsymbol{\gamma}_j^{\mathrm{T}} \tag{5.26}$$

$$\therefore, \quad \boldsymbol{\psi}_j = \frac{1}{\sigma_j}\mathbf{U}\boldsymbol{\gamma}_j. \tag{5.27}$$

An alternative way of carrying out the SVD of $\mathbf{X}$ is to first find the $\mathbf{\Psi}$ matrix. This can be done by computing the eigenvectors of $\mathbf{U}\mathbf{U}^{\mathrm{T}}$, since $\mathbf{U}\mathbf{U}^{\mathrm{T}} = \mathbf{\Psi}\mathbf{\Lambda}\mathbf{\Psi}^{\mathrm{T}}$. From there, the eigenvalues will represent the singular values and $\mathbf{\Gamma}$ can be calculated as follows:

$$\boldsymbol{\gamma}_j = \frac{1}{\vartheta_j}\mathbf{U}^{\mathrm{T}}\boldsymbol{\psi}_j. \tag{5.28}$$

5.2.2.3 SVD and KLD equivalence

The Proper Orthogonal values and modes obtained from the KLD, snapshot method, or SVD are equivalent. Throughout the literature, many researchers have made a similar observation. For example, in [136] and [150], the authors showed how SVD can be interpreted as a POD method, while Wu et al. [151] and Liang et al. [142] mathematically proved how KLD and SVD solve the same eigenproblem and have an asymptotic connection through a so-called *covariance* matrix.

The first step to prove this equivalence is to start with the eigenproblem defined from section 5.2.2.2. As such, the eigenproblem, which is used to compute the eigenvectors, $\mathbf{\Psi}$, is presented as:

$$\mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{\Psi}^{\mathrm{T}} = \mathbf{\Lambda}\mathbf{\Psi}^{\mathrm{T}}. \tag{5.29}$$

Pre-multiplication of Eq. (5.29) with $\frac{1}{N}$ gives

$$\frac{1}{N}\mathbf{U}\mathbf{U}^{\mathrm{T}}\mathbf{\Psi}^{\mathrm{T}} = \frac{1}{N}\mathbf{\Lambda}\mathbf{\Psi}^{\mathrm{T}}. \tag{5.30}$$

If Eq. (5.30) is stated for each of its eigenvalues and corresponding eigenvectors, then

$$\frac{1}{N}\mathbf{U}\mathbf{U}^{\mathrm{T}}\boldsymbol{\psi}_i = \frac{1}{N}\vartheta_i^2\boldsymbol{\psi}_i. \tag{5.31}$$

The expression $\frac{1}{N}\mathbf{U}\mathbf{U}^{\mathrm{T}}$ is similar to the autocorrelation matrix given in Eq. (5.8), leading to the eigenproblem presented in Eq. (5.31) to correspond to the one given in Eq. (5.7). Therefore, the Proper Orthogonal Modes from the SVD are directly given by $\mathbf{\Psi}$ while the Proper Orthogonal Values are computed from the singular values:

$$\lambda_i = \frac{1}{N}\vartheta_i^2. \tag{5.32}$$

The equivalence between the SVD and snapshot method can also be shown. To do so, the eigenproblem given in Eq. (5.22) is also first pre-multiplied by $\frac{1}{N}$ to obtain

$$\frac{1}{N}\mathbf{U}^{\mathrm{T}}\mathbf{U}\mathbf{\Gamma}^{\mathrm{T}} = \frac{1}{N}\mathbf{\Lambda}\mathbf{\Gamma}^{\mathrm{T}}, \tag{5.33}$$

which can also be written as

$$\frac{1}{N}\mathbf{U}^{\mathrm{T}}\mathbf{U}\boldsymbol{\gamma}_i = \frac{1}{N}\vartheta_i^2\boldsymbol{\gamma}_i. \tag{5.34}$$

Eq. (5.34) is in this case the same as with $\frac{1}{N}\mathbf{U}^{\mathrm{T}}\mathbf{U}$ represents the covariance matrix (Eq. (5.12)) and $\mathbf{\Gamma}$ its eigenvectors. The eigenvalues of the covariance matrix are the Proper Orthogonal Values and are computed by Eq. (5.32). To recover the Proper Orthogonal modes as described in the snapshot method, Eq. (5.32) is reformulated as:

$$\vartheta_i = \sqrt{N\lambda_i}. \tag{5.35}$$

and then substituted in Eq. (5.27), giving

$$\boldsymbol{\psi}_i = \frac{1}{\sqrt{N\lambda_i}}\mathbf{U}\boldsymbol{\gamma}_i. \tag{5.36}$$

Eq. (5.36) is similar to Eq. (5.13) and therefore it is proven that $\boldsymbol{\psi}_i$ are the Proper Orthogonal Modes.

5.2.3 Ensemble matrix

In solid mechanics, the sets of data, $\mathbf{U}$, are equivalent to solution fields generated from simulations with almost similar geometry, boundary conditions and, most importantly, physical problem setup. This fact is crucial as it ensures that the behaviour of the datasets is governed by the same physical behaviour. $\mathbf{U}$, which is also known as ensemble solution fields, consists of temporal or parametric discretised solution fields, $\mathbf{u}_i$. As such, it can be built up in two ways. Firstly, one can consider multiple datasets arising

from different parameter values whose solution fields are assembled together for one particular time step. Hence, in this case, i will refer to a parameteric space and the POMs extracted from $\mathbf{U}$ will describe variation across the discretised parameter space. For the second definition of $\mathbf{U}$, the solution fields of a single dataset for a series of sequential time steps, which correspond to a specific parameter setting, is used. Consequently, i will be part of a discretised time space and the POMs will describe the behaviour of the data across the temporal space. The solution fields can be any continuum mechanical quantity such as displacement, stress or strain [29, 152, 153].

The selection of datasets plays an important role in ROMs. They are crucial with respect to the quality of the subspaces which directly affect the accuracy of the ROM. Usually, highly detailed sets of data are favoured to capture as much information as possible. Creating those datasets requires certain perturbations in their respective problem definitions so that they have an adequate degree of differences. These differences can therefore be used to capture the problem behaviour through the subspaces. The perturbation, on the other hand, can be defined in a variety of ways. For example, it can be changes in material parameters, location, and intensity of boundary constraints, and also in the duration/time steps of the simulations. As such, two ways can be used to obtain high fidelity datasets. The first is to consider highly discretised geometry, where the element size is very small; and the second is to decrease the size of the perturbation.

Across the literature, one finds two types of definitions for $\mathbf{U}$, namely the *non-zero mean* and *zero mean* methods. In the non-zero mean method, the obtained solutions field is directly assembled in the $\mathbf{U}$ matrix. This has been observed in the work of Niroomandi et al. [29], Ahlman et al. [154], Rapún and Vega [155] and more recently, in Xie et al. [156]. But in the case of the zero-mean method, $\mathbf{U}$ need to first undergo a pre-processing stage before being used for any ROM calculation. This pre-processing stage consists of computing the column-mean of $\mathbf{U}$, defined as $\bar{\mathbf{u}}$, and subtracting $\bar{\mathbf{u}}$ from each column of $\mathbf{U}$ to form $\bar{\mathbf{U}}$:

$$\bar{\mathbf{U}} = \mathbf{U} - \bar{\mathbf{u}}. \tag{5.37}$$

This procedure is undertaken in the work of [1, 137, 157, 158]. There, the new ensemble, $\bar{\mathbf{U}}$, will then be further used to extract the POMs and POVs. As noted by Tamura et al. [159], these are different to those obtained from the non-zero mean method. One of the main reasons for removing the mean from the ensemble is to make sure that the POD calculations will focus only on the perturbations of the data [157]. According to Tamura et al. [159] and Chatterjee [157], this effect can be properly explained by looking at the visual representation of the POMs and the corresponding data. Consider a 2D axis plot where the data are represented by a point cloud clustered in an elliptical shape away from the origin as shown in Fig. 5.5. For such a setup, the first POM, which is represented as a vector, starts from the origin, similar to the one given in Fig. 5.4, and goes through the centre of gravity of the clustered point set [160]. Yet, when the mean is extracted from the data, the point cloud shifts to the origin where it is now centred, as shown in Fig. 5.6. This new position leads to a new dominant POM, which is completely different to the non-zero mean one. Finding which of the two procedures leads to a better solution is still in open discussion, Honeine [161], as both methods are still currently used. Hence, this research also investigated the effect of the non-zero-mean and zero-mean methods within the cardiac modelling framework, in order to ascertain which is more suitable.

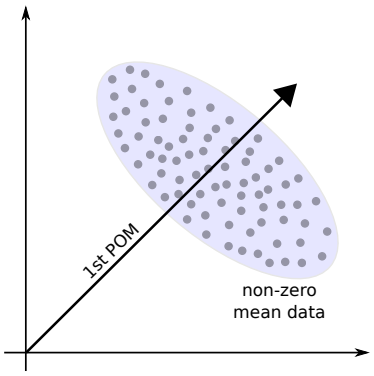


Figure 5.5: Cluster of non-centred data points with its respective dominant POM.

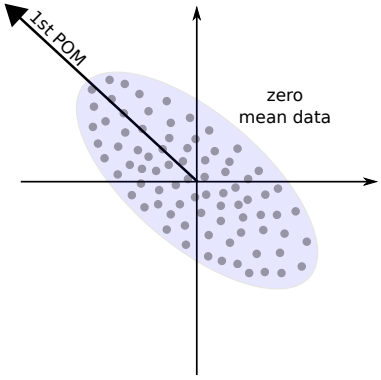


Figure 5.6: Cluster of centred data points with its respective dominant POM.

5.2.4 Application of POD to mechanics

In this section, the properties of the Proper Orthogonal values and modes are looked at. Understanding their characteristics is of prime importance in order to reduce a problem size without drastic loss of accuracy in the solution fields. This aspect was crucial when modelling the heart in an accurate manner within a restrictive time frame.

As mentioned previously, the eigenvectors obtained from Eq. (5.7), Eq. (5.13) and Eq. (5.15) are known as the Proper Orthogonal Modes or basis functions. They are orthogonal spaces that define a series of superimposed linear functions or subspaces of the data, $\mathbf{U}$, from which they have been extracted [80]. Those data, on the other hand, can be either linear or non-linear in nature. The POMs, obtained from POD, are optimal as opposed to those obtained from other methods such as the Fourier series [80], and are consequently the most representative states of $\mathbf{U}$. In the case where POMs are constructed from $\mathbf{U}$, which is a collection of time-consecutive displacement degree of freedom, $\mathbf{u}_i$, then those POMs are time-independent orthonormal [162].

As indicated by the SVD method, the associated POVs are non-negative. Each POV is usually associated to a fraction of the energy in the system and as such, the sum of all the POVs gives the total energy in the system [134]. Since, the POVs are also relatively of different magnitude, they are said to be an indication of the importance of their corresponding POMs. Therefore, the higher the POV, the more important or dominant the POM.

The Proper Orthogonal Decomposition method can be used for the purpose of a Reduced-Order Method (ROM). It can simplify the solving of a complex spatio-temporal dynamic system with a large number of equations without altering its dominant behaviour. POD makes use of the Proper Orthogonal Modes to project a Galerkin system of equation onto a low dimensional space, whereby the solving takes place [163]. Once done, the POMs are then used again to project back the solution to the high dimensional space in order to recover the nodal displacement solutions.

Consider a data matrix, $\mathbf{U} \in \mathbb{R}^{M \times N}$, of M degrees of freedom and N time/calculation steps or snapshots, which has been decomposed by SVD using Eq. (5.15). With $\mathbf{\Psi}$ representing the POMs, then $\mathbf{\Phi} = \mathbf{\Psi}$ and the projection into the low-dimensional space of the displacement fields, $\mathbf{U}$, can be defined by

the following relationship

$$\mathbf{U} = \mathbf{\Phi Z}, \tag{5.38}$$

such that $\mathbf{\Phi} = (\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_M)$, $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N)$ and the coefficient $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N)$. Now, if the Finite Element equation of motion is defined, then

$$\mathbf{M\ddot{U}} + \mathbf{C\dot{U}} + \mathbf{KU} = \mathbf{F}, \tag{5.39}$$

with $\mathbf{M}$ being the mass matrix, $\mathbf{C}$, the damping matrix, $\mathbf{K}$, the stiffness matrix and $\mathbf{F}$, a dynamic force matrix. $\ddot{\mathbf{U}}$ and $\dot{\mathbf{U}}$ are the acceleration and the velocity vector. Substituting Eq. (5.38) into Eq. (5.39) and pre-multiplying by $\mathbf{\Phi}^{\mathrm{T}}$, yields

$$\mathbf{\Phi}^{\mathrm{T}}\mathbf{M\Phi\ddot{Z}} + \mathbf{\Phi}^{\mathrm{T}}\mathbf{C\Phi\dot{Z}} + \mathbf{\Phi}^{\mathrm{T}}\mathbf{K\Phi Z} = \mathbf{\Phi}^{\mathrm{T}}\mathbf{F}, \tag{5.40}$$

which forms a new system of equation in the subspaces of $\mathbf{\Phi}$. In order to get back the solution, $\mathbf{U}$, then Eq. (5.38) is applied. To reduce the number of equations in a controlled manner, $\boldsymbol{\phi}_j$ can be selected individually. Suppose that the reduced $\mathbf{\Phi}$ is denoted by $\tilde{\mathbf{\Phi}} \in \mathbb{R}^{M\times r}$, where $1 \leq r < M$. Then, Eq. (5.38) is now posed as

$$\underset{(M\times N)}{\mathbf{U}} \approx \underset{(M\times r)}{\tilde{\mathbf{\Phi}}}\,\underset{(r\times N)}{\tilde{\mathbf{Z}}}, \tag{5.41}$$

with $\tilde{\mathbf{Z}}$ being consequently reduced from $\mathbb{R}^{M\times N}$ to $\mathbb{R}^{r\times N}$. Reformulating Eq. (5.40), one finally gets

$$\underset{(r\times r)(r\times N)}{\tilde{\mathbf{M}}}\,\underset{}{\ddot{\tilde{\mathbf{Z}}}} + \underset{(r\times r)(r\times N)}{\tilde{\mathbf{C}}}\,\underset{}{\dot{\tilde{\mathbf{Z}}}} + \underset{(r\times r)(r\times N)}{\tilde{\mathbf{K}}}\,\underset{}{\tilde{\mathbf{Z}}} = \underset{(r\times N)}{\tilde{\mathbf{F}}}, \tag{5.42}$$

where

$$\tilde{\mathbf{M}} = \underset{(r\times M)(M\times M)(M\times r)}{\tilde{\mathbf{\Phi}}^{\mathrm{T}}\,\mathbf{M}\,\tilde{\mathbf{\Phi}}}, \qquad \tilde{\mathbf{C}} = \underset{(r\times M)(M\times M)(M\times r)}{\tilde{\mathbf{\Phi}}^{\mathrm{T}}\,\mathbf{C}\,\tilde{\mathbf{\Phi}}},$$
$$\tilde{\mathbf{K}} = \underset{(r\times M)(M\times M)(M\times r)}{\tilde{\mathbf{\Phi}}^{\mathrm{T}}\,\mathbf{K}\,\tilde{\mathbf{\Phi}}}, \qquad \tilde{\mathbf{F}} = \underset{(r\times N)(M\times N)}{\tilde{\mathbf{\Phi}}^{\mathrm{T}}\,\mathbf{F}}.$$

Hence, the new system of equations that needs to be solved is now smaller and therefore requires less computational resource and time. Regarding Eq. (5.41), the approximately equal sign is used since the disregarded POMs represent a loss of information. But, in order to minimise the loss, the POMs have to be carefully selected. According to Kerschen et al. [144], the Proper Orthogonal Values can be used as a guideline to do so. In [164], Barbic and James show that for each POM, there is a corresponding POV, which is represented by the eigenvalue obtained from the SVD and KLD method. Usually, the POM represents a modal mode which characterises a specific deformation of the domain, while the corresponding POV is attributed to the captured energy from that particular deformation [144]. In order

to find the total energy discarded, Liang et al. [142] produced the following equation:

$$\varepsilon_{\text{loss}} = \sum_{i=1+r}^{M} \lambda_i, \tag{5.43}$$

which defines that the mean square error, $\varepsilon_{\text{loss}}$, is the sum of the POVs whose POMs, within the range of r to M, have been discarded. In that sense, in order to find a proper balance between selecting a few POMs and conserving at the same time the largest possible amount of energy, the POD basis associated with the highest POVs is selected, as the former represents the dominant modes of deformation. Interestingly, it was found that with very few modes one can easily conserve approximately more than 99 % of the energy in the system. This is because if, for example, one looks at the change in magnitude of the POVs (sorted in descending order) against the number of POVs as shown in Fig. 5.7, then it can be observed that the first few POVs represent the largest amount of energy. In [15], Lin et al. noted that, with their set of data comprising 24 snapshots, they needed only the first two POVs to achieve an energy conservation of 99.9997 %.

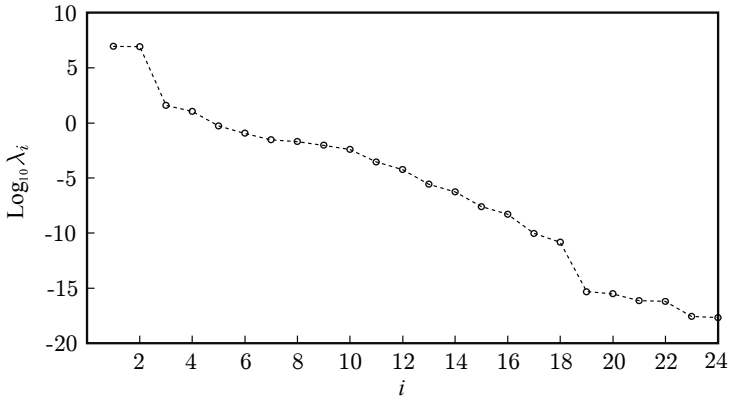


Figure 5.7: Plot of Magnitude of POVs with number of POVs [15].

With the application and choice of the POMs introduced, an example is used to illustrate the POD-based ROM in dynamics in the next section.

5.2.5 Strut example

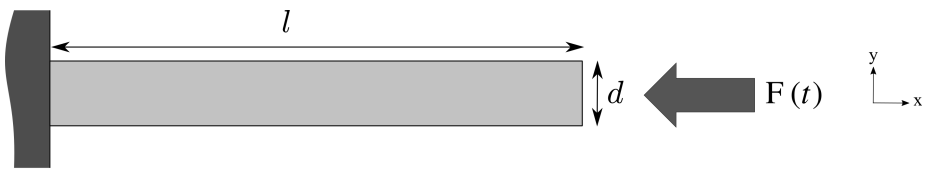


Figure 5.8: 2D strut beam subjected to a dynamic load.

The problem investigated in this section is a 2D strut beam subjected to a dynamic load, as shown in Fig. 5.8. The beam is 1 m in length and 0.1 m in width. Following the discretisation of the strut domain, 2211 nodes and 2000 quadrilateral elements were obtained. With each node allowed to move axially

(x-direction) and transversely (y-direction), the total number of degrees of freedom defined for the system is 4422. As Neumann boundary condition, the free end of the strut is subjected to a dynamic force that takes the following form:

$$\mathrm{F}(t) = 5\left[\cos\left(\frac{2\pi}{10\times10^{-6}}t+\pi\right)+1\right]. \tag{5.44}$$

The trigonometrical dynamic force represents a wavefront axial surface traction, $\mathrm{F}(t)$. Imposition of the Dirichlet boundary condition on the fixed end, as shown in Fig. 5.8, consists in preventing the nodes from moving along the x and y directions, i.e. $u_x = u_y = 0$. The material properties of the beam are defined by a density of $8000\,\mathrm{kg\,m^{-3}}$ and a Poisson ratio of 0.3, which will remain constant for all calculations. The Young's modulus is set to 200 GPa. The problem is solved using a MATLAB finite element code through Eq. (5.39), and the displacement solution fields are stored off-line in a storage medium. The calculation is run from 0 s to 800×10^{-6} s simulation time to allow the wavefront axial surface traction to travel forth and back at least once. The time steps are the same for all simulations and are chosen in such a way that the *Courant-Friedrichs-Lewy* condition is satisfied. Consequently, a total of 4822 time steps are needed. Then, for different energy levels (91 %, 96 % and 99 %), the POD method is employed and the results are plotted. Finally, to quantify the quality of the PODI results, the L_2-error norm of the PODI solution, $\mathcal{U}^{\mathrm{PODI}}$, is compared to the FEM one, which is defined as the exact solution, $\mathcal{U}^{\mathrm{EXACT}}$, through the following equation:

$$L_2\text{-error} = \frac{\left\|\mathcal{U}^{\mathrm{PODI}} - \mathcal{U}^{\mathrm{EXACT}}\right\|}{\left\|\mathcal{U}^{\mathrm{EXACT}}\right\|}, \tag{5.45}$$

Figs. 5.9, 5.10 and 5.11 compare the reduced model (blue line) with respect to the actual full FE model (red line) of the change in displacement over time at a point located in the middle of the beam. The number of equations in the FE model were subsequently decreased from the 4422 to 11 in model C, 3 in model B, and 1 in model A. The reduction, however, brought in some errors, as part of the energy from the complete system was not accounted for in the reduced model. The errors computed were 27.0 % in model A, 22.4 % in model B and 9.6 % in model C. Nevertheless, the reduced systems were solved at a faster rate, about 19 times quicker, while still capturing the main features of the full solution.

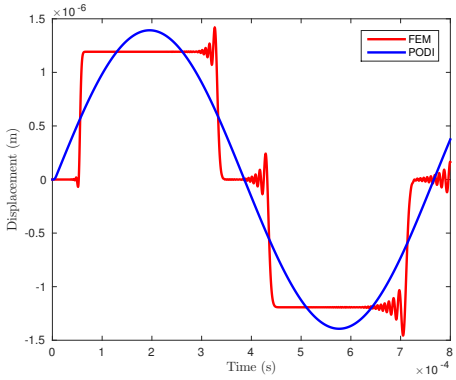


Figure 5.9: Reduced model A at 90% energy conserved.

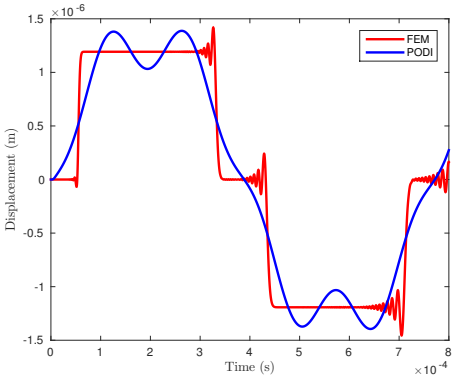


Figure 5.10: Reduced model B at 95% energy conserved.

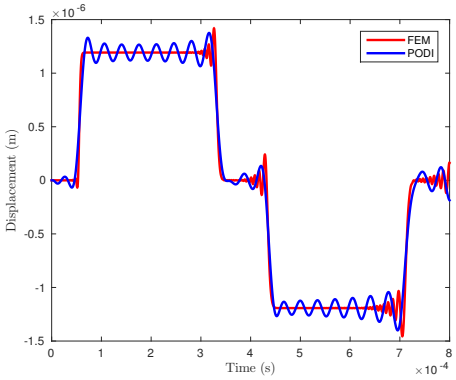


Figure 5.11: Reduced model C at 99% energy conserved.

5.3 Proper Orthogonal Decomposition with Interpolation

5.3.1 Basics

In section 5.2, the Proper Orthogonal values and modes were extracted from a data matrix which is considered to be obtained from experimental data or simulations. However, these data corresponded to a specific set of conditions from which they were calculated, therefore limiting the use of the POVs and POMs for these instances only. But in computational modelling, there is always a need for simulations to be carried out for a wide range of conditions (for example, different material properties, or different applied forces of different load intensities or locations) in order to find the optimum solution to a problem. In this case, the data matrix was not available for each scenario and therefore made the extraction of the POD basis impossible.

One solution to this problem was proposed by Ly and Tran in [1] and is known as the *Proper Orthogonal Decomposition with Interpolation* method (PODI). The basic idea is that the displacement field of an unknown problem is not computed by means of solving a mechanics equation system, but by employing an interpolation scheme based on a limited number of pre-existing displacement field solutions

which share the same problem characteristics. Each of those datasets $\mathbf{u}^a$, where $\mathbf{u}^a \in \left\{\mathbf{u}^1, \ldots, \mathbf{u}^p\right\}$, corresponds to a set of parameters, $\boldsymbol{\theta}^a$, where $\boldsymbol{\theta}^a \in \left\{\boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^p\right\}$. Here, $\left\{\boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^p\right\}$ is defined as a sequence of parameter sets. As such, an ensemble of displacement field vectors $\left\{\mathbf{u}^1, \ldots, \mathbf{u}^p\right\}$ corresponds to the matrix $\mathbf{U}$.

From there, the POD calculation is carried out and used to transfer the datasets to a low-dimensional space using the POMs, where an interpolation technique is employed to approximate the solution of a similar problem in the low-dimensional space. Once this is done, this interpolated solution is projected back to the high-dimensional space. The derivation of the method is given by first applying Eq. (5.41) to each dataset $\mathbf{u}^a$:

$$\mathbf{u}^a \approx z_1^a \boldsymbol{\Phi}^1 + z_2^a \boldsymbol{\Phi}^2 + \cdots + z_r^a \boldsymbol{\Phi}^r, \tag{5.46}$$

or in matrix form expressed as

$$\begin{bmatrix} u_1^a \\ u_2^a \\ \vdots \\ u_{m-1}^a \\ u_m^a \end{bmatrix} \approx z_1^a \begin{bmatrix} \phi_1^1 \\ \phi_2^1 \\ \vdots \\ \phi_{m-1}^1 \\ \phi_m^1 \end{bmatrix} + z_2^a \begin{bmatrix} \phi_1^2 \\ \phi_2^2 \\ \vdots \\ \phi_{m-1}^2 \\ \phi_m^2 \end{bmatrix} + \cdots + z_r^a \begin{bmatrix} \phi_1^r \\ \phi_2^r \\ \vdots \\ \phi_{m-1}^r \\ \phi_m^r \end{bmatrix}. \tag{5.47}$$

Collating all datasets approximated by Eq. (5.46), we arrive at Eq. (5.41), which is expanded in matrix form as

$$\begin{bmatrix} u_1^1 & u_1^2 & \cdots & u_1^{p-1} & u_1^p \\ u_2^1 & u_2^2 & & & u_2^p \\ \vdots & & \ddots & & \vdots \\ u_{m-1}^1 & & & \ddots & u_{m-1}^p \\ u_m^1 & u_m^2 & \cdots & u_m^{p-1} & u_m^p \end{bmatrix} \approx \begin{bmatrix} \phi_1^1 & \phi_1^2 & \cdots & \phi_1^r \\ \phi_2^1 & \phi_2^2 & \cdots & \phi_2^r \\ \vdots & \vdots & \cdots & \vdots \\ \phi_{m-1}^1 & \phi_{m-1}^2 & \ddots & \phi_{m-1}^r \\ \phi_m^1 & \phi_m^2 & \cdots & \phi_m^r \end{bmatrix} \begin{bmatrix} z_1^1 & z_1^2 & \cdots & z_1^p \\ z_2^1 & z_2^2 & \cdots & z_2^p \\ \vdots & \vdots & \cdots & \vdots \\ z_{r-1}^1 & z_{r-1}^2 & \ddots & z_{r-1}^p \\ z_r^1 & z_r^2 & \cdots & z_r^p \end{bmatrix}. \tag{5.48}$$

Using Eq. (5.48), then the coefficients, $\tilde{\mathbf{Z}} = \left\{\tilde{\mathbf{z}}^1, \ldots, \tilde{\mathbf{z}}^p\right\}$, can be found as follows:

$$\tilde{\mathbf{Z}} = \tilde{\boldsymbol{\Phi}}^{\mathrm{T}} \mathbf{U}. \tag{5.49}$$

Now, let us suppose that an unknown displacement field $\hat{\mathbf{u}}$, with $\hat{\mathbf{u}} \notin \left\{\mathbf{u}^1, \ldots, \mathbf{u}^p\right\}$, corresponds to the set of parameters $\hat{\boldsymbol{\theta}}$, with $\boldsymbol{\theta}^1 < \hat{\boldsymbol{\theta}} \leq \boldsymbol{\theta}^p$ and $\hat{\boldsymbol{\theta}} \notin \left\{\boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^p\right\}$. Similarly, as done in Eq. (5.46), $\hat{\mathbf{u}}$ is found using the above POD-basis, and is therefore approximated via Eq. (5.41) as

$$\hat{\mathbf{u}} \approx \hat{z}_1 \boldsymbol{\phi}^1 + \hat{z}_2 \boldsymbol{\phi}^2 + \cdots + \hat{z}_r \boldsymbol{\phi}^r \tag{5.50}$$

$$\approx \tilde{\boldsymbol{\Phi}} \hat{\mathbf{z}}. \tag{5.51}$$

However, the coefficients $\hat{\mathbf{z}} = \{\hat{z}_1, \hat{z}_2, \ldots, \hat{z}_r, \}$ are unknown and cannot be directly determined. Ly and

Tran therefore suggested to interpolate $\hat{\mathbf{z}}$ from matrix $\tilde{\mathbf{Z}}$ through

$$\hat{\mathbf{z}} = \tilde{\mathbf{Z}}\mathbf{N}, \tag{5.52}$$

where $\mathbf{N}$ are the interpolants that can be derived from

$$\hat{\boldsymbol{\theta}} = \boldsymbol{\Theta}\mathbf{N}, \tag{5.53}$$

where matrix $\boldsymbol{\Theta} = \left\{\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \ldots, \boldsymbol{\theta}^p\right\}$ contains as column vectors the different sets of parameters corresponding to the known solutions.

Different kinds of interpolation techniques can be used to find $\mathbf{N}$. In [1] and [165], Ly and Tran and Tan et al. used cubic spline interpolants, while in [30], Niroomandi et al. found that a piece-wise linear interpolation was better suited, compared to using a geometrical approach based on the Grassman manifold (as suggested in [44]). Alonso et al. [166] made use of a bivariate interpolation method of scattered data obtained from [167] and Druault et al. [46] employed a spline polynomial. Nevertheless, no research has been done investigating which is more suitable for coefficient interpolations and, as noted by Tan et al. [165] and Burkardt et al. [140], no indication of the smoothness of the coefficients has been discussed so far. The smoothness of the coefficients, along with the linearity properties of the POD, are research efforts that can provide a greater clarity to the PODI method through a rigorous mathematical study. However, this work does not intend to carry out such an investigation, as the numerical application considered in this thesis can provide, on its own, an insight into the suitability of PODI to highly non-linear problems.

On a final note, the PODI technique is not only restricted to interpolate sets of simulations with different parameter values. It could also be applied to interpolate between time steps, that is, creating a smooth transition of the displacement field through time [46]. This can be done by replacing the columns of the data matrix $\mathbf{U}$ with the displacement vector for different intervals in time. The time steps are then used to calculate the interpolants.

5.3.2 Moving Least Square Approximation method

As mentioned previously, the PODI method is based on the use of an interpolation technique. In this research, the Moving Least Square Approximation method (MLS) [120] was chosen, as it can deal with problems of arbitrary dimensionality and different size of data points.

Let us consider a function $f(\boldsymbol{\theta})$ defined over the domain $\mathcal{M}$ which here is not a geometrical domain, because each point in $\mathcal{M}$ is not associated with spatial coordinates but with a set of parameters, $\boldsymbol{\theta}$ representing the characterising properties of a problem's solution, e.g. stiffness, anisotropy etc. A possible approximation for $f(\boldsymbol{\theta})$ is given by a polynomial $\mathbf{P}(\boldsymbol{\theta})$ and its non-constant coefficients $\mathbf{a}(\boldsymbol{\theta})$:

$$f^h(\boldsymbol{\theta}) = \mathbf{P}(\boldsymbol{\theta}) \cdot \mathbf{a}(\boldsymbol{\theta}). \tag{5.54}$$

Now, let domain $\mathcal{M}$ be discretised by a finite number of parameter sets $\left\{\boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^p\right\}$, the so-called *particles* scattered in domain $\mathcal{M}$. Each particle is associated with a so-called *weight function* Φ of compact support. The size of the support can be individually defined for each particle $\boldsymbol{\theta}^I$, $I = 1, p$ by ϱ_I, the so-called *influence radius* of Φ. The collection of particles with a non-vanishing weight function at

point $\boldsymbol{\theta}$ constitutes its support denoted by Λ. Based on this set of particles, a weighted least square fit in the vicinity of point $\boldsymbol{\theta}$ can be constructed according to

$$J\left(\mathbf{a}\left(\boldsymbol{\theta}\right)\right) := \sum_{I \in \Lambda} \left[\mathbf{P}\left(\boldsymbol{\theta}^I \cdot \mathbf{a}\left(\boldsymbol{\theta}\right) - f\left(\boldsymbol{\theta}^I\right)\right)\right]^2 \Phi\left(\frac{\boldsymbol{\theta} - \boldsymbol{\theta}^I}{\varrho_I}\right). \tag{5.55}$$

The unknown coefficients $\mathbf{a}(\boldsymbol{\theta})$ can be readily determined by minimising the squared and weighted error between the approximated and actual values of $f(\boldsymbol{\theta}^I)$ at each of the particles $\boldsymbol{\theta}^I \in \Lambda$. That is minimising, the function J (Eq. (5.55)), with respect to $\mathbf{a}(\boldsymbol{\theta})$. Finally, the coefficients $\mathbf{a}(\boldsymbol{\theta})$ are substituted into Eq. (5.54) and the approximation of function $f(\boldsymbol{\theta})$ takes the following form:

$$f^h\left(\boldsymbol{\theta}\right) = \mathbf{P}\left(\boldsymbol{\theta}\right) \cdot \mathbf{M}^{-1}\left(\boldsymbol{\theta}\right) \sum_{I \in \Lambda} \left\{\mathbf{P}\left(\boldsymbol{\theta}^I\right) \Phi\left(\frac{\boldsymbol{\theta} - \boldsymbol{\theta}^I}{\varrho_I}\right) f_I\right\}, \tag{5.56}$$

where $\mathbf{M}(\boldsymbol{\theta})$ is the so-called moment matrix of the weight function Φ

$$\mathbf{M}\left(\boldsymbol{\theta}\right) = \sum_{I \in \Lambda} \mathbf{P}\left(\boldsymbol{\theta}^I\right) \mathbf{P}\left(\boldsymbol{\theta}^I\right) \Phi\left(\frac{\boldsymbol{\theta} - \boldsymbol{\theta}^I}{\varrho_I}\right), \tag{5.57}$$

and f_I are the so-called particle parameters.

Clearly, as the least square fit (Eq. (5.55)) only includes a very limited number of sample points, i.e. the particles $\boldsymbol{\theta}^I \in \Lambda$, the local character of the approximation $f^h(\boldsymbol{\theta})$ is ensured. With the refinement of the particle distribution the support of the weight functions can be chosen smaller and the approximation $f^h(\boldsymbol{\theta})$ converges for $\varrho_I \to 0$ to the exact function $f(\boldsymbol{\theta})$. The minimum number of supporting particles for any point $\boldsymbol{\theta}$ is dictated by the invertibility requirement of Eq. (5.57), that is the chosen basis polynomial. The smoothness of the MLS-approximation (Eq. (5.56)) depends on the continuity of the basis polynomial $\mathbf{P} \in C^m\left(\Omega\right)$ as well as the weight function $\Phi \in C^l\left(\Omega\right)$ and it holds $f^h \in C^k$ with $k = \min(l, m)$ [120]. The chosen complete basis polynomial in two dimensions is given by:

$$\mathbf{P}\left(\boldsymbol{\theta}\right) = \left[1, \theta^1, \theta^2\right], \tag{5.58}$$

and the weight function is based on a cubic spline:

$$w\left(r\right) = \begin{cases} \frac{2}{3} - 4r^2 + 4r^3 & \text{for } |r| \leq \frac{1}{2} \\ \frac{4}{3} - 4r + 4r^2 - \frac{4}{3}r^3 & \text{for } |r| \leq \frac{1}{2} \leq 1 \\ 0 & \text{for } |r| > 1 \end{cases}. \tag{5.59}$$

For the two-dimensional domain, the weight function can be constructed by the following expression:

$$\Phi\left(\frac{\boldsymbol{\theta} - \boldsymbol{\theta}^I}{\varrho_I}\right) = w\left(\frac{\theta_1 - \theta_1^I}{\varrho_I}\right) w\left(\frac{\theta_2 - \theta_2^I}{\varrho_I}\right). \tag{5.60}$$

To assess the interpolating power of MLS, a 1D and 2D example is briefly looked at. For the 1D scenario, a domain spanning from -5 m to 5 m is discretised using nodes uniformly spaced at 0.5 m. A quadratic polynomial of order 2 is used along with a cubic spline weight function to set up the MLS

approximants. The equation to be interpolated is a cubic polynomial:

$$f(x) = 1 \times 10^5 + x^3. \tag{5.61}$$

Following the interpolation procedure, the exact curve, given by Eq. 5.61 is compared with the interpolated one, as shown in Fig. 5.12. The cumulative L_2-norm error over the whole domain was 7.57×10^{-6}. However, they are not equally distributed everywhere. As shown on Fig. 5.13, the border region of the domain has the highest error as opposed to the middle ones. This is due to the supporting zone having fewer supporting nodes at the boundary. Additionally, it is observed that in the middle of the domain, the error distribution is not constant. This is due to the ability of MLS to almost exactly reproduce the exact value at a point of interest, while introducing errors between the nodes.

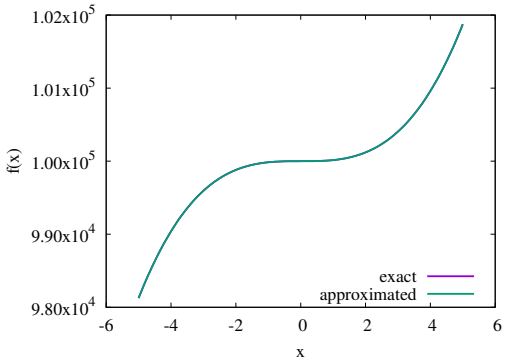


Figure 5.12: Cubic polynomial function and the MLS interpolated curve.

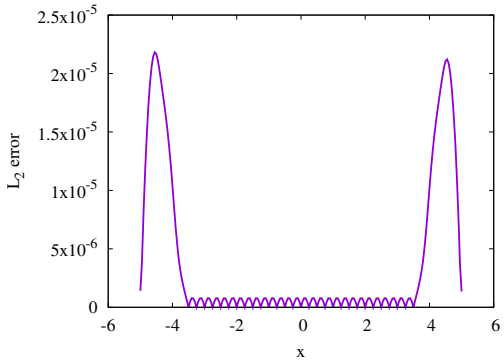


Figure 5.13: Error distribution across the domain.

The 2D scenario was also investigated with a square domain. The limits of the domain were -5 m to 5 m along the x and y direction. The cubic polynomial to be interpolated is now given as:

$$f(x) = 1 \times 10^5 + x^3 + y^3, \tag{5.62}$$

which, when plotted, produces the curve given in Fig. 5.14. Following the interpolation, the cumulative L2-norm was found to be only 1.06×10^{-5}, indicating good interpolating characteristics. Additionally, the error distribution was similar to the 1D scenario, where the error was higher along the boundary but lower in the middle of the domain, as indicated in Fig. 5.15.

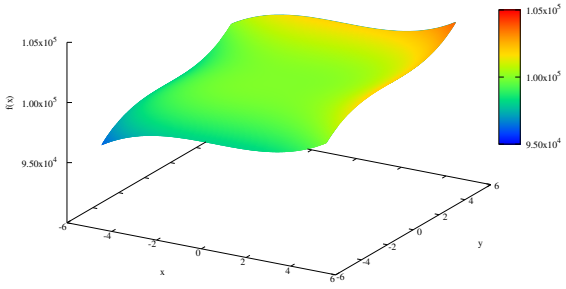


Figure 5.14: Exact 2D function.

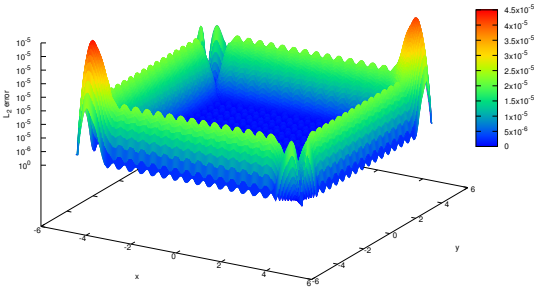


Figure 5.15: Error distribution across the 2D domain.

Based on the derivation and the above two examples, it can be observed that MLS can be extended to higher dimensions easily.

5.3.3 PODI implementation

This section outlines the general usage of PODI, coupled with a database. As such, a step-by-step description of our approach to set up the PODI algorithm is given. For the purposes of cardiac modelling, presented in Chapter 3, a PODI program was implemented in C++ as it provided the means to carry out computationally expensive tasks very efficiently.

The layout of our implementation was split into three main processes: *database construction*, *reduced order calculation* and finally, *post-processing*. Each process is discussed below and accompanied by a work flow chart of a complete simulation as illustrated in Fig. 5.16a, and in a chart focusing on the detailed steps of the PODI algorithm, as shown in Fig. 5.16b.

Database construction

The first step of the database construction procedure is to create all required datasets. An in-house code called SESKA, which is a hybrid code supporting FEM and MLS approximation, was run on a high-performance computer to produce multiple datasets at a faster rate. Each dataset consisted of a collection of solution fields, such as displacement, stress and strain, that was stored off-line for each time step of the simulation.

Regarding the database itself, a basic database management system was used where all datasets were stored locally on a hard-drive, structured into folders. A registry was then created through a comma-separated values (CSV) file. Any new dataset added to the database was registered afterwards in order to be used in the PODI calculation. Each entry in the registry not only recorded the location of the dataset but also its characterising parameter values, such as constitutive law constants, geometrical quantities, etc.

Query Procedure

After constructing the database, the reduced order calculation was started. The first step was to load the registry and read the set of parameters for which the calculation was going to be carried out.

Using the set of parameters, the supporting zone was established. Each entry of the database was then assessed in order to determine if they belonged to the supporting zone. If the query returned an insufficient number of datasets to ensure invertibility of the moment matrix, then the PODI calculation cannot proceed and the MLS influence radius must be increased. In those cases, the calculation was restarted with a new specified influence.

ROM Calculation

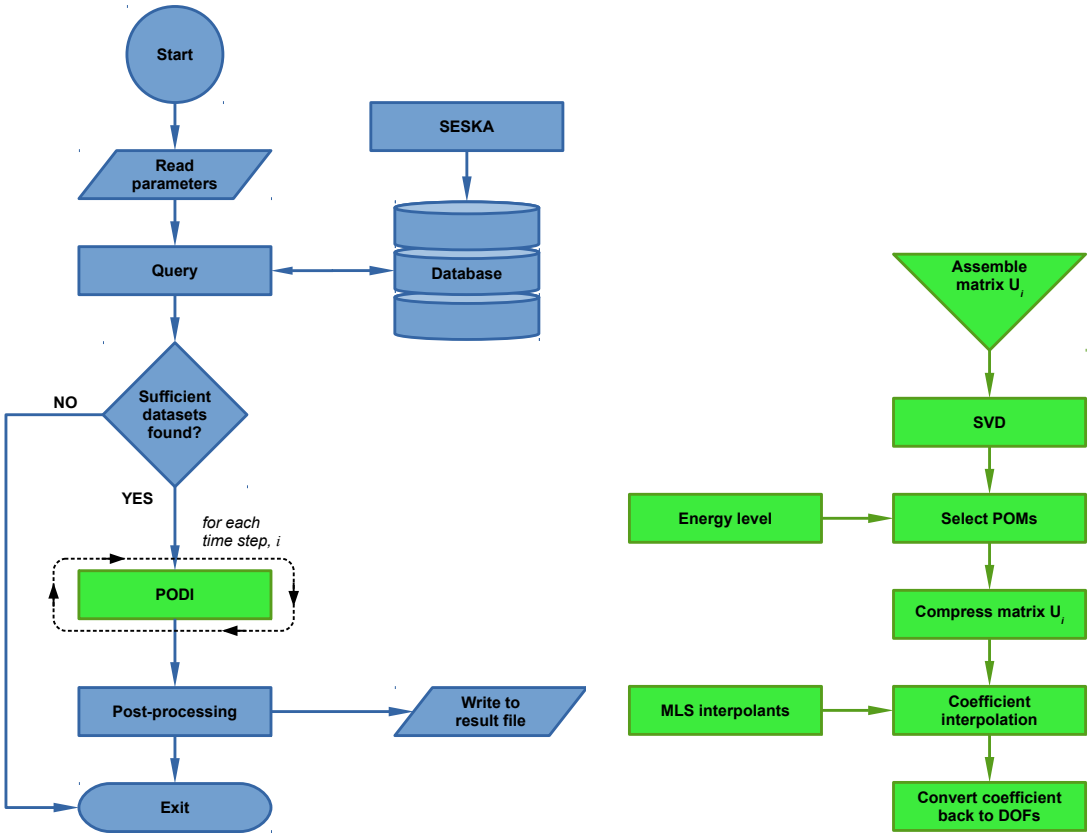
From each selected dataset, a solution field corresponding to a particular time step, i, and result field (such as displacement, stress and strain), was extracted. It was then condensed into a column form by stacking each of the degrees of freedom one after the other. In order to construct the ensemble matrix, the columns of all the selected dataset were assembled one after the other to form $\mathbf{U}_i$. The zero-mean ensemble, $\bar{\mathbf{U}}_i$, can be consequently computed by using Eq. (5.37)

The proper orthogonal decomposition of $\mathbf{U}_i$ or $\bar{\mathbf{U}}_i$ can be carried out using the two methods, which were introduced in Section 5.2. Both methods were implemented in order to render the whole PODI calculation more efficient. Hence, in the case where the number of rows was smaller than the number of columns, then the SVD was used. However, in the opposite scenario, where the number of rows were higher than the number of columns, the snapshot method was employed. The Linear Algebra Package (LAPACK) [168] was utilised to compute eigenvalues, eigenvectors and the Singular value decomposition required by the PODI algorithm. This package provided efficient matrix and vector manipulation methods, implemented in FORTRAN-90.

After selecting the dominant modes of a particular $\mathbf{U}_i$ or $\bar{\mathbf{U}}_i$ matrix, the ensemble matrix was then reduced, using Eq. (5.38). For the interpolation part in the PODI method, the MLS scheme was then called to compute the approximation functions, which were subsequently used to calculate the coefficients $\tilde{\mathbf{Z}}$ in Eq. (5.52). In order to recover the actual degrees of freedom of the unknown solution from those coefficients, Eq. (5.41) was again applied. If a zero-mean ensemble matrix was used, then the mean vector, employed to computed $\bar{\mathbf{U}}_i$, was added to the recovered degrees of freedom solution.

Postprocessing

Once the PODI calculation was completed, the results were saved and could be viewed for example by the post-processing software GiD (CIMNE International Center for Numerical Methods in Engineering).



(a) Overall process. (b) PODI process.

Figure 5.16: Flow chart of PODI simulation program.

5.4 Degrees of freedom standardisation methods

One of the drawbacks observed in the PODI method is that all datasets need to have a common geometry and underlying mesh configuration, even though their mechanical behaviour is almost identical. In the case where different geometries and mesh configuration are used, then $\mathbf{U}_i$ is malformed. Consider the example where the PODI calculation is carried out from five different datasets with geometries of different dimensions and mesh densities: model A = 400 DOFs, model B = 1000 DOFs, model C = 600 DOFs, model D = 2000 DOFs and model E = 1200 DOFs. Assuming all five models have been selected for the

interpolation, then the assembled $\mathbf{U}_i$ matrix will take the following form:

$$\mathbf{U}_i = \begin{bmatrix} u_1^1 & u_1^2 & u_1^3 & u_1^4 & u_1^5 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{400}^1 & \vdots & \vdots & \vdots & \vdots \\ & \vdots & u_{600}^3 & \vdots & \vdots \\ & u_{1000}^2 & & \vdots & \vdots \\ & & & \vdots & u_{1200}^5 \\ & & & u_{2000}^4 & \end{bmatrix}. \tag{5.63}$$

The type of matrices indicated in Eq. (5.63) cannot be further used for any matrix operations due to incompatible columns resulting from missing matrix entries and row-wise aligned DOFs not sharing the same spatial location. With cardiac modelling, such problems are likely to be encountered, especially when moving towards the concept of patient-specific modelling where the heart, for each individual, has distinctive features but behaves globally the same way. Consequently, this research intended to extend the usage of the PODI technique such that diverse geometrical shapes do not pose a problem during the interpolation process.

Across the literature, two recent papers by Amsallem et al. [68] and González et al. [69] have discussed the problem of extracting modes from different mesh configurations. In Amsallem et al., the POMs of different mesh configurations are transformed to the POM of a reference mesh configuration through a minimisation process. However, even though the transformed POMs conserved their orthogonal properties, it is not clear if they also conserved their optimality properties as being the most dominant modes, and how the whole transformation procedure impacts the total calculation time. In González et al., the approach was different. The authors embedded different liver geometries in a benchmark cube mesh grid and computed a so-called *distant field* with respect to the boundary of the organ. Following that, they then employed a method called *Locally Linear Embedding* to find the weightage of each registered liver to reproduce the geometrical shape of the liver at hand, and carry out an interpolation. However, it remains unclear as to how the nodes inside the liver geometry were treated. Therefore, due to uncertainties with respect to these approaches, the method proposed by Amsallem et al. [68] and González et al. [69] was not followed in this research, and an alternative approach was proposed. This alternative approach is referred to as the *degrees of freedom standardisation method*.

The human heart is geometrically unique. However, for people belonging to the same age, gender and health condition group, their hearts behave similarly. This similarity in behaviour across different hearts means that their POMs can also be extracted. But due to the uniqueness of their geometry, which leads to incompatible column matrices, this research proposed that all solution fields associated to these hearts needed to first go through a standardisation process. This standaradisation procedure, which took place during the pre-processing stage, consisted of projecting the solution field on the *template nodes*. The template nodes, which are points defined spatially across a template geometry, were the same for every dataset. The projection was based on an interpolation scheme that interpolates the solution fields from datasets to the template nodes. After the projection process, all the solution fields shared the same degree of freedom. Consequently, the data matrix, $\mathbf{U}_i$, had the same number of rows, for each column,

corresponding to the number of DOFs of the template nodes.

