Towards a fast and stable dynamic skeletal muscle model

Von der Fakultät Bau- und Umweltingenieurwissenschaften und
dem Stuttgart Research Centre for Simulation Technology
der Universität Stuttgart zur Erlangung der Würde eines
Doktor-Ingenieurs (Dr.-Ing.) genehmigte Abhandlung

Vorgelegt von

Mylena Mordhorst

aus
Eckernförde

Hauptberichter:	Prof. Oliver Röhrle, PhD
Mitberichter:	Prof. Dr. Bernard Haasdonk
	Prof. Dr.-Ing. Christian J. Cyron

Tag der mündlichen Prüfung: 16.04.2020

Institut für Modellierung und Simulation Biomechanischer Systeme
der Universität Stuttgart
2020

Report No.: CBM-05 (2020)
Institute for Modelling and Simulation of Biomechanical Systems
Chair of Continuum Biomechanics and Mechanobiology
University of Stuttgart, Germany, 2020

Editor:
Prof. O. Röhrle, PhD

© Mylena Mordhorst
Institute for Modelling and Simulation of Biomechanical Systems
Chair of Continuum Biomechanics and Mechanobiology
University of Stuttgart
Pfaffenwaldring 5a
70569 Stuttgart, Germany

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without the permission in writing of the author.

ISBN 978-3-946412-04-5
(D 93 – Dissertation der Universität Stuttgart)

Deutschsprachige Zusammenfassung

Die Modellierung und Simulation des gesamten Bewegungsapparats wird heutzutage gerne dazu genutzt, experimentelle Studien zu unterstützen, oder zu ersetzen. Insbesondere auch dort, wo diese an ihre Grenzen stoßen, kommen Simulationen zum Einsatz. Beispiele hierfür wären simulierte Crashtests in der Automobilindustrie, in denen der Fokus auf dem Verhalten und Schutz der Insassen liegt, oder im medizinischen Bereich die Verbesserung des Tragekomforts von Prothesen durch Simulationen des Übergangs von Muskelstumpf und Prothesenschaft und den dort auftretenden mechanischen Kräften. Als ein wesentlicher Bestandteil des Bewegungsapparats, ist insbesondere die Modellierung der Skelettmuskulatur von großer Bedeutung. Aufgrund ihres komplexen strukturellen und funktionellen Aufbaus ist dies allerdings eine sowohl herausfordernde, als auch sehr rechenintensive Aufgabe. Die vorliegende Dissertation leistet einen wertvollen Beitrag für die Entwicklung eines numerisch stabilen und schnellen Skelettmuskelmodells, welches in den oben genannten Beispielen seine Anwendung finden könnte. Dazu wurde ein dreidimensionales, dynamisches, inkompressibles und nichtlineares Skelettmuskelmodell erstellt, mithilfe der Finite-Elemente-Methode im Raum und dem impliziten Euler Verfahren in der Zeit diskretisiert und die Struktur des resultierenden Differentialgleichungssystems sorgfältig untersucht. Anschließend wurden die in der Modellreduktion gängigen Methoden der Approximation durch reduzierte Basen (RB) und die Singulärwertzerlegung oder auch Hauptachsentransformation für ihre Anwendung auf dieses Skelettmuskelmodell angepasst und optimiert. Auf diese Weise konnte ein stabiles reduziertes Skelettmuskelmodell erzeugt werden, welches durch ein Differentialgleichungssystem mit lediglich 15% der Größe des ursprünglichen Systems beschrieben werden kann. Folgende Erkenntnisse wurden als hierfür wesentlich ausgearbeitet: (i) Um die Struktur des ursprünglichen Skelettmuskelmodells im reduzierten Modell zu erhalten, muss als Unterraum für die Geschwindigkeits-Freiheitsgrade derselbe gewählt werden wie für die Verschiebungs-Freiheitsgrade. (ii) Die Größe des Unterraums für die Geschwindigkeiten muss kleiner oder gleich groß wie die des Unterraums für die Verschiebungen sein. (iii) Für die Berechnung der reduzierten Basen mittels der Singulärwertzerlegung sollte als Norm jene gewählt werden, die zum jeweiligen Lösungsraum gehört. (iv) Die Stabilität des reduzierten Skelettmuskelmodells wird hauptsachlich durch die Wahl der Größe des Unterraums für den Druck im Verhältnis zur Größe des Unterraums für die Verschiebungen beeinflusst. (v) Eine Erweiterung des Unterraums für die Verschiebungen durch sogenannte Supremizer wirkt sich vorteilhaft auf die Stabilität des reduzierten Systems aus.

Abstract

Computational models and simulations are widely used to substitute or support all sorts of experiments. Especially in the field of biomechanics, they represent a great opportunity to provide further insight into structural and functional properties of biological tissue. Modelling the musculoskeletal system in particular is a challenging and computationally expensive task. Therefore, this contribution investigates the possibility to reduce the computational effort of a dynamic skeletal muscle model making use of model order reduction (MOR) methods. For that purpose, a three-dimensional, nonlinear, dynamic skeletal muscle model based on the theory of incompressible finite hyperelasticity is introduced. After discretisation in space and time, using the mixed Taylor-Hood finite elements and the implicit Euler scheme, respectively, the obtained complex and high-dimensional differential algebraic equation system describing the three fields position, velocity and pressure, is investigated from a theoretical as well as computational point of view. Furthermore, the stability issues, encountered with a reduced-order model, built by projecting each field of the high-dimensional model onto a reduced subspace, are demonstrated. The reason for these problems is additionally investigated and confirmed from the theoretical perspective. In order to propose a suitable approach for obtaining a stable reduced-order skeletal muscle model, the well-established technique of combining the reduced basis (RB) approximation with the proper orthogonal decomposition (POD) needs to be customised. Therefore, the performance with respect to stability, efficiency and accuracy of different reduced-order models (ROM), built from various combinations and sizes of subspaces, each of them again constructed from differently calculated POD bases, is compared. The key findings and resulting recommendations for the construction of a stable reduced-order skeletal muscle model are (i) The velocity POD basis has to be chosen equal to the position POD basis, i.e. the same subspace for both fields is required in order to preserve the structure of the original high-dimensional model. (ii) For the reduced sizes of the velocity and the position space, $r_v \leq r_u$ has to hold. (iii) The POD bases should be computed to be optimal in the inner product norm. (iv) The stability of the ROM mainly depends on the ratio between the size of the reduced position space, r_u, and the reduced pressure space, r_p. (v) Enriching the position (and velocity) POD basis with approximate supremizer solutions is beneficial to gain stability. This way, a significant dimensional reduction of the governing system of differential algebraic equations to around 15 % of the original system size can be achieved, while preserving the original structure and stability. Furthermore, considering that exclusively using the RB approximation, the evaluation of nonlinear components still requires operations depending on the original high dimension, an acceptable speedup of nearly 2 is obtained. With these results, this work makes a valuable contribution towards the aim of providing a stable and fast dynamic skeletal muscle model. It represents a step forward in making for example many-query applications and real time simulations feasible.

Contents

1 Introduction

Modelling the musculoskeletal system is by now a common approach to enhance or support experimental studies. Especially in cases, where experiments face limitations, e.g. from an ethical point of view or due to time and monetary constraints, modelling and simulation can be utilised instead or supplementary in order to increase the understanding of the considered problem. As a prominent application, car crash simulations, particularly also in the context of autonomous driving, that aim to investigate the passenger behaviour and develop strategies for passenger protection, can be named. Also in the process of prosthesis design, skeletal muscle models are employed in order to e.g. improve the fitting of the stump into the socket. Even simulating a scenario, where the stump controls a more sophisticated prosthesis, is a possible application from the medical field. Lastly, modelling the neuromuscular system by coupling neurophysiology and skeletal muscle mechanics with the aim of improving the understanding of the complex involved processes and their interactions, can be named as a motivational example.
For the given possible applications, continuum-mechanical models based on the theory of finite hyperelasticity that have the advantage of adequately representing the complex muscular structure and that are capable of not only predicting the mechanical behaviour under external conditions, but also computing muscle and contact forces, are required. However, these multiscale models mostly require very fine finite element discretisations and thus are computationally expensive. Especially in the many-query context, which for example can arise when patient-specific data is needed, or if parameter studies for healthy and pathological conditions shall be conducted, the simulations become prohibitively expensive or even unfeasible. For this reason, one needs to find ways to speed up the simulations and this is where the mathematical field of model order reduction (MOR) suggests itself.

One possibility of reducing the computational effort of simulating a high-dimensional system of differential equations, in the context of MOR referred to as full-order model (FOM), is projection-based MOR. Essentially, this approach aims at reducing the number of degrees of freedom (dof) of the system to solve by projecting it onto a lower dimensional subspace of the original high-dimensional solution space. That way, a reduced-order model (ROM) of lower dimension is obtained. For the determination of a suitable low-dimensional subspace, the proper orthogonal decomposition (POD) is a well-established and widely applied method. The idea of the POD is based on a split of necessary computations into an offline and an online phase. Therein, it is assumed that during the offline phase time and resources are unlimited, i.e. computations involving high-dimensional operations can be executed, while aiming at computing only operations of low-dimensional complexity during the online phase. Since the POD yields the best approximation on given training data and has the advantage of being very flexible in its application to different models, it was considered a suitable method to obtain a reduced skeletal muscle model. In cases, where the system additionally contains nonlinearities, an exclusive use

of the POD is not sufficient to obtain a computational speedup, since the evaluation of reduced nonlinear terms still depends on the original high dimension. For those cases, so-called hyperreduction methods need to be considered.

Assuming that the FOM is well defined, MOR generally seeks to preserve its structure in the ROM. For that reason, this contribution tries to thoroughly set up and investigate the full-order skeletal muscle model before proceeding further to the task of building a (stable and fast) reduced-order skeletal muscle model. Properties of the FOM, which are focused on are (i) dynamics, (ii) incompressibility and (iii) nonlinearity. The reason being that they determine the structure of the differential equation system to solve, while offering the possibility to enhance or exchange contributions like material models. Especially (i) the dynamics and (ii) the incompressibility constraint, determine the overall structure, yielding a differential algebraic equation system (DAE) with three fields, namely the position, the velocity and the pressure, to be solved for. Additionally, the nonlinearity (iii) requires to perform a linearisation at some point of the solution procedure, which, together with the other two main properties, leads to the task of solving a saddle point problem.
The challenge of constructing a stable and fast skeletal muscle model thus lies in considering and preserving these structural properties of the FOM when trying to set up a suitable method to obtain the ROM. The discretised incompressible Navier-Stokes equations, describing an incompressible fluid flow, yield a similar structure as the incompressible skeletal muscle model. Since they seem to be far more analysed and understood from a mathematical perspective, their numerical treatment provides guidance for this work, particularly in choosing suitable projection spaces for the three different fields. The suggestion of enriching projection spaces with so-called supremizer solutions offered by Rozza & Veroy [41] and enhanced later by Ballarin et al. [4] is investigated additionally to the POD in this contribution.

The intention of this dissertation is to provide the necessary details in a comprehensive and comprehensible form, such that it can serve as a starting point for future investigations. To that purpose, Chapter 2 first introduces the fundamental continuum-mechanical equations to describe a general incompressible solid in three dimensions in a dynamic setting. Subsequently, this concept is extended to the specific case of the skeletal muscle model in Chapter 3. Therein, for two reasons great care is taken to derive the discretised system of equations and describe the necessary steps and difficulties of the solution procedure. Firstly, this chapter shall offer guidance to future users or developers of the model and the code, by providing the necessary details on the implementation. Secondly, as already explained above, revealing the structure of the full model will be beneficial for the anticipated application of projection-based MOR. The approach from general to specific is adopted for the MOR part of this work as well. Chapter 4 describes the projection of a general first-order differential equation system onto a subspace and the calculation of a POD basis in general, before these methods are extended to the skeletal muscle model in Chapter 5. Equipped with the necessary theoretical details, the next two chapters show the results using the proposed methods. Therefore, Chapter 6 introduces three different examples with increasing complexity, which will be used in the second part of this chapter to investigate and point out observed issues of the FOM, before they serve for the analysis of different ROM in Chapter 7. Finally, Chapter 8 concludes with a discussion of the overall results and, based on that, suggestions for future work.

2 3D nonlinear incompressible solid dynamics

This chapter introduces the fundamentals for modelling an incompressible solid in a continuum-mechanical framework and a dynamic setting with a nonlinear material behaviour. Commonly, skeletal muscle tissue is considered a (nearly) incompressible material (see e.g. [18] for a recent review), where incompressibility in this sense means that it does not undergo any volumetric deformation. Furthermore, as most biological tissues, skeletal muscle tissue exhibits a highly nonlinear material behaviour. Without going into specifics of skeletal muscle tissue, here, the basic concepts and equations to describe a general nonlinear, incompressible solid are introduced. For this purpose, the chapter is divided into four sections. Section 2.1 provides the fundamental tools to describe a continuum-mechanical solid. Then, this is extended to the case of incompressible solid dynamics in Section 2.2, which means, that an additional constraint equation is introduced. These sections are mainly based on introductory textbook knowledge, where the author can particularly recommend the books of [10, 29]. Subsequently, a short introduction to constitutive modelling is given in Section 2.3. As skeletal muscle tissue is (simply speaking) composed of single muscle fibres, this is done on the basis of a transversely isotropic hyperelastic material, which can account for the preferred direction. This section additionally suggests some specific strain energy functions suitable to describe such materials. As we are dealing with nonlinear material behaviour, Section 2.4 concludes this chapter by explaining the common procedures of linearising nonlinear constitutive laws.

2.1 Fundamentals of continuum mechanics

This section introduces kinematic relations, stress concepts and balance relations. It is by no means exhaustive as this is not the focus of this work. However, it includes the notations, equations and concepts that will be used herein. For a more detailed introduction, e.g. into tensor notation and calculus, the reader is referred to e.g. [10, 29, 36].

Motion, deformation and strain measure

For a body $\Omega_0 \subset \mathbb{R}^3$ at a time $t = t_0$, each material particle can be described by a coordinate vector $\boldsymbol{X} \in \Omega_0$. Assuming that the body undergoes a deformation, the referential position $\boldsymbol{X}$ of each particle changes into an actual position at time t, which can be described by a coordinate vector $\boldsymbol{x} \in \Omega$, with $\Omega \subset \mathbb{R}^3$ being the body in the actual configuration at time t. Mathematically, this deformation can be described by a so-called

motion function

$$\boldsymbol{\chi} : \Omega_0 \mapsto \Omega, \quad \boldsymbol{X} \longrightarrow \boldsymbol{\chi}(\boldsymbol{X}, t) =: \boldsymbol{x}(\boldsymbol{X}, t)\,, \tag{2.1}$$

which maps the referential position $\boldsymbol{X} \in \Omega_0$ to the actual position $\boldsymbol{x} \in \Omega$.
To describe the relative spatial position of two neighbouring particles in the actual configuration, one defines the deformation gradient $\boldsymbol{F} = \boldsymbol{F}(\boldsymbol{x}(\boldsymbol{X}, t)) \in \mathbb{R}^{3\times3}$ and additionally the Jacobian $J \in \mathbb{R}$ to relate the current to the referential volume,

$$\boldsymbol{F} := \frac{\partial \boldsymbol{\chi}(\boldsymbol{X}, t)}{\partial \boldsymbol{X}} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}, \quad J := \det(\boldsymbol{F})\,. \tag{2.2}$$

As strain measure, which describes the relative position of neighbouring particles inside the body, the right Cauchy-Green deformation tensor $\boldsymbol{C} \in \mathbb{R}^{3\times3}$ is commonly used. It is defined as

$$\boldsymbol{C} := \boldsymbol{F}^T\boldsymbol{F}, \text{ with } \det(\boldsymbol{C}) = J^2\,. \tag{2.3}$$

In general, within this work, we apply the Lagrange framework (also referred to as material description) and describe physical quantities with respect to the referential coordinates, i.e. in terms of $\boldsymbol{x} = \boldsymbol{\chi}(\boldsymbol{X}, t)$, such that the material time derivative and the partial time derivative are equivalent. Therefore, with $\nabla(\cdot) := \frac{\partial(\cdot)}{\partial \boldsymbol{X}}$, we denote the gradient with respect to the state in the reference configuration. The same holds for the divergence operator $\nabla \cdot (\cdot)$. Furthermore, concerning the tensor calculus notation employed in this work, a double contraction is written by a single dot between the two tensors (see e.g. Equation (2.9)), while a normal multiplication uses no symbol between the two quantities (see e.g. Equation (2.3)).

Stress tensors

The internal forces, which neighbouring particles of the body exert on each other, are described by stresses or stress tensors. The Cauchy stress tensor $\boldsymbol{T}$, also referred to as the true stress, is defined through the Cauchy theorem $\boldsymbol{t} = \boldsymbol{T}\boldsymbol{n}$, where $\boldsymbol{t}, \boldsymbol{n} \in \mathbb{R}^3$ are the traction vector and the surface outward unit normal vector respectively. Relating the actual surface force element to the actual area element, the symmetric Cauchy stress tensor is evidently a quantity of the actual configuration. In order to enable a Lagrange formulation, two additional stress tensors are introduced. First, the 1st Piola-Kirchhoff stress tensor, $\boldsymbol{P} = \det(\boldsymbol{F})\boldsymbol{T}\boldsymbol{F}^{-T}$, which relates the the actual surface force element to the referential area element. This tensor is a so-called two-field tensor, which has one basis in the actual and one basis in the referential configuration and is (in general) not symmetric. And second, the 2nd Piola-Kirchhoff stress tensor, $\boldsymbol{S} = \det(\boldsymbol{F})\boldsymbol{F}^{-1}\boldsymbol{T}\boldsymbol{F}^{-T}$, which lives completely on the reference configuration and is symmetric. From these definitions, obviously, the relation

$$\boldsymbol{P} = \boldsymbol{F}\boldsymbol{S} \quad \Longleftrightarrow \quad \boldsymbol{S} = \boldsymbol{F}^{-1}\boldsymbol{P} \tag{2.4}$$

between the 1st and the 2nd Piola-Kirchhoff stress tensors exist. This is employed in the constitutive modelling process later in Section 2.3. There, additionally, use is made

of the concept of conjugate variables (strain and stress measures), where in the course of this work, the two work conjugate pairs $\left\{\boldsymbol{P}, \dot{\boldsymbol{F}}\right\}$ and $\left\{\boldsymbol{S}, \frac{1}{2}\dot{\boldsymbol{C}}\right\}$ are used.

Balance of linear momentum

The derivation of the balance principles in general, is done via axiomatically introducing relations, that hold globally for the entire body. Subsequently, the equations that apply for each material point, i.e. the local relations, are derived. The balance of linear momentum, also called NEWTON's 1st law or CAUCHY's 1st equation of motion (c.f. e.g. [29]) characterises the resultant force. It is derived from the relation between the total linear momentum $\boldsymbol{L}(t)$ and the resultant force $\boldsymbol{f}(t)$

$$\dot{\boldsymbol{L}}(t) := \frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega_0} \rho_0(\boldsymbol{X})\frac{\partial \boldsymbol{x}(\boldsymbol{X},t)}{\partial t}\,\mathrm{d}V \overset{!}{=} \boldsymbol{f}(t)\,, \tag{2.5}$$

with $\rho_0(\boldsymbol{X}) \in \mathbb{R}$ being the density in reference configuration. In its local referential form it is given as

$$\rho_0(\boldsymbol{X})\frac{\partial^2 \boldsymbol{x}(\boldsymbol{X},t)}{\partial t^2} = \nabla \cdot \boldsymbol{P}(\boldsymbol{F}(\boldsymbol{X},t),t) + \boldsymbol{b}(\boldsymbol{X},t) \quad \forall \boldsymbol{X} \in \Omega_0 \subset \mathbb{R}^3\,, \tag{2.6}$$

where $\boldsymbol{b}(\boldsymbol{X},t)$ denotes the body force. This is the equation that describes the minimisation of mechanical energy and that needs to be solved to describe the dynamics of a solid in a continuum-mechanical framework. The existence of a solution, i.e. a stationary deformation state, can be guaranteed by choosing a polyconvex strain energy function (whose derivative is inserted into $\boldsymbol{P}$, c.f. Section 2.3) together with well-defined, compatible boundary conditions. Details on the existence and uniqueness of local and global minima of Equation (2.6) are a field on its own and lie outside the scope of of this work. The interested reader is referred to e.g. Ball [3] for a start.

2.2 Incompressible solid dynamics

Materials that do not undergo volumetric changes are referred to as incompressible materials. As the determinant of the deformation gradient relates the actual volume to the referential volume, these materials are characterised by $\det(\boldsymbol{F}) = 1$. From a mathematical point of view, one has to solve a differential equation, that is Equation (2.6), subject to an additional (incompressibility) constraint equation to describe the dynamics of an incompressible solid. The extended problem thus can be formulated as:

$$\begin{aligned}
&\text{Find } \boldsymbol{x}(\boldsymbol{X},t) \in \Omega \subset \mathbb{R}^3, \text{ such that } \forall\, \boldsymbol{X} \in \Omega_0 \subset \mathbb{R}^3 \\
&\rho_0(\boldsymbol{X})\frac{\partial^2 \boldsymbol{x}(\boldsymbol{X},t)}{\partial t^2} = \nabla \cdot \boldsymbol{P}(\boldsymbol{F}(\boldsymbol{X},t),t) + \boldsymbol{b}(\boldsymbol{X},t) \\
&\quad \text{subject to} \quad 0 = \det \boldsymbol{F}(\boldsymbol{x}(\boldsymbol{X},t),t) - 1\,.
\end{aligned} \tag{2.7}$$

Again, the existence of a solution to this constraint problem, which is solved by the method of LAGRANGE multipliers, can be assured by an appropriate choice of the con-

stitutive equation and the boundary conditions (seee.g. [3]). As nicely explained in [10], it is a common procedure to separate the volumetric from the isochoric (or distortional), i.e. the volume preserving components of the deformation, when dealing with constitutive modelling of incompressible materials. This can be realised by a multiplicative split of the deformation gradient and the strain measures and an additive split of the stress tensors.

Volumetric isochoric split of the deformation gradient

The deformation gradient $\boldsymbol{F}$ can be expressed in terms of a volumetric and a volume preserving (isochoric) part as follows:

$$\boldsymbol{F} = \underbrace{(J^{\frac{1}{3}}\boldsymbol{I})}_{\text{volumetric}} \underbrace{\bar{\boldsymbol{F}}}_{\text{isochoric}}, \quad \text{with } \bar{\boldsymbol{F}} := J^{-\frac{1}{3}}\boldsymbol{F}. \tag{2.8}$$

Evidently, $\det(\bar{\boldsymbol{F}}) = 1$ holds for the isochoric contribution $\bar{\boldsymbol{F}}$. This multiplicative split is inherited by the right Cauchy-Green deformation tensor, whose isochoric part is given as $\bar{\boldsymbol{C}} := J^{-\frac{2}{3}}\boldsymbol{C}$.

Volumetric deviatoric split of stress tensors

The Cauchy stress tensor can be additively split into an isochoric and a volumetric part:

$$\boldsymbol{T} = \boldsymbol{T}_E - p\boldsymbol{I} \quad \text{with } p := -\frac{1}{3}\mathrm{tr}\,(\boldsymbol{T}) = -\frac{1}{3}\boldsymbol{T}\cdot\boldsymbol{I}\,. \tag{2.9}$$

The isochoric part, the so-called "extra stress" tensor, $\boldsymbol{T}_E$, is trace-free,

$$\begin{aligned}\mathrm{tr}\,(\boldsymbol{T}_E) &= \mathrm{tr}\,(\boldsymbol{T}+p\boldsymbol{I}) = \mathrm{tr}\left(\boldsymbol{T}-\frac{1}{3}\mathrm{tr}\,(\boldsymbol{T})\boldsymbol{I}\right) = \mathrm{tr}\,(\boldsymbol{T}) - \mathrm{tr}\left(\frac{1}{3}\mathrm{tr}\,(\boldsymbol{T})\boldsymbol{I}\right)\\ &= \mathrm{tr}\,(\boldsymbol{T}) - \frac{1}{3}\mathrm{tr}\,(\boldsymbol{T})\mathrm{tr}\,(\boldsymbol{I}) = \mathrm{tr}\,(\boldsymbol{T}) - \mathrm{tr}\,(\boldsymbol{T}) = 0\,,\end{aligned} \tag{2.10}$$

and hence often also referred to as deviatoric (part of the) stress tensor.
The same split can be performed on the other stress tensors. For the 1st Piola-Kirchhoff stress tensor, one has

$$\boldsymbol{P} = \boldsymbol{P}_{\mathrm{E}} - pJ\boldsymbol{F}^{-T} \quad \text{with } p := -\frac{1}{3}J^{-1}\boldsymbol{P}\cdot\boldsymbol{F}\,. \tag{2.11}$$

And equivalently, for the 2nd Piola-Kirchhoff stress tensor

$$\boldsymbol{S} = \boldsymbol{S}_{\mathrm{E}} - pJ\boldsymbol{C}^{-1} \quad \text{with } p := -\frac{1}{3}J^{-1}\boldsymbol{S}\cdot\boldsymbol{C}\,. \tag{2.12}$$

Note that for these two stress tensors, the traces of $\boldsymbol{P}_{\mathrm{E}}$ and $\boldsymbol{S}_{\mathrm{E}}$ do not vanish. However, with the definition of a "generalised trace operator", making use of the conjugate pairs $\left\{\boldsymbol{P}, \dot{\boldsymbol{F}}\right\}$ and $\left\{\boldsymbol{S}, \frac{1}{2}\dot{\boldsymbol{C}}\right\}$, it holds that

$$\mathrm{tr}_F(\boldsymbol{P}_{\mathrm{E}}) := \boldsymbol{P}_{\mathrm{E}}\cdot\boldsymbol{F} = 0 \quad \text{and} \quad \mathrm{tr}_C(\boldsymbol{S}_{\mathrm{E}}) := \boldsymbol{S}_{\mathrm{E}}\cdot\boldsymbol{C} = 0\,. \tag{2.13}$$

2.3 Constitutive equations

To complete the equations that are necessary to describe an incompressible solid, as a last step, the (1st PIOLA-KIRCHHOFF) stress tensor needs to be specified, i.e. a constitutive equation, suitable to describe the behaviour of the material under consideration has to be formulated. For details on the subject of constitutive modelling and the requirements on thermodynamic consistency, the interested reader is referred to the book [50] and references therein. Nevertheless, to be able to comprehend where the equations come from, this section tries to show their derivation as short as possible, only including the steps that are considered necessary.
An elastic material is defined as a material whose constitutive behaviour is only a function of the current state of deformation. In the special case of a hyperelastic material, the work done by the stresses during a deformation process depends only on the initial state at time t_0 and the final configuration at time t, i.e. the material behaviour is path-independent (c.f. e.g. [35, 36]). For this case, the 1st PIOLA-KIRCHHOFF stress tensor can be derived from the second law of thermodynamics (entropy balance) and a scalar-valued strain energy function ψ, here $\psi = \psi(\boldsymbol{F})$.
We start the derivation with the CLAUSIUS-PLANCK inequality

$$\mathcal{D}_{\text{int}} = \boldsymbol{P} \cdot \dot{\boldsymbol{F}} - \dot{\psi} \geq 0\,. \tag{2.14}$$

Inserting the derivative of the strain energy function, one obtains

$$\boldsymbol{P} \cdot \dot{\boldsymbol{F}} - \dot{\psi} = \boldsymbol{P} \cdot \dot{\boldsymbol{F}} - \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} \cdot \dot{\boldsymbol{F}} = \left(\boldsymbol{P} - \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} \right) \cdot \dot{\boldsymbol{F}} \overset{!}{\geq} 0\,. \tag{2.15}$$

Since Equation (2.15) has to hold for arbitrary $\dot{\boldsymbol{F}}$, the expression in parentheses has to vanish (COLEMAN-NOLL argument) and therefore one obtains the 1st PIOLA-KIRCHHOFF stress tensor from the strain energy function ψ as

$$\boldsymbol{P} = \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}}\,. \tag{2.16}$$

Note that this procedure becomes rather involved for more complex material behaviour such as e.g. a viscoelastic formulation including internal variables. However, this is beyond the scope of this work.

2.3.1 Incompressible hyperelasticity

For the case of an incompressible hyperelastic material, the deformation gradient $\boldsymbol{F}$ and thus its time derivative $\dot{\boldsymbol{F}}$ cannot take arbitrary values anymore, since the following restriction exists,

$$\det(\boldsymbol{F}(\boldsymbol{X}, t)) = J(\boldsymbol{X}, t) \equiv 1\,, \quad \Longrightarrow \quad 0 = \dot{J} = J \boldsymbol{F}^{-T} \cdot \dot{\boldsymbol{F}}\,. \tag{2.17}$$

Therefore, Equation (2.16) does not hold any longer. From Equations (2.15) and (2.17) it can be concluded that both terms left from the scalar dot, are orthogonal to the plane

of admissible $\dot{\boldsymbol{F}}$ and therefore proportional to each other, i.e.

$$\boldsymbol{P} - \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} = \gamma J \boldsymbol{F}^{-T} \quad \Longleftrightarrow \quad \boldsymbol{P} = \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} + \gamma J \boldsymbol{F}^{-T} \,. \tag{2.18}$$

for an arbitrary scalar $\gamma \in \mathbb{R}$.

Remark:
This is illustrated nicely in [10]. Furthermore, they give the advise to retain the Jacobian J in this equation also for the case of an incompressible material.

Relation to hydrostatic pressure

Recalling the volumetric deviatoric split of the stress tensors (c.f. Equations (2.11)), one can relate the arbitrary scalar γ to the hydrostatic pressure p. Insertion of Equation $(2.18)_2$ into Equation $(2.11)_2$, i.e. $p = -1/3\, J^{-1} \boldsymbol{P} \cdot \boldsymbol{F}$, yields

$$\begin{aligned}
& -p = \frac{1}{3} J^{-1} \left(\frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} + \gamma J \boldsymbol{F}^{-T} \right) \cdot \boldsymbol{F} \\
\Longleftrightarrow \; & -p = \frac{1}{3} J^{-1} \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} \cdot \boldsymbol{F} + \frac{1}{3} J^{-1} \gamma J \boldsymbol{F}^{-T} \cdot \boldsymbol{F} \\
\Longleftrightarrow \; & -p = \frac{1}{3} J^{-1} \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} \cdot \boldsymbol{F} + \frac{1}{3} \gamma \boldsymbol{F}^{-T} \boldsymbol{F}^{T} \cdot \boldsymbol{I} \\
\Longleftrightarrow \; & -p = \frac{1}{3} J^{-1} \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} \cdot \boldsymbol{F} + \gamma \,.
\end{aligned} \tag{2.19}$$

It follows that $-p = \gamma$ if and only if $\frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}} \cdot \boldsymbol{F} = 0$. This is the case if $\frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}}$ is deviatoric (in the sense of $\mathrm{tr}_F(\star) = (\star) \cdot \boldsymbol{F}$), which means that the strain energy can be expressed in terms of the deviatoric component $\bar{\boldsymbol{F}} = J^{-\frac{1}{3}} \boldsymbol{F}$ of the deformation gradient, i.e. $\psi = \psi(\bar{\boldsymbol{F}})$. Thus, one defines

$$\bar{\psi}(\boldsymbol{F}) := \psi(\bar{\boldsymbol{F}}) \quad \Longrightarrow \quad \boldsymbol{P}_{\mathrm{E}} := \frac{\partial \bar{\psi}(\boldsymbol{F})}{\partial \boldsymbol{F}} \;\; \text{and} \;\; -p = \gamma \,. \tag{2.20}$$

Analogously, this can be done for the 2nd Piola-Kirchhoff stress tensor, such that one obtains

$$\boldsymbol{S}_{\mathrm{E}} := 2 \frac{\partial \bar{\psi}(\boldsymbol{C})}{\partial \boldsymbol{C}} = 2 \frac{\partial \psi(\bar{\boldsymbol{C}})}{\partial \boldsymbol{C}} \,. \tag{2.21}$$

Furthermore, Equation (2.4) also holds for the deviatoric extra stresses and thus

$$\boldsymbol{P}_{\mathrm{E}} = \boldsymbol{F} \boldsymbol{S}_{\mathrm{E}} = 2 \boldsymbol{F} \frac{\partial \psi(\bar{\boldsymbol{C}})}{\partial \boldsymbol{C}} \,. \tag{2.22}$$

Unfortunately, at least to the authors impression, the volumetric deviatoric split for incompressible materials and the consequences thereof, are rarely accurately formulated, which results in misunderstanding and errors. Also the importance of keeping the Jacobian J in the expression of the (1st Piola-Kirchhoff) stress tensor at this point (at least) for further derivations, is often neglected. Therefore, it is explained and derived in

detail here.

2.3.2 Transversely isotropic hyperelasticity

This section shall just give a brief overview over the principles that need to be fulfilled to derive the necessary equations to model a general transversely isotropic material. It is kept short on purpose and the interested reader is referred to e.g. [50] for the complete theory behind this.
In constitutive modelling, the principle of material objectivity requires the material response to be invariant under rigid body motions of the actual configuration. However, for an arbitrary rotation $\boldsymbol{Q} \in \mathcal{SO}(3)$ the equality $\psi(\boldsymbol{QF}) = \psi(\boldsymbol{F})$ does not necessarily hold for arbitrary strain energy functions ψ. Therefore, the deformation gradient $\boldsymbol{F}$ is not a suitable deformation measure for this application. In order to fulfil the invariance condition, the strain energy function needs to be expressed in terms of the right CAUCHY-GREEN deformation tensor $\boldsymbol{C}$, i.e.

$$\psi = \psi(\boldsymbol{C})\,. \tag{2.23}$$

Another principle to consider is the principle of material symmetry. While the mechanical behaviour of a purely isotropic material is independent of the orientation of the material in the reference configuration with respect to the applied forces, the response of a transversely isotropic material in contrast, depends on its orientation, e.g. due to fibres embedded in a matrix. Therefore, a fibre direction $\boldsymbol{a}_0 \in \mathbb{R}^3, \|\boldsymbol{a}_0\|_2 = 1$ is introduced. The associated structural tensor $\boldsymbol{\mathcal{M}} := \boldsymbol{a}_0 \otimes \boldsymbol{a}_0$ has the following properties:

$$\boldsymbol{\mathcal{M}}^T = \boldsymbol{\mathcal{M}}\,, \quad \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{M}} \quad \text{and} \quad \operatorname{tr}(\boldsymbol{\mathcal{M}}) = 1\,. \tag{2.24}$$

The concept of transversal isotropy makes use of an additional quantity / measure, the so-called fibre stretch $\lambda_f := \|\boldsymbol{a}\|_2$, which is defined as the length of the fibre in the actual configuration $\boldsymbol{a} := \boldsymbol{F}\boldsymbol{a}_0$.
So, for a transversely isotropic material, the strain energy is a function of the tensors $\boldsymbol{C}$ and $\boldsymbol{\mathcal{M}}$. Additionally, to obtain a consistent formulation of the constitutive equations, it is convenient to express the dependency on these two tensor arguments in terms of so-called scalar (mixed) invariants. This way, the constitutive equation becomes invariant under rotations of the respective symmetry group of the considered material and thus the principle of material symmetry is a priori fulfilled.