The complete PODI calculation procedure, including the standardisation process goes as follows. Let's assume that a database of N datasets, each having a different heart geometry with different mesh discretisations, be defined as $\left\{\mathbf{\Omega}^1, \ldots, \mathbf{\Omega}^N\right\}$ with their respective dataset nodes $\left\{\mathbf{x}^1, \ldots, \mathbf{x}^N\right\}$. $\mathbf{\Omega}^{\mathrm{T}}$ representing the domain of the template nodes and $\mathbf{\Omega}^{\mathrm{U}}$, the domain of the geometry for which the solution field is unknown and for which the PODI calculation will be run for. Hence, their corresponding set of nodes are $\mathbf{x}^{\mathrm{T}}$ and $\mathbf{x}^{\mathrm{U}}$, respectively.

The first step consists of placing the template geometry on $\left\{\mathbf{\Omega}^1, \ldots, \mathbf{\Omega}^N\right\}$ and then locating the template points inside $\left\{\mathbf{\Omega}^1, \ldots, \mathbf{\Omega}^N\right\}$. Finding these locations is important in the projection method. It allows the identification of neighbouring nodes for each template node and consequently, sets up a MLS interpolation scheme. This interpolation scheme then helps in projecting the solution fields of $\mathbf{x}^1, \ldots, \mathbf{x}^N$ onto the template nodes, $\mathbf{x}^{\mathrm{T}}$, resulting in standardised solution fields. With all the solution fields standardised, the $\mathbf{U}_i$ matrix can now be built-up properly and the PODI calculation carried out, with the resulting solution fields given at the template nodes, $\mathbf{x}^{\mathrm{T}}$. In order to transfer the interpolated results to the geometry of the problem-at-hand, another projection was carried out, with an MLS interpolation scheme set up for the problem nodes, $\mathbf{x}^{\mathrm{U}}$, to interpolate the solution fields from $\mathbf{\Omega}^{\mathrm{T}}$. A visual representation is shown in Fig. 5.17.

The most challenging aspect of the above proposed standardisation procedure was to locate a node inside a geometry in order to find its neighbours. A simple strategy could have been used where the neighbours are identified based on their spatial distances. However, such a setup leads to finding distances of all the nodes in $\mathbf{\Omega}^i$, with $\mathbf{\Omega}^i \in \left\{\mathbf{\Omega}^1, \ldots, \mathbf{\Omega}^N\right\}$, with respect to every node in $\mathbf{\Omega}^{\mathrm{T}}$. If this process were to be repeated for N datasets, the computational time and resources needed to do so would be tremendously expensive. Another problem was the definition of a MLS influence radius. This was required to identify which nodes were close enough to be considered as neighbours. Assuming an inappropriate value could easily have led to difficulties and invalid interpolation, particularly when non-convex geometries such as the heart, where large radii can find neighbours across walls separated by the ventricular chamber, is being dealt with. A small radius could possibly have helped, but the risk of not finding enough neighbouring nodes for the MLS moment matrix invertibility was high.

To prevent this problem, a point-in-polygon algorithm (PIP) was used. With PIP, it is first assumed that the nodes of the domains $\left\{\mathbf{\Omega}^1, \ldots, \mathbf{\Omega}^N\right\}$, $\mathbf{\Omega}^T$ and $\mathbf{\Omega}^U$ are connected by vertices. These vertices define elements of a discretised domain. The goal of the PIP is to find in which element a node of interest is located. Once identified, the nodes of that element, which is referred to as an *element of interest*, can then be labelled as neighbours. Using the mesh connectivity feature eventually solved the problems associated with the distance approach. Firstly, not all the elements needed to be looped over. Once a node was found to belong to a specific element, the search algorithm could be terminated. The only circumstance in which the search algorithm can fail is when the node lies on the surface of an element. Hence, in this case, the first element in which it was found was automatically assigned to it. Secondly, the definition of neighbour nodes could be regulated by finding the elements connected to the element of interest. These neighbouring elements could consequently have their nodes participating in the interpolation procedure. Hence, such a setup had the main advantage of not selecting nodes across void spaces. Due to these advantageous characteristics, the Point-In-Polygon algorithm was implemented. The PIP mechanism and its implementation are provided in Appendix B.

Two geometrically different configurations are considered as template geometries. These geometries are a cube and a bi-ventricle heart. The reason why these were chosen is due to their influence in build-up of the $\mathbf{U}_i$. In the incoming sections, these two different template geometries are introduced.

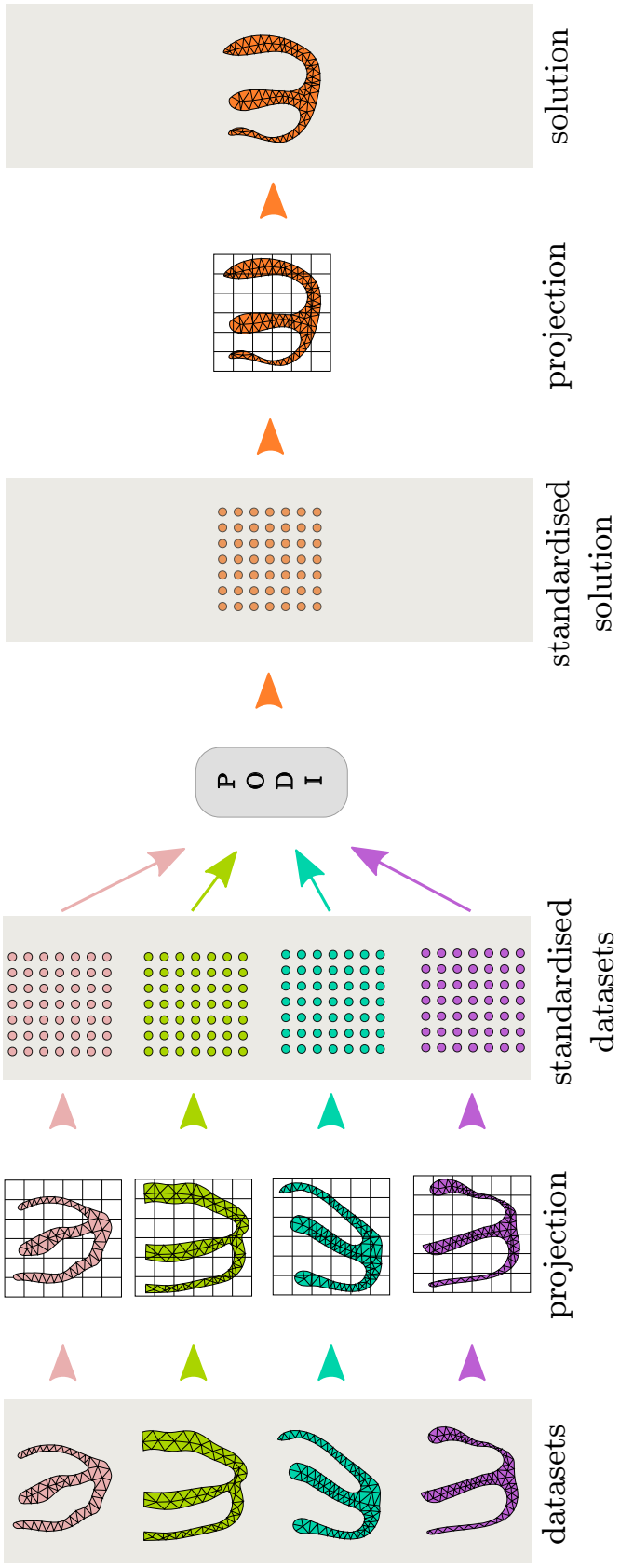


Figure 5.17: Flow of the PODI process when coupled with a standardisation method using a cube grid template.

5.4.1 Cube template standardisation

The cube template standardisation is the simplest of both standardisation methods. The template nodes of the cube grid are looked for inside the dataset's geometry mesh and the solution fields are interpolated on the template nodes. In order to ensure that all nodes of the datasets participate in the interpolation process, a large enough cubic grid is needed to encase the widest heart model. Hence, the preprocessing stage of the PODI calculation involves finding the endmost coordinates along every direction of all hearts. For any given direction, the difference between the largest and smallest coordinate will provide one of the cube's dimensions. After determining those dimensions, the cube can then be outlined and discretised using hexahedral elements to define the template nodes and their connectivity.

One problem that may be encountered with the above arrangement is the contribution of dataset surface nodes that are missing from the template nodes solutions. To explain why this is the case, consider the following 2D scenario, where solution fields of three hearts have been standardised using a 7×7 square grid of template nodes, as illustrated in Fig. 5.18. The usual procedure consists of first running the PIP algorithm for every template node across the hearts' elements to identify which ones are found inside and outside the heart geometry. Once done, the standardisation process can take place. First, the neighbouring heart's nodes are found. An interpolation scheme is then set up and the datasets solution fields are interpolated. With the solution fields defined at the template nodes, the corresponding matrix $\mathbf{U}_i$ is assembled. Even though all template nodes share the same spatial degrees of freedom and could be arranged column-wise, i.e, side by side, in the $\mathbf{U}_i$ matrix, their interpolated solution fields data are inconsistent. This inconsistency is due to the unique shape of every heart, which leads some template nodes, especially those located close to the surfaces of the heart geometries, to belong inside some datasets hearts but not others. This is likely to arise if the hearts' surfaces are not smooth or they are of extreme sizes. In this special scenario, entries that have missing interpolated solution fields are filled with a zero value. This is chosen as the solution fields do not span outside of the heart domain and the MLS scheme does not have extrapolation properties.

5.4.2 Heart template standardisation

An alternative suggested standardisation procedure is to make use of a heart template. Here, the heart of every dataset is first morphed onto a template-shaped heart. After the morphing process, the dataset nodes will therefore be clustered on the template nodes, ensuring that they will always belong to one of the elements of every dataset heart. Doing so conserves the solution details of the datasets when transferred to the template nodes, as opposed to the cube grid approach where a zero value is introduced.

As template heart geometry, a statistical average heart is ideal, as it has the most common geometrical features found in all the dataset hearts. This characteristic is useful, particularly during the morphing phase where the movement of the dataset nodes needs to be limited. Doing so ensures that all the heart's regions are correctly mapped on the template geometry, thus preserving accuracy during the projection stage.

Within the field of computer vision, the morphing process is commonly known as *registration*. It is an important tool for mapping two clouds of points onto each other. Three different techniques are commonly used. The first one is *rigid registration*, which consists of pure translation; the second is an affine registration which, additional to translation, allows for scaling and rotation. Finally, the third

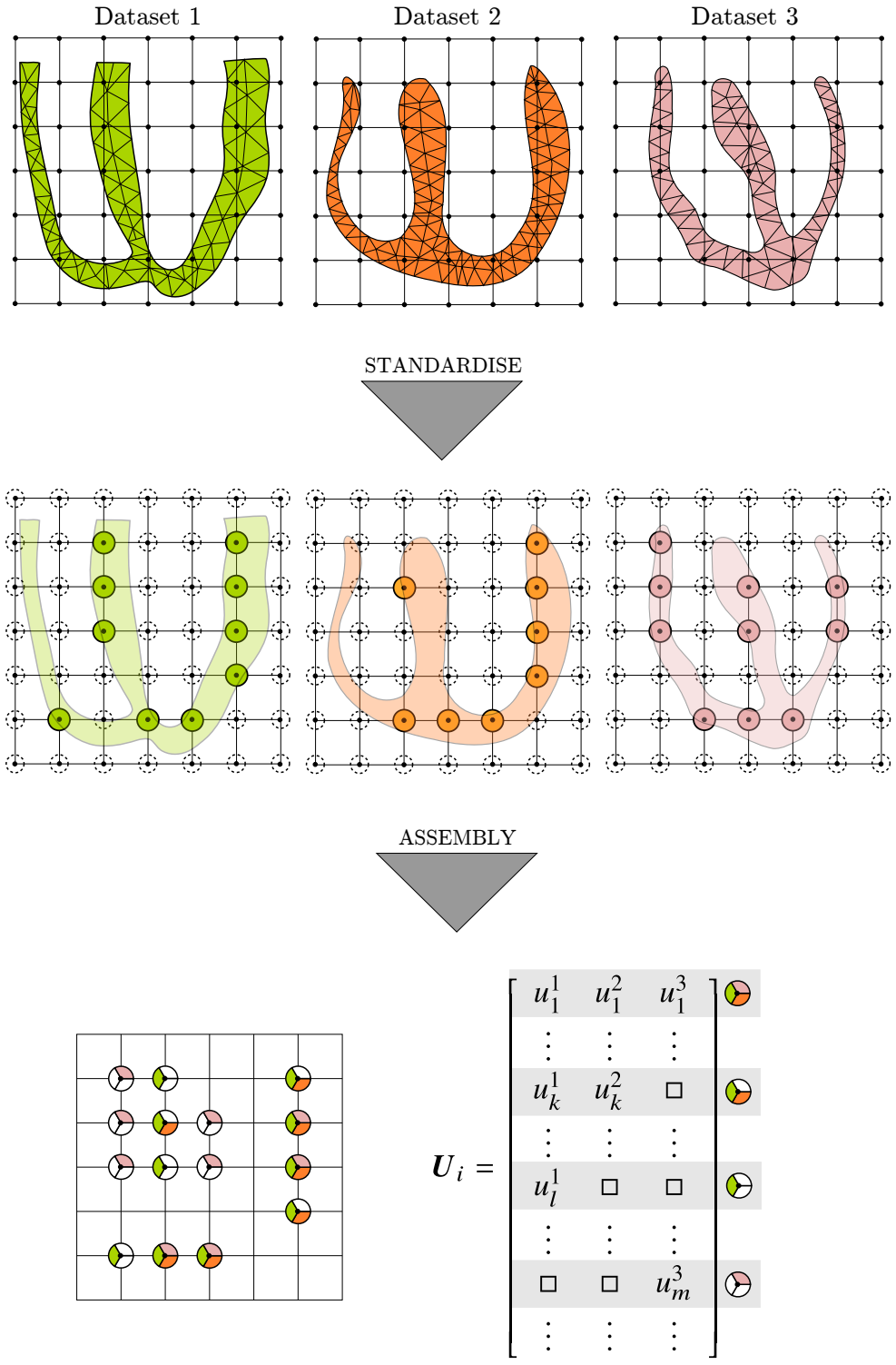


Figure 5.18: Incompatible $\mathbf{U}_i$ entries from cube template registration.

type of registration is called *non-rigid registration*. Rigid and affine registration methods allow for the preservation of the overall geometric details and affect the whole geometry. In the non-rigid scenario, the geometry can be completely morphed into another one, and the registration can be localised across the geometry. Due to these advantages, a non-rigid based registration method was used in this research.

An examination of the literature indicated that heart registration has previously been attempted. For example, in Sermesant et al. [118], a mass centre alignment followed by a *principal axes-based registration* and an affine registration, was employed to deform an idealised bi-ventricle heart model to MRIs. Toussaint et al. [169] made use of a log-diffeomorphic registration to project MRIs of a ventricle onto a perfectly truncated ellipsoid, while Lamata et al. [170] introduced a mesh warping technique that requires the conversion of the idealised geometry to a binary file, similar to MRI, before the registration. A common aspect across these registration methods is their MRI or binary images requirements, which are not available in this research. Currently, the template geometry and the source geometry are both defined by a mesh discretisation that consists of nodes and elements. Hence, to make use of the available geometries, a non-rigid point-based registration, called the *Coherent Point Drift* method, was utilised.

5.4.2.1 The Coherent Point Drift Method

The Coherent Point Drift method, CPD, introduced by Myronenko et al. [70], is a probabilistic method based on the *Maximum Likelihood* estimation technique and involves a motion coherence constraint over a velocity field in order to allow for the smooth movement of points from one spatial location to another. The following derivation of the Coherent Point Drift algorithm is a summary based on the work of [70]. For a more in-depth reading and understanding of the method, the reader is referred to [70, 71].

The Coherent Point Drift method starts by first considering two set of points. The first is the template points set, $\mathbf{Y}$, while the second is the data points set, $\mathbf{X}$. It is assumed that both set of points are matrices. The number of rows of these matrices corresponds to the number of points, while their number of columns relates to the dimension, D, of the points. As such,

$$\begin{aligned}\mathbf{Y} &= (\mathbf{y}_1, \ldots, \mathbf{y}_M)^{\mathrm{T}} \\ \mathbf{X} &= (\mathbf{x}_1, \ldots, \mathbf{x}_N)^{\mathrm{T}},\end{aligned}$$

where M is the number of template points and N is the number of data points in their respective set. The next step of the method is to first assume that each point in $\mathbf{Y}$ is the centroid of a Gaussian Mixture model (GMM). A GMM model is a way of clustering data into groups, over which a Gaussian probability distribution is fitted. Following the definition of each y_M as being the centroid, a density function of the Gaussian mixture probability is given as:

$$p(\mathbf{x}) = \sum_{m=1}^{M+1} P(m)\, p(\mathbf{x}|m) \tag{5.64}$$

where $p(\mathbf{x}|m)$ is defined as:

$$p(\mathbf{x}|m) = \frac{1}{(2\pi\sigma^2)^{\frac{D}{2}}}\, e^{-\frac{\|\mathbf{x}-\mathbf{y}_m\|^2}{2\sigma^2}}. \tag{5.65}$$

To Eq. 5.64, an additional probability distribution, $p(\mathbf{x}|M+1) = \frac{1}{N}$, is imposed so as to account for noises and outliers. If equal isotropic covariance, σ^2, and equal membership probabilities, $P(m) = \frac{1}{M}$, for every existing GMM are introduced, then both terms can be added to Eq. (5.64) by using a weight function, w^{CPD}:

$$p(\mathbf{x}) = w^{\text{CPD}} \frac{1}{N} + \left(1 - w^{\text{CPD}}\right) \sum_{m=1}^{M} \frac{1}{M} p(\mathbf{x}|m) . \tag{5.66}$$

Next, the GMM centroids are reformulated in terms of a velocity function, v, which is used to update the position of $\mathbf{Y}$ as:

$$\mathcal{T}(\mathbf{Y}, v) = \mathbf{Y} + v(\mathbf{Y}) . \tag{5.67}$$

To find an estimate of v, a negative log-likelihood function is minimised while imposing independent and identically distributed data assumptions and adding a regularisation term, $\phi(v)$:

$$f\left(v, \sigma^2\right) = -\sum_{n=1}^{N} \log \sum_{m=1}^{M+1} P(m)\, p(\mathbf{x}|m) + \frac{\lambda^{\text{CPD}}}{2} \phi(v) , \tag{5.68}$$

with λ^{CPD} being the constant regulating the contribution of the regularisation term. In order to find v and σ^2, the *Expectation Maximisation* algorithm is employed. Through the use of variational calculus, Eq. (5.68) is minimised to obtain:

$$Q\left(v, \sigma^2\right) = \frac{1}{2\sigma^2} \sum_{m,n=1}^{M,N} P^{\text{old}}(m|\mathbf{x}_n) \left\|\mathbf{x}_n - (\mathbf{y}_m + v(\mathbf{y}_m))\right\|^2 + \frac{N_{\mathbf{P}} D}{2} \log \sigma^2 + \frac{\lambda^{\text{CPD}}}{2} \phi(v) , \tag{5.69}$$

where $P^{\text{old}}(m|\mathbf{x}_n)$ is defined as a posteriori probability distribution and $N_{\mathbf{P}} = \sum_{n=1}^{N} \sum_{m=1}^{M} P^{\text{old}}(m|\mathbf{x}_n)$. The regularisation term, $\phi(v)$, is chosen to take the form of:

$$\phi(v) = \int_{\mathbb{R}^D} \frac{|\widetilde{v}(\mathbf{s})|}{\widetilde{G}(\mathbf{s})}, \tag{5.70}$$

where $\mathbf{s}$ is the so-called *frequency domain variable* and $\widetilde{G}$ is taken to be a *Gaussian kernel*. A Gaussian kernel is chosen since it is positive definite symmetric and provides a mean to regulate spatial smoothness. More importantly, it also allows the regularisation term to be equivalent to the one presented in the *Motion Coherence Theory*, which involves defining a coherent velocity field across a set of points with no prior data of their motion [171]. The function that minimises Eq. (5.69) is of the form $v(\mathbf{z}) = \sum_{m=1}^{M} \mathbf{W}_m^{\text{CPD}} G(\mathbf{z}, \mathbf{y}_m)$ with $\mathbf{W}_m^{\text{CPD}}$ determined using the following expression:

$$\left(\mathbf{G} + \lambda^{\text{CPD}} \sigma^2 \text{diag}(\mathbf{P1})^{-1}\right) \mathbf{W}^{\text{CPD}} = \text{diag}(\mathbf{P1})^{-1} \mathbf{PX} - \mathbf{Y}. \tag{5.71}$$

$\mathbf{W}^{\text{CPD}}$, of size $M \times D$, is the list of weight values. $\mathbf{G}$ is known as the Gram matrix with the size of

$M \times M$ and whose components are given by:

$$G_{ij} = e^{-\frac{1}{2}\left\|\frac{\mathbf{y}_i - \mathbf{y}_j}{\beta^{\text{CPD}}}\right\|^2}. \tag{5.72}$$

$\mathbf{Y}$ is updated through $\mathbf{Y}_{\text{update}} = \mathcal{T}\left(\mathbf{Y}, \mathbf{W}^{\text{CPD}}\right) = \mathbf{Y} + \mathbf{G}\mathbf{W}^{\text{CPD}}$ while the matrix of posterior probabilities, $\mathbf{P}$, can be evaluated through:

$$P_{mn} = \frac{\exp^{-\frac{1}{2\sigma^2}\left\|\mathbf{x}_n - \left(\mathbf{y}_m + \mathbf{G}(\mathbf{m},\cdot)\mathbf{W}^{\text{CPD}}\right)\right\|^2}}{\sum_{k=1}^{M} \exp^{-\frac{1}{2\sigma^2}\left\|\mathbf{x}_n - \left(\mathbf{y}_k + \mathbf{G}(\mathbf{m},\cdot)\mathbf{W}^{\text{CPD}}\right)\right\|} + \frac{w^{\text{CPD}}}{1 - w^{\text{CPD}}} \frac{\left(2\pi\sigma^2\right)^{D/2} M}{N}}. \tag{5.73}$$

Finally, to compute σ^2, the derivative of Eq. (5.69) is formulated and equated to zero so as to obtain the following expression:

$$\sigma^2 = \frac{1}{N_{\mathbf{P}} D}\left(\text{tr}\left(\mathbf{X}^{\text{T}}\text{diag}\left(\mathbf{P}^{\text{T}}\mathbf{1}\right)\mathbf{X}\right) - 2\,\text{tr}\left(\left(\mathbf{P}\mathbf{X}\right)^{\text{T}}\mathbf{T}\right) + \text{tr}\left(\mathbf{T}^{\text{T}}\text{diag}\left(\mathbf{P}\mathbf{1}\right)\mathbf{T}\right)\right). \tag{5.74}$$

The complete registration algorithm can therefore be given as:

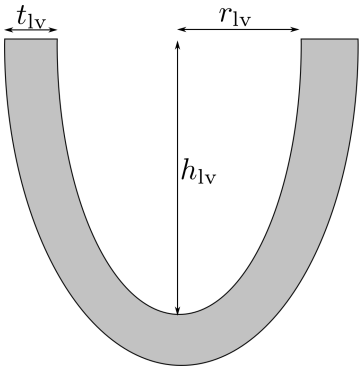
Algorithm 1 CPD algorithm

1: Initialise λ^{CPD} and β^{CPD}
2: Define $\mathbf{W}^{\text{CPD}}$ and $\sigma^2 = \frac{1}{DNM}\sum_{m,n=1}^{M,N}\|\mathbf{x}_n - \mathbf{y}_m\|^2$
3: Setup $\mathbf{G}$ using $G_{ij} = e^{-\frac{1}{2}\left\|\frac{\mathbf{y}_{0i} - \mathbf{y}_{0j}}{\beta^{\text{CPD}}}\right\|^2}$
4: **repeat**
5: Calculate $\mathbf{P}$ using Eq. (5.73)
6: Using Eq. (5.71), find $\mathbf{W}^{\text{CPD}}$
7: Determine $N_{\mathbf{P}} = \mathbf{1}^{\text{T}}\mathbf{P}\mathbf{1}$, $\mathbf{Y}_{\text{update}} = \mathbf{Y} + \mathbf{G}\mathbf{W}^{\text{CPD}}$
8: Evaluate σ^2 through Eq. (5.74)
9: **until** Eq. (5.69) converges
10: Compute $\mathbf{T} = \mathbf{Y} + \mathbf{G}\mathbf{W}^{\text{CPD}}$

5.4.2.2 CPD parameter optimisation study

The CPD algorithm is driven by three variables that influence the registration. The variables are w, λ^{CPD} and β^{CPD}, respectively. According to Myronenko and Xubo Song [71], their chosen default set of CPD parameters has been empirically chosen as $w = 0.3$, $\lambda^{\text{CPD}} = 2$ and $\beta^{\text{CPD}} = 2$ for the 3D registration. However, their registrations were restricted to 3D surfaces, which in this research is not the case as 3D volumes will be dealt with. Additionally, since the data points of the geometries used in this work belong to a mesh discretisation, there is a need for the CPD parameters to be optimised so that the mesh is not distorted.

To investigate the new set of CPD parameters, a 3D human left ventricle is considered. Due to the non-availability of MRI data to extract the geometry, a half-cut ellipsoid, shown in Fig. 5.19 and built using measured left ventricle dimensions is used.



Dimension	Value	Reference
Radius, r_{lv}	1.55 cm	[63, 64]
Thickness, t_{lv}	1.27 cm	[18]
Height, h_{lv}	7.35 cm	[65]

Figure 5.19: Idealised left ventricle.

The left ventricle is then discretised using tetrahedral elements to obtain a spatial distribution of points. To represent the template and the data geometry, two mesh configurations are defined. The template mesh consists of 974 nodes and 4060 elements, while the data mesh is made of 771 nodes and 3059 elements. Even though their discretisation is different, both mesh configurations have the same geometrical shape. To make the data mesh different for the purpose of registration, deformations in the form of translation, stretch and rotation are applied. As such, the two mesh configurations are now different. The template mesh and the data mesh are given in Fig. 5.20 and Fig. 5.21, respectively.

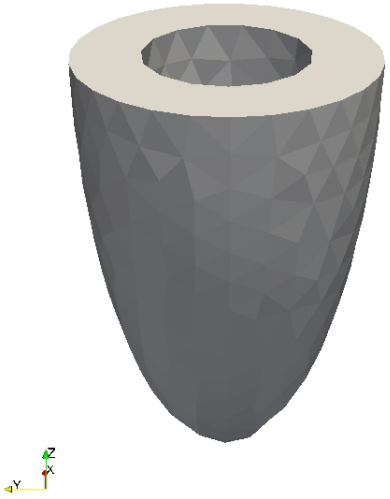


Figure 5.20: Template mesh.

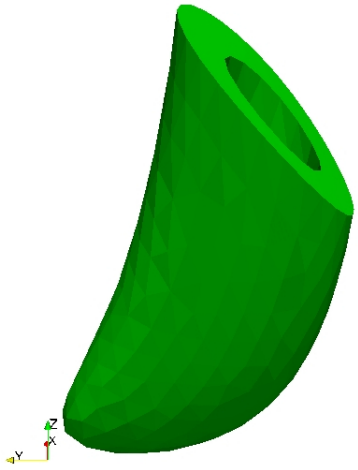


Figure 5.21: Deformed data mesh.

The CPD algorithm used in this research is based on a MATLAB code written by Myronenko, one of the authors of [71]. At the time of writing, the code could be publicly accessed from: https://sites.google.com/site/myronenko/research/cpd. To start the registration procedure, the MATLAB's code default parameters of $w^{\mathrm{CPD}} = 0.7$, $\lambda^{\mathrm{CPD}} = 2$ and $\beta^{\mathrm{CPD}} = 3$ were assigned. Following the CPD calculation, the results were then analysed by comparing the deformed data mesh with the original one. In this case, the un-deformed data mesh configuration was considered the *exact* one as its shape was the same as the template mesh. A visual representation of the obtained registered mesh compared to the template one is shown in Fig. 5.22.

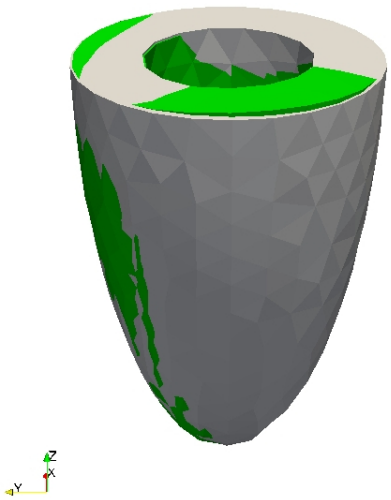


Figure 5.22: Superimposed template mesh (grey) and registered data mesh (green).

To measure the error in registration, a software called *Cloudcompare* [172] was used. It allowed the computation of distance between two cloud points using a modified technique based on the *Hausdorff distance* method. This technique finds the degree of resemblance of two different geometries, each defined by a different number of points [173]. The lower is the distance recorded, the better is the resemblance. With regard to the mesh quality, the VTK library [174] was made use of. Two properties of the mesh were measured, namely, the minimum angle inside each element and the shape of the elements. A high angle and shape value corresponds to better mesh quality. For each property, their highest, mean and lowest value were recorded to have an insight into their respective distribution across the whole geometrical mesh. With the error and quality measurements chosen, an analysis for each parameter was then carried out:

Beta, β^{CPD}: The parameter, β^{CPD}, where $\beta^{\text{CPD}} > 0$, was used when setting up the Graham matrix, $\mathbf{G}$, from Eq. (5.72) and therefore it directly affected the regularisation term given by Eq. (5.70). According to Myronenko et al. [70] and Myronenko and Xubo Song [71], the smoothness of the regularisation term is controlled by β^{CPD}. A lower value of β^{CPD} makes the selected Gaussian kernel more local and therefore results in a more localised smoothing registration. For large values, however, the registration is dominated by translation. To find the ideal β^{CPD} for the registration, β^{CPD} is varied from 0 to 6 before the results were plotted in Fig. 5.23 to Fig. 5.25.

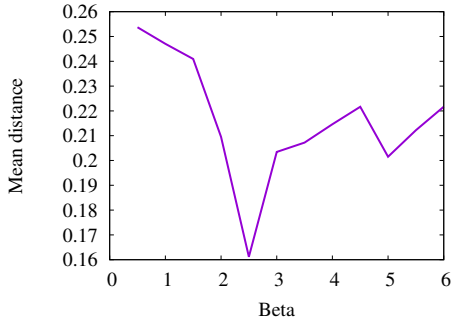


Figure 5.23: Change in mean distance with respect to Beta, β^{CPD}.

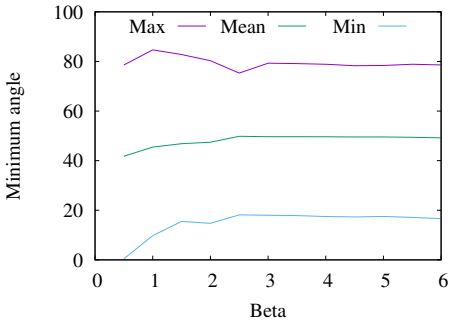


Figure 5.24: Change in minimum angle with respect to Beta, β^{CPD}.

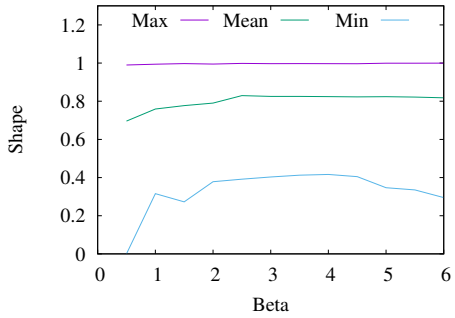


Figure 5.25: Change in shape with respect to Beta, β^{CPD}.

In Fig. 5.23, the mean distance decreases, as β^{CPD} increased from 0, to reach its lowest value at $\beta^{\text{CPD}} = 2.5$. However, from there onwards, the mean distance continuously increased. For the minimum angle and element shape property given in Fig. 5.24 and Fig. 5.25 respectively, their minimum and averaged values increased till $\beta^{\text{CPD}} = 2.5$. For the maximum values, they slightly decreased or stayed constant. With higher β^{CPD} values, the minimum angle and element shape mostly remained uniform or slightly degraded. Due to these observations, β^{CPD} was set to 2.5.

Lambda, λ^{CPD}: The parameter, λ^{CPD} is defined to be always positive, $\lambda^{\text{CPD}} > 0$ and was used in Eq. (5.71). The role of λ^{CPD} helps in finding a balance between the Expectation Maximisation, which results in a data fitting procedure [70], and the smoothing regularisation term, which add the contribution of the Gaussian kernel to the registration procedure, as given in Eq. (5.69). A very low magnitude of λ^{CPD} enforced the Gaussian kernel through the Graham matrix, $\mathbf{G}$, while a high value of λ^{CPD} imposed the terms $\sigma^2 d\,(\mathbf{P1})^{-1}$ as the main component of the left end side of Eq. (5.71). For the registration of the 3D idealised left ventricle, λ^{CPD} was varied from 0 to 10 and the results were plotted in Figs. 5.26 - 5.28, respectively.

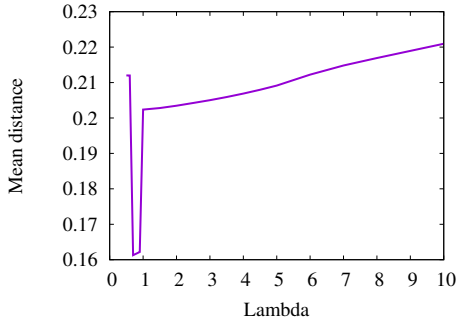


Figure 5.26: Change in mean distance with respect to lambda, λ^{CPD}.

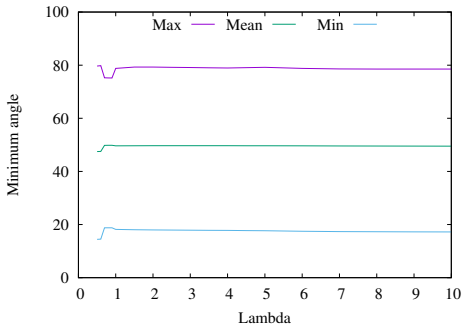


Figure 5.27: Change in minimum angle with respect to lambda, λ^{CPD}.

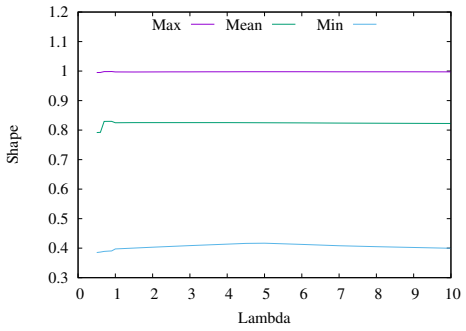


Figure 5.28: Change in shape with respect to lambda, λ^{CPD}.

From Fig. 5.26, the increase of λ^{CPD} led to higher mean distances which, however, for $0.7 \leq \lambda^{\text{CPD}} < 1$, recorded a sharp drop. This drop signified a much better registration and was also observed in the form of an upward kink for the min and mean curves across Fig. 5.27 and Fig. 5.28 respectively. Consequently, $\lambda^{\text{CPD}} = 0.7$, corresponding to the lowest mean distance value, was chosen for all registrations performed in Section 6.4.2.

Weight, w^{CPD}: The final parameter to be investigated was the weight parameter. w^{CPD} corresponds to the fraction of outlier points within a set of points and has the ability to increase the robustness of CPD against noises [70]. It was used to compute $\mathbf{P}$ using Eq. (5.73). According to Myronenko and Xubo Song [71], w^{CPD} should be defined between 0 and 1. As such, this interval was used for the current investigation.

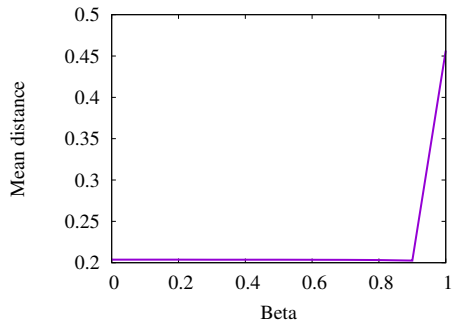


Figure 5.29: Change in mean distance with respect to Weight, w^{CPD}.

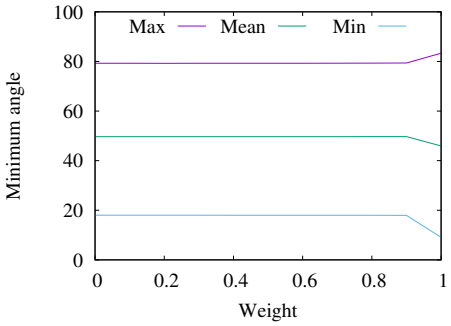


Figure 5.30: Change in minimum angle with respect to Weight, w^{CPD}.

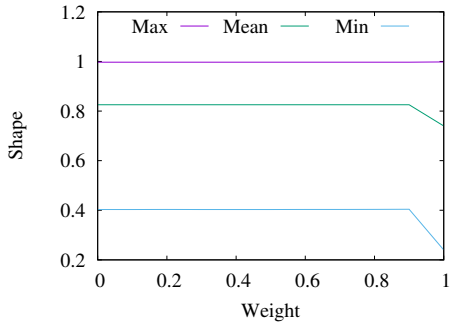


Figure 5.31: Change in shape with respect to Weight, w^{CPD}.

The results obtained are presented in Figs. 5.29 - 5.31. Since no outliers or noises were incorporated in both the template and deformed data point sets, the variation of the weight value did not have a significant effect on the registration, except for the range of $0.9 \leq w^{\text{CPD}} \leq 1$ where the term $\left(2\pi\sigma^2\right)^{\frac{D}{2}} \frac{M}{N}$ is enforced in Eq. (5.73). In that case, the registration degraded as the mean distance increased, while the minimum angle and shape suffered a drop. As such, the initial weight value, $w^{\text{CPD}} = 0.7$ was kept for all calculations.

From the above results, it was observed that only parameters β^{CPD} and λ^{CPD} are the main participants in the registration of the 3D ventricle. Due to the absence of outliers, the weight constant, w^{CPD}, had a negligible or detrimental effect. In addition, the plots of the mesh quality measures demonstrate an underlying relationship between the mean distance error and the mesh properties. The minimum angle and the shape were found to be of higher magnitude when the registration produces a low mean distance, indicating an overall increase of elements quality.

Part III

Application of Cardiac modelling and Reduced Order Methods

Chapter 6

Application of PODI to Cardiac Modelling

Following the introduction of all aspects of this research, the PODI method is now applied to cardiac modelling. The structure of this chapter is laid out as follows: in the first section, a benchmark problem is looked at in order to understand the mechanism of the PODI algorithm. The second section talks about cardiac geometry along with its material properties and boundary conditions used. PODI is then used in the third section to model the passive filling of the heart, while considering variations in elastic material behaviour and anisotropic properties. The fourth section investigates the application of Degrees of Freedom Standardisation procedure for different heart geometries datasets. Finally, in the last section, a whole heartbeat cycle simulation is carried out using PODI.

6.1 Benchmark problem

The problem investigated in Section 5.2.5 was also a benchmark problem. However, in this case, the dimensions of the problem were modified. The beam's length was reduced to 0.5 m while it's width was increased to 0.1 m. The domain was discretised using 4221 nodes and 4000 quadrilateral elements, leading to a total of 8442 degrees of freedom. For the PODI calculation, the Young's modulus was varied to create a one-dimensional parametric database. The variation took place between the parametric range of 200 GPa to 300 GPa with an increment of 10 GPa, creating 11 datasets which were stored off-line. The simulation time started from 0 s to 200×10^{-6} s. With a simulation time interval of 5.08×10^{-8} s, 3937 time steps were generated. The total calculation time of each dataset was found to be approximately 459.5 s.

With the database created, the PODI calculations were carried out. The displacement field solution for $E = 255$ GPa, a parametric point that is within the defined range of the parametric domain but is not part of the database, is sought. The PODI energy thresholds were investigated for 80 %, 85 %, 90 %, 95 %, 97 %, 99 %, 99.9 %, 99.99 % and 100 %. For the interpolation process, an MLS influence zone of radius 20 GPa was chosen to provide support for the parameter domain point $E = 255$ GPa, thus ensuring invertibilty of the MLS moment matrix.

The first set of results concerned the performance of PODI in terms of calculation time. The PODI's

displacement field was obtained within 16-19 s, which is about 24 times faster than the FEM calculations. Breaking down the stated PODI calculation time, approximately 13.2 s were used for reading the off-line displacement fields, representing about 76 % of the total time. It took 0.5 seconds to generate the MLS interpolants. The rest consisted of the PODI calculation itself, which entailed POVs and POMs extraction, projection of $\mathbf{U}$ into the low-dimensional space, interpolation, and projection of the solution back to the high-dimensional space for all time steps. This took on average 3.9 s, which was 22 % of the total time. As can be seen, although PODI is considerably faster than the FEM method, most of its calculation time was spent reading data from the off-line storage medium.

This gain in calculation speed came at the expense of the solution accuracy. Following the computation of the L_2-error norm of the PODI's displacement field, it was found that PODI introduced an average cumulative error of about 0.02 across the range of energy conserved, i.e. 80 % to 100 %. However, given the low magnitude of the error and the high calculation speed-up, PODI was considered an efficient and worthwhile method.

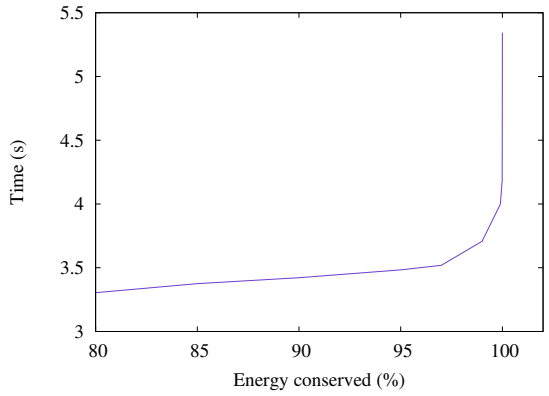


Figure 6.1: Change in calculation time with increase in energy conservation level.

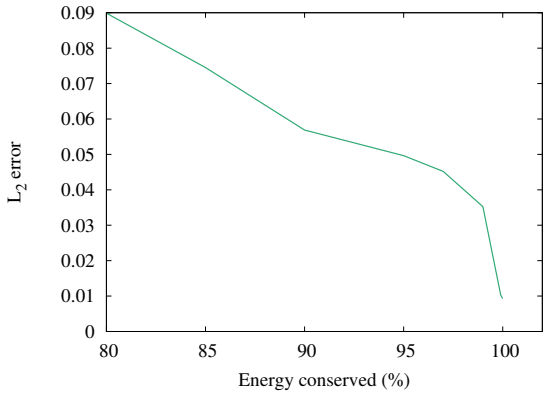


Figure 6.2: Change in calculation error with increase in energy conservation level.

The performance of the PODI method was investigated in great details, such as the impact of the components of the PODI method on accuracy and calculation time. Firstly, the energy conservation varied from 80 % to 100 % to monitor its effect on the PODI calculation. From Fig. 6.1, it can be observed that the calculation time increased exponentially as more POMs were conserved. But on the other hand, this led to a drop in error from 9×10^{-2} to 9×10^{-3} as indicated in Fig. 6.2. The displacement field was plotted at a particular node for different energies, and the increase in accuracy can be seen in Figs. 6.3 – 6.6. This trend, relating the energy conservation to the PODI calculation time and error norm, can be explained by considering the POMs and POVs generated across all iteration steps. For every time step, POD, or more specifically the snapshot method, was used to determine those POMs and POVs, which, for a particular time step, looked like those given in Fig. 6.7. The red part of the plots shows higher magnitudes of axial displacement, while the blue region indicates lower magnitudes. The corresponding energy of each POV is given in Fig. 6.8, where the y-axis is logarithmic. In order to generate more accurate solution fields, more POMs were needed to enrich the low-dimensional space that was created for interpolation. As more POMs were used, the computed coefficients became more enriched with detailed information. This led to a more accurate solution during the interpolation process, as less errors were introduced during the back and forth data projection from the low-dimensional space. However,

this increase in POMs also led to an increase in size of the projection matrix and accordingly, to a larger number of matrix multiplication operations. Additionally, since the dimensions of the low-dimensional space were higher, POD interpolation included more contributions, Eq. (5.52), and hence required more computational time.

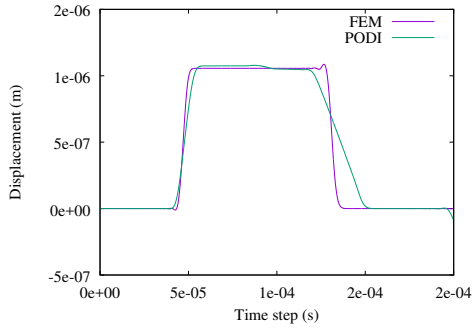


Figure 6.3: PODI with 80 % of energy conserved.

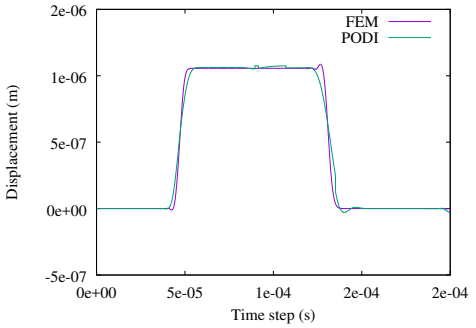


Figure 6.4: PODI with 99 % of energy conserved.

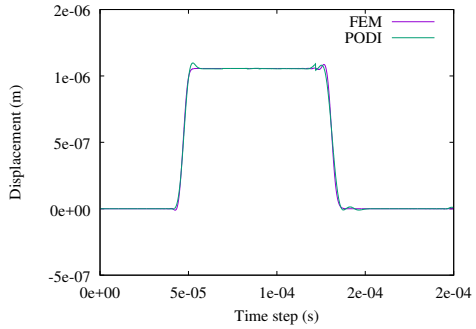


Figure 6.5: PODI with 99.9 % of energy conserved.

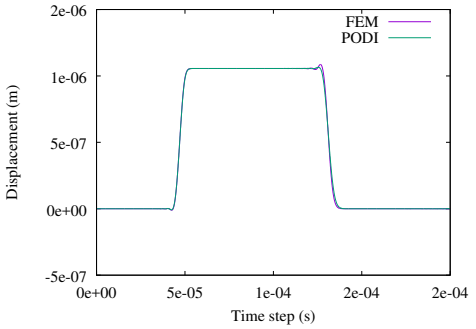
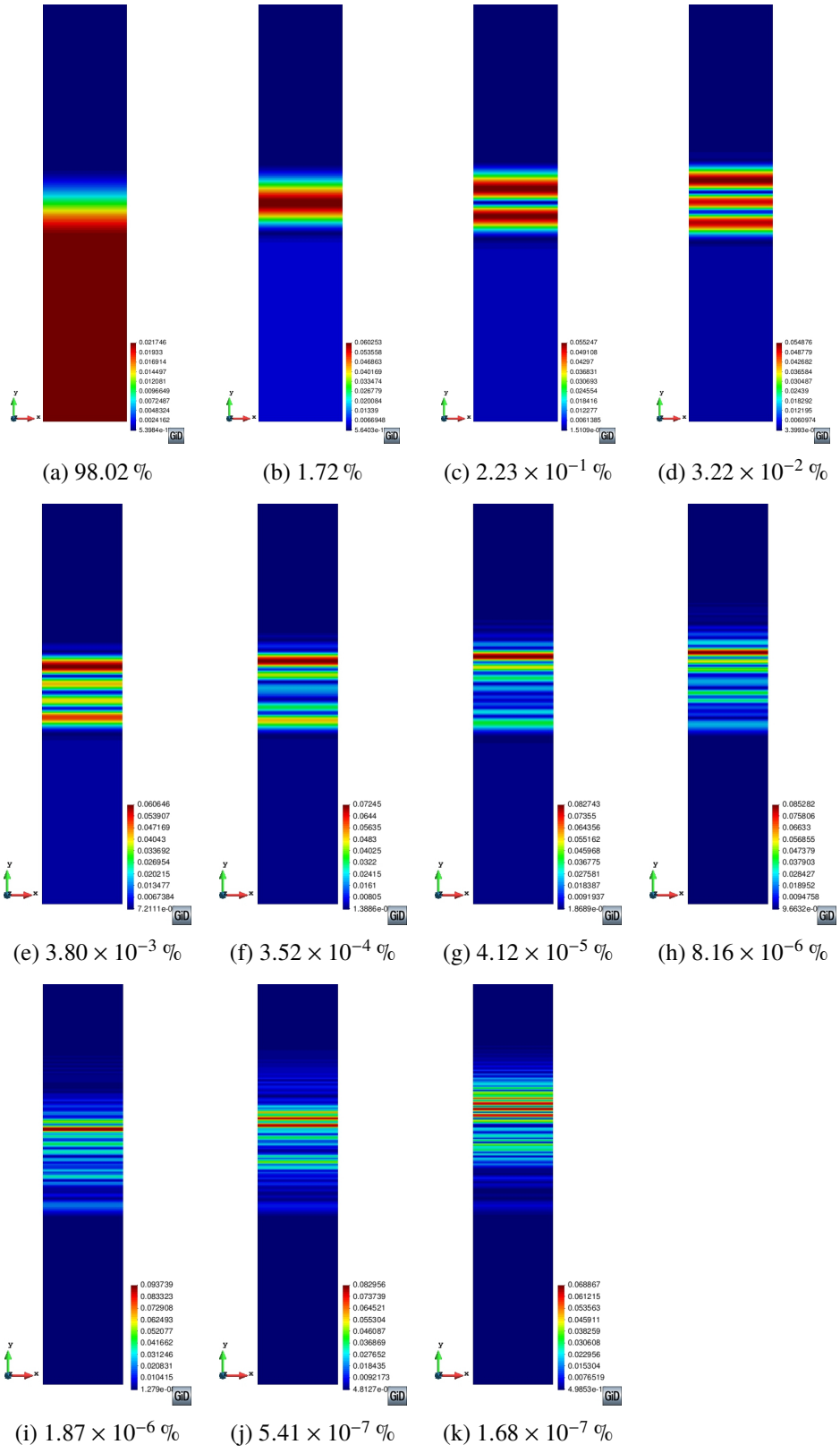


Figure 6.6: PODI with 100 % of energy conserved.



(a) 98.02 %
(b) 1.72 %
(c) 2.23×10^{-1} %
(d) 3.22×10^{-2} %
(e) 3.80×10^{-3} %
(f) 3.52×10^{-4} %
(g) 4.12×10^{-5} %
(h) 8.16×10^{-6} %
(i) 1.87×10^{-6} %
(j) 5.41×10^{-7} %
(k) 1.68×10^{-7} %

Figure 6.7: Plot of each POM and the percentage energy it accounts for.

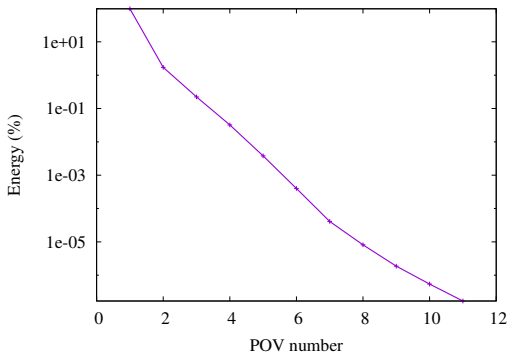


Figure 6.8: Energy fraction represented by each POV.

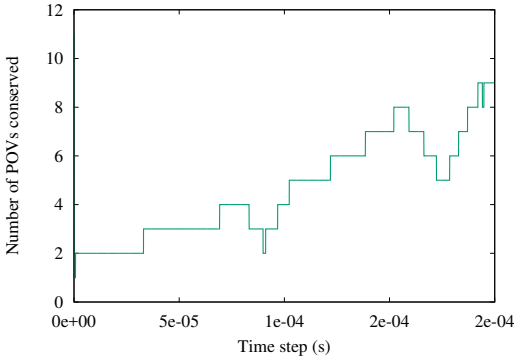


Figure 6.9: Number of POVs conserved across time steps.

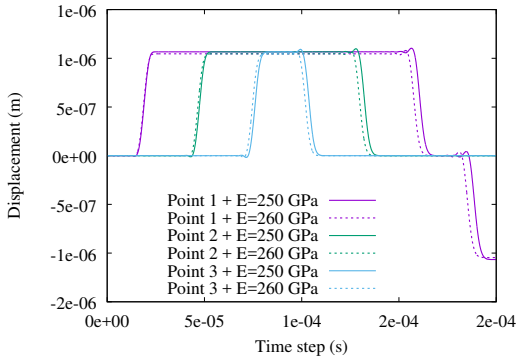


Figure 6.10: PODI with 80 % of energy conserved.

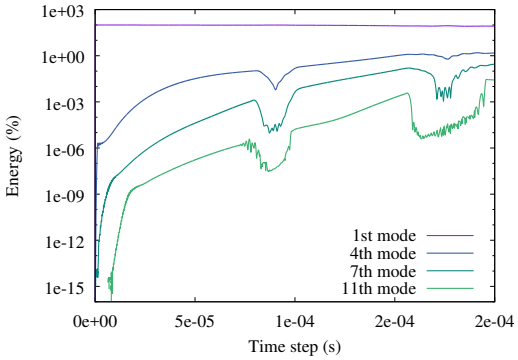


Figure 6.11: Variation of energy conserved across time steps.

Two interesting final observations were made when looking at the number of POMs conserved throughout a simulation. Across time steps, the number of POVs conserved increased. However, the increase was not monotonous and was interrupted at two specific locations where the axial force wave reflected against the fixed end. In order to explain this observation, the evolution of the number of POVs conserved, shown in Fig. 6.9, was compared with the displacement field of the supporting datasets. Two datasets, corresponding to a Young's modulus, E, of 250 GPA and 260 GPA were chosen, since they were closer to the problem-at-hand, which was $E = 255$ GPA. Three equally spaced points along the beam were selected and their displacement field plotted, in Fig. 6.10.

With an increase in the Young's modulus value, the wavefront axial force travelled faster due to elastic wave propagation [92]. The quicker the wavefront, the earlier the nodal displacement rises and drops back to zero. However, it can also be observed in Fig. 6.10 that the difference between the displacement field associated to a particular point for the different Young's moduli, increased across the time steps. With those increments, the matrix $\mathbf{U}_i$, assembled at these later time steps, contained dissimilar solution fields. Therefore, the decomposition of $\mathbf{U}_i$ resulted in POMs whose corresponding energy levels were more scattered across the POVs. Such effect can be observed, if the energy conserved is plotted, as shown in Fig. 6.11, where the most dominant POM loses energy to the other POMs. This loss of energy, therefore, led to more POVs being conserved.

With regard to the two dips registered in the curves of Figs. 6.9 and 6.11, it was found that they

occurred during the wave reflections against the fixed support of the beam. In those instances, the displacement fields across the beam's geometrical domains were very similar, since both beams had already experienced the same magnitude of maximum compression. These compressions were the same, since the axial wavefront force profile and magnitude did not change. Consequently, the decomposition of $\mathbf{U}_i$ at those time steps had fewer highly energetic modes. However, it was also observed that the second dip did not drop to the same level as the first one, and seems to have been wider. This effect could be attributed to out-of-phase phenomena that start to be more dominant after the wave reflects against the fixed support, as explained earlier.

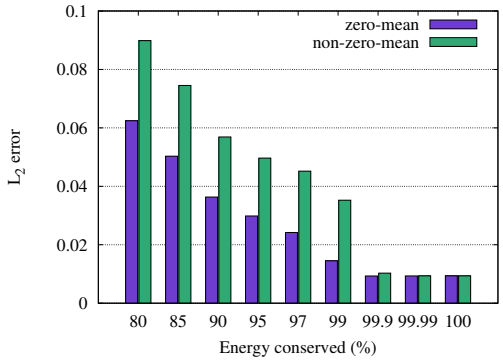


Figure 6.12: Difference in displacement error between the zero-mean and non-zero-mean method.

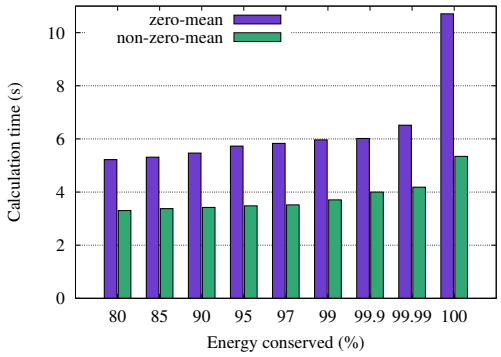


Figure 6.13: Change in calculation time with increase in energy conservation level.

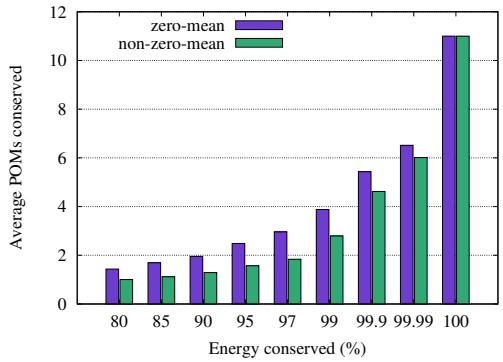


Figure 6.14: Change in average POMs conserved with increase in energy conservation level.

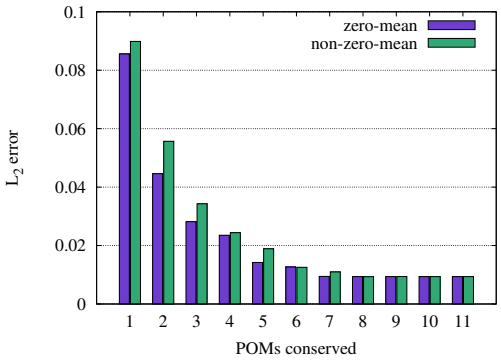


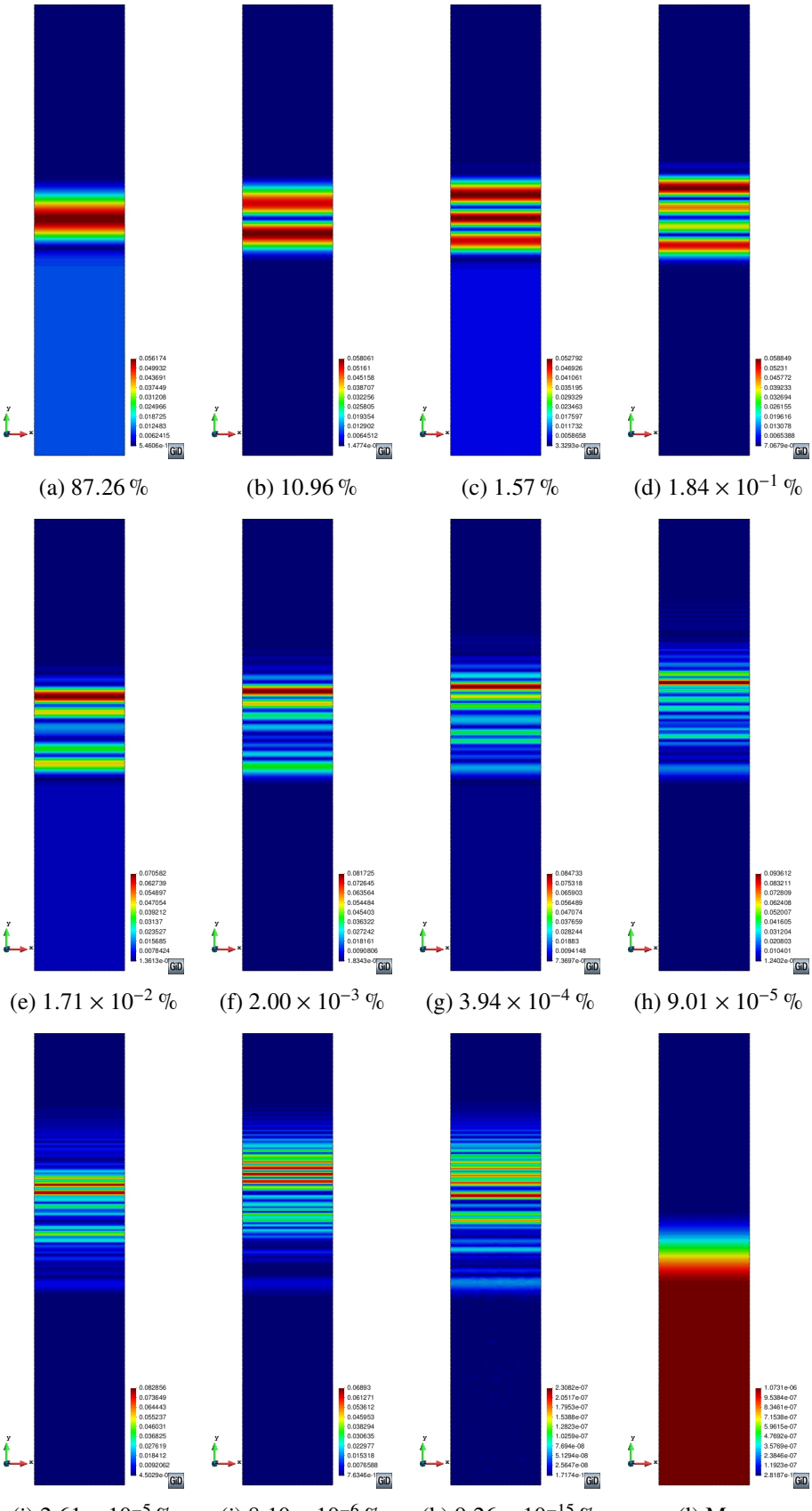
Figure 6.15: Change in error with increase in POMs conserved.

The use of the non-zero-mean as compared with the zero-mean data matrix $\bar{\mathbf{U}}_i$ obtained from Eq. (5.37), was next investigated. As outlined in Section 5.2.3, it is still unclear whether it affects the performance of PODI. With a zero-mean matrix, the centroid of the results data is shifted to the origin of the Cartesian axes and this leads to POMs being re-arranged along the results data. Using the same strut beam example, the above results were recomputed with the zero-mean matrix $\bar{\mathbf{U}}_i$ and compared against the non-zero-mean results in order to find how the zero-mean matrix affected the overall properties of the PODI method. The error trend across the levels of energy conserved, i.e. 80 % - 100 %, was found to be similar to that given earlier in Fig. 6.2. However, the zero-mean method produced a more accurate solution for a specific energy level, which can be observed in Fig. 6.12. This reduction in error ranged

between 30 % to 60 % at 80 % - 99 % energy conserved. Beyond 99.9 %, the error decreased from 9 % to 0 %. Accordingly, the use of a zero-mean data matrix led to a general reduction of interpolation error. However, the zero-mean method created additional computation overheads considering that the mean needed to be determined, subtracted from every column of $\mathbf{U}_i$ and finally added to the high-dimensional interpolated results. As such, the PODI calculation was 50 % to 65 % slower, as shown in Fig. 6.13, than the non-zero-mean method while still maintaining a similar calculation time-energy conservation profile to that of the non-zero-mean method given in Fig. 6.1. This increase in calculation time cannot be fully attributed to the zero-mean procedure. In fact, it was noticed that the number of POMs conserved in order to achieve the targeted energy was increasing, as shown in Fig. 6.14. This increase in number of POMs being selected also translated into more matrix-multiplication operations during the projection from high- to low-dimensional space and vice versa, along with more coefficients that had to be interpolated. The need to conserve more POMs also implies that the energy of each POM was lower. This was observed when comparing the POMs and POVs at the same time step, in both methods. The new POMs and their corresponding POVs are plotted in Fig. 6.16, and were matched against those given in Fig. 6.7. Based on this comparison, it was firstly noted that the first POM was 10 % less dominant. This loss of energy was found to be redistributed amongst the remaining POMs, as their associated energy was considerably higher. Secondly, the structure of the POMs was no longer the same. Even when compared individually, the POMs looked different, and it was noticed that the first POM of the zero-mean matrices corresponded to that of the second POM of the non-zero-mean matrices. This observation was also made for all the rest of the POMs, except for the last one. Additionally, the mean displacement field, illustrated in Fig. 6.16(l), was found to be similar to the first POM of the non-zero-mean method, given in Fig. 6.7(a).

The zero-mean effect on the POVs and POMs was also observed by Tamura et al. [159]. In their paper, they also found that the first POM of the non-zero-mean method was similar to the mean vector of $\mathbf{U}_i$, which indicated that the latter may have been captured by the non-zero-mean method instead of the most dominant mode of disturbances of the zero-mean data matrix. Such a scenario would be valid only if the major component of the disturbances coincided with the direction of the mean vector [159, 160]. In this example, it was not the case, as the major component of the disturbances, which was the first POM of the zero-mean method, was not similar to the non-zero-mean POM. This discrepancy has lead to the belief that the non-zero-mean POMs do not truly account for the disturbances in the data. Due to this fact, the low-dimensional space does not represent the true essence of the data. This, therefore, translates to the non-optimal transformation of the interpolated coefficients, which leads to a less accurate solution at the end of the PODI calculation.

From the results, it was deduced that the zero-mean method redefined the POMs of the non-zero-mean method. This redefinition was the result of $\bar{\mathbf{U}}_i$ retaining only the deviation of the data from the mean data. Consequently, the POMs were properly centred at the centroid of the data, hence enabling their correct orientation, as explained in Section 5.2.3.



(a) 87.26 % (b) 10.96 % (c) 1.57 % (d) 1.84×10^{-1} %

(e) 1.71×10^{-2} % (f) 2.00×10^{-3} % (g) 3.94×10^{-4} % (h) 9.01×10^{-5} %

(i) 2.61×10^{-5} % (j) 8.10×10^{-6} % (k) 9.26×10^{-15} % (l) Mean

Figure 6.16: Plot of each POM and the percentage energy it accounted for.

6.2 Cardiac problem definitions

In this section, the heart model used in this research is introduced. The problem definition consists of setting up a heart geometry, determining which of the constitutive parameters would be used to construct the parametric domain, and constructing the simulated-heart databases.

6.2.1 Geometry and boundary conditions

In this research, MRIs of a human bi-ventricle heart were not available. Hence, a computer-generated 3D geometry of the right and left ventricle, representing a bi-ventricle (BV) model, was created. To come up with a realistic geometry, important aspects of a real human heart were conserved. Those aspects were cavity volumes, wall thickness, ventricular diameter and ventricular depth. A cross-section of the geometry is given in Fig. 6.17, and its dimensions, extracted from the literature, correspond to the end-systole phase of the heart, as presented in Tab. 6.1.

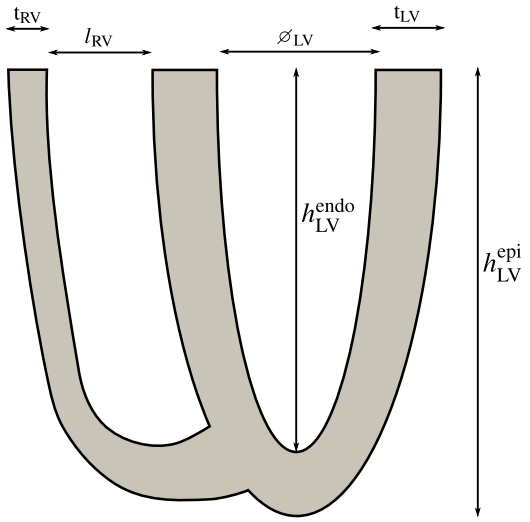


Figure 6.17: Cross-section of bi-ventricle model.

	Values		Reference
	Range	**Choice**	
Left Ventricle			
Volume (mL)	22 - 48	36.98	[18, 59]
Base diameter, $\varnothing_{LV}$ (cm)	2.2 - 4	3.1	[63, 64]
Wall thickness, t_{LV} (cm)	1.09 - 1.45	1.27	[65]
Diameter-depth ratio	2 - 3	2.37	[18]
Endocardium height, h_{LV}^{endo} (cm)	-	7.35	-
Epicardium height, h_{LV}^{epi} (cm)	-	8.62	-
Right Ventricle			
Volume (mL)	24 - 62	51.21	[18, 59]
Base diameter, l_{RV} (cm)	1.2 - 2.6	1.9	[60]
Wall thickness, t_{RV} (cm)	3.5 - 8.5	6.35	[62]

Table 6.1: Dimensions of the bi-ventricle model.

h_{LV}^{endo} and h_{RV}^{epi} were not directly obtained from the literature. The former was computed using the diameter to depth ratio and the base diameter measure, while the latter is the summation of h_{LV}^{endo} and t_{LV}. The left ventricle (LV) was first created using two halved ellipsoids. On the left hand side of the LV, the surface was extruded by a distance of l_{RV} to obtain the right ventricle endocardium and $l_{RV} + t_{RV}$ was used for the right ventricle epicardium. The final 3D geometry generated is given in Fig. 6.18.

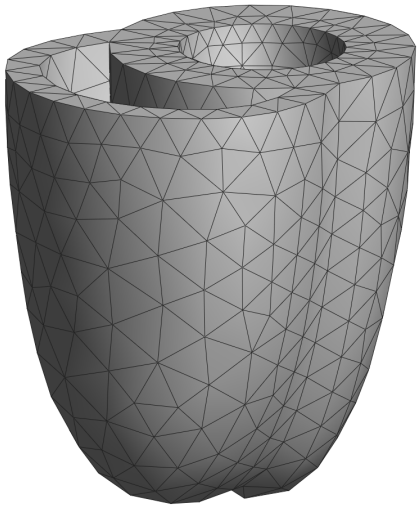


Figure 6.18: 3D geometry of a bi-ventricle heart.

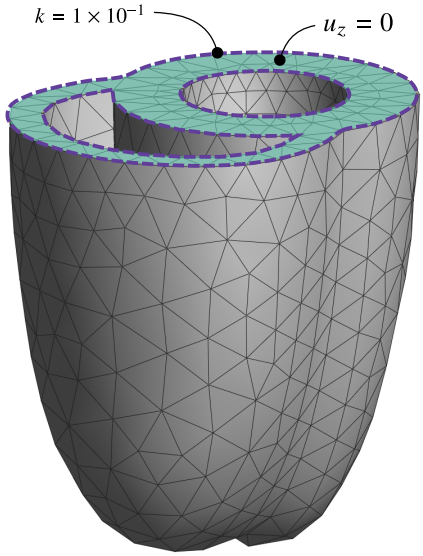


Figure 6.19: Dirichlet boundary conditions applied to the bi-ventricle heart model.

With the geometry defined, the boundary conditions were then applied. The Dirichlet boundary conditions were first considered. Two different ones were imposed on the basal surface of the BV, as shown in Fig. 6.19. The first boundary condition enforced zero z-direction displacement [20]. The

reason for doing this was to emulate the presence of connections to the major blood vessels at the base, preventing the basal surface to move up or down. Secondly, an elastic boundary condition with a prescribed stiffness of 1×10^{-1} kN mm^{-1} was applied along the endocardium and epicardium lines on the base. These boundary conditions allowed the thickening and thinning of the base wall during contraction, but restricted rotation about the longitudinal axis.

The Neumann boundary conditions were next applied. To model blood filling the ventricles, a surface pressure was applied. As indicated in Chapter 2, the left ventricle needs to develop a higher pressure when pumping blood back to the arterial as opposed to the right ventricle, which pumps blood to the lung. Hence, in this research, the left ventricular pressure was set to be higher than the right ventricular pressure as given in Tab. 6.2.

	Pressure, kPa	Referrences
Left ventricular pressure, LVP	1.5	[175, 176, 177]
Right ventricular pressure, RVP	1	[176, 177]

Table 6.2: Ventricular pressure of the bi-ventricle model.

With the geometry set up and the boundary conditions applied, the choice of material properties is elaborated on in the following sections.

6.2.2 Material properties

The material properties that were assigned to the bi-ventricle heart model were the fibre orientation and the parameters for the passive elastic material model, the active tension model and the Windkessel model. In all cases, experimental data were not available and were therefore extracted from the literature.

6.2.2.1 Fibre orientation

The fibre orientation that was used was based on the DTMRI data of Rohmer et al. [9]. It provided both fibre and sheet angle in accordance with other experimental work such as that of Lombaert et al. [84] and allowed the construction of all three orthogonal fibre orientations in a 3D space. In Rohmer et al. [9], a human heart left ventricle was scanned and its fibre angles averaged for different zones of the left ventricle. In order to simplify the procedure of the fibre assignment on the geometry, the basal angles were used and assigned to the whole geometry. The values retained from the provided experimental data are given in Tab. 6.3.

	Min	Mean	Max
Epicardium fibre, θ^{epi}	-66	-57	-48
Endocardium fibre, θ^{edo}	55	72	89
Epicardium transverse fibre, β^{epi}	57	45	34
Endocardium transverse fibre, β^{endo}	-26	-10	6

Table 6.3: Fibre and sheet angle across the epicardium and endocardium of the bi-ventricular heart [9].

Using the chosen data, the fibre and sheet angles were assigned to both the right and left ventricle respectively. After employing the algorithm given in Section 3.4.1.3 to compute the surface fibre

orientation, the spatial fibre distribution throughout the myocardial wall, was generated using the Moving Least Square method, as shown in Fig. 6.20.

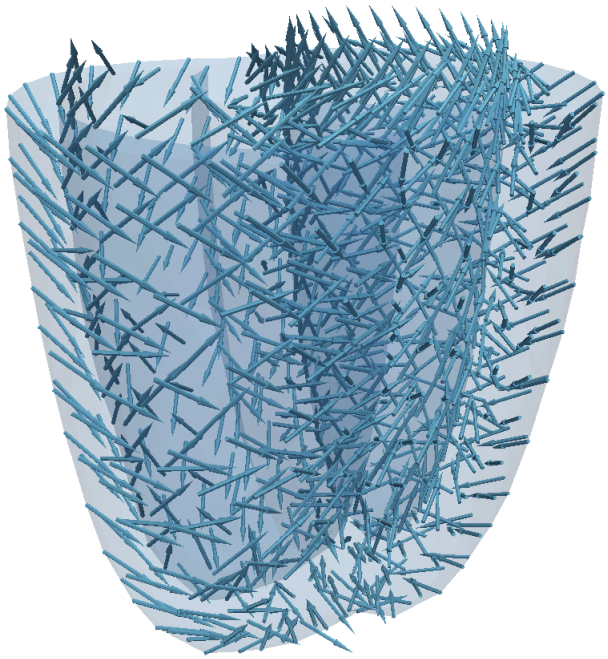


Figure 6.20: 3D fibre distribution across the bi-ventricle heart.

6.2.2.2 Material constants and Windkessel parameters

For the diastole, isovolumetric contraction/relaxation and ejection phase of a heartbeat, the material parameters, listed in Tab. 6.4, had to be quantified. The active stress parameters are given in Tab. 3.1, and were proven to be applicable to the human heart [107], while the Windkessel parameters values are extracted from [2] and modified in such a way that an adequate pressure-volume curve during ejection is obtained. However, the parameters of the passive model were still to be found. Triaxial test data were chosen to do so. The recent work of Sommer et al. [14] was used, as it is the only available experimental data extracted from the human myocardium. Using the simulated triaxial shear modes as given in Section 3.4.1.2, a parameter identification process was carried out to match the shear stress-shear strain curve of all modes of the experimental data of [14] by varying the constitutive parameters.

Model	Parameters
Passive	A, A_{comp}, a_1, a_2, a_3, a_4, a_5, a_6
Active stress	T_{max}, Ca_0, $(Ca_0)_{max}$, B, l_0, t_0, m, b
Three-element Windkessel	C, R_a, R_p

Table 6.4: Parameters of the passive, active and three-element Windkessel model.

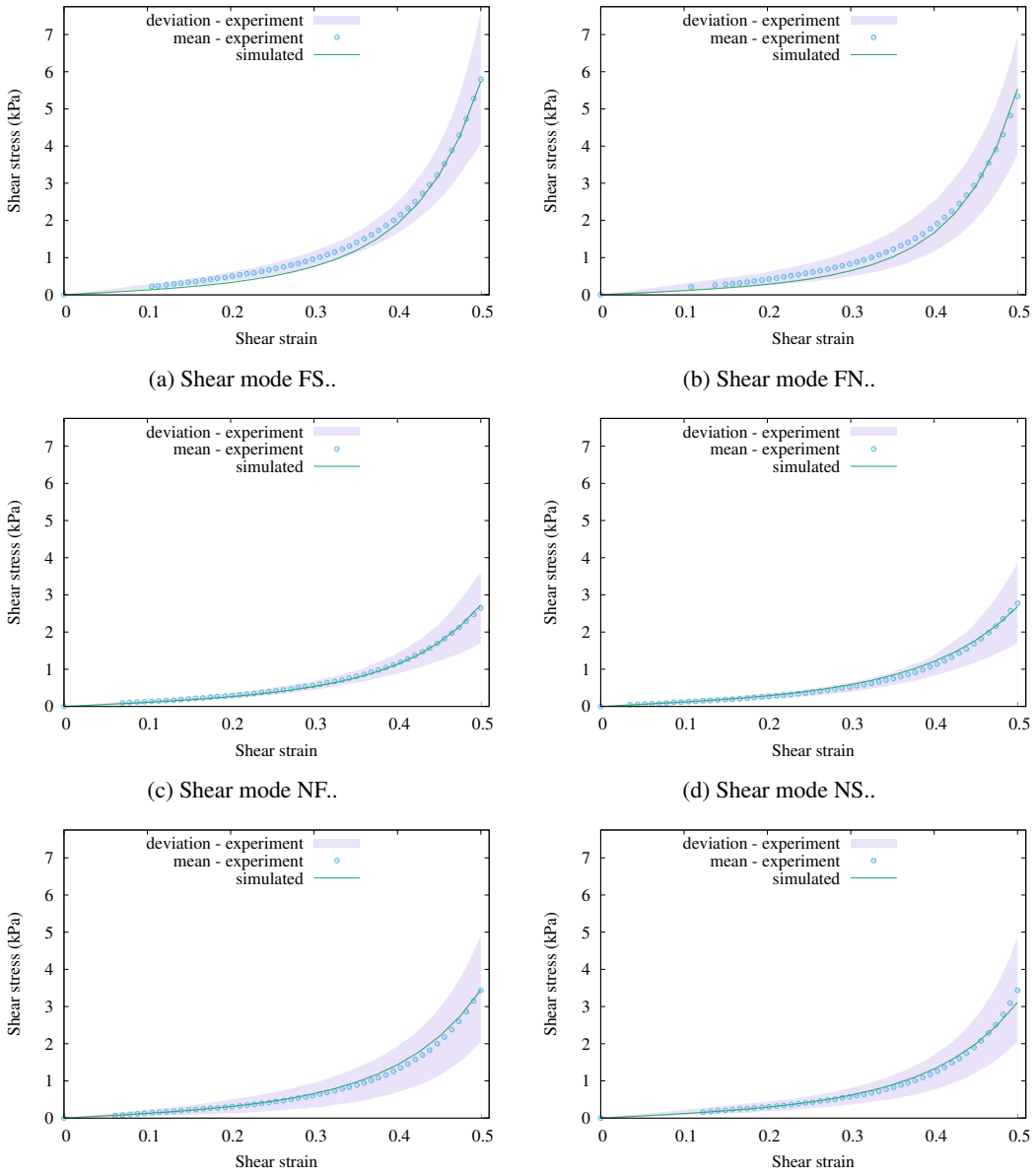
To start the identification process of the parameters, a cube of the same size used in [14], i.e 4 cm $\times$ 4 cm $\times$ 4 cm, was created. As boundary conditions, the bottom side of the cube was fixed in the

x, y and z-direction, while the top of the cube was fixed only in the z-direction to preserve the flatness of the surface of the cube during shearing. A prescribed displacement of 2 mm was applied on the top of the cube to simulate the shearing experiment. The fibre orientation of the shear specimen was also assigned to the cube. In [14], Sommer et al. recorded the change in angle of the fibre direction across the myocardium thickness and found it to be $14.8 \pm 6.9\,^{\circ}\,\mathrm{mm}^{-1}$. Using the fact that the length of the cube was 4 mm, it was determined that the total change in fibre angle across the cube was approximately 60°. The middle region of the basal region of the left ventricle, where the myocardium specimen was extracted, usually coincides with the region where the fibre angle is zero, i.e the fibre is parallel to the circumferential axis of the heart. Consequently, the change in fibre angle of the shear cube was assumed to be equally distributed across its two faces, parallel to the epicardium and endocardium, with angles of 30° and −30°, respectively. On the other hand, the sheet angle was assumed to be zero, since the cube sample was from the mid-wall in the basal region.

With the geometry and fibres set up, a starting set of material parameters was assumed. The shear cube was then modelled for all the six different modes, shown in Fig. 3.4, by adequately changing the location of the boundary conditions. Due to the non-linearity of the deformation, the prescribed displacement was applied in 20 incremental steps, and the shear stress recorded. The six shear stress-shear strain graphs were then compared to those in [14] and the summed squared error was determined. Using the latter, the Levenberg-Marquardt algorithm was employed to minimise this error. The parameters used for the LVM algorithm were the damping parameter, $\lambda = 1$, and scaling parameter, $\nu = 2$. For further details on these parameters, refer to Chapter 7. The calibrated parameters were A, a_1, a_2, a_3, a_4, a_5 and a_6. A_{comp} was not included in the list, since it was observed to cause calculation instabilities during some of the trial runs, an issue that has been confirmed in the literature by Yin et al. [94]. Consequently, it was fixed to 100 kPa. Using different starting points, the identification process was found to converge to the set of parameters given in Tab. 6.5. The resulting shear stress-shear strain curves of all shear modes were plotted with the experimental data curve, as given in Fig. 6.21.

A (kPa)	a_1	a_2	a_3	a_4	a_5	a_6
0.19	12.70	−8.36	−8.56	11.22	14.25	9.15

Table 6.5: Identified passive material parameters based on triaxial shear experiment.



(a) Shear mode FS..

(b) Shear mode FN..

(c) Shear mode NF..

(d) Shear mode NS..

(e) Shear mode SF..

(f) Shear mode SN..

Figure 6.21: Calibrated shear modes.

6.2.3 Database construction

With the material parameters defined, the database was then constructed. Three parameters were selected to represent the dimension of the parametric domain: A, θ^{epi} and θ^{endo}. Our choice of parameters was restricted to only three, due to the *curse of dimensionality* problem [178, 179]. This problem relates to the number of points that need to added to the database whenever a large dimension is considered. If a specific number of dataset, n_{dataset}, is defined per dimension, then the number of simulations required to build the database is $(n_{\text{dataset}})^d$, where d is the number of dimensions. Hence, if d increases, then $(n_{\text{dataset}})^d$ grows exponentially.

The selection of ranges for the parametric domain of our database required a different methodology because they could not be obtained directly from the literature. Parameters such as θ^{epi} and θ^{endo} were obtained from Rohmer et al. [9], as given in Tab. 6.3. However, A was yet to be found and was not a measured quantity that was readily available. In order to circumvent this problem, the end-diastolic volume of the left and right ventricles was used.

The advantage of using ventricular volumes is that they are determined from MRI and their range is well established in the literature, as indicated by Sandstede et al. [18]. As such, the calibrated materials were first applied to the bi-ventricle model and the respective end-diastolic volumes were obtained. With the parameters of Tab. 6.5, the end-diastolic volume of the left ventricle was found to be 103 mL while the right ventricle was 116 mL. Those values fit within the observed range from Sandstede et al. [18], as given in Tab. 6.6.

Ventricle	End-diastolic Volume (mL)		
	Lower	Mean	Upper
Left	81	108	135
Right	84	115	146

Table 6.6: End-diastolic volume of the left and right ventricle obtained from Sandstede et al. [18].

In order to obtain a valid bound of the stress scaling parameter, A, it was varied until the lower and upper limits of the left and right ventricular volumes were obtained. Such an approach was also applied to θ^{epi} and θ^{endo} even though their experimental data was available. With A defined using the calibrated parameter set, θ^{epi} and θ^{endo} were alternatively varied until the cavity volume bounds were obtained. Hence, the meaningful physiological range for A, θ^{epi} and θ^{endo} used to define the boundary of the database, was:

Parameter		Range	
		Lower	Upper
A	(kPa)	0.04	0.22
θ^{epi}	(°)	-48	-66
θ^{endo}	(°)	55	89

Table 6.7: Database range of A, θ^{epi} and θ^{epi}.

6.3 PODI modelling of diastolic filling with variations in material properties

With the cardiac problems defined along with the parametric ranges, the PODI simulations could be carried out. The PODI implementation used in Section 6.1 was now converted to C++ for faster and more efficient calculation. Overall, two sets of simulations were undertaken, focused on the diastolic filling phase and considering variations in the elastic material behaviour of the myocardium. A second set of

simulations dealing with multi-parametric calculation was then performed to investigate the influence of the database and MLS on the PODI accuracy.

6.3.1 Single-parametric calculation

The first example is a single parametric database where only the material parameter A in Eq. (3.31) was chosen to range from 0.04 to 0.22 at an interval of 0.0225. The total number of datasets available in the database was only 9, as shown in Tab. 6.8. For the PODI calculation, the dataset 5, where $A = 0.13$, was omitted from the database and defined as the unknown for which the solution fields had to be found. The number of datasets included in the MLS-interpolation is 4: 0.085, 0.1075, 0.1525 and 0.175, respectively. The energy level conservation was set to a minimum of 99 %.

Dataset ID	1	2	3	4	5	6	7	8	9
A	0.040	0.0625	0.085	0.1075	0.13	0.1525	0.175	0.1975	0.22

Table 6.8: Bi-ventricle model database with varying A values.

The PODI calculation, of the displacement, stress and strain solution field, was completed in 14.05 s. This was approximately 2050 times faster than the EFG simulation which, carried out in 58 incremental time steps, took about 8 h using two CPU cores. However, the PODI computation time included the reading of the database from a hard-disk medium, the snapshot calculation, and writing the results back to the files. If the snapshot calculation was looked at alone, (consisting of the arrangement of the data matrix $\mathbf{U}_i$, as stated in Sec. 5.3.3, solving the eigenvalue problem, selecting the most dominant POMs, reducing the data matrix, interpolating the coefficient, and projecting the result back into the high-dimensional space for all 58 time steps), then the total calculation time was only 0.43 s. This calculation time comprised finding the solution of all the fields such as displacement, stress and strain. If a specific one of these were studied, such as the displacement field, it was found that only 0.054 s in calculation time was then needed. Hence, this sub-second calculation speed, which translated to 1074 Hz of calculation frequency per step, showed that the PODI method can be very efficient.

However, when the eigenvalue decomposition was carried out across the steps, the most dominant mode, on average, accounted for 99.66 %, 99.70 % and 99.59 % for the displacement, stress and strain results. Hence, only one POM was found to be sufficient to represent the whole behaviour of the heart for each incremental calculation step, reducing the matrix size from 2868×4 to 1×4.

Regarding the error between the exact and interpolated results, the L_2-error norm computed for the displacement field was found to be 0.035, while those of the stress and strain results were similarly small, at 0.144 and 0.042, respectively.

Another result that is usually considered to be important in cardiac mechanics modelling, is the relationship between progressing cavity volume and pressure in the ventricles. When the approximated and reference solution curves of that relationship are compared with each other, an error of 5.87×10^{-3} is found for the left ventricle and 1.94×10^{-3} for the right ventricle. This difference indicates the good interpolation characteristics of PODI, which can be observed in Fig. 6.22 and Fig. 6.23.

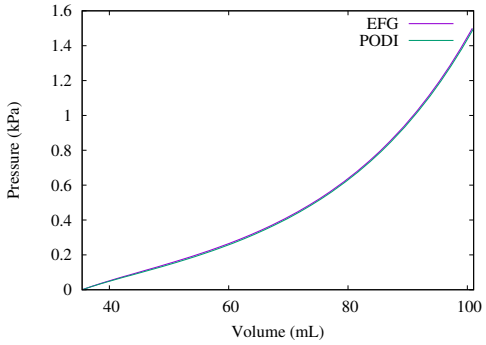


Figure 6.22: LV Pressure-Volume relationship.

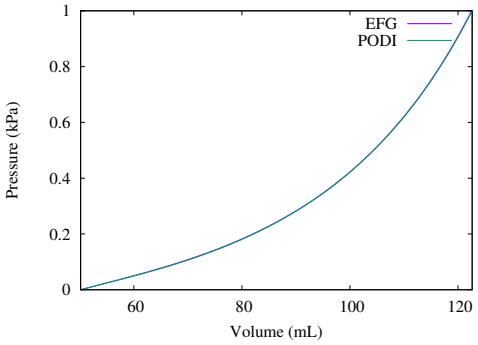


Figure 6.23: RV Pressure-Volume relationship.

The L_2 error norm of the displacement field and cavity volume and pressure curve was plotted in Fig. 6.24 for the whole domain, in order to investigate which part of it provided the most reliable results. Based on the results obtained, it was found that the centre region of the domain gave results that were about 4 to 8 times more accurate than the ones on the boundary. This effect was due to the number of supports available for the MLS interpolation. At the centre of the domain, the influence radius can span on the left and right hand side of the point of interest to mobilise more datasets. However, as the boundary is approached, the number of datasets available along that direction is lower than the one on the other side, and therefore introduces more errors during the interpolation.

Another observation relates to the non-symmetric error distribution across the domain. The errors seemed to be higher at lower A values than at higher ones. The reason for such a discrepancy was the large displacement field at smaller A value, which corresponded to lower stiffness. Therefore, interpolating between higher values induced larger errors.

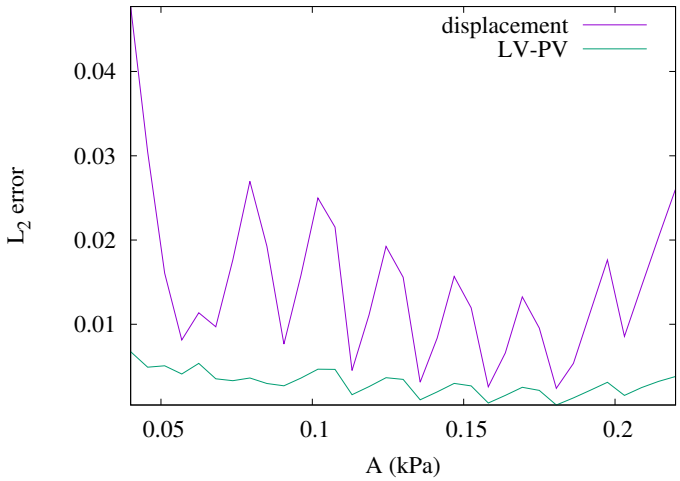


Figure 6.24: Variation of error across the database.

Finally, as discussed in Section 5.2.3, the zero-mean effect of POD within the cardiac mechanic framework was looked at. Some of the above calculations were carried out again to compare performance and accuracy. Early calculations in Section 6.1 showed that numerical issues were introduced when using

the zero-mean PODI method. These problems originate from the eigenvalue decomposition calculations. When looking at the evolution of the magnitude of the POV calculated from $\mathbf{U}_i$ across all the iterations, as given in Figs. 6.25 - 6.28, it was found that the POVs of the non-zero-mean method were larger than the zero-mean method ones. However, most importantly, negative eigenvalues were yielded, as indicated by the broken lines in Fig. 6.28. This difference is explained by the fact that, with the subtraction of the mean vector, the overall magnitude of the entries of $\mathbf{U}_i$ are lowered. However, working with these low values, which are quite often close to computer limits, can render eigenvalue decomposition algorithms such as those provided in the LAPACK library ineffective, and thus incorporate significant errors. These low values were found to heavily influence the last POMs, where their POVs were relatively small. In this case, even though the last POV was negative, the POM was invalid, with LAPACK returning a vector of "*NAN*" (not a number) values.

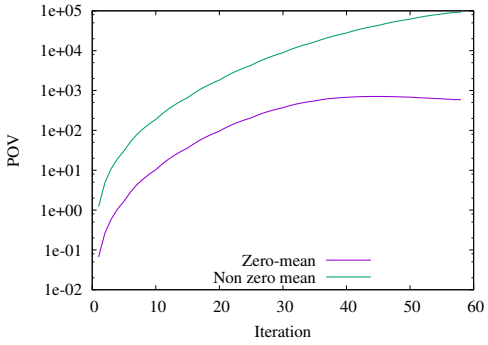


Figure 6.25: Magnitude of the 1st POV across the iterations.

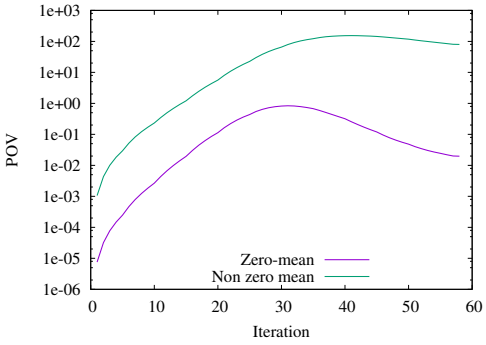


Figure 6.26: Magnitude of the 2nd POV across the iterations.

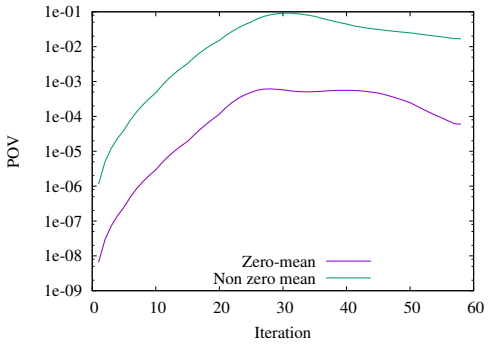


Figure 6.27: Magnitude of the 3rd POV across the iterations.

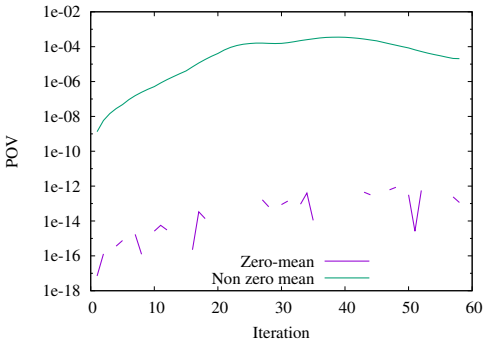


Figure 6.28: Magnitude of the 4th POV across the iterations.

To fix the issue of invalid POVs and POMs while still using the zero-mean method, a cut-off criterion is imposed on the outputs of LAPACK. In the literature [72], this cut-off criterion is used where it is set to 1×10^{-7}. In this research, the limit was set to zero and as such, any eigenvalue falling below zero was discarded, along with their corresponding eigenvector. This resulted in some steps having their total energy being computed from only three POVs. This cut-off criteria measure was not found to heavily impact the calculation performance and accuracy. Therefore, the next step was to proceed with the analytical comparison between the non-zero-mean and zero-mean method. The error norm was the first

result investigated. Overall, the zero-mean error proved to be more accurate than the non-zero-mean error. For low energy range, i.e. between 80 % to 99 %, the error dropped sharply by 55 %. However, between 99 % and 100 % of energy, the drop was less steep, with the error falling from 3 % to 0 %, as shown in Fig. 6.29. This exponential drop in error experienced by the non-zero-mean calculation was similarly observed in Luo et al. [180] when applying POD to FEM. From there, it could be concluded that there is a decrease in difference between the zero-mean and non-zero-mean error as the energy conserved increased. This observation seems to be in accordance with the results provided in the benchmark problem example in Section 6.1, and could possibly imply that the first few POMs generated by the zero-mean method were more accurate. But for the last few POMs, it seemed to be less able to mobilise the behaviour of the problem from the selected datasets and was therefore more error-prone. This effect was found to be the opposite for the non-zero-mean method, where the first POMs led to large computational errors, while the last few were more accurate and therefore reduced the error substantially. However, in the end, when 100 % energy was conserved, both methods gave the same error, showing that both sets of POMs mobilised the same behaviour overall.

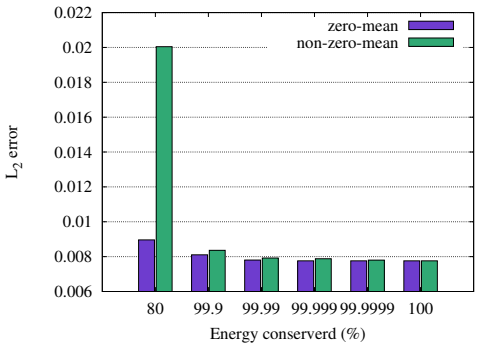


Figure 6.29: Difference in error between zero-mean and non-zero-mean displacement field results for different conserved energies.

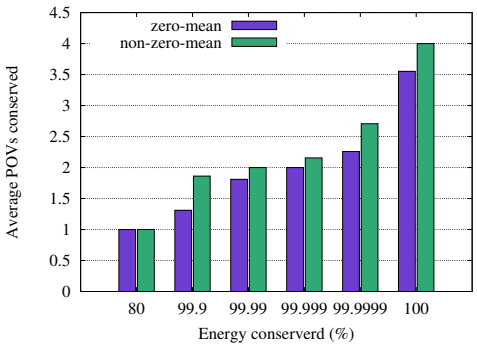


Figure 6.30: Difference in average number of POVs conserved between zero-mean and non-zero-mean displacement field results for different conserved energies.

In terms of POVs conserved, it was noted that, as observed in the benchmark example, the zero-mean method conserved about 14 % less POVs on average than the non-zero-mean method, as shown in Fig. 6.30. Even though the number of POMs was lower during the PODI calculation, a better accuracy was achieved, as described earlier. This contrast seems to reinforce the opinion that the dominant POMs of the zero-mean method are of higher quality that those of the non-zero-mean ones.

Even though the number of POVs conserved was less, the zero-mean calculations consistently took 3 % longer than the non-zero-mean method, which is believed to be due to the implementation of the zero-mean method. This value was much smaller than those recorded in the benchmark problem. For the displacement field calculation and other field computation results, this increase translated to 1.7×10^{-3} s and 1.3×10^{-2} s respectively. Due to these magnitudes being very small, they were considered negligible.

From the above results that consisted of comparing the zero-mean with the non-zero-mean methods, it was observed that the redefinition of the POVs and POMs led to significant changes in the PODI method's behaviour. This observation is in line with the observations made in Section 6.1. The lower magnitude of the POVs, which led to an increase in energetic importance of the less dominant POMs, was in agreement

with the work of Tamura et al. [159] and Cadima and Jolliffe [160]. This effect can be attributed to the subtraction of the mean vector from the data matrix $\mathbf{U}_i$ which therefore leads to a decrease in the overall magnitude of the entries of the matrix. For a more in-depth mathematical explanation, the reader is referred to the work of Cadima and Jolliffe [160]. Overall, it is believed that the gain in accuracy of the zero-mean method outweighs the increase in calculation time. As such, the zero-mean method is found to be better suited for cardiac modelling and was used for the subsequent PODI calculations.

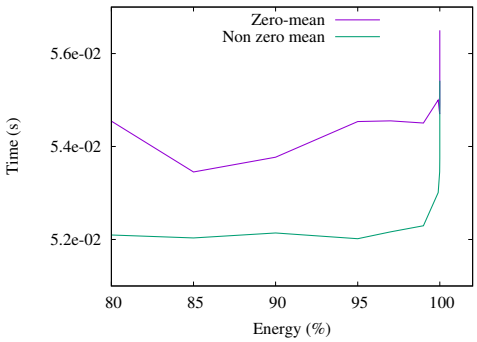


Figure 6.31: Difference in PODI calculation time between zero-mean and non-zero-mean displacement field results for different conserved energies.