2.3.3 Introduction of invariants and their derivatives

The following principle invariants (I_1, I_2, I_3) and mixed invariants (I_4, I_5) of the right CAUCHY-GREEN deformation gradient $\boldsymbol{C}$ and the structural tensor $\boldsymbol{\mathcal{M}}$ are introduced:

$$\begin{aligned}
I_1 &:= \operatorname{tr}(\boldsymbol{C}) = \boldsymbol{C} \cdot \boldsymbol{I}\,,\\
I_2 &:= \operatorname{tr}(\operatorname{cof}(\boldsymbol{C})) = \frac{1}{2}\left[\operatorname{tr}(\boldsymbol{C})^2 - \operatorname{tr}(\boldsymbol{C}^2)\right]\,,\\
I_3 &:= \det(\boldsymbol{C}) = J^2\,,
\end{aligned}$$

$$I_4 := \mathrm{tr}\,(\boldsymbol{\mathcal{M}}\boldsymbol{C}) = \boldsymbol{a}_0 \cdot \boldsymbol{C}\,\boldsymbol{a}_0 = \boldsymbol{F}\boldsymbol{a}_0 \cdot \boldsymbol{F}\boldsymbol{a}_0 = \boldsymbol{a}\cdot\boldsymbol{a} = \|\boldsymbol{a}\|_2^2 = \lambda_f^2\,,$$
$$I_5 := \mathrm{tr}\,(\boldsymbol{\mathcal{M}}\boldsymbol{C}^2)\,.$$

With those, the strain energy function for a transversely isotropic material is written as

$$\psi_{\mathrm{trviso}} = \psi_{\mathrm{trviso}}(I_1, I_2, I_3, I_4, I_5) = \psi_{\mathrm{iso}}(I_1, I_2, I_3) + \psi_{\mathrm{aniso}}(I_1, I_2, I_3, I_4, I_5)\,. \tag{2.25}$$

A common simplification (in particular for fibre reinforced materials) is to assume a decoupling of isotropic and anisotropic energy terms, which leads to the form

$$\psi_{\mathrm{trviso}} = \psi_{\mathrm{trviso}}(I_1, I_2, I_3, I_4, I_5) = \psi_{\mathrm{iso}}(I_1, I_2, I_3) + \psi_{\mathrm{aniso}}(I_4, I_5)\,. \tag{2.26}$$

As they are needed in the course of this chapter, the derivatives of the invariants with respect to the right Cauchy-Green deformation gradient $\boldsymbol{C}$ are also derived at this point. The general rule needed in this context is the derivative of the trace operator with respect to a tensor, or even more general the derivative of the scalar product of two tensors with respect to a tensor. For arbitrary tensors $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{R}^{3\times3}$ it holds

$$\frac{\partial(\boldsymbol{A}\cdot\boldsymbol{B})}{\partial\boldsymbol{C}} = \left(\frac{\partial\boldsymbol{A}}{\partial\boldsymbol{C}}\right)^T\boldsymbol{B} + \left(\frac{\partial\boldsymbol{B}}{\partial\boldsymbol{C}}\right)^T\boldsymbol{A} \quad \text{and thus} \tag{2.27}$$

$$\frac{\partial\,\mathrm{tr}\,(\boldsymbol{A})}{\partial\boldsymbol{C}} = \frac{\partial(\boldsymbol{A}\cdot\boldsymbol{I})}{\partial\boldsymbol{C}} = \left(\frac{\partial\boldsymbol{A}}{\partial\boldsymbol{C}}\right)^T\boldsymbol{I}\,. \tag{2.28}$$

Making use of this rule and keeping in mind that $\boldsymbol{C}$ and $\boldsymbol{\mathcal{M}}$ are symmetric tensors, one derives *

$$\frac{\partial I_1}{\partial\boldsymbol{C}} = \frac{\partial\,\mathrm{tr}\,(\boldsymbol{C})}{\partial\boldsymbol{C}} = \left(\frac{\partial\boldsymbol{C}}{\partial\boldsymbol{C}}\right)^T\boldsymbol{I} = \left((\boldsymbol{I}\otimes\boldsymbol{I})^{\overset{23}{T}}\right)^T\boldsymbol{I} = (\boldsymbol{I}\otimes\boldsymbol{I})^{\overset{23}{T}}\boldsymbol{I} = \boldsymbol{I}\,, \tag{2.29}$$

$$\begin{aligned}\frac{\partial I_2}{\partial\boldsymbol{C}} &= \frac{\partial\,\frac{1}{2}\left[\mathrm{tr}\,(\boldsymbol{C})^2 - \mathrm{tr}\,(\boldsymbol{C}^2)\right]}{\partial\boldsymbol{C}} = \frac{1}{2}\left[\frac{\partial\,\mathrm{tr}\,(\boldsymbol{C})^2}{\partial\boldsymbol{C}} - \frac{\partial\,\mathrm{tr}\,(\boldsymbol{C}^2)}{\partial\boldsymbol{C}}\right]\\ &= \frac{1}{2}\left[2\,\mathrm{tr}\,(\boldsymbol{C})\boldsymbol{I} - \frac{\partial(\boldsymbol{C}\cdot\boldsymbol{C})}{\partial\boldsymbol{C}}\right] = \frac{1}{2}\left[2\,\mathrm{tr}\,(\boldsymbol{C})\boldsymbol{I} - 2\,(\boldsymbol{I}\otimes\boldsymbol{I})^{\overset{23}{T}}\boldsymbol{C}\right]\\ &= \frac{1}{2}\left[2\,\mathrm{tr}\,(\boldsymbol{C})\boldsymbol{I} - 2\,\boldsymbol{C}\right] = I_1\boldsymbol{I} - \boldsymbol{C}\,,\end{aligned} \tag{2.30}$$

$$\frac{\partial I_3}{\partial\boldsymbol{C}} = \frac{\partial\,\det(\boldsymbol{C})}{\partial\boldsymbol{C}} = \det(\boldsymbol{C})\boldsymbol{C}^{-T} = I_3\boldsymbol{C}^{-1}\,, \tag{2.31}$$

$$\frac{\partial J}{\partial\boldsymbol{C}} = \frac{\partial\,I_3^{\frac{1}{2}}}{\partial\boldsymbol{C}} = \frac{\partial\,I_3^{\frac{1}{2}}}{\partial I_3}\frac{\partial I_3}{\partial\boldsymbol{C}} = \frac{1}{2}I_3^{-\frac{1}{2}}I_3\boldsymbol{C}^{-1} = \frac{1}{2}I_3^{\frac{1}{2}}\boldsymbol{C}^{-1} = \frac{1}{2}J\boldsymbol{C}^{-1}\,, \tag{2.32}$$

$$\frac{\partial I_4}{\partial\boldsymbol{C}} = \frac{\partial\,\mathrm{tr}\,(\boldsymbol{\mathcal{M}}\boldsymbol{C})}{\partial\boldsymbol{C}} = \frac{\partial(\boldsymbol{\mathcal{M}}\cdot\boldsymbol{C})}{\partial\boldsymbol{C}} = (\boldsymbol{I}\otimes\boldsymbol{I})^{\overset{23}{T}}\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{M}}\,. \tag{2.33}$$

The two subsequent sections introduce possible strain energy functions for the isotropic as

* $(\boldsymbol{I}\otimes\boldsymbol{I})^{\overset{23}{T}}$ is the fourth-order identity tensor, where $(\cdot)^{\overset{23}{T}}$ means a transpose of bases two and three. It is also common to use the notation $\mathbb{I}$.

well as for the anisotropic contribution. The reason for this rather detailed introduction lies in the incompressibility, which requires a thoughtful derivation of the extra stress tensors.

2.3.4 Possible incompressible isotropic contributions

In this section, two specific strain energy functions for isotropic materials, namely the NEO-HOOKE and the DEMIRAY material shall be introduced and the corresponding 1st and 2nd PIOLA-KIRCHHOFF stress tensors are derived. As explained in Section 2.3.1, it proofs useful to express the strain energy in terms of deviatoric measures in the case of an incompressible material, thus

$$\psi_{\text{iso}} = \psi_{\text{iso}}(\bar{\boldsymbol{C}}, \boldsymbol{X}) = \psi_{\text{iso}}(\bar{I}_1, \bar{I}_2, \bar{I}_3)\,, \tag{2.34}$$

with

$$\bar{I}_1 := \text{tr}\,(\bar{\boldsymbol{C}}) = \text{tr}\,\left(J^{-\frac{2}{3}}\boldsymbol{C}\right) = J^{-\frac{2}{3}}\text{tr}\,(\boldsymbol{C}) = J^{-\frac{2}{3}} I_1 \tag{2.35}$$

$$\begin{aligned}\bar{I}_2 &:= \text{tr}\,(\text{cof}\,(\bar{\boldsymbol{C}})) = \frac{1}{2}\left[\text{tr}\,(\bar{\boldsymbol{C}})^2 - \text{tr}\,(\bar{\boldsymbol{C}}^2)\right] = \frac{1}{2}\left[\text{tr}\,(J^{-\frac{2}{3}}\boldsymbol{C})^2 - \text{tr}\,(J^{-\frac{4}{3}}\boldsymbol{C}^2)\right] \\ &= \frac{1}{2}\left[\left(J^{-\frac{2}{3}}\text{tr}\,(\boldsymbol{C})\right)^2 - J^{-\frac{4}{3}}\text{tr}\,(\boldsymbol{C}^2)\right] = \frac{1}{2}J^{-\frac{4}{3}}\left[\text{tr}\,(\boldsymbol{C})^2 - \text{tr}\,(\boldsymbol{C}^2)\right] \\ &= J^{-\frac{4}{3}} I_2\end{aligned} \tag{2.36}$$

$$\begin{aligned}\bar{I}_3 &:= \det(\bar{\boldsymbol{C}}) = \det\left(J^{-\frac{2}{3}}\boldsymbol{C}\right) = J^{-\frac{2}{3}}\det(\boldsymbol{C}) = J^{-\frac{2}{3}} I_3 \\ &= J^{-\frac{2}{3}} J^2 = J^{\frac{4}{3}}\end{aligned} \tag{2.37}$$

Note that there exists the dependency $\bar{I}_i = \bar{I}_i(J, I_i), \forall i \in \{1, 2, 3\}$. Again, for further derivations, it is essential to keep the Jacobian in these expressions, even for the case of an incompressible material.

Incompressible Neo-Hooke solid

The NEO-HOOKE material is a simplification of the MOONEY-RIVLIN material and based on the principle invariant I_1 only. It depends on a single scalar material parameter c_{10} [MPa] and is given in its general form (see e.g. [35, 36]) by,

$$\psi^{\text{NH}}(I_1, I_2) = \psi^{\text{NH}}(I_1) = c_{10}(I_1 - 3)\,. \tag{2.38}$$

Remark:
Sometimes the function is also formulated as $\psi^{\text{NH}}(I_1) = \frac{1}{2}\mu(I_1 - 3)$. Here, one needs to pay attention to the factor of two, when establishing material parameters found in literature.

For an incompressible NEO-HOOKE material the strain energy is a function of I_1 and J,

$$\psi^{\text{NH}}(\bar{\boldsymbol{C}}) = \psi^{\text{NH}}(\bar{I}_1) = c_{10}(\bar{I}_1 - 3) = c_{10}(J^{-\frac{2}{3}} I_1 - 3) = \psi^{\text{NH}}(J, I_1)\,. \tag{2.39}$$

It follows

$$\begin{aligned}\frac{\partial \psi^{\mathrm{NH}}(J,I_1)}{\partial \boldsymbol{C}} &= \frac{\partial \psi^{\mathrm{NH}}(J,I_1)}{\partial I_1}\frac{\partial I_1}{\partial \boldsymbol{C}} + \frac{\partial \psi^{\mathrm{NH}}(J,I_1)}{\partial J}\frac{\partial J}{\partial \boldsymbol{C}} \\ &= c_{10}J^{-\frac{2}{3}}\boldsymbol{I} + c_{10}\left(-\frac{2}{3}\right)J^{-\frac{5}{3}}I_1\frac{1}{2}J\boldsymbol{C}^{-1} \\ &= c_{10}J^{-\frac{2}{3}}\boldsymbol{I} - c_{10}\frac{1}{3}J^{-\frac{2}{3}}I_1\boldsymbol{C}^{-1} \\ &= c_{10}J^{-\frac{2}{3}}\left(\boldsymbol{I} - \frac{1}{3}I_1\boldsymbol{C}^{-1}\right).\end{aligned} \tag{2.40}$$

Inserting this result into Equations (2.21) and (2.22) respectively, the 1st and 2nd Piola-Kirchhoff extra stress tensor for a Neo-Hooke solid are obtained as

$$\boldsymbol{S}_{\mathrm{E}}^{\mathrm{NH}} = 2c_{10}J^{-\frac{2}{3}}\left(\boldsymbol{I} - \frac{1}{3}I_1\boldsymbol{C}^{-1}\right), \text{ and} \tag{2.41}$$

$$\boldsymbol{P}_{\mathrm{E}}^{\mathrm{NH}} = 2c_{10}J^{-\frac{2}{3}}\left(\boldsymbol{F} - \frac{1}{3}I_1\boldsymbol{F}^{-T}\right). \tag{2.42}$$

Remark:
There exist three reasons, why it is essential to keep the dependency on J at this point also for the incompressible case.

1. Requirement of a stress-free reference configuration:

   In the reference configuration, it holds $\boldsymbol{F} = \boldsymbol{I} \Longrightarrow \boldsymbol{C}^{-1} = \boldsymbol{I}, I_1 = 3$. Insertion into Equations (2.41) and (2.42) yields $\boldsymbol{S}_{\mathrm{E}}^{\mathrm{NH}} = \boldsymbol{P}_{\mathrm{E}}^{\mathrm{NH}} = \boldsymbol{0}$. This would not be the case without the deviatoric volumetric split and without retaining the dependency on J.

2. Further derivation of elasticity tensors:

   Dropping the dependency on J at this point already, would result in missing terms when deriving the elasticity tensors later on for the linearisation in Section 3.2.

3. According to [10] it is also improving the numerical performance of the FE-code, when the J is kept in the formulation as numerical inaccuracies are balanced better.

This material law was chosen, as it represents one of the simpler isotropic material models and shall thus be employed for first tests in the course of this work.

Incompressible Demiray solid

The Demiray strain energy function is one of the simpler exponential material laws, accounting for the strain-hardening (of skeletal muscle tissue) by an exponential term. In this work, it serves as example for demonstrating the effect of an increasing material complexity on the investigated methods. For further possible choices of hyperelastic strain energy functions that are considered suitable for soft biological tissues, the interested reader is referred to the relatively recent review paper [15]. Like the Neo-Hooke material, the Demiray strain energy is only I_1-based. Furthermore, it depends on two

scalar material parameters c_1 [MPa], c_2 [–] and is given in its general form as (c.f. [19])

$$\psi^{\mathrm{D}}(I_1) = \frac{c_1}{c_2}\left(\exp\left(\frac{1}{2}c_2(I_1-3)\right)-1\right). \tag{2.43}$$

For an incompressible DEMIRAY material, the derivative of the strain energy with respect to the right CAUCHY-GREEN deformation gradient $\boldsymbol{C}$ is obtained as

$$\begin{aligned}\frac{\partial\psi^{\mathrm{D}}(\bar{\boldsymbol{C}})}{\partial\boldsymbol{C}} &= \frac{\partial\psi^{\mathrm{D}}(J,I_1)}{\partial I_1}\frac{\partial I_1}{\partial\boldsymbol{C}} + \frac{\partial\psi^{\mathrm{D}}(J,I_1)}{\partial J}\frac{\partial J}{\partial\boldsymbol{C}} \\ &= \frac{1}{2}c_1 J^{-\frac{2}{3}}\exp\left(\frac{1}{2}c_2(J^{-\frac{2}{3}}I_1-3)\right)\boldsymbol{I} \\ &\quad -\frac{1}{3}c_1 J^{-\frac{5}{3}}I_1\exp\left(\frac{1}{2}c_2(J^{-\frac{2}{3}}I_1-3)\right)\frac{1}{2}J\boldsymbol{C}^{-1}.\end{aligned} \tag{2.44}$$

Inserting this result into Equations (2.21) and (2.22) respectively, the 1st and 2nd PIOLA-KIRCHHOFF extra stress tensor for a DEMIRAY solid are given as

$$\boldsymbol{S}^{\mathrm{D}}_{\mathrm{E}} = c_1 J^{-\frac{2}{3}}\exp\left(\frac{1}{2}c_2(J^{-\frac{2}{3}}I_1-3)\right)\left(\boldsymbol{I}-\frac{1}{3}I_1\boldsymbol{C}^{-1}\right), \text{ and} \tag{2.45}$$

$$\boldsymbol{P}^{\mathrm{D}}_{\mathrm{E}} = c_1 J^{-\frac{2}{3}}\exp\left(\frac{1}{2}c_2(J^{-\frac{2}{3}}I_1-3)\right)\left(\boldsymbol{F}-\frac{1}{3}I_1\boldsymbol{F}^{-T}\right). \tag{2.46}$$

Remarks:

1. Setting $c_2 = 0$ corresponds to the NEO-HOOKE stress tensors.

2. Insertion of $\boldsymbol{F} = \boldsymbol{I}$ yields a stress-free reference configuration here as well.

2.3.5 Possible anisotropic contribution

Lastly, a specific strain energy function describing anisotropic material behaviour needs to be introduced and the corresponding summands for the 1st and 2nd PIOLA-KIRCHHOFF stress tensors shall be derived. As investigated and explained in e.g. [42] and [27], the anisotropic strain energy needs to be expressed in terms of $\boldsymbol{C}$ and its invariants also in the case of an incompressible material (not using the deviatoric measure $\bar{\boldsymbol{C}}$), i.e.

$$\psi_{\mathrm{aniso}} = \psi_{\mathrm{aniso}}(\boldsymbol{C},\boldsymbol{X}) = \psi_{\mathrm{aniso}}(I_4,I_5). \tag{2.47}$$

Since the aim of this work is to provide and investigate a framework, which is suitable to describe skeletal muscle material behaviour in general, we will not go into further details of constitutive modelling, but consider it sufficient to choose one of the available anisotropic strain energy functions and fit them to available experimental data (see Section 6.1.3). For the time being, the strain energy function proposed by Holzapfel et al. [30], originally developed for arterial walls, is chosen.

Holzapfel strain energy function

The HOLZAPFEL strain energy function is an exponential material law based on the fourth (mixed) invariant, I_4, and depends on two scalar material parameters a_1 [MPa] and a_2 [–]. It is given as (c.f. [30])

$$\psi^{\mathrm{HA}}(I_4) = \frac{1}{2}\frac{a_1}{a_2}\left(\exp\left(a_2(I_4-1)^2\right)-1\right) . \tag{2.48}$$

Taking the derivative with respect to $\boldsymbol{C}$ yields

$$\frac{\partial \psi^{\mathrm{HA}}(I_4)}{\partial \boldsymbol{C}} = \frac{\partial \psi^{\mathrm{HA}}(I_4)}{\partial I_4}\frac{\partial I_4}{\partial \boldsymbol{C}} = a_1(I_4-1)\exp\left(a_2(I_4-1)^2\right)\boldsymbol{\mathcal{M}} . \tag{2.49}$$

Inserting this result into Equations (2.21) and (2.22) respectively, the summands for the 1st and 2nd PIOLA-KIRCHHOFF extra stress tensors for the HOLZAPFEL material are given as

$$\boldsymbol{S}_{\mathrm{E}}^{\mathrm{HA}} = 2a_1(I_4-1)\exp\left(a_2(I_4-1)^2\right)\boldsymbol{\mathcal{M}} , \tag{2.50}$$
$$\boldsymbol{P}_{\mathrm{E}}^{\mathrm{HA}} = 2a_1(I_4-1)\exp\left(a_2(I_4-1)^2\right)\boldsymbol{F}\boldsymbol{\mathcal{M}} . \tag{2.51}$$

For $\boldsymbol{F}=\boldsymbol{I}$, it is $I_4 = \|\boldsymbol{a}_0\|_2^2 = 1$ and thus it can easily be shown that the requirement of a stress-free reference configuration is met here as well.

2.4 Linearisation of nonlinear constitutive material laws

In the special case of nonlinear solid mechanics, it is necessary to perform a linearisation at some point of the solution procedure. This requires certain derivatives, which yield higher-order tensors. Here, the necessary operations for this work are listed or derived. The list is by no means complete. However, the aim is to provide the reader with the necessary rules that are applied within the derivations. For more detailed explanations and derivations see, e.g. [47]
In Section 2.3.3 the derivatives of the invariants with respect to the right CAUCHY-GREEN deformation gradient $\boldsymbol{C}$ have already been derived, as they were necessary to obtain the specific forms of the 1st and 2nd PIOLA-KIRCHHOFF extra stress tensors. For the linearisation within the setting established in this work, the derivative of the 2nd PIOLA-KIRCHHOFF extra stress tensor with respect to the deformation gradient $\boldsymbol{F}$, i.e. $\frac{\partial \boldsymbol{P}_{\mathrm{E}}}{\partial \boldsymbol{F}}$ is needed. This fourth-order tensor is often referred to as the analytical tangent modulus or elasticity tensor.

Tensor calculus: Product rules in general

The general product rules for derivatives of combinations of scalars, vectors and tensors, that are used in the course of this section are listed in the following.
Let $\alpha, \beta \in \mathbb{R}$ be arbitrary scalars, $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{3\times 1}$ arbitrary vectors and $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{3\times 3}$

arbitrary tensors. Then it holds

$$\frac{\partial(\alpha\beta)}{\partial \boldsymbol{B}} = \alpha\frac{\partial\beta}{\partial \boldsymbol{B}} + \beta\frac{\partial\alpha}{\partial \boldsymbol{B}}\,, \tag{2.52}$$

$$\frac{\partial(\alpha\boldsymbol{A})}{\partial \boldsymbol{B}} = \boldsymbol{A}\otimes\frac{\partial\alpha}{\partial \boldsymbol{B}} + \alpha\frac{\partial \boldsymbol{A}}{\partial \boldsymbol{B}} \quad\Longrightarrow\quad \frac{\partial(\alpha\boldsymbol{A})}{\partial \boldsymbol{A}} = \boldsymbol{A}\otimes\frac{\partial\alpha}{\partial \boldsymbol{A}} + \alpha\left(\boldsymbol{I}\otimes\boldsymbol{I}\right)^{\overset{23}{T}}\,, \tag{2.53}$$

$$\frac{\partial(\boldsymbol{A}^T\boldsymbol{A})}{\partial \boldsymbol{A}} = \left(\boldsymbol{A}^T\otimes\boldsymbol{I}\right)^{\overset{23}{T}} + \left(\boldsymbol{I}\otimes\boldsymbol{A}\right)^{\overset{24}{T}}\,, \tag{2.54}$$

$$\frac{\partial(\boldsymbol{A}\boldsymbol{B})}{\partial \boldsymbol{B}} = \left(\boldsymbol{A}\otimes\boldsymbol{I}\right)^{\overset{23}{T}}\,, \quad \text{if } \boldsymbol{A}\neq\boldsymbol{A}(\boldsymbol{B})\,, \tag{2.55}$$

$$\frac{\partial(\boldsymbol{A}\boldsymbol{B})}{\partial \boldsymbol{A}} = \left(\boldsymbol{I}\otimes\boldsymbol{B}^T\right)^{\overset{23}{T}}\,, \quad \text{if } \boldsymbol{B}\neq\boldsymbol{B}(\boldsymbol{A})\,, \tag{2.56}$$

$$\frac{\partial(\boldsymbol{A}\boldsymbol{v})}{\partial \boldsymbol{u}} = \left(\frac{\partial \boldsymbol{A}}{\partial \boldsymbol{u}}\right)^{\overset{23}{T}}\boldsymbol{v} + \boldsymbol{A}\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{u}}\,. \tag{2.57}$$

The last product rule (2.57) is additionally directly applied here in the context in which it is used later in Section 3.3.3. Further, index notation is used in order to understand where the transpose of the bases 2 and 3 comes from.

$$\begin{aligned}\frac{\partial \boldsymbol{P}(\boldsymbol{u})\boldsymbol{n}^k}{\partial \boldsymbol{u}^j} &= \frac{\partial P_{ab}n_b^k\boldsymbol{e}_a}{\partial u_c^j\boldsymbol{e}_c} = \frac{\partial P_{ab}n_b^k}{\partial u_c^j}(\boldsymbol{e}_a\otimes\boldsymbol{e}_c) = \left(\frac{\partial P_{ab}}{\partial u_c^j}n_b^k + P_{ab}\frac{\partial n_b^k}{\partial u_c^j}\right)(\boldsymbol{e}_a\otimes\boldsymbol{e}_c)\\ &= \frac{\partial P_{ab}}{\partial u_c^j}n_b^k(\boldsymbol{e}_a\otimes\boldsymbol{e}_c) = \frac{\partial P_{ab}}{\partial u_c^j}(\boldsymbol{e}_a\otimes\boldsymbol{e}_c\otimes\boldsymbol{e}_b)n_f^k\boldsymbol{e}_f = \left(\frac{\partial \boldsymbol{P}(\boldsymbol{u})}{\partial \boldsymbol{u}^j}\right)^{\overset{23}{T}}\boldsymbol{n}^k\end{aligned} \tag{2.58}$$

Derivatives of some scalar invariants with respect to $\boldsymbol{F}$

The derivative of terms with the invariants (i.e. scalars) of $\boldsymbol{C}$ and $\boldsymbol{\mathcal{M}}$, occurring in the utilised stress tensors, with respect to the deformation gradient $\boldsymbol{F}$, yield second-order tensors:

$$\frac{\partial J^{-q}}{\partial \boldsymbol{F}} = \frac{\partial J^{-q}}{\partial J}\frac{\partial J}{\partial \boldsymbol{F}} = -qJ^{-q-1}J\boldsymbol{F}^{-T} = -qJ^{-q}\boldsymbol{F}^{-T} \tag{2.59}$$

$$\begin{aligned}\frac{\partial I_1}{\partial \boldsymbol{F}} &= \frac{\partial\,\mathrm{tr}\,(\boldsymbol{C})}{\partial \boldsymbol{F}} = \left(\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{F}}\right)^T\frac{\partial\,\mathrm{tr}\,(\boldsymbol{C})}{\partial \boldsymbol{C}} \overset{(2.29)}{=} \left(\frac{\partial(\boldsymbol{F}^T\boldsymbol{F})}{\partial \boldsymbol{F}}\right)^T\boldsymbol{I}\\ &\overset{(2.54)}{=} \left[\left(\boldsymbol{F}^T\otimes\boldsymbol{I}\right)^{\overset{23}{T}} + \left(\boldsymbol{I}\otimes\boldsymbol{F}\right)^{\overset{24}{T}}\right]^T\boldsymbol{I} = 2\boldsymbol{F}\end{aligned} \tag{2.60}$$

$$\begin{aligned}\frac{\partial I_2}{\partial \boldsymbol{F}} &= \left(\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{F}}\right)^T\frac{\partial I_2}{\partial \boldsymbol{C}} \overset{(2.31)}{=} \left[\left(\boldsymbol{F}^T\otimes\boldsymbol{I}\right)^{\overset{23}{T}} + \left(\boldsymbol{I}\otimes\boldsymbol{F}\right)^{\overset{24}{T}}\right]^T(I_1\boldsymbol{I}-\boldsymbol{C})\\ &\overset{(2.60)}{=} 2I_1\boldsymbol{F} - 2\boldsymbol{F}\boldsymbol{C}\end{aligned} \tag{2.61}$$

$$\frac{\partial I_4}{\partial \boldsymbol{F}} = \left(\frac{\partial \boldsymbol{C}}{\partial \boldsymbol{F}}\right)^T\frac{\partial I_4}{\partial \boldsymbol{C}} \overset{(2.33)}{=} 2\boldsymbol{F}\boldsymbol{\mathcal{M}} \tag{2.62}$$

Derivatives of some second-order tensors with respect to $\boldsymbol{F}$

The derivatives with respect to the deformation gradient $\boldsymbol{F}$ of terms that include the second-order tensors $\boldsymbol{F}$, $\boldsymbol{C}$ and $\boldsymbol{\mathcal{M}}$, yield fourth-order tensors:

$$\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{F}} = \mathbb{I} = (\boldsymbol{I} \otimes \boldsymbol{I})^{23}_{T} \tag{2.63}$$

$$\frac{\partial \boldsymbol{F}^{-T}}{\partial \boldsymbol{F}} = \left(\frac{\partial \boldsymbol{F}^{-1}}{\partial \boldsymbol{F}}\right)^{12}_{T} = \left(-\left(\boldsymbol{F}^{-1} \otimes \boldsymbol{F}^{-T}\right)^{23}_{T}\right)^{12}_{T} = -\left(\boldsymbol{F}^{-T} \otimes \boldsymbol{F}^{-T}\right)^{24}_{T} \tag{2.64}$$

$$\frac{\partial (\boldsymbol{F}\boldsymbol{\mathcal{M}})}{\partial \boldsymbol{F}} \overset{(2.56)}{=} (\boldsymbol{I} \otimes \boldsymbol{\mathcal{M}})^{23}_{T} \tag{2.65}$$

For the last derivative use was made of the symmetry of $\boldsymbol{\mathcal{M}}$.

3 Full-order skeletal muscle model

The aim of this chapter is to introduce a skeletal muscle model in a numerically accessible algebraic formulation as basis for the subsequent chapters. In the context of model order reduction, this will be referred to as the full-order model, and represent the starting point for the application of model order reduction methods. The chapter is divided into three sections. Section 3.1 supplements the 1st PIOLA-KIRCHHOFF stress tensor derived from constitutive equations in the previous chapter with further terms that are necessary to describe the specific material response of skeletal muscle tissue. Then, in Section 3.2, the elasticity tensor needed in the linearisation step of the solution method is derived for that specific material model. Starting from the continuous problem in its strong form, deriving the continuous formulation in its weak form and discretising it by finite elements in space and an implicit EULER scheme in time, Section 3.3 provides the finite element matrix formulation of the skeletal muscle model.

3.1 Skeletal muscle specific material response

The complex architecture and function of skeletal muscle makes it quite a challenging material to model. The interested reader is referred to the book of Fung [22] and the elaborate review paper of Röhrle et al. [39] as an overview over the topic and a starting point for deeper knowledge and further references. The muscle fibres embedded in the extracellular matrix do not only contribute to the passive material behaviour, but additionally exhibit an active material response. Aspects that are considered important for investigating and making universally valid statements of the performance of a full- and a reduced-order model, and should therefore be incorporated in the model of this work, are (i) dynamics, (ii) incompressibility, and nonlinear constitutive equations for the (iii) isotropic passive material response of extracellular matrix and skeletal muscle fibres and for the (iv) anisotropic passive and (v) active material response of skeletal muscle fibres. As a lot of development and progress is going on in the field of constitutive modelling of soft biological tissue in general and skeletal muscle tissue in particular, the aim of this work is to provide a framework, where the specific material response could potentially be easily exchanged to a newly developed, more appropriate material law.
Mathematically, the incorporation of these aspects can be formulated by the following additive split of the 1st PIOLA-KIRCHHOFF stress tensor (c.f. Equation (2.11)),

$$\boldsymbol{P} = \boldsymbol{P}_{\mathrm{E}}^{\mathrm{nl}} + \boldsymbol{P}_{\mathrm{E}}^{\mathrm{visc}} - p\, J \boldsymbol{F}^{-T}\,, \tag{3.1}$$

where $\boldsymbol{P}_{\mathrm{E}}^{\mathrm{visc}}$ describes the viscous material response, and the nonlinear deviatoric extra stress, $\boldsymbol{P}_{\mathrm{E}}^{\mathrm{nl}}$, is further subdivided into the passive and active contributions from muscle

fibres and extracellular matrix

$$\boldsymbol{P}_{\mathrm{E}}^{\mathrm{nl}} = \boldsymbol{P}_{\mathrm{E}}^{\mathrm{iso}} + \boldsymbol{P}_{\mathrm{E}}^{\mathrm{aniso}} + \boldsymbol{P}_{\mathrm{E}}^{\mathrm{act}} \,. \tag{3.2}$$

For the incompressible isotropic contribution, $\boldsymbol{P}_{\mathrm{E}}^{\mathrm{iso}}$, we will (depending on the desired complexity) utilise either the Neo-Hooke or the Demiray material in the form derived in Section 2.3.4. The anisotropic passive contribution, $\boldsymbol{P}_{\mathrm{E}}^{\mathrm{aniso}}$, will be described by the Holzapfel material law derived in Section 2.3.5.
Even though this work mainly intends to investigate the structure and stability of the overall model and means to provide a proof of concept for a generally accepted skeletal muscle material, one should mention at this point, that the transversal isotropic passive material behaviour of skeletal muscle tissue is still a (controversially) discussed topic. Böl et al. [9] for example showed, that the passive response of skeletal muscle tissue is not simply comparable to the behaviour of general fibre reinforced materials, where the fibres merely influence the response to applied traction forces, but that additionally the fibres alternate the behaviour under compression. Furthermore, there is a lot of research going on in improving and enhancing skeletal muscle material models, e.g. using a microstructurally and homogenisation-based approach (c.f. [7]).
The two additional terms that were not discussed yet, are the active contribution and a viscous contribution.

Active contribution

The active response of skeletal muscle originates from contractions within the skeletal muscle fibres, or more specifically from contractions of the sarcomeres. This contribution is naturally acting in fibre direction and thus, like the passive anisotropic contribution, depends on the fourth (mixed) invariant, I_4. Additionally, there exist complex force-length and force-velocity dependencies, but since these aspects do not influence the structure of the discretised equation system, they are omitted here. A very simple formulation of the active stress tensor suitable for insertion into a continuum-mechanical model and sufficient for a proof of concept (c.f. [39])

$$\boldsymbol{P}_{\mathrm{E}}^{\mathrm{act}}(\alpha, I_4) = \alpha \cdot p_{\max}\, \boldsymbol{F}\boldsymbol{\mathcal{M}} \,, \tag{3.3}$$

was incorporated. Therein, $\alpha \in [0,1]$ is the normalised active stress, which can be interpreted as an activation parameter with $\alpha = 0$ meaning that no active behaviour is present and $\alpha = 1$ standing for a fully activated muscle. Furthermore, $p_{\max}\,[\mathrm{MPa}]$ is a material parameter denoting the maximum stress during an isometric contraction.

Viscous contribution

While experiments typically state the passive response of skeletal muscle tissue as viscoelastic (e.g. [48]), most state of the art skeletal muscle models simply incorporate a hyperelastic material behaviour. Especially in a dynamic setting, however, Van Loocke et al. [48] show that it is required and crucial to utilise a viscoelastic material model to appropriately capture the muscles dynamic behaviour. Since dynamics is one of the five aspects (i)–(v) that we want to consider in the skeletal muscle model, the need to additionally

incorporate a viscous contribution clearly exists. As a preliminary and rather simple approach, inspired by the response of a single dashpot (strain rate is proportional to stress), this is incorporated into the stress response by adding the term

$$\boldsymbol{P}_{\mathrm{E}}^{\mathrm{visc}} = \eta \dot{\boldsymbol{F}}\,, \tag{3.4}$$

with η [MPa ms] being a parameter describing the dynamic viscosity. This formulation corresponds to the KELVIN-VOIGT model, represented by a single dashpot and an elastic spring in parallel. Being aware of the shortcomings of the KELVIN-VOIGT model that is known to be incapable of properly describing stress relaxation and to react to a jump in the strain variable with an infinite stress, this contribution surely is questionable. For future work, it is advisable to replace this with an appropriate, more sophisticated viscoelasticity formulation, e.g. the one derived for large deformations by Reese & Govindjee [38]. However, this is beyond the scope of this work, whose focus is supposed to be on the model order reduction of an existing full-order skeletal muscle model.

3.2 Elasticity tensor for skeletal muscle material

Having all necessary contributions to the 1st PIOLA-KIRCHHOFF stress tensor for a specific material response of skeletal muscle tissue at hand, the corresponding elasticity tensor, which is needed in the linearisation step, can now be derived. At this point the importance of keeping the Jacobian of the deformation gradient, J, in the stress tensor formulation, even in the case of an incompressible material with $J = 1$, becomes obvious. Without the additional dependency on J, many terms resulting from the partial derivatives with respect to J would be missing.
Making use of the derivatives introduced in Section 2.4, the derivations for each term are relatively straightforward, however, rather complex due to lengthy terms and nested partial derivatives. For that reason, the following summarises only the results. They are sorted by the occurring fourth-order tensor bases.