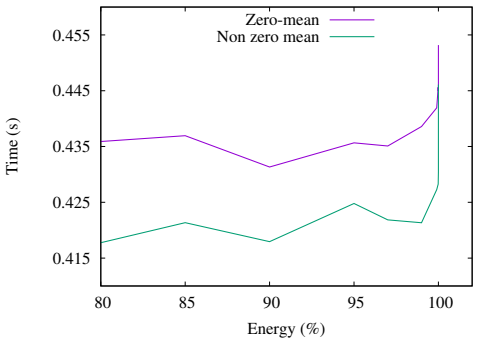


Figure 6.32: Difference in PODI calculation time between all zero-mean and non-zero-mean results for different conserved energies.

6.3.2 Multi-parametric PODI calculation

As a second example, we considered the BV model again, but the dimension of the parametric domain $\mathcal{M}$ in this case was increased to three, including in addition to A the fibre angles at the epicardium and endocardium, f_{epi} and θ_{endo}, respectively. The database was extended accordingly by corresponding datasets, and the MLS approximants used in the PODI-method were set up in three dimensions, referring to A, θ_{epi} and θ_{endo}.

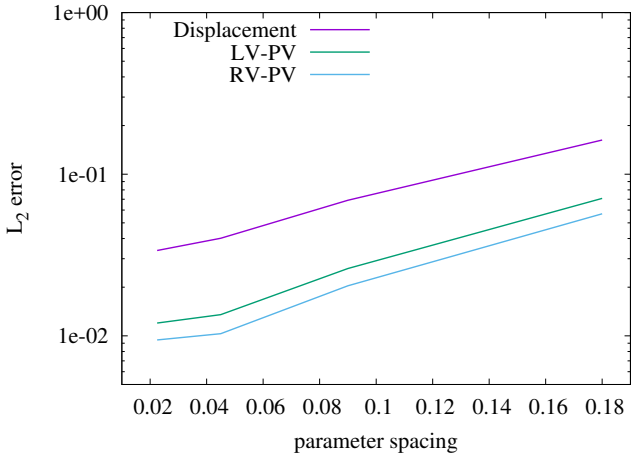
To start, a database sensitivity analysis was carried out to determine how the PODI-solution error with respect to specific parameters was influenced by (i) different discretisation levels of the domain $\mathcal{M}$ and (ii) different influence domain sizes and number of supporting datasets used in the MLS approximation.

The parametric ranges for this sensitivity analysis are given in Tab. 6.7. To construct the database, four levels of discretisation refinement were considered, as listed in Tab. 6.9.

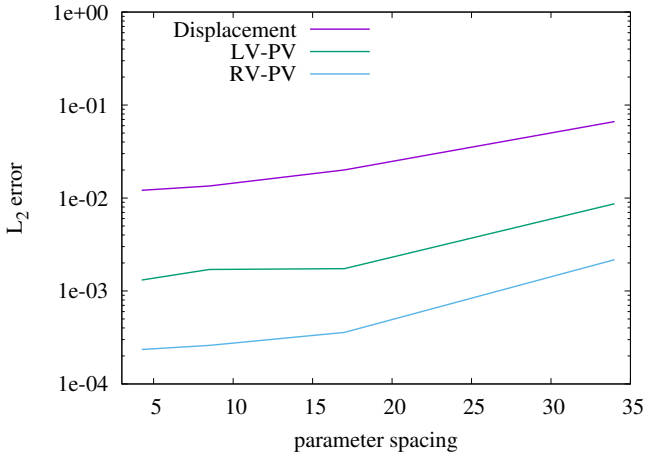
Level	Spacing, s			Number of datasets
	A (kPa)	θ_{epi} (°)	θ_{edo} (°)	
1	0.18	18	34	8
2	0.09	9	17	27
3	0.045	4.5	8.5	125
4	0.0225	2.25	4.25	729

Table 6.9: Parametric discretisation levels.

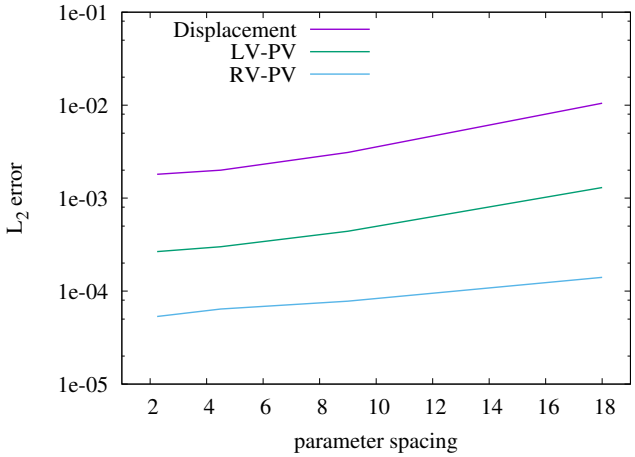
The first objective was to study the influence of the parameteric discretisation level on the accuracy of PODI using only the minimum required MLS influence radius, where the latter was dictated by the invertibility requirement of the moment matrix Eq. (5.57). To achieve this, the influence radius was continuously lowered to meet the required number of neighbours which were 6, 2 and 6 for A, θ_{epi} and θ_{endo}. For each level of discretisation, the PODI calculations were carried out for a bi-ventricle problem using $A = 0.13$, $\theta_{\text{epi}} = -57^\circ$ and $\theta_{\text{endo}} = 72^\circ$. This parameter choice referred to a dataset point that lay in the middle of the parametric domain of our database.



(a) Error sensitivity of parameter A with respect to dataset density: $\frac{\Delta L_2}{\Delta s} = [0.29, 1.04]$ for displacement, $[0.07, 0.5]$ for LV-PV and $[0.04, 0.41]$ for RV-PV.



(b) Error sensitivity of parameter θ_{endo} with respect to dataset density: $\frac{\Delta L_2}{\Delta s} = \left[3.16 \times 10^{-4}, 2.72 \times 10^{-3}\right]$ for displacement, $\left[9.16 \times 10^{-5}, 4.08 \times 10^{-4}\right]$ for LV-PV and $\left[6.08 \times 10^{-6}, 1.06 \times 10^{-4}\right]$ for RV-PV.



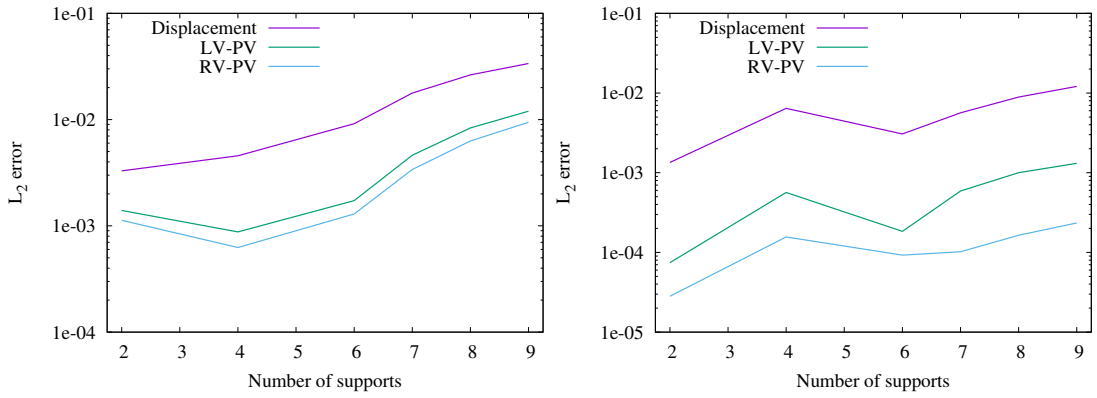
(c) Error sensitivity of parameter θ_{epi} with respect to dataset density: $\frac{\Delta L_2}{\Delta s} = \left[8.22 \times 10^{-4}, 8.46 \times 10^{-5}\right]$ for displacement, $\left[1.49 \times 10^{-5}, 9.54 \times 10^{-5}\right]$ for LV-PV and $\left[4.84 \times 10^{-6}, 6.93 \times 10^{-6}\right]$ for RV-PV.

Figure 6.33: Plots of L_2-error norm in respect of parametric spacing.

From Fig. 6.33, it can be readily seen that the L_2-error norms continuously decrease along with the discretisation spacing, s, which is expected and observed when the MLS approximation technique is conventionally applied to a spatial domain.

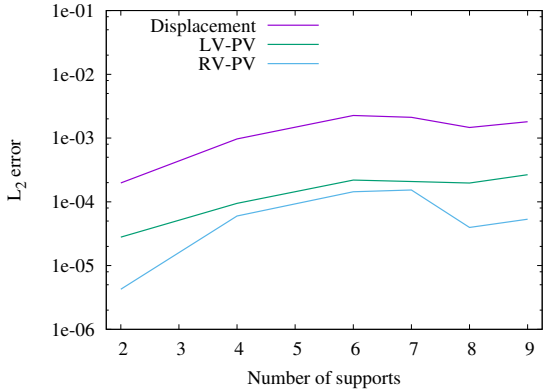
Clearly, as the supporting parameter sets get closer to the unknown parametric set, the quality of the PODI-solution increases. The next observation made concerned the magnitude of the errors for each parameter. For this particular database set up, it was found that θ_{epi} and θ_{endo} were less likely to have a major impact on the error as compared to the stress scaling coefficient, A, the reason being that both former parameters already had low error norms, which tended to stagnate as the discretisation level increased, while the error drop was more pronounced for the fibre-angle parameters.

The second objective aimed at finding the optimum number of support required for the PODI algorithm to produce solutions with a low error magnitude. Consequently, the database consisted of datasets with the lowest discretisation spacing in order to have a sufficiently large number of supporting parameter sets to track the support-size L_2-error norm relationship. In order to keep the number of supporting particles fixed, the influence radius was continuously re-adjusted until the desired number of particles participating in the PODI calculation were found.



(a) Parameter: A.

(b) Parameter: θ_{endo}.



(c) Parameter: θ_{epi}.

Figure 6.34: Plots of L_2-error norm with respect to number of supports.

The results given in Fig. 6.34 clearly show that 2 is the optimal size of support that would yield PODI solutions errors of less than 1×10^{-1}. Accordingly, one could also expect fast calculation speed, because

the number of involved datasets was low. Note that the usage of only two supporting particles to compute the MLS-approximants can be carried out, as the basis polynomial in the MLS-scheme is of an order of 1. Higher polynomial orders could also have been investigated, but this would negatively impact on the calculation time, as more supporting particles would be needed to satisfy the invertibility requirement of the MLS moment matrix. Even so, if the parametric dimension needed to be increased, this would lead to a larger number of unknown polynomial coefficients in Eq. (5.58).

After carrying out the sensitivity analysis, a new database was created. The level 4 discretisation refinement (see Tab. 6.9) was used for each dimension of the parametric domain and different parameter combinations were considered. As before, the problem-at-hand corresponded to a bi-ventricle with $A = 0.096\,25\,\text{kPa}$, $\theta_{\text{epi}} = -53.625°$ and $\theta_{\text{endo}} = 65.625°$.

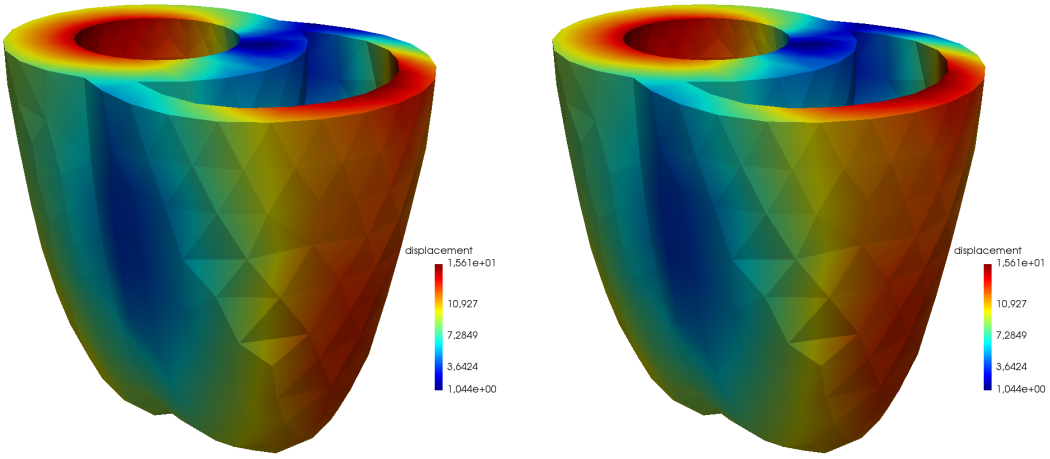
The PODI calculation was carried out with the datasets selected by the algorithm, as listed in Tab. 6.10. The algorithm represented local support, which provided the best possible interpolation quality. Starting with the displacement field, the L_2-error norm of value 3.03×10^{-3} was found to be less than that obtained for the single parametric PODI-calculation in Section 6.3.1. A visual inspection of the PODI and EFG result revealed that they were both almost identical, as shown in Fig. 6.35. For this PODI calculation, it was determined that three of the eight POMs that were generated (shown in Fig. 6.36), needed to be retained to conserve more than 99 % of the energy. According to Fig. 6.37, which shows the distribution of the magnitude of the POVs, the three most dominant modes accounted for 56.13 %, 42.32 % and 1.54 % of the energy of the system.

With regard to the other results, the stress field error of value 1.38×10^{-2} was found to be equally low in magnitude and almost the same as the single parametric PODI result, whereas the strain result error of 5.94×10^{-3} was considerably lower. Finally, the cavity pressure volume evolution shown in Fig. 6.38 exhibited an error of 1.64×10^{-3} for the left ventricle, again lower than that obtained in Section 6.3.1. In summary, these errors can be considered negligible due to their already low magnitude.

In terms of calculation time, all solution sets were obtained within ~0.88 s. Regarding individual results, the displacement field solution was computed within 0.11 s (9.1 Hz) for all 100 incremental time steps, as previously used in the EFG simulation. If only one simulation time step were considered, then the PODI-calculation would take only 1.1×10^{-3} s. This is equivalent to about 909 Hz, which is within the operational frequency of 500 - 1000 Hz of haptic devices [30].

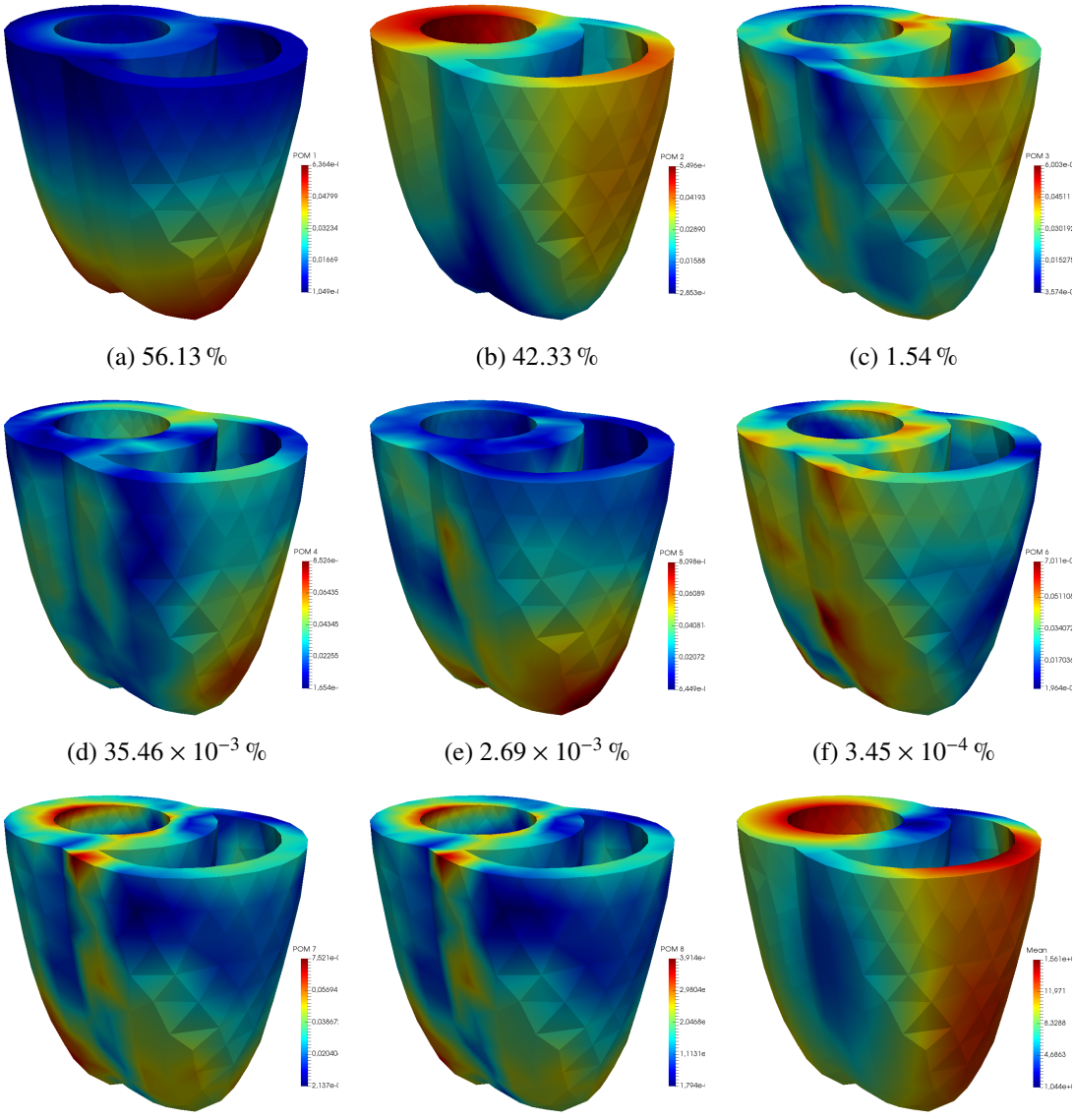
Parameter	Supports							
	1	2	3	4	5	6	7	8
A (kPa)	0.085	0.085	0.085	0.085	0.1075	0.1075	0.1075	0.1075
θ_{epi} (°)	-52.5	-52.5	-54.75	-54.75	-52.5	-52.5	-54.75	-54.75
θ_{endo} (°)	63.5	67.75	63.5	67.75	63.5	67.75	63.5	67.75

Table 6.10: Parameters of supporting datasets.



(a) PODI Result. (b) EFG Result.

Figure 6.35: Displacement field results.



(a) 56.13 % (b) 42.33 % (c) 1.54 %

(d) 35.46×10^{-3} % (e) 2.69×10^{-3} % (f) 3.45×10^{-4} %

(g) 1.76×10^{-6} % (h) 2.90×10^{-14} % (i) Mean

Figure 6.36: Plot of all eight POMs specifying the associated energy contributions.

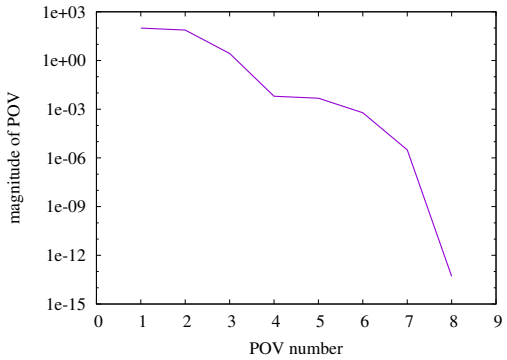


Figure 6.37: Distribution of the magnitude of POVs.

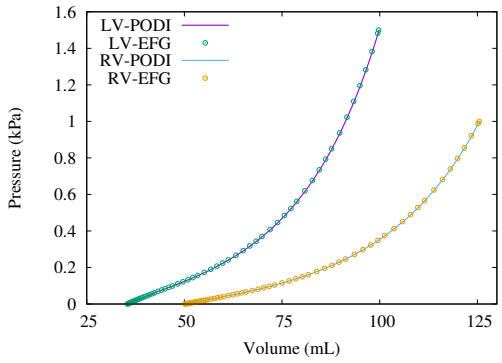


Figure 6.38: Cavity pressure-volume curve generated from the EFG and PODI calculation.

6.4 PODI modelling of diastolic filling with variations in heart anatomy

With more and more research focusing on patient-specific cardiac modelling, the application of PODI is adapted correspondingly. Under realistic conditions, the database is expected to be filled with datasets of healthy or diseased hearts corresponding to people of different age and sex groups. Thus, each heart is unique in terms of geometrical shape. However, across all hearts, the same behaviours are expected to be extracted by POD. With the uniqueness of each heart geometry, the mesh configuration in terms of number of nodes, nodal coordinates, and number of elements, is believed to vary considerably. This consequently leads to an inadequate set up of the $\mathbf{U}_i$ matrix, as discussed in Section 5.4. To resolve this issue, the Degree of Freedom Standardisation procedure is employed. Two standardisation procedures were considered in this research. The cube template standardisation and the heart template standardisation.

6.4.1 Cube template standardisation

The application of the cube template standardisation process was then investigated. The problem looked at was a bi-ventricle geometry with the same boundary conditions as given in Section 6.2. However, in this case, different geometrical mesh configurations were considered. Even though the element type was kept the same, i.e, tetrahedral, their sizes were varied. Hence, the total number of nodes was different across datasets. Overall, four different mesh discretisations were generated, as given in Tab. 6.11.

Models	BV-1	BV-2	BV-3	BV-4
Nodes	882	902	912	922

Table 6.11: BV with different mesh discretisations.

To define the database, BV-1, BV-2 and BV-4 were used. BV-3 is the geometry for which the unknown solution was computed. An additional step was undertaken in order to harden the difficulty of this registration process and increase the number of datasets. A perturbation was applied to each mesh configuration such that the geometrical shape of each heart was different. The perturbation was kept

minor as it was expected that, in a realistic scenario, hearts will be chosen from databases which are assumed to have similar properties in terms of features such as age, gender, fitness, etc. The applied perturbation was a result of two forms of transformation which were applied as *rotation* and *scaling*.

Rotation: A rotation is applied along each major axis, i.e. x, y and z, separately. The angle of rotation varied linearly from the apex to the base of the heart within the range of 0° and 20°. The equations used are:

$$\theta_x = \frac{C_x - C_x^{\min}}{C_x^{\max} - C_x^{\min}} \times 20^\circ \qquad \theta_y = \frac{C_y - C_y^{\min}}{C_y^{\max} - C_y^{\min}} \times 20^\circ \qquad \theta_z = \frac{C_z - C_z^{\min}}{C_z^{\max} - C_z^{\min}} \times 20^\circ \tag{6.1}$$

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix} \quad R_y = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \quad R_z = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.2}$$

C_x, C_y and C_z are the spatial coordinates of the mesh node, $\mathbf{C}$. $C_{(\cdot)}^{\max}$ and $C_{(\cdot)}^{\min}$ are the largest and smallest coordinate among all nodes in the dataset mesh configuration. θ_x, θ_y and θ_z are the angles of rotation and R_x, R_y and R_z are the corresponding rotational matrices.

Scaling: A scaling transformation was also used to change the size of the hearts. This was applied by multiplying each coordinate by a scaling coefficient:

$$C_x^s = s_x C_x \qquad C_y^s = s_y C_y \qquad C_z^s = s_z C_z \tag{6.3}$$

where C_x, C_y and C_z are the coordinates, C_x^s, C_y^s and C_z^s are the transformed coordinates and s_x, s_y and s_z are the corresponding scaling coefficient. The hearts selected from the database were considered to be very similar. As such, the scaling parameters were chosen to be quantitatively small. Hence, ±2.5 % was assigned to s_x, s_y and s_z to produce volumetrically larger or smaller hearts.

To ensure a diversity in geometries, combinations of different rotation angles, rotation axes and scaling values were utilised. These transformations were then applied to BV-1, BV-2 and BV-4. A total of nine unique hearts were produced in order to build up a single parametric database, where each unique heart was then assigned randomly to a stress scaling coefficient value given in Tab. 6.8. Fig. 6.39 gives a summary of the different hearts generated and their respective A values.

The transformation used in this research, influenced only the nodes and did not conserve the mesh elements definition. Due to the non-rigid transformation across the geometry, nodes can swap across an element, leading to poor or negative Jacobian elements, and overall, to an overall degradation of the neighbouring elements as well. As such, a check was carried out on all deformed geometries by comparing specific mesh quality characteristics, such as minimum angle and quality, which reflected the element's shape. The quality characteristics were compared using their minimum, averaged and maximum values recorded across all elements. These values are tabulated in Tab. 6.12.

Model	Min angle			Shape		
	Min	Mean	Max	Min	Mean	Max
BV-1	18.39	49.9283	77.8076	0.347837	0.836159	0.999987
BV-1A	15.5634	49.0237	80.2624	0.271707	0.821095	0.997955
BV-1B	16.6777	49.4234	77.1712	0.354342	0.827918	0.99911
BV-1C	14.5702	49.832	77.839	0.357583	0.833783	0.999113
BV-2	16.6556	49.4919	77.5505	0.308307	0.831383	0.999994
BV-2A	13.5433	48.646	80.4202	0.26639	0.817224	0.997948
BV-2B	14.218	48.9233	76.8646	0.298402	0.822857	0.999048
BV-2C	15.2369	49.3633	78.3148	0.317643	0.828224	0.999191
BV-3	7.24251	50.0013	77.7868	0.243803	0.836502	0.999974
BV-3A	5.11933	49.1242	79.7777	0.186757	0.821001	0.997232
BV-3B	8.21837	49.4605	76.8688	0.261851	0.828585	0.998768
BV-3C	8.12451	49.8763	78.7736	0.265335	0.833684	0.99946

Table 6.12: Mesh quality details for each perturbed BV geometry.

From the mesh quality results, it was found that in most cases, the minimum angle and the shape quality decreased slightly from the transformation, which was interpreted as a degradation of the mesh. However, those degradations were limited. No negative angles were recorded and the shape quality was always greater than zero. Hence, the perturbed meshes could be used for the rest of this research.

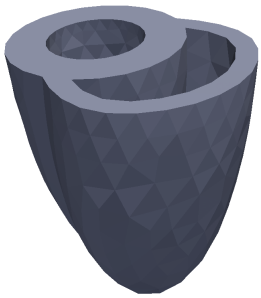
The problem that needed to be solved for was another perturbed heart geometry, created from model BV-3. A rotation of 20° was applied about its y-axis, while the whole geometry was scaled down by 2.5 %, thus resulting in a mesh configuration, shown in Fig. 6.40. The problem setup presented in this section is similar to the one given in Section 6.3.1. Following the same procedure, the dataset with $A = 0.13$ was excluded from the database. The perturbed heart geometry created from BV-3 was then assigned with an excluded stress scaling coefficient, while the PODI calculation was set to a targeted energy level of 99 %.

The last step was to set up the cube grid. The cube was constructed in such a way that it encompassed all the hearts, hence ensuring that all data would be captured by the grid. In order to obtain the dimensions of the cube, the maximum and minimum coordinates were extracted from the hearts stored in the database and the problem-at-hand. A cube was then created from those extracted dimensions and discretised with a constant spacing along every direction, resulting in a total of 294 template nodes, as shown in Fig. 6.41.

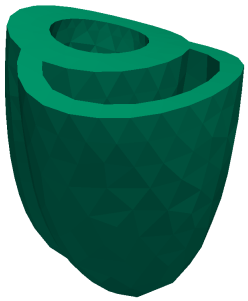
(a) Model BV-1A: A = 0.04 kPa, rotation = R_x, scale = −2.5..

(b) Model BV-3A: A = 0.0625 kPa, rotation = R_x, scale = −2.5..

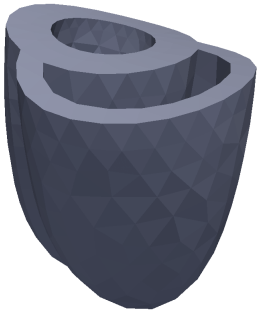
(c) Model BV-2A: A = 0.085 kPa, rotation = R_x, scale = 2.5..



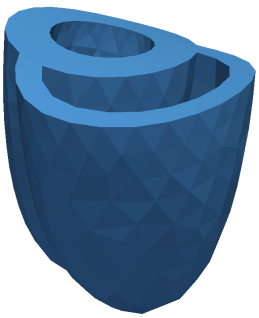
(d) Model BV-3C: A = 0.1075 kPa, rotation = R_z, scale = −2.5..



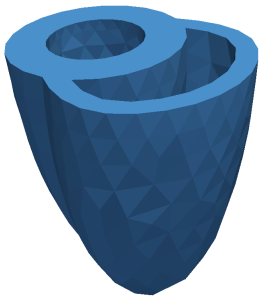
(e) Model BV-2B: A = 0.13 kPa, rotation = R_y, scale = −2.5..



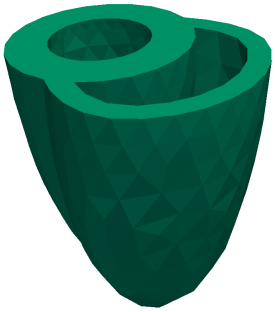
(f) Model BV-3B: A = 0.1525 kPa, rotation = R_y, scale = −2.5..



(g) Model BV-1B: A = 0.175 kPa, rotation = R_y, scale = 2.5..



(h) Model BV-1C: A = 0.1975 kPa, rotation = R_z, scale = −2.5..



(i) Model BV-2C: A = 0.22 kPa, rotation = R_z, scale = 2.5..

Figure 6.39: Perturbed heart geometries. Their number of nodes, their transformation details and stress scaling coefficients are given below each heart.

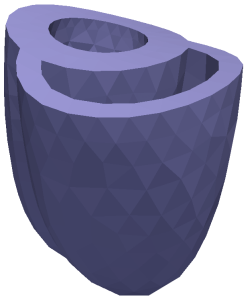


Figure 6.40: For a new heart geometry, A = 0.13 kPa: nodes = 912, rotation = R_y, scale = −2.5.

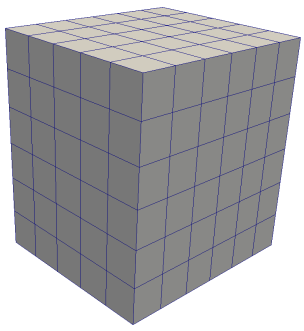
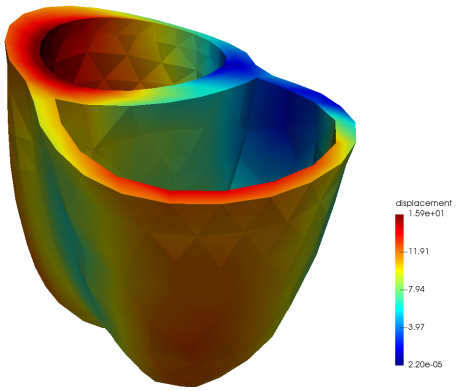


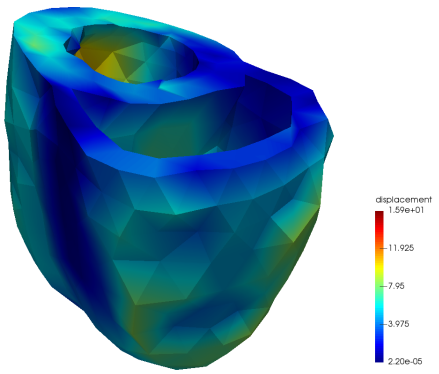
Figure 6.41: Cube grid with 294 nodes.

With the problem setup completed, the PODI calculation was carried out. It took on average a total time of about 4 min. This calculation time included reading the dataset from the database, projecting the results from the hearts to the grid, setting up interpolants, carrying out the PODI calculation, projecting the PODI results from the grid back to the heart of the problem-at-hand, and post-processing the results. The costliest operation was projecting the results from the dataset heart geometries to the template grid, and from the grid back to the heart geometry of the problem-at-hand. This procedure took about 3.80 min, that is, 95 % of the total calculation time. On the other hand, the reduced order calculation took just 0.26 s, where the displacement field computation lasted for only 0.029 s for all 58 time steps. This shows again that sub-second calculations can be achieved, as the displacement calculation frequency was about 2000 Hz. This calculation frequency was higher than the one recorded in Section 6.3.2 since the number of template nodes was fewer here, resulting in smaller $\mathbf{U}_i$ matrices.

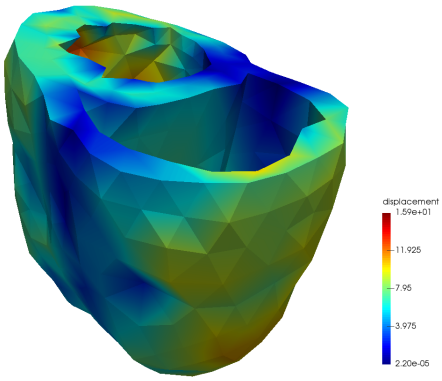
The L_2-error norm was found to be 0.46 for the displacement field, 0.89 for the strain field, and 0.98 for the stress field. For the pressure-volume relationship curve, the errors were 0.23 and 0.22 for the left and right ventricle. Even though these errors were higher than those presented in Section 6.3.1, they were still within acceptable ranges. However, when the final deformation was plotted, it was found that non-physical deformations were present. The solution fields, such as the displacement field shown in Fig. 6.42b, were not smooth, as opposed to the full-scale simulation in Fig. 6.42a. Such observations are found to be more dominant over the geometrical surface. The main reason why those non-smooth solution fields were obtained is because, as explained in Section 5.4.1, template nodes that do not contain any dataset standardised solution fields are used during the projection process. These template nodes have, instead, an assigned zero value and consequently, their contribution during the projection process leads to solution fields of low magnitude, as noticed on the surface nodes in Fig. 6.42b.



(a) EFG..



(b) PODI results from cube grid made up of 294 nodes..



(c) PODI results from cube grid made up of 990 nodes..

Figure 6.42: Displacement field solution at end-diastole.

Cube grid	CG-1	CG-2	CG-3	CG-4	CG-5	CG-6
Nodes	294	336	448	504	720	990

Table 6.13: Cube grid with different mesh discretisations.

An attempt was made to increase the number of grid nodes such that the negative effect of the poor surface interpolation could be decreased as the higher number of grid nodes, located inside the dataset hearts, would be able to raise the surface solution field's accuracy. To do so, five other grids with different grid node densities, were considered, as shown in Tab. 6.13.

The obtained results showed an overall increase in the accuracy of the PODI solutions, as given in Fig. 6.43. The displacement field was about 22 % more accurate, while the LV pressure-volume curve error dropped significantly, by 26 %, as the grid discretisation reached 990 nodes. A cut through the BV models confirmed the drop in error, as shown in Fig. 6.45, with the geometrical inner nodes showing improvements, while the lateral wall of the right ventricle showed more expansion. However, when the deformed state was investigated, the localised non-physical deformations were still dominant on the surfaces of the geometry and the magnitude of the displacement field was considerably lower, as shown

in the contour plot of Fig. 6.42c. The only region that showed improvements was the epicardium surface of the right ventricle.

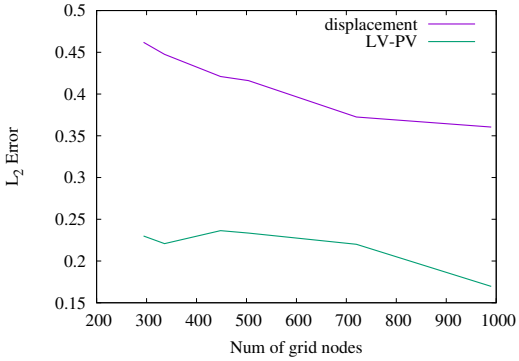


Figure 6.43: Change in L_2-error norm as the number of grid nodes increases.

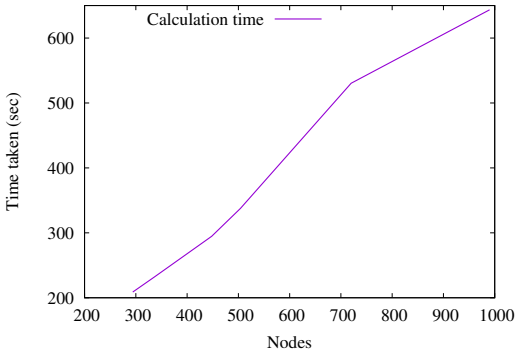
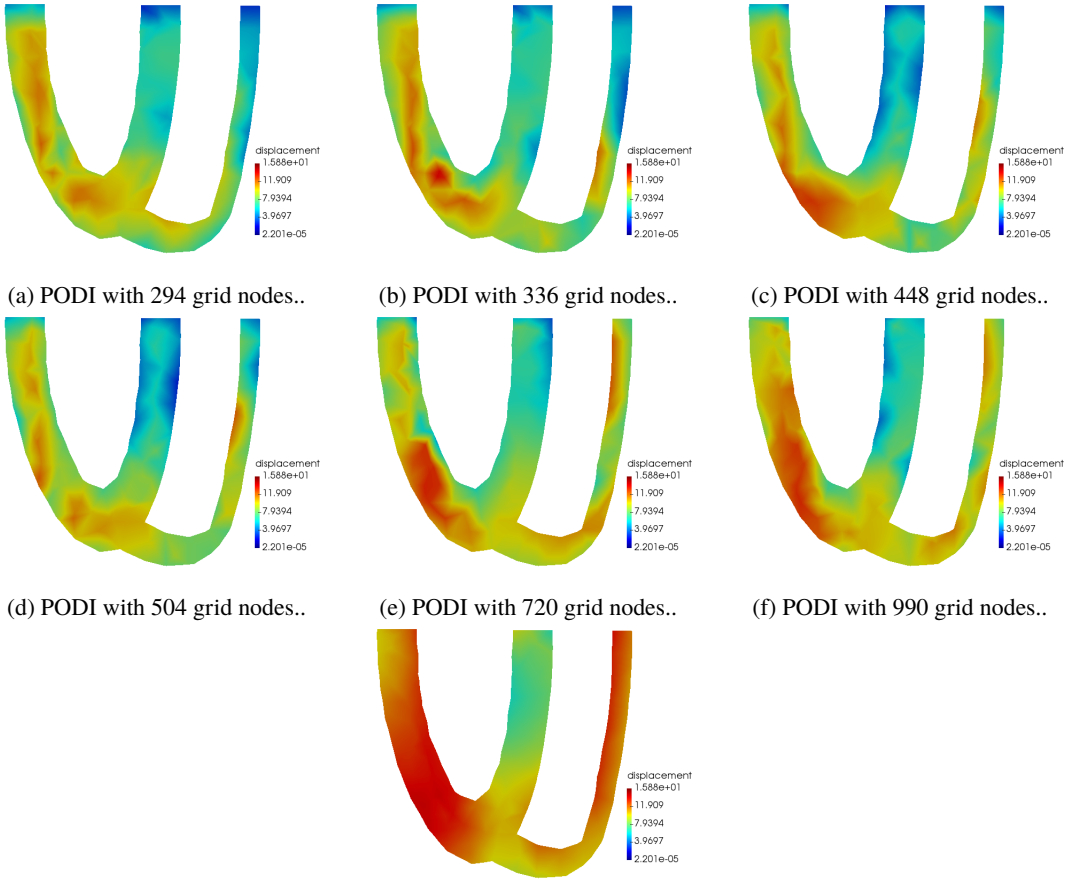


Figure 6.44: Change in calculation time as the number of grid nodes increases.

Regarding the PODI calculation time, it was found to be increasing almost linearly, since more grid nodes were present, as shown in Fig. 6.44. This was due to the fact that the PIP algorithm and the template projections required more time.

The three most dominant POMs were found to be needed in order to reach the minimum specified energy limit for the displacement, strain and stress field data. Further investigation of the POVs revealed that the energy of the first, second and third POM was about 61.6 %, 30.5 % and 7.8 % respectively. The fourth POM was below 1.04×10^{-14} %. As such, it could be deduced that POD struggled to find the most dominant mode since the energies were more scattered, as opposed to those observed in Section 6.3.1, where the first POM already accounted for about 99 % energy. This scattering effect showed that the data of the $\mathbf{U}_i$ matrix are gathered from disorganised data, which could have arisen from the introduction of the zero entries, therefore leading to poor results. Consequently, an alternative method, where the zero entries are omitted, is explored in the next section.



(a) PODI with 294 grid nodes..

(b) PODI with 336 grid nodes..

(c) PODI with 448 grid nodes..

(d) PODI with 504 grid nodes..

(e) PODI with 720 grid nodes..

(f) PODI with 990 grid nodes..

(g) EFG..

Figure 6.45: PODI Displacement field plot based on the cube template standardisation method for different numbers of grid nodes.

6.4.2 Heart template standardisation

An alternative approach to standardise the degrees of freedom, is to make use of a heart template. The major advantage expected to be gained from the new template was the elimination of both the incompatible columns and zero entries problems in the matrix $\mathbf{U}_i$. This was achieved by first registering every patient-specific heart onto a template heart before projecting the dataset's results field onto the template nodes. The dataset geometry and template geometry shared the same anatomical features such as the left and right ventricle, the apex, base surface, left and right ventricular wall and the septum wall. Consequently, any template node should be located on the dataset's geometry after the registration process.

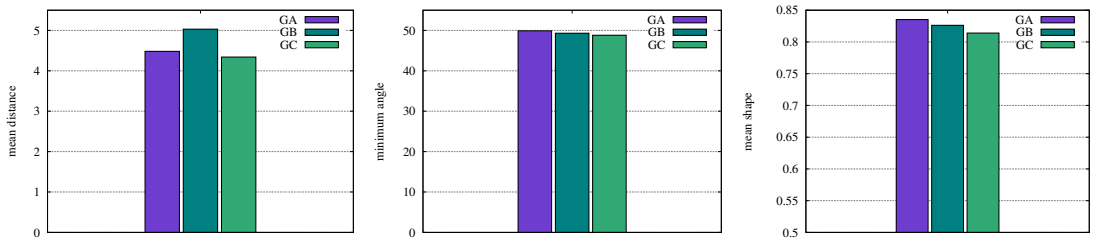
The heart template standardisation required an additional step to the cube template standardisation given in Section 6.4.1. This additional step involved the registration of the patient-specific heart to the template heart. The registration process occurred at two instances during the PODI calculation: firstly, before the dataset solutions were projected on the template nodes, and secondly, before the reverse projection of the PODI results onto the problem-at-hand nodes. The registration method used was the Coherent Point Drift algorithm, as introduced in Section 5.4.2.

Parameter	β^{CPD}	λ^{CPD}	w^{CPD}
Value	2.5	0.7	0.7

Table 6.14: Coherent point drift algorithm parameter values.

In order to elaborate on this new approach, the problem configuration, introduced in Section 6.4.1, was used. Before the actual PODI could be initiated, the geometry registration of all the datasets and the problem-at-hand was carried out on a bi-ventricle template heart, similar to Fig. 6.18, with the help of the CPD algorithm. In this case, the number of nodes on the template heart was 347. For all the CPD calculations, the parameters found in Section 5.4.2 and given in Tab. 6.14, were used. The registration of each heart took on average 17 s for 285 iterations. The mapped geometries are given in Figs. 6.47 - 6.49. The quality of the registration and resulting mesh were afterwards checked to ensure that the elements of the mapped geometries had not substantially degraded. For that, three groups of quantitative results were considered: GA, GB and GC. GA represented the mesh quality of the original unperturbed meshes, BV1, BV2 and BV4, which were used as a baseline, while GB referred to the meshes perturbed by rotation and translation. Finally, GC represented the quality of the registered meshes.

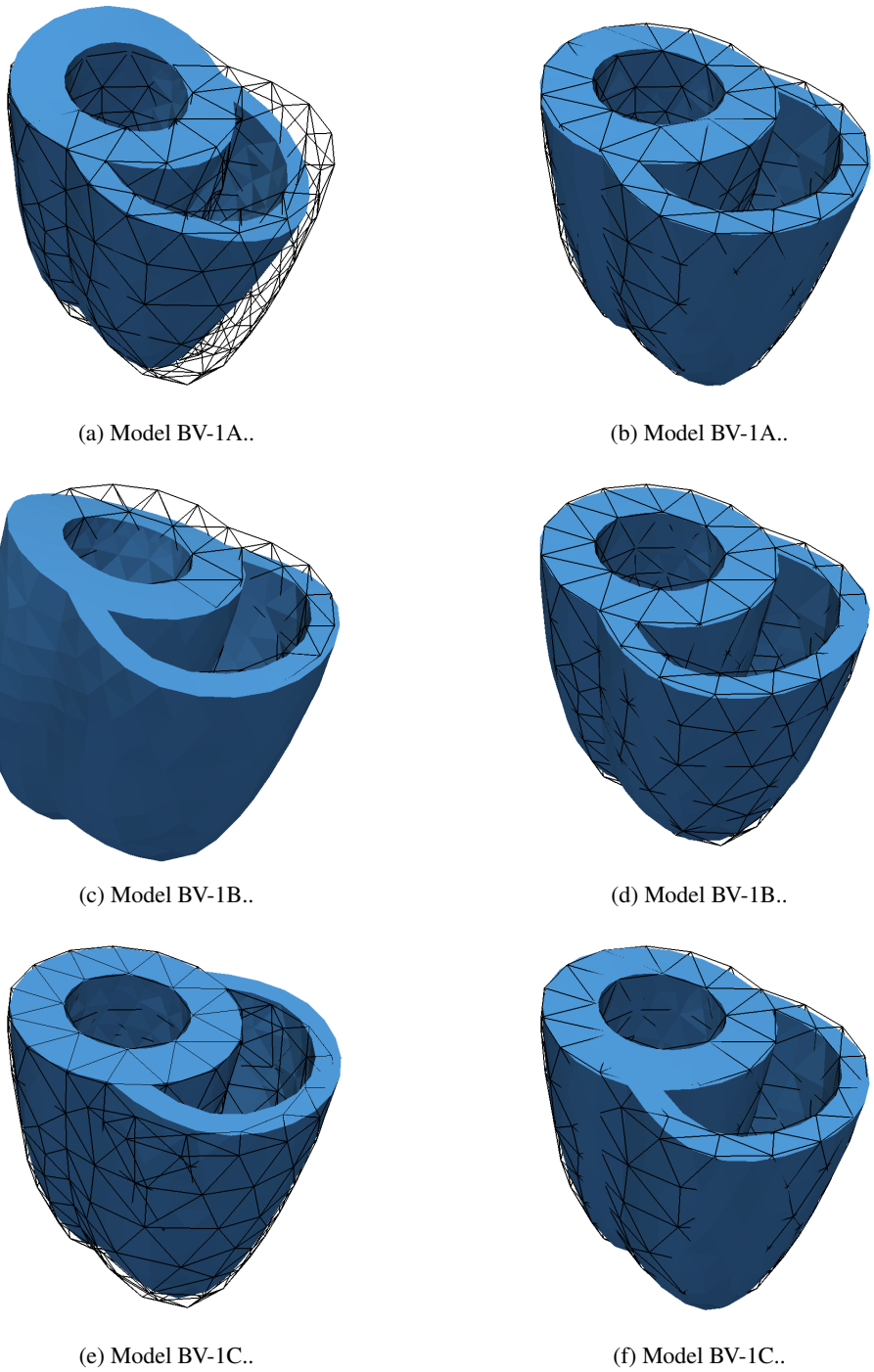
The mesh quality metrics considered were the nodal mean distance (which was computed using a software package called CloudCompare [172], based on the Hausdorff's distance measure), the minimum element angle, and the element shape quality. The latter two were determined using the VTK library [174]. The results obtained were compiled as histograms and are given in Fig. 6.46.



(a) Mean distance. (b) Minimum angle. (c) Mean shape.

Figure 6.46: Mesh quality before and after perturbation, and after registration.

The plot of Fig. 6.46a describes the evolution of the mean distance. A lower value is equivalent to a better registration, as the nodes of the registered mesh are closer to the template mesh. Following the perturbation, the mean distance increased. However, the CPD registration managed to almost restore the mean distance to the GA level, meaning that the meshes recovered their original configurations: BV-1, BV-2 and BV-4. The shape and minimum angle trends were both similar. The perturbation process caused a slight decrease in the mesh quality. Despite the failure of the registration process to restore part of the degradation, the latter could be considered negligible due to its very small effect. Overall, no negative element volumes were recorded. Therefore, the registration could be regarded as successful.



(a) Model BV-1A..

(b) Model BV-1A..

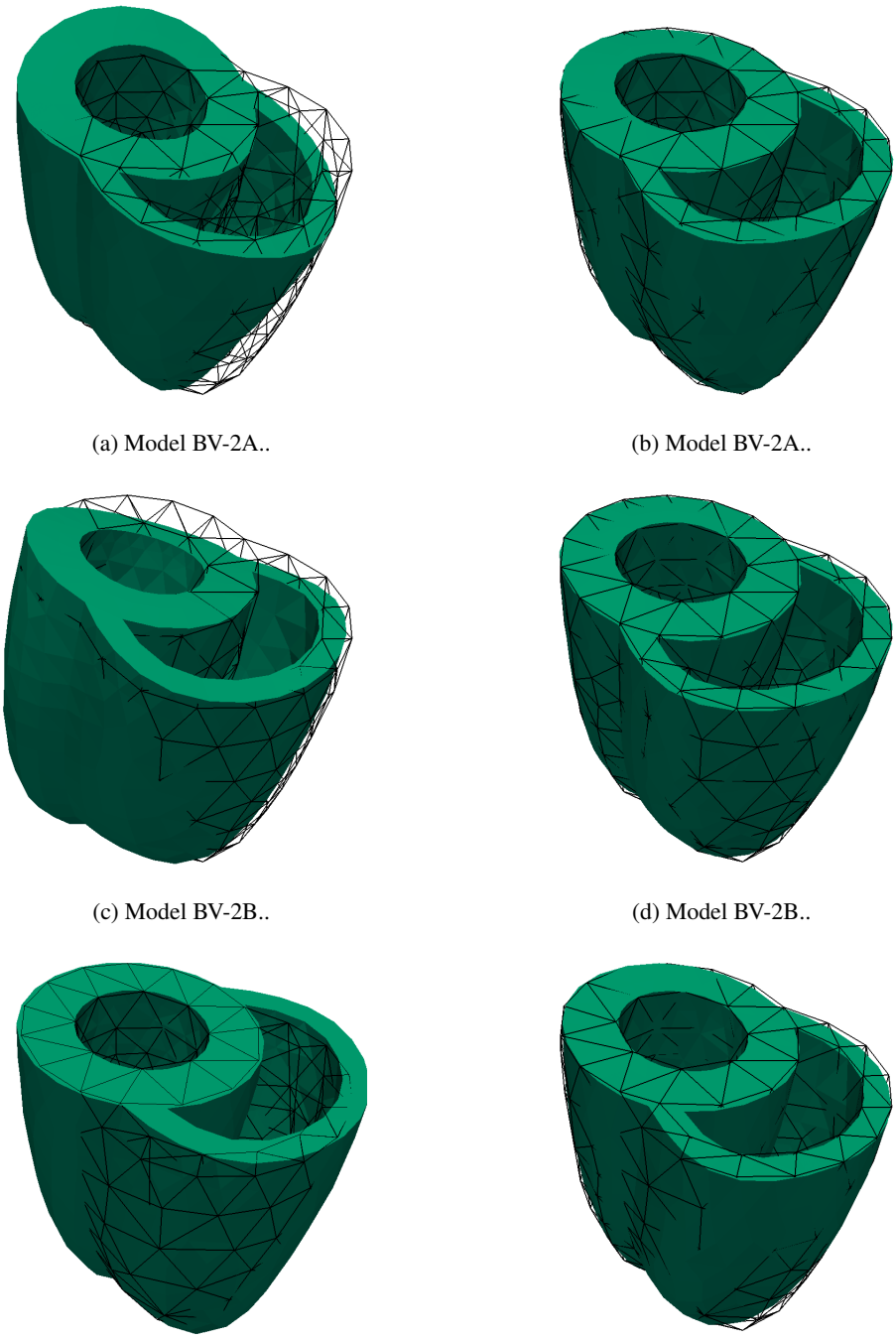
(c) Model BV-1B..