3.2.1 Incompressible isotropic contribution

Incompressible Neo-Hooke contribution

$$\begin{aligned}\frac{\partial \boldsymbol{P}_{\mathrm{E}}^{\mathrm{NH}}}{\partial \boldsymbol{F}} = {} & 2c_{10}J^{-\frac{2}{3}}\left(\boldsymbol{I}\otimes\boldsymbol{I}\right)^{23}_{T} - \frac{4}{3}c_{10}J^{-\frac{2}{3}}\left(\boldsymbol{F}\otimes\boldsymbol{F}^{-T}\right) - \frac{4}{3}c_{10}J^{-\frac{2}{3}}\left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}\right) \\ & + \frac{4}{9}c_{10}J^{-\frac{2}{3}}I_1\left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}^{-T}\right) + \frac{2}{3}c_{10}J^{-\frac{2}{3}}I_1\left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}^{-T}\right)^{24}_{T}\end{aligned} \tag{3.5}$$

Incompressible Demiray contribution

For readability, the exponential expression that occurs repeatedly, is abbreviated with $\exp[\cdots] := \exp\left[\frac{1}{2}c_2(J^{-\frac{2}{3}}I_1 - 3)\right]$.

$$\begin{aligned}\frac{\partial \boldsymbol{P}_{\mathrm{E}}^{\mathrm{D}}}{\partial \boldsymbol{F}} = {} & c_1 J^{-\frac{2}{3}} \exp[\cdots] (\boldsymbol{I}\otimes\boldsymbol{I})^{23}_{T} + c_1c_2 J^{-\frac{4}{3}} \exp[\cdots] (\boldsymbol{F}\otimes\boldsymbol{F}) \\ & - \left(\frac{1}{3}c_1c_2J^{-\frac{4}{3}}I_1 + \frac{2}{3}c_1J^{-\frac{2}{3}}\right)\exp[\cdots]\left(\boldsymbol{F}\otimes\boldsymbol{F}^{-T}\right) \\ & - \left(\frac{1}{3}c_1c_2J^{-\frac{4}{3}}I_1 + \frac{2}{3}c_1J^{-\frac{2}{3}}\right)\exp[\cdots]\left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}\right) \\ & + \left(\frac{1}{9}c_1c_2J^{-\frac{4}{3}}I_1^2 + \frac{2}{9}c_1J^{-\frac{2}{3}}I_1\right)\exp[\cdots]\left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}^{-T}\right) \\ & + \frac{1}{3}c_1J^{-\frac{2}{3}}I_1\exp[\cdots]\left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}^{-T}\right)^{24}_{T}\end{aligned} \tag{3.6}$$

Like previously for the stress tensor, for the elasticity tensor it holds as well that in the case of $c_2 = 0$ it is $\frac{\partial \boldsymbol{P}_{\mathrm{E}}^{\mathrm{D}}}{\partial \boldsymbol{F}} = \frac{\partial \boldsymbol{P}_{\mathrm{E}}^{\mathrm{NH}}}{\partial \boldsymbol{F}}$.

3.2.2 Anisotropic passive and active contributions

Holzapfel contribution

$$\begin{aligned}\frac{\partial \boldsymbol{P}_{\mathrm{E}}^{\mathrm{HA}}}{\partial \boldsymbol{F}} = {} & 2c_1(I_4-1)\exp\left(c_2(I_4-1)^2\right)(\boldsymbol{I}\otimes\boldsymbol{\mathcal{M}})^{23}_{T} \\ & + 4c_1\exp\left(c_2(I_4-1)^2\right)\left(2c_2(I_4-1)^2+1\right)(\boldsymbol{F}\boldsymbol{\mathcal{M}}\otimes\boldsymbol{F}\boldsymbol{\mathcal{M}})\end{aligned} \tag{3.7}$$

Active contribution

$$\frac{\partial \boldsymbol{P}_{\mathrm{E}}^{\mathrm{act}}}{\partial \boldsymbol{F}} = \alpha\cdot p_{\max}(\boldsymbol{I}\otimes\boldsymbol{\mathcal{M}})^{23}_{T} \tag{3.8}$$

3.2.3 Viscous contribution

Making use of the relation $\dot{\boldsymbol{F}} = \boldsymbol{L}\boldsymbol{F}$, where $\boldsymbol{L} := \nabla\dot{\boldsymbol{x}}$ is the so-called spatial velocity gradient, one obtains

$$\frac{\partial \boldsymbol{P}_{\mathrm{E}}^{\mathrm{visc}}}{\partial \boldsymbol{F}} = \frac{\partial\left(\eta\dot{\boldsymbol{F}}\right)}{\partial \boldsymbol{F}} = \eta\frac{\partial(\boldsymbol{L}\boldsymbol{F})}{\partial \boldsymbol{F}} = \eta(\boldsymbol{L}\otimes\boldsymbol{I})^{23}_{T}\,. \tag{3.9}$$

Remark:
Note that this part is only added for completeness at this point. Since during the discretisation process (c.f. Section 3.3) it is separated from the rest of the extra stress tensor and contributes to the linear term $\boldsymbol{D}\boldsymbol{u}'(t) = \boldsymbol{D}\boldsymbol{v}(t)$ of the problem, there is no need for a further derivative with respect to $\boldsymbol{F}$.

3.3 Discretisation and implementation

Evidently, for the derived sophisticated differential equations, no analytical solution is available and thus numerical solution methods need to be employed in order to obtain an approximate solution of the problem. For the spatial discretisation, the finite element method, which is commonly used for problems in (solid) mechanics, is applied. For detailed introductions into the theory of finite elements, the reader is referred to the textbooks of e.g. Braess [11], Brenner & Scott [12], Ern & Guermond [20] and Zohdi [52]. The discretisation in time is carried out using the simple implicit EULER scheme. Certainly, there are more advanced higher-order time integration schemes or index reduction methods for differential algebraic equations available. However, in order to access and investigate the overall solution procedure at any necessary step involved, a rather simple self-implemented scheme was decided on.
The purpose of this section is to provide the necessary insights into the specific properties of the full-order model. The first reason for a detailed derivation lies in the anticipated application of model order reduction methods. As those methods aim at preserving the structure and properties of the full-order system, one obviously has to be aware of those. The second reason is to provide something like a manual for future users or developers of the code. The model was implemented and solved within the MATLAB library KerMor [51], which provides routines for model order reduction of dynamical systems using subspace projection and nonlinear approximation. In order to understand and be able to modify and enhance the implementation, it will hopefully be beneficial to provide the details and a notation similar to that adopted for implementation.

3.3.1 Discretisation in space

The continuous formulation of the problem in its strong form was derived in the previous sections and can be summarised as

$$\begin{aligned}
&\text{Find } \boldsymbol{x}(\boldsymbol{X},t) \in \Omega \subset \mathbb{R}^3, p(\boldsymbol{X},t) \in \mathbb{R} \text{ such that } \forall\, \boldsymbol{X} \in \Omega_0 \subset \mathbb{R}^3 \\
&\rho_0(\boldsymbol{X}) \frac{\partial^2 \boldsymbol{x}(\boldsymbol{X},t)}{\partial t^2} = \nabla \cdot \boldsymbol{P}(\boldsymbol{x}(\boldsymbol{X},t), p(\boldsymbol{X},t), t) \\
&\text{subject to} \quad 0 = \det \boldsymbol{F}(\boldsymbol{x}(\boldsymbol{X},t),t) - 1 \qquad (3.10) \\
&\text{with} \quad \boldsymbol{P}(\boldsymbol{x}(\boldsymbol{X},t), p(\boldsymbol{X},t), t) = \boldsymbol{P}_E^{nl}(\boldsymbol{x}(\boldsymbol{X},t), p(\boldsymbol{X},t), t) + \boldsymbol{P}_E^{\text{visc}}(\boldsymbol{x}(\boldsymbol{X},t), t) \\
&\qquad\qquad - p(\boldsymbol{X},t)\, J(\boldsymbol{x}(\boldsymbol{X},t)) \boldsymbol{F}^{-T}(\boldsymbol{x}(\boldsymbol{X},t)) \,.
\end{aligned}$$

Note that the body forces term $\boldsymbol{b}(\boldsymbol{X},t)$ is dropped here (and from here on), as one can assume that those forces are negligibly small compared to the other forces that act on the muscle. To make the continuous problem accessible to numerical solution methods, its algebraic formulation is needed. To that purpose, as a first step, the continuous formulation in its weak form is derived. This step is often referred to as the method of weighted residuals and forms the starting point for the finite element method (as well as the boundary element method).

Continuous formulation in its weak form

Let $\mathcal{V}_{\boldsymbol{u}} := \mathcal{V}_u \times \mathcal{V}_u \times \mathcal{V}_u$ and $\mathcal{V}_p$ denote the position (or displacement) and pressure space, respectively. It is common (c.f. e.g. [5, 24, 34]) to assume that the domain of the internal elastic energy for the material under consideration is a subset of a SOBOLEV space. More precisely, the material formulations are defined and derived in such a way, that they fulfil certain criteria, e.g. polyconvexity (c.f. [3]), to make sure, that the solutions exist and lie in the correct spaces. Therefore, we set

$$\mathcal{V}_p := \mathcal{H}^0(\Omega_0) = \mathcal{L}^2(\Omega_0) := \{p : \Omega_0 \to \mathbb{R}, \boldsymbol{X} \mapsto p(\boldsymbol{X}) |\ \|p\|_{\mathcal{L}^2} < \infty\} \tag{3.11}$$

as the space of square-integrable functions on the reference domain Ω_0 and

$$\mathcal{V}_u := \mathcal{H}^1(\Omega_0) := \left\{u \in \mathcal{L}^2(\Omega_0) |\ \nabla u \in \mathcal{L}^2(\Omega_0)\right\} \tag{3.12}$$

as the space of square-integrable functions on Ω_0, whose first partial derivative is square-integrable as well.
$\mathcal{V}_p$ is equipped with the $\mathcal{L}^2$-norm, $\|p\|_{\mathcal{V}_p}$, which is induced by the inner product

$$\langle p, q\rangle_{\mathcal{V}_p} := \int_{\Omega_0} p(\boldsymbol{X})\, q(\boldsymbol{X})\, \mathrm{d}\boldsymbol{X}\,, \quad \Longrightarrow \quad \|p\|^2_{\mathcal{V}_p} := \langle p, p\rangle_{\mathcal{V}_p}\,, \text{for } p, q : \Omega_0 \to \mathbb{R}\,. \tag{3.13}$$

Moreover, $\mathcal{V}_u$ is equipped with the $\mathcal{H}^1$-norm, $\|u\|_{\mathcal{V}_u}$, which is induced by the inner product

$$\langle u, v\rangle_{\mathcal{V}_u} := \int_{\Omega_0} u(\boldsymbol{X})\, v(\boldsymbol{X})\, \mathrm{d}\boldsymbol{X} + \int_{\Omega_0} \nabla u(\boldsymbol{X}) \cdot \nabla v(\boldsymbol{X})\, \mathrm{d}\boldsymbol{X}\,, \tag{3.14}$$

$$\Longrightarrow \quad \|u\|^2_{\mathcal{V}_u} := \langle u, u\rangle_{\mathcal{V}_u}\,, \text{for } u, v : \Omega_0 \to \mathbb{R}\,. \tag{3.15}$$

With these inner product spaces at hand, the continuous formulation of problem (3.10) in its weak form is obtained by multiplication with suitable test functions.

$$\text{Find } (\boldsymbol{x}, p) \in \mathcal{V}_{\boldsymbol{u}} \times \mathcal{V}_p \text{ such that } \forall\, (\hat{\boldsymbol{x}}, \hat{p}) \in \mathcal{V}_{\boldsymbol{u}} \times \mathcal{V}_p \tag{3.16}$$

$$\int_{\Omega_0} \rho_0(\boldsymbol{X}) \frac{\partial^2 \boldsymbol{x}(\boldsymbol{X}, t)}{\partial t^2} \cdot \hat{\boldsymbol{x}}(\boldsymbol{X})\, \mathrm{d}\boldsymbol{X} = \int_{\Omega_0} \left[\nabla \cdot \boldsymbol{P}\left(\boldsymbol{x}(\boldsymbol{X}, t), p(\boldsymbol{X}, t), t\right)\right] \cdot \hat{\boldsymbol{x}}(\boldsymbol{X})\, \mathrm{d}\boldsymbol{X}$$

$$\text{s.t.} \int_{\Omega_0} \left[\det \boldsymbol{F}(\boldsymbol{x}(\boldsymbol{X}, t)) - 1\right] \hat{p}(\boldsymbol{X})\, \mathrm{d}\boldsymbol{X} = 0\,.$$

To further transform Equation (3.16) into the common form, the product rule of differentiation (c.f. Equation (3.17)) and the GAUSS integral theorem (c.f. Equation (3.18)) are applied. As those are not everywhere in literature consistently given or named and as the notation varies a lot, they are introduced here in correspondence with the adopted notation:

$$\nabla \cdot \left(\boldsymbol{P}^T \hat{\boldsymbol{x}}\right) = (\nabla \cdot \boldsymbol{P}) \cdot \hat{\boldsymbol{x}} + \boldsymbol{P} \cdot \nabla \hat{\boldsymbol{x}} \Leftrightarrow (\nabla \cdot \boldsymbol{P}) \cdot \hat{\boldsymbol{x}} = -\boldsymbol{P} \cdot \nabla \hat{\boldsymbol{x}} + \nabla \cdot \left(\boldsymbol{P}^T \hat{\boldsymbol{x}}\right) \tag{3.17}$$

$$\int_{\Omega_0} \nabla \cdot \left(\boldsymbol{P}^T \hat{\boldsymbol{x}}\right) \mathrm{d}\boldsymbol{X} = \int_{\partial\Omega_0} \boldsymbol{P}^T \hat{\boldsymbol{x}} \cdot \mathrm{d}\boldsymbol{A} = \int_{\partial\Omega_0} \boldsymbol{P}^T \hat{\boldsymbol{x}} \cdot \boldsymbol{N} \, \mathrm{d}A = \int_{\partial\Omega_0} \boldsymbol{P}\boldsymbol{N} \cdot \hat{\boldsymbol{x}} \, \mathrm{d}A \tag{3.18}$$

Note that the (surface) integral (3.18) vanishes for functions $\hat{\boldsymbol{x}}|_{\partial\Omega_0} = \boldsymbol{0}$. Therefore, dividing the surface $\partial\Omega_0 := \partial\Omega_0^{\mathrm{D}} \cup \partial\Omega_0^{\mathrm{N}}$ into disjunct DIRICHLET and NEUMANN boundary parts, defining a shifted subspace of test functions that fulfil homogeneous DIRICHLET boundary conditions

$$\mathcal{T}_{\boldsymbol{u}} := \left\{ \hat{\boldsymbol{x}} \in \mathcal{V}_{\boldsymbol{u}} \,|\, \hat{\boldsymbol{x}} = \boldsymbol{0} \text{ on } \partial\Omega_0^{\mathrm{D}} \right\} \subset \mathcal{V}_{\boldsymbol{u}} \,, \tag{3.19}$$

and omitting any NEUMANN boundary conditions at this point, the weak formulation of problem (3.10) writes without loss of generality:

$$\begin{aligned} &\text{Find } (\boldsymbol{x}, p) \in \mathcal{V}_{\boldsymbol{u}} \times \mathcal{V}_p \text{ such that } \forall\, (\hat{\boldsymbol{x}}, \hat{p}) \in \mathcal{T}_{\boldsymbol{u}} \times \mathcal{V}_p \\ &\qquad \int_{\Omega_0} \rho_0(\boldsymbol{X}) \frac{\partial^2 \boldsymbol{x}(\boldsymbol{X}, t)}{\partial t^2} \cdot \hat{\boldsymbol{x}}(\boldsymbol{X}) \, \mathrm{d}\boldsymbol{X} = -\int_{\Omega_0} \boldsymbol{P}\left(\boldsymbol{x}(\boldsymbol{X}, t), p(\boldsymbol{X}, t), t\right) \cdot \nabla \hat{\boldsymbol{x}}(\boldsymbol{X}) \, \mathrm{d}\boldsymbol{X} \\ &\text{s.t.} \quad \int_{\Omega_0} \left[\det \boldsymbol{F}(\boldsymbol{x}(\boldsymbol{X}, t)) - 1\right] \hat{p}(\boldsymbol{X}) \, \mathrm{d}\boldsymbol{X} = 0 \,. \end{aligned} \tag{3.20}$$

Algebraic formulation

To formulate the finite element approximation of problem (3.20), two finite-dimensional subspaces, dependent on the computational mesh size $h > 0$, are introduced.

$$\mathcal{V}_p^h \subset \mathcal{V}_p \text{ of dimension } N_p \in \mathbb{N} \text{ with orthonormal basis } \{\psi_i(\boldsymbol{X})\}_{i=1}^{N_p} \tag{3.21}$$

$$\mathcal{V}_u^h \subset \mathcal{V}_u \text{ of dimension } N \in \mathbb{N} \text{ with orthonormal basis } \{\varphi_i(\boldsymbol{X})\}_{i=1}^{N} \tag{3.22}$$

Note that the finite element approximation subspaces inherit the inner products and norms of the exact spaces.
Furthermore, two bijections between $\mathbb{R}^{N_p}$ and $\mathcal{V}_p^h$ for the approximate pressure $p^h \in \mathcal{V}_p^h$ and $\mathbb{R}^N$ and $\mathcal{V}_u^h$ for the approximate position $\boldsymbol{x}^h \in \mathcal{V}_{\boldsymbol{u}}^h$ are defined for algebraic purposes:

$$p^h(\boldsymbol{X}, t) = \sum_{i=1}^{N_p} w_i(t)\psi_i(\boldsymbol{X}) = \boldsymbol{w}\boldsymbol{\psi} \qquad \text{with } w_i : [0, T] \to \mathbb{R} \,, \tag{3.23}$$

$$\begin{aligned} \boldsymbol{x}^h(\boldsymbol{X}, t) &= \sum_{i=1}^{N} \boldsymbol{u}[i](t) \odot \boldsymbol{\varphi}[i](\boldsymbol{X}) && \text{with } \boldsymbol{u}[i] : [0, T] \to \mathbb{R}^3 \\ &= \sum_{i=1}^{N} \boldsymbol{u}[i](t)\varphi_i(\boldsymbol{X}) = \boldsymbol{u}\boldsymbol{\varphi} && \text{as } \mathcal{V}_{\boldsymbol{u}}^h = \mathcal{V}_u^h \times \mathcal{V}_u^h \times \mathcal{V}_u^h \,. \end{aligned} \tag{3.24}$$

Therein, $\boldsymbol{u} \in \mathbb{R}^{3\times N}$ and $\boldsymbol{w} \in \mathbb{R}^{1\times N_p}$ are the coefficient vectors for the approximate position and pressure, while $\boldsymbol{\varphi} := (\varphi_i(\boldsymbol{X}))_{i=1}^{N} \in \mathbb{R}^{N\times 1}$, $\boldsymbol{\psi} := (\psi_i(\boldsymbol{X}))_{i=1}^{N_p} \in \mathbb{R}^{N_p\times 1}$ contain the

corresponding evaluations of basis functions respectively. In order to account for the three-dimensionality of the problem, i.e. the x-, y- and z-direction, a triple index indicated by square brackets

$$\boldsymbol{\varphi}[i] := \begin{pmatrix} \varphi_i \\ \varphi_i \\ \varphi_i \end{pmatrix}, \boldsymbol{u}[i] := \begin{pmatrix} u[i]_1 \\ u[i]_2 \\ u[i]_3 \end{pmatrix} = \begin{pmatrix} u_{3(i-1)+1} \\ u_{3(i-1)+2} \\ u_{3(i-1)+3} \end{pmatrix} = u[i]_n \boldsymbol{e}_n \in \mathbb{R}^3, \forall\, i = 1, ..., N,$$

and the HADAMARD or SCHUR product, indicated by $\odot$, meaning a componentwise multiplication of the respective vectors, were introduced. Note that by introducing the triple index, the summation always runs from $i = 1, ..., N$ instead of $i = 1, ..., 3N$ and therefore one deals with 3×1 vectors and matrix-vector-multiplications instead of scalars and scalar products of two matrices in the proceeding derivations. This is feasible, as (naturally) the same basis $\{\varphi_i(\boldsymbol{X})\}_{i=1}^N$ is chosen for the x-, y- and z- components of the approximate position $\boldsymbol{x}^h \in \mathcal{V}_{\boldsymbol{u}}^h$.
With Equation (3.24) and the definition *

$$\boldsymbol{\varphi}_i^{\nabla} := \nabla \varphi_i = \begin{pmatrix} \frac{\partial \varphi_i}{\partial X_1} \\ \frac{\partial \varphi_i}{\partial X_2} \\ \frac{\partial \varphi_i}{\partial X_3} \end{pmatrix} =: \begin{pmatrix} \varphi_i^{\nabla_1} \\ \varphi_i^{\nabla_2} \\ \varphi_i^{\nabla_3} \end{pmatrix} = \varphi_i^{\nabla_o} \boldsymbol{e}_o \in \mathbb{R}^3, \forall\, i = 1, ..., N, \tag{3.25}$$

one obtains the discrete representation of the deformation gradient as

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{x}^h(\boldsymbol{X}, t)) = \boldsymbol{F}(\boldsymbol{u}(t)) &= \sum_{i=1}^N \boldsymbol{u}[i] \otimes \boldsymbol{\varphi}_i^{\nabla} = \sum_{i=1}^N \boldsymbol{u}[i] \left(\boldsymbol{\varphi}_i^{\nabla}\right)^T \\ &= \sum_{i=1}^N u[i]_n \varphi_i^{\nabla_o} (\boldsymbol{e}_n \otimes \boldsymbol{e}_o) =: \sum_{i=1}^N f_{no}^i (\boldsymbol{e}_n \otimes \boldsymbol{e}_o). \end{aligned} \tag{3.26}$$

Using a GALERKIN projection approach the resulting algebraic formulation reads

Find $(\boldsymbol{x}^h, p^h) \in \mathcal{V}_{\boldsymbol{u}}^h \times \mathcal{V}_p^h$ such that

$$\begin{aligned} \int_{\Omega_0} \rho_0(\boldsymbol{X}) \frac{\partial^2 \boldsymbol{x}^h(\boldsymbol{X}, t)}{\partial t^2} \varphi_k(\boldsymbol{X}) \, \mathrm{d}\boldsymbol{X} = \\ - \int_{\Omega_0} \boldsymbol{P}\left(\boldsymbol{x}^h(\boldsymbol{X}, t), p^h(\boldsymbol{X}, t), t\right) \boldsymbol{\varphi}_k^{\nabla}(\boldsymbol{X}) \, \mathrm{d}\boldsymbol{X} \quad \forall\, k = 1, ..., N \end{aligned} \tag{3.27}$$

$$\text{s.t.} \int_{\Omega_0} \left[\det \boldsymbol{F}(\boldsymbol{x}^h(\boldsymbol{X}, t)) - 1\right] \psi_k(\boldsymbol{X}) \, \mathrm{d}\boldsymbol{X} = 0 \qquad \forall\, k = 1, ..., N_p. \tag{3.28}$$

*Note that for the index o EINSTEIN summation convention is used.

Finite element matrix formulation

This paragraph is kept as short as possible and as detailed as necessary. As the assembly of the discrete system with all its integrands on the element level is important for the work with the software environment and also might be important to future work for the approximation of nonlinear contributions, these shall briefly be derived at this point. Note that as a result of this approach, the notation is not exact, but the reader that is familiar with the finite element method, should be capable of following and understanding the necessary details.
To obtain a finite element formulation, the (reference) domain Ω_0 is now partitioned into $N_e \in \mathbb{N}$ smaller parts, the finite elements, i.e.

$$\Omega_0 = \bigcup_{e=1}^{N_e} \Omega^e \,. \tag{3.29}$$

Moreover, nodes are defined on the corners (and certain positions on the edges) of the elements (here: in total N nodes for the position and N_p nodes for the pressure) and element numbers as well as global and local node numbers ($N_b \in \mathbb{N}$ nodes per element) are allocated. This construction is referred to as the finite element mesh. Each node of the finite element mesh is assigned to a basis function (here: φ_i for the position and ψ_i for the pressure), which is defined on a referential element and have a local support only. This property later yields the sparse matrix structure of the system. Because of this approach, each sum from $i = 1, ..., N\,(N_p)$ can be replaced by two sums over $e = 1, ..., N_e$ and $n = 1, ..., N_b$ without accounting twice for any contributions.
For the so-called mixed finite elements, where two or more fields need to be approximated, the basis functions have to fulfil certain conditions in order to obtain a stable finite element formulation. In the case at hand with position and pressure field, this condition is the so-called inf-sup condition. One choice to satisfy the discrete inf-sup condition (for more details, see e.g. [5], [45]) are the hexahedral Taylor-Hood ($\mathbb{Q}_2^{(27)} - \mathbb{Q}_1$) elements, i.e. triquadratic shape functions for the position field ($N_b := 27$) and linear shape functions for the pressure field ($N_b := 8$).
As a last step in the discretisation procedure, the continuous integrals need to be approximated by Gaussian quadrature with a certain number of Gauss points $N_g \in \mathbb{N}$ (here: $N_g := 3^3 = 27$).
Summing up the previous paragraph and introducing the notation that is meant in the following, we replace

$$\int_{\Omega_0} \sum_{i=1}^{N} I_i^k \overset{\text{FE}}{\Longrightarrow} \sum_{e=1}^{N_e} \int_{\Omega^e} \sum_{i=1}^{N_b} \omega_e I_{e,i}^k \overset{\text{GQ}}{\Longrightarrow} \sum_{e=1}^{N_e} \sum_{g=1}^{N_g} \sum_{i=1}^{N_b} \omega_{eg} I_{eg,i}^k \implies \sum_{p} \omega_p I_i^k \,.$$

Therein, I_i^k means an integrand depending on the basis and test functions $\varphi_{i/k}\,(\psi_{i/k})$, ω_α, $\alpha \in \{e, eg\}$ is a weighting factor containing the weighting factors that result from the finite element transformation between local (referential) and global coordinate system (keyword: isoparametric concept) and the Gauss quadrature weights. The integrands $I_{\alpha,i}^k$, $\alpha \in \{e, eg\}$ mean, that the integrand is evaluated on the appropriate node with the appropriate shape function at the appropriate Gauss point and so on. Instead of

always writing these three sums, they are replaced by one summation over p for further simplification.

Finally, the weak problem in its algebraic formulation (3.27)– (3.28) can be written down in its finite element matrix form. Therefore, the position and pressure coefficient vectors $\boldsymbol{u}(t) \in \mathbb{R}^{3N}$ and $\boldsymbol{w}(t) \in \mathbb{R}^{N_p}$ are introduced. The left hand side of Equation (3.27) becomes the mass matrix times acceleration part $\boldsymbol{M}\boldsymbol{u}''(t)$, where

$$\boldsymbol{M}[[i][k]] := \begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix} \rho_0 \sum_p \omega_p \, \varphi_i \varphi_k \; \in \mathbb{R}^{3x3} \quad \forall \, i,k = 1,...,N \, . \tag{3.30}$$

The right-hand side of Equation (3.27) is split into three discrete parts according to the split of the 1st Piola-Kirchhoff stress tensor (c.f. Equation (3.10)): The linear viscous damping contribution $\boldsymbol{D}\boldsymbol{u}'(t)$, where

$$\boldsymbol{D}[[i][k]] := \begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix} \eta \sum_p \omega_p \, \boldsymbol{\varphi}_i^{\nabla} \cdot \boldsymbol{\varphi}_k^{\nabla} \; \in \mathbb{R}^{3x3} \quad \forall \, i,k = 1,...,N \, , \tag{3.31}$$

the contribution of the nonlinear deviatoric extra stress

$$\boldsymbol{K}[k](\boldsymbol{u}(t)) := \sum_p \omega_p \boldsymbol{P}_E^{nl} \, \boldsymbol{\varphi}_k^{\nabla} \; \in \mathbb{R}^{3x1} \quad \forall \, k = 1,...,N \, , \tag{3.32}$$

and the volumetric contribution $\boldsymbol{A}(\boldsymbol{u}(t))\, \boldsymbol{w}(t)$, which depends linearly on the pressure coefficients and nonlinearly on the position coefficients. The matrix $\boldsymbol{A}(\boldsymbol{u}(t)) \in \mathbb{R}^{3N \times N_p}$ is defined as

$$\boldsymbol{A}[[k],i](\boldsymbol{u}(t)) := \sum_p \omega_p \psi_i (\det \boldsymbol{F}) \boldsymbol{F}^{-T} \boldsymbol{\varphi}_k^{\nabla} \; \in \mathbb{R}^{3x1} \quad \forall \, k = 1,...,N \, , \forall \, i = 1,...,N_p \, . \tag{3.33}$$

The incompressibility constraint, c.f. Equation (3.28), is nonlinear in the position coefficients and its discretised components read

$$\boldsymbol{G}_k(\boldsymbol{u}(t)) := \sum_p \omega_p (\det \boldsymbol{F} - 1) \psi_k \; \in \mathbb{R} \quad \forall \, k = 1,...,N_p \, . \tag{3.34}$$

With these definitions, one finally arrives at the finite-dimensional problem in matrix form of problem (3.10), which is a problem to be solved for the coefficients in $\mathbb{R}$.

$$\begin{aligned} &\forall t \in [0,T] \text{ find } (\boldsymbol{u}(t), \boldsymbol{w}(t)) \in \mathbb{R}^{3N} \times \mathbb{R}^{N_p}, \text{ such that} \\ &\quad \boldsymbol{M}\, \boldsymbol{u}''(t) = -\left[\boldsymbol{D}\, \boldsymbol{u}'(t) + \boldsymbol{K}(\boldsymbol{u}(t)) - \boldsymbol{A}(\boldsymbol{u}(t))\, \boldsymbol{w}(t)\right] \\ &\quad \text{s.t. } \boldsymbol{0} = \boldsymbol{G}(\boldsymbol{u}(t)) \, . \end{aligned} \tag{3.35}$$

Introducing a discrete velocity coefficient vector $\boldsymbol{v}(t) := \boldsymbol{u}'(t) \in \mathbb{R}^{3N}$, this second-order system can be further transformed into a first-order system:

$$\begin{aligned} &\forall t \in [0,T] \text{ find } (\boldsymbol{u}(t), \boldsymbol{v}(t), \boldsymbol{w}(t)) \in \mathbb{R}^{3N} \times \mathbb{R}^{3N} \times \mathbb{R}^{N_p}, \text{ such that} \\ &\qquad \boldsymbol{u}'(t) = \boldsymbol{v}(t) \\ &\boldsymbol{M}\,\boldsymbol{v}'(t) = -\boldsymbol{D}\,\boldsymbol{v}(t) - \boldsymbol{K}(\boldsymbol{u}(t)) + \boldsymbol{A}(\boldsymbol{u}(t))\,\boldsymbol{w}(t) \\ &\qquad \text{s.t. } \boldsymbol{0} = \boldsymbol{G}(\boldsymbol{u}(t))\,. \end{aligned} \tag{3.36}$$

In a block matrix formulation one can assemble the whole system of full dimension $\bar{d} := 2 \times 3N + N_p$ (dependency on t dropped for improved readability) as

$$\underbrace{\begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{M} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix}}_{=:\bar{\boldsymbol{M}} \in \mathbb{R}^{\bar{d}\times\bar{d}}} \underbrace{\begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ \boldsymbol{w} \end{pmatrix}'}_{=:\bar{\boldsymbol{x}}' \in \mathbb{R}^{\bar{d}}} = \underbrace{\begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{D} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix}}_{=:\bar{\boldsymbol{D}} \in \mathbb{R}^{\bar{d}\times\bar{d}}} \underbrace{\begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ \boldsymbol{w} \end{pmatrix}}_{=:\bar{\boldsymbol{x}}} + \underbrace{\begin{pmatrix} \boldsymbol{v} \\ -\boldsymbol{K}(\boldsymbol{u}) + \boldsymbol{A}(\boldsymbol{u})\,\boldsymbol{w} \\ \boldsymbol{G}(\boldsymbol{u}) \end{pmatrix}}_{=:\bar{\boldsymbol{K}}(\bar{\boldsymbol{x}}) \in \mathbb{R}^{\bar{d}\times\bar{d}}}. \tag{3.37}$$

This equation is the starting point for the model order reduction and in this context, it is referred to as the full-order model.

Discrete inner products

As already mentioned, the discrete finite element approximation spaces $\mathcal{V}_p^h, \mathcal{V}_{\boldsymbol{u}}^h$ inherit the inner products and norms of the exact spaces $\mathcal{V}_p, \mathcal{V}_{\boldsymbol{u}}$ respectively. Therefore, we introduce the equivalent discrete inner products as *

$$\forall\, \boldsymbol{p}, \boldsymbol{q} \in \mathbb{R}^{N_p} : \langle \boldsymbol{p}, \boldsymbol{q} \rangle_{\boldsymbol{H}_p} := \langle \boldsymbol{H}_p\,\boldsymbol{p}, \boldsymbol{q} \rangle_2 \quad \text{where } (\boldsymbol{H}_p)_{ij} := \langle \psi_i, \psi_j \rangle_{\mathcal{V}_p}\,, \tag{3.38}$$

$$\forall\, \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{3N} : \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\boldsymbol{H}_{\boldsymbol{u}}} := \langle \boldsymbol{H}_{\boldsymbol{u}}\,\boldsymbol{u}, \boldsymbol{v} \rangle_2 \quad \text{where } \boldsymbol{H}_{\boldsymbol{u}}\,[[i][j]] := \langle \boldsymbol{\varphi}[i], \boldsymbol{\varphi}[j] \rangle_{\mathcal{V}_{\boldsymbol{u}}}\,. \tag{3.39}$$

The matrices $\boldsymbol{H}_p \in \mathbb{R}^{N_p \times N_p}$, $\boldsymbol{H}_{\boldsymbol{u}} \in \mathbb{R}^{3N\times 3N}$ are often referred to as mass matrices. However, in this work this might lead to confusion with the mass matrix $\boldsymbol{M}$ resulting from the dynamics of the problem, therefore they will be called inner product matrices. Making use of the definitions (3.13) and (3.14) of the inner products and the derived finite element forms of the mass matrix (3.30) and the damping matrix (3.31), one can directly see the equivalence for the assembly of the inner product matrices, whose components are given as

$$(\boldsymbol{H}_p)_{ij} := \sum_p \omega_p\, \psi_i \psi_j \;\in \mathbb{R} \quad \forall\, i,j = 1, ..., N_p\,. \tag{3.40}$$

$$\boldsymbol{H}_{\boldsymbol{u}}\,[[i][j]] := \begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix} \left(\sum_p \omega_p\, \varphi_i \varphi_j + \sum_p \omega_p\, \boldsymbol{\varphi}_i^{\nabla} \cdot \boldsymbol{\varphi}_j^{\nabla} \right) \in \mathbb{R}^{3\times 3} \quad \forall\, i,j = 1, ..., N\,.$$

Note that $\boldsymbol{H}_{\boldsymbol{u}} = \rho_0^{-1}\boldsymbol{M} + \eta^{-1}\boldsymbol{D}$.

*Note that the square brackets indicate the triple index and thus Equation (3.39) means a component wise application of the inner product defined in Equation (3.14).