(d) Model BV-1B..

(e) Model BV-1C..

(f) Model BV-1C..

Figure 6.47: Registration of perturbed BV-1 meshes before registration (left side) and after registration (right side).



(a) Model BV-2A..

(b) Model BV-2A..

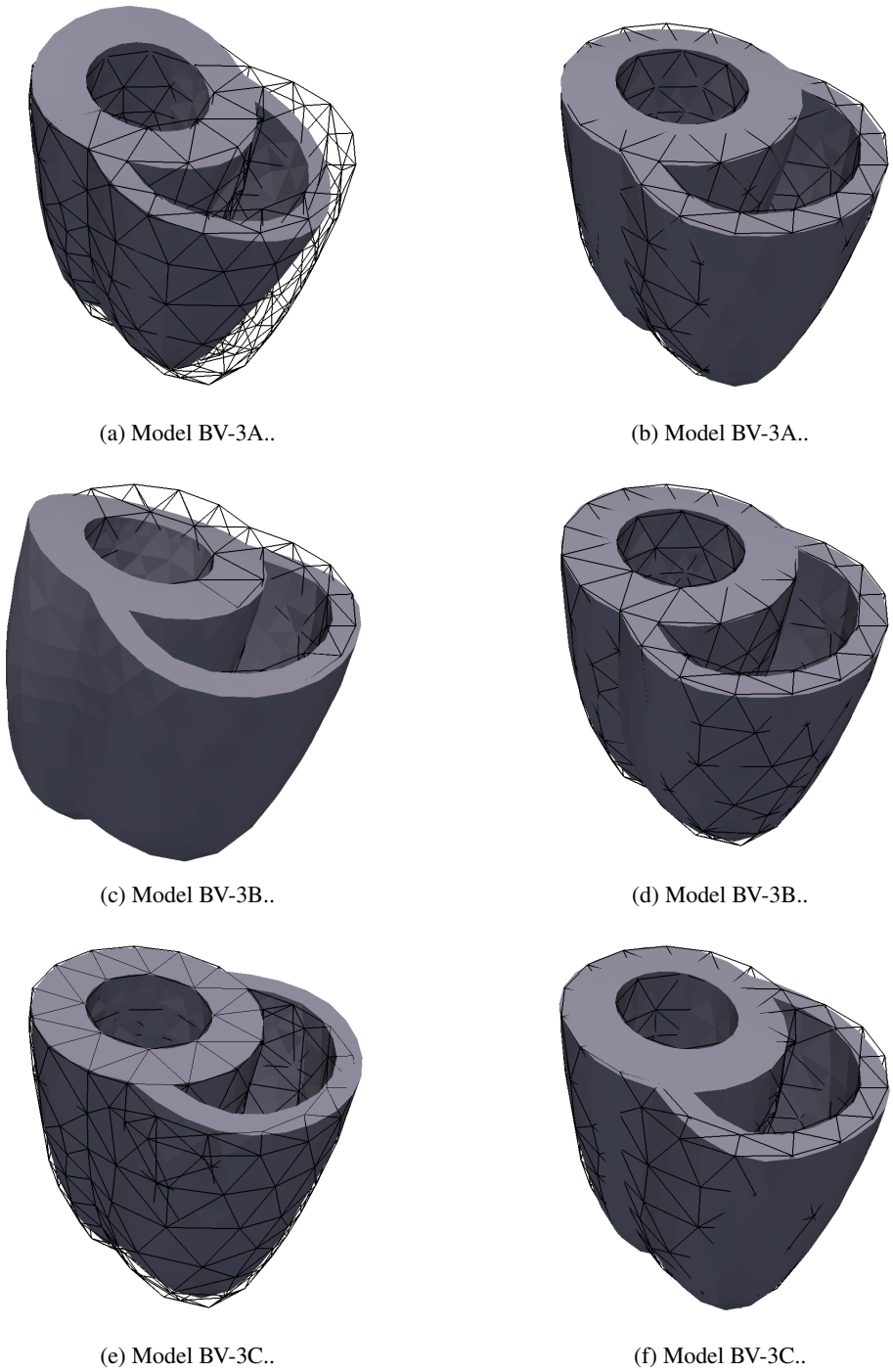
(c) Model BV-2B..

(d) Model BV-2B..

(e) Model BV-2C..

(f) Model BV-2C..

Figure 6.48: Registration of perturbed BV-2 meshes before registration (left side) and after registration (right side).



(a) Model BV-3A..

(b) Model BV-3A..

(c) Model BV-3B..

(d) Model BV-3B..

(e) Model BV-3C..

(f) Model BV-3C..

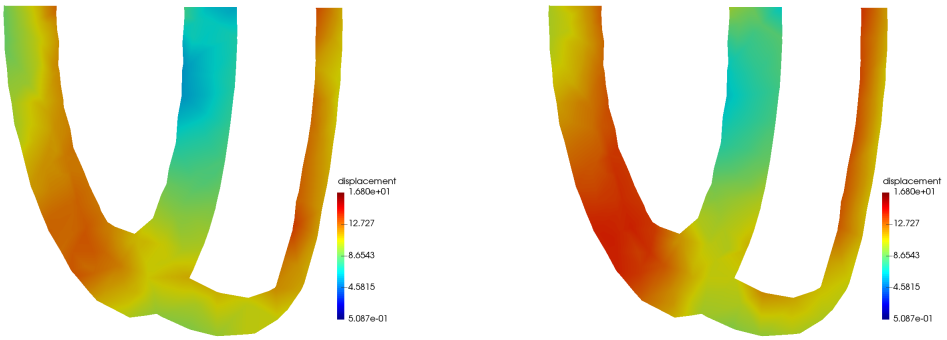
Figure 6.49: Registration of perturbed BV-3 meshes before registration (left side) and after registration (right side).

With the registration completed, the projection and PODI simulation were then carried out for a problem with a stress scaling coefficient set to 0.13 kPa and an energy conservation of 99 %. The calculation took in total 5.8 min, including the registration of the four selected datasets and problem-at-hand geometries on the template heart. In this case too, the projection of the solutions back and forth

from the template was the most expensive operation, as 90 % of the calculation time was spent during that process.

In terms of solution accuracy, the L_2-error norm was better for the heart grid template than the cube template, since it was about 0.22, representing a 39 % drop. This increase in solution quality and smoothness could also be confirmed when a BV slice was visually compared, firstly against the EFG solution, as given in Fig. 6.50, and secondly against the cube template solutions, as shown in Fig. 6.45. Also, these results were found to benefit the pressure-volume relationship curves of the left and right ventricle, as their errors were 3.1×10^{-2} and 4.2×10^{-2}, which was a drop of 81 % and 77 % compared to those of the cube template approach. The strain and stress solutions, on the other hand, had a higher error norm which translated to an increase of 41 % and 31 %, respectively.

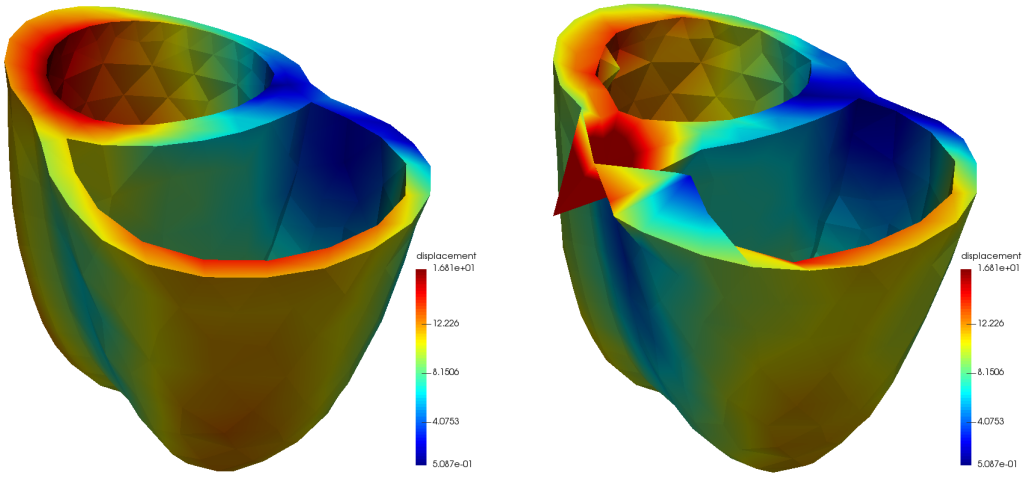
Regarding PODI's calculation characteristics, the three most dominant POMs were conserved in all cases. This was due to the fact that the first POM accounted for only 53.5 %, while the second and third were for 32.7 % and 13.7 % respectively. The fourth and last POM was below 1×10^{-13} %. The energy of the first POM, in this example, is lower than the 62 % energy obtained with the cube grid template. This loss in energy seemed to have been redistributed to the third POM, as the energy of the latter almost doubled. The results presented here, show that the POMs energies were more scattered for the heart grid template method as opposed to the cube grid template, despite accurate solutions being obtained.



(a) PODI solution. (b) EFG solution.

Figure 6.50: Comparison of the PODI solution obtained from a heart template of 347 nodes against the EFG solution.

Even though the displacement field error norm and contour plot slice, of Fig. 6.50a, showed good results, the overall deformation of the bi-ventricle at the end of the diastole still contained a few localised and irregular mesh distortions as shown in Fig. 6.51. These irregularities were individual nodes having non-smooth random movements. This therefore led to a non-smooth displacement field plot which was dominant around the right ventricle and specially at the intersection of the left and right ventricular walls. These localised irregularities were supported by the fact that some registration processes, even though being successful globally over the whole heart, visually showed unsuccessful mapping around the LV-RV wall connection, which can be observed in Figs. 6.47f, 6.48f and 6.49f. One of the main reason for the unsuccessful mapping, was that these locations contained sharp edges, which the CPD algorithm had difficulty in handling properly. Also, this problem can be accentuated, especially when not enough nodes are present around those corners, leading to a lack of information for the registration algorithm to work properly.



(a) EFG solution. (b) PODI solution.

Figure 6.51: Comparing the EFG solution with the PODI solution obtained from a heart grid of 347 nodes.

To solve this problem, the impact of different nodal densities used for the heart grid template on the registration process was investigated. As such, four additional heart templates were considered. The geometry of the templates was kept the same as before. The discretisation was on the other hand refined continuously, leading to more elements and nodes. The discretisation refinement steps were similar to the cube grid discretisations given in Tab. 6.13, which ensured comparable results. The previous heart template was labelled as HG-1, while the refined discretised models were labelled HG-2 to HG-5. Tab. 6.15 is given below to indicate the number of nodes of each heart template.

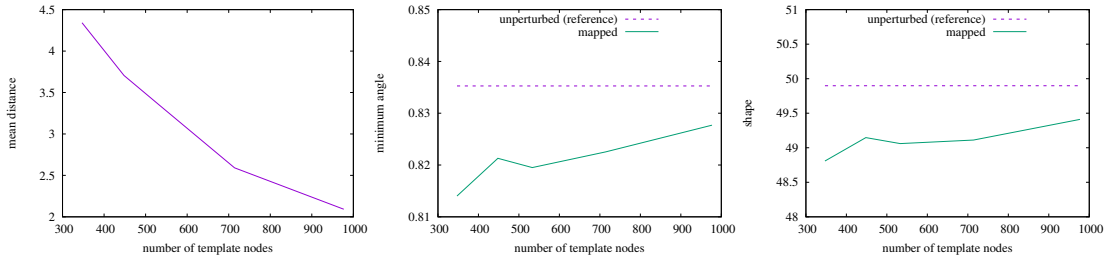
Heart grid	HG-1	HG-2	HG-3	HG-4	HG-5
Nodes	347	448	533	714	977

Table 6.15: Heart grid template with different mesh discretisations.

Using the new heart templates, the performance improvement was separately investigated for the registration of the dataset and problem-at-hand geometries, and also for the projection of the solution fields back and forth from the template geometry. Starting with the CPD registration, the mean distance of the nodes, the average minimum element angle, and the element shape, were the criteria used to assess the quality of the registration. The compiled results are given in Fig. 6.53.

(a) Model BV-1C registered on HG1..
(b) Model BV-1C registered on HG2..
(c) Model BV-1C registered on HG3..
(d) Model BV-1C registered on HG4..
(e) Model BV-1C registered on HG5..
(f) Model BV-2C registered on HG1..
(g) Model BV-2C registered on HG2..
(h) Model BV-2C registered on HG3..
(i) Model BV-2C registered on HG4..
(j) Model BV-2C registered on HG5..
(k) Model BV-3C registered on HG1..
(l) Model BV-3C registered on HG2..
(m) Model BV-3C registered on HG3..
(n) Model BV-3C registered on HG4..
(o) Model BV-3C registered on HG5..

Figure 6.52: Registration of perturbed BV-3 mesh.



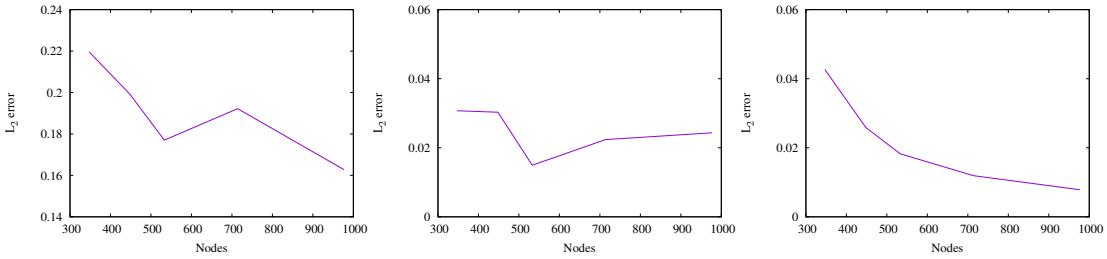
(a) Mean distance. (b) Minimum angle. (c) Shape.

Figure 6.53: Evolution of the mesh quality measures as the number of template nodes increases.

From Fig. 6.53a, it was found that the mean distance decreased continuously with the increase in the number of template nodes. This effect indicated that the registration had been enhanced, leading to the lowering of the average distance between the template nodes and the registered dataset nodes. The average minimum angle and shape of the registered heart elements also showed improvement. As indicated by the curve in Fig. 6.53b and Fig. 6.53c, the minimum angle and shape increased gradually as more grid nodes were provided, moving closer to their reference points, shown as dotted lines on their respective plots, being the unperturbed mesh configuration. Based on these results, it can be deduced that providing more points to the mesh template promotes better registration with less mesh distortion. This point can be supported by a visual inspection of the registration given in Fig. 6.52. In all the three cases, the registration appeared to ameliorate around the LV-RV connection and the RV wall at the basal and apical region.

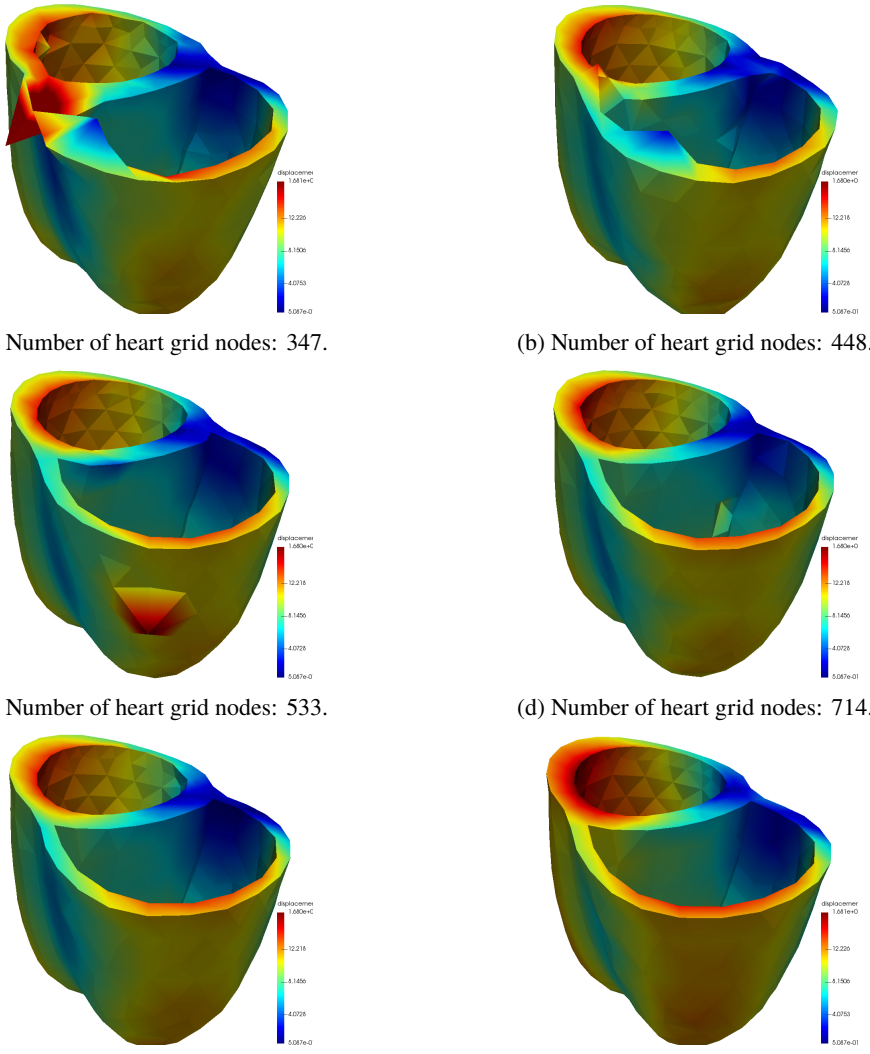
Following the registration phase, the PODI calculations were carried out. The results were extracted and the errors computed. The displacement field error shows, in Fig. 6.54a, an improvement of 25 % as the number of template grid nodes increased. The left and right ventricles volumes behave similarly as their error dropped by 21 % and 81 % respectively, as shown in Fig. 6.54b and Fig. 6.54c. Finally, with respect to the end-diastole deformation, the final mesh configuration was found to be in better condition. From results obtained with the heart template grid HG-1 to HG-4, irregular mesh distortions were still observed, but with decreasing numbers and severity as the grid nodes number increased. Using HG-5, no distortions were visible. The deformed meshes are given in Fig. 6.55.

The energy evolution of the different PODI calculations were the final results analysed. As in the case of the cube grid calculations, the three most dominant POMs were conserved to achieve the minimum energy requirement. This was due to the scattered energies across the four POMs. However, the influence of this effect was limited as the number of grid nodes increased. Based on Fig. 6.56, it was found that the energy of the first POM changed from 53 % to 77 %. This rise correlated to the POD being able to extract the modes in a better way and also isolate the dominant modes more appropriately. For the POD to behave this way, it meant that the projection of the dataset solution fields was carried out consistently on the template heart, indicating that all the details of the solution fields were conserved. Hence, these results confirmed that increasing the number of template nodes led to better PODI solutions, as the registration produced better mappings, which promoted both better interpolation and more accurate POMs.



(a) Displacement. (b) Left ventricle cavity volume. (c) Right ventricle cavity volume.

Figure 6.54: Variation of the L_2-error norm as the number of heart template nodes increases.



(a) Number of heart grid nodes: 347.

(b) Number of heart grid nodes: 448.

(c) Number of heart grid nodes: 533.

(d) Number of heart grid nodes: 714.

(e) Number of heart grid nodes: 977.

(f) EFG.

Figure 6.55: PODI displacement field solution obtained with heart template grid nodes ranging from 347 to 977. The last plot is the exact EFG solution, included for comparison.

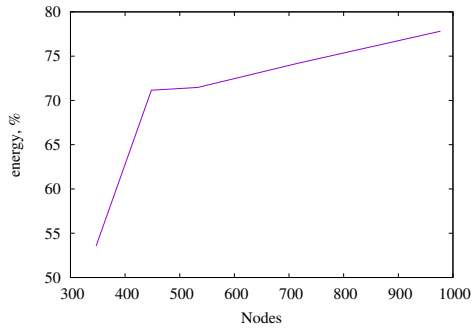


Figure 6.56: Evolution of the energy of the first POMs as the number of heart template grid nodes are increased.

6.5 PODI modelling of the full heart cycle

This section carries out a single full heartbeat reduced order calculation to demonstrate the ability of PODI to be easily extended to the modelling of different heartbeat phases. This consisted of simulating the diastole filling, isovolumetric contraction, ejection and isovolumetric relaxation.

6.5.1 Temporal standardisation approach

As stated in Section 3.4, the diastole filling calculation was the only pressure-driven phase. That is, the pressure was applied incrementally till the end-diastolic pressure was reached during the EFG simulations. These increments were re-adjusted continuously and depended on the numerical stability of the calculation. All EFG simulations carried out in the previous sections were found to have consistent increments. That is, the simulation time steps were the same across all datasets of a database. However, in the case of isovolumetric contraction, ejection and isovolumetric relaxation, the setup was different. During those phases, the calculation was displacement-driven. That is, a volume change was prescribed in the form of a displacement field on the nodes along the cavity wall, and the ventricular pressure was solved for. This method led to robust calculations under multiple loading conditions, as demonstrated in Skatulla and Sansour [2]. For a displacement-driven calculation, especially during the active tension phase, the simulation time is a function of the sarcomere length, which in turn is a function of the strains, as given in Eq. (3.49) and Eq. (3.51). Since for different stress scaling coefficients and fibre directions the strain state changed at the end of the diastole filling, each simulation started the isovolumetric contraction phase with a different sarcomere length and consequently, different relaxation times. Hence, the simulation timeline of every heart problem evolved as a function of the stress scaling coefficients and fibre directions, even though the active stress and Windkessel parameters remained constant. This effect led to calculation-step sizes and the number of steps being inconsistent across datasets. As such, at any point along the simulation timeline, two heart states of two datasets, could be out of phase. For example, one heart could still be at the end of isovolumetric contraction while another was already in the ejection phase. The evolution of the simulation timeline could not be dealt with using the current implementation of the PODI method. To solve this problem, a *time standardisation scheme* was proposed.

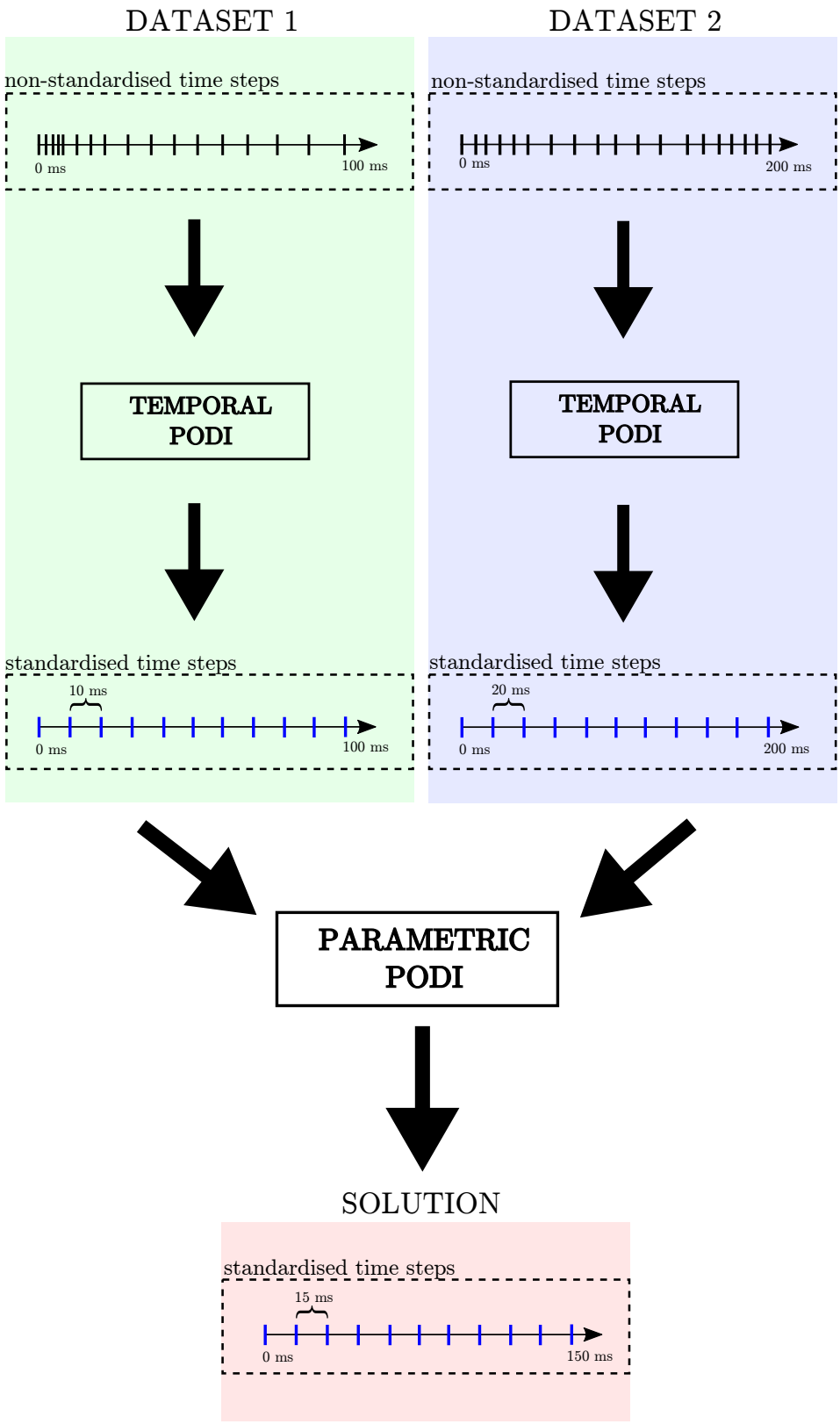


Figure 6.57: Visual representation of a PODI calculation based on the time standardisation process.

The time standardisation scheme consisted of defining a standardised timeline to which the dataset simulation timeline and its corresponding solutions were converted. This standardised timeline is a series of equally spaced time points that span over a whole heartbeat cycle, and are referred as *template simulation time steps* or *standardised time steps*. The conversion process consisted of interpolating the dataset solution fields to the standardised timeline. Once done, the PODI calculation could proceed as previously described in Section 5.3.3. The PODI results obtained belonged to the reference timeline as shown in Fig. 6.57. One major problem encountered during the early implementation stage of this approach was that the last simulation time step of each phase of a heartbeat was not properly captured. This was because as the timeline was discretised by a uniform time interval, it was very unlikely that a standardised time step would fall exactly at the change of a heartbeat phase. Very small time intervals could be considered to minimise this problem, but the PODI calculation time would then be negatively impacted. To circumvent this problem, a modified approach was undertaken where the time discretisation process was not utilised over a full heartbeat, but instead, across each phase of a heart beat. To do so, the simulation time step at the start and end of each cardiac phase were identified. The template simulation time steps were then determined by discretising the cardiac phase timeline between the start and end point. For each dataset, their cardiac phases were split into equal numbers of time points, to form a vector of template simulation time steps, $\mathcal{T}_i^p$. Here, i corresponds to the dataset and p represents the phase of the heartbeat, $p \in \{\text{DF, IC, EJ, IR}\}$ with DF being the diastole filling, IC the isovolumetric contraction, EJ the ejection and IR the isovolumetric relaxation. At that moment, the template simulation time steps did not have any associated solution fields. To obtain these solution fields, a new set of PODI calculations was carried out. The goal of this new set of PODI calculations was to interpolate, on a low-dimensional space, the non-standardised timeline solution fields to the standardised fields. In this case, an interpolation scheme was built up based on the actual time steps of the non-standardised timeline, referred to as a *temporal PODI calculation*. The Temporal PODI calculation approach has been used in the literature [46] to obtain smooth transition of results across a specific time-frame. However, it should be noted that this type of PODI calculation was different to the one presented in Section 5.3.1, where the interpolation scheme was set up around material parameters, and is alternatively known as a *parameteric PODI calculation*.

In the temporal PODI calculations, the supporting time steps are found in the non-standardised timeline vector, $[\mathbf{T}]_i^p$ and are used to interpolate for $\left[\mathcal{T}_j\right]_i^p$, where $\mathcal{T}_j$ is one of the template time steps. Hence, the interpolants are defined through:

$$\left[\mathcal{T}_j\right]_i^p = \mathbf{N}_{\text{time}}^{\text{MLS}} \cdot [\mathbf{T}]_i^p \,. \tag{6.4}$$

Regarding, $\mathbf{U}_i$, the ensemble data matrix of the temporal PODI calculation, it is built differently to that proposed in Section 5.3.1. In this case, the columns no longer represented the solutions of different parametric datasets at a specific time step, as shown in Fig. 6.58a. Instead, the solution for a particular parametric dataset at different non-standardised time steps were stacked column-wise, as given in Fig. 6.58b. Thus, the POMs extracted represented the behaviour of the ensembled solution field, across the selected time steps. The number of rows of the two different ensemble matrices were the same, as the degrees of freedom of the solution fields did not change. With the data matrix and interpolation scheme set up, the temporal PODI calculation could be carried out for all reference time steps of every selected dataset.

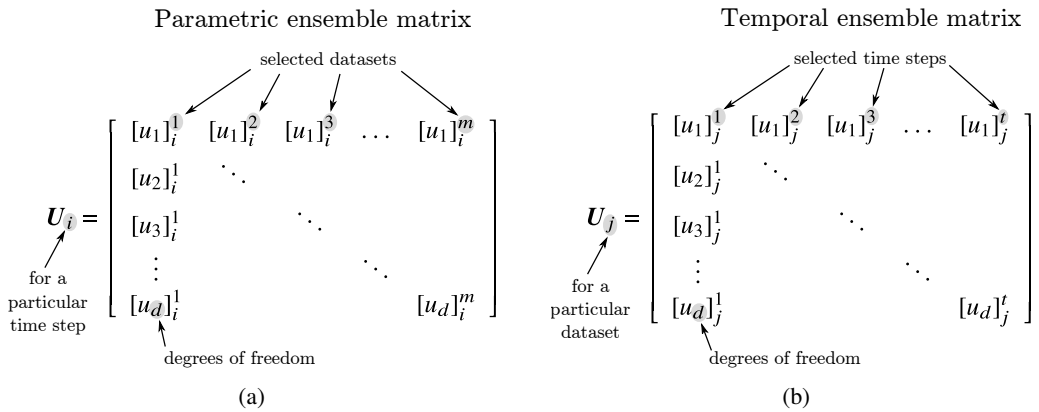


Figure 6.58: Comparison of (a) non-temporal against (b) temporal PODI ensemble matrix layout.

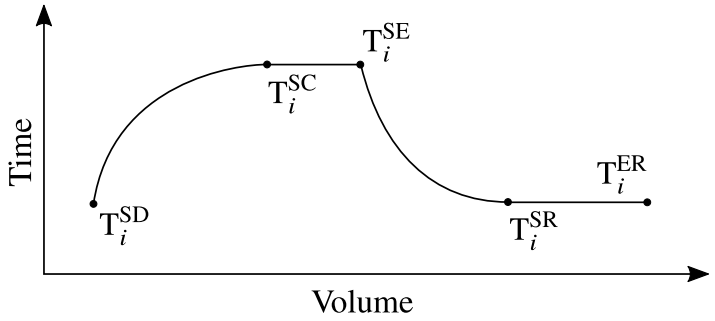


Figure 6.59: Identification of the end-time steps of each phase across a volume-time graph of a dataset, i. Note that the graph is not drawn to scale.

After the standardisation of the datasets timeline and solution fields, the next step was to create the standardised timeline for the PODI problem-at-hand. The current procedure employed is the interpolation of the starting and ending time steps of each phase from the selected datasets. To do so, those end time steps were first compiled in vectors defined by: the start of diastole filling, $\mathbf{T}^{\mathrm{SD}}$, start of isovolumetric contraction, $\mathbf{T}^{\mathrm{SC}}$, start of ejection, $\mathbf{T}^{\mathrm{SE}}$, start of isovolumetric relaxation, $\mathbf{T}^{\mathrm{SR}}$, and finally, the end of isovolumetric relaxation, $\mathbf{T}^{\mathrm{ER}}$. The end of diastole filling, end of isovolumetric contraction and end of ejection were not considered since they were technically the same as the start of isovolumetric contraction, $\mathbf{T}^{\mathrm{SC}}$, start of ejection, $\mathbf{T}^{\mathrm{SE}}$ and start of isovolumetric relaxation, $\mathbf{T}^{\mathrm{SR}}$, respectively. An example of the end time steps identification for a dataset, i, is given in Fig. 6.59 and the compilation of time step vectors was carried out as follows:

$$\left[\mathbf{T}^{\mathrm{SD}}\right]^{\mathrm{T}} = \left[T_1^{\mathrm{SD}},\ \ldots,\ T_i^{\mathrm{SD}},\ \ldots,\ T_m^{\mathrm{SD}}\right] \tag{6.5}$$

$$\left[\mathbf{T}^{\mathrm{SC}}\right]^{\mathrm{T}} = \left[T_1^{\mathrm{SC}},\ \ldots,\ T_i^{\mathrm{SC}},\ \ldots,\ T_m^{\mathrm{SC}}\right] \tag{6.6}$$

$$\left[\mathbf{T}^{\mathrm{SE}}\right]^{\mathrm{T}} = \left[T_1^{\mathrm{SE}},\ \ldots,\ T_i^{\mathrm{SE}},\ \ldots,\ T_m^{\mathrm{SE}}\right] \tag{6.7}$$

$$\left[\mathbf{T}^{\mathrm{SR}}\right]^{\mathrm{T}} = \left[T_1^{\mathrm{SR}},\ \ldots,\ T_i^{\mathrm{SR}},\ \ldots,\ T_m^{\mathrm{SR}}\right] \tag{6.8}$$

$$\left[\mathbf{T}^{\mathrm{ER}}\right]^{\mathrm{T}} = \left[T_1^{\mathrm{ER}},\ \ldots,\ T_i^{\mathrm{ER}},\ \ldots,\ T_m^{\mathrm{ER}}\right], \tag{6.9}$$

where m, is the number of selected datasets for the PODI calculation. Once these end time step vectors were compiled, an interpolation scheme was carried out along each end time steps' vector of the PODI problem-at-hand. The MLS interpolation scheme was again employed here and is built up from the selected-datasets parameters, since, as indicated earlier, the latter was responsible for the evolution of the simulation time steps. The interpolation process was carried out for each end-point vector as follows:

$$\hat{T}^e = \mathbf{N} \cdot \mathbf{T}^e, \tag{6.10}$$

where $e \in \{\text{SD, SC, SE, SR, ER}\}$. Once the end time steps of the problem-at-hand were obtained, the standardised timeline, $\hat{\mathbf{T}}^e$, were determined. The parametric PODI calculation, which was the same as the PODI calculation given in Section 5.3, could take place afterwards to obtain the solution fields of the time steps, $\hat{\mathbf{T}}^e$, of the problem-at-hand.

6.5.2 Numerical example

With the time standardisation process introduced, an example was considered. For the full heart cycle PODI problem, a bi-ventricle model, different to the one given in Section 6.2.1, was used. The new geometry was a replicate of the idealised BV given in [97]. The idealised BV is shown in Fig. 6.60.

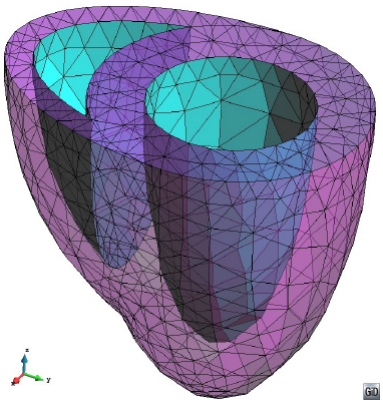


Figure 6.60: The idealised bi-ventricle model.

The geometry was discretised using linear tetrahedral elements resulting in 193 nodes and 550 elements. As Dirichlet boundary conditions, the border lines, lying on the surface of the base were treated as elastic springs, set to a constant value of $1 \times 10^{-2}\,\text{kN mm}^{-1}$, while the base surface's movement along the z-axis was prevented. As Neumann boundary condition, surface pressures were applied to the left and right ventricle. However, in this example, the left ventricle end-diastole surface pressure, $P_{\text{ED}}^{\text{LV}}$, varied between 1.0 kPa and 2.0 kPa, as it was used to create a database. Using an increment of 0.25 kPa, a one-dimensional database was constructed with $P_{\text{ED}}^{\text{LV}} \in \{1.00, 1.25, 1.75, 2.00\}$. The pressure of $P_{\text{ED}}^{\text{LV}} = 1.50\,\text{kPa}$ was not included as it represented the problem which PODI was run for. It should be noted that the dimensional space of the database was different to previous PODI examples that were considered, as they were based on material parametric dimensions. To prevent numerical instabilities during the full-scale simulation of the datasets, the left ventricular to right ventricular pressure ratio was set to 1.1:0.95, as given in [2]. Using the specified left ventricular pressure, the right ventricular

pressures were consequently determined, as given in Tab. 6.16. Finally, the values of the cardiac material parameters for the active stress and WindKessel model were as given in Tab. 6.17 and Tab. 6.18 respectively. Regarding the time standardisation calculation, each phase of a heartbeat was discretised into a total of 100 time steps. Hence, the total number of time steps for a full heart beat cycle was 400. The temporal PODI computation was then carried out using an energy conservation level of 99.9 %.

	Ventricle	Dataset			
		1	2	3	4
Pressure (kPa)	LV	1.00	1.25	1.75	2.00
	RV	0.86	1.08	1.51	1.73

Table 6.16: PODI database constructed from different ventricular pressures.

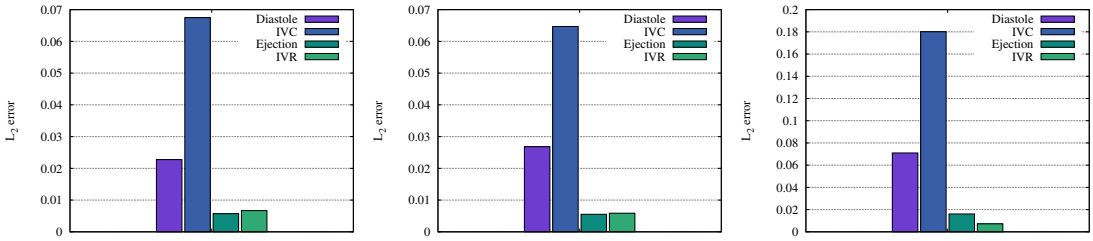
T_{max}	Ca_0	Ca_0^{max}	B	l_0	t_0	m	b
135.7 kPa	4.35 µmol	4.35 µmol	4.75 µm	1.58 µm	80 ms	1.0489 s µm^{-1}	−1.429 ms

Table 6.17: Active stress parameters.

	$C\left(\text{mm kPa}^{-1}\right)$	$R\left(\text{kPa s mm}^{-3}\right)$	$R_0\left(\text{kPa s mm}^{-3}\right)$
LV	4000	1×10^{-4}	1×10^{-5}
RV	1000	4×10^{-4}	4×10^{-5}

Table 6.18: Three-element Windkessel parameters.

Following the PODI calculation, the results were analysed. The first set of results looked at was the performance of the ROM calculation. Using only one processor, the full EFG simulation took about 6.5 h, while the PODI computation lasted, on average, for 10.5 s. This showed that PODI is about 2200 times faster. This PODI computation time was slightly lower than those recorded in Section 6.3.1 due to the fact that only 5 s was spent in reading the off-line dataset results from the hard-disk medium, as opposed to 10.5 s in the previously referred to section. If the reading time had not counted, then the current PODI calculation would have been about 40 % higher. This increase in calculation time was due to a higher number of calculation steps being processed; 401 in this example as opposed to only 58 in Section 6.3.1. The time standardisation procedure itself took about 2.8 s, which was only about 26 % of the total calculation time. This standardisation time could have been reduced if fewer standardised time steps were used. The displacement field calculations took around 0.084 s, which represented an increase of 55 % over the same calculation time recorded in Section 6.3.1, and can be again attributed to the higher number of steps being processed.



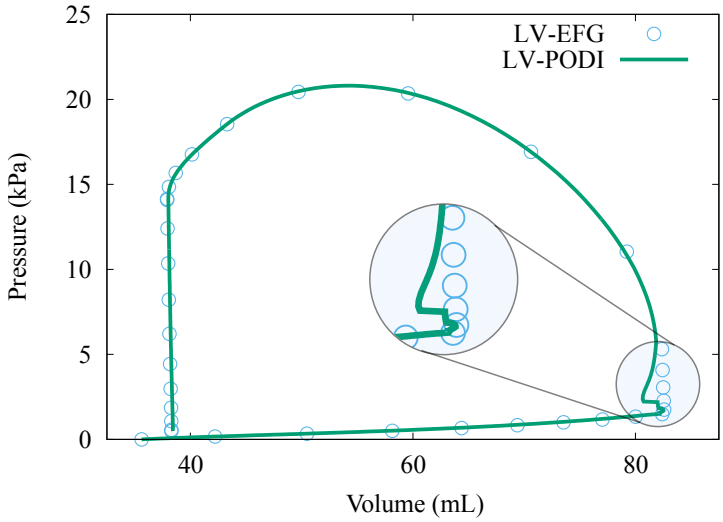
(a) Displacement field. (b) Strain field. (c) Stress field.

Figure 6.61: Solution field error across the diastole, isovolumetric contraction (ICV), ejection and iso-volumetric relaxation (IVR).

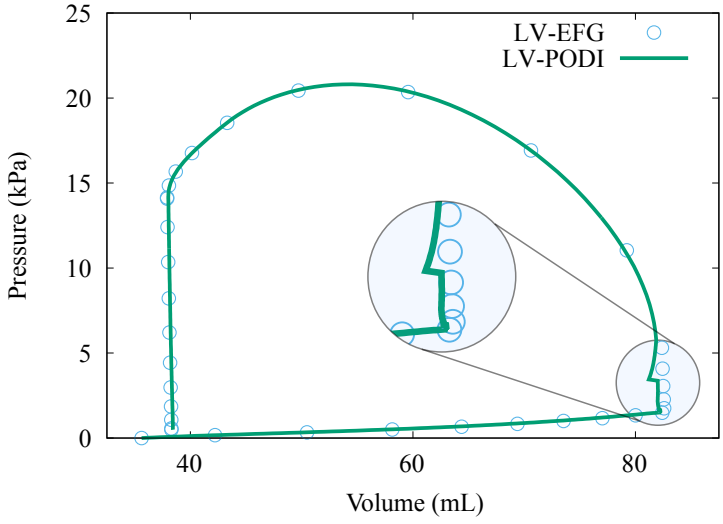
With regard to the accuracy of the solution, a different approach was undertaken to define the exact result against which the PODI results could be compared. Since the full-scale simulation results were based on a different series of time steps, they also had to be standardised. For this stardardisation procedure, a temporal PODI calculation was again carried out. However, a 100 % energy conservation was specified in order to conserve most details of the exact solution during the reduced order interpolation. This process is acknowledged to introduce errors which, however, can be considered minimal and negligible. After the exact results had been determined using EFG, the PODI calculations accuracy was then investigated. On average, the error norm of the displacement, stress and strain fields was 0.044, 0.06 and 0.038, respectively. However, it was observed that those errors were not equally distributed across the different phases of a heartbeat, as shown in Fig. 6.61. The highest error was recorded during the isovolumetric contraction phase. It was about 1.5 to 2 times higher than the diastole filling phase. Those of the ejection and isovolumetric relaxation phases were considerably lower, as they were about 78 % less than that of the diastole filling phase. These error values showed very little variation, as the energy conserved varied from 70 % to 100 %. But for the pressure-volume relationship of the left ventricle, some benefits were observed. At 70 % of the energy conserved, the distribution of the left ventricle PV error across the phases was similar to the displacement, stress and strain field. The pressure volume curve during the IVC showed non-physical behaviour where an expected straight line was found to be broken and skewed in different directions. Yet, as the conserved energy was increased to 100 %, the accuracy of the calculation increased, showing that the pressure-volume error norm during IVC decreased from 0.011 to 0.05. This decrease in error is reflected on the PV plot in Figs. 6.62a, 6.62b and 6.62c for the energy conservation of 70 %, 90 % and 99.99 % respectively. Additionally, it was observed that the PODI PV loop of the left and right ventricles seemed to fit as expectedly between the PV loops of the PODI calculation selected datasets, as shown in Fig. 6.63 and Fig. 6.64. The end-diastole, end-IVC, end-ejection and end-IVR volume and pressure recorded were in accordance with the trend-line established from the database. Interestingly, it was noted that the PODI calculation managed to capture the convergence point of the dataset PV curves for both the left and right ventricle, that occurred during the ejection phase.

The results provided above show that the most unstable region of the PODI calculation during a full heartbeat occurred across the isovolumetric contraction phase. On the left and right ventricular PV loops, given in Fig. 6.63 and Fig. 6.64, it can be observed that the IVC phase was the only region where the difference in volume between the closest selected datasets was larger. This difference was not visible during the diastole filling, but was partly present at the start of ejection, and minimal during the

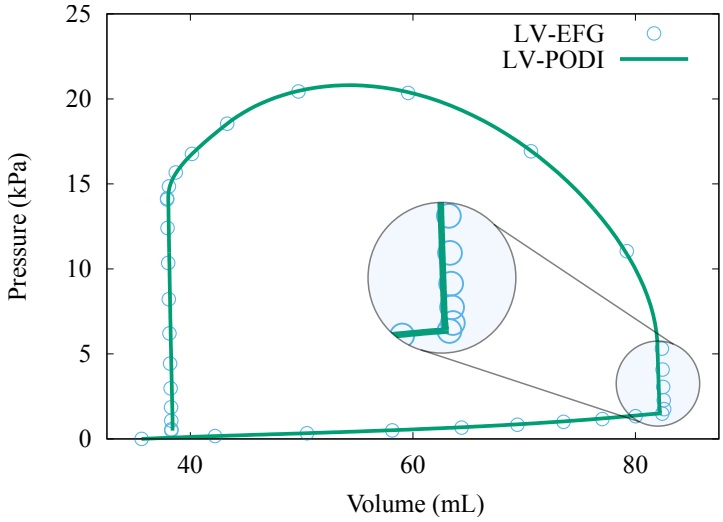
isovolumetric relaxation phase. Due to the volume difference, the most dominant proper orthogonal modes during IVC were less energetic, as shown in Fig. 6.65, where the energy of the first POM dropped from 99 % to around 60 % during the IVC time steps. Another unstable region was observed across the heartbeat timeline when looking at the number of POMs conserved, as shown in Fig. 6.66. As expected, an increase in POMs conserved took place during the IVC. But that increase was also maintained during the start of ejection. However, across the region where the curves of both the neighbouring datasets met, the number of POMs conserved decreased. This decrease was due the POMs extracted by the POD being more energetic, as the dataset solution fields were almost the same. Still, at the end of ejection, the difference in the dataset volumes was larger, again causing an increase in the number of POMs conserved. During the isovolumetric relaxation phase, the dataset volumes were closer to each other, leading to only a few POMs being conserved. However, at the end of the IVR, the number of POMs conserved increased as the diastole phase started again, with a larger difference in dataset volumes. Based on these results, it could be concluded that the instabilities in the PODI calculation of a heartbeat arise principally during the IVC and also at the phase transition of IVC to ejection, ejection to IVR and IVR to diastole. As such, at those regions it is important to provide more solution fields to the PODI calculation. Hence, an approach where the standardised steps are more concentrated in those regions and the PODI calculation have a larger number of selected datasets, can lead to smaller errors.



(a) Energy conserved: 70 %.



(b) Energy conserved: 90 %.



(c) Energy conserved: 99.99 %.

Figure 6.62: Comparing the left ventricular pressure-volume curve generated by EFG against PODI at increasing energy conservation levels.

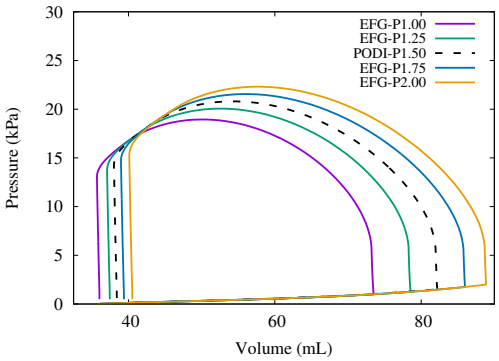


Figure 6.63: Left ventricular pressure-volume curve of the PODI selected dataset and the PODI solution.