Lastly, the overall inner product matrix for the first-order system is assembled in a block form

$$\bar{\boldsymbol{H}} := \begin{pmatrix} \boldsymbol{H_u} & & \\ & \boldsymbol{H_u} & \\ & & \boldsymbol{H}_p \end{pmatrix} \in \mathbb{R}^{\bar{d}\times\bar{d}}. \tag{3.41}$$

3.3.2 Discretisation in time

As a last step, the first-order system of the form

$$\bar{\boldsymbol{M}}\bar{\boldsymbol{x}}'(t) = \bar{\boldsymbol{D}}\bar{\boldsymbol{x}}(t) + \bar{\boldsymbol{K}}(\bar{\boldsymbol{x}}(t)) =: \boldsymbol{g}(\bar{\boldsymbol{x}}(t)) \quad \Longleftrightarrow \quad \boldsymbol{0} = -\bar{\boldsymbol{M}}\bar{\boldsymbol{x}}'(t) + \boldsymbol{g}(\bar{\boldsymbol{x}}(t)), \tag{3.42}$$

with initial condition

$$\bar{\boldsymbol{x}}(0) = \begin{pmatrix} \boldsymbol{u}(0) \\ \boldsymbol{v}(0) \\ \boldsymbol{w}(0) \end{pmatrix} = \begin{pmatrix} \boldsymbol{u}_0 \\ \boldsymbol{v}_0 \\ \boldsymbol{0} \end{pmatrix} =: \bar{\boldsymbol{x}}_0 \in \mathbb{R}^{\bar{d}}, \tag{3.43}$$

needs to be discretised in time. For implementation purposes, a right-hand side function `odefun` $\boldsymbol{g}$ was defined as

$$\boldsymbol{g} : \mathbb{R}^{\bar{d}} \to \mathbb{R}^{\bar{d}}, \bar{\boldsymbol{x}} \mapsto \boldsymbol{g}(\bar{\boldsymbol{x}}) := \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{D} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ \boldsymbol{w} \end{pmatrix} + \begin{pmatrix} \boldsymbol{v} \\ -\boldsymbol{K}(\boldsymbol{u}) + \boldsymbol{A}(\boldsymbol{u})\,\boldsymbol{w} \\ \boldsymbol{G}(\boldsymbol{u}) \end{pmatrix}. \tag{3.44}$$

For stability reasons, an implicit time discretisation of Equation (3.42) is chosen, here the implicit EULER scheme. For a time step size $\mathrm{d}t \in \mathbb{R}$ and discrete time instants $t_i := i \cdot \mathrm{d}t$, $i \in \mathbb{N}_0$, define the corresponding state $\bar{\boldsymbol{x}}_i := \bar{\boldsymbol{x}}(t_i)$, $i \in \mathbb{N}_0$. Then, the derivative can be approximated by the difference quotient $\bar{\boldsymbol{x}}'(t) = \frac{1}{\mathrm{d}t}(\bar{\boldsymbol{x}}_{i+1} - \bar{\boldsymbol{x}}_i)$, and inserted into Equation (3.42) to obtain the residuum formulation or implicit function

$$\boldsymbol{0} = \boldsymbol{f}(\bar{\boldsymbol{x}}_{i+1}) := -\bar{\boldsymbol{M}}(\bar{\boldsymbol{x}}_{i+1} - \bar{\boldsymbol{x}}_i) + \mathrm{d}t\,\boldsymbol{g}(\bar{\boldsymbol{x}}_{i+1}). \tag{3.45}$$

This nonlinear equation can be solved in each time step by performing a NEWTON iteration. Assuming the i^{th} state is known, one sets the NEWTON initial guess and compute the k^{th} NEWTON update

$$\bar{\boldsymbol{x}}_{i+1}^{(0)} := \bar{\boldsymbol{x}}_i, \tag{3.46}$$

$$\bar{\boldsymbol{x}}_{i+1}^{(k+1)} = \bar{\boldsymbol{x}}_{i+1}^{(k)} - \boldsymbol{J}\boldsymbol{f}(\bar{\boldsymbol{x}}_{i+1}^{(k)})^{-1}\boldsymbol{f}(\bar{\boldsymbol{x}}_{i+1}^{(k)}), \tag{3.47}$$

by solving the following linear system of equations

$$\boldsymbol{J}\boldsymbol{f}(\bar{\boldsymbol{x}}_{i+1}^{(k)}) \cdot \Delta\bar{\boldsymbol{x}}_{i+1}^{(k)} = \boldsymbol{f}(\bar{\boldsymbol{x}}_{i+1}^{(k)}) \tag{3.48}$$

for the NEWTON increment $\Delta\bar{\boldsymbol{x}}_{i+1}^{(k)} := \bar{\boldsymbol{x}}_{i+1}^{(k)} - \bar{\boldsymbol{x}}_{i+1}^{(k+1)}$.

Therein, one can provide the analytical Jacobian, which has the structure

$$\boldsymbol{Jf}(\bar{\boldsymbol{x}}) := \frac{\partial \boldsymbol{f}(\bar{\boldsymbol{x}})}{\partial \bar{\boldsymbol{x}}} = -\bar{\boldsymbol{M}} + \mathrm{d}t\, \boldsymbol{Jg}(\bar{\boldsymbol{x}}) \tag{3.49}$$

$$= \left(\begin{array}{cc|c} -\boldsymbol{I} & \mathrm{d}t\, \boldsymbol{I} & \boldsymbol{0} \\ \mathrm{d}t \left[-\frac{\partial \boldsymbol{K}(\boldsymbol{u})}{\partial \boldsymbol{u}} + \frac{\partial \boldsymbol{A}(\boldsymbol{u})\boldsymbol{w}}{\partial \boldsymbol{u}}\right] & -\boldsymbol{M} - \mathrm{d}t\, \boldsymbol{D} & \mathrm{d}t\, \boldsymbol{A}(\boldsymbol{u}) \\ \hline \mathrm{d}t\, \boldsymbol{A}(\boldsymbol{u})^T & \boldsymbol{0} & \boldsymbol{0} \end{array}\right) \in \mathbb{R}^{\bar{d}\times\bar{d}}, \tag{3.50}$$

as it holds that $\frac{\partial \boldsymbol{G}(\boldsymbol{u})}{\partial \boldsymbol{u}} = \boldsymbol{A}(\boldsymbol{u})^T \in \mathbb{R}^{N_p \times 3N}$ (c.f. next section, Equation (3.51)).

3.3.3 Derivation of Jacobian blocks

In Section 3.3.1 the discrete expressions for the implicit function $\boldsymbol{f}$ were derived. For the linearised problem to be solved within each NEWTON iteration, one needs to provide the derivatives of the discrete operators contained in the Jacobian (c.f. Equation (3.50)), i.e. $\frac{\partial \boldsymbol{G}(\boldsymbol{u})}{\partial \boldsymbol{u}}$, $\frac{\partial \boldsymbol{K}(\boldsymbol{u})}{\partial \boldsymbol{u}}$ and $\frac{\partial \boldsymbol{A}(\boldsymbol{u})\boldsymbol{w}}{\partial \boldsymbol{u}}$.

$$\frac{\partial \boldsymbol{G}(\boldsymbol{u})}{\partial \boldsymbol{u}} = \begin{pmatrix} \frac{\partial \boldsymbol{G}_1(\boldsymbol{u})}{\partial \boldsymbol{u}[1]} & \cdots & \frac{\partial \boldsymbol{G}_1(\boldsymbol{u})}{\partial \boldsymbol{u}[N]} \\ \vdots & \ddots & \vdots \\ \frac{\partial \boldsymbol{G}_{N_p}(\boldsymbol{u})}{\partial \boldsymbol{u}[1]} & \cdots & \frac{\partial \boldsymbol{G}_{N_p}(\boldsymbol{u})}{\partial \boldsymbol{u}[N]} \end{pmatrix} \in \mathbb{R}^{N_p \times 3N}, \text{ where } \forall\, k = 1, ...N_p, \forall\, j = 1, ..., N$$

$$\frac{\partial \boldsymbol{G}_k(\boldsymbol{u})}{\partial \boldsymbol{u}[j]} = \sum_p \omega_p \left(\boldsymbol{\varphi}_j^{\nabla}\right)^T (\det \boldsymbol{F}) \boldsymbol{F}^{-T} \psi_k \overset{(3.33)}{=} \boldsymbol{A}[[j], k](\boldsymbol{u})^T \in \mathbb{R}^{1\times 3} \tag{3.51}$$

Therein, use was made of the derivative

$$\begin{aligned} \frac{\partial \boldsymbol{F}(\boldsymbol{u})}{\partial \boldsymbol{u}[j]} &\overset{(3.26)}{=} \frac{\partial}{\partial \boldsymbol{u}[j]} \left(\sum_{i=1}^{N} f_{no}^i \left(\boldsymbol{e}_n \otimes \boldsymbol{e}_o\right)\right) = \sum_{i=1}^{N} \frac{\partial u[i]_n \varphi_i^{\nabla_o}}{\partial u[j]_k} \left(\boldsymbol{e}_n \otimes \boldsymbol{e}_o \otimes \boldsymbol{e}_k\right) \\ &= \frac{\partial u[j]_n \varphi_j^{\nabla_o}}{\partial u[j]_k} \left(\boldsymbol{e}_n \otimes \boldsymbol{e}_o \otimes \boldsymbol{e}_k\right) = \delta_{nk} \varphi_j^{\nabla_o} \left(\boldsymbol{e}_n \otimes \boldsymbol{e}_o \otimes \boldsymbol{e}_k\right) \\ &= \varphi_j^{\nabla_o} \left(\boldsymbol{e}_k \otimes \boldsymbol{e}_o \otimes \boldsymbol{e}_k\right) = \varphi_j^{\nabla_o} \delta_{io} \left(\boldsymbol{e}_k \otimes \boldsymbol{e}_i \otimes \boldsymbol{e}_k\right) \\ &= \left(\boldsymbol{e}_k \otimes \boldsymbol{e}_i \otimes \boldsymbol{e}_k \otimes \boldsymbol{e}_i\right) \varphi_j^{\nabla_o} \boldsymbol{e}_o = \left(\boldsymbol{I} \otimes \boldsymbol{I}\right)^{\overset{23}{T}} \boldsymbol{\varphi}_j^{\nabla}. \end{aligned} \tag{3.52}$$

For the discrete representation of the elasticity tensor, one obtains

$$\frac{\partial \boldsymbol{K}(\boldsymbol{u})}{\partial \boldsymbol{u}} = \begin{pmatrix} \frac{\partial \boldsymbol{K}[1](\boldsymbol{u})}{\partial \boldsymbol{u}[1]} & \cdots & \frac{\partial \boldsymbol{K}[1](\boldsymbol{u})}{\partial \boldsymbol{u}[N]} \\ \vdots & \ddots & \vdots \\ \frac{\partial \boldsymbol{K}[N](\boldsymbol{u})}{\partial \boldsymbol{u}[1]} & \cdots & \frac{\partial \boldsymbol{K}[N](\boldsymbol{u})}{\partial \boldsymbol{u}[N]} \end{pmatrix} \in \mathbb{R}^{3N \times 3N}, \text{ where } \forall\, k, j = 1, ...N$$

$$\frac{\partial \boldsymbol{K}[k](\boldsymbol{u})}{\partial \boldsymbol{u}[j]} = \sum_p \omega_p \frac{\partial}{\partial \boldsymbol{u}[j]} \left[\boldsymbol{P}_E^{nl}(\boldsymbol{u}(t)) \boldsymbol{\varphi}_k^{\nabla}\right] \in \mathbb{R}^{3\times 3}. \tag{3.53}$$

The computation of the integrand works as follows (omitting the indices and dependency on t for readability):

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{u}[j]}\left[\boldsymbol{P}_E^{nl}(\boldsymbol{u})\boldsymbol{\varphi}_k^\nabla\right] &\overset{(2.58)}{=} \left(\frac{\partial \boldsymbol{P}(\boldsymbol{u})}{\partial \boldsymbol{u}[j]}\right)^{\overset{23}{T}} \boldsymbol{\varphi}_k^\nabla = \left(\frac{\partial \boldsymbol{P}(\boldsymbol{u})}{\partial \boldsymbol{F}}\frac{\partial \boldsymbol{F}(\boldsymbol{u})}{\partial \boldsymbol{u}[j]}\right)^{\overset{23}{T}} \boldsymbol{\varphi}_k^\nabla \\ &\overset{(3.52)}{=} \left(\frac{\partial \boldsymbol{P}(\boldsymbol{u})}{\partial \boldsymbol{F}}\,(\boldsymbol{I}\otimes\boldsymbol{I})^{\overset{23}{T}}\,\boldsymbol{\varphi}_j^\nabla\right)^{\overset{23}{T}} \boldsymbol{\varphi}_k^\nabla = \left(\left(\frac{\partial \boldsymbol{P}(\boldsymbol{u})}{\partial \boldsymbol{F}}\right)^{\overset{23}{T}} \boldsymbol{\varphi}_j^\nabla\right)\boldsymbol{\varphi}_k^\nabla \\ &= \left(\frac{\partial \boldsymbol{P}(\boldsymbol{u})}{\partial \boldsymbol{F}}\right)^{\overset{23}{T}} (\boldsymbol{\varphi}_k^\nabla \otimes \boldsymbol{\varphi}_j^\nabla)\,.\end{aligned} \tag{3.54}$$

The derivation of the discretised volumetric part of the stress tensor is analogous, but more complex to write down due to the additional multiplication with the discrete pressure coefficient vector $\boldsymbol{w}$. One obtains

$$\begin{aligned}\frac{\partial \boldsymbol{A}(\boldsymbol{u})\boldsymbol{w}}{\partial \boldsymbol{u}} &= \begin{pmatrix} \frac{\partial \boldsymbol{A}(\boldsymbol{u})\boldsymbol{w}[1]}{\partial \boldsymbol{u}[1]} & \cdots & \frac{\partial \boldsymbol{A}(\boldsymbol{u})\boldsymbol{w}[1]}{\partial \boldsymbol{u}[N]} \\ \vdots & \ddots & \vdots \\ \frac{\partial \boldsymbol{A}(\boldsymbol{u})\boldsymbol{w}[N]}{\partial \boldsymbol{u}[1]} & \cdots & \frac{\partial \boldsymbol{A}(\boldsymbol{u})\boldsymbol{w}[N]}{\partial \boldsymbol{u}[N]} \end{pmatrix}, \in \mathbb{R}^{3N\times 3N}, \text{ where } \forall\, k,j = 1,\ldots N \\ \frac{\partial \boldsymbol{A}(\boldsymbol{u})\boldsymbol{w}[k]}{\partial \boldsymbol{u}[j]} &= \sum_p \omega_p\, p^h \frac{\partial}{\partial \boldsymbol{u}[j]}\left[(\det \boldsymbol{F})\boldsymbol{F}^{-T}\boldsymbol{\varphi}_k^\nabla\right] \in \mathbb{R}^{3\times 3}\,.\end{aligned} \tag{3.55}$$

Following Equation (3.54), the integrand is computed as

$$\frac{\partial}{\partial \boldsymbol{u}[j]}\left[(\det \boldsymbol{F})\boldsymbol{F}^{-T}\boldsymbol{\varphi}_k^\nabla\right] = \left(\frac{\partial (\det \boldsymbol{F})\boldsymbol{F}^{-T}}{\partial \boldsymbol{F}}\right)^{\overset{23}{T}} (\boldsymbol{\varphi}_k^\nabla \otimes \boldsymbol{\varphi}_j^\nabla) =: (\mathbb{A})^{\overset{23}{T}} (\boldsymbol{\varphi}_k^\nabla \otimes \boldsymbol{\varphi}_j^\nabla)\,, \tag{3.56}$$

with

$$\begin{aligned}\mathbb{A} &:= \frac{\partial (\det \boldsymbol{F})\boldsymbol{F}^{-T}}{\partial \boldsymbol{F}} \overset{(2.53)}{=} \boldsymbol{F}^{-T}\otimes\frac{\partial (\det \boldsymbol{F})}{\partial \boldsymbol{F}} + (\det \boldsymbol{F})\frac{\partial \boldsymbol{F}^{-T}}{\partial \boldsymbol{F}} \\ &\overset{(2.64)}{=} \boldsymbol{F}^{-T}\otimes(\det \boldsymbol{F})\boldsymbol{F}^{-T} + (\det \boldsymbol{F})\left[-\left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}^{-T}\right)^{\overset{24}{T}}\right] \\ &= (\det \boldsymbol{F})\left[\left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}^{-T}\right) - \left(\boldsymbol{F}^{-T}\otimes\boldsymbol{F}^{-T}\right)^{\overset{24}{T}}\right].\end{aligned} \tag{3.57}$$

3.4 Implementational details of the FOM

As already mentioned, the dynamic skeletal muscle model was implemented and solved within the MATLAB library KerMor [51], which can be downloaded from GitHub (https://github.com/KerMor). KerMor makes extensive use of object oriented programming. It provides routines for model order reduction of (first-order, single-field) nonlinear

dynamical systems using subspace projection and nonlinear approximation. Among other sample applications, it provides the class `+muscle`, consisting of several routines for the finite element simulation of the mechanics of an incompressible solid. More precisely, it assembles and solves the discretised equation system consisting of the momentum balance subject to the incompressibility constraint in its first-order form (3.37) as described in the previous sections. It already contains assembly routines for the 2nd PIOLA-KIRCHHOFF stress tensor, i.e. the nonlinear term $\boldsymbol{K}(\boldsymbol{u})$, and the corresponding elasticity tensor, i.e. the block $(\boldsymbol{Jg})_{21}$ of the analytical Jacobian, for some simple constitutive laws such as e.g. the NEO-HOOKE material. Those are implemented in the functions `evaluate()` and `getStateJacobianImpl()` respectively and subsequently used or called by routines contained in the core class `models.BaseSecondOrderSystem`. The derivation and the employed notation in the previous sections of this chapter tried to relate the theoretical concepts as well as possible to the utilised implementational approach. The default solution procedure calls the built-in MATLAB solver `ode15i`.
However, several bug fixes, alterations and additions were necessary in order to correctly simulate the FOM, i.e. the dynamic skeletal muscle model as it was introduced in this chapter. Most importantly those contained the correct implementation of the consequences from the volumetric deviatoric split for incompressible materials (c.f. Section 2.3). Not only the routines for the evaluation of the 2nd PIOLA-KIRCHHOFF stress tensor had to be revised, but more importantly also the computation of the corresponding discretised 4th-order elasticity tensor. The function `getStateJacobianImpl()` was rewritten from scratch, now making use of the concepts of tensor calculus for higher-order tensors. This way, the assembly procedure is not only readable and comprehensible for the common solid mechanics community, but additionally, it became a lot easier to exchange the utilised constitutive law. During this process, several (assembly) routines of the `models.BaseSecondOrderSystem` class have been reviewed and if necessary they were modified. For example, the saddle point structure of the overall Jacobian (c.f. Equation (3.50)) was implemented explicitly, making use of the relation $\left(\frac{\partial \boldsymbol{G}(\boldsymbol{u})}{\partial \boldsymbol{u}}\right)^T = \boldsymbol{A}(\boldsymbol{u}) = \frac{\partial(-\boldsymbol{K}(\boldsymbol{u})+\boldsymbol{A}(\boldsymbol{u})\,\boldsymbol{w})}{\partial \boldsymbol{w}}$, which became obvious during a clear re-derivation of the discretised equation system. Despite those corrections and modifications, some convergence issues remained when trying to simulate more elaborate examples. Therefore, the solution procedure was changed to call a self implemented implicit EULER scheme in combination with a modified NEWTON iteration. Additionally, the viscous damping contribution was omitted (by setting the viscosity $\eta = 0$) in every performed simulation. Being aware of the shortcomings of the implemented KELVIN-VOIGT model and not trusting the existing implementation in KerMor completely, this seemed to be the safest choice considering that essentially, the aim of this work was to build a ROM based on an existing FOM. More details on solver aspects and convergence issues of the FOM are described in Section 6.2. Lastly, the implementation of three subclasses of the abstract class `models.muscle.AMuscleConfig` for skeletal muscle simulations (c.f. Section 6.1), each of them offering different options with respect to e.g. geometry, discretisation, material law or simulation scenario, is a noteworthy contribution to KerMor and its class `+muscle`.

4 Reduced-order modelling for nonlinear dynamical systems

This chapter introduces the projection-based model order reduction technique by means of calculating the reduced basis via the well-established proper orthogonal decomposition. For an overview on (projection-based) model order reduction, the interested reader is referred to e.g. Antoulas & Sorensen [1], Hesthaven et al. [28] and Quarteroni et al. [37]. The aim of this chapter is to provide the necessary theoretical background of the methods employed, on the basis of a general first-order dynamical system.
As derived in the previous chapters, the dynamical system, which we aim to model in this work, contains nonlinear terms. In order to obtain a significant speedup in that case, it is not sufficient to merely project the equation system onto a lower dimensional subspace, since that approach, standing alone, still requires computations depending on the dimension of the full-order model. In order to overcome this issue, so-called hyper-reduction methods have proven useful. There exist several methods for the additional reduction of nonlinearities, e.g. the discrete empirical interpolation method (DEIM) originally proposed by Chaturantabut & Sorensen [16] or the energy conserving sampling and weighting method (ECSW), proposed by Farhat et al. [21]. Especially the ECSW could be a promising method to investigate for the problem here, since it does not only make use of the finite element structure by evaluating the nonlinear terms on a submesh, but additionally preserves energetic aspects of the system. However, these methods are not applied (yet) in this work, as they first require a stable projected model.

4.1 Reduced basis approximation

The idea of a reduced basis (RB) approximation is to replace a high-dimensional reference model, referred to as full-order model (FOM), by a reduced-order model (ROM) that is computationally less involved, i.e. that decreases the CPU time. This can be achieved by finding a problem-specific subspace of the full solution space $\mathcal{V}^h$, the so-called reduced space $\mathcal{V}^r \subset \mathcal{V}^h$ of lower dimension $\bar{r}$, and performing a subspace projection of the full-order system of dimension $\bar{d}$. RB approximation subspaces again inherit the inner products and norms of the FOM (here FE) spaces, i.e. also the ones of the exact spaces $\mathcal{V}$. As relatively recent introductory textbooks, the works of Hesthaven et al. [28] and Quarteroni et al. [37] can be recommended for an elaborate overview.
Starting point for this section is the finite element matrix formulation of the discretised problem, more specifically, the first-order system (c.f. Equation (3.42)) derived in Section 3.3, i.e.

$$\bar{\boldsymbol{M}}\bar{\boldsymbol{x}}' = \bar{\boldsymbol{D}}\bar{\boldsymbol{x}} + \bar{\boldsymbol{K}}(\bar{\boldsymbol{x}})\,[= \boldsymbol{g}(\bar{\boldsymbol{x}})] \quad \Longleftrightarrow \quad \boldsymbol{0} = -\bar{\boldsymbol{M}}\bar{\boldsymbol{x}}' + \boldsymbol{g}(\bar{\boldsymbol{x}})\,. \tag{4.1}$$

Therein, the mathematical quantities may depend on a parameter vector $\boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^p$, $p \in \mathbb{N}$, and/or a discrete time instant $t_i \in \mathbb{R}$, $i = 0, ..., n_t$. However, here these dependencies are not explicitly written down if not absolutely necessary. In the context of this whole chapter, we will refer to Equation (4.1) as the (nonlinear first-order) full-order model (FOM) of dimension $\bar{d}$.
Furthermore, whenever it is necessary or wanted to keep the theory as general as possible, we write e.g. an inner product as $\langle \cdot, \cdot \rangle_{\boldsymbol{\mathcal{A}}}$, where $\boldsymbol{\mathcal{A}}$ is a general symmetric matrix, mostly $\boldsymbol{\mathcal{A}} \in \{\boldsymbol{I}, \bar{\boldsymbol{H}}\}$, while just speaking of either orthogonal or orthonormal.

4.1.1 Link between the finite element and reduced spaces

Not only the ROM, but also already the FOM is a discrete approximation of the real state that is described by the continuous equations. In both cases, one introduces approximation spaces and bijections between these spaces and coefficient spaces in $\mathbb{R}^n$, $n \in \mathbb{N}$. The purpose of this section is to derive and show the relation between all those spaces and their bases. Therefore, let $s \in \mathcal{V}$ be the state of interest and $\hat{s} \in \hat{\mathcal{V}}$ in any finite-dimensional subspace $\hat{\mathcal{V}} \subset \mathcal{V}$ its discrete approximation.
Then, for the finite element approximation, i.e. the FOM, one defines a subspace $(\hat{\mathcal{V}} =)\mathcal{V}^h \subset \mathcal{V}$ of dimension $\bar{d}$ with orthonormal basis $\{\phi_i(\boldsymbol{X})\}_{i=1}^{\bar{d}}$, and an approximate state $(\hat{s} =)s^h \in \mathcal{V}^h$. Moreover, a bijection between $\mathcal{V}^h$ and $\mathbb{R}^{\bar{d}}$ is stated by

$$s^h(\boldsymbol{X}, t) = \sum_{i=1}^{\bar{d}} \bar{x}_i(t)\phi_i(\boldsymbol{X}) = \bar{\boldsymbol{x}}^T\boldsymbol{\phi}\,, \quad \text{with } \bar{\boldsymbol{x}}, \boldsymbol{\phi} \in \mathbb{R}^{\bar{d}}\,, \tag{4.2}$$

such that instead of solving for the state $(\hat{s} =)s^h \in \mathcal{V}^h$ one solves for the coefficient vector $\bar{\boldsymbol{x}} \in \mathbb{R}^{\bar{d}}$.
Equivalently, for the reduced basis approximation, i.e. the ROM, one defines a subspace $(\hat{\mathcal{V}} =)\mathcal{V}^r \subset \mathcal{V}$ of dimension $\bar{r}$ with orthonormal basis $\{\nu_i(\boldsymbol{X})\}_{i=1}^{\bar{r}}$, and an approximate state $(\hat{s} =)s^r \in \mathcal{V}^r$. Moreover, a bijection between $\mathcal{V}^r$ and $\mathbb{R}^{\bar{r}}$ is stated by

$$s^r(\boldsymbol{X}, t) = \sum_{i=1}^{\bar{r}} \bar{z}_i(t)\nu_i(\boldsymbol{X}) = \bar{\boldsymbol{z}}^T\boldsymbol{\nu}\,, \quad \text{with } \bar{\boldsymbol{z}}, \boldsymbol{\nu} \in \mathbb{R}^{\bar{r}}\,, \tag{4.3}$$

such that instead of solving for the state $(\hat{s} =)s^r \in \mathcal{V}^r$ one solves for the coefficient vector $\bar{\boldsymbol{z}} \in \mathbb{R}^{\bar{r}}$.
As both, s^h and s^r, approximate the state of interest, one can establish a relation between the two coefficient vectors $\bar{\boldsymbol{x}} \in \mathbb{R}^{\bar{d}}$ and $\bar{\boldsymbol{z}} \in \mathbb{R}^{\bar{r}}$:

$$\boldsymbol{\phi}^T\bar{\boldsymbol{x}} = \bar{\boldsymbol{x}}^T\boldsymbol{\phi} = s^h = \hat{s} = s^r = \bar{\boldsymbol{z}}^T\boldsymbol{\nu} = \boldsymbol{\nu}^T\bar{\boldsymbol{z}} \quad \xrightarrow{\boldsymbol{\phi}\boldsymbol{\phi}^T = \boldsymbol{I}_{\bar{d}}} \quad \bar{\boldsymbol{x}} = \underbrace{\boldsymbol{\phi}\boldsymbol{\nu}^T}_{=:\boldsymbol{R} \in \mathbb{R}^{\bar{d}\times\bar{r}}}\bar{\boldsymbol{z}}\,. \tag{4.4}$$

It is important, to emphasise here, that the matrix $\boldsymbol{R} \in \mathbb{R}^{\bar{d}\times\bar{r}}$, which relates the two coefficient vectors $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{z}}$ is not a basis of the reduced space $\mathcal{V}^r$. As will be seen later in Section 4.2, the matrix $\boldsymbol{R} = \bar{\boldsymbol{V}}$ is called a POD basis, when computed by means of a POD, which might lead to confusion, as often, the terms basis or space and matrix are used interchangeably.

Rather, with respect to the bases the relation

$$\boldsymbol{\nu} = \boldsymbol{R}^T \boldsymbol{\phi} \quad \Longrightarrow \quad \forall\, i = 1, ..., \bar{r} \quad \nu_i(\boldsymbol{X}) = \sum_{j=1}^{\bar{d}} r_{ji}\, \phi_j(\boldsymbol{X}) \tag{4.5}$$

can be concluded from Equation (4.4). This means that the ROM basis functions $\{\nu_i(\boldsymbol{X})\}_{i=1}^{\bar{r}}$ of space $\mathcal{V}^r$ are solutions expressed with respect to the FOM basis $\{\phi_i(\boldsymbol{X})\}_{i=1}^{\bar{d}}$ of space $\mathcal{V}^h$ by coefficients that are the components of the columns of $\boldsymbol{R}$ (c.f. [4]).

4.1.2 Subspace projection

Let $\mathcal{V}^r, \mathcal{W}^r \subset \mathcal{V}^h$ be two subspaces of the FOM space, with $\bar{r} := \dim(\mathcal{V}^r) = \dim(\mathcal{W}^r) \ll \dim(\mathcal{V}^h) =: \bar{d}$ and let $\bar{\boldsymbol{V}}, \bar{\boldsymbol{W}} \in \mathbb{R}^{\bar{d} \times \bar{r}}$ be two orthonormal matrices ($\bar{\boldsymbol{V}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}} = \boldsymbol{I}_{\bar{r}} = \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{W}} \in \mathbb{R}^{\bar{r} \times \bar{r}}$), whose column vectors span (in the sense of (4.5), i.e. being the coefficient vectors!) the subspaces $\mathcal{V}^r$ and $\mathcal{W}^r$ respectively. As was shown in Equation (4.4), the full coefficient vector $\bar{\boldsymbol{x}} \in \mathbb{R}^{\bar{d}}$ can be approximated by $\bar{\boldsymbol{x}} \approx \bar{\boldsymbol{V}} \bar{\boldsymbol{z}}$, with a reduced coefficient vector $\bar{\boldsymbol{z}} \in \mathbb{R}^{\bar{r}}$. Insertion of this approximation into the residuum equation $(4.1)_2$ yields

$$\boldsymbol{0} = -\bar{\boldsymbol{M}} \bar{\boldsymbol{V}} \bar{\boldsymbol{z}}' + \boldsymbol{g}(\bar{\boldsymbol{V}} \bar{\boldsymbol{z}})\,. \tag{4.6}$$

The reduced coefficient vector $\bar{\boldsymbol{z}} \in \mathbb{R}^{\bar{r}}$ can be determined through either a Galerkin or a Petrov-Galerkin projection. That is, in case of a Galerkin projection, the residual is imposed to be orthogonal to $\mathcal{V}^r$. The latter case is an oblique projection onto the so-called test space $\mathcal{W}^r$. Given a FOM, a corresponding ROM is uniquely defined by its associated (Petrov-)Galerkin projector $\Pi^{\boldsymbol{\mathcal{A}}}_{\bar{\boldsymbol{V}},\bar{\boldsymbol{W}}} := \bar{\boldsymbol{V}}(\bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}})^{-1} \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}}$ and $\Pi^{\boldsymbol{\mathcal{A}}}_{\bar{\boldsymbol{V}},\bar{\boldsymbol{W}}}$ again is uniquely defined by the subspaces spanned by the bases $\bar{\boldsymbol{V}}, \bar{\boldsymbol{W}}$ (see e.g. Volkwein [49] and Carlberg et al. [14]). As the Galerkin projection represents the specific case of a Petrov-Galerkin projection with $\bar{\boldsymbol{W}} := \bar{\boldsymbol{V}}$, the concept of subspace projection is now further explained by means of an arbitrary test space $\mathcal{W}^r$.
Performing the projection, the differential equation and initial condition of the reduced-order system of dimension $\bar{r}$ are obtained as

$$\bar{\boldsymbol{W}}^T \bar{\boldsymbol{M}} \bar{\boldsymbol{V}} \bar{\boldsymbol{z}}' = \bar{\boldsymbol{W}}^T \boldsymbol{g}(\bar{\boldsymbol{V}} \bar{\boldsymbol{z}}) \quad \text{with} \quad \bar{\boldsymbol{z}}(0) = \bar{\boldsymbol{z}}_0 := \bar{\boldsymbol{V}}^{-1} \bar{\boldsymbol{x}}_0\,. \tag{4.7}$$

Remarks:
Note here, that there exist two possibilities to define the reduced initial state.

1. The one chosen here follows a consistency argument, such that $\bar{\boldsymbol{V}} \bar{\boldsymbol{z}}(0) \stackrel{!}{=} \bar{\boldsymbol{x}}(0) = \bar{\boldsymbol{x}}_0$.
   - $\bar{\boldsymbol{V}}^{-1}$ means the Moore-Penrose pseudo inverse $\bar{\boldsymbol{V}}^\star := \left(\bar{\boldsymbol{V}}^T \bar{\boldsymbol{V}}\right)^{-1} \bar{\boldsymbol{V}}^T$.
   - In this approach, for the case $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$, i.e. $\bar{\boldsymbol{V}}^T \bar{\boldsymbol{V}} = \boldsymbol{I}$, one simply computes $\bar{\boldsymbol{z}}(0) = \bar{\boldsymbol{V}}^T \bar{\boldsymbol{x}}_0$.
   - In this approach, the initial condition is not influenced by the choice of whether a Petrov-Galerkin or a Galerkin projection is performed.

2. A second option follows from the oblique projection of the full state vector, i.e. $\bar{\boldsymbol{z}} = \left(\bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}}\right)^{-1} \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{x}}$, which naturally yields

$$\bar{\boldsymbol{z}}_0 = \bar{\boldsymbol{z}}(0) = \left(\bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}}\right)^{-1} \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{x}}(0) = \left(\bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}}\right)^{-1} \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{x}}_0 \,.$$

   - For a Galerkin projection together with the choice $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$, this results in the same as possibility one.

The reduced mass matrix is defined as

$$\bar{\boldsymbol{M}}^r := \bar{\boldsymbol{W}}^T \bar{\boldsymbol{M}} \bar{\boldsymbol{V}} \in \mathbb{R}^{\bar{r} \times \bar{r}} \,, \tag{4.8}$$

and the reduced, low-dimensional right-hand side function is

$$\boldsymbol{g}^r : \mathbb{R}^{\bar{r}} \to \mathbb{R}^{\bar{r}},\ \bar{\boldsymbol{z}} \mapsto \boldsymbol{g}^r(\bar{\boldsymbol{z}}) := \bar{\boldsymbol{W}}^T \boldsymbol{g}(\bar{\boldsymbol{V}} \bar{\boldsymbol{z}}) \,. \tag{4.9}$$

It can easily be seen that this way, the structure of the full-order system is inherited by the reduced system. The same holds for the steps in the solution process of the reduced nonlinear dynamical system, where equivalently to Equation (3.45) the reduced implicit function is given as

$$\boldsymbol{0} = \ \boldsymbol{f}^r(\bar{\boldsymbol{z}}_{i+1}) := -\bar{\boldsymbol{M}}^r (\bar{\boldsymbol{z}}_{i+1} - \bar{\boldsymbol{z}}_i) + \mathrm{d}t\, \boldsymbol{g}^r(\bar{\boldsymbol{z}}_{i+1}) \in \mathbb{R}^{\bar{r}} \,. \tag{4.10}$$

Likewise (c.f. Equation (3.49)), the reduced Jacobian is obtained from Equation (4.10) as

$$\boldsymbol{J}\boldsymbol{f}^r(\bar{\boldsymbol{z}}) := \frac{\partial \boldsymbol{f}^r(\bar{\boldsymbol{z}})}{\partial \bar{\boldsymbol{z}}} = -\bar{\boldsymbol{M}}^r + \mathrm{d}t\, \boldsymbol{J}\boldsymbol{g}^r(\bar{\boldsymbol{z}}) \in \mathbb{R}^{\bar{r} \times \bar{r}} \,, \tag{4.11}$$

where the nonlinear contribution is derived by applying the chain rule

$$\boldsymbol{J}\boldsymbol{g}^r(\bar{\boldsymbol{z}}) \overset{(4.9)}{:=} \frac{\partial \left(\bar{\boldsymbol{W}}^T \boldsymbol{g}(\bar{\boldsymbol{V}} \bar{\boldsymbol{z}})\right)}{\partial \bar{\boldsymbol{z}}} = \bar{\boldsymbol{W}}^T \boldsymbol{J}\boldsymbol{g}(\bar{\boldsymbol{V}} \bar{\boldsymbol{z}}) \bar{\boldsymbol{V}} \in \mathbb{R}^{\bar{r} \times \bar{r}} \,. \tag{4.12}$$

4.1.3 Projection error

To adequately choose the reduced spaces and assess their quality, it is common to look at the projection error. For the (Petrov-)Galerkin projection, this can be decomposed into two orthogonal components

$$\begin{aligned} \boldsymbol{\varepsilon} := \bar{\boldsymbol{x}} - \bar{\boldsymbol{V}} \bar{\boldsymbol{z}} &= \bar{\boldsymbol{x}} - \Pi^{\boldsymbol{\mathcal{A}}}_{\bar{\boldsymbol{V}}, \bar{\boldsymbol{W}}} \bar{\boldsymbol{x}} + \Pi^{\boldsymbol{\mathcal{A}}}_{\bar{\boldsymbol{V}}, \bar{\boldsymbol{W}}} \bar{\boldsymbol{x}} - \bar{\boldsymbol{V}} \bar{\boldsymbol{z}} \\ &= \bar{\boldsymbol{x}} - \bar{\boldsymbol{V}} \left(\bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}}\right)^{-1} \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{x}} + \bar{\boldsymbol{V}} \left(\bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}}\right)^{-1} \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{x}} - \bar{\boldsymbol{V}} \bar{\boldsymbol{z}} \\ &= \left[\boldsymbol{I} - \bar{\boldsymbol{V}} \left(\bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}}\right)^{-1} \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}}\right] \bar{\boldsymbol{x}} + \bar{\boldsymbol{V}} \left[\left(\bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}}\right)^{-1} \bar{\boldsymbol{W}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{x}} - \bar{\boldsymbol{z}}\right] \,. \end{aligned} \tag{4.13}$$

The first error component is orthogonal to $\mathcal{V}^r$ and can be precomputed. Thus it is perfectly suited to serve as an a priori error estimate.

4.2 Proper orthogonal decomposition (POD)

From the explanations in the previous section, it becomes obvious that naturally one of the main tasks in projection-based MOR is to obtain a suitable subspace. Suitable in this context means that it should reduce the problem dimension as much as possible while preserving accuracy and stability as well as possible. One way of determining an appropriate reduced basis is the proper orthogonal decomposition (POD). In the literature, due to the different fields it was developed for and applied in, it is also referred to as Hotelling-Transformation (c.f. [31]), principal component analysis (PCA), or Karhunen-Loeve transformation (c.f. [44]). For a detailed introduction into the theory of POD, the interested reader is referred to introductory textbooks or lecture notes, where among others, Quarteroni et al. [37] and Volkwein [49] can be recommended.

4.2.1 Offline-online decomposition

The POD is based on a split of the necessary computations into an offline (or training) and an online phase. By performing this offline-online decomposition, one aims at precomputing every expensive operation, depending on the full dimension $\bar{d}$, in an offline stage (with basically no time limitation), while the online stage is supposed to only perform computations with the complexity $\bar{r}$ of the reduced system. This can be achieved as follows:

Offline phase

1. Definition of a set of $n := (n_t + 1) \cdot n_p$ training parameters or tuples $\Xi := \left\{(t_i, \boldsymbol{\mu}_j) \,|\, t_i \in [0, T], \boldsymbol{\mu}_j \in \mathcal{P}, i = 0, \ldots, n_t, j = 1, \ldots, n_p\right\}$.
2. Precomputation of training data $\boldsymbol{S}_{\boldsymbol{\mu}_j} := \left[\bar{\boldsymbol{x}}_{\boldsymbol{\mu}_j}(t_0), \ldots, \bar{\boldsymbol{x}}_{\boldsymbol{\mu}_j}(t_{n_t})\right] \in \mathbb{R}^{\bar{d} \times (n_t+1)}, \boldsymbol{\mu}_j \in \mathcal{P}$ and assembly of $\boldsymbol{S} := \left[\boldsymbol{S}_{\boldsymbol{\mu}_1}, \ldots, \boldsymbol{S}_{\boldsymbol{\mu}_{n_p}}\right] \in \mathbb{R}^{\bar{d} \times n}$ (also called snapshot matrix) with rank $m \leq \min\{\bar{d}, n\}$.
3. Performing a POD (c.f. Section 4.2.2) on $\boldsymbol{S}$, to obtain the reduced coefficient matrix $\bar{\boldsymbol{V}} \in \mathbb{R}^{\bar{d} \times l}$ (where $1 \leq l \leq \bar{d}$ is the chosen maximum size for the POD basis), and choosing the necessary size $\bar{r} \ll l$ for the reduced system.
4. Precomputation of the linear contributions, i.e. the reduced matrices $\bar{\boldsymbol{M}}^r$ and $\bar{\boldsymbol{D}}^r$.

Online phase

1. Selection of a (new) parameter $\boldsymbol{\mu} \in \mathcal{P}$.
2. Solving the reduced system (4.7) for $\bar{\boldsymbol{z}}(t, \boldsymbol{\mu}) \in \mathbb{R}^{\bar{r}}$.
3. Reconstruction of the full state space solution $\bar{\boldsymbol{x}}(t, \boldsymbol{\mu}) = \bar{\boldsymbol{V}} \bar{\boldsymbol{z}}(t, \boldsymbol{\mu}) \in \mathbb{R}^{\bar{d}}$.

Remark:
Step 2 of the online phase becomes more intricate, when dealing with nonlinear systems. As visible in Equations (4.9) and (4.12), the low-dimensional, nonlinear right-hand side

function $\boldsymbol{g}^r$ and Jacobian $\boldsymbol{J}\boldsymbol{g}^r$ still require computations depending on the dimension of the full-order model. This is where methods that additionally approximate projected nonlinearities, like the already mentioned DEIM and ECSW, need to be applied in order to achieve a sufficient computational efficiency.

4.2.2 POD basis computation

POD seeks an orthogonal projector $\Pi^{\boldsymbol{\mathcal{A}}}_{\bar{\boldsymbol{V}},\bar{\boldsymbol{V}}}$ that minimises the integrated first component of the projection error (c.f. Equation (4.13)), i.e.

$$\int_0^T \left\| \bar{\boldsymbol{x}}(t) - \Pi^{\boldsymbol{\mathcal{A}}}_{\bar{\boldsymbol{V}},\bar{\boldsymbol{V}}} \bar{\boldsymbol{x}}(t) \right\|^2_{\boldsymbol{\mathcal{A}}} \mathrm{d}t = \int_0^T \left\| \bar{\boldsymbol{x}}(t) - \bar{\boldsymbol{V}}\bar{\boldsymbol{V}}^T \boldsymbol{\mathcal{A}}\, \bar{\boldsymbol{x}}(t) \right\|^2_{\boldsymbol{\mathcal{A}}} \mathrm{d}t \,. \tag{4.14}$$

In a discrete setting with precomputed snapshot data $\boldsymbol{S} = [\bar{\boldsymbol{x}}_1, ..., \bar{\boldsymbol{x}}_n] \in \mathbb{R}^{\bar{d}\times n}$ for different parameters and different time steps, this means to find

$$\min_{\boldsymbol{v}_1,...,\boldsymbol{v}_l \in \mathbb{R}^{\bar{d}}} \sum_{k=1}^{n} \left\| \bar{\boldsymbol{x}}_k - \sum_{i=1}^{l} \langle \bar{\boldsymbol{x}}_k, \boldsymbol{v}_i \rangle_{\boldsymbol{\mathcal{A}}} \boldsymbol{v}_i \right\|^2_{\boldsymbol{\mathcal{A}}} \quad \text{s.t. } \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle_{\boldsymbol{\mathcal{A}}} = \delta_{ij} \quad \forall\, i,j = 1, ..., l \tag{4.15}$$

for a chosen $l \in \{1, ..., m\}$. Instead of solving problem (4.15), one can equivalently solve the constrained optimisation problem

$$\max_{\boldsymbol{v}_1,...,\boldsymbol{v}_l \in \mathbb{R}^{\bar{d}}} \sum_{i=1}^{l} \sum_{k=1}^{n} \left| \langle \bar{\boldsymbol{x}}_k, \boldsymbol{v}_i \rangle_{\boldsymbol{\mathcal{A}}} \right|^2 \quad \text{s.t. } \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle_{\boldsymbol{\mathcal{A}}} = \delta_{ij} \quad \forall\, i,j = 1, ..., l \,. \tag{4.16}$$

The following Theorem states the relation between the POD and a singular value decomposition (SVD) and can be looked up (including further details) e.g. in Volkwein [49].

Theorem 1 *(POD computation)*
Let $\boldsymbol{S} = [\bar{\boldsymbol{x}}_1, ..., \bar{\boldsymbol{x}}_n] \in \mathbb{R}^{\bar{d}\times n}$ be a given matrix with rank $m \leq \min\{\bar{d}, n\}$. Let $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{\bar{d}\times\bar{d}}$ be a symmetric, positive definite matrix and define $\boldsymbol{S}^{\boldsymbol{\mathcal{A}}} := \boldsymbol{\mathcal{A}}^{\frac{1}{2}}\boldsymbol{S}$. Further, let $\boldsymbol{S}^{\boldsymbol{\mathcal{A}}} = \bar{\boldsymbol{V}}^{\boldsymbol{\mathcal{A}}}\boldsymbol{\Sigma}\left(\bar{\boldsymbol{U}}^{\boldsymbol{\mathcal{A}}}\right)^T$ be the singular value decomposition of $\boldsymbol{S}^{\boldsymbol{\mathcal{A}}}$, where $\bar{\boldsymbol{V}}^{\boldsymbol{\mathcal{A}}} = \left[\bar{\boldsymbol{v}}^{\boldsymbol{\mathcal{A}}}_1, ..., \bar{\boldsymbol{v}}^{\boldsymbol{\mathcal{A}}}_{\bar{d}}\right] \in \mathbb{R}^{\bar{d}\times\bar{d}}$ and $\bar{\boldsymbol{U}}^{\boldsymbol{\mathcal{A}}} = \left[\bar{\boldsymbol{u}}^{\boldsymbol{\mathcal{A}}}_1, ..., \bar{\boldsymbol{u}}^{\boldsymbol{\mathcal{A}}}_n\right] \in \mathbb{R}^{n\times n}$ are orthogonal matrices containing the left and right singular vectors respectively, and the matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{\bar{d}\times n}$ contains the singular values $\sigma_i \in \mathbb{R}$, $i = 1, ..., m$, on a diagonal starting on the upper left.
Then, for $l \in \{1, ..., m\}$ the solution to problem (4.16) is given by the vectors $\bar{\boldsymbol{v}}_i := \boldsymbol{\mathcal{A}}^{-\frac{1}{2}}\bar{\boldsymbol{v}}^{\boldsymbol{\mathcal{A}}}_i$, $i = 1, ..., l$. Moreover, $\arg\max$ (4.16) $= \sum_{i=1}^{l} \sigma_i^2$.

Remarks:

1. For $l \in \{1, ..., m\}$, the vectors $\{\bar{\boldsymbol{v}}_i\}_{i=1}^{l}$ are called POD basis of rank l. A vector $\bar{\boldsymbol{v}}_i$ is often also called i^{th} POD mode.

2. For all $l \leq m$, the POD basis is optimal in the mean among all rank l approximations to the columns of $\boldsymbol{S}$.

3. For $\bar{\boldsymbol{V}} := [\bar{\boldsymbol{v}}_1, ..., \bar{\boldsymbol{v}}_l] = \boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \bar{\boldsymbol{V}}^{\boldsymbol{\mathcal{A}}} \in \mathbb{R}^{\bar{d} \times l}$, $l \leq m$, it holds $\bar{\boldsymbol{V}}^T \boldsymbol{\mathcal{A}} \bar{\boldsymbol{V}} = \left(\boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \bar{\boldsymbol{V}}^{\boldsymbol{\mathcal{A}}}\right)^T \boldsymbol{\mathcal{A}} \left(\boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \bar{\boldsymbol{V}}^{\boldsymbol{\mathcal{A}}}\right) = \left(\bar{\boldsymbol{V}}^{\boldsymbol{\mathcal{A}}}\right)^T \bar{\boldsymbol{V}}^{\boldsymbol{\mathcal{A}}} = \boldsymbol{I}_l$, i.e. $\bar{\boldsymbol{V}}$ is orthonormal in the inner product norm and thus suitable as a projection (or reduced coefficient) matrix.

4. For $\bar{\boldsymbol{V}} \in \mathbb{R}^{\bar{d} \times l}$, $l \leq m$, the POD approximation error can be quantified using the singular values σ_i, $i = l+1, ..., m$, as $\arg\min(4.15) = \sum_{i=l+1}^{m} \sigma_i^2$. Therefore, the singular value decay is considered an a priori error estimate, which helps to choose an appropriate reduced model size $\bar{r} \ll l$.

5. If a POD basis was computed based on Theorem 1, this will be written here in short as $\bar{\boldsymbol{V}} = \boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}} \boldsymbol{S}\right)$.

6. Instead of computing a singular value decomposition, one could equivalently solve the eigenvalue problem of the symmetric correlation matrix $\boldsymbol{C}^{\boldsymbol{\mathcal{A}}} := \left(\boldsymbol{S}^{\boldsymbol{\mathcal{A}}}\right)^T \boldsymbol{S}^{\boldsymbol{\mathcal{A}}} = \boldsymbol{S}^T \boldsymbol{\mathcal{A}} \boldsymbol{S}$ (often referred to as method of snapshots). However, as $\kappa\left(\boldsymbol{C}^{\boldsymbol{\mathcal{A}}}\right) = \kappa\left(\boldsymbol{S}^{\boldsymbol{\mathcal{A}}}\right)^2$, and due to round-off errors introduced through the computation of $\boldsymbol{C}^{\boldsymbol{\mathcal{A}}}$, this might lead to inaccurate results for the modes associated with small singular values (c.f. e.g. [37]). Therefore, computing the singular value decomposition is preferred over solving the eigenvalue problem if the matrix $\boldsymbol{\mathcal{A}}^{\frac{1}{2}}$ is available.

As a matter of fact, the POD basis is the best approximation on the given training data in the mean-square sense. This of course raises the question, which data should be computed during the offline phase and included in the training data $\boldsymbol{S}$. One suggestion is to utilise a POD-Greedy procedure (see Haasdonk & Ohlberger [26], Haasdonk [25]), during which the snapshots are computed iteratively, choosing the next parameter based on the occurrence of largest errors. As in this work, the POD-Greedy method is not applied, it is not further explained or discussed.

5 Reduced-order skeletal muscle model

5.1 Subspace projection for constrained and multi-field problems

In the previous chapter, the reduced basis approximation for a general nonlinear dynamical system of the form $\bar{\boldsymbol{M}}\bar{\boldsymbol{x}}'(t) = \boldsymbol{g}(\bar{\boldsymbol{x}}(t))$ was introduced. Now, these equations will be derived for the specific case of the dynamic skeletal muscle model with its block structure arising from the three fields contained in the system.
The full coefficient vector of the first-order system, $\bar{\boldsymbol{x}} = (\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})^T \in \mathbb{R}^{\bar{d}}$, is replaced by a reduced coefficient vector $\bar{\boldsymbol{z}} = (\boldsymbol{z}_u, \boldsymbol{z}_v, \boldsymbol{z}_w)^T \in \mathbb{R}^{\bar{r}}$, with $\boldsymbol{z}_u \in \mathbb{R}^{r_u}$, $\boldsymbol{z}_v \in \mathbb{R}^{r_v}$, $\boldsymbol{z}_w \in \mathbb{R}^{r_p}$ and $\bar{r} := r_u + r_v + r_p$, where for the reduced dimensions of the three fields it is $r_u, r_v \ll 3N$ and $r_p \ll N_p$ (with N and N_p being the number of nodes for the position and the pressure respectively). More detailed, the FOM construction (c.f. procedure described in Section 4.1.1)

$$s^h = \left(\boldsymbol{x}^h, (\boldsymbol{x}^h)', p^h\right)^T = \bar{\boldsymbol{x}}^T\boldsymbol{\phi} \quad \in \left(\mathcal{V}_{\boldsymbol{u}}^h \times \mathcal{V}_{\boldsymbol{u}}^h \times \mathcal{V}_p^h\right) \tag{5.1}$$
$$\text{with coefficients } \bar{\boldsymbol{x}} = \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ \boldsymbol{w} \end{pmatrix} \in \mathbb{R}^{\bar{d}=3N+3N+N_p} \text{ and basis } \boldsymbol{\phi} = \begin{pmatrix} [\boldsymbol{\varphi}] \\ [\boldsymbol{\varphi}] \\ \boldsymbol{\psi} \end{pmatrix} \in \mathbb{R}^{\bar{d}}$$

is replaced by a ROM construction of the form

$$s^r := (\boldsymbol{x}^r, (\boldsymbol{x}^r)', p^r)^T = \bar{\boldsymbol{z}}^T\boldsymbol{\nu} \quad \in \left(\mathcal{V}_{\boldsymbol{u}}^r \times \mathcal{V}_{\boldsymbol{v}}^r \times \mathcal{V}_p^r\right) \subset \left(\mathcal{V}_{\boldsymbol{u}}^h \times \mathcal{V}_{\boldsymbol{u}}^h \times \mathcal{V}_p^h\right) \tag{5.2}$$
$$\text{with coefficients } \bar{\boldsymbol{z}} = \begin{pmatrix} \boldsymbol{z}_u \\ \boldsymbol{z}_v \\ \boldsymbol{z}_w \end{pmatrix} \in \mathbb{R}^{\bar{r}=r_u+r_v+r_p} \text{ and basis } \boldsymbol{\nu} = \begin{pmatrix} [\boldsymbol{\nu}_u] \\ [\boldsymbol{\nu}_v] \\ \boldsymbol{\nu}_w \end{pmatrix} \in \mathbb{R}^{\bar{r}}\,.$$

This means, each field can be approximated by a different subspace $\mathcal{V}_\star^r$, with basis $\boldsymbol{\nu}_\star$, that is related to the FE basis by a matrix $(\boldsymbol{R}_\star =)\, \boldsymbol{V}_\star$, $\star \in \{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}\}$. The same holds for the test spaces $\mathcal{W}_\star^r$. These matrices are assembled in two orthonormal block-diagonal matrices

$$\bar{\boldsymbol{V}} := \begin{pmatrix} \boldsymbol{V}_u & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{V}_v & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{V}_w \end{pmatrix} \in \mathbb{R}^{\bar{d}\times\bar{r}} \text{ and } \bar{\boldsymbol{W}} := \begin{pmatrix} \boldsymbol{W}_u & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W}_v & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{W}_w \end{pmatrix} \in \mathbb{R}^{\bar{d}\times\bar{r}}\,, \tag{5.3}$$

with $\boldsymbol{V}_u, \boldsymbol{W}_u \in \mathbb{R}^{3N\times r_u}$, $\boldsymbol{V}_v, \boldsymbol{W}_v \in \mathbb{R}^{3N\times r_v}$ and $\boldsymbol{V}_w, \boldsymbol{W}_w \in \mathbb{R}^{N_p\times r_p}$, such that $\bar{\boldsymbol{x}} \approx \bar{\boldsymbol{V}}\bar{\boldsymbol{z}}$.
The relation between the FOM and ROM bases can be stated in this block matrix shape

as

$$\begin{pmatrix}[\boldsymbol{\nu}_u]\\ [\boldsymbol{\nu}_v]\\ \boldsymbol{\nu}_w\end{pmatrix} \overset{(4.5)}{=} \begin{pmatrix}\boldsymbol{V}_u^T & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{V}_v^T & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{V}_w^T\end{pmatrix} \begin{pmatrix}[\boldsymbol{\varphi}]\\ [\boldsymbol{\varphi}]\\ \boldsymbol{\psi}\end{pmatrix} = \begin{pmatrix}\boldsymbol{V}_u^T[\boldsymbol{\varphi}]\\ \boldsymbol{V}_v^T[\boldsymbol{\varphi}]\\ \boldsymbol{V}_w^T\boldsymbol{\psi}\end{pmatrix} . \tag{5.4}$$

Remark:
Note that the FOM basis fulfils the LBB (or inf-sup) stability condition*. The POD bases (coefficient matrices) $\boldsymbol{V}_\star$ need to be chosen carefully such that the ROM basis fulfils the LBB condition as well. This will be investigated further in Sections 5.2 and 5.3.

Following the steps in Section 4.1.2, now the reduced system components can be assembled. The reduced initial condition is

$$\bar{\boldsymbol{z}}(0) = \begin{pmatrix}\boldsymbol{z}_u(0)\\ \boldsymbol{z}_v(0)\\ \boldsymbol{z}_w(0)\end{pmatrix} = \begin{pmatrix}\boldsymbol{z}_{u_0}\\ \boldsymbol{z}_{v_0}\\ \boldsymbol{z}_{w_0}\end{pmatrix} := \begin{pmatrix}\boldsymbol{V}_u^{-1}\boldsymbol{u}_0\\ \boldsymbol{V}_v^{-1}\boldsymbol{v}_0\\ \boldsymbol{0}\end{pmatrix} \underset{(4.7)_2,(5.3)_1}{\overset{(3.43)}{=}} \bar{\boldsymbol{V}}^{-1}\bar{\boldsymbol{x}}_0 \in \mathbb{R}^{\bar{r}} . \tag{5.5}$$

The reduced mass matrix is assembled with the definitions of Equations (4.8), (5.3) and (3.37), yielding

$$\begin{aligned}\bar{\boldsymbol{M}}^r = \bar{\boldsymbol{W}}^T\bar{\boldsymbol{M}}\bar{\boldsymbol{V}} &= \begin{pmatrix}\boldsymbol{W}_u^T & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{W}_v^T & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{W}_w^T\end{pmatrix} \begin{pmatrix}\boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{M} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0}\end{pmatrix} \begin{pmatrix}\boldsymbol{V}_u & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{V}_v & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{V}_w\end{pmatrix}\\ &= \begin{pmatrix}\boldsymbol{W}_u^T\boldsymbol{V}_u & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{W}_v^T\boldsymbol{M}\boldsymbol{V}_v & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0}\end{pmatrix} \in \mathbb{R}^{\bar{r}\times\bar{r}} .\end{aligned} \tag{5.6}$$

Inserting the Definitions (5.3) and (3.44) into Equation (4.9), yields the reduced right-hand side function `odefun` of dimension $\bar{r}$, as

$$\begin{aligned}\boldsymbol{g}^r(\bar{\boldsymbol{z}}) &= \bar{\boldsymbol{W}}^T\boldsymbol{g}(\bar{\boldsymbol{V}}\bar{\boldsymbol{z}})\\ &= \begin{pmatrix}\boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & -\boldsymbol{W}_v^T\boldsymbol{D}\boldsymbol{V}_v & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0}\end{pmatrix} \begin{pmatrix}\boldsymbol{z}_u\\ \boldsymbol{z}_v\\ \boldsymbol{z}_w\end{pmatrix} + \begin{pmatrix}\boldsymbol{W}_u^T\boldsymbol{V}_v\boldsymbol{z}_v\\ \boldsymbol{W}_v^T\left[-\boldsymbol{K}(\boldsymbol{V}_u\boldsymbol{z}_u) + \boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\,\boldsymbol{V}_w\boldsymbol{z}_w\right]\\ \boldsymbol{W}_w^T\boldsymbol{G}(\boldsymbol{V}_u\boldsymbol{z}_u)\end{pmatrix} .\end{aligned} \tag{5.7}$$

The nonlinear contribution of the reduced block Jacobian is derived from Equations (4.12) and (5.7). It is

$$\begin{aligned}\boldsymbol{J}\boldsymbol{g}^r(\bar{\boldsymbol{z}}) &= \bar{\boldsymbol{W}}^T\boldsymbol{J}\boldsymbol{g}(\bar{\boldsymbol{V}}\bar{\boldsymbol{z}})\bar{\boldsymbol{V}}\\ &= \left(\begin{array}{ccc|c}\boldsymbol{0} & \boldsymbol{W}_u^T\boldsymbol{V}_v & \boldsymbol{0}\\ \boldsymbol{W}_v^T\left[-\frac{\partial\boldsymbol{K}(\boldsymbol{V}_u\boldsymbol{z}_u)}{\partial(\boldsymbol{V}_u\boldsymbol{z}_u)} + \frac{\partial\boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\boldsymbol{V}_w\boldsymbol{z}_w}{\partial(\boldsymbol{V}_u\boldsymbol{z}_u)}\right]\boldsymbol{V}_u & -\boldsymbol{W}_v^T\boldsymbol{D}\boldsymbol{V}_v & \boldsymbol{W}_v^T\boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\boldsymbol{V}_w\\ \hline \boldsymbol{W}_w^T\boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)^T\boldsymbol{V}_u & \boldsymbol{0} & \boldsymbol{0}\end{array}\right) .\end{aligned} \tag{5.8}$$

*LBB stands for Ladyzhenskaya-Babuška-Brezzi

Note that the transpose of the reduced Jacobian block $(\boldsymbol{Jg}^r)_{23}$, i.e. $\left[\boldsymbol{W}_v^T \boldsymbol{A}(\boldsymbol{V}_u \boldsymbol{z}_u)\boldsymbol{V}_w\right]^T = \boldsymbol{V}_w^T \boldsymbol{A}(\boldsymbol{V}_u \boldsymbol{z}_u)^T \boldsymbol{W}_v \in \mathbb{R}^{r_p \times r_v}$ is not equal to the reduced Jacobian block $(\boldsymbol{Jg}^r)_{31} = \boldsymbol{W}_w^T \boldsymbol{A}(\boldsymbol{V}_u \boldsymbol{z}_u)^T \boldsymbol{V}_u \in \mathbb{R}^{r_p \times r_u}$ anymore for a PETROV-GALERKIN projection and/or for arbitrary matrices $\boldsymbol{V}_u \neq \boldsymbol{V}_v$. In particular, for dimensions $r_u \neq r_v$, they do not even have the same size. We will further elaborate on this observation in Section 5.2. For future use, we introduce the abbreviation $(\boldsymbol{A}^r(\boldsymbol{z}_u))^T$ for the reduced Jacobian block $(\boldsymbol{Jg}^r)_{31}$, i.e.

$$\boldsymbol{A}^r(\boldsymbol{z}_u) := \boldsymbol{V}_u^T \boldsymbol{A}(\boldsymbol{V}_u \boldsymbol{z}_u)\boldsymbol{W}_w \in \mathbb{R}^{r_u \times r_p} . \tag{5.9}$$

5.2 Options for subspace computations and combinations

Similar to Section 5.1, where the concept of the general subspace projection was extended to the block structure of the dynamic skeletal muscle model, the POD basis computation shall now be applied to this case with its coefficient vectors $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$ of the three fields that need to be approximated as well. From now on, the case of a GALERKIN projection is considered, i.e. we set $\bar{\boldsymbol{W}} = \bar{\boldsymbol{V}}$ and only need to focus on choices for the three blocks $\boldsymbol{V}_u$, $\boldsymbol{V}_v$ and $\boldsymbol{V}_w$ of Equation (5.3). Furthermore, the reduced bases are always computed by means of different versions of a POD on snapshot data, thus we generally speak of a POD-GALERKIN ROM as it is also done e.g. in [4].
The training data or snapshot matrix for the case at hand consists of three parts,

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{S}_u \\ \boldsymbol{S}_v \\ \boldsymbol{S}_w \end{bmatrix} \in \mathbb{R}^{\bar{d} \times n} , \tag{5.10}$$

where n was the number of training parameters, i.e. the number of snapshots obtained during the offline phase (see Section 4.2.1). Therefore, the obvious choice for the POD bases is

$$\begin{aligned} \boldsymbol{V}_u &:= \boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}} \boldsymbol{S}_u\right) \in \mathbb{R}^{3N \times r_u} \quad \text{with } \boldsymbol{\mathcal{A}} \in \{\boldsymbol{H_u}, \boldsymbol{I}_{3N}\} , \\ \boldsymbol{V}_v &:= \boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}} \boldsymbol{S}_v\right) \in \mathbb{R}^{3N \times r_v} \quad \text{with } \boldsymbol{\mathcal{A}} \in \{\boldsymbol{H_u}, \boldsymbol{I}_{3N}\} , \\ \boldsymbol{V}_w &:= \boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}} \boldsymbol{S}_w\right) \in \mathbb{R}^{N_p \times r_p} \quad \text{with } \boldsymbol{\mathcal{A}} \in \{\boldsymbol{H}_p, \boldsymbol{I}_{N_p}\} . \end{aligned} \tag{5.11}$$

However, going back to the starting point, i.e. the discretisation of the (original) second-order system, one could also think of another option. For the procedure followed so far, the second-order system was first transformed into a first-order system and subsequently this first-order full system was projected to obtain a first-order reduced system. This way, a first-order system ROM was obtained (c.f. Section 5.1 and Equations (4.7), (5.3)

– (5.7)), which for the case of a GALERKIN projection has the form

$$\begin{pmatrix} \boldsymbol{V}_u^T\boldsymbol{V}_u & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{V}_v^T\boldsymbol{M}\boldsymbol{V}_v & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_u \\ \boldsymbol{z}_v \\ \boldsymbol{z}_w \end{pmatrix}' = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{V}_v^T\boldsymbol{D}\boldsymbol{V}_v & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_u \\ \boldsymbol{z}_v \\ \boldsymbol{z}_w \end{pmatrix} + \begin{pmatrix} \boldsymbol{V}_u^T\boldsymbol{V}_v\boldsymbol{z}_v \\ \boldsymbol{V}_v^T\left[-\boldsymbol{K}(\boldsymbol{V}_u\boldsymbol{z}_u) + \boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\,\boldsymbol{V}_w\boldsymbol{z}_w\right] \\ \boldsymbol{V}_w^T\boldsymbol{G}(\boldsymbol{V}_u\boldsymbol{z}_u) \end{pmatrix}. \tag{5.12}$$

This procedure suggests a second option in inverse order, which would be to first project the second-order system and subsequently transform the obtained reduced second-order system into a first-order system ROM. To derive the final form of the system for the second option, starting point is the constrained second-order system of dimension $\tilde{d} := 3N + N_p$ (c.f. Equation (3.35)) in block-matrix format

$$\underbrace{\begin{pmatrix} \boldsymbol{M} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{pmatrix}}_{=:\tilde{\boldsymbol{M}}\in\mathbb{R}^{\tilde{d}\times\tilde{d}}} \underbrace{\begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{w} \end{pmatrix}''}_{=\tilde{\boldsymbol{x}}''\in\mathbb{R}^{\tilde{d}}} = \underbrace{\begin{pmatrix} -\boldsymbol{D} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{pmatrix}}_{=:\tilde{\boldsymbol{D}}\in\mathbb{R}^{\tilde{d}\times\tilde{d}}} \underbrace{\begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{w} \end{pmatrix}}_{=:\tilde{\boldsymbol{x}}} + \underbrace{\begin{pmatrix} -\boldsymbol{K}(\boldsymbol{u}) + \boldsymbol{A}(\boldsymbol{u})\,\boldsymbol{w} \\ \boldsymbol{G}(\boldsymbol{u}) \end{pmatrix}}_{=:\tilde{\boldsymbol{K}}(\tilde{\boldsymbol{x}})\in\mathbb{R}^{\tilde{d}\times\tilde{d}}}. \tag{5.13}$$

Then, the full coefficient vector $(\boldsymbol{u}, \boldsymbol{w})^T \in \mathbb{R}^{3N+N_p}$ can be approximated by a reduced coefficient vector as $(\boldsymbol{z}_u, \boldsymbol{z}_w)^T \in \mathbb{R}^{r_u+r_p}$

$$\begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{w} \end{pmatrix} \approx \begin{pmatrix} \boldsymbol{V}_u & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{V}_w \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_u \\ \boldsymbol{z}_w \end{pmatrix}, \quad \text{with } \boldsymbol{V}_u \in \mathbb{R}^{3N\times r_u}, \boldsymbol{V}_w \in \mathbb{R}^{N_p\times r_p}. \tag{5.14}$$

Inserting this approximation into Equation (5.13) and performing the GALERKIN projection yields the ROM in its second-order system format

$$\begin{pmatrix} \boldsymbol{V}_u^T\boldsymbol{M}\boldsymbol{V}_u & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_u \\ \boldsymbol{z}_w \end{pmatrix}'' = \begin{pmatrix} -\boldsymbol{V}_u^T\boldsymbol{D}\boldsymbol{V}_u & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_u \\ \boldsymbol{z}_w \end{pmatrix} + \begin{pmatrix} \boldsymbol{V}_u^T\left[-\boldsymbol{K}(\boldsymbol{V}_u\boldsymbol{z}_u) + \boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\,\boldsymbol{V}_w\boldsymbol{z}_w\right] \\ \boldsymbol{V}_w^T\boldsymbol{G}(\boldsymbol{V}_u\boldsymbol{z}_u) \end{pmatrix}. \tag{5.15}$$

Introducing a reduced velocity coefficient vector $\boldsymbol{z}_v := \boldsymbol{z}_u' \in \mathbb{R}^{r_u}$, this system can now be transformed into the reduced first-order system

$$\begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{V}_u^T\boldsymbol{M}\boldsymbol{V}_u & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_u \\ \boldsymbol{z}_v \\ \boldsymbol{z}_w \end{pmatrix}' = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{V}_u^T\boldsymbol{D}\boldsymbol{V}_u & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_u \\ \boldsymbol{z}_v \\ \boldsymbol{z}_w \end{pmatrix} + \begin{pmatrix} \boldsymbol{z}_v \\ \boldsymbol{V}_u^T\left[-\boldsymbol{K}(\boldsymbol{V}_u\boldsymbol{z}_u) + \boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\,\boldsymbol{V}_w\boldsymbol{z}_w\right] \\ \boldsymbol{V}_w^T\boldsymbol{G}(\boldsymbol{V}_u\boldsymbol{z}_u) \end{pmatrix}. \tag{5.16}$$

Comparing the two ROM, i.e. Equations (5.12) with (5.16), one can see that they are the same only for the choice $\boldsymbol{V}_v = \boldsymbol{V}_u$.

Remark:
For a matrix $\boldsymbol{V}_u$ that is orthonormal with respect to a weighted inner product, the $\left(\bar{\boldsymbol{M}}\right)_{11}$ block in Equation (5.12) is not identity. However, the whole equation represented by the first blocks can be multiplied by $\left(\boldsymbol{V}_u^T\boldsymbol{V}_u\right)^{-1}$, yielding the identity when choosing $\boldsymbol{V}_v = \boldsymbol{V}_u$.