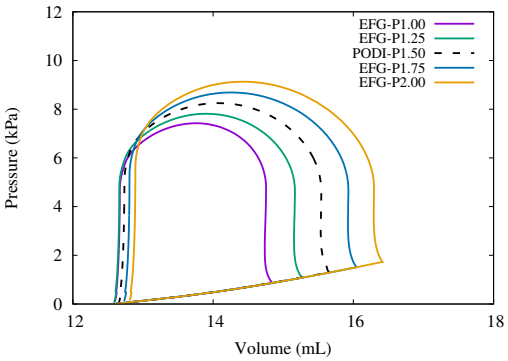


Figure 6.64: Right ventricular pressure-volume curve of the PODI selected dataset and the PODI solution.

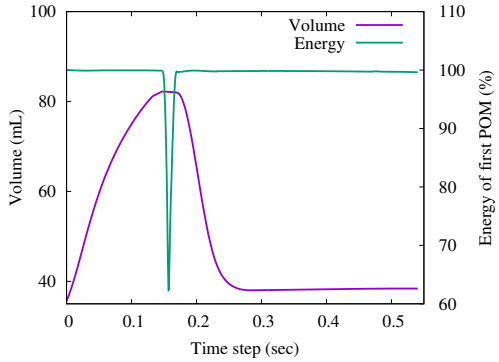


Figure 6.65: Change of the left ventricular volume and energy of the most dominant mode.

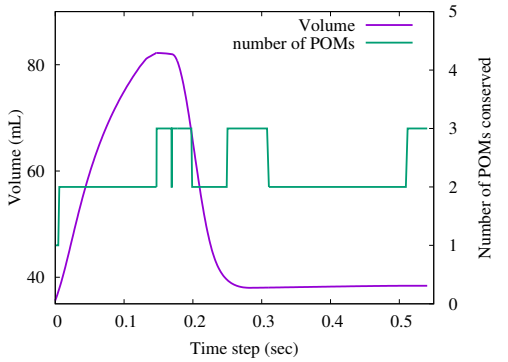


Figure 6.66: Change of the left ventricular volume and number of POMs conserved.

Part IV

Proper Orthogonal Decomposition with Interpolation-based Material Parameter Optimisation

Chapter 7

Levenberg-Marquardt Method

7.1 Introduction

ROM calculations are often used to speed up computationally expensive methods that require simulation results as input. One of these computationally expensive methods is parameter optimisation. Several studies have been conducted to investigate the coupling of ROM and optimisation techniques. For example, Rozza et al. [181] applied the Proper Orthogonal Decomposition to transport mechanics, Choi et al. [179] used a gradient-based optimisation scheme along with interpolated POMs to achieve realistic wing design of an aircraft, and van Doren et al. [182] used a reduced optimisation in the context of water-flooding a heterogeneous reservoir. In the end, all of them managed to accelerate their computations while still maintaining a reasonable level of accuracy. The PODI method provides a very fast means of obtaining full solution fields of unknown problems. Consequently, using this advantageous characteristic can be helpful in accelerating other highly computationally demanding calculations or methods where those solution fields are used as input. One such method that was looked at in this research is the Levenberg-Marquardt Algorithm (LVM), which is a parameter identification or curve-fitting tool that helps in fitting functions to data points In cardiac modelling, such tools are often required to find material constants from experimental data or physiological behaviour of the heart, such as end-diastolic volumes. Given that cardiac modelling is computationally expensive, coupling the PODI method with the LVM algorithm helps in finding those parameters faster and more efficiently.

The application of PODI along with optimisation methods has been observed in the literature, for example, in My-Ha et al. [183]. My-Ha et al. employed a form of PODI with a least square optimisation method within the framework of free surface shape response as a result of bubbles. To that extent, they achieved close to real-time optimisation. Iuliano and Quagliarella [67] also employed the PODI method with Radial basis function along with an evolution optimisation algorithm for the aerobic shape design of an airfoil. The difference between their work and this study, is the usage of PODI with a different interpolation method along with the application of LVM for parameter identification of the heart material properties.

7.2 Theory

7.2.1 Cost function

The Levenberg-Marquardt method consists of minimising the sum of square distance errors between data points and a function. To do so, the method entails the following steps: (i) An initial error is computed between the computed data points or the entire fitted function and a corresponding reference; (ii) the parameters which need to be optimised are perturbed; (iii) a new error is computed; and (iv) the new error and the old one are compared against each other.

The definition of LVM is constructed from a nonlinear least-square optimisation ansatz. Here, the theory by Guyon and Riche [78] is adopted. Consider a vector of experimental data, $\mathbf{z}$, to which a function, $\boldsymbol{f}(\mathbf{x})$ is to be fitted. The function is defined by a set of unknowns, $\mathbf{x}$, which belongs to a n-dimensional hypercube parametric domain, $S \subset \mathbb{R}^n$. As such, the so-called *cost function* of the LVM is posed at any iteration k in terms of the first order linearisation of $\boldsymbol{f}(\mathbf{x}_k)$ as:

$$\begin{aligned} \underset{x \in S}{\text{minimise}} \quad & J_{\text{LVM}} = \frac{1}{2} \left\| \boldsymbol{f}(\mathbf{x}_k) + \boldsymbol{\nabla f}(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k) - \mathbf{z} \right\|^2 \\ \text{subject to} \quad & g(x) = \frac{1}{2} \left\| \mathbf{x}_{k+1} - \mathbf{x}_k \right\|^2 - \delta_k < 0, \end{aligned} \tag{7.1}$$

where $\|\cdot\|$ is known as the L_2-norm. The constraint introduced in Eq. (7.1) is used to enforce a stopping criterion with a specified threshold, δ_k. Therefore, the change in the parameter values is not allowed to move beyond a trusted region, which in this case is the boundary of the hypercube, S. $\boldsymbol{\nabla f}(\mathbf{x}_k)$ is the gradient of the function, $\boldsymbol{f}$, with respect to $\mathbf{x}$ and is called the Jacobian, $\mathbf{J} = \boldsymbol{\nabla f}(\mathbf{x}_k)$. Following the solving of the LVM problem, the corresponding equation is obtained:

$$\left(\boldsymbol{\nabla f}(\mathbf{x}_k)^T \boldsymbol{\nabla f}(\mathbf{x}_k) + \lambda_k^{\text{LVM}} \mathbf{I} \right) (\mathbf{x}_{k+1} - \mathbf{x}_k) = -\boldsymbol{\nabla f}(\mathbf{x}_k)^T \left(\boldsymbol{f}(\mathbf{x}_k) - \mathbf{z} \right). \tag{7.2}$$

λ^{LVM} is the Lagrange multiplier constant and $\mathbf{I}$, the identity matrix. $\boldsymbol{\nabla f}(\mathbf{x}_k)^T \boldsymbol{\nabla f}(\mathbf{x}_k) = \mathbf{J}^T \mathbf{J} = \mathbf{H}$ is referred as the *Hessian matrix*. It is a square matrix whose size is dictated by the number of parameters in $\mathbf{x}$, i.e. n. In order to find the update of the parameter vector, $\mathbf{x}_{k+1}$, in the LVM algorithm, Eq. (7.2) is restated such that:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left(\boldsymbol{\nabla f}(\mathbf{x}_k)^T \boldsymbol{\nabla f}(\mathbf{x}_k) + \lambda_k^{\text{LVM}} \mathbf{I} \right)^{-1} \boldsymbol{\nabla f}(\mathbf{x}_k)^T \left(\boldsymbol{f}(\mathbf{x}_k) - \mathbf{z} \right). \tag{7.3}$$

Although the presented equation behaves very well in some cases, it has a critical flaw. As discussed by Transtrum and Sethna [184], LVM, as it is, can become insensitive if the magnitude of the parameters in $\mathbf{x}$ differs dramatically when moving along some parameter space where the cost function forms a plateau. In those regions, the search algorithm tries to find a minima by significantly increasing the parameter values, which causes so-called *parameter evaporation*. In order to remedy this problem, the solution of replacing the identity matrix from Eq. (7.3) with another term to allow for automatic re-scaling of the parameters has been established [185].

In order to find an automatic way of rescaling the parameters after every step, an approach based on replacing the identity matrix by a damping term, which is a function of the Jacobian, is considered. To do

so, the LVM equations are first reformulated by a scaled variable for each iteration k [78]:

$$[\mathbf{D}]_k \, \bar{\mathbf{x}} = \mathbf{x}. \tag{7.4}$$

$\bar{\mathbf{x}}$ is the vector of scaled parameters and $[\mathbf{D}]_k$ is a diagonal matrix which is positive definite. The choice of a representative term for the diagonal matrix is based on two conditions. The first condition is when some prior knowledge of the parametric space surrounding the solution is known. In such cases, the starting set of parameters is placed along the diagonal of $[\mathbf{D}]_k$. For the second condition, where there is a lack of information, the Jacobian is utilised in the form of:

$$[\mathbf{D}_i]_k = \left\| \frac{\partial}{\partial \mathbf{x}_i} \boldsymbol{f} \left(\mathbf{x}_k \right) \right\|^{-1}, \tag{7.5}$$

where $i \in \{1, \ldots, n\}$. A special scenario that can occur is when one of the entries of the Jacobian is 0. This consequently results in unreliable iteration steps which can cause instability issues [78]. Hence, a practical solution to this problem is to replace 0 with the value from the previous iteration, $k-1$, of the same entry. With the scaling variable defined, Eq. (7.1) is reformulated by substituting Eq. (7.4) into its minimisation constraints:

$$\begin{aligned} \underset{\mathrm{x} \in S}{\text{minimise}} \quad & J_{\mathrm{LVM}} = \frac{1}{2} \left\| \boldsymbol{f} \left(\mathbf{x}_k \right) + \boldsymbol{\nabla f} \left(\mathbf{x}_k \right) \left(\mathbf{x}_{k+1} - \mathbf{x}_k \right) - \mathbf{z} \right\|^2 \\ \text{subject to} \quad & \bar{g} \left(\bar{\mathbf{x}} \right) = \frac{1}{2} \left\| \bar{\mathbf{x}}_{k+1} - \bar{\mathbf{x}}_k \right\|^2 - \delta_k < 0. \end{aligned} \tag{7.6}$$

Solving Eq. (7.6) leads to:

$$\left([\mathbf{D}]_k^T \, \boldsymbol{\nabla f}^T \boldsymbol{\nabla f} \; [\mathbf{D}]_k + \lambda_k^{\mathrm{LVM}} \mathbf{I} \right) \left(\bar{\mathbf{x}}_{k+1} - \bar{\mathbf{x}}_k \right) = - \, [\mathbf{D}]_k^T \, \boldsymbol{\nabla f}^T \left(\boldsymbol{f} \left([\mathbf{D}]_k \, \bar{\mathbf{x}}_k \right) - \mathbf{z} \right). \tag{7.7}$$

Multiplying the above equation by $[\mathbf{D}]_k^{-T}$ and substituting in Eq. (7.4), the formulation, proposed by Marquardt, is recovered:

$$\left(\boldsymbol{\nabla f}^T \boldsymbol{\nabla f} + \lambda_k^{\mathrm{LVM}} \, [\mathbf{D}]_k^{-T} \, [\mathbf{D}]_k^{-1} \right) \left(\mathbf{x}_{k+1} - \mathbf{x}_k \right) = -\boldsymbol{\nabla f}^T \left(\boldsymbol{f} \left(\mathbf{x}_k \right) - \mathbf{z} \right). \tag{7.8}$$

$[\mathbf{D}]_k^{-T} \, [\mathbf{D}]_k^{-1}$ is referred as the *damping matrix* and ensures that the LVM algorithm is insensitive to parameter rescaling. By setting $[\mathbf{D}]_k^{-T} \, [\mathbf{D}]_k = I$, Eq. (7.2) can be recovered. λ^{LVM} is defined as the *damping parameter* as it regulates the contribution of the damping matrix when computing $\mathbf{x}_{k+1}$ for the next iteration step.

7.2.2 Search path

The search direction undertaken by the LVM method is controlled by the damping parameter, λ^{LVM}, in Eq. (7.2). In order to illustrate this, two scenarios are looked at. Firstly, λ^{LVM} is set very high, therefore causing the unit matrix to be more dominant than the Hessian matrix, $\boldsymbol{\nabla f}^T \boldsymbol{\nabla f}$. Consequently, Eq. (7.2) can be simplified to:

$$\left(\lambda_k^{\mathrm{LVM}} \mathbf{I} \right) \left(\mathbf{x} - \mathbf{x}_k \right) = -\boldsymbol{\nabla f} \left(\mathbf{x}_k \right)^T \left(\boldsymbol{f} \left(\mathbf{x}_k \right) - \mathbf{z} \right). \tag{7.9}$$

With the term $-\nabla \boldsymbol{f}\left(\mathbf{x}_k\right)$ being a descent direction opposite to the gradient function, $\boldsymbol{f}$, Eq. (7.9) looks very similar to the *steepest descent method*, which is known to find the region of the solution very efficiently, but to be very slow in converging to the solution [186]. In the second scenario, λ^{LVM} is assumed to be very small, leading to a negligible influence of the dampening effect on the LVM algorithm. As a result, Eq. (7.2) simplifies to:

$$\left(\nabla \boldsymbol{f}\left(\mathbf{x}_k\right)^T \nabla \boldsymbol{f}\left(\mathbf{x}_k\right)\right)\left(\mathbf{x}-\mathbf{x}_k\right)=-\nabla \boldsymbol{f}\left(\mathbf{x}_k\right)^T\left(\boldsymbol{f}\left(\mathbf{x}_k\right)-\mathbf{z}\right). \tag{7.10}$$

The computation of the next iterative step of Eq. (7.10) takes the form of $\mathbf{x}_{k+1}=\mathbf{x}_k-\frac{-\nabla \boldsymbol{f}\left(\mathbf{x}_k\right)^T}{\nabla \boldsymbol{f}\left(\mathbf{x}_k\right)^T \nabla \boldsymbol{f}\left(\mathbf{x}_k\right)}\left(\boldsymbol{f}\left(\mathbf{x}_k\right)-\mathbf{z}\right)$, which relates to a Gauss-Newton-type formulation, which is known to converge very rapidly in vicinity of the solution [187]. In this sense, the damping parameter initially takes large values until reaching the close neighbourhood of the solution. Its value then decreases to ensure rapid convergence. Consequently, the Levenberg-Marquardt method is considered to be robust, as it made up of two gradient methods which complement each other.

7.2.3 Parameter constraints

Parameter constraint methods are important in LVM to restrict movement of the algorithm over specific parametric ranges. This restriction can be crucial in certain cases when a function is only valid along a particular range. One such case is the linear elastic equation which only allows the Poisson ratio to span from -1 to 0.5 in order to yield positive Young's Modulus [188].

In this research, a simplified boundary enforcement method was employed. For each parameter, their lower bound and upper bound will be explicitly indicated in two vectors: $\mathbf{x}_{\min}$ and $\mathbf{x}_{\max}$ respectively. Such bounds ensure that the parametric domain is a hypercube. To guarantee that the LVM algorithm stays within these bounds, the latter are applied as constrains to the LVM formulation in the following manner [78]:

$$\begin{aligned}
\underset{\mathrm{x}\in S}{\text{minimise}} \quad & J_{\mathrm{LVM}}=\frac{1}{2}\left\|\boldsymbol{f}\left(\mathbf{x}_k\right)+\nabla \boldsymbol{f}\left(\mathbf{x}_k\right)\left(\mathbf{x}_{k+1}-\mathbf{x}_k\right)-\mathbf{z}\right\|^2 \\
\text{subject to} \quad & g\left(\mathrm{x}\right)=\frac{1}{2}\left\|\mathbf{x}_{k+1}-\mathbf{x}_k\right\|^2-\delta_k<0 \\
& \mathbf{x}_k+\left(\mathbf{x}_{k+1}-\mathbf{x}_k\right)-\mathbf{x}_{\max} \leq 0 \\
& -\mathbf{x}_k-\left(\mathbf{x}_{k+1}-\mathbf{x}_k\right)+\mathbf{x}_{\min} \leq 0.
\end{aligned} \tag{7.11}$$

To formulate the Lagrange function as in Sec. 7.2.1, Lagrange multipliers are introduced. However, due to additional constraints, $\boldsymbol{\mu}$ and $\boldsymbol{\gamma}$ is utilised along with λ^{LVM}. By imposing the Kuhn and Tucker conditions,

Eq. (7.11) evolves into:

$$\left(\boldsymbol{\nabla f}\left(\mathbf{x}_k\right)^T \boldsymbol{\nabla f}\left(\mathbf{x}_k\right) + \lambda_k^{\mathrm{LVM}}\mathbf{I}\right)\left(\mathbf{x}_{k+1} - \mathbf{x}_k\right) + \boldsymbol{\mu} + \boldsymbol{\gamma} = -\boldsymbol{\nabla f}\left(\mathbf{x}_k\right)^T\left(\boldsymbol{f}\left(\mathbf{x}_k\right) - \mathbf{z}\right) \tag{7.12}$$

$$\mu_i\left(\mathbf{x}_{k,i} + \left(\mathbf{x}_{k+1} - \mathbf{x}_k\right)_i - \mathbf{x}_{\max,i}\right) = 0 \tag{7.13}$$

$$\gamma_i\left(-\mathbf{x}_{k,i} - \left(\mathbf{x}_{k+1} - \mathbf{x}_k\right)_i + \mathbf{x}_{\min,i}\right) = 0 \tag{7.14}$$

$$\boldsymbol{\mu} \geq 0 \tag{7.15}$$

$$\boldsymbol{\gamma} \geq 0 \tag{7.16}$$

$$\mathbf{x}_k + \left(\mathbf{x}_{k+1} - \mathbf{x}_k\right) - \mathbf{x}_{\max} \leq 0 \tag{7.17}$$

$$-\mathbf{x}_k - \left(\mathbf{x}_{k+1} - \mathbf{x}_k\right) + \mathbf{x}_{\min} \leq 0. \tag{7.18}$$

The Kuhn-Tucker conditions, Eqs. (7.13) - (7.14), ensure that either the maximum or the minimum parameter boundary violation is targeted depending on which is applicable. As such, the following conditions are implied:

$$\mu\begin{cases} = 0 & \text{if } \gamma \geq 0, \quad \therefore,\ h_i = [x_{max}]_i - [x_k]_i \\ \geq 0 & \text{if } \gamma = 0, \quad \therefore,\ h_i = [x_{min}]_i - [x_k]_i\,. \end{cases} \tag{7.19}$$

The imposition of those boundary constraints falls under the category of constraint optimisation problems [189]. As such, the *active set method* is used [78].

7.2.4 Jacobian

One of the important aspects of the Levenberg-Marquardt algorithm is to correctly compute the Jacobian. The latter is crucial since it defines the Hessian matrix which must be inverted in order to compute the parameter step, that is, its update. Inappropriate estimation of the Jacobian can lead to non-invertibility or singularity issues such that the changes in parameters are incorrectly calculated.

The Jacobian is also referred as the sensitivity matrix [190] due to the fact that it represents the change in the cost function, $\boldsymbol{f}\left(\mathbf{x}\right)$, with respect to the change in parameter, $\mathbf{x}$, and is represented as:

$$J = \boldsymbol{\nabla f}\left(\mathbf{x}\right) = \begin{bmatrix} \boldsymbol{\nabla} f_1\left(\mathbf{x}\right) \\ \vdots \\ \boldsymbol{\nabla} f_m\left(\mathbf{x}\right) \end{bmatrix} \tag{7.20}$$

$$= \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \\ \frac{\partial f_m}{\partial x_1} & & \frac{\partial f_m}{\partial x_n} \end{bmatrix}, \tag{7.21}$$

where m is the number of points laying along the curve of the cost function and n, the number of parameters. Setting up the Jacobian requires evaluating the derivative of the cost function in close form. However, the PODI or EFG methods do not include the underlying formulation of the cost function, as the derivatives of such functions are not available. In order to circumvent this problem, different methods exist [187]. Two very popular methods to estimate the Jacobian are the Broyden rank-1 and the central difference method. The Broyden rank-1 can provide good convergence and stability. However, it can also

very likely not be able to find the solution [184], and sometimes introduces error for highly non-linear problems [191]. As such, the forward difference method is made use of in the following form [187]:

$$J_{ij} = \frac{\partial f_i}{\partial x_j} = \frac{f_i\left(\mathbf{x} + \delta\mathbf{x}\right) - f_i\left(\mathbf{x}\right)}{\|\delta\mathbf{x}\|}. \tag{7.22}$$

The estimation of the Jacobian is the most expensive part of the LVM algorithm. It requires n full scale simulations if the size of $\mathbf{x}$ is n. If n is quite large, then the total simulation time is correspondingly large.

7.3 Implementation details of LVM

Following the theoretical introduction of the LVM algorithm, the implementational details of LVM is presented in a flow chart (Algorithm 2). In particular, it illustrates the procedure to update λ^{LVM} which includes the use of an additional scaling parameter, ν^{LVM}. Moreover, different stopping mechanisms are also implemented. One of them is used to define the completion of an iteration and can only be satisfied if the error due to the new parameter, ϵ_{new}, is smaller than the updated one. If this is the case, then the parameter vector, $\mathbf{x}$, is updated along with $\mathbf{y}$ to compute a new scaling and Hessian matrix. However, in the case that the error due to the new parameter is not smaller, then $\mathbf{x}$, $\mathbf{y}$ and the scaling matrix, $\mathbf{D}$, remain unchanged. In this case the iteration is not considered completed. λ^{LVM} is then updated and consequently, the parametric step, $\mathbf{h}$, is re-computed. This whole sequence is then repeated till ϵ_{new} registers a drop.

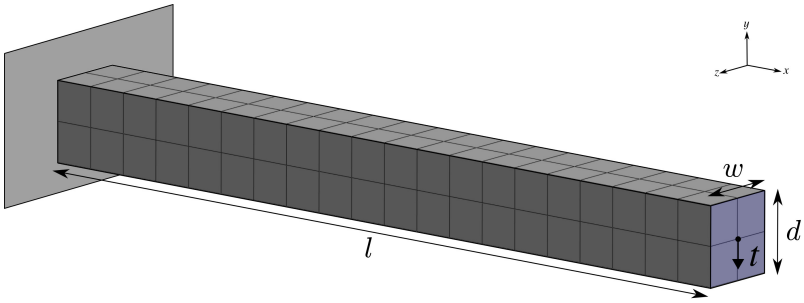
7.4 Example

An example was considered, in order to explore the use of the Levenberg-Marquardt method. As such, the MATLAB code developed by Guyon, the author of [78], was coupled with our in-house code SESKA, introduced in Appendix A. At any stage where LVM needs to evaluate $\boldsymbol{f}\left(\mathbf{x}_k\right)$, SESKA is executed in order to run an EFG simulation.

The example consisted of a cantilever beam which was fixed on one end, setting all displacement components $u_x = u_y = u_z = 0$, and subjected to a traction force of $1\,\mathrm{N/m^2}$ on the other end. The beam was 10 m long with a square cross-sectional of 1 m × 1 m as shown in Fig. 7.1. The geometry was then discretised using hexahedral elements, leading to a total number of 189 nodes. A traction force was applied in increments of $0.2\,\mathrm{N/m^2}$ to the free end. As constitutive law, a *Saint Venant-Kirchhoff* material was used, formulated from the Poisson's ratio, ν and the Young's modulus, E.

Algorithm 2 Levenberg-Marquardt algorithm

Initialise $\mathbf{x}$, γ_ϵ, γ_{tol}, λ^{LVM}, ν^{LVM}, $\text{Max}_{\text{iterations}}$
Compute $\mathbf{y} = \mathbf{f}(\mathbf{x})$ - $\mathbf{z}$ and $\mathbf{J}$
Num of experiment points, $N_{\text{exp}} = \text{length}(\mathbf{z})$
Average error, $\epsilon = \frac{\mathbf{y}^T\mathbf{y}}{2\times N_{\text{exp}}}$
updateDH = true
while $k < \text{Max}_{\text{iterations}}$ **do**
▷ *Update* $\boldsymbol{D}$ *and* $\boldsymbol{h}$
  if updateDH = true **then**
    Setup $\mathbf{D}$
    Determine $\mathbf{r} = -\boldsymbol{\nabla f}\left(\boldsymbol{f}\left(\mathbf{x}_k\right) - \mathbf{z}\right)$
    Calculate $\mathbf{H} = \boldsymbol{\nabla f}^T \boldsymbol{\nabla f}$
    updateDH = false
  $\mathbf{G} = \mathbf{H} + \lambda^{\text{LVM}}\mathbf{D}^{-T}\mathbf{D}^{-1}$
  Parameter spacing, $\mathbf{h} = \frac{\mathbf{r}}{\mathbf{G}}$
▷ *Boundary constraints calculation*
  Count number of violated bounds, N_{VB}
  while $N_{\text{VB}} > 0$ **do**
    Activate critical bound
    Enforce h_i^{viol} from Eq. (7.19)
    Re-compute h_i for non-active bounds
    Count number of violated bounds, N_{VB}
  $\mathbf{x}_{\text{new}} = \mathbf{x} + \mathbf{h}$
  Compute $\mathbf{y}_{\text{new}}$ and $\mathbf{J}_{\text{new}}$ using $\mathbf{x}_{\text{new}}$
  Find $L = \|\boldsymbol{f}\left(\mathbf{x}_k\right) - \mathbf{r}\|^2 - \|\boldsymbol{f}\left(\mathbf{x}_k\right) + \nabla\boldsymbol{f}\left(\mathbf{x}_k\right)\left(\mathbf{x}_{k+1} - \mathbf{x}_k\right)\|$
▷ *Determine* $\lambda_{\text{new}}^{\text{LVM}}$:
  Compute $\epsilon_{\text{new}} = \frac{\mathbf{y}_{\text{new}}^T\mathbf{y}_{\text{new}}}{2\times N_{\text{exp}}}$
  if $2 \times N_{\text{exp}} \times (\epsilon - \epsilon_{\text{new}}) > (0.75 \times L)$ **then**
    $\lambda_{\text{new}}^{\text{LVM}} = \frac{\lambda^{\text{LVM}}}{\nu^{\text{LVM}}}$
  else if $2 \times N_{\text{exp}} \times (\epsilon - \epsilon_{\text{new}}) \leq (0.25 \times L)$ **then**
    $\lambda_{\text{new}}^{\text{LVM}} = \lambda^{\text{LVM}} \times \nu^{\text{LVM}}$
▷ *Conclude as one iteration if error is small*
  if $\epsilon > \epsilon_{\text{new}}$ **then**
    $\mathbf{x} = \mathbf{x}_{\text{new}}$
    $\mathbf{y} = \mathbf{y}_{\text{new}}$
    $\epsilon = \epsilon_{\text{new}}$
    $k = k + 1$
    updateDH = true
▷ *Check for stopping criterion*:
  if $(\epsilon_{\text{new}} < \gamma_\epsilon)$ or $\left(\lambda^{\text{LVM}} < 1 \times 10^{-20}\right)$ or $(\text{norm}(\mathbf{h}) < \gamma_{\text{tol}} \times \text{norm}(\mathbf{x}_{\text{new}}))$ **then**
    Break



Dimensions	Value
Length, l	10 m
Depth, d	1 m
Width, w	1 m
Traction, t	1 kN/m^2

Figure 7.1: Cantilever beam example for the LVM.

The LVM is used to optimise only parameter E whereas ν is kept constant at 0.33. The admissible bounds of E are demarcated by $x_{min} = E_{min} = 50\,\mathrm{N/m^2}$ and $x_{max} = E_{max} = 25\,000\,\mathrm{N/m^2}$. $\boldsymbol{f}\,(x)$, which is equivalent to $\boldsymbol{f}\,(E)$, is the final tip displacement, u_y^E, of the centre node at the cross-section surface of the cantilever beam's free end. The reference tip deflection z is obtained by running a preliminary simulation with $E = 1100\,\mathrm{N/m^2}$ and therefore, it is expected that LVM will converge to this solution. The LVM parameters are set as follows: $\gamma_{\mathrm{tol}} = 1 \times 10^{-10}$ and $\mathrm{Max}_{\mathrm{iterations}} = 500$. For a starting point of $x = E = 8000\,\mathrm{N/m^2}$, the different scaling methods and the influence of λ^{LVM} and ν^{LVM} are investigated by looking at the convergence quality and calculation stability through the number of completed iterations and the number of times the boundary constraints were violated. In particular, the following factors deserve closer inspection:

Scaling methods: The scaling matrix is important for the proper definition of the damping matrix and the convergence of LVM. To investigate this effect, three methods are looked at. In the first method, the scaling matrix is set equivalent to an identity matrix. For the second method, it is assumed that a prior knowledge of the parametric space is known and therefore the diagonal of $\mathbf{D}$ is set using the initial parameters. Finally, in the third case, $\mathbf{D}$ is set up using the Jacobian and therefore corresponds to Eq. (7.5). The results obtained are summarised in Figs. 7.2, 7.4 and 7.3

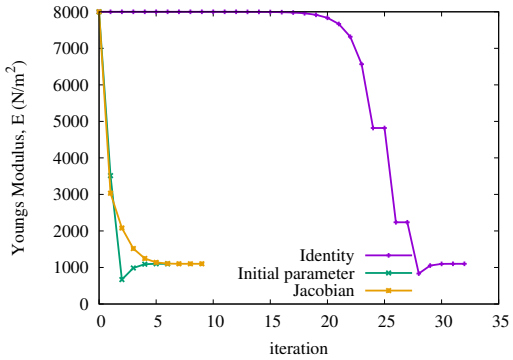


Figure 7.2: Change in LVM step direction with respect to different scaling methods.

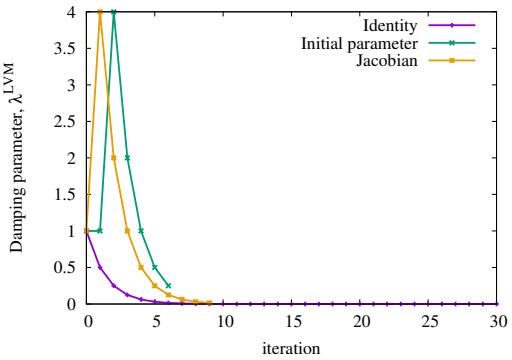


Figure 7.3: Change in the damping parameter of LVM with respect to different scaling methods.

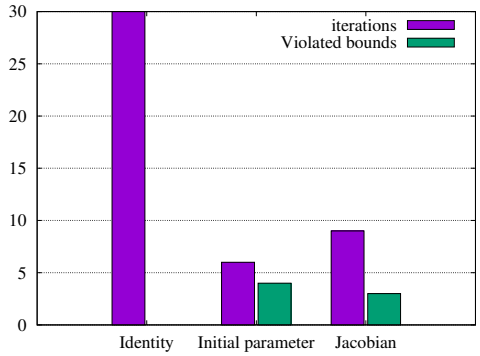


Figure 7.4: Change in number of iterations and violated bounds of LVM with respect to different scaling methods.

From the plot of Fig. 7.2, it is found that all the three methods converged to the exact solution. However, each required a different number of iterations. When the damping matrix is set as an identity matrix, LVM takes far more steps and also overshoots the solution when close to the convergence point. Such an observation is also made with regard to the initial parameter scaling method, but in the case of the latter, the calculation takes considerably less iteration steps while activating the boundary constraint algorithm only four times, as seen in the histogram of Fig. 7.4. Yet, for the case of the damping matrix defined by the Jacobian, the convergence to the solution is more smooth without overshooting the solution, and less likely to move out of the parametric space. As such, the Jacobian method was chosen in this research.

Damping parameter, λ^{LVM}: The damping parameter λ^{LVM} directly influences the step size of LVM by regularising the contribution of the damping matrix and by alternating between the steepest descent method and the Gauss-Newton method. To observe its effect on the calculations, a wide spectrum of magnitudes of λ^{LVM} is investigated, ranging from 1×10^{-5} to 1×10^{3}.

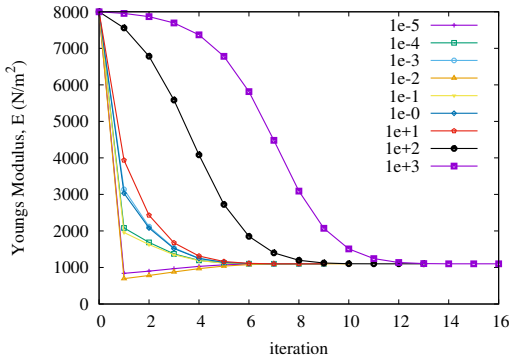


Figure 7.5: Change of Young's Modulus across iterations for different values of λ^{LVM}.

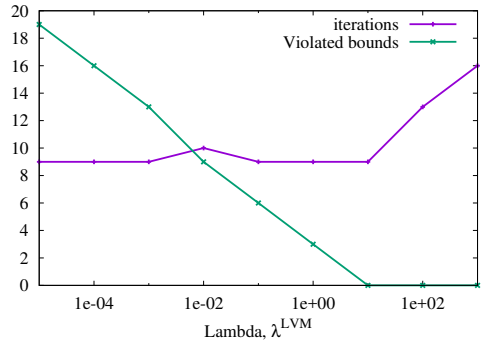


Figure 7.6: Change in number of iterations and violated bounds of LVM with respect to different values of λ^{LVM}.

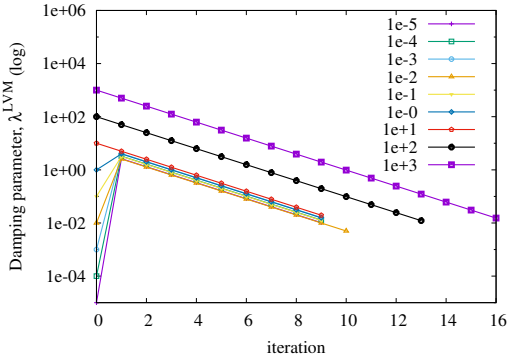


Figure 7.7: Evolution of λ^{LVM} across iterations for different starting values of λ^{LVM}.

From Fig. 7.2, it is found that as λ^{LVM} increases, the convergence of LVM to the solution is more smooth, with the occurrence of overshooting of the solution being less likely. This rise in stability seems to have also been achieved due to a decrease in the number of violated bounds but an increase in the number of iterations, as shown in Fig.7.6. A gradual increase of λ^{LVM} from 1×10^{-5} to 1×10^{3} implies a gradual shift of the LVM initial marching phase mechanism from the Gauss-Newton to the steepest descent method. On Fig. 7.7, it can be seen that even though a low λ^{LVM} value is started with, the algorithm favours an initial increase of lambda, up to 10, in order to favour the steepest descent mechanism. 10 is also the value after which no boundary violation is recorded. For any $\lambda^{\text{LVM}} > 10$, λ^{LVM} exponentially decreases (Fig. 7.7 uses a logarithmic y-scale axis) in order to eventually favour the Gauss-Newton method.

Scaling parameter, ν^{LVM}: To determine how fast the transition should happen between the two convergence mechanism of LVM, ν^{LVM} is used. If there is a negative change in error in the cost function, ν^{LVM} is used to decrease λ^{LVM} while for a positive change, ν^{LVM} then is used to increase λ^{LVM}. To estimate its influence on LVM algorithm, ν^{LVM} is varied from 1.1 to 10.

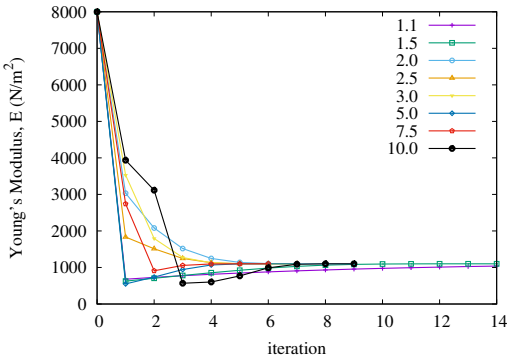


Figure 7.8: Change of Young's Modulus across iterations for different values of ν^{LVM}.

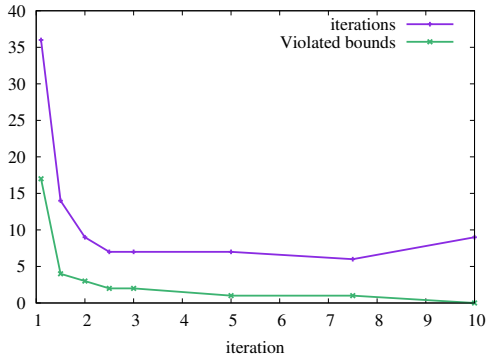


Figure 7.9: Change in number of iterations and violated bounds of LVM with respect to different values of ν^{LVM}.

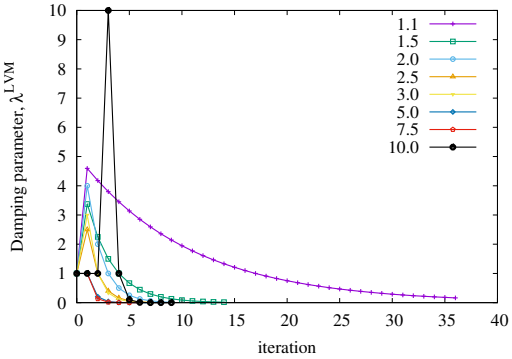


Figure 7.10: Evolution of λ^{LVM} across iterations for different values of ν^{LVM}.

From the results obtained from Fig. 7.8, it is found that when $2 < \nu^{\text{LVM}} \leq 3$, LVM converges smoothly and does not move beyond the solution point. However, Fig. 7.9 shows that the number of violated bounds and iterations decreases as ν^{LVM} increases. The reason for such an effect is, if ν^{LVM} is small, then a large number of iteration are spent around the region of large gradients, which is far from the solution. Lower ν^{LVM} values leads to a smaller increase in the initial λ^{LVM}, value as expected, due to $\lambda^{\text{LVM}}_{k+1} = \lambda^{\text{LVM}}_{k} \times \nu^{\text{LVM}}$. However, it is critical that the influence of the steepest descent method is reduced, once the parameters are close to the solution.

Chapter 8

Application of PODI to the Levenberg-Marquardt Method

In this section, parameter identification of cardiac mechanics problems using the Proper Orthogonal Decomposition with Interpolation and the Levenberg-Marquardt method (PODI-LVM) is investigated. Considering that the LVM-based computational inverse parameter optimisation is an iterative procedure involving a multitude of sequential numerical simulations, it is expected that the fast computation times achieved by PODI will significantly decrease computational costs of this task.

As such, two heart model examples are looked at where the end-diastole volume is targeted by searching for suitable material parameters. The first model is a single parametric domain, which is used to verify and validate the coupled Levenberg-Marquardt and the current PODI implementation, while the second example deals with a more complex problem where the database is multi-dimensional. At the end of the calculations, the performance impact on calculation time and accuracy are compared against a conventional full-scale modelling approach using the Element-free Galerkin method coupled with the Levenberg-Marquardt method (EFG-LVM).

8.1 Single-dimensional problem

For a single-dimensional problem, a heart geometry consisting of only the left ventricle is investigated. The geometry is defined by two half-truncated ellipsoids, according to:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1, \tag{8.1}$$

where a and b refer to the diameters while c relates to the depth of the ellipsoid. For a circular cross-section ellipsoid, $a = b$. Two separate ellipsoids of different diameter are created, one for the epicardium and another one for the endocardium. The dimensions and the depth to diameter ratio have been chosen such that the epicardium defines the end-systolic volume of a human left ventricle. Using a prescribed wall thickness, the dimension of the larger-diameter ellipsoid can consequently be obtained. The experimental data used to construct the ellipsoids are given in Tab. 6.1. The Neumann and Dirichlet boundary conditions used are identical to the ones applied to the bi-ventricle in Section 6.2.1. The

geometrical discretisation consists of 937 irregularly spaced mesh-free particles resulting in 2811 degrees of freedom, as shown in Fig. 8.1.

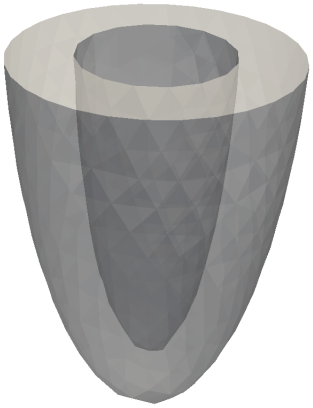


Figure 8.1: Left ventricle used for 1D LVM calculations.

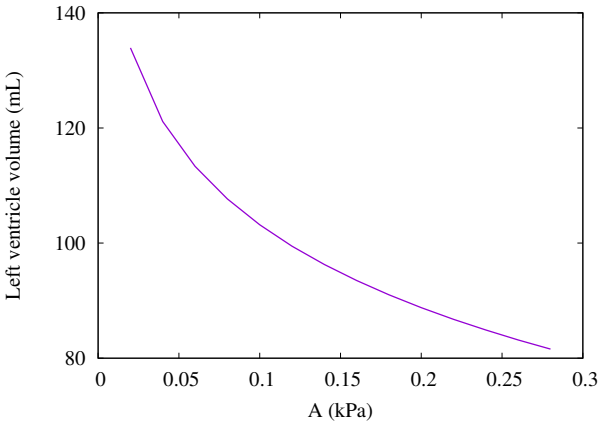


Figure 8.2: Evolution of the end-diastolic left ventricular volume across time.

In terms of material properties, most of the cardiac passive material parameters of Eq. 3.10 are fixed. $a_1 \ldots a_6$ are given in Tab. 6.5 and A_{comp} is set to 100 kPa. The fibre angles are also kept constant with θ_{epi} and θ_{endo} being $-57°$ and $59°$ respectively. The stress scaling factor, A, is the only parameter that is allowed to vary. As such, it will be the parameter optimised by the LVM algorithm to produce the approximated solutions.

The choice of the parametric range of A is based on the approach taken in Section 6.2.3, where the experimental upper and lower bound of end-diastole volumes are employed. As such, the new limits of A for the LV are found to be 0.02 kPa and 0.28 kPa. One interesting side observation made is the difference in the range of A for the LV and BV model, where the latter is found to be between 0.04 kPa and 0.22 kPa. Such a variation can be explained by the absence of a right ventricle. Here, the optimization target for LVM is chosen as an end-diastolic volume of $V_{\text{target}}^{\text{ED}} = 110$ mL which is located in the centre of the parametric space. In practice, the target volume would usually be obtained from clinical results. The starting value of stress scaling parameter, A^{start}, is given to be 0.26 kPa such that the left ventricle end-diastole volume is far from the targeted one and the LVM will be able to produce an elaborate path that will allow observation of the characteristics and differences between the LVM-PODI and LVM-EFG methods.

The above setup provides enough information to start the standard EFG-LVM-based parameter optimisation procedure. However, for the case of the PODI-LVM-based approach, a database of pre-computed full-scale simulation results is still required to be populated. These simulations need to cover the whole range of parameters previously specified for A and produce different end-diastolic volumes defining the upper and lower bounds of the database. Within those bounds, a parameter spacing of 0.02 kPa is selected to produce in total 14 full-scale simulation datasets. The end-diastole volume distribution across the database is given in Fig. 8.2.

With the LVM problem described, the calculations are then carried out. The PODI-LVM simulation is overall considerably faster. It took about 131 s using only one processor, while the EFG-LVM simulation took about 134 327 s, equivalent to 1 d and 13.3 h, using 2 processors. This result was as expected since

standalone PODI calculations had already been proven in Section 6.3.1 to be completed within less than 15 s.

Given that PODI simulations introduce additional approximation errors in their results when compared to EFG, it is also expected that the end point of the PODI-LVM and EFG-LVM will be different. This is because the PODI results fed into the LVM algorithm are interpolated from pre-existing full-scale simulation results, whereas EFG results have been obtained by continuously solving the governing equations for each LVM-updated value of A. In this sense, different optimisation paths and results may arise. An analysis of the current end result of the LVM algorithm supports this fact. For the PODI-LVM, $A^{\text{end}}_{\text{PODI}}$ is 6.42×10^{-2} kPa while for EFG-LVM, $A^{\text{end}}_{\text{EFG}} = 6.438 \times 10^{-2}$ kPa is achieved. If the EFG-LVM is considered the exact solution, then the error of the PODI-LVM simulations is computed to be about 0.35 %. The path of both methods is very similar, as shown in Fig. 8.3. This observation is confirmed, as the PODI error of A is always below 1 % across all LVM iterations. Based on these error values, which are considered to be negligibly small, it can be deduced that the accuracy of the PODI-LVM solution is acceptable.

Other aspects of the PODI-LVM algorithm are also analysed. Firstly, the evolution of the parameter λ^{LVM} is plotted against the LVM iterations, as shown in Fig. 8.4, and are found to be the same for both PODI and EFG, respectively. Secondly, this study looked at the evolution of the LVM's Hessian matrix for both methods. Their general evolutions are similar, however, the gradients using PODI are significantly smaller for the last three iteration steps, as can be seen in Fig. 8.5. This difference may indicate that PODI and EFG act on two different error planes. In this study, the term "*error-plane*" refers to the evaluation of the cost function (Eq. 7.1) using $V^{\text{ED}}_{\text{target}}$ across the parametric space. Since the PODI gradients are consistently higher than the EFG one, it is believed that the PODI error plane is also different and more specifically, steeper than the EFG one.

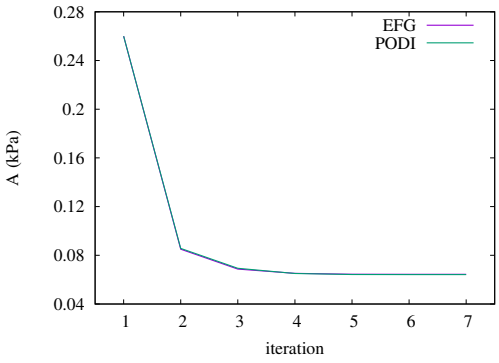


Figure 8.3: Evolution of A across iterations.

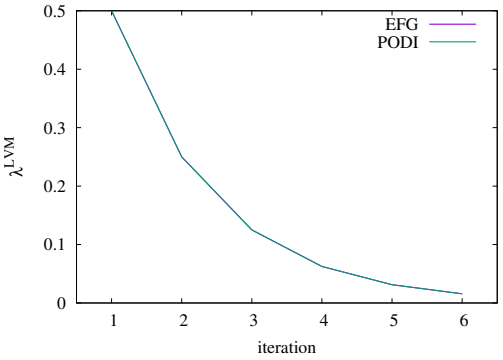


Figure 8.4: Evolution of lambda, λ^{LVM} across iterations.

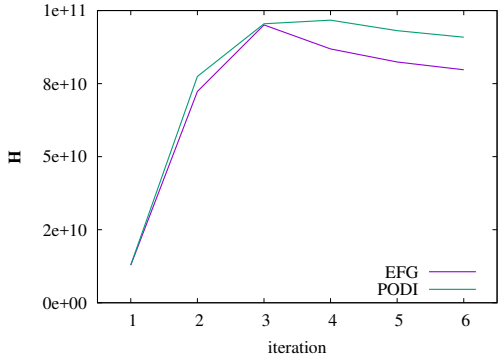


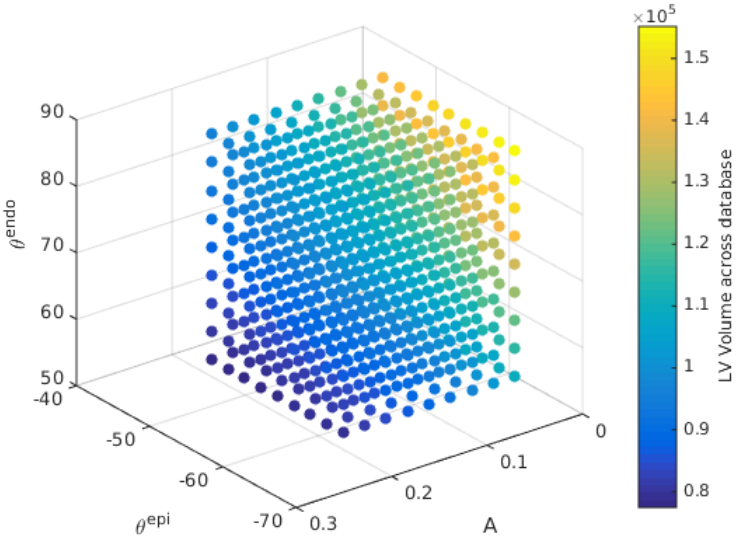
Figure 8.5: Evolution of the LVM's Hessian matrix, **H**, across iterations.

Combining LVM with POD, that is, the use of POD as a reduced order method but without interpolation, it is found that it results in fewer iteration steps than when using actual full-scale simulations [72]. In this research, however, where PODI was employed instead, the behaviour closely resembles that of EFG simulations. Such behaviour may be an indication that PODI better conserves the inherent qualities and behaviour of the full-scale simulations as provided by the corresponding datasets. A common trait across both POD and PODI optimisation calculations is that they are of several magnitudes faster than the standard approach, that is, LVM-EFG.

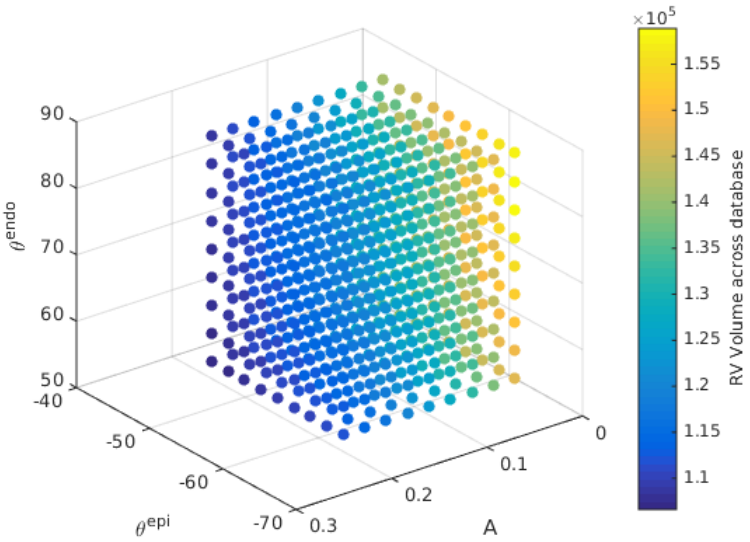
With the single-dimension parameter optimisation showing promising results, the complexity of the problem is now raised to a multi-dimensional problem that is investigated in the next section.