For completeness, we also look at the Jacobians of the reduced right-hand side function `odefun` for the two ROM, i.e. $\boldsymbol{Jg}^r(\bar{\boldsymbol{z}}) \in \mathbb{R}^{\bar{r}\times\bar{r}}$. For the first option, this was derived in Section 5.1, c.f. Equation (5.8), and simplifies to

$$\left(\begin{array}{cc|c} \boldsymbol{0} & \boldsymbol{V}_u^T\boldsymbol{V}_v & \boldsymbol{0} \\ \boldsymbol{V}_v^T\left[-\frac{\partial \boldsymbol{K}(\boldsymbol{V}_u\boldsymbol{z}_u)}{\partial(\boldsymbol{V}_u\boldsymbol{z}_u)} + \frac{\partial \boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\boldsymbol{V}_w\boldsymbol{z}_w}{\partial(\boldsymbol{V}_u\boldsymbol{z}_u)}\right]\boldsymbol{V}_u & -\boldsymbol{V}_v^T\boldsymbol{D}\boldsymbol{V}_v & \boldsymbol{V}_v^T\boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\boldsymbol{V}_w \\ \hline \boldsymbol{V}_w^T\boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)^T\boldsymbol{V}_u & \boldsymbol{0} & \boldsymbol{0} \end{array}\right) \tag{5.17}$$

in the Galerkin projection case. For the second option, we obtain

$$\left(\begin{array}{cc|c} \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{V}_u^T\left[-\frac{\partial \boldsymbol{K}(\boldsymbol{V}_u\boldsymbol{z}_u)}{\partial(\boldsymbol{V}_u\boldsymbol{z}_u)} + \frac{\partial \boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\boldsymbol{V}_w\boldsymbol{z}_w}{\partial(\boldsymbol{V}_u\boldsymbol{z}_u)}\right]\boldsymbol{V}_u & -\boldsymbol{V}_u^T\boldsymbol{D}\boldsymbol{V}_u & \boldsymbol{V}_u^T\boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)\boldsymbol{V}_w \\ \hline \boldsymbol{V}_w^T\boldsymbol{A}(\boldsymbol{V}_u\boldsymbol{z}_u)^T\boldsymbol{V}_u & \boldsymbol{0} & \boldsymbol{0} \end{array}\right). \tag{5.18}$$

As already observed at the end of Section 5.1, for the first case $(\boldsymbol{Jg}^r(\bar{\boldsymbol{z}}))_{23}^T \neq (\boldsymbol{Jg}^r(\bar{\boldsymbol{z}}))_{31}$. First of all, this means that the system of the ROM obtained this way has a different structure than the system of the FOM, where $(\boldsymbol{Jg}(\bar{\boldsymbol{x}}))_{23}^T = (\boldsymbol{Jg}(\bar{\boldsymbol{x}}))_{31}$ (c.f. Equation (3.50)). Usually, this is something, which is avoided in MOR and a structure preserving method is preferred. Secondly, one looses the saddle point property of the linear solve as part of the Newton iteration in the first case (c.f. Benzi et al. [6])*.
Another observation that could be an indication that the choice $\boldsymbol{V}_v = \boldsymbol{V}_u$ is from a theoretical point of view more appropriate than $\boldsymbol{V}_v \neq \boldsymbol{V}_u$, is the relation between the FOM and ROM bases, c.f. Equation (5.4). Therein,

$$[\boldsymbol{\nu}_u] = \boldsymbol{V}_u^T[\boldsymbol{\varphi}] \quad \text{and} \quad [\boldsymbol{\nu}_v] = \boldsymbol{V}_v^T[\boldsymbol{\varphi}]\,. \tag{5.19}$$

Again, if one wants to preserve the properties of the FOM in the ROM, one should choose $[\boldsymbol{\nu}_v] = [\boldsymbol{\nu}_u]$, i.e. $\boldsymbol{V}_v = \boldsymbol{V}_u$, as in the FOM the same FE basis $[\boldsymbol{\varphi}]$ is used for the displacement/position and for the velocity fields.

This consideration yields to the question of how to choose the basis $\boldsymbol{V}_u = \boldsymbol{V}_v$, as we have snapshot data $\boldsymbol{S}_u, \boldsymbol{S}_v \in \mathbb{R}^{3N\times n}$ available that can be used for the computation. Setting

$$\boldsymbol{S}_{uv}^{\beta\gamma} := [\beta\boldsymbol{S}_u, \gamma\boldsymbol{S}_v] \in \mathbb{R}^{3N\times 2n} \quad \text{with } \beta,\gamma \in [0,1]\,, \tag{5.20}$$

*Note that the block rows one and two of the system need to be permuted to obtain a real saddle point structure. However, the Matlab solvers seem to inherently perform such operations if necessary, since it was observed that it makes no difference whether a permutation was performed prior to the linear solve or not.

one can compute a POD basis

$$\boldsymbol{V}_{uv}^{\beta\gamma} := \boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}} \boldsymbol{S}_{uv}^{\beta\gamma}\right) \in \mathbb{R}^{3N \times r_u} \quad \text{with } \boldsymbol{\mathcal{A}} \in \{\boldsymbol{H_u}, \boldsymbol{I}_{3N}\}, \tag{5.21}$$

which in the limit cases represents the separate POD on the different training data ($\boldsymbol{V}_{uv}^{10} = \boldsymbol{V}_u$ and $\boldsymbol{V}_{uv}^{01} = \boldsymbol{V}_v$), while allowing differently weighted combinations of the training data for $\beta, \gamma \in (0, 1]$.

5.3 Supremizer enrichment

As a last theoretical consideration, the choices for the subspace sizes r_u, r_v, r_p and their combination shall be looked at. Usually, in the single field case, the singular value decay of the computed POD is taken as a measure to determine the appropriate reduced size (c.f. Section 4.2.2 Remark 4). Technically, this is possible here as well. During the offline phase, one obtains the singular values of the POD bases $\boldsymbol{V}_u$, $\boldsymbol{V}_v$, $\boldsymbol{V}_{uv}^{\beta\gamma}$, $\boldsymbol{V}_w$. One could look at each of those and choose the sizes r_u, r_v, r_p accordingly. However, the conditions ensuring that the FOM is solvable and stable, which are fulfilled by the choice of the Taylor-Hood $\mathbb{Q}_2^{(27)} - \mathbb{Q}_1$ elements, need to be met by a ROM as well. Therefore, it seems essential, to have the right ratio not only between the reduced sizes r_u and r_v, but also, maybe even more importantly, between $r_u\,(+r_v)$ and r_p.
The work of Rozza & Veroy (c.f. [4, 40, 41]) on the steady incompressible Navier-Stokes equations, investigates the stability of a ROM constructed by means of a reduced basis, and proposes solutions for the stabilisation of the ROM. These are based on enriching the velocity POD basis with so-called supremizers, which originate from the inf-sup condition of the FOM. The proposed concept has also been successfully applied to optimal control problems with constraints by e.g. Bader et al. [2].
Since the discrete equations of the steady incompressible Navier-Stokes problem yield a saddle point problem as well, though only with two fields, the supremizer enrichment suggests itself for the application to the skeletal muscle model with its incompressibility constraint and three fields. In this section, the basic concept of the supremizer enrichment shall briefly be introduced for the dynamic skeletal muscle model in its discretised form derived in Chapter 3. For more details, the interested reader is referred to Ballarin et al. [4]. Therein, the authors elaborately derive the concept for the POD-Galerkin approximation of the parametrised steady incompressible Navier-Stokes equations and that article served as a guide for the transfer to the incompressible skeletal muscle model. Additionally, Bathe [5] and Brezzi [13] can provide a deeper theoretical background on the conditions (ellipticity and inf-sup condition) ensuring the solvability and stability of mixed finite element models in general.

5.3.1 Definition and computation of (approximate) supremizers

The inf-sup (or LBB) condition for the discretised FOM (c.f. the Jacobian in Equation (3.50)), i.e. in the linear solve) is equivalent to the condition $\ker(\boldsymbol{A}(\boldsymbol{u})) = \{\boldsymbol{0}\}$ and

states

$$\exists\, \beta^h > 0 \text{ such that } \quad \inf_{\boldsymbol{w}\neq\boldsymbol{0}} \sup_{\boldsymbol{u}\neq\boldsymbol{0}} \frac{\boldsymbol{w}^T \mathrm{d}t\boldsymbol{A}(\boldsymbol{u})^T \boldsymbol{u}}{\|\boldsymbol{u}\|_{\mathcal{V}_{\boldsymbol{u}}^h} \|\boldsymbol{w}\|_{\mathcal{V}_p^h}} \geq \beta^h \,. \tag{5.22}$$

It is satisfied by the choice of the Taylor-Hood FE spaces. As explained in [4], for any $\boldsymbol{w} \neq \boldsymbol{0}$, the element that realises the supremum in the inf-sup condition (5.22) is called supremizer. It is given by the solution $\boldsymbol{s} = \boldsymbol{s}(\boldsymbol{u}, \boldsymbol{w})$ of the problem

$$\boldsymbol{H}_{\boldsymbol{u}}\, \boldsymbol{s}(\boldsymbol{u}, \boldsymbol{w}) = \mathrm{d}t\boldsymbol{A}(\boldsymbol{u})\boldsymbol{w} \quad \Longrightarrow \quad \boldsymbol{s}(\boldsymbol{u}, \boldsymbol{w}) = \boldsymbol{H}_{\boldsymbol{u}}^{-1} \mathrm{d}t\boldsymbol{A}(\boldsymbol{u})\boldsymbol{w} \in \mathbb{R}^{3N} \,. \tag{5.23}$$

An equivalent inf-sup condition needs to be satisfied by the ROM. That is

$$\exists\, \beta^r > 0 \text{ such that } \quad \inf_{\boldsymbol{z}_w\neq\boldsymbol{0}} \sup_{\boldsymbol{z}_u\neq\boldsymbol{0}} \frac{\boldsymbol{z}_w^T \,\mathrm{d}t\boldsymbol{A}^r(\boldsymbol{z}_u)^T \boldsymbol{z}_u}{\|\boldsymbol{z}_u\|_{\mathcal{V}_{\boldsymbol{u}}^r} \|\boldsymbol{z}_w\|_{\mathcal{V}_p^r}} \geq \beta^h \,. \tag{5.24}$$

Unfortunately, the reduced spaces $\mathcal{V}_{\boldsymbol{u}}^r, \mathcal{V}_p^r$ obtained by the POD-Galerkin projection as described in the previous section, do not guarantee the fulfilment of the reduced inf-sup condition (5.24), even though their basis functions are obtained through a FOM satisfying the inf-sup condition (5.22), c.f. [4]. For this reason, the authors of [4, 41] suggest to enrich the velocity space in a suitable way, more specifically with (exact or approximate) supremizer solutions.
In order to transfer their approach of the approximate supremizer enrichment (see Ballarin et al. [4, Algorithm 2]) to the dynamic skeletal muscle problem with three fields, the following additions to the workflow proposed so far are made.
Steps 2 and 3 of the offline phase (c.f. Section 4.2.1) are extended by further computations. In step 2, the snapshot data

$$\begin{bmatrix} \boldsymbol{S}_u \\ \boldsymbol{S}_v \\ \boldsymbol{S}_w \end{bmatrix} = \begin{bmatrix} \cdots \boldsymbol{u}_i \cdots \\ \cdots \boldsymbol{v}_i \cdots \\ \cdots \boldsymbol{w}_i \cdots \end{bmatrix} \in \mathbb{R}^{\bar{d}\times n}, \;\; i = 1, ..., n \,, \tag{5.25}$$

is computed in the usual way for all n training parameters. Afterwards, (approximate) supremizer training data $\boldsymbol{S}_s \in \mathbb{R}^{3N\times n}$ is additionally computed by

$$\boldsymbol{s}_i := \boldsymbol{s}(\boldsymbol{u}_i, \boldsymbol{w}_i) = \boldsymbol{H}_{\boldsymbol{u}}^{-1} \mathrm{d}t\boldsymbol{A}(\boldsymbol{u}_i)\boldsymbol{w}_i \quad \text{and} \quad \boldsymbol{S}_s := [\cdots \boldsymbol{s}_i \cdots] \,, \quad i = 1, ..., n \,. \tag{5.26}$$

In step 3 an additional POD is performed on the training data $\boldsymbol{S}_s$, yielding a supremizer basis function matrix

$$\boldsymbol{V}_s := \boldsymbol{H}_{\boldsymbol{u}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{H}_{\boldsymbol{u}}^{\frac{1}{2}} \boldsymbol{S}_s\right) \in \mathbb{R}^{3N\times r_s} \,. \tag{5.27}$$

This way, together with (5.11) and (5.21), the offline step 3 consists of (at least) five different POD computations*.

*Note that in step 2, we compute vectors $\hat{\boldsymbol{s}}_i := \mathrm{d}t\boldsymbol{A}(\boldsymbol{u}_i)\boldsymbol{w}_i = \boldsymbol{H}_{\boldsymbol{u}}\, \boldsymbol{s}_i$ and obtain a training data matrix $\hat{\boldsymbol{S}}_s = \boldsymbol{H}_{\boldsymbol{u}}\, \boldsymbol{S}_s \Longleftrightarrow \boldsymbol{S}_s = \boldsymbol{H}_{\boldsymbol{u}}^{-1} \hat{\boldsymbol{S}}_s$. Therefore, subsequently in step 3, we compute the supremizer POD basis by $\boldsymbol{V}_s = \boldsymbol{H}_{\boldsymbol{u}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{H}_{\boldsymbol{u}}^{-\frac{1}{2}} \hat{\boldsymbol{S}}_s\right)$.

5.3.2 Stability considerations for projection with a supremizer enriched basis

According to Ballarin et al. [4], there are (at least) two factors that influence the overall stability of the system. These are the (i) approximation stability and the (ii) algebraic stability. While the former is related to the inf-sup condition and therefore improved by enriching the POD basis with supremizer solutions, the latter in enhanced by the orthonormality of the basis functions.
However, the separate construction of an orthonormal position POD basis

$$\boldsymbol{V}_u := \boldsymbol{H_u}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{H_u}^{\frac{1}{2}} \boldsymbol{S}_u\right) \in \mathbb{R}^{3N \times r_u}, \tag{5.28}$$

and an orthonormal supremizer POD basis

$$\boldsymbol{V}_s := \boldsymbol{H_u}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{H_u}^{\frac{1}{2}} \boldsymbol{S}_s\right) \in \mathbb{R}^{3N \times r_s}, \tag{5.29}$$

as described in the previous section, and subsequently concatenating these two bases to one basis $\left(\boldsymbol{V}_u \, \boldsymbol{V}_s\right) \in \mathbb{R}^{3N \times r_u + r_s}$, does not yield an orthonormal POD basis. For this case one would obtain

$$\begin{pmatrix} \boldsymbol{V}_u^T \\ \boldsymbol{V}_s^T \end{pmatrix} \boldsymbol{H_u} \left(\boldsymbol{V}_u \, \boldsymbol{V}_s\right) = \begin{pmatrix} \boldsymbol{V}_u^T \boldsymbol{H_u} \boldsymbol{V}_u & \boldsymbol{V}_u^T \boldsymbol{H_u} \boldsymbol{V}_s \\ \boldsymbol{V}_s^T \boldsymbol{H_u} \boldsymbol{V}_u & \boldsymbol{V}_s^T \boldsymbol{H_u} \boldsymbol{V}_s \end{pmatrix} = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{V}_u^T \boldsymbol{H_u} \boldsymbol{V}_s \\ \boldsymbol{V}_s^T \boldsymbol{H_u} \boldsymbol{V}_u & \boldsymbol{I} \end{pmatrix}, \tag{5.30}$$

where the off-diagonal blocks are non-zero. Thus, this way, one gains approximation stability only by the sacrifice of algebraic stability.
In order to construct a supremizer POD basis, such that $\boldsymbol{V}_u$ and $\boldsymbol{V}_s$ are mutually orthogonal, i.e. $\boldsymbol{V}_u^T \boldsymbol{H_u} \boldsymbol{V}_s = \boldsymbol{0}$, one can make use of the idea utilised in the POD-Greedy procedure (see Haasdonk [25]).
Adherent to the short notation introduced in Section 4.2.2, computing a POD basis

$$\boldsymbol{V}_s := \boldsymbol{H_u}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{H_u}^{\frac{1}{2}} \left(\boldsymbol{S}_s - \boldsymbol{V}_u \boldsymbol{V}_u^T \boldsymbol{H_u} \boldsymbol{S}_s\right)\right), \tag{5.31}$$

means, it holds

$$\boldsymbol{H_u}^{\frac{1}{2}} \left(\boldsymbol{S}_s - \boldsymbol{V}_u \boldsymbol{V}_u^T \boldsymbol{H_u} \boldsymbol{S}_s\right) = \boldsymbol{V} \boldsymbol{\Sigma} \bar{\boldsymbol{U}}^T, \quad \text{and} \quad \boldsymbol{V}_s = \boldsymbol{H_u}^{-\frac{1}{2}} \boldsymbol{V}. \tag{5.32}$$

According to the properties of the singular value decomposition

$$\operatorname{span}\left\langle \boldsymbol{H_u}^{\frac{1}{2}} \left(\boldsymbol{S}_s - \boldsymbol{V}_u \boldsymbol{V}_u^T \boldsymbol{H_u} \boldsymbol{S}_s\right)\right\rangle = \operatorname{span}\left\langle \boldsymbol{V} \right\rangle, \tag{5.33}$$

and thus there exists a matrix $\boldsymbol{C} \in \mathbb{R}^{3N \times 3N}$ with

$$\boldsymbol{V} = \boldsymbol{C} \cdot \left(\boldsymbol{H_u}^{\frac{1}{2}} \boldsymbol{S}_s - \boldsymbol{H_u}^{\frac{1}{2}} \boldsymbol{V}_u \boldsymbol{V}_u^T \boldsymbol{H_u} \boldsymbol{S}_s\right). \tag{5.34}$$

It follows that the POD basis vectors of $\boldsymbol{V}_s$ constructed as in Equation (5.31) are ortho-

normal to the POD basis vectors in $\boldsymbol{V}_u$:

$$\begin{aligned}\boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}} \boldsymbol{V}_s = \boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}}^{\frac{1}{2}} \boldsymbol{H}_{\boldsymbol{u}}^{\frac{1}{2}} \boldsymbol{V}_s = \boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}}^{\frac{1}{2}} \boldsymbol{V} &\overset{(5.34)}{=} \boldsymbol{C} \cdot \left(\boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}}^{\frac{1}{2}} \left(\boldsymbol{H}_{\boldsymbol{u}}^{\frac{1}{2}} \boldsymbol{S}_s - \boldsymbol{H}_{\boldsymbol{u}}^{\frac{1}{2}} \boldsymbol{V}_u \boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}} \boldsymbol{S}_s \right) \right) \\ &= \boldsymbol{C} \cdot \left(\boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}} \boldsymbol{S}_s - \underbrace{\boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}} \boldsymbol{V}_u}_{\boldsymbol{I}} \boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}} \boldsymbol{S}_s \right) \\ &= \boldsymbol{C} \cdot \left(\boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}} \boldsymbol{S}_s - \boldsymbol{V}_u^T \boldsymbol{H}_{\boldsymbol{u}} \boldsymbol{S}_s \right) = \boldsymbol{0} \,. \end{aligned} \tag{5.35}$$

Using this approach, it is possible to gain approximation stability while maintaining algebraic stability.
Note that $\boldsymbol{V}_u$ should only contain those modes that are also later on used for building the ROM, i.e. $\boldsymbol{V}_u \in \mathbb{R}^{d \times r_u}$, $r_u \ll l$, in order not to exclude more modes than necessary from the additional space.

5.4 Implementational details of the ROM

KerMor provides a variety of routines and possibilities for the model order reduction of first-order and single-field nonlinear dynamical systems using subspace projection and nonlinear approximation. With those, a general problem of the form introduced in Chapter 4 can be reduced e.g. by means of the described technique of RB approximation combined with the POD. However, for the dynamic skeletal muscle model with its three fields and its block structure as a result from the additional constraint equation and the transformation of the second-order system into a first-order system, the existing routines had to be customised and extended to consider the aspects that were derived from the theoretical perspective in the previous sections of this chapter.
To begin with, the precomputation of training data during the offline phase 2 in the function `off2_genTrainingData()` of the class `BaseFullModel`, was extended by the additional calculation of approximate supremizer solutions as defined in Equation (5.26). For the evaluation itself, a function `getSupremizer()` was implemented in the same way as the existing functions `evaluate()` and `getStateJacobianImpl()` describing the dynamics of the `+muscle` class. Since this computation requires the inner product matrix $\boldsymbol{H}_{\boldsymbol{u}}$, this was computed (together with the inner product matrix $\boldsymbol{H}_p$) adopting the same procedure as for the mass and damping matrix inside the `+muscle.System` class, which is a subclass of the already mentioned class `models.BaseSecondOrderSystem`. The function `off3_computeReducedSpace()` of the class `BaseFullModel` subsequently initiates the computation of the POD on the precomputed training data. In order to account for the data of the three different fields and the approximate supremizers, this routine was modified such that it runs over the four types of training data and each time calls the function `generateReducedSpace()` with the correct corresponding arguments (e.g. which dof to include, which ones to exclude, or which norm to choose). The arguments are determined inside the `spacereduction.BaseSpaceReducer` class, where the default choice can be altered by specifying (i.e. overwriting the superclass function) a `configureModelFinal()` function in each of the three examples, i.e. the `+muscle` subclasses. The last important contribution to KerMor in the context of the reduced model routines is the extension of the assembly procedure, i.e. the building of the reduced model,

which is initiated by the function `buildReducedModel()` of the class `BaseFullModel`, or more specifically, the function `build()` inside the `ReducedSystem` class. The function `assembleProjectionMatrices()` was extended to account for multiple projection spaces and by the option of choosing those according to previously specified arguments, which are passed to the `buildReducedModel()` call. The subsequent assembly of the reduced system as it was derived in Section 5.1 (i.e. reduced initial condition, reduced mass and damping matrix, evaluation of the reduced right-hand side function `odefun` and the reduced Jacobian) is adapted by specifying several functions in a subclass `ReducedSecondOrderSystem`, which overwrite and/or make use of the existing corresponding superclass functions.

6 Analysis of the FOM

This chapter intends to investigate and verify the full-order model. For that reason, Section 6.1 introduces three types of examples with increasing complexity. Those examples are additionally used for the investigation of the POD-Galerkin reduction in Chapter 7. Therefore, this chapter could also be considered as a documentation of the offline phase and is relatively detailed. Since already among these FOM simulations, stability issues occur, instabilities arising later in the POD-Galerkin ROM possibly originate from the FOM simulation. Hence Section 6.2 describes the issues that could already be observed for some of the FOM simulations. It also provides more details on the solution procedure, like Newton tolerance and maximum number of time steps and the like and tries to find explanations for the problems that occur. As a final verification of the FOM a mesh convergence study was performed for all examples and the results are summarised in Section 6.3.

6.1 Testing examples

This section introduces three examples with increasing complexity, which are being used throughout this work. The complexity increase is with respect to the material model as well as the applied boundary conditions. The examples have different purpose from validation of the FE code over investigating suitable solution and reduction methods to investigations on the influence of the increasing complexity on the chosen methods.

6.1.1 Quasi-static examples with an analytical solution

In this section, three different examples, for which an analytical solution can be derived in a quasi-static setting, are introduced. They will be used to compare the simulated time-converged state with the quasi-static analytical solution. For the hyperelastic material the nonlinear, but still simple Neo-Hooke material law (c.f. Equation (2.42)) is chosen. The purpose of these three examples is mainly for validation of the implementations within the FE code. They will be referred to as Example(s) 1A, 1B, 1C.
The general equations for an incompressible Neo-Hooke solid in the quasi-static setting are (c.f. Equation (3.10))

$$\nabla \cdot \boldsymbol{P} = 0 \qquad \text{in } \Omega_0\,, \tag{6.1}$$

$$\det \boldsymbol{F} - 1 = 0 \qquad \text{in } \Omega_0\,, \tag{6.2}$$

$$\boldsymbol{x} = \boldsymbol{x}^{\mathrm{D}} \qquad \text{on } \Gamma^{\mathrm{D}}\,, \tag{6.3}$$

$$\boldsymbol{P}\boldsymbol{N} = \boldsymbol{t} \qquad \text{on } \Gamma^{\mathrm{N}}\,, \tag{6.4}$$

$$\boldsymbol{P} = \boldsymbol{P}^{\mathrm{NH}}(\boldsymbol{F}) = 2c_{10}J^{-\frac{2}{3}}\left(\boldsymbol{F} - \frac{1}{3}I_1\boldsymbol{F}^{-T}\right) - pJ\boldsymbol{F}^{-T}\,. \tag{6.5}$$

In the following, the basis notation, i.e. $\boldsymbol{e}_i$ and $(\boldsymbol{e}_i \otimes \boldsymbol{e}_j)$ are omitted for improved readability and for saving space.

6.1.1.1 Example 1A: Uniaxial extension

Derivation of the analytic solution

Let $\boldsymbol{X} \in \Omega_0 := [0, L_x] \times [-L_y/2, L_y/2] \times [-L_z/2, L_z/2]$. The motion function or actual position for an incompressible uniaxial extension in positive x-direction about a stretch $\lambda := l_x/L_x$ is given as

$$\begin{aligned}
\boldsymbol{x} &= \begin{pmatrix} \lambda X_1 \\ \frac{1}{\sqrt{\lambda}} X_2 \\ \frac{1}{\sqrt{\lambda}} X_3 \end{pmatrix} \quad \Longrightarrow \quad \boldsymbol{F} = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \frac{1}{\sqrt{\lambda}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{\lambda}} \end{pmatrix} = \boldsymbol{F}^T \,, \\
\boldsymbol{C} &= \begin{pmatrix} \lambda^2 & 0 & 0 \\ 0 & \frac{1}{\lambda} & 0 \\ 0 & 0 & \frac{1}{\lambda} \end{pmatrix} \,, \quad \boldsymbol{F}^{-1} = \begin{pmatrix} \frac{1}{\lambda} & 0 & 0 \\ 0 & \sqrt{\lambda} & 0 \\ 0 & 0 & \sqrt{\lambda} \end{pmatrix} = \boldsymbol{F}^{-T} \,, \\
J &= \lambda \frac{1}{\sqrt{\lambda}} \frac{1}{\sqrt{\lambda}} = 1 \,, \qquad I_1 = \lambda^2 + \frac{2}{\lambda} \,.
\end{aligned} \tag{6.6}$$

Furthermore, for uniqueness of the solution, DIRICHLET boundary conditions need to be applied. Here, all nodes on the zero x-face are fixed in x-direction and additionally, the middle node of that face is fixed in y- and z-direction (see Figure 6.1 below).
In order to obtain the analytical expression of the pressure p in terms of the corresponding stretch λ, the stress tensor for the specific deformation needs to be derived.

$$\begin{aligned}
\boldsymbol{P}^{\mathrm{NH}}(\boldsymbol{F}) &\overset{(6.5)}{\underset{(6.6)}{=}} 2c_{10} \begin{pmatrix} \lambda - \frac{1}{3}\left(\lambda^2 + \frac{2}{\lambda}\right)\frac{1}{\lambda} & 0 & 0 \\ 0 & \frac{1}{\sqrt{\lambda}} - \frac{1}{3}\left(\lambda^2 + \frac{2}{\lambda}\right)\sqrt{\lambda} & 0 \\ 0 & 0 & \frac{1}{\sqrt{\lambda}} - \frac{1}{3}\left(\lambda^2 + \frac{2}{\lambda}\right)\sqrt{\lambda} \end{pmatrix} \\
&\quad - \begin{pmatrix} \frac{1}{\lambda} p & 0 & 0 \\ 0 & \sqrt{\lambda}\, p & 0 \\ 0 & 0 & \sqrt{\lambda}\, p \end{pmatrix} \\
&= \begin{pmatrix} \frac{4}{3}c_{10}\lambda - \frac{4}{3}c_{10}\frac{1}{\lambda^2} - \frac{1}{\lambda} p & 0 & 0 \\ 0 & \frac{2}{3}c_{10}\frac{1}{\sqrt{\lambda}} - \frac{2}{3}c_{10}\lambda^2\sqrt{\lambda} - \sqrt{\lambda}\, p & 0 \\ 0 & 0 & P_{33} = P_{22} \end{pmatrix}
\end{aligned} \tag{6.7}$$

As for the unconstrained, uniaxial extension in x-direction the y- and z-directions are free, the corresponding stress components $P_{22} = P_{33} \overset{!}{=} 0$. Thus, p is obtained as

$$0 = \frac{2}{3}c_{10}\frac{1}{\sqrt{\lambda}} - \frac{2}{3}c_{10}\lambda^2\sqrt{\lambda} \quad \Longleftrightarrow \quad p = \frac{2}{3}c_{10}\frac{1}{\lambda} - \frac{2}{3}c_{10}\lambda^2 = \frac{2}{3}c_{10}\left(\frac{1}{\lambda} - \lambda^2\right) . \tag{6.8}$$

Note that the pressure does not depend on the location, i.e. that it is constant over the domain Ω.
If, instead of DIRICHLET position boundary conditions on the positive x-face, it is necessary or wanted to apply equivalent NEUMANN traction boundary conditions describing

the same problem, the traction force vector on the face with unit normal $\boldsymbol{N} = (1, 0, 0)^T$ is required to be

$$\boldsymbol{t} = \boldsymbol{P}^{\mathrm{NH}}\boldsymbol{N} = \boldsymbol{P}^{\mathrm{NH}} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} P_{11} \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 2c_{10}\left(\lambda - \frac{1}{\lambda^2}\right) \\ 0 \\ 0 \end{pmatrix} . \tag{6.9}$$

Now, for specific choices for the stretch λ and for the material parameter c_{10}, the values for the corresponding traction force entry t_1 and resulting pressure p can be calculated.

Simulated test case

For the tests to be performed with this Example 1A, the NEO-HOOKE parameter was set to $c_{10} = 17.85\,\mathrm{e}{-3}\,\mathrm{MPa}$ and the dimensions $L_x = 1\,\mathrm{mm}$, $L_y = L_z = 0.2\,\mathrm{mm}$ together with a stretch value $\lambda = 1.2$, as visualised in Figure 6.1, were chosen. Insertion of these values into Equations (6.9) and (6.8) derived above, yields a NEUMANN traction force entry of $t_1 = 18.048\,\mathrm{e}{-3}\,\mathrm{MPa}$ and a corresponding pressure of $p = -7.219\,\mathrm{e}{-3}\,\mathrm{MPa}$.

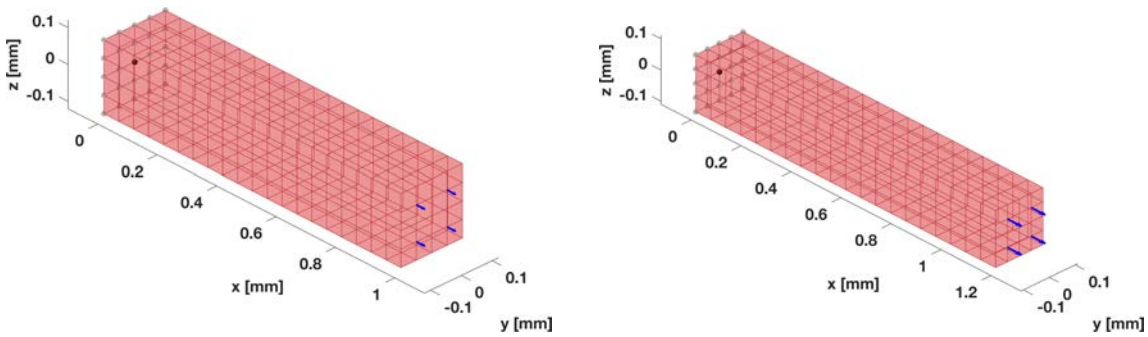


Figure 6.1: *The setup for the uniaxial extension Example 1A. Here exemplarily shown with a spatial discretisation* $\mathrm{d}x = 0.1$ *resulting in* 40 *elements. On the zero* x*-face, the zero-*DIRICHLET *boundary conditions are visible, where the grey spheres indicate the nodes, where only the* x*-direction is fixed, while the black sphere indicates that that node is fixed in all three directions. Moreover, the blue arrows indicate the applied* NEUMANN *boundary conditions on the positive* x*-face. Left: Reference configuration. Right: Final configuration for the stretch* $\lambda = 1.2$.

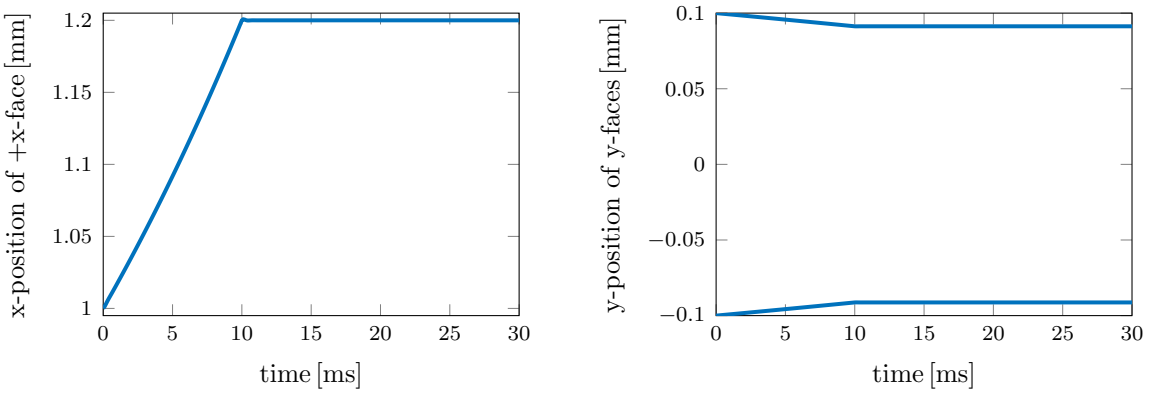


Figure 6.2: *The positions of the faces over time for the uniaxial extension Example 1A. Left: x-position of the positive x-face. It is clearly visible that the converged solution fulfils $x_1 = \lambda X_1 = 1.2 \cdot 1 = 1.2$. Right: y-position of the positive (top line) and negative (bottom line) y-face. The analytic quasi-static solution $x_2 = \frac{1}{\sqrt{\lambda}} X_2 \approx \pm 0.913 \cdot 0.1 = \pm 0.0913$ is also the result of the converged dynamic simulation. Note that the results are the same for the z-direction.*

As already mentioned, the time-converged state shall be compared with this quasi-static solution. Therefore, during an entire simulation time of $T = 30\,\text{ms}$ and a time step size $\mathrm{d}t = 0.1\,\text{ms}$, the traction force $\boldsymbol{t}$ was applied linearly increasing within the first $10\,\text{ms}$ and subsequently kept constant until the end. The final pressure, the displacement in x-direction of the loose end, i.e. the positive x-face (mean over nodal values), the displacements in y-direction of the negative and positive y-faces (mean over nodal values) and the displacements in z-direction of the negative and positive z-faces (mean over nodal values) are extracted and used for the comparison (see Figures 6.2 and 6.3).

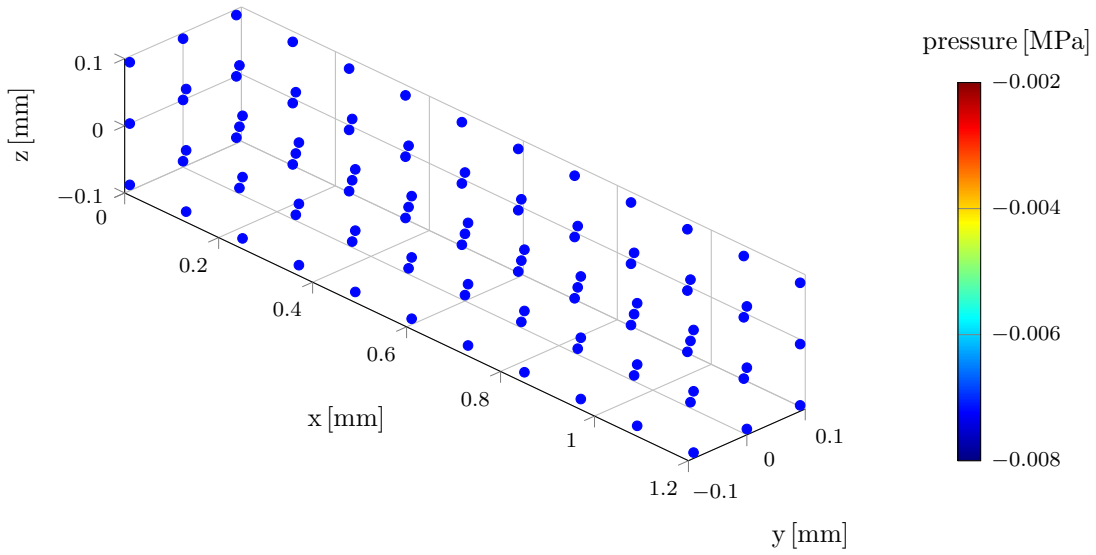


Figure 6.3: *The final pressure is constant over the domain and corresponds to the analytic solution $p = -0.0072\,MPa$.*

6.1.1.2 Example 1B: Simple shear

Derivation of the analytic solution

Let $\boldsymbol{X} \in \Omega_0 := [0, L_x] \times [0, L_y] \times [0, L_z]$. The motion function for an incompressible simple shear deformation in the x-z-plane about a shear strain γ is given as

$$\begin{aligned}
\boldsymbol{x} &= \begin{pmatrix} X_1 + \gamma X_3 \\ X_2 \\ X_3 \end{pmatrix} \quad \Longrightarrow \quad \boldsymbol{F} = \begin{pmatrix} 1 & 0 & \gamma \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \boldsymbol{F}^T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \gamma & 0 & 1 \end{pmatrix}, \\
\boldsymbol{C} &= \begin{pmatrix} 1 & 0 & \gamma \\ 0 & 1 & 0 \\ \gamma & 0 & \gamma^2 + 1 \end{pmatrix}, \quad \boldsymbol{F}^{-1} = \begin{pmatrix} 1 & 0 & -\gamma \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \boldsymbol{F}^{-T} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\gamma & 0 & 1 \end{pmatrix}, \\
J &= 1, \qquad\qquad I_1 = \gamma^2 + 3.
\end{aligned} \tag{6.10}$$

Furthermore, for uniqueness of the solution, DIRICHLET boundary conditions need to be applied. Here, all nodes on the zero z-face are fixed in x-, y- and z-direction (see Figure 6.4 below).
In order to obtain the analytical expression of the pressure p in terms of the corresponding shear strain γ, again the stress tensor for the specific deformation needs to be derived.

$$\begin{aligned}
\boldsymbol{P}^{\mathrm{NH}}(\boldsymbol{F}) &\overset{(6.5)}{\underset{(6.10)}{=}} 2c_{10} \begin{pmatrix} 1 - \frac{1}{3}(\gamma^2 + 3) & 0 & \gamma \\ 0 & 1 - \frac{1}{3}(\gamma^2 + 3) & 0 \\ \frac{1}{3}(\gamma^2 + 3)\,\gamma & 0 & 1 - \frac{1}{3}(\gamma^2 + 3) \end{pmatrix} - \begin{pmatrix} p & 0 & 0 \\ 0 & p & 0 \\ -\gamma p & 0 & p \end{pmatrix} \\
&= \begin{pmatrix} -\frac{2}{3}c_{10}\gamma^2 - p & 0 & 2c_{10}\gamma \\ 0 & -\frac{2}{3}c_{10}\gamma^2 - p & 0 \\ \frac{2}{3}c_{10}\gamma^3 + 2c_{10}\gamma + \gamma p & 0 & -\frac{2}{3}c_{10}\gamma^2 - p \end{pmatrix}
\end{aligned} \tag{6.11}$$

For this simple shear problem, boundary conditions are applied in x- and z-direction while the y-direction is free. Thus, the corresponding stress component $P_{22} \overset{!}{=} 0$ and p is obtained as

$$0 = -\frac{2}{3}c_{10}\gamma^2 - p \quad \Longleftrightarrow \quad p = \frac{2}{3}c_{10}\gamma^2 . \tag{6.12}$$

Again, the pressure does not depend on the location, i.e. it is constant over the domain.
If, instead of DIRICHLET position boundary conditions, it is necessary or wanted to apply equivalent NEUMANN traction boundary conditions describing the same problem, the traction vectors on the faces with unit normals $\boldsymbol{N} = (1, 0, 0)^T$, $(-1, 0, 0)^T$ and $(0, 0, 1)^T$ are required to be

$$\boldsymbol{t}^{x^{\pm}} = \boldsymbol{P}^{\mathrm{NH}} \begin{pmatrix} \pm 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \pm 2c_{10}\gamma \end{pmatrix}, \quad \boldsymbol{t}^{z^{+}} = \boldsymbol{P}^{\mathrm{NH}} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 2c_{10}\gamma \\ 0 \\ 0 \end{pmatrix}. \tag{6.13}$$

There is no traction force on the y-faces. Now, for specific choices for the shear strain γ and for the material parameter c_{10}, the values for the corresponding traction force vectors and resulting pressure p can be calculated.

Simulated test case

For the tests to be performed with this Example 1B, the NEO-HOOKE parameter was set to $c_{10} = 17.85\,\mathrm{e}{-3}\,\mathrm{MPa}$ and the dimensions $L_x = 1\,\mathrm{mm}$, $L_y = L_z = 0.2\,\mathrm{mm}$ together with a shear strain $\gamma = 0.35$, as visualised in Figure 6.4, were chosen. Insertion of these values into Equations (6.13) and (6.12) derived above, yields NEUMANN traction force vector entries of $t_3^{x^\pm} = \pm 12.495\,\mathrm{e}{-3}\,\mathrm{MPa}$ and $t_1^{z^+} = 12.495\,\mathrm{e}{-3}\,\mathrm{MPa}$ and a corresponding pressure of $p = -1.458\,\mathrm{e}{-3}\,\mathrm{MPa}$.

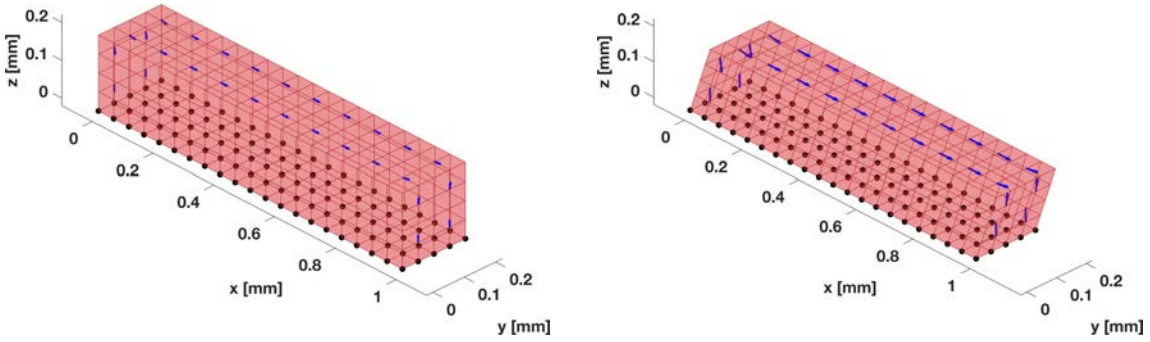


Figure 6.4: *The setup for the simple shear Example 1B. Here, exemplarily shown with a spatial discretisation* $\mathrm{d}x = 0.1$ *resulting in 40 elements. On the zero z-face, the zero-*DIRICHLET *boundary conditions are visible, where the black spheres indicate that all nodes are fixed in all three directions. Moreover, the blue arrows indicate the applied* NEUMANN *boundary conditions on the zero and the positive x-face as well as on the positive z-face. Left: Reference configuration. Right: Final configuration for the shear strain* $\gamma = 0.35$.

As already mentioned, the time-converged state shall be compared with this quasi-static solution. Therefore, (like in Example 1A) during an entire simulation time of $T = 30\,\mathrm{ms}$ and a time step size $\mathrm{d}t = 0.1\,\mathrm{ms}$, the traction forces $\boldsymbol{t}^{x^\pm}$ and $\boldsymbol{t}^{z^+}$ were applied linearly increasing within the first 10 ms before keeping them constant until the end. The final pressure, the displacement in x-direction over time of the top, i.e. the positive z-face (mean over nodal values) and the z- versus x-position of the right, i.e. the positive x-face (mean over nodal values) are extracted and used for the comparison (see Figures 6.5 and 6.6).

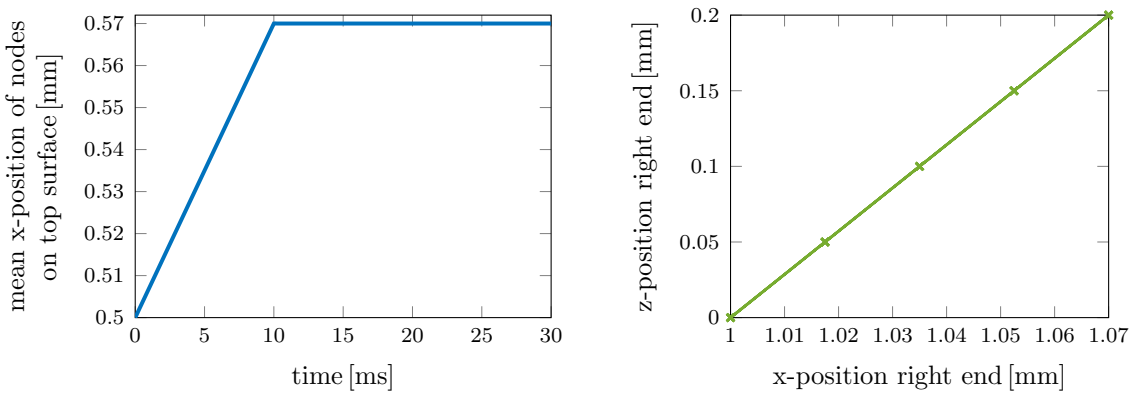


Figure 6.5: *The extracted results for the simple shear Example 1B.*
Left: The mean x-position of all nodes on the top surface ($X_3 = 0.2$) over time. It is $x_1 = X_1 + \gamma X_3 = X_1 + 0.35 \cdot 0.2 = X_1 + 0.07$. As $X_1 \in [0, 1]$, one can see that the converged dynamic solution corresponds to the analytic quasi-static solution.
Right: As a simple representation to show that the x-position of the other horizontal node layers ($X_3 < 0.2$) is correct as well and that $x_3 = X_3$ is satisfied, the x-position of the right, i.e. the positive x-face, is plotted versus the z-position of these nodes.

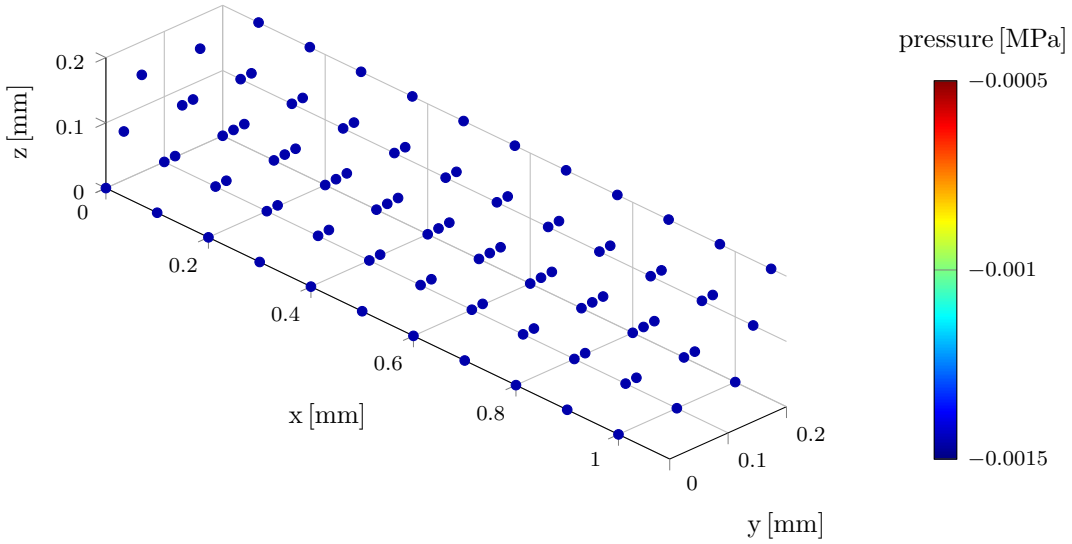


Figure 6.6: *The final pressure is constant over the domain and corresponds to the analytic solution $p = -0.00146\,MPa$.*

6.1.1.3 Example 1C: Pure shear

Derivation of the analytic solution

Let $\boldsymbol{X} \in \Omega_0 := [-L_x/2, L_x/2] \times [-L_y/2, L_y/2] \times [-L_z/2, L_z/2]$. The motion function or actual position for an incompressible pure shear deformation with elongation about a stretch $\lambda := l_z/L_z$ in z-direction while being shortened perpendicularly in y-direction is

given as

$$\begin{aligned}
\boldsymbol{x} &= \begin{pmatrix} X_1 \\ \frac{1}{\lambda} X_2 \\ \lambda X_3 \end{pmatrix} \quad \Longrightarrow \quad \boldsymbol{F} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\lambda} & 0 \\ 0 & 0 & \lambda \end{pmatrix} = \boldsymbol{F}^T \,, \\
\boldsymbol{C} &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\lambda^2} & 0 \\ 0 & 0 & \lambda^2 \end{pmatrix} , \boldsymbol{F}^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \frac{1}{\lambda} \end{pmatrix} = \boldsymbol{F}^{-T} \,, \\
J &= 1 \frac{1}{\lambda} \lambda = 1 \,, \qquad I_1 = 1 + \lambda^2 + \frac{1}{\lambda^2} \quad .
\end{aligned} \tag{6.14}$$

In order to obtain the analytical expression of the pressure p in terms of the corresponding stretch λ, the stress tensor for the specific deformation needs to be derived.

$$\begin{aligned}
\boldsymbol{P}^{\mathrm{NH}}(\boldsymbol{F}) \overset{(6.5)}{\underset{(6.14)}{=}} \; & 2c_{10} \begin{pmatrix} 1 - \frac{1}{3}\left(1 + \lambda^2 + \frac{1}{\lambda^2}\right) & 0 & 0 \\ 0 & \frac{1}{\lambda} - \frac{1}{3}\left(1 + \lambda^2 + \frac{1}{\lambda^2}\right)\lambda & 0 \\ 0 & 0 & \lambda - \frac{1}{3}\left(1 + \lambda^2 + \frac{1}{\lambda^2}\right)\frac{1}{\lambda} \end{pmatrix} \\
& - \begin{pmatrix} p & 0 & 0 \\ 0 & \lambda p & 0 \\ 0 & 0 & \frac{1}{\lambda} p \end{pmatrix} \quad = \begin{pmatrix} P_{11}^{NH,ps} & 0 & 0 \\ 0 & P_{22}^{NH,ps} & 0 \\ 0 & 0 & P_{33}^{NH,ps} \end{pmatrix} ,
\end{aligned} \tag{6.15}$$

with

$$\begin{aligned}
P_{11}^{NH,ps} &:= 2c_{10} - \frac{2}{3} c_{10} \left(1 + \lambda^2 + \frac{1}{\lambda^2}\right) - p \,, \\
P_{22}^{NH,ps} &:= 2c_{10} \left(\frac{1}{\lambda} - \frac{1}{3}\left(1 + \lambda^2 + \frac{1}{\lambda^2}\right)\lambda\right) - \lambda p \,, \\
P_{33}^{NH,ps} &:= 2c_{10} \left(\lambda - \frac{1}{3}\left(1 + \lambda^2 + \frac{1}{\lambda^2}\right)\frac{1}{\lambda}\right) - \frac{1}{\lambda} p \,.
\end{aligned}$$

For the pure shear problem, zero-DIRICHLET boundary conditions are applied on the positive and negative x-face in x-direction and NEUMANN boundary conditions on the faces in z-direction (c.f. Figure 6.7), while the faces in y-direction are free. Thus, the corresponding stress component $P_{22} \overset{!}{=} 0$ and p is obtained as

$$p = 2c_{10} \frac{1}{\lambda} \left(\frac{1}{\lambda} - \frac{1}{3}\left(1 + \lambda^2 + \frac{1}{\lambda^2}\right)\lambda\right) = 2c_{10} \left(\frac{1}{\lambda^2} - \frac{1}{3}\left(1 + \lambda^2 + \frac{1}{\lambda^2}\right)\right) . \tag{6.16}$$

Again, the pressure does not depend on the location, i.e. it is constant over the domain Ω.

If, instead of DIRICHLET position boundary conditions on the z-faces, it is necessary or wanted to apply the equivalent NEUMANN traction boundary conditions describing the same problem, the traction vectors on the faces with unit normals $\boldsymbol{N} = (0, 0, 1)^T$ and

$(0, 0, -1)^T$ are required to be

$$\boldsymbol{t}^{z^\pm} = \boldsymbol{P}^{\mathrm{NH}} \begin{pmatrix} 0 \\ 0 \\ \pm 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \pm 2c_{10}\left(\lambda - \frac{1}{\lambda^3}\right) \end{pmatrix}. \tag{6.17}$$

Now, for specific choices for the stretch λ and for the material parameter c_{10}, the values for the corresponding traction force vectors and resulting pressure p can be calculated.

Simulated test case

For the tests to be performed with this Example 1C, the NEO-HOOKE parameter was set to $c_{10} = 5.0\,\mathrm{e}{-3}\,\mathrm{MPa}$ and the dimensions $L_x = L_y = 0.2\,\mathrm{mm}$, $L_z = 1\,\mathrm{mm}$ together with a stretch value $\lambda = 2$, as visualised in Figure 6.7 were chosen. Insertion of these values into Equations (6.17) and (6.16) derived above, yields NEUMANN traction force vector entries of $t_3^{z^\pm} = \pm 18.75\,\mathrm{e}{-3}\,\mathrm{MPa}$ and a corresponding pressure of $p = -15.0\,\mathrm{e}{-3}\,\mathrm{MPa}$.
As already mentioned, the time-converged state shall be compared with this quasi-static solution. Therefore, (like in Examples 1A and 1B) during an entire simulation time of $T = 30\,\mathrm{ms}$ and a time step size $\mathrm{d}t = 0.1\,\mathrm{ms}$, the traction forces $\boldsymbol{t}^{z^\pm}$ were applied linearly increasing within the first 10 ms before keeping them constant until the end.

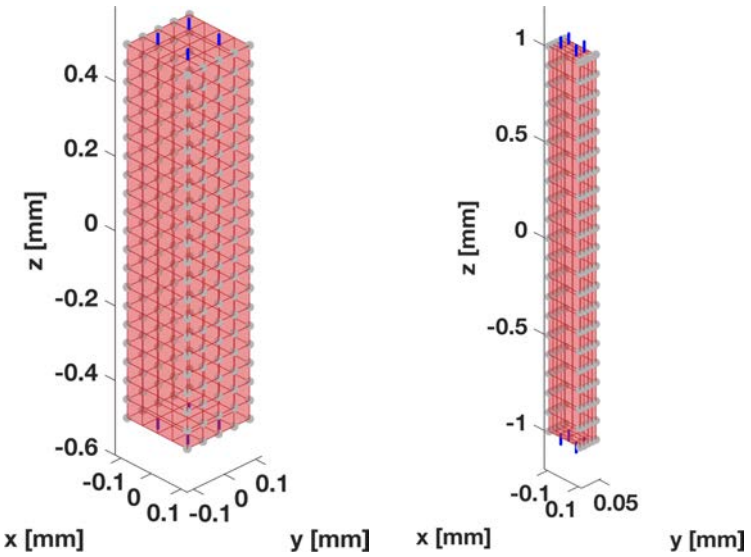


Figure 6.7: *The setup for the pure shear Example 1C. Here exemplarily shown with a spatial discretisation* $\mathrm{d}x = 0.1$ *resulting in 40 elements. On the positive and negative x-face, the zero-*DIRICHLET *boundary conditions are visible, where the grey spheres indicate that the nodes are fixed in x-direction only. Moreover, the blue arrows indicate the applied* NEUMANN *boundary conditions on the negative and positive z-face. Left: Reference configuration. Right: Final configuration for the stretch* $\lambda = 2$.

The final pressure, the displacement in y-direction over time of the y-faces (mean over nodal values) and in z-direction over time of the z-faces (mean over nodal values) are extracted and used for the comparison (see Figures 6.8 and 6.9).

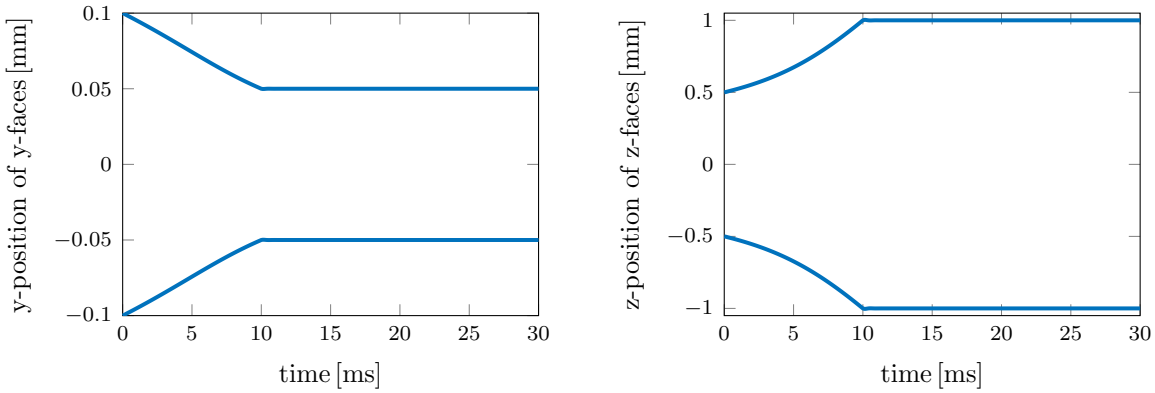


Figure 6.8: *The extracted results for the pure shear Example 1C. Clearly, the converged dynamic solution corresponds to the analytic quasi-static solution.*
Left: The mean y-position of all nodes on the positive (top line) and negative (bottom line) y-face over time. It is $x_2 = \frac{1}{\lambda} X_2 = \frac{1}{2} \cdot (\pm 0.1) = \pm 0.05$.
Right: The mean z-position of all nodes on the positive (top line) and negative (bottom line) z-face over time. It is $x_3 = \lambda X_3 = 2 \cdot (\pm 0.5) = \pm 1.0$.

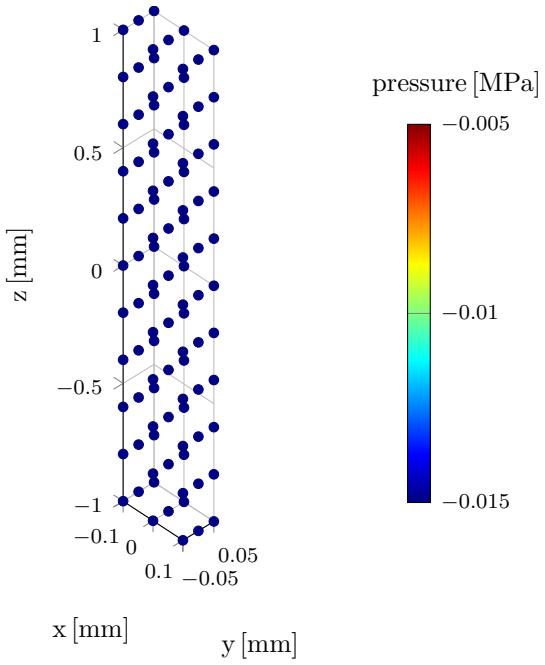


Figure 6.9: *The final pressure is constant over the domain and corresponds to the analytic solution* $p = -0.015$ *MPa.*

6.1.2 Dynamic example with a simple material law

This example, with a moderate complexity, was chosen for tests concerning the stability of the FOM as well as for testing the performance of different POD-Galerkin ROM. It will be referred to as Example 2 and is supposed to fulfil the following requirements.

- Have the size of a human muscle.

- Exhibit a deformation that includes extension and shear modes, in order to cover the full possible range of motions.
- The deformation over the domain has to result in a pressure distribution that is of higher order than linear and a distribution of higher order than quadratic for the displacement and the velocity. This is important for the subsequent convergence study in Section 6.3, since otherwise, the solution would already lie in the finite element solution space and is independent of the mesh.
- It should have the possibility to vary at least one or even more parameters, such that different training data can be collected during the MOR offline phase (see Section 4.2.1).

For those reasons, a cubic geometry of $20 \times 20 \times 20\,\mathrm{mm}$ is chosen, which is fixed with zero-DIRICHLET boundary conditions in all three directions on the negative x-face, while different traction boundary conditions can be applied to the positive x-face. This setup is shown in Figure 6.10 on the left. By constraining the nodes on the negative x-face in all directions, a non-uniform lateral contraction is achieved.

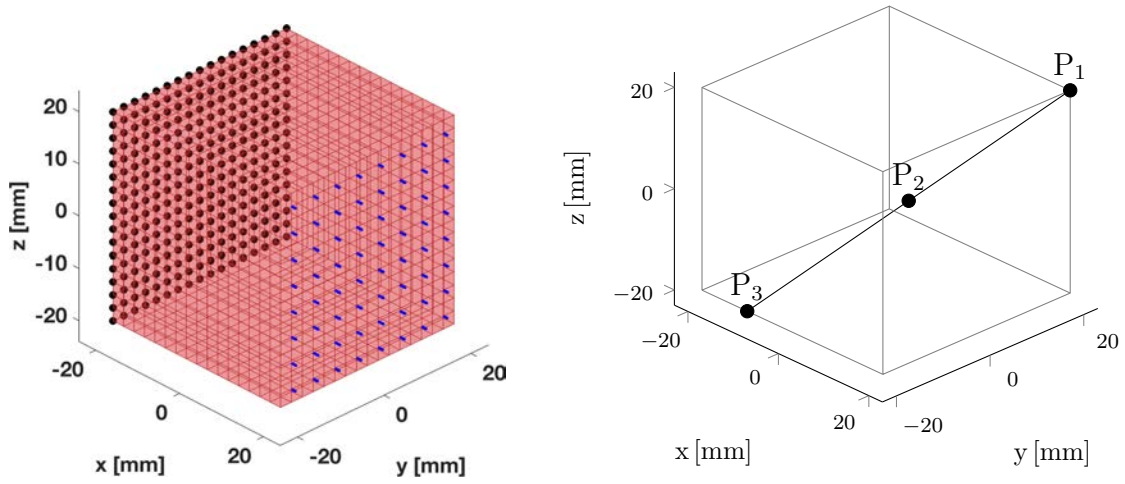


Figure 6.10: *Left: The geometrical setup of Example 2. Zero-*DIRICHLET *boundary conditions are applied to the nodes belonging to the negative x-face, while the positive x-face is subject to a traction* NEUMANN *boundary condition.*
Right: The location of the three points $P_1 := (20, 20, 20)$, $P_2 := (5, 0, 0)$ and $P_3 := (-10, -20, -20)$ chosen for evaluation purpose.

The muscle material density is assumed to be $1.1\,\mathrm{e}{-3}\,\mathrm{g/mm^3}$ (c.f. [32]). As material law, the rather simple, yet already nonlinear NEO-HOOKE material is chosen. To obtain a meaningful value for the parameter c_{10}, the model is fitted to the experimental data from Takaza et al. [46], who performed experiments on the pig longissimus dorsi muscle. There, the true stress in x-direction, i.e. $T_{11}(\lambda)$, during a uniaxial extension experiment is documented. As derived in Example 1A (c.f. Equations (6.7)(6.8)), P_{11}^{NH} and p are

$$P_{11}^{\mathrm{NH}} = \frac{4}{3}c_{10}\lambda - \frac{4}{3}c_{10}\frac{1}{\lambda^2} - \frac{1}{\lambda}p \quad \text{and} \quad p = \frac{2}{3}c_{10}\left(\frac{1}{\lambda} - \lambda^2\right). \tag{6.18}$$

Therefore,

$$P_{11}^{\mathrm{NH}}(\lambda) = 2\,c_{10}\left(\lambda - \frac{1}{\lambda^2}\right) \quad \Longrightarrow \quad T_{11}^{\mathrm{NH}}(\lambda) = \lambda P_{11}^{\mathrm{NH}}(\lambda) = 2\,c_{10}\left(\lambda^2 - \frac{1}{\lambda}\right). \tag{6.19}$$

This expression can now be fitted to the experimental data using the MATLAB curve fitting toolbox, which yields $c_{10} = 38.18\,\mathrm{e}{-3}\,\mathrm{MPa}$. As can be seen in Figure 6.11, the NEO-HOOKE material law is not a good choice to approximate skeletal muscle material as the exponential increase (J-like curve) cannot be captured. However, for the purpose of Example 2, this is sufficiently complex.
Three different angles were chosen for the application of the NEUMANN force boundary conditions on the positive x-face. These were $0\,^\circ$ (BC1), $90\,^\circ$ (BC2) and $45\,^\circ$ (BC3), resulting in final, deformed configurations as depicted in Figure 6.12. The total simulation time covered 100 ms and the chosen time step size was $\mathrm{d}t = 0.1\,\mathrm{ms}$. As in Example 1, the traction force boundary condition was increased linearly up to a chosen maximum value. Here it was applied within 50 ms in the interval $[10, 60]$ ms and subsequently kept constant until the end.

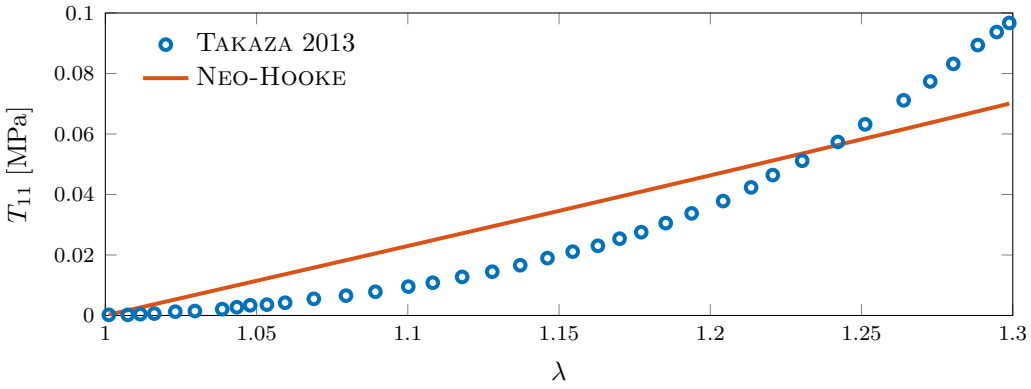


Figure 6.11: *The fit of the* NEO-HOOKE *material to the muscle data from Takaza et al. [46].*

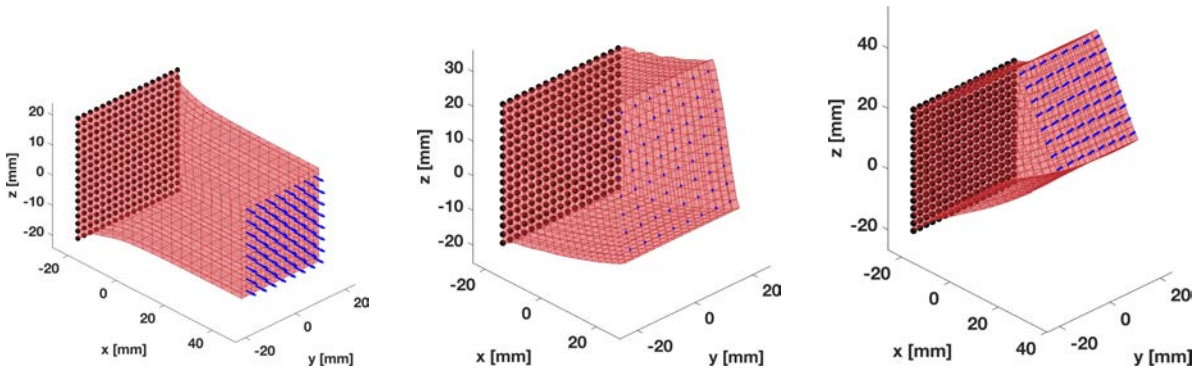


Figure 6.12: *Examples for the final, deformed configurations of Example 2. Here exemplarily shown with a spatial discretisation* $\mathrm{d}x = 5$ *that results in 512 elements and* $(2 \times 13\,872 + 729 =)\ 28\,473$ *dof.*
Left: BC1, the applied uniaxial force here is 0.1 *MPa.*
Middle: BC2, the applied shear force here is 0.01 *MPa.*
Right: BC3, the applied combined uniaxial and shear force here is 0.05 *MPa.*

Since dynamic processes are considered, also the deformation states in between the reference and the final configuration are of interest. Furthermore, deformation and pressure for this Example 2 should vary in space. For these reasons, the positions and the velocities in x-, y- and z-direction for three points roughly on a diagonal through the cube (c.f. Figure 6.10) are monitored over time and plotted in a two-dimensional plot. In these plots the inertia and damping effects become visible. The points are $P_1 := (20, 20, 20)$ (dashed lines), $P_2 := (5, 0, 0)$ (dotted lines) and $P_3 := (-10, -20, -20)$ (dash-dot lines). Additionally, the final pressure distribution over the domain is displayed at each node. Below, Figures 6.13–6.15 show the plots for the three types of boundary conditions for the case of a spatial discretisation $\mathrm{d}x = 5$. Note that due to the approximation of the pressure with linear finite elements, while the displacements are approximated with quadratic elements, the number of nodes in the pressure plots naturally is less than in the plots in Figure 6.12 showing the final, deformed configurations.

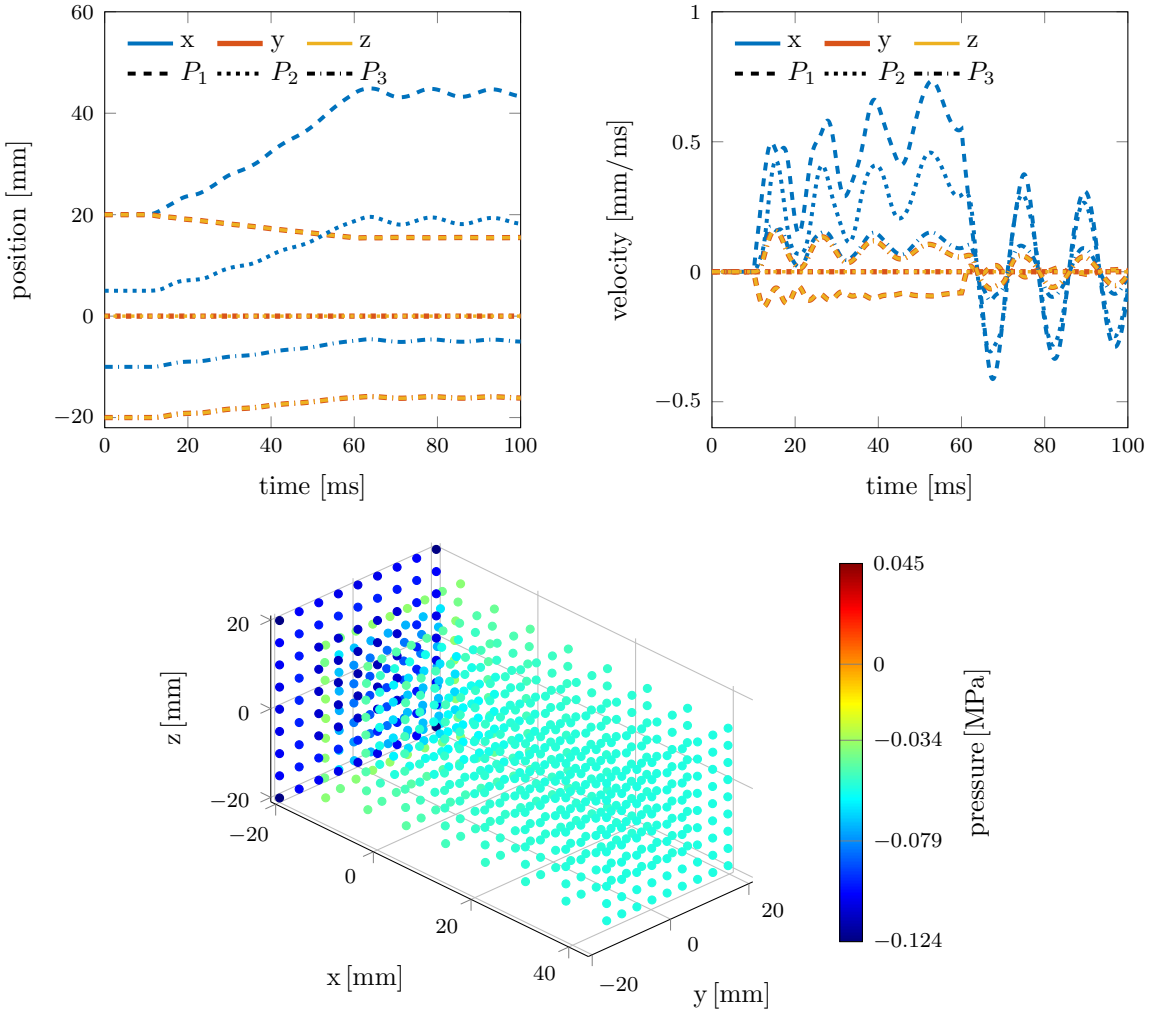


Figure 6.13: *Example 2, case BC1, final applied force* 0.1 *MPa:*
Position (top left) and velocity (top right) of the three points P_1, P_2 and P_3 in x-, y- and z-direction over time and final pressure at nodal positions (bottom).

For the applied force boundary conditions, the results (c.f. Figures 6.13, 6.14, 6.15 position

plots) show a maximum stretch λ of around 1.6, 1.3 and 1.7 for BC1, BC2 and BC3, respectively. As the material law was fitted to experimental data that ranges up to a stretch of 1.3, one could of course and should question the validity of those results in terms of how close they represent the behaviour of skeletal muscle tissue in reality. Nevertheless, in the context of this work, these simulations are reasonable, since the mathematical properties and the numerical behaviour of the model in general are of interest. As already observed at the beginning of this section, a NEO-HOOKE material law is not capable of capturing the realistic skeletal muscle behaviour anyway. Though, the experiments performed by [46] showed a failure of the pig skeletal muscle tissue at a stretch of 1.65 in fibre direction, which then again additionally supports the adequacy of the simulations.

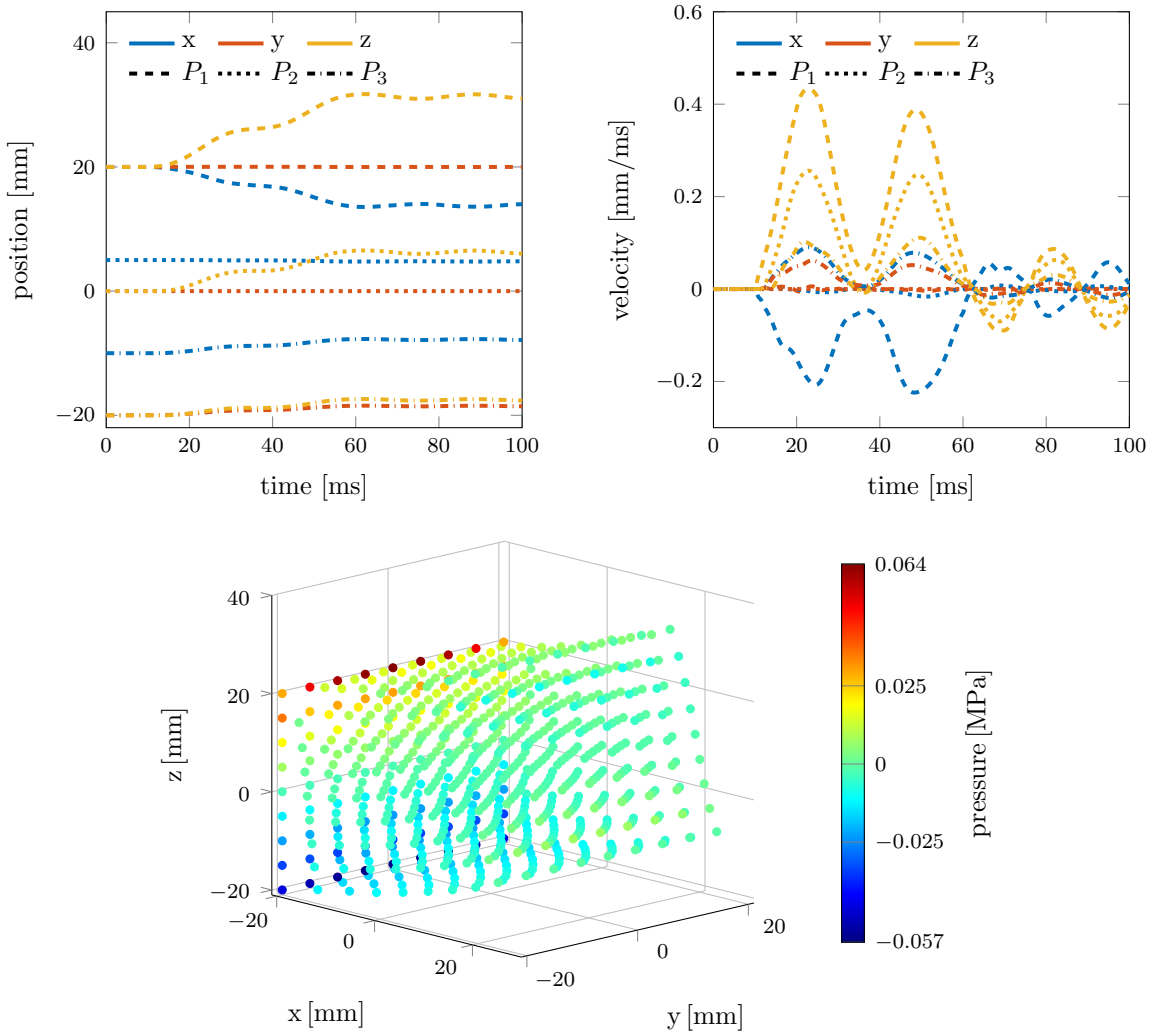


Figure 6.14: *Example 2, case BC2, final applied force* 0.01 *MPa:*
Position (top left) and velocity (top right) of the three points P_1, P_2 and P_3 in x-, y- and z-direction over time and final pressure at nodal positions (bottom).

The maximum velocities in x-, y- and z- direction for each type of applied boundary condition for this example are summarised in Table 6.1

	$\max v_x$	$\max v_y$	$\max v_z$	$\max (v_x + v_y + v_z)$
BC1	0.8	0.2	0.2	1.2
BC2	0.3	0.1	0.5	0.9
BC3	0.8	0.3	1.8	2.9

Table 6.1: *Approximate maximum velocities [mm/ms] for Example 2 in x-, y- and z-direction for each of the three boundary conditions.*

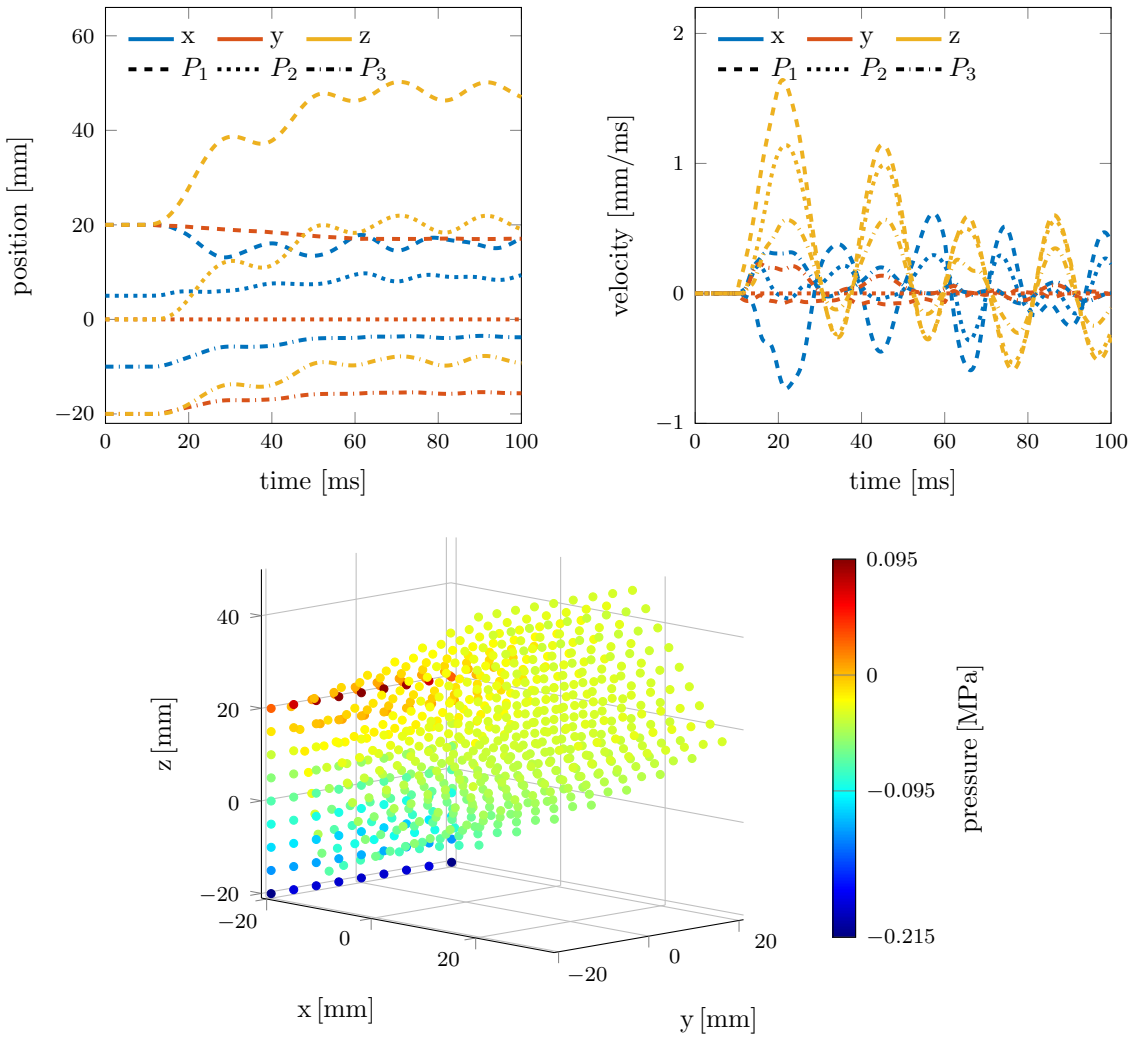


Figure 6.15: *Example 2, case BC3, final applied force* 0.05 *MPa:*
Position (top left) and velocity (top right) of the three points P_1, P_2 and P_3 in x-, y- and z-direction over time and final pressure at nodal positions (bottom).

6.1.3 Example with realistic skeletal muscle material behaviour

The main purpose of this third example is to show, how the most appropriate methods, which were chosen based on the investigations with Example 2, are influenced when the complexity of the underlying material (higher nonlinearity, adding anisotropy and activity) and the geometry are increased.

The material law as it was derived in Section 3.1 (c.f. Equation (3.2)) with the isotropic DEMIRAY and the anisotropic HOLZAPFEL contribution is utilised here. Again, the material parameters (4 altogether) have to be fitted to experimental data. For that purpose, a two step procedure is carried out.
In a first step, the isotropic DEMIRAY contribution is fitted to the experimental data from Takaza et al. [46]. To this end, the expression for the T_{11}^{D}-component needs to be derived for the uniaxial extension case. Insertion of $\boldsymbol{F}$, $\boldsymbol{F}^{-T}$ and I_1 as in Equations (6.6) into the expression derived for the 1st PIOLA-KIRCHHOFF extra stress tensor (c.f. (2.46)) yields

$$\begin{aligned}\boldsymbol{P}_{\mathrm{E}}^{\mathrm{D}}(\boldsymbol{F}) = \; & c_1 \exp\left(\frac{1}{2}c_2(\lambda^2+\frac{2}{\lambda}-3)\right)\begin{pmatrix}\lambda-\frac{1}{3}\left(\lambda^2+\frac{2}{\lambda}\right)\frac{1}{\lambda} & 0 & 0\\ 0 & \frac{1}{\sqrt{\lambda}}-\frac{1}{3}\left(\lambda^2+\frac{2}{\lambda}\right)\sqrt{\lambda} & 0\\ 0 & 0 & P_{22}\end{pmatrix}\\ & -\begin{pmatrix}\frac{1}{\lambda}p & 0 & 0\\ 0 & \sqrt{\lambda}p & 0\\ 0 & 0 & \sqrt{\lambda}p\end{pmatrix}. \end{aligned} \tag{6.20}$$

With the knowledge of $P_{22} = P_{33} = 0$, the expression for the pressure can be derived as

$$\begin{aligned} p &= c_1 \exp\left(\frac{1}{2}c_2(\lambda^2+\frac{2}{\lambda}-3)\right)\frac{1}{\lambda} - \frac{1}{3}c_1 \exp\left(\frac{1}{2}c_2(\lambda^2+\frac{2}{\lambda}-3)\right)\left(\lambda^2+\frac{2}{\lambda}\right)\\ &= c_1 \exp\left(\frac{1}{2}c_2(\lambda^2+\frac{2}{\lambda}-3)\right)\left(\frac{1}{\lambda}-\frac{1}{3}\lambda^2-\frac{2}{3}\frac{1}{\lambda}\right)\\ &= \frac{1}{3}c_1 \exp\left(\frac{1}{2}c_2(\lambda^2+\frac{2}{\lambda}-3)\right)\left(\frac{1}{\lambda}-\lambda^2\right), \end{aligned} \tag{6.21}$$

and inserted into Equation (6.20), yielding

$$P_{11}^{\mathrm{D}}(\lambda) = c_1 \exp\left(\frac{1}{2}c_2(\lambda^2+\frac{2}{\lambda}-3)\right)\left(\lambda-\frac{1}{\lambda^2}\right). \tag{6.22}$$

Thus, the expression for the true stress in x-direction during an uniaxial extension experiment is given for the DEMIRAY strain energy as

$$T_{11}^{\mathrm{D}}(\lambda) = \lambda P_{11}^{\mathrm{D}}(\lambda) = c_1 \exp\left(\frac{1}{2}c_2(\lambda^2+\frac{2}{\lambda}-3)\right)\left(\lambda^2-\frac{1}{\lambda}\right). \tag{6.23}$$

Fitting this formula to the experimental data using the MATLAB curve fitting toolbox yields $c_1 = 33.49\,\mathrm{e}{-3}\,\mathrm{MPa}$ and $c_2 = 10.45\,[\text{-}]$. The result is plotted in blue in Figure 6.16.
Now, in a second step, the material parameters for the HOLZAPFEL strain energy need to be determined. To this end, it is assumed that the muscle material response in fibre direction is 20% stiffer than in cross-fibre direction (c.f. e.g. [33]). Note that this assumption is a controversially discussed topic and there exist different experimental results on the muscle tissue stiffness in fibre and in cross-fibre direction. For this work and this example in particular, however, the intention is to show the effect of adding complexity to the material model on the developed methods. Therefore, for this proof of concept, it is not significant, whether the material is assumed to be stiffer in fibre or in cross-fibre direction. Again, the T_{11}^{HA}-component needs to be derived for the uniaxial extension case.

For muscle fibres in x-direction and a uniaxial extension in fibre direction, one has

$$\boldsymbol{\mathcal{M}} = \boldsymbol{a}_0 \otimes \boldsymbol{a}_0 = \begin{pmatrix}1\\0\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\\0\end{pmatrix} = \begin{pmatrix}1&0&0\\0&0&0\\0&0&0\end{pmatrix}, \quad \boldsymbol{F}\boldsymbol{\mathcal{M}} \overset{(6.6)_1}{=} \begin{pmatrix}\lambda&0&0\\0&0&0\\0&0&0\end{pmatrix}, \tag{6.24}$$

$$I_4 = \boldsymbol{F}\boldsymbol{a}_0 \cdot \boldsymbol{F}\boldsymbol{a}_0 = \begin{pmatrix}\lambda\\0\\0\end{pmatrix} \cdot \begin{pmatrix}\lambda\\0\\0\end{pmatrix} = \lambda^2 . \tag{6.25}$$

As can be seen from $\boldsymbol{F}\boldsymbol{\mathcal{M}}$, the anisotropic contribution only adds up to the P_{11}-component and does not affect P_{22} and P_{33}, thus leaving the pressure expression as in the pure isotropic Demiray case (c.f. Equation (6.21)). Inserting Equations (6.24) and (6.25) into the expression derived for the 1st Piola-Kirchhoff extra stress tensor (c.f. Equation (2.51)) and multiplying with λ, yields the expression for the true stress in x-/fibre-direction during a uniaxial extension experiment in x-/fibre-direction for the Holzapfel strain energy,

$$T_{11}^{\mathrm{HA}}(\lambda) = \lambda P_{11}^{\mathrm{HA}}(\lambda) = 2a_1(\lambda^2 - 1)\exp\left(a_2(\lambda^2-1)^2\right)\lambda^2 . \tag{6.26}$$

With this at hand, an additively composed true stress in x-direction

$$T_{11}^{\mathrm{D+HA}}(\lambda, a_1, a_2) := T_{11}^{\mathrm{D}}[c_1 = 33.49\,\mathrm{e}-3, c_2 = 10.45](\lambda) + T_{11}^{\mathrm{HA}}(\lambda, a_1, a_2) , \tag{6.27}$$

depending on λ, a_1, a_2 but with fixed parameters c_1, c_2 is defined. Subsequently, Equation (6.27) is fitted using the Matlab curve fitting toolbox to the experimental data from [46] multiplied by the factor 1.2. This yields $a_1 = 4.345\,\mathrm{e}{-3}\,\mathrm{MPa}$ and $a_2 = 1.434\,[\text{-}]$. The corresponding curves are plotted in red in Figure 6.16.

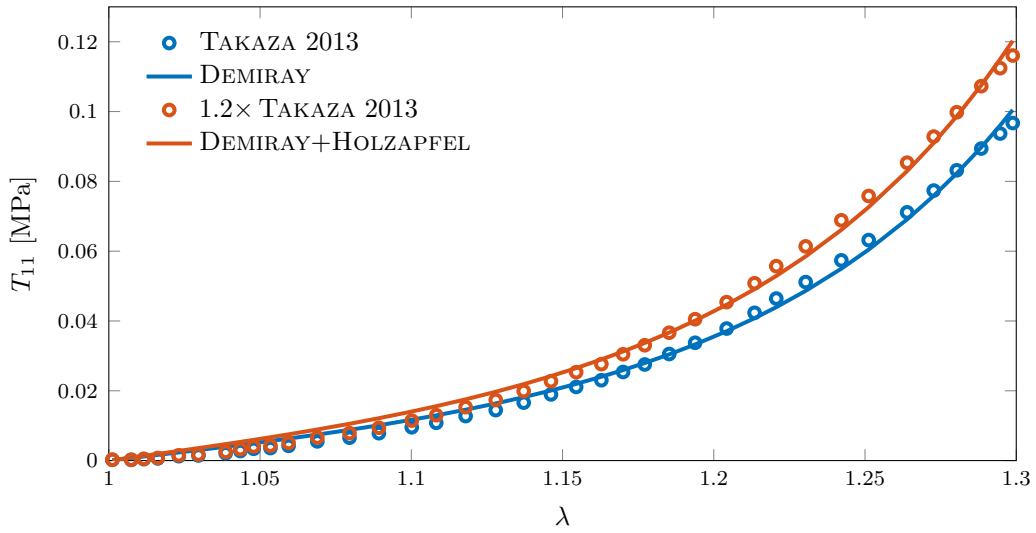


Figure 6.16: *The fit of the passive material model to the muscle data from Takaza et al. [46].*

The last parameter to be determined, is the maximum force $p_{\max}$, needed in the (considerably simplified) active stress contribution (3.3). According to [8] this is 0.2 MPa. Table 6.2 summarises the chosen material parameters for Examples 2 and 3.

parameter	value	unit
muscle density ρ_0	1.1 e−3	g/mm^3
NEO-HOOKE c_{10}	38.18 e−3	MPa
DEMIRAY c_1	33.49 e−3	MPa
DEMIRAY c_2	10.45	-
HOLZAPFEL a_1	4.345 e−3	MPa
HOLZAPFEL a_2	1.434	-
$p_{\max}$	0.2	MPa

Table 6.2: *The material parameters for Examples 2 and 3.*

6.1.3.1 Example 3A: Cubic muscle geometry

This example shall demonstrate the difference when increasing the complexity of the material model. Therefore, the same geometrical setup and zero-DIRICHLET boundary conditions as for Examples 2 were chosen (c.f. Section 6.1.2, Figure 6.10). Furthermore, the entire simulation time was 100 ms and the time step size dt = 0.1 ms.
As a first test scenario, referred to as BCxz, NEUMANN force boundary conditions were applied in a 45 °-angle to the positive x-face, again increased linearly in the interval $[10, 60]$ ms to a chosen maximum value and subsequently kept constant (c.f. Example 2, BC3). Thus, anisotropy and a stronger nonlinearity are added to the material law in comparison to Example 2, BC3.
In a second test scenario, referred to as BCxact, NEUMANN force boundary conditions were applied to the positive x-face at an angle of 0 °. Again, the force was increased linearly in the interval $[5, 55]$ ms up to a maximum value of 75 e−3 MPa and subsequently kept constant (c.f. Example 2, BC1). Additionally, the muscle was subsequently activated for 20 ms in the interval $[60, 80]$ ms up to a chosen maximum activation level $\alpha_{\max} \in (0, 1]$. That is, anisotropy, a stronger nonlinearity and activation are added to the material law in comparison to Example 2, BC1. Figure 6.17 shows the simulation protocol and the final deformed configuration.

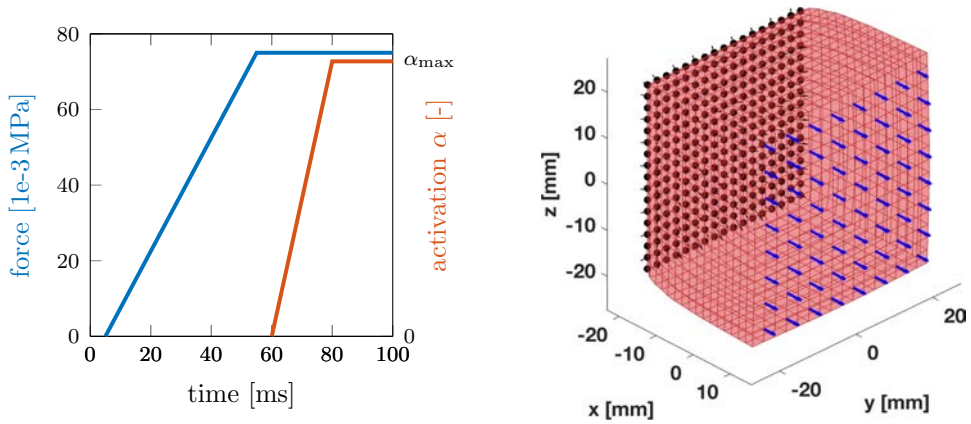


Figure 6.17: *Example 3A BCxact. Left: The simulation protocol. Right: The final, deformed configuration for the case of a complete activation, i.e.* $\alpha_{\max} := 1$.

Like for Examples 2, the position and the velocity in x-, y- and z-direction are monitored

over time and plotted in a two-dimensional plot for the points $P_1 := (20, 20, 20)$ (dashed lines), $P_2 := (5, 0, 0)$ (dotted lines) and $P_3 := (-10, -20, -20)$ (dash-dot lines) (c.f. Figure 6.10). For BCxz, only the final pressure distribution over the domain is displayed at each node, while for BCxact, additionally the pressure at time $t = 57\,\text{ms}$, i.e. before the activation starts but after the maximum force level was reached, is displayed. Below, Figures 6.18 and 6.19 show the plots for both types of boundary conditions for the case of a spatial discretisation $\mathrm{d}x = 5$.

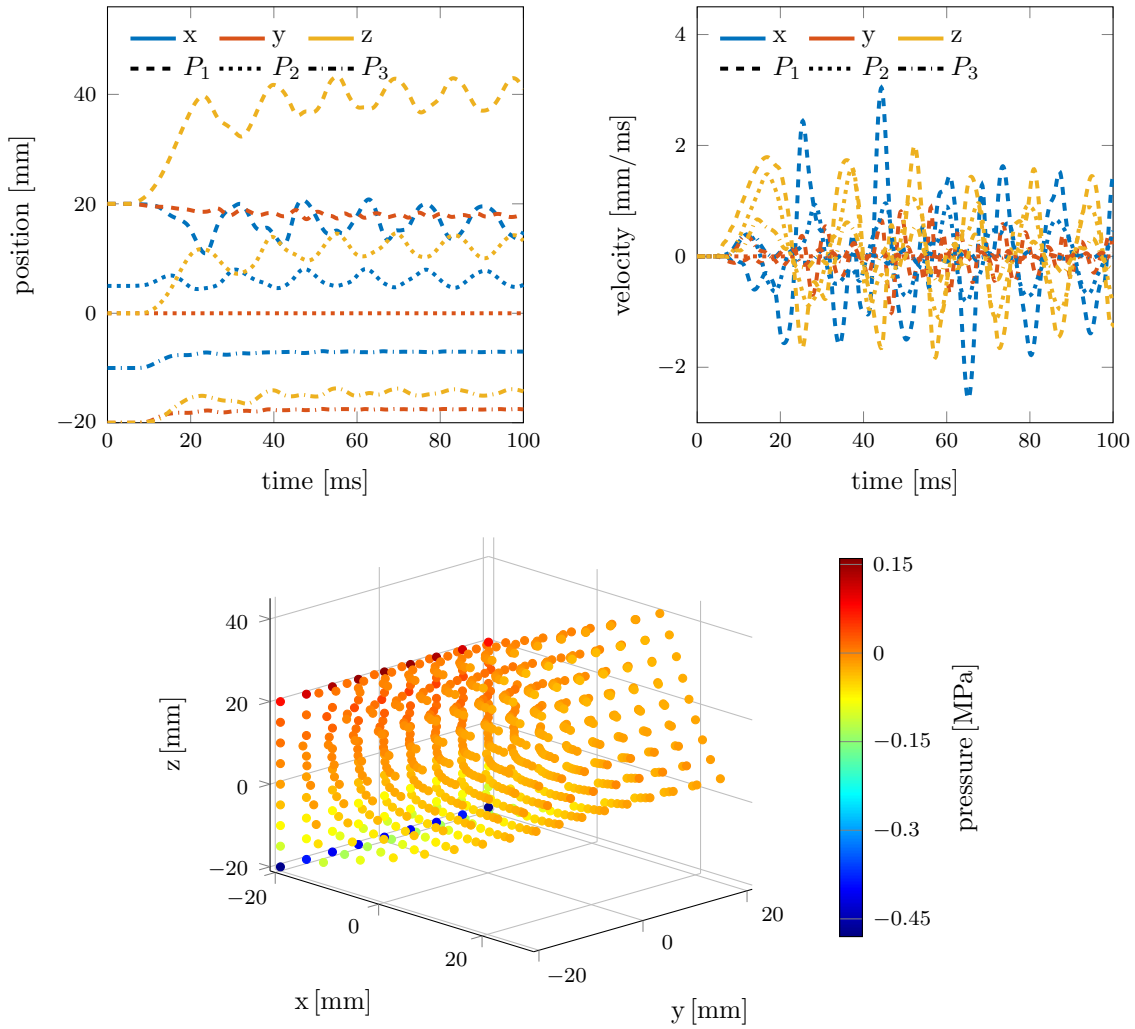


Figure 6.18: *Example 3A, case BCxz with $\mu_3 = 0.045$ [MPa]: Position (top left) and velocity (top right) of the three points P_1, P_2 and P_3 in x-, y- and z-direction over time and final pressure at nodal positions (bottom).*

For case BCxz and an applied force of 0.045 MPa, the maximum stretch is around 1.55 (c.f. Figure 6.18 top left). This is very similar to the stretch obtained in Example 2, BC3 (stretch of 1.7 when applying a force of 0.05 MPa). Further, one can observe that the oscillations are a lot more pronounced in this example than in Example 2, BC3. This becomes also obvious in the maximum velocity, especially in x-direction, which is four times higher for this material. The maximum velocities for Examples 3 are summarised in Table 6.3.

	$\max v_x$	$\max v_y$	$\max v_z$	$\max (v_x + v_y + v_z)$
Ex3A, BCxz	3.5	0.5	2.0	6.0
Ex3A, BCxact	2.0	0.5	0.5	3.0
Ex3B	0.1	0.7	0.2	1.0

Table 6.3: *Approximate maximum absolute velocities [mm/ms] for Examples 3 in x-, y- and z-direction for each of the three test cases.*

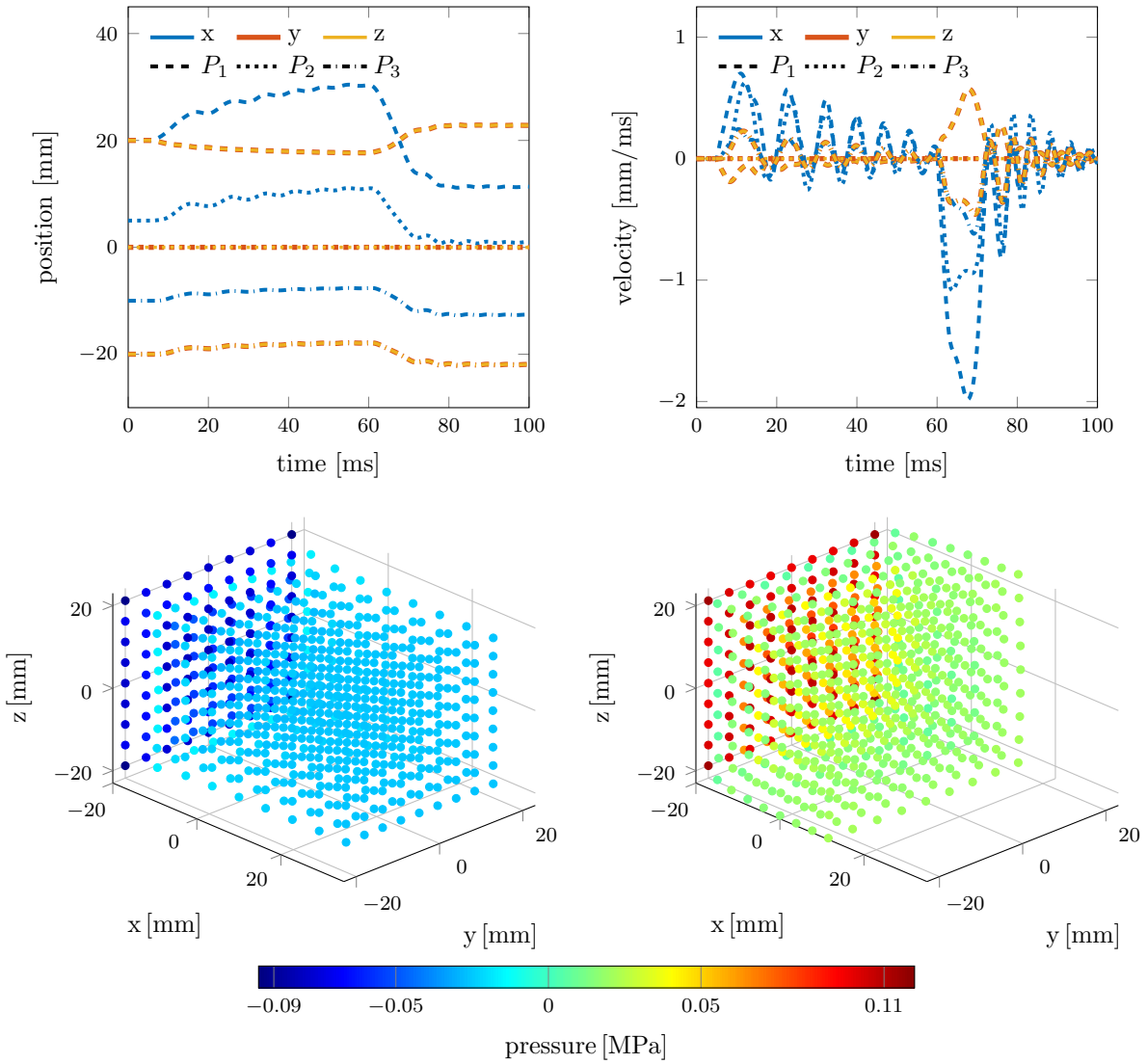


Figure 6.19: *Example 3A BCxact with $\alpha_{\max} := 1$:*
Position (top left) and velocity (top right) of the three points P_1, P_2 and P_3 in x-, y- and z-direction over time and pressure at nodal positions at time $t = 57$ ms (bottom left) and at the end (bottom right).

For Example 3A, BCxact, the applied traction force of 0.075 MPa yields a maximum stretch of around 1.25 (c.f. Figure 6.19 top left). Compared with the maximum stretch obtained in Example 2, BC1 (stretch of 1.6 for applied force of 0.1 MPa), this shows a

material response that is stiffer in fibre direction. This observation is as expected, since the anisotropic term was added to the passive material, which was assumed to be 1.2 times stiffer (fitted to a 1.2 times stiffer stress-strain curve). Furthermore, this means that this test case is in the range of the experiments performed by Takaza et al. [46]. While only applying the traction force, the extension of the material leads to negative pressure values. Then, when adding activation, the muscle contracts and the pressure becomes positive (c.f. Figure 6.19). The maximum absolute velocity of 2.0 mm/ms in x-direction is reached for this example when the active material response starts.

6.1.3.2 Example 3B: Idealised fusiform muscle geometry

This last example shall show the difference in the performance of the methods, when not only increasing the complexity of the material model but additionally choosing a more complex geometry resulting in a non-uniform mesh. For that purpose, an idealised fusiform muscle geometry was chosen, where both ends are fixed with zero-DIRICHLET boundary conditions (c.f. Figure 6.20 left). The test scenario covered a total simulation time of 50 ms. The time step size was chosen as in the previous examples ($\mathrm{d}t = 0.1$ ms). During the interval $[10, 40]$ ms, the muscle activation was linearly increased up to a chosen maximum activation level $\alpha_{\max} \in (0, 1]$. This caused a muscle contraction and resulted in a more pronounced convexity in the middle of the muscle, while the ends became thinner (c.f. Figure 6.20 right). Naturally, this effect is stronger, the higher the final maximum activation level.

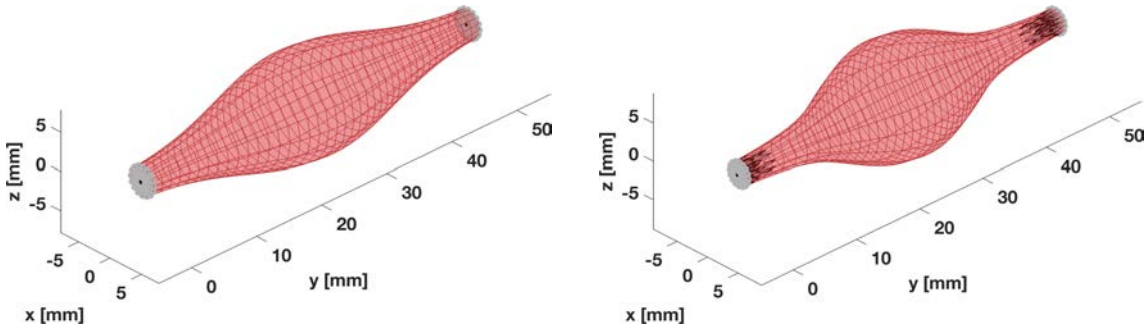


Figure 6.20: *The setup for Example 3B with the idealised fusiform muscle geometry. The length of the muscle is* 50 *mm, and the diameter in the referential state ranges between* 13 *mm at the muscle's thickest area and approximately* 3.6 *mm at the end points. Here, exemplarily shown with a spatial discretisation* $\mathrm{d}x \approx 2.5$ *mm resulting from choosing* $(1 + 2) \times 4 \times 20 = 240$ *elements. On the y-faces, the zero-*DIRICHLET *boundary conditions are visible, where the grey spheres