8.2 Multi-dimensional problem

For the multi-dimensional LVM problem, two types of complexities are introduced. Firstly, in addition to the elastic stress-scaling factor, A, the anisotropy characterised by parameters θ^{epi} and θ^{endo}, is included in the list of parameters to be varied. Secondly, a bi-ventricle model is used instead of a single ventricular one. The required database makes use of the existing BV database created in Section 6.3.2. The same levels of parameter domain discretisation are considered so that the convergence and accuracy of the PODI-LVM algorithm can be studied. There are four levels of discretisation refinement, namely PODI-DB1, PODI-DB2, PODI-DB3 and PODI-DB4.



(a) Left ventricular end-diastolic volumes (unit: mL).



(b) Right ventricular end-diastolic volumes (unit: mL).

Figure 8.6: Ventricular end-diastolic volumes across the level 4 discretised database.

The distribution of the left and right ventricle end-diastole volumes is given on a 4D plot in Fig. 8.6. The three axes of the plots are the parameters A, θ^{epi} and θ^{endo}. Every dataset is represented by a circle which is then shaded in a colour-coded format referring to different ventricular volumes. The bright yellow colour defines a maximum volume, which is 155.17 mL for the LV and 158.89 mL for the RV. The lowest volume, which is 77.45 mL for the LV and 106.61 mL for the RV, corresponds to the dark blue colour. Hence, any volume between the upper and lower bound of the range is interpolated, as represented on the colour bar.

Using the 4D plots generated, the volume distribution is analysed in order to understand the influence of each parameter on the BV model. In Fig. 8.6a, it can be observed that as the value of A and θ^{epi} decrease while θ^{endo} increases, the left ventricular volume rises. This behaviour can be explained by assessing each parameter individually. In the case of the stress scaling coefficient, decreasing its magnitude leads to a less stiff myocardium, which therefore allows the left ventricular wall to be more compliant during expansion. For θ^{epi} and θ^{endo}, the corresponding decrease and increase of angles leads to the fibres being more closely aligned with the longitudinal axis of the LV and consequently larger end-diastole volume, a similar observation also reported by Palit et al. [192]. This alignment is believed, to favour less resistance along the circumferential direction of the ventricle, leading to more expansion in the transverse direction and allowing the diameter of the ventricle to extend effortlessly. Regarding the right ventricle, similar behaviour is observed, showing that the effect of the stress scaling coefficient and fibre angle are similar across the heart. Due to this reason, an approach where both the left ventricular and right ventricular volumes are used to find the material values, is deemed inadequate. If the two ventricular volumes are selected such that they are located at opposite sides of their respective volume-parameter domain (Fig. 8.6a and Fig. 8.6b, respectively), then the solution obtained using the LVM is bound to be not feasible, and is unrealistic. Instead, a point in the domain representing a compromise will be targeted. Consequently, this approach is not pursued.

In this research, the left ventricular volume was the sole data used in the parameter identification process. This choice was made at the expense of the right ventricular volume, based on the fact that the left ventricular wall is mechanically the dominant part in a BV model and is therefore more likely to dictate how the heart behaves globally.

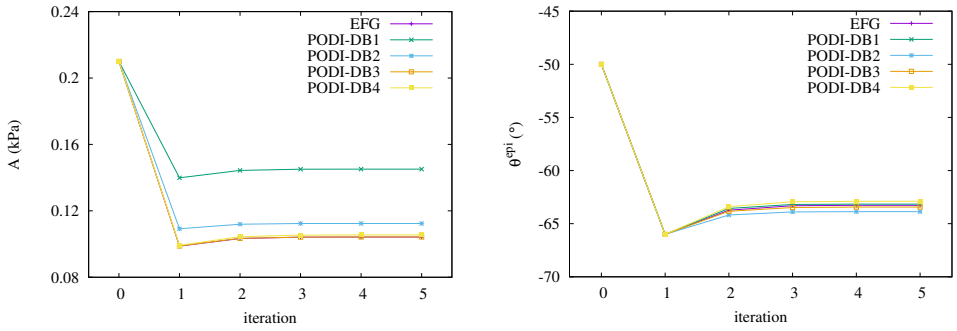
With the database populated, the LVM problem is next defined. The energy conserved is set to 99 % and the targeted end-diastole volume is specified to be 110 mL for the left ventricle of the BV. λ^{LVM} and ν^{LVM} are both set to their default values, which are 1 and 0.5 respectively, as indicated in Section 7.4. The stopping criterion specified an accuracy of 1×10^{-2} which is deemed reasonable since the magnitude of the targeted volume is of several orders higher. The parameter optimisation calculation is carried out for both the EFG and PODI method, with the former representing the exact solution.

Type	No. of datasets	Parameters		
		A	θ_{epi}	θ_{endo}
PODI-DB1	8	0.145	-63.16	76.28
PODI-DB2	27	0.112	-63.87	73.74
PODI-DB3	125	0.104	-63.44	72.91
PODI-DB4	729	0.106	-62.89	73.71
EFG	-	0.104	-63.27	73.30

Table 8.1: Final optimised parameter values.

The first set of results analysed is the calculation speed. On average, PODI-LVM took about 631 s while EFG-LVM lasted for about 1 d and 17 h, or more precisely, 147 025 s. As expected, parameter optimisation using PODI is by a factor 233 quicker than with EFG. In both cases, 24 consecutive simulations are required. The number of required PODI simulations remained constant for the different

parameter domain discretisation refinements. Likewise, the number of iteration steps for both the PODI-LVM and EFG-LVM, is exactly the same. In total, five iteration steps are required to converge to a solution. On the other hand, with regard to the optimisation path taken and the solution reached, some differences could be observed. The end points reached by both methods are summarised in Tab. 8.1. In all cases, the PODI results are close to the ones achieved by EFG, with a targeted end-diastolic LV volume of 110 mL. However, it is noticed that the least accurate results are obtained with the coarser discretised databases, namely PODI-DB1 and PODI-DB2. With increasing discretisation, the accuracy improved, where PODI-DB3 and PODI-DB4 provided the best fit. This trend is also confirmed when looking at the evolution of each parameter across the LVM iterations, as given in Fig. 8.7. As the database is built from precomputed EFG simulation results, it can be expected to find for PODI-LVM a similar optimisation path and behaviour as that observed for EFG-LVM. This was indeed found for a decreasing parameter domain spacing, which is in line with the results of the single-dimensional parameter domain investigated in Section 6.3.2.



(a) Parameter: A.

(b) Parameter: θ^{epi}.

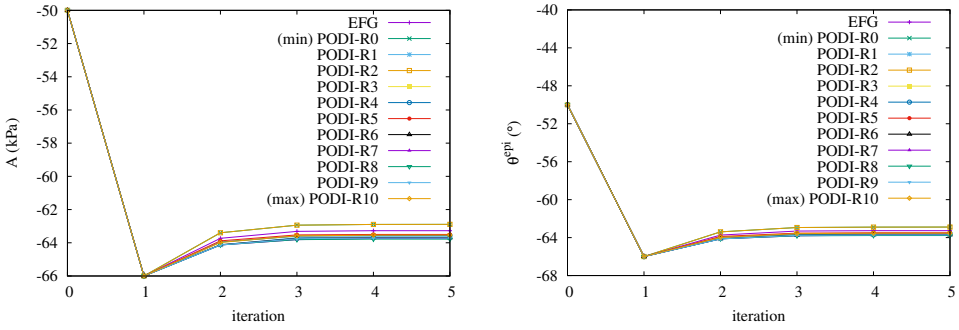
(c) Parameter: θ^{endo}.

Figure 8.7: Progression of the parameters across the LVM iterations.

An interesting observation is also made from the plots in Fig. 8.7 when comparing the error of the final value of each parameter as obtained with PODI-LVM. The maximum L_2-error norm, across the different stress scaling coefficients, reaches as high as 0.4, which is considerably higher when compared against θ^{epi} and θ^{endo} where the highest error is only about 4.1×10^{-2}. This discrepancy can be related to the preliminary study done in Section 6.3.2. There, it is observed that, as opposed to other parameters, A had more impact on the accuracy of the PODI solution when the databases are refined. This is deduced from the plots in Fig. 6.33 where comparing the error gradients of each parameter, showed that refining

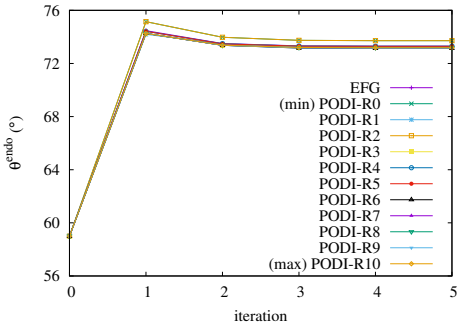
the parametric spacing of A leads to a bigger drop in error. Given that the same BV databases are used here, different sensitivity of the interpolation with respect to parameters A, θ^{epi} and θ^{endo}, as previously reported in Section 6.3.2, can also be expected to affect the PODI-LVM results. Indeed, refining the parameter spacing of the datasets with respect to A shows greater improvements in solution accuracy. For θ^{epi} and θ^{endo}, the change in error across the database is already minimal. As such, refining the datasets spacing along those parameters, does not provide a greater gain in accuracy.

Further investigations are carried out to determine the effects of different influence radii on LVM. In order to have a wider range of supporting nodes, the database with the highest refinement level, PODI-DB4, is used. The radius, r_{database}, is expanded in 10 equally spaced increments, where $r_{\text{database}} \in [D, 2D]$ and D is the uniform distance between two parametric points. Consequently, the 10 calculations are labelled as PODI-R1 to PODI-R10. The POD energy conservation is fixed at 99 %. The generated results are analysed and plotted in Figs. 8.8a - 8.8c. From these figures, it can be observed that the change in influence radius does not lead to drastic changes in the final solutions of the LVM. All the PODI solutions are close to the EFG-LVM one, without any apparent trends. The convergence behaviour of both the EFG-LVM and PODI-LVM calculations, is the same as each of the required five iterations.



(a) Parameter: A.

(b) Parameter: θ^{epi}.



(c) Parameter: θ^{endo}.

Figure 8.8: Progression of the parameters across the LVM iterations for different influence radius.

Finally, the variation in energy conserved is considered. As above, the PODI-DB4 database is utilised to mobilise a different number of datasets, leading to a large spread of energies which can be investigated. To favour the latter, the influence radius is chosen slightly larger. Consequently, on average, 51 datasets are selected, with the minimum being 27 on the border of the database and a maximum of 64 in the middle. The targeted volume and cut-off tolerance values of the LVM algorithm are kept to 110 mL and

1×10^{-2}.

From the results obtained, the parametric solution difference between EFG-LVM and PODI-LVM is measured via the L_2-error norm for their respective energy conservation level and is plotted in Fig. 8.9. A general trend across the plot shows that the errors decrease as energy increases. However, from 70 % to 95 % and from 99.9 % to 100 %, no consequent changes are registered. For the former range, only 1 POM is conserved (as given in Fig. 8.12), indicating that the first POM accounted for more than 95 % of energy while for the second range, it is believed that low energetic POMs could not influence the change in error norm as those behavioural modes are noises with negligible effect. In-between 95 % and 99.9 %, a sharp drop of error is registered, leading to a deeper investigation of the path direction undertaken by the PODI-LVM calculations. For all conserved energy levels, the first step of the LVM-PODI calculations are the same in terms of datasets mobilised. This is because of the direction of descent, which is the gradient of LVM, yet to be computed, leading the initial gradient to be the same and the supporting datasets to have similar parametric points along each parametric dimension, as given in Tab. 8.2. However, in the next step, the descent direction is computed from the previous step. Since the previous step is influenced by the energy conserved, the descent direction changes across the different levels of energy conserved. Since they are all different, the gradient of descent is considered to be a function of the energy conserved by PODI. The descent direction generally influences the trajectory of the PODI-LVM calculation. However, in this case, the change in gradient of descent is only influential for those calculations where higher levels of energy are conserved. This can be seen when comparing the two energy groups of range 70 % to 95 % and 99 % to 100 %. With the path direction changing, the supporting datasets also change. The parametric points of the new supporting datasets, for the energy conservation levels of 99 % to 100 %, along each parametric dimension is compiled in Tab. 8.3. With Tab. 8.2 and Tab. 8.3 compared, it can be found that for parameter A, the dataset referring to $A = 0.1525\,\text{kPa}$ is removed at the expense of $A = 0.0625\,\text{kPa}$, where the latter is indeed closer to the EFG-LVM reference solution of $A = 0.104\,\text{kPa}$. For f_{epi}, the datasets of the parametric points associated with $f_{\text{epi}} = 59.25°$ are added, while none are removed. Since the new supported datasets are closer to the actual solution and more datasets are available to adequately define the POMs, a sharp increase in calculation accuracy is obtained.

The sharp accuracy increase is localised around 99 % which can again be attributed due to the different error plane generated and consequently leads to a different path. Due to the discrete parameter domain, a slight change of path leads to different mobilised datasets. As can be seen in Fig. 8.11, the 99 % calculation mobilised around 54 datasets, which is more than those for the energetic range of 70 % to 95 % and more than those of 99.9 % to 100 %. As such, it is believed that for the 70 % to 95 % case, not enough datasets are available to generate accurate MLS interpolants, whereas those of 99.9 % to 100 % provided too much data, which consequently lead to a more smoothing effect than interpolatory ones. Hence, the 99 % energy PODI calculation mobilised the right amount of datasets to give the most accurate solution.

Finally, with regard to the performance of PODI-LVM, each of the calculations took on average 3517 s to converge to the solution. This value is higher than the one recorded at the start of this section, where it is around 630 s. The main reason for such an increase is due to the usage of a larger influence radius, which mobilises a higher number of datasets. Across the conserved energies, the PODI calculation times are found to increase, as shown in Fig. 8.10, which is to be expected due to the increase in POMs used for interpolation. Additionally, this gradual increment is due to the process of reading a different number of datasets as shown in Fig. 8.11, which on average took more than 80 % of a PODI-LVM calculation time.

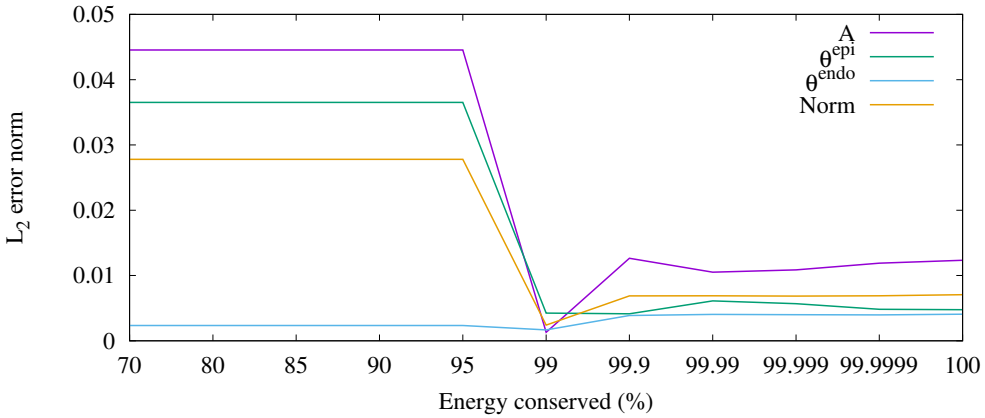


Figure 8.9: Parametric error as the energy conserved is varied.

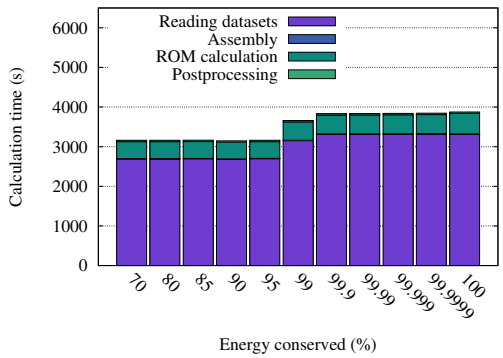


Figure 8.10: Change in LVM-PODI time with respect to energy conserved.

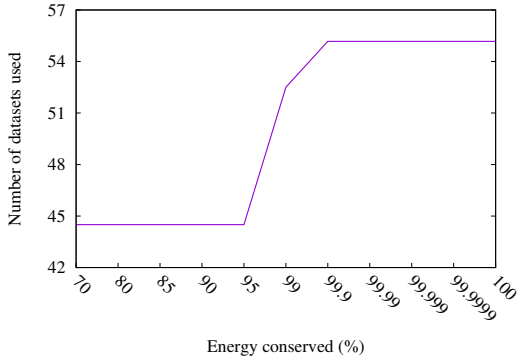


Figure 8.11: Change in number of datasets used for the PODI-LVM calculation with respect to energy conserved.

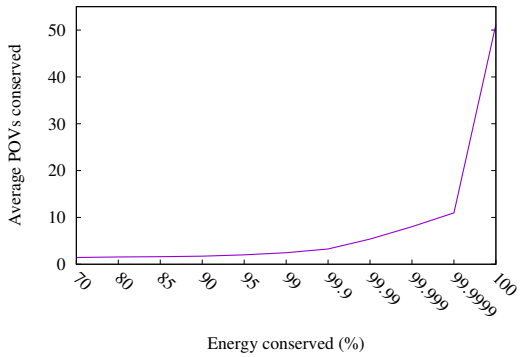


Figure 8.12: Change in POVs with respect to energy conserved for the LVM-PODI calculations.

Parameters	Selected datasets		
A (kPa)	0.175,	0.1975,	0.22
f_{epi} (°)	48,	50.25,	52.5
f_{endo} (°)	55,	59.25,	63.5

Table 8.2: Parameters selected by PODI-LVM along each parametric dimension during the first iteration.

Parameters	Energy conservation range, %	Selected datasets				
A (kPa)	70-95		0.085,	0.1075,	0.13,	0.1525
	99-100	0.0625,	0.085,	0.1075,	0.13	
f_{epi} (°)	70-95		61.5,	63.75,	66	
	99-100	59.25,	61.5,	63.75,	66	
f_{endo} (°)	70-95	67.75,	72,	76.25,	80.5	
	99-100	67.75,	72,	76.25,	80.5	

Table 8.3: Datasets selected for each parametric dimension and energy conservation level.

Part V

Summary, Conclusion and Future work

Chapter 9

Summary

Following the results and observations given in Chapters 6 and 8, the key findings of this research work are now highlighted and summarised. The main aspect of this research is to achieve real-time time modelling of the heart, which corresponds to a calculation frequency that is greater than 300 Hz. Additionally, the quality of the results is analysed and ways to enhance their accuracy are investigated.

The first example that was considered was a benchmark problem written in MATLAB. The latter was a strut subjected to a wave force travelling back and forth. A database of varying Young's modulus values was constructed and the PODI calculation ran for Young's modulus located in the middle of the parametric domain. The results obtained and presented in Section 6.1 were promising, with the calculation time ranging between 16 s to 19 s, which was 24 times speedup compared to full-scale simulations. However, it was found that 76 % of the calculation time was spent on reading off-line results stored on a storage medium, which in this research was a hard-disk. Generating the MLS interpolants took about 0.5 s while the PODI calculation took on average 3.9 s. For the displacement field, the calculation frequency of 1009 Hz per step was reached. The error was found to be minimal across the displacement field results. Further study was carried out on the ensemble matrix of the PODI calculation in order to find out whether the zero-mean or non-zero-mean method is better since, across the literature, such as [1, 29, 137, 152, 153, 156, 157, 158], no general consensus was found. Using the same strut example, both methods were compared. It was first found that the zero-mean method increased the calculation time by 50 % to 65 %, compared to the non-zero-mean method. This was due to the additional operations of computing the column mean vector using the ensemble matrix and then subtracting it from every column of the ensemble matrix. For the zero-mean ensemble matrix, it was found that the energy of the first POM was redistributed to the less energetic ones. However, due to the subtraction process leading to a decrease of magnitude in the entries of the zero-mean ensemble matrix, the last POMs often had an associated low or negative energy. Nevertheless, the zero-mean was found to adequately capture the dominant modes as an increase in solution accuracy of up to 60 % was achieved, for the range of 80 % - 99 % energy conserved.

With these encouraging results, the PODI method was implemented in C++ to increase its overall efficiency when dealing with cardiac examples. A cardiac example was considered in this work, which was a bi-ventricle 3D model constructed from dimensions extracted from experimental data, and was presented in Section 6.2. From there, a fibre generator algorithm was implemented to assign one fibre and

two cross fibre directions to the heart geometries in order to have a realistic heart model behaviour. The algorithm, based on the work of [102], was modified using Rodrigues' rotation formula in order to have a varying sheet and sheet-normal fibre directions across the transmural direction of the heart's ventricle, as given in Section 3.4.1.3. In the first example discussed in Section 6.3.1, a single parametric database of a BV model was constructed. The parametric domain, in this case, was defined using the stress scaling coefficient, and each dataset consisted of diastole filling solution fields. The PODI calculation was found to be 2050 times faster than the Element-free Galerkin method full-scale simulation, as it took only 14.05 s. The computation of the displacement field of the BV for a single step was found to be carried out at a frequency of 1074 Hz. The total calculation time of the displacement, stress and strain fields were found to be ready within 0.43 s. The accuracy of all the results was high, especially the pressure-volume curve relationship, which is an important information for medical doctors, had a L_2-error norm below 6×10^{-3}. The evolution of the displacement field and pressure-volume curve error across the parametric domain was analysed and found to be higher at the boundary than in the middle. This behaviour is rooted in the general MLS interpolation characteristics, which arises due to the availability of fewer data at the boundary. This also demonstrates that the PODI method inherits some of the main characteristics of the interpolation scheme it makes use of. A comparison of the zero-mean method against the non-zero mean method was also investigated in this example. The same behaviour, noted in the benchmark problem, was also observed here. However, in this case, the calculation time was longer by only 3 %, while the accuracy of the displacement field solution increased by 55 % between the energy conservation range of 80 % – 99 %. As such, it was decided that the zero-mean method would be set as the default PODI calculation set-up.

The next step of this research work was then to carry out a multi-dimensional PODI calculation while looking at the evolution of the error norm when different parametric discretisation and the MLS supporting zone were used. In this case, three parameters were used to establish the parametric domain as shown in Section 6.3.2. They were the stress scaling coefficient and the two fibre angles, on the endocardium and epicardium. The first investigation that was carried out was monitoring the influence of reducing the parameter spacing in the parametric domain. In this case, it was found that the error was most sensitive to the stress scaling parameter, as its error gradient was higher than the fibre angles. As such, smaller stress scaling parameter spacing will lead to a faster decrease in error. The second most sensitive parameter was found to be the endocardium fibre angle, followed closely by the epicardium fibre angle. The second investigation consisted of looking at how the support radius of the MLS interpolation influences the PODI calculation, and more specifically, the optimum number of supports required to produce the most accurate solutions. From the results presented, it was found that the most reliable solutions were obtained when the support radius is very localised. That is when only the two closest datasets across each parametric dimension were chosen. Consequently, running a PODI calculation using the 8 neighbouring datasets from the current 3D parametric database led to solution fields that were found to be more accurate than those obtained with the single parametric PODI calculation. In terms of displacement field calculation speed, a calculation frequency of 909 Hz was achieved. This frequency was found to be within the operational frequency range of 500 – 1000 Hz of haptic devices [30].

Afterwards, in Section 6.4 the PODI procedure was extended to accommodate data extracted from arbitrary heart anatomical configurations and corresponding mesh-free particle discretisations. This approach was found useful for patient-specific PODI cardiac modelling since every heart is geometrically

unique. A new database consisting of nine different hearts was created to represent a variation of unique hearts. They had different mesh configurations and shapes. Before carrying out the PODI calculation, a template grid was needed to standardise all solution fields of the datasets in order to extract their POMs. Once the PODI results were obtained, they were then projected to the geometry of the problem-at-hand. Two standardisation schemes have been investigated. The first was the cube grid method. The PODI calculation took on average 4 min. The costliest operation was standardising the solution fields of the datasets, as it took about 95 % of the total time. The displacement field error norm was high and a visual inspection of the displacement plot revealed non-uniform deformations, which can be interpreted as being non-physical, across the whole geometry. Increasing the number of cube grid points decreased the displacement field error norm only slightly, while the calculation time increased linearly. The non-physical deformations on the surface of the problem-at-hand heart geometry were still present, the reason being due to the empty entries of the ensemble matrix being up filled with zeros. The second standardisation scheme was the heart template approach. In this case, the dataset hearts were first registered onto a template heart using the Coherent Point Drift algorithm. Since all dataset nodes were closely centred around the template nodes, empty entries were not encountered. The result obtained from this setup revealed that the solution fields were more smooth and accurate. However, some of the non-physical behaviours, observed in the cube grid approach, were still present. This was found to be due to the registration failing around sharp edges of the heart template geometry, where there was a lack of points. As such, template hearts with higher nodes densities were studied. Those template hearts were found to decrease the solution field errors, concentrate and increase the energy of the most dominant modes, and most importantly, prevent the occurrence of non-physical deformations. In terms of PODI calculation time, the heart template standardisation process took slightly longer than the cube template grid, as it was 5.8 min.

Next, a full heart cycle was modelled in Section 6.5. The main aspect of that section was the synchronisation of the phases of a heartbeat, the time standardisation procedure, and temporal PODI calculation. For that, a database was built using different LV end-diastole pressures, and applied to a bi-ventricle model. The results showed that PODI was about 2200 times faster than the full-scale simulation. In terms of error, the displacement, stress and strain fields were found to be within the reasonable limit. However, when the distribution of the displacement error was looked at across different phases of a heartbeat, it was found that the error during isovolumetric contraction was the highest, about three times higher than during diastole filling. This was confirmed when looking at the pressure-volume relationship of the LV, as a non-smooth curve was obtained during IVC. The ejection and isovolumetric relaxation phases had the lowest error overall. When a deeper investigation was carried out, it was found that the solution fields were the most different across the datasets during the IVC, as the cavity volumes of the datasets were far apart. A POV analysis revealed that the first POM during the IVC stage was less energetic. Consequently, the energy conserved had to be increased in order to obtain a low error solution field and smooth LV-PV relationship.

From the above results, it was deduced that PODI can produce solution fields of a problem-at-hand rapidly and accurately. As such, the final step of this research was to investigate the coupling of PODI with the Levenberg-Marquardt algorithm, a gradient-based optimisation algorithm. The first example that was looked at in Section 8.1, was an LV with a targeted end-diastole volume. The stress scaling was the only parameter allowed to vary. The PODI-LVM calculation was found to be about 1025 times

faster than the full-scale simulation, which was EFG-LVM. The end-parameter-point reached by both methods were slightly different. This slight difference was considered as an error by the PODI calculation, which was below 1 %. This error was present during the whole convergence path of the PODI-LVM calculation. As such, it is believed that the PODI-LVM calculation may be acting on a different error plane to the EFG-LVM calculation. A second example, where the fibre direction along with the stress scaling parameter, was investigated. In this case, a bi-ventricle model was used and again the goal of the LVM calculation was to reach a particular LV end-diastole volume. The databases considered for the PODI-LVM calculations were the same as those in the multi-dimensional PODI calculation carried out. Overall, three different types of studies were conducted. The first one investigated the influence of decreasing parametric spacings. In that case, it was noted that the end-parametric point solution became more accurate. However, the drop of error was not the same across all parameters. Its drop was more consequent for the stress scaling parameter. This result was found to be in line with the earlier database sensitivity analysis carried out in Section 8.2, showing that the stress scaling parameter was the most error-sensitive parameter. The next investigation carried out was to look at the influence of changing the supporting radius of the PODI calculation for a fixed database discretisation. In this case, no drastic changes in error were observed. The last study to be conducted was the effect of an increase in energy conservation across the PODI-LVM calculation path. Here, it was found that, in general, increasing the conserved energy modified the LVM path and made the end-parametric-point more accurate. More importantly, it showed that, increasing the energy conserved, made the PODI-LVM act similarly to the EFG-LVM calculation. The main reason why this was the case was because the database of the PODI-LVM calculation was built using EFG calculations.

Chapter 10

Conclusion

From the results generated through-out this thesis, it can be deduced that real-time modelling of the heart is achievable. However, this would be feasible only under circumstances where the process time of reading datasets from a hard-disk medium, standardisation of steps and geometries, and post-processing of results are omitted. The actual PODI calculation itself is very efficient and fast. Even with an interpreted language such as MATLAB, calculation frequencies beyond 1000 Hz can be attained. To make use of this fast computation speed, the post-processing phase should be coupled with a fast visualisation tool in order to make the whole PODI calculation process adaptable to haptic environments.

Regarding the technical characteristics of the PODI method, several observations are made. Firstly, the POVs are good indicators of how dominant the behaviour or POMs are. A very sharp decline of the energies, from the most to the least dominant POM, indicates that the behaviour of the datasets have been easily identified. Hence, the PODI solution is likely to be accurate and generated in a short time frame, as only the few most dominant modes will be used. In the case where the energy is scattered across the POMs, difficulty in establishing the dominant modes is encountered. Hence, the accuracy of the PODI solutions drops dramatically. This has been found to be the case during, firstly, the degrees of freedom standardisation process, where the registration of the dataset geometry has not been carried out successfully, and secondly, during a full heart cycle simulation, especially with the IVC PODI calculation phase. The second observation made is with regard to the interpolation scheme of PODI. It is found that properties of the former directly influence the PODI calculation. The database analysis, along with the study of a different supporting radius, reveals the quality of the PODI solution fields to be affected by the MLS interpolation scheme. As such, choosing the adequate interpolation method is of prime importance. However, this decision process can be made easier if the smoothness of the ensemble data in the low-dimensional space is investigated, as noted by Tan et al. [165] and Burkardt et al. [140].

The other objective of this research dealt with the extension of the PODI method to the use of arbitrary meshes. This was also successfully achieved. Two methods were considered. The first, which is the cube grid template approach, has the advantage of providing faster solutions. However, it does not conserve the details of the datasets' projected solution fields. As such, the PODI results were not smooth and accurate. This was due to the zero value introduced for any template mode lying outside the dataset heart geometry. Consequently, the interpolated solution assigned to the template nodes are derived from dataset solutions and the introduced zero-values. The heart grid template approach was found to work significantly better.

After the CPD registration, the details of the datasets projected solution fields are conserved and as such, the PODI results are smooth and carry less errors. The main reason for such effect is due to the enforcing of all dataset heart nodes to be around the template nodes after the registration. This, therefore, ensures that the template nodes are assigned interpolated solutions from the dataset solution fields only. However, this is only achieved if the template mesh has enough points during the CPD calculation to allow for a proper registration. One limitation found with both template approaches is that real-time modelling is difficult to attain. The registration process is computationally expensive due to the Point-In-Polygon problem and the projection of the solution fields in both approaches, and the CPD algorithm, in the heart template approach. Two ways that could perhaps solve these problems are by considering parallel computing of those algorithms, and to look for a fast, non-rigid transformation method.

The final step of this research was to look at the full heartbeat PODI calculation. A time standardisation method was implemented and found to work well. Its main advantage is that it can identify the start and ending time step of each phase of a heartbeat, which are found to evolve with different set of parameters. Hence, those time points can be found for any problem-at-hand calculation and thus, the new PV loop can be correctly reproduced. Additionally, the time standardisation process employs the temporal PODI calculation, which can be used to reduce the number of steps across any phase of a heart-beat for any datasets. Thus, the PODI calculation can still be carried out at real-time speed. The only limitation that was found during the time standardisation process is that high energy conservation values are required to ensure that accurate and smooth solutions are obtained during the IVC phase.

Finally, as a direct and useful application of cardiac mechanics PODI, its coupling with a gradient-based optimisation method such as the Levenberg-Marquardt algorithm, was explored. As expected, the PODI-LVM calculation speed was considerably faster. However, it was found to reach a slightly different solution. This difference was found to be present during the whole trajectory of the PODI-LVM calculation. Hence, it is believed that PODI could be acting on a different error plane than that of the EFG-LVM method. Nevertheless, it was observed that PODI-LVM was behaving in a similar fashion to the EFG-LVM method. One reason why this could be the case is due to the fact that the datasets of the PODI calculations where all constructed using EFG.

Chapter 11

Future work

The content of the future work presented below are some of the work that could further be investigated to enhance the applicability of cardiac modelling within the reduced order modelling framework:

Linear properties of PODI. The reconstruction of a displacement field from POMs and coefficients as shown in Eq. 5.1, shows that the POD method is a linear procedure. This is irrespective of whether the ensemble matrix from which the POMs were extracted contained non-linear data [126]. As such, the application of those POMs to non-linear problems raises a lot of questions about the approximative characteristics of the POD in generating results of high accuracy. Within the context of the PODI method, the usage of linear POMs to create low dimensional spaces over which non-linear data from the ensemble matrix are projected, is poorly understood. Hence, Tan et al. [165] and Burkardt et al. [140] questioned the smoothness of the coefficients in the reduced dimensional spaces. Yet, a rigorous mathematical investigation of the POD method can only be beneficial. Such a study could potentially help in understanding under which circumstances the POD generates optimal results, and more importantly, also, explain the behaviour of projected data in the low-dimensional space. In the context of the PODI method, this will essentially provide more details on the smoothness properties of the projected coefficients, and consequently help in choosing a more appropriate interpolation scheme.

Non-uniform parametric domain. In this work, only uniformly spaced parametric point domains have been investigated. These domains provide the advantage of consistent error distributions across the database, which consequently allows for analysis of the different methods coupled with PODI. However, in practice, such a domain setup is not recommended. One of the biggest problems that may be faced later, is database refinement. Usually, two ways can be considered for uniform database refinement. In the first case, a new grid can be continuously generated with a slightly smaller spacing. In such a condition, the whole database has to be recomputed again. In the second scenario, new dataset parametric points could be placed specifically in the middle, between every other existing parametric point. In that case, the previous simulated parametric points do not need to be recomputed. However, adding a new point between existing points leads to an exponential rise in the number of points to be created. In both strategies, database refinement is very time-consuming and thus, increasing the accuracy of PODI can be a very expensive process. Many researchers across the literature have devised different methods, which are commonly classified as *sampling algorithms*. One of them that looks favourable is the *Latin*

Hypercube [179], where points are randomly scattered across the domain. However, this procedure is constrained by the enforcement of at least one point along each dimension and a balance number of points across all the regions of the database, a tricky condition to achieve. A very promising alternative to the Latin Hypercube method is the use of a sample adaptive approach known as the *Greedy Algorithm* [179]. The Greedy Algorithm has been utilised in multiple research [178, 179, 193, 194, 195] and has provided the major advantage of avoiding the *curse of dimensionality* issue [178, 179]. The method is an iterative procedure based on an error estimate. The first step consists of first locating the maximum error in the parametric domain database, followed by the addition of a dataset of a full-scale simulation solution field at that particular parametric point. These two steps are then repeated until the maximum error drops below a specified-error-threshold. Hence, the generated database will always ensure that the POMs are of adequate quality to produce accurate solutions, while the cost associated with constructing a database can be driven down.

Parameter identification using both PODI and EFG. The LVM approach considered in this research is only on one calculation method, that is, either the PODI or the EFG method. An alternative scenario that can be considered is the usage of PODI and EFG together. In this setup, a calculation run of LVM will be based on the solution produced by PODI and EFG. This calculation will start first with PODI in order to locate the solution *zone*. The solution zone can be defined by a tolerance value, ϵ_{zone}, where according to Section 7.3, $\epsilon_{\text{zone}} > \epsilon_{\text{new}}$. Once ϵ_{zone} is reached, EFG is then used instead of PODI in order to reach the solution. This hybrid setup, which will be computationally more efficient than EFG-LVM and less fast than PODI-LVM, can present some advantageous qualities. For example, the accuracy of the solution will now be highly dependent on EFG. As such, the database used for the PODI calculation does not need to be very accurate any more. The database parametric discretisation need only be at a level that will allow LVM to differentiate between local and global minima. Hence, the database generation can be less tedious. Secondly, in terms of computation efficiency, the hybrid method is expected to be better than the EFG-LVM method. The initial usage of PODI is expected to accelerate the solution zone search. The only calculation phase that will be computationally expensive, will be the convergence to the actual LVM solution using EFG. As such, ϵ_{zone} is likely to play an important role in determining how long the calculation will take. Assuming a too high ϵ_{zone} will lead to many EFG calculations.

Registration of patient-specific hearts. In this research, Magnetic Resonance Images (MRI) of patient-specific hearts were not available and as such, an idealised one was used. However, in practice, it is expected that MRIs will be available. A straightforward approach will be to create a 3D mesh geometry out of every series of MRIs and then to follow the PODI setup procedure, given in Section 5.4. However, an alternative method that could be looked at is the registration of the template mesh onto MRIs. This method does not require the creation of a geometry out of the MRIs before registering it on the template one, and will consequently reduce the time taken. In this new scenario, the template mesh is fitted to the left and right ventricle of the heart's MRIs. Since the same template mesh is used for all patient-specific hearts, the total degrees of freedom are the same and their spatial location is attached to the same feature of every heart (e.g. apex, LV septal, RV basal, etc.). Hence, the $\mathbf{U}_i$ matrix can still be assembled. Using this approach, the CPD registration procedure is no longer valid as it does not provide the ability to handle MRIs, which are binary images. To circumvent this problem, a 3D binary-mesh registration algorithm

can be used. These binary-based algorithms require a once-off conversion of the template heart mesh into binary images. These template binary images are then registered onto the patient-specific MRIs with a set of generated mapping functions. Using those mapping functions, the template mesh can then be deformed into the patient-specific ones. This method has proven to be successfully applied to complex geometries such as a carotid in [196] and also to heart geometries [197].

Part VI

Appendix

Appendix A

EFG Implementation

The element-free Galerkin method was implemented by Dr Sebastian Skatulla under the name of SESKA, by using a programming language called C++. Since C++ is a compiled low level programming language, its operations are carried out more efficiently and quickly on *Processing Units*, PU. The cardiac mathematical models introduced in Chapter 3 and calculation procedures given earlier in that chapter have been coded in such a way that they can be run on machines consisting of a single core or multi-core processors. To do so, SESKA makes use of the *Message Passing Interface* standard library, MPI. The library allows communication between processes belonging to the same program and initiated on different cores/processors/nodes/clusters for data exchange and synchronisation. Through that interface, SESKA is able to split its internal workload into smaller tasks, which can be simultaneously processed by the PUs. Hence, SESKA has the ability to solve for small problems on a desktop computer and bigger ones on a high-performance machine such as the cluster, within a smaller time frame.

In addition to the MPI library, SESKA employs other libraries, such as: *ParMetis*, *Linear Algebra Package* and *Portable, Extensible Toolkit for Scientific Computation*, to carry out process-extensive work more efficiently:

- **ParMetis**: Parmetis is a parallelised library used to partition unstructured discretised domains consisting of meshes so that matrices, built when setting the equation system, are more likely to be sparse, use less memory, and are quicker to solve. To achieve this feat, ParMetis generates a graph of the mesh and sets up a re-ordering scheme of the nodes, while scaling efficiently as the problem grows bigger.

- **Linear Algebra Package**: The Linear Algebra Package, also known as LAPACK, is a set of Fortran routines built from the Basic Linear Algebra Subprograms (BLAS), and allows for linear algebra operations. Different scalar, vector and matrix operations can be performed from the three internally defined levels. Level 1 allows for vector only manipulations such as dot products. Vector-matrix operations is contained in level 2. Finally, level 3 enables matrix-matrix handling. Apart from these, other mathematical functions are available. Some of them are LU decomposition, LQ factorisation, computing inverse matrices and extracting eigenvalues and eigenvectors. One use of LAPACK in SESKA is to invert the moment matrix during the Moving Least Approximation calculation, as described in Section 5.3.2.

- **Portable, Extensible Toolkit for Scientific Computation**: Portable, Extensible Toolkit for Scientific Computation, abbreviated to PETSc, is a library which is used for data structures and solving partial differential equations for very large problems with the help of MPI. Several useful routines are available where sparse matrices can be handled and matrices efficiently assembled. A list of preconditioners is also present in order to enhance matrices in such way that numerical instabilities are less likely to be encountered during numerical solving. Additionally, different linear, Newton-based non-linear, and time stepping ordinary differential equation solvers are provided for faster and more stable solution convergence.

Appendix B

Point-In-Polygon Problem

During the DOF standardisation procedure, an interpolation scheme is set up to interpolate the solution of one mesh to the other before and after the registration process. In order to set up the interpolation scheme, MLS requires one to find an appropriate set of supporting nodes. This set of supporting nodes needs to be locally scattered around the point at which the interpolation is carried out. In this work, the element of the interpolation point is required to be inside the element whose nodes will consequently be defined as supporting nodes. Finding out if the interpolation point is inside an element correlates to a point-in-polygon problem (PIP).

Different algorithms can be used to solve the PIP [198, 199, 200, 201]. One method that has been employed in this research is the surface orthogonal method. In this method, the surface of a polygon is first rigidly transformed to align to the x plane and is placed at the origin, as presented in Fig. B.1. Then, the rest of the polygon and the point in question are identified if they are lying in the same negative or positive regime of the y-axis. For a point to be considered inside the polygon, the rest of the polygon and the point in question must always lie in the same regime during the identification procedure of all surfaces.

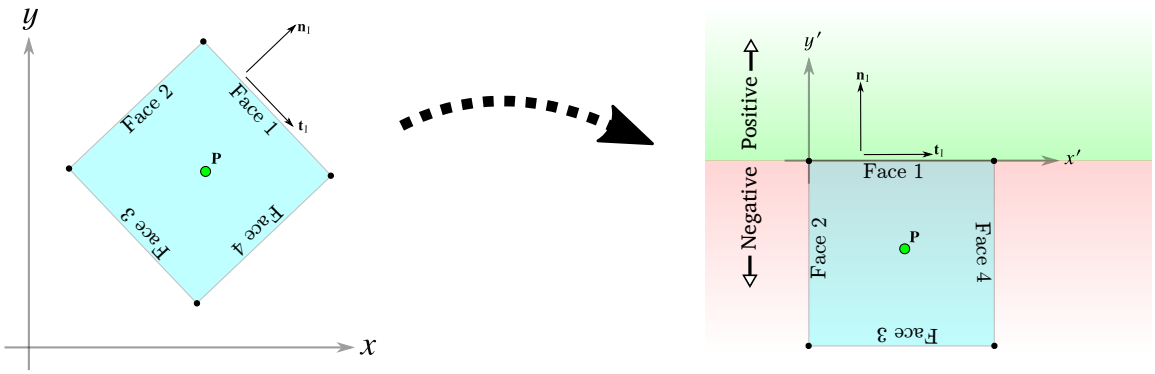


Figure B.1: Transformation of polygon to the new coordinate space..

In this research a 3D surface orthogonal algorithm was implemented. To better explain the method, let us consider a point **P**, located in a hexahedral mesh discretised domain. The number of hexahedral

elements is e_{total}. Each hexahedral element is defined by 6 surfaces where each one is defined as S_e^i with $i \in \{1, \ldots, 6\}$ and $e \in \{1, \ldots, e_{\text{total}}\}$. The number of nodes of the element is n_{total}, which is exactly 8 for a hexagonal element. To determine if **P** belongs to a specific element, e, the following steps are carried out:

- Firstly, the surface S_e^i, is isolated and its three surface orthogonal vectors are determined.

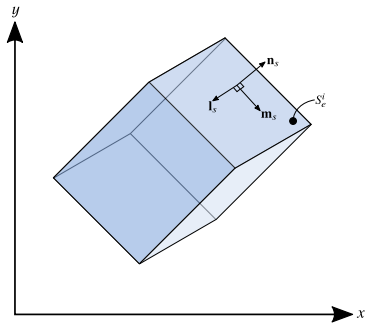


Figure B.2: Hexahedral element in the global coordinate space..

The first orthonormal vector is defined as one of the edges of the surface and is referred to as $\mathbf{l}_s$. To obtain the surface normal, another edge vector $\tilde{\mathbf{l}}_s$, different to $\mathbf{l}_s$, is selected and a cross-product operation is carried out:

$$\mathbf{n}_s = \mathbf{l}_s \otimes \tilde{\mathbf{l}}_s. \tag{B.1}$$

As such, $\mathbf{n}_s$ is perpendicular to $\mathbf{l}_s$. To obtain the third surface orthonormal vector, $\mathbf{m}_s$, the following calculation is carried out:

$$\mathbf{m}_s = \mathbf{n}_s \otimes \mathbf{l}_s. \tag{B.2}$$

The assembly of the three surface orthonormal vector forms a surface local basis matrix, **B**. In this case, $\mathbf{n}_s$ has intentionally been placed at the end so as the face of the element always points upward in the transformed coordinate space.

$$\mathbf{B} = \begin{bmatrix} \mathbf{m}_s^{\mathrm{T}} \\ \mathbf{l}_s^{\mathrm{T}} \\ \mathbf{n}_s^{\mathrm{T}} \end{bmatrix} \tag{B.3}$$

- Next, the whole element and point **P** are transformed from the global Cartesian coordinate space to the local basis one using **B**. Additionally, a point is selected from the surface S and defined as the origin by subtracting its coordinates from all nodal coordinates of the element and the point **P**.

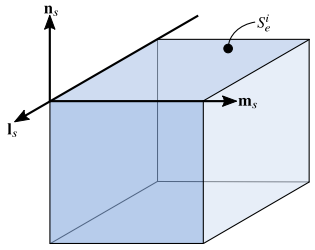


Figure B.3: Transformation of hexahedral element to the local basis coordinate space..

- A point **c**, which belongs to the element but does not lie on the surface, S, is chosen. The sign of the coordinate entry, corresponding to the index of $\mathbf{n}_s$ in **B**, of **c** is compared to the equivalent coordinate in **P**. If the signs are the same, **P** can *possibly* lie inside the element.
- For a point to be considered inside an element, the above sign comparison should hold true for all surfaces of the same element.

A summary of the above explanation is given in the form of an algorithm in Algo. 3.

Algorithm 3 Point-in-polygon algorithm

```
1:  Initialise: e_inside = −1, i = 0
2:  for e in e_total do
3:      Extract the list of surface elements
4:      Set i = 0
5:      for i in 6 do
6:          Extract nodal surface details
7:          Compute orthonormal vectors of the surface
8:          Set up local basis transformation matrix, B
9:          Transform nodes of element, e, and point of interest, P
10:         if element & point of interest lies in the same regime then
11:             i = i + 1
12:     if i = 6 then
13:         e_inside = e
14:         break
```

Part VII

Bibliography

Bibliography

[1] Hung V. Ly and Hien T. Tran. Modeling and control of physical processes using proper orthogonal decomposition. *Mathematical and Computer Modelling*, 33:223 – 236, 2001. doi:10.1016/S0895-7177(00)00240-5. <ce:title>Computation and control {VI} proceedings of the sixth Bozeman conference</ce:title>.

[2] S. Skatulla and C. Sansour. On a path-following method for non-linear solid mechanics with applications to structural and cardiac mechanics subject to arbitrary loading scenarios. *International Journal of Solids and Structures*, 96:181–191, October 2016. doi:10.1016/j.ijsolstr.2016.06.009.

[3] Leaving Certificate Biology. The circulatory system. Webpage. URL http://leavingbio.net/circulatory%20system/circulatory%20system.htm. Last access date (Time): 8 Sep 2015 (11:57).

[4] OpenStax College. *Anatomy & Physiology*. OpenStax College, August 2013.

[5] Gerald Buckberg, J.I.E. Hoffman, Aman MahaJan, Saleh Saleh, and Cecil Coghlan. Cardiac mechanics revisited: The relationship of cardiac architecture to ventricular function. *Circulation*, 118(24):2571–2587, December 2008. doi:10.1161/CIRCULATIONAHA.107.754424.

[6] Carmine D. Clemente. *Anatomy: A regional atlas of the human body*. Lippincott Williams & Wilkins, 5th edition, 2005.

[7] I.J. LeGrice, B.H. Smaill, L.Z. Chai, S.G. Edgar, J.B. Gavin, and P.J. Hunter. Laminar structure of the heart: ventricular myocyte arrangement and connective tissue architecture in the dog. *American Journal of Physiology. Heart and Circulatory Physiology*, 269(2):H571–H582, August 1995.

[8] Paul P. Lunkenheimer, Klaus Redmann, Natalie Kling, Xiaoji Jiang, Kai Rothaus, Colin W. Cryer, Frank Wübbeling, Peter Niederer, Philipp U. Heitz, Siew Yen Ho, and Robert H. Anderson. Three-dimensional architecture of the left ventricular myocardium. *The Anatomical Record Part A: Discoveries in Molecular, Cellular, and Evolutionary Biology*, 288A(6):565–578, June 2006. doi:10.1002/ar.a.20326.

[9] Damien Rohmer, Arkadiusz Sitek, and Grant T Gullberg. Reconstruction and visualization of fiber and laminar structure in the normal human heart from ex vivo diffusion tensor magnetic resonance imaging (dtmri) data. *Investigative Radiology*, 42(11):777–789, November 2007. doi:10.1097/RLI.0b013e3181238330.

[10] Achilles J. Pappano and Withrow Gil Wier. *Cardiovascular Physiology*. Mosby physiology monograph series. Elsevier Mosby, Philadelphia, PA, 10th edition, 2013. doi:10.1016/B978-3-3086-7400-1.10001-0.

[11] J.M. Guccione, G.S. Le Prell, P.P. de Tombe, and W.C. Hunter. Measurements of active myocardial tension under a wide range of physiological loading conditions. *Journal of Biomechanics*, 30(2): 189–192, February 1997. doi:10.1016/S0021-9290(96)00122-4.

[12] John E. Hall. *Guyton and Hall Textbook of Medical Physiology*. Elservier, 13th edition, 2016. ISBN 9781455770052.

[13] Gerhard Sommer, Daniel Ch. Haspinger, Michaela Andrä, Michael Sacherer, Christian Viertler, Peter Regitnig, and Gerhard A. Holzapfel. Quantification of Shear Deformations and Corresponding Stresses in the Biaxially Tested Human Myocardium. *Annals of Biomedical Engineering*, 43(10): 2334–348, October 2015. doi:10.1007/s10439-015-1281-z.

[14] Gerhard Sommer, Andreas J. Schriefl, Michaela Andrä, Michael Sacherer, Christian Viertler, Heimo Wolinski, and Gerhard A. Holzapfel. Biomechanical properties and microstructure of human ventricular myocardium. *Acta Biomaterialia*, 24:172–192, June 2015. doi:10.1016/j.actbio.2015.06.031.

[15] Wu Zhong Lin, Yao Jiang Zhang, and Er Ping Li. Proper Orthogonal Decomposition in the Generation of Reduced Order Models for Interconnects. *IEEE Transaction on Advanced Packaging*, 31(3)(3):627, August 2008. doi:10.1109/TADVP.2008.927820.

[16] Arnold M. Katz. *Physiology of the heart*. Lippincott Williams & Wilkins, Philadelphia, USA, 5th edition, November 2010.

[17] J.M. Guccione, L.K. Waldman, and A.D. McCulloch. Mechanics of Active Contraction in Cardiac Muscle: Part II-Cylindrical Models of the Systolic Left Ventricle. *Journal of Biomechanical Engineering*, 115(1):82–90, February 1993. doi:10.1115/1.2895474.

[18] J. Sandstede, C. Lipke, M. Beer, S. Hofmann, T. Pabst, W. Kenn, S. Neubauer, and D. Hahn. Age- and gender-specific differences in left and right ventricular cardiac function and mass determined by cine magnetic resonance imaging. *European Radiology*, 10(3):438–442, February 2000. doi:10.1007/s003300050072.

[19] Brian Baillargeon, Nuno Rebelo, David D. Fox, Robert L. Taylor, and Ellen Kuhl. The Living Heart Project: A robust and integrative simulator for human heart function. *European Journal of Mechanics - A/Solids*, 48:38–47, November 2014. doi:10.1016/j.euromechsol.2014.04.001.

[20] D. Legner, S. Skatulla, J. Mbewu, R.R. Rama, B.D. Reddy, C. Sansour, N.H. Davies, and T. Franz. Studying the influence of hydrogel injections into the infarcted left ventricle using the element-free Galerkin method. *International Journal for Numerical Methods in Biomedical Engineering*, 30(3): 416–429, March 2014. doi:10.1002/cnm.2610.

[21] John Dolbow and Ted Belytschko. An introduction to programming the meshless element free galerkin method. *Archives of Computational Methods in Engineering*, 5(3):207–241, September 1998. doi:10.1007/BF02897874.

[22] David B. Davidson. *Computational Electromagnetics for RF and Microwave Engineering*. Cambridge University Press, Cambridge, United Kingdom, 2nd edition, November 2010.

[23] K.D. Costa, P.J. Hunter, J.S. Wayne, L.K. Waldman, J.M. Guccione, and A.D. McCulloch. A Three-Dimensional Finite Element Method for Large Elastic Deformations of Ventricular Myocardium: II - Prolate Spheroidal Coordinates. *Journal of Biomechanical Engineering*, 118(4):464 – 472, December 1996. doi:10.1115/1.2796032.

[24] F.J. Vetter and A.D. McCulloch. Three-dimensional stress and strain in passive rabbit left ventricle: a model study. *Annals of Biomedical Engineering*, 28(7):781–792, 2000. doi:10.1114/1.1289469.

[25] Joseph C. Walker, Mark B. Ratcliffe, Peng Zhang, Arthur W. Wallace, Bahar Fata, Edward W. Hsu, David Saloner, and Julius M. Guccione. Mri-based finite-element analysis of left ventricular aneurysm. *American Journal of Physiology. Heart and Circulatory Physiology.*, 289(2):H692–H700, 2005. doi:10.1152/ajpheart.01226.2004.

[26] Steven Niederer and Nicolas Smith. The Role of the Frank-Starling Law in the Transduction of Cellular Work to Whole Organ Pump Function: A Computational Modeling Analysis. *PLoS Computational Biology*, 5(4):e1000371, April 2009. doi:10.1371/journal.pcbi.1000371.

[27] Pierre Lafortune, Ruth Arís, Mariano Vázquez, and Guillaume Houzeaux. Coupled electromechanical model of the heart: Parallel finite element formulation. *International Journal for Numerical Methods in Biomedical Engineering*, 28(1):72–86, January 2012. doi:10.1002/cnm.1494.

[28] Z.A. Taylor, M. Cheng, and S. Ourselin. High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. *IEEE transactions on medical imaging*, 27(5):650–663, 2008. doi:10.1109/TMI.2007.913112.

[29] S. Niroomandi, I. Alfaro, E. Cueto, and F. Chinesta. Real-time deformable models of non-linear tissues by model reduction techniques. *Computer Methods and Programs in Biomedicine*, 91(3): 223–231, 2008. doi:10.1016/j.cmpb.2008.04.008.

[30] S. Niroomandi, I. Alfaro, E. Cueto, and F. Chinesta. Accounting for large deformations in real-time simulations of soft tissues based on reduced-order models. *Computer Methods and Programs in Biomedicine*, 105(1):1–12, 2012. doi:10.1016/j.cmpb.2010.06.012.

[31] Stephane Cotin, Hervé Delingette, and Nicholas Ayache. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions on Visualization and Computer Graphics*, 5(1): 62–73, 1999. doi:10.1109/2945.764872.

[32] U. Meier, O. López, C. Monserrat, M. C. Juan, and M. Alcañiz. Real-time deformable models for surgery simulation: A survey. *Computer Methods and Programs in Biomedicine*, 77(3):183–197, March 2005. doi:10.1016/j.cmpb.2004.11.002.

[33] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum*, 25(4):809–836, December 2006. doi:10.1111/j.1467-8659.2006.01000.x.

[34] L.P. Nedel and D. Thalmann. Real time muscle deformations using mass-spring systems. In *Proceedings. Computer Graphics International (Cat. No.98EX149)*, pages 156–165. IEEE Comput. Soc, 1998. ISBN 0-8186-8445-3. doi:10.1109/CGI.1998.694263.

[35] Tiantian Liu, Adam W. Bargteil, James F. O'Brien, and Ladislav Kavan. Fast simulation of mass-spring systems. *ACM Transactions on Graphics*, 32(6):1–7, November 2013. doi:10.1145/2508363.2508406.

[36] Vincent Luboz, Jim Kyaw-Tun, Sayan Sen, Roger Kneebone, Robert Dickinson, Richard Kitney, and Fernando Bello. Real-time stent and balloon simulation for stenosis treatment. *Visual Computer*, 30(3):1–9, March 2013. doi:10.1007/s00371-013-0859-4.

[37] H. Delingette. Toward realistic soft-tissue modeling in medical simulation. *Proceedings of the IEEE*, 86(3):512–523, March 1998. doi:10.1109/5.662876.

[38] Jeffrey Berkley, George Turkiyyah, Daniel Berg, Mark Ganter, and Suzanne Weghorst. Real-Time Finite Element Modeling for Surgery Simulation: An Application to Virtual Suturing. *IEEE Transactions on Visualization and Computer Graphics*, 10(3):314–325, 2004. doi:10.1109/TVCG.2004.1272730.

[39] Yi Je Lim and Suvranu De. Real time simulation of nonlinear tissue response in virtual surgery using the point collocation-based method of finite spheres. *Computer Methods in Applied Mechanics and Engineering*, 196(31-32):3011–3024, June 2007. doi:10.1016/j.cma.2006.05.015.

[40] Hadrien Courtecuisse, Hoeryong Jung, Jérémie Allard, Christian Duriez, Doo Yong Lee, and Stéphane Cotin. GPU-based real-time soft tissue deformation with cutting and haptic feedback. *Progress in Biophysics and Molecular Biology*, 103(2-3):159–168, December 2010. doi:10.1016/j.pbiomolbio.2010.09.016.

[41] Zhonghua Lu, Venkata S Arikatla, Zhongqing Han, Brian F. Allen, and Suvranu De. A physics-based algorithm for real-time simulation of electrosurgery procedures in minimally invasive surgery. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 10(4):495–504, December 2014. doi:10.1002/rcs.1561.

[42] R. Everson and L. Sirovich. Karhunen-loeve procedure for gappy data. *Journal of Optical Society of America A*, 12(8):1657–1664, August 1995. doi:10.1364/JOSAA.12.001657.

[43] Nathan J. Falkiewicz and Carlos E. S. Cesnik. Proper Orthogonal Decomposition for Reduced-Order Thermal Solution in Hypersonic Aerothermoelastic Simulations. *AIAA Journal*, 49(5):994–1009, May 2011. doi:10.2514/1.J050701.

[44] David Amsallem, Julien Cortial, and Charbel Farhat. Towards real-time computational-fluid-dynamics-based aeroelastic computations using a database of reduced-order information. *AIAA Journal*, 48(9):2029–2037, September 2010. doi:10.2514/1.J050233.

[45] Erik Adler Christensen, Morten Brøns, and Jens Nørkær Sørensen. Evaluation of pod-based decomposition techniques applied to parameter-dependent non-turbulent flows. *SIAM Journal on Scientific Computing*, 21(4):1419–1434, 2000. doi:10.1137/S1064827598333181.

[46] Philippe Druault, Philippe Guibert, and Franck Alizon. Use of proper orthogonal decomposition for time interpolation from piv data: Application to the cycle-to-cycle variation analysis of in-cylinder engine flows. *Experiments in Fluids*, 39:1009–1023, 2005. doi:10.1007/s00348-005-0035-3.

[47] Ioannis T. Georgiou and Jamal Sansour. Analyzing the finite element dynamics of nonlinear in-plane rods by the method of proper orthogonal decomposition. In S. Idelsohn, E. Onate, and E. Dvorkin, editors, *Computational Mechanics, New Trends and Applications*, Barcelona, Spain, 1998. CIMNE.

[48] M. Amabili, A. Sarkar, and M.P. Païdoussis. Chaotic vibrations of circular cylindrical shells: Galerkin versus reduced-order models via the proper orthogonal decomposition method. *Journal of Sound and Vibration*, 290(3–5):736 – 762, March 2006. doi:10.1016/j.jsv.2005.04.034.

[49] Ioannis Georgiou. Advanced proper orthogonal decomposition tools: Using reduced order models to identify normal modes of vibration and slow invariant manifolds in the dynamics of planar nonlinear rods. *Nonlinear Dynamics*, 41(1-3):69–110, August 2005. doi:10.1007/s11071-005-2793-0.

[50] Riccardo Mariani and Daniele Dessi. Analysis of the global bending modes of a floating structure using the proper orthogonal decomposition. *Journal of Fluids and Structures*, 28(0):115–134, January 2012. doi:10.1016/j.jfluidstructs.2011.11.009.

[51] S. Han and B Feeny. Application of Proper Orthogonal Decomposition To Structural Vibration Analysis. *Mechanical Systems and Signal Processing*, 17(5):989–1001, September 2003. doi:10.1006/mssp.2002.1570.

[52] M.A. Trindade, C. Wolter, and R. Sampaio. Karhunen–Loève Decomposition of coupled axial/bending vibrations of beams subject to impacts. *Journal of Sound and Vibration*, 279(3–5): 1015–1036, January 2005. doi:10.1016/j.jsv.2003.11.057.

[53] P.B. Gonçalves, F.M.A. Silva, and Z.J.G.N. Del Prado. Low-dimensional models for the nonlinear vibration analysis of cylindrical shells based on a perturbation procedure and proper orthogonal decomposition. *EUROMECH Colloquium 483, Geometrically non-linear vibrations of structures*, 315(3):641 – 663, August 2008. doi:10.1016/j.jsv.2008.01.063.

[54] John C. Brigham and Wilkins Aquino. Inverse viscoelastic material characterization using pod reduced-order modeling in acoustic-structure interaction. *Computer Methods in Applied Mechanics and Engineering*, 198(9-12):893–903, February 2009. doi:10.1016/j.cma.2008.10.018.

[55] David Amsallem, Julien Cortial, Kevin Carlberg, and Charbel Farhat. A method for interpolating on manifolds structural dynamics reduced-order models. *International Journal for Numerical Methods in Engineering*, 80(9):1241–1258, 2009. doi:10.1002/nme.2681.

[56] RaJan F. Coelho, Piotr Breitkopf, Catherine Knopf-Lenoir, and Pierre Villon. Bi-level model reduction for coupled problems. Application to a 3D wing. *Structural and Multidisciplinary Optimization*, 39(4):401–418, October 2009. doi:10.1007/s00158-008-0335-3.

[57] Jin H. Ko, Kwangsub Jung, Seongseop Kim, Wonbae Kim, and Maenghyo Cho. A proper orthogonal decomposition for parametric study of the mechanical behavior of nanowires. *Journal of Mechanical Science and Technology*, 25(1):157–162, September 2011. doi:10.1007/s12206-010-1019-7.

[58] François Haddad, Sharon A. Hunt, David N. Rosenthal, and Daniel J. Murphy. Right ventricular function in cardiovascular disease, part I: Anatomy, physiology, aging, and functional assessment of the right ventricle. *Circulation*, 117(11):1436–48, March 2008. doi:10.1161/CIRCULATIONAHA.107.653576.

[59] Lucy E. Hudsmith, Steffen E. Petersen, Jane M. Francis, Matthew D. Robson, and Stefan Neubauer. Normal Human Left and Right Ventricular and Left Atrial Dimensions Using Steady State Free Precession Magnetic Resonance Imaging. *Journal of Cardiovascular Magnetic Resonance*, 7(5): 775–782, October 2009. doi:10.1080/10976640500295516.

[60] Jesper Kjaergaard, Claus Leth Petersen, Andreas Kjaer, Bente Krogsgaard Schaadt, Jae K. Oh, and Christian Hassager. Evaluation of right ventricular volume and function by 2D and 3D echocardiography compared to MRI. *European Journal of Echocardiography*, 7(6):430–438, December 2006. doi:10.1016/j.euje.2005.10.009.

[61] Roberto M. Lang, Michelle Bierig, Richard B. Devereux, Frank A. Flachskampf, Elyse Foster, Patricia A. Pellikka, Michael H. Picard, Mary J. Roman, James Seward, Jack S. Shanewise, Scott D. Solomon, Kirk T. Spencer, Martin St John Sutton, and William J. Stewart. Recommendations for Chamber Quantification: A Report from the American Society of Echocardiography's Guidelines and Standards Committee and the Chamber Quantification Writing Group, Developed in ConJunction with the European Association of Echocardiograph. *Journal of the American Society of Echocardiography*, 18(12):1440–1463, December 2005. doi:10.1016/j.echo.2005.10.005.

[62] Minako Oikawa, Yutaka Kagaya, Hiroki Otani, Masahito Sakuma, Jun Demachi, Jun Suzuki, Tohru Takahashi, Jun Nawata, Tatsuo Ido, Jun Watanabe, and Kunio Shirato. Increased [18F]fluorodeoxyglucose accumulation in right ventricular free wall in patients with pulmonary hypertension and the effect of epoprostenol. *Journal of the American College of Cardiology*, 45 (11):1849–1855, June 2005. doi:10.1016/j.jacc.2005.02.065.

[63] Paul Stolzmann, Hans Scheffel, Pedro Trigo Trindade, André R. Plass, Lars Husmann, Sebastian Leschka, Michele Genoni, Borut Marincek, Philipp A. Kaufmann, and Hatem Alkadhi. Left ventricular and left atrial dimensions and volumes: comparison between dual-source CT and echocardiography. *Investigative radiology*, 43(5):284–289, 2008. doi:10.1097/RLI.0b013e3181626853.

[64] Lisa A. Simmons, Adrian G. Gillin, and Richmond W. Jeremy. Structural and functional changes in left ventricle during normotensive and preeclamptic pregnancy. *American journal of physiology. Heart and circulatory physiology*, 283(4):H1627–H1633, 2002. doi:10.1152/ajpheart.00966.2001.

[65] S.F. Yiu, M. Enriquez-Sarano, C. Tribouilloy, J.B. Seward, and A.J. Tajik. Determinants of the degree of functional mitral regurgitation in patients with systolic left ventricular dysfunction: A quantitative clinical study. *Circulation*, 102(12):1400–1406, 2000. doi:10.1161/01.CIR.102.12.1400.

[66] Lawrence G. Rudski, Wyman W. Lai, Jonathan Afilalo, Lanqi Hua, Mark D. Handschumacher, Krishnaswamy Chandrasekaran, Scott D. Solomon, Eric K. Louie, and Nelson B. Schiller. Guidelines for the echocardiographic assessment of the right heart in adults: a report from the American Society of Echocardiography. *Journal of the American Society of Echocardiography :*

official publication of the American Society of Echocardiography, 23(7):685–713; quiz 786–788, 2010. doi:10.1016/j.echo.2010.05.010.

[67] Emiliano Iuliano and Domenico Quagliarella. Proper Orthogonal Decomposition, surrogate modelling and evolutionary optimization in aerodynamic design. *Computers & Fluids*, 84:327–350, September 2013. doi:10.1016/j.compfluid.2013.06.007.

[68] David Amsallem, Radek Tezaur, and Charbel Farhat. Real-time solution of linear computational problems using databases of parametric reduced-order models with arbitrary underlying meshes. *Journal of Computational Physics*, 326(1):373 – 397, December 2016. doi:10.1016/j.jcp.2016.08.025.

[69] David González, Elías Cueto, and Francisco Chinesta. Computational Patient Avatars for Surgery Planning. *Annals of Biomedical Engineering*, 44(1):35–45, January 2016. doi:10.1007/s10439-015-1362-z.

[70] Andriy Myronenko, Xubo Song, and Miguel Á Carreira-Perpiñán. Non-rigid point set registration: Coherent Point Drift. In B. Schölkopf and J. Platt and T. Hoffman, editor, *Advances in Neural Information Processing Systems 19*, pages 1009–1016. MIT Press, Cambridge, MA, 2007. ISBN 0162-8828. doi:10.1109/TPAMI.2010.46.

[71] Andriy Myronenko and Xubo Song. Point Set Registration: Coherent Point Drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2262–2275, December 2010. doi:10.1109/TPAMI.2010.46.

[72] Corey Winton, Jackie Pettway, C.T. Kelley, Stacy Howington, and Owen J. Eslinger. Application of Proper Orthogonal Decomposition (POD) to inverse problems in saturated groundwater flow. *Advances in Water Resources*, 34(12):1519–1526, December 2011. doi:10.1016/j.advwatres.2011.09.007.

[73] M. Boulakia, E. Schenone, and J-F. Gerbeau. Reduced-order modeling for cardiac electrophysiology. Application to parameter identification. *International Journal for Numerical Methods in Biomedical Engineering*, 28:727–744, March 2012. doi:10.1002/cnm.2465.

[74] Lik Chuan Lee, Liang Ge, Zhihong Zhang, Matthew Pease, SerJan D. Nikolic, Rakesh Mishra, Mark B. Ratcliffe, and Julius M. Guccione. Patient-specific finite element modeling of the cardiokinetix parachute device: effects on left ventricular wall stress and function. *Medical & Biological Engineering & Computing*, 52(6):557–566, May 2014. doi:10.1007/s11517-014-1159-5.

[75] Michael J. Moulton, Lawrence L. Creswell, Ricardo L. Actis, Kent W. Myers, Michael W. Vannier, Barna A. Szabo, and Michael K. Pasque. An inverse approach to determining myocardial material properties. *Journal of Biomechanics*, 28(8):935–948, August 1995. doi:10.1016/0021-9290(94)00144-S.

[76] Cristian A. Linte, Marcin Wierzbicki, Terry M. Peters, and Abbas Samani. Towards a biomechanics-based technique for assessing myocardial contractility: an inverse problem approach. *Computer Methods in Biomechanics and Biomedical Engineering*, 11(3):243–255, 2008. doi:10.1080/10255840701704553.

[77] J. Rijcken, P.H.M. Bovendeerd, A.J.G. Schoofs, D.H. Van Campent, and T. Arts. Optimization of cardiac fiber orientation for homogeneous fiber strain at beginning of ejection. *Journal of Biomechanics*, 30(10):1041–1049, 1997. doi:10.1016/S0021-9290(97)00064-X.

[78] Frederic Guyon and Rodolphe Le Riche. Least Squares parameter estimation and the Levenberg-Marquardt Algorithm: Deterministic Analysis, Sensistivities and Numerical Experiments. Technical report 041/99, INSA de Rouen, November 2000.

[79] Alexander I.J. Forrester and Andy J. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1-3):50–79, January 2009. doi:10.1016/j.paerosci.2008.11.001.

[80] Philip Holmes, John L. Lumley, Gahl Berkooz, and Clarence W. Rowley. *Turbulence, Coherent Structures, Dydynamical Systems and Symmetry*. Cambridge Monographs on Mechanics. Cambridge University Press, The Edinburg Building, Cambridge CB2 8RU, UK, 2nd edition, 2012.

[81] Rodney Rhoades and David R. Bell. *Medical Physiology: Principles for Clinical Medicine*. Wolters Kluwer Health/Lippincott Williams & Wilkins, 4th edition, 2013.

[82] Mladen J. Kocica, Antonio F. Corno, Francesc Carreras-Costa, Manel Ballester-Rodes, Mark C. Moghbel, Clotario N.C. Cueva, Vesna Lackovic, Vladimir I. Kanjuh, and Francisco Torrent-Guasp. The helical ventricular myocardial band: global, three-dimensional, functional architecture of the ventricular myocardium. *European Journal of Cardio-Thoracic Surgery*, 29(SUPPL. 1):S21–S40, April 2006. doi:10.1016/j.ejcts.2006.03.011.

[83] Socrates Dokos, Bruce H Smaill, Alistair A. Young, and Ian J. LeGrice. Shear properties of passive ventricular myocardium. *American Journal of Physiology. Heart and circulatory physiology*, 283 (6):H2650–H2659, 2002. doi:10.1152/ajpheart.00111.2002.

[84] Hervé Lombaert, Jean Marc Peyrat, Pierre Croisille, Stanislas Rapacchi, Laurent Fanton, Farida Cheriet, Patrick Clarysse, Isabelle Magnin, Hervé Delingette, and Nicholas Ayache. Human atlas of the cardiac fiber architecture: Study on a healthy population. *IEEE Transactions on Medical Imaging*, 31(7):1436–1447, 2012. doi:10.1109/TMI.2012.2192743.

[85] Satoshi Nakatani. Left ventricular rotation and twist: Why should we learn? *Journal of Cardiovascular Ultrasound*, 19(1):1, 2011. doi:10.4250/jcu.2011.19.1.1.

[86] Stephen H. Gilbert, Alan P. Benson, Pan Li, and Arun V. Holden. Regional localisation of left ventricular sheet structure: integration with current models of cardiac fibre, sheet and band structure. *European Journal of Cardio-Thoracic Surgery*, 32(2):231 – 249, August 2007. doi:10.1016/j.ejcts.2007.03.032.

[87] K Sagawa. The end-systolic pressure-volume relation of the ventricle: definition, modifications and clinical use. *Circulation*, 63(6):1223 – 1227, June 1981. doi:10.1161/01.CIR.63.6.1223.

[88] Thomas Mase, Ronald Smelser, and George Mase. *Continuum Mechanics for Engineers*. Taylor & Francis Group, 3rd edition, 2010.

[89] W. Michael Lai, David Rubin, and Erhard Krempl. *Introduction to continuum mechanics / W. Michael Lai, David Rubin, Erhard Krempl.* Elsevier, 4th edition, 2010.

[90] Gerhard A. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. John Wiley & Sons Ltd, 1st edition, 2000.

[91] Thomas J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1987.

[92] O.C. Zienkiewicz and R.L. Taylor. *The finite element method*. Butterworth-Heinemann, 2000.

[93] Taras P. Usyk, Ian J. LeGrice, and Andrew D. McCulloch. Computational model of three-dimensional cardiac electromechanics. *Computer Visualisation in Science*, 4:249–257, 2002. doi:10.1007/s00791-002-0081-9.

[94] F.C. Yin, C.C. Chan, and R.M. Judd. Compressibility of perfused passive myocardium. *American Journal of Physiology. Heart and Circulatory Physiology*, 271(5):H1864–H1870, November 1996.

[95] Linda L. Demer and Frank C.P. Yin. Passive biaxial mechanical properties of isolated canine myocardium. *Journal of Physiology*, 339:615–630, June 1983. doi:10.1113/jphysiol.1983.sp014738.

[96] G. A. Holzapfel and R. W. Ogden. On planar biaxial tests for anisotropic nonlinearly elastic solids. A continuum mechanical framework. *Mathematics and Mechanics of Solids*, 14(5):474–489, July 2009. doi:10.1177/1081286507084411.

[97] S. Göktepe, S.N.S. Acharya, J. Wong, and E. Kuhl. Computational modeling of passive myocardium. *International Journal for Numerical Methods in Biomedical Engineering*, 27(1):1–12, January 2011. doi:10.1002/cnm.1402.

[98] P.M. Nielsen, I.J. Le Grice, B.H. Smaill, and P.J. Hunter. Mathematical model of geometry and fibrous structure of the heart. *The American Journal of Physiology*, 260(4):H1365–78, April 1991.

[99] R.A. Greenbaum, S.Y. Ho, D.G. Gibson, A.E. Becker, and R.H. Anderson. Left ventricular fibre architecture in man. *Heart*, 45(3):248–263, March 1981. doi:10.1136/hrt.45.3.248.

[100] Katherine B Harrington. Direct measurement of transmural laminar architecture in the anterolateral wall of the ovine left ventricle: new implications for wall thickening mechanics. *AJP: Heart and Circulatory Physiology*, 288(3):H1324–H1330, November 2004. doi:10.1152/ajpheart.00813.2004.

[101] G. Seemann, D.U.J. Keller, D.L. Weiss, and O. Dossel. Modeling human ventricular geometry and fiber orientation based on diffusion tensor MRI. In IEEE, editor, *Computers in Cardiology*, pages 801–804, Valencia, September 2006. ISBN 978-1-4244-2532-7.

[102] Jonathan Wong and Ellen Kuhl. Generating fibre orientation maps in human heart models using Poisson interpolation. *Computer Methods in Biomechanics and Biomedical Engineering*, 17(11): 1217–1226, August 2014. doi:10.1080/10255842.2012.739167.

[103] S. Rossi, T. Lassila, R. Ruiz-Baier, A. Sequeira, and A. Quarteroni. Thermodynamically consistent orthotropic activation model capturing ventricular systolic wall thickening in cardiac electromechanics. *European Journal of Mechanics, A/Solids*, 48:129 – 142, November 2013. doi:10.1016/j.euromechsol.2013.10.009.

[104] J.M. Guccione, K.D. Costa, and A.D. McCulloch. Finite element stress analysis of left ventricular mechanics in the beating dog heart. *Journal of Biomechanics*, 28(10):1167–1177, October 1995. doi:10.1016/0021-9290(94)00174-3.

[105] J.C. Walker, M.B. Ratcliffe, P. Zhang, A.W. Wallace, E.W. Hsu, D.A. Saloner, and J.M. Guccione. Magnetic resonance imaging-based finite element stress analysis after linear repair of left ventricular aneurysm. *The Journal of Thoracic and Cardiovascular Surgery*, 135(5):1094–1102, May 2008. doi:10.1016/j.jtcvs.2007.11.038.

[106] J.F. Wenk, P. Eslami, Z. Zhang, C. Xu, E. Kuhl, J.H. Gorman III, J.D. Robb, M.B. Ratcliffe, R.C. Gorman, and J.M. Guccione. A novel method for quantifying the in-vivo mechanical effect of material injected into a myocardial infarction. *The Annals of Thoracic Surgery*, 92(3):935–941, September 2011. doi:10.1016/j.athoracsur.2011.04.089.

[107] M. Genet, L.C. Lee, R. Nguyen, H. Haraldsson, G. Acevedo-Bolton, Z. Zhang, L. Ge, K. Ordovas, S. Kozerke, and J.M. Guccione. Distribution of normal human left ventricular myofiber stress at end diastole and end systole: a target for in silico design of heart failure treatments. *Journal of Applied Physiology*, 117(2):142–152, July 2014. doi:10.1152/japplphysiol.00255.2014.

[108] Vincent Creigen, Simon Van Mourik, Vivi Rottsch, and Michel Vellekoop. Modeling a heart pump. In Rob H. Bisseling, Karma Dajani, Tammo Jan Dijkema, Johan van de Leur, and Paul A. Zegeling, editors, *Proceedings of the* 58^{th} *European Study Group Mathematics with Industry*, pages 7–26, Netherlands, January 2007. Utrecht University. URL `http://purl.tue.nl/359912123746164.pdf`.

[109] Paola Molino, Catherine Cerutti, Claude Julien, Guy Cuisinaud, Marie-Paule Gustin, and Christian Paultre. Beat-to-beat estimation of windkessel model parameters in conscious rats. *American Journal of Physiology. Heart and Circulatory Physiology*, 274(1):H171–H177, January 1998.

[110] J.T. Ottesen and M. Danielsen. Modeling ventricular contraction with heart rate changes. *Journal of Theoretical Biology*, 222(3):337–346, June 2003. doi:10.1016/S0022-5193(03)00040-7.

[111] J. Sainte-Marie, D. Chapelle, R. Cimrman, and M. Sorine. Modeling and estimation of the cardiac electromechanical activity. *Computer and structures*, 84:1743–1759, September 2006. doi:10.1016/j.compstruc.2006.05.003.

[112] R.C.P. Kerckhoffs, P.H.M. Bovendeerd, J.C.S. Kotte, F.W. Prinzen, K. Smits, and T. Arts. Homogeneity of cardiac contraction despite physiological asynchrony of depolarization: A model study. *Annals of Biomedical Engineering*, 31(5):536–547, January 2003. doi:10.1114/1.1566447.

[113] P.H.M. Bovendeerd, T. Arts, J.M. Huyghe, D.H. Van Campen, and R.S. Reneman. Dependence of local left ventricular wall mechanics on myocardial fiber orientation: A model study. *Journal of Biomechanics*, 25(10):1129–1140, October 1992. doi:10.1016/0021-9290(92)90069-D.

[114] Roy C.P. Kerckhoffs, Maxwell L. Neal, Quan Gu, James B. Bassingthwaighte, Jeff H. Omens, and Andrew D. McCulloch. Coupling of a 3D Finite Element Model of Cardiac Ventricular Mechanics to Lumped Systems Models of the Systemic and Pulmonic Circulation. *Annals of Biomedical Engineering*, 35(1):1–18, January 2007. doi:10.1007/s10439-006-9212-7.

[115] W. Voelker, H. Reul, G. Nienhaus, T. Stelzer, B. Schmitz, A. Steegers, and K.R. Karsch. Comparison of valvular resistance, stroke work loss, and Gorlin valve area for quantification of aortic stenosis. An in vitro study in a pulsatile aortic flow model. *Circulation*, 91(4):1196–1204, February 1995. doi:10.1161/01.CIR.91.4.1196.

[116] Stephanie Parragh, Bernhard Hametner, and Siegfried Wassertheurer. Influence of an asymptotic pressure level on the windkessel models of the arterial system. *IFAC-PapersOnLine*, 48(1):17–22, 2015. doi:10.1016/j.ifacol.2015.05.125.

[117] Nico Westerhof, Jan-Willem Lankhaar, and Berend E. Westerhof. The arterial Windkessel. *Medical & Biological Engineering & Computing*, 47(2):131–141, February 2009. doi:1007/s11517-008-0359-2.

[118] M. Sermesant, P. Moireau, O. CaMara, J. Sainte-Marie, R. Andriantsimiavona, R. Cimrman, D.L.G. Hill, D. Chapelle, and R. Razavi. Cardiac function estimation from mri using a heart model and data assimilation: Advances and difficulties. *Medical Image Analysis*, 10(4):642–656, 2006. doi:10.1016/j.media.2006.04.002.

[119] T. Belytschko, Y.Y. Lu, and L. Gu. Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering*, 37(2):229–256, January 1994. doi:10.1002/nme.1620370205.

[120] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981. doi:10.1090/S0025-5718-1981-0616367-1.

[121] T. Belytschko, Y. Y. Lu, L. Gu, and M. Tabbara. Element-free Galerkin methods for static and dynamic fracture. *International Journal of Solids and Structures*, 32(17-18):2547–2570, 1995. doi:10.1016/0020-7683(94)00282-2.

[122] Jeffrey H. Omens, Taras P. Usyk, Zuangjie Li, and Andrew D. McCulloch. Muscle LIM protein deficiency leads to alterations in passive ventricular mechanics. *American Journal of Physiology. Heart and circulatory physiology*, 282(2):H680–H687, 2002. doi:10.1152/ajpheart.00773.2001.

[123] R.C.P. Kerckhoffs, Sarah N. Healy, Taras P. Usyk, and Andrew D. McCulloch. Computational methods for cardiac electromechanics. *Proceedings of the IEEE*, 94(4):769–782, April 2006. doi:10.1109/JPROC.2006.871772.

[124] W. Aquino, J.C. Brigham, C.J. Earls, and N. SukuMar. Generalized finite element method using proper orthogonal Decomposition. *International Journal for Numerical Methods in Engineering*, 79(7):887–906, August 2009. doi:10.1002/nme.2604.

[125] Mathew F. Barone, Irina Kalashnikova, Matthew R. Brake, and Daniel J. Segalman. Reduced order modeling of fluid/structure interaction. Sandia report SAND2009-7189, Sandia National Laboratories, Albuquerque, New Mexico 87185 and Livermore, California 94550, September 2009. URL `http://www.sandia.gov/~ikalash/rom_ldrd_sand.pdf`.

[126] Philip Holmes, John L. Lumley, and Gahl Berkooz. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press, 1st edition, 1996.

[127] D.D. Kosambi. Statistics in function space. *J, Indian Math. Soc.*, 7:76–88, 1943. doi:10.1007/978-81-322-3676-4_15.

[128] M. Loève. Functions aleatoire de second ordre. *Comptes Rendus Acad. Sci. Paris*, -:220, 1945.

[129] K. Karhunen. Zur spektraltheorie stochastischer prozesse. *Ann. Acad. Sci. Fennicae*, A1:34, 1946.

[130] V.S. Pougachev. General theory of the correlations of random functions. *Izv. Akad. Nauk. SSSR. Math. Ser.*, 17:401–2, 1953.

[131] A.M. Obukhov. Statistical decription of continuous fields. *Trudy Geofizicheskogo Instituta, Akademiya Nauk SSSR*, 24:3–42, 1954.

[132] E.N. Lorenz. *Empirical Orthogonal functions and statistical weather prediction*. MIT Press, Cambridge, MA, 1956.

[133] M. Kirby and L. Sirovich. Application of the Karhunen-Loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):103–108, 1990. doi:10.1109/34.41390.

[134] Matthew F. Barone, Daniel J. Segalman, Heidi Thornquist, and Irina Kalashnikova. Galerkin reduced order models for compressible flow with structural interaction. *46th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA 2008-612:1–25, January 2008. doi:10.2514/6.2008-612.

[135] P.G. Cizmas, A. Palacios, T. O'Brien, and M. Syamlal. Proper-orthogonal Decomposition of spatio-temporal patterns in fluidized beds. *Chemical Engineering Science*, 58(19):4417–4427, October 2003. doi:10.1016/S0009-2509(03)00323-3.

[136] M. Hinze and S. Volkwein. *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lecture Notes in Computational Science and Engineering*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. ISBN 978-3-540-24545-2. doi:10.1007/3-540-27909-1.

[137] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America*, 4(3):519 – 524, March 1987. doi:10.1364/JOSAA.4.000519.

[138] P. Krysl, S. Lall, and J. E. Marsden. Dimensional Model Reduction in Non-linear Finite Element Dynamics of Solids and Structures. *International Journal for Numerical Methods in Engineering*, 51(4):479–504, June 2001. doi:10.1002/nme.167.

[139] Huanrong Li, Zhendong Luo, and Jing Chen. Numerical simulation based on POD for two-dimensional solute transport problems. *Applied Mathematical Modelling*, 35(5):2489–2498, 2011. doi:10.1016/j.apm.2010.11.064.

[140] John Burkardt, Qiang Du, Max Gunzburger, and Hyung-Chun Lee. Reduced order modeling of complex systems. *NA03 Dundee*, pages 29–38, 2003. URL `http://www.personal.psu.edu/qud2/Res/Pre/bdgl03dundee.pdf`.

[141] James Chen. Image compression with svd. HTML, December 2000. URL `http://fourier.eng.hmc.edu/e161/lectures/svdcompression.html`. Last access date (Time): 04 Jul 2013 (22:32).

[142] Y.C. Liang, H.P. Lee, S.P. Lim, W.Z. Lin, K.H. Lee, and C.G. Wu. Proper Orthogonal Decomposition and its Application-Part I: Theory. *Journal of Sound and Vibration*, 252(3):527–544, May 2002. doi:10.1006/jsvi.2001.4041.

[143] Emmie's LSI-SVD Module. An introduction to singular value decomposition. Website. URL http://langvillea.people.cofc.edu/DISSECTION-LAB/Emmie'sLSI-SVDModule/p4module.html. National Science Foundation-funded virtual laboratory.

[144] G. Kerschen, J-C. Golinval, A.F. Vakakis, and L.A. Bergman. The method of proper orthogonal decomposition for dynamical characterisation and order reduction of mechanical systems: An overview. *Nonlinear Dynamics*, 41(1):147–169, August 2005. doi:10.1007/s11071-005-2803-2.

[145] B.F. Feeny and R. Kappagantu. on the Physical Interpretation of Proper Orthogonal Modes in Vibrations. *Journal of Sound and Vibration*, 211(4):607–616, April 1998. doi:10.1006/jsvi.1997.1386.

[146] Jeff M. Phillips. L15: Singular value decomposition. Portable Document Format, Spring 2013. URL www.cs.utah.edu/~jeffp/teaching/cs5955/L15-SVD.pdf. Last access date (Time): 10 Jul 2013(23:18).

[147] Jonathan Shlens. A tutorial on principle component analysis. Portable Document Format, April 2014.

[148] Greg Fasshauer. Singular value decomposition. Portable Document Format. URL http://www.math.iit.edu/~fass/477577_Chapter_2.pdf. Last access date (Time): 26 Jun 2013 (11:29).

[149] Jim Lambers. The svd algorithm. Lecture 6 Notes, November 2010.

[150] S. Volkwein. Model reduction using proper orthogonal Decomposition. *Lecture Notes*, 2008. URL http://www.math.uni-konstanz.de/numerik/personen/volkwein/teaching/POD-Vorlesung.pdf.

[151] C.G. Wu, Y.C. Liang, W.Z. Lin, H.P. Lee, and S.P. Lim. A note on equivalence of proper orthogonal Decomposition methods. *Journal of Sound and Vibration*, 265(5):1103–1110, August 2003. doi:10.1016/S0022-460X(03)00032-4.

[152] S. Han and B. F. Feeny. Enhanced Proper Orthogonal Decomposition for the Modal Analysis of Homogeneous Structures. *Journal of Vibration and Control*, 8(1):19–40, January 2002. doi:10.1177/107754602023518.

[153] Troy R. Smith, Jeff Moehlis, and Philip Holmes. Low-dimensional modelling of turbulence using the proper orthogonal Decomposition: A tutorial. *Nonlinear Dynamics*, 41(1-3):275–307, August 2005. doi:10.1007/s11071-005-2823-y.

[154] Daniel Ahlman, Fredrik Söderlund, Jelliffe Jackson, Andrew Kurdila, and Wei Shyy. Proper Orthogonal Decomposition for Time-Dependent Lid-Driven Cavity Flows. *Numerical Heat Transfer, Part B: Fundamentals*, 42(4):285–306, October 2002. doi:10.1080/10407790190053950.

[155] María-Luisa Rapún and José M. Vega. Reduced order models based on local POD plus Galerkin projection. *Journal of Computational Physics*, 229(8):3046–3063, April 2010. doi:10.1016/j.jcp.2009.12.029.

[156] Dan Xie, Min Xu, Honghua Dai, and Earl H. Dowell. Proper orthogonal Decomposition method for analysis of nonlinear panel flutter with thermal effects in supersonic flow. *Journal of Sound and Vibration*, 337:263–283, February 2015. doi:10.1016/j.jsv.2014.10.038.

[157] Anindya Chatterjee. An introduction to the Proper Orthogonal Decomposition. *IEEE Photonics Technology Letters*, 21(8):C3–C3, April 2009. doi:10.1109/LPT.2009.2020494.

[158] A. Placzek, D.M. Tran, and R. Ohayon. Hybrid proper orthogonal Decomposition formulation for linear structural dynamics. *Journal of Sound and Vibration*, 318(4-5):943–964, December 2008. doi:10.1016/j.jsv.2008.05.015.

[159] Y. Tamura, S. Suganuma, H. Kikuchi, and K. Hibi. Proper Orthogonal Decomposition of Random Wind Pressure Field. *Journal of Fluids and Structures*, 13(7-8):1069–1095, October 1999. doi:10.1006/jfls.1999.0242.

[160] Jorge Cadima and Ian Jolliffe. On relationships between uncentred and column-centred principal component analysis. *Pakistan Journal of Statistics*, 25(4):473–503, October 2009. doi:10.1.1.323.9742.

[161] Paul Honeine. An eigenanalysis of data centering in machine learning. *arXiv.org*, pages 1–14, July 2014.

[162] Paul G.A. Cizmas, Brian R. Richardson, Thomas A. Brenner, Thomas J. O'Brien, and Ronald W. Breault. Acceleration techniques for reduced-order models based on proper orthogonal Decomposition. *Journal of Computational Physics*, 227(16):7791–7812, August 2008. doi:10.1016/j.jcp.2008.04.036.

[163] Philip J. Holmes, John L. Lumley, Gahl Berkooz, Jonathan C. Mattingly, and Ralf W. Wittenberg. Low-dimensional models of coherent structures in turbulence. *Physics Reports*, 287(4)(4):337–384, August 1997. doi:10.1016/S0370-1573(97)00017-3.

[164] Jernej Barbic and Doug James. Real-Time Subspace Integration for St.Venant-Kirchhoff Deformable Models. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2005*, 24(3): 982–990, July 2005. doi:10.1145/1073204.1073300.

[165] Bui-Thanh Tan, Karen Willcox, and Murali Damodaran. Applications of proper orthogonal decomposition for inviscid transonic aerodynamics. Technical report, Singapore-MIT Alliance, 2003.

[166] D. Alonso, A. Velazquez, and J.M. Vega. A method to generate computationally efficient reduced order models. *Computational Methods in Applied Mechanical Engineering*, 198:2683–2691, July 2009. doi:10.1016/j.cma.2009.03.012.

[167] H. Akima. A method of bivariate interpolation and smooth surface fitting for irregular distributed data points. *ACM Trans. Math. Softw.*, 4:148–164, June 1978. doi:10.1145/355780.355786.

[168] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. HamMarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 3rd edition, 1999. doi:10.1137/1.9780898719604.

[169] Nicolas Toussaint, Christian T. Stoeck, Tobias Schaeffter, Sebastian Kozerke, Maxime Sermesant, and Philip G. Batchelor. In vivo human cardiac fibre architecture estimation using shape-based diffusion tensor processing. *Medical Image Analysis*, 17(8):1243–1255, December 2013. doi:10.1016/j.media.2013.02.008.

[170] Pablo Lamata, Matthew Sinclair, Eric Kerfoot, Angela Lee, Andrew Crozier, BoJan Blazevic, Sander Land, Adam J Lewandowski, David Barber, Steve Niederer, and Nic Smith. An automatic service for the personalization of ventricular cardiac meshes. *Journal of the Royal Society Interface*, 11(91):20131023, December 2014. doi:10.1098/rsif.2013.1023.

[171] Alan L. Yuille and Norberto M. Grzywacz. A mathematical analysis of the motion coherence theory. *International Journal of Computer Vision*, 3(2):155–175, June 1989. doi:10.1007/BF00126430.

[172] CloudCompare. Cloudcompare (version 2.6), 2016. URL `http://www.cloudcompare.org/`.

[173] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993. doi:10.1109/34.232073.

[174] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit*. Kitware Inc., 4th edition, 2006. ISBN 978-1-930934-19-1.

[175] Wolfgang A. Goetz, Emmanuel Lansac, Hou-Sen Lim, Patricia A. Weber, and Carlos M.G. Duran. Left ventricular endocardial longitudinal and transverse changes during isovolumic contraction and relaxation: a challenge. *American Journal of Physiology. Heart and circulatory physiology*, 289(1):H196–H201, July 2005. doi:10.1152/ajpheart.00867.2004.

[176] Clifford R. Greyson. Pathophysiology of right ventricular failure. *Critical Care Medicine*, 36(1 Suppl):S57–65, January 2008. doi:10.1097/01.CCM.0000296265.52518.70.

[177] Vincent B. Ho and Gautham P. Reddy. *Cardiovascular Imaging*. Elsevier-Health Sciences Division, Philadelphia, US, 1st edition, November 2010.

[178] A. Paul-Dubois-Taine and D. Amsallem. An adaptive and efficient greedy procedure for the optimal training of parametric reduced-order models. *International Journal for Numerical Methods in Engineering*, 102(5):1262–1292, May 2015. doi:10.1002/nme.4759.

[179] Y. Choi, D. Amsallem, and C. Farhat. Gradient-based Constrained Optimization Using a Database of Linear Reduced-Order Models. *arXiv preprint arXiv:1506.07849*, pages 1–28, 2015.

[180] Zhendong Luo, Hong Li, Yanjie Zhou, and Zhenghui Xie. A reduced finite element formulation based on POD method for two-dimensional solute transport problems. *Journal of Mathematical Analysis and Applications*, 385(1):371–383, January 2012. doi:10.1016/j.jmaa.2011.06.051.

[181] G. Rozza, D.B.P. Huynh, and A.T. Patera. Reduced Basis Approximation and a Posteriori Error Estimation for Affinely Parametrized Elliptic Coercive Partial Differential Equations. *Archives of Computational Methods in Engineering*, 15(3):229–275, September 2008. doi:10.1007/s11831-008-9019-9.

[182] Jorn F. M. van Doren, Renato MarkoviNović, and Jan-Dirk Jansen. Reduced-order optimal control of water flooding using proper orthogonal Decomposition. *Computational Geosciences*, 10(1): 137–158, March 2006. doi:10.1007/s10596-005-9014-2.

[183] Dao. My-Ha, K. M. Lim, B. C. Khoo, and Karen E. Willcox. Real-time optimization using proper orthogonal Decomposition: Free surface shape prediction due to underwater bubble dynamics. *Computers & Fluids*, 36(3):499–512, 2007. doi:10.1016/j.compfluid.2006.01.016.

[184] Mark K. Transtrum and James P. Sethna. Improvements to the Levenberg-Marquardt algorithm for nonlinear least-squares minimization. *ARXIV*, (2):32, January 2012.

[185] Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, June 1963. doi:10.1137/0111030.

[186] A. Croeze, L. Pittman, and W. Reynolds. Solving Nonlinear Least-Squares Problems With the Gauss-Newton and Levenberg-Marquardt Methods. Portable Document format. URL `https://www.math.lsu.edu/system/files/MunozGroup1-Paper.pdf`.

[187] Henri P. Gavin. The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems. Technical report, Department of Civil and Environmental Engineering, Duke University, 2013.

[188] H. Gercek. Poisson's ratio values for rocks. *International Journal of Rock Mechanics and Mining Sciences*, 44(1):1–13, January 2007. doi:10.1016/j.ijrmms.2006.04.011.

[189] You-Lin Xu and Yong Xia. *Structural Health Monitoring of Long-Span Suspension Bridges*. Taylor & Francis Group (CRC Press), 1st edition, 2012.

[190] Nasibeh Asa Golsorkhi and Hojat Ahsani Tehrani. Levenberg-Marquardt Method for Solving the Inverse Heat Transfer Problems. *Journal of Mathematics and computer science*, 13:300–310, October 2014.

[191] Martin Stiftinger. Calculation of the Jacobian matrix. Webpage, August 1995.

[192] Arnab Palit, Sunil K. Bhudia, Theodoros N. Arvanitis, Glen A. Turley, and Mark A. Williams. Computational modelling of left-ventricular diastolic mechanics: Effect of fibre orientation and right-ventricle topology. *Journal of Biomechanics*, 48(4):604–612, February 2015. doi:10.1016/j.jbiomech.2014.12.054.

[193] L. Iapichino and S. Volkwein. Optimization strategy for parameter sampling in the reduced basis method. *IFAC-PapersOnLine*, 48(1):707–712, 2015. doi:10.1016/j.ifacol.2015.05.020.

[194] Markus A. Dihlmann and Bernard Haasdonk. Certified PDE-constrained parameter optimization using reduced basis surrogate models for evolution problems. *Computational Optimization and Applications*, 60(3):753–787, April 2015. doi:10.1007/s10589-014-9697-1.

[195] Maciej Balajewicz, David Amsallem, and Charbel Farhat. Projection-based model reduction for contact problems. *Proceedings of the 2011 American Control Conference*, pages 1885–1891, March 2015. doi:10.1002/nme.

[196] D.C. Barber, E. Oubel, A.F. Frangi, and D.R. Hose. Efficient computational fluid dynamics mesh generation by image registration. *Medical Image Analysis*, 11(6):648–662, December 2007. doi:10.1016/j.media.2007.06.011.

[197] Pablo Lamata, Steven Niederer, David Nordsletten, David C. Barber, Ishani Roy, D. Rod Hose, and Nic Smith. An accurate, fast and robust method to generate patient-specific cubic Hermite meshes. *Medical Image Analysis*, 15(6):801–813, December 2011. doi:10.1016/j.media.2011.06.010.

[198] Kai Hormann and Alexander Agathos. The point in polygon problem for arbitrary polygons. *Computational Geometry: Theory and Applications*, 20(3):131–144, November 2001. doi:10.1016/S0925-7721(01)00012-8.

[199] Min Chen and Teeter Townsend. Efficient and Consistent Algorithms for Determining the Containment of Points in Polygons and Polyhedra. In *Eurographics 87*, pages 423–437, G. Marechal, 1987. Elsevier Science Publishers B.V. (North-Holland). doi:10.1.1.173.1916.

[200] David G. Alciatore and Rick Miranda. A winding number and point-in-polygon algorithm, 1995.

[201] Paul S. Heckbert. *Graphics Gems IV*. Academic Press, Boston MA, 1994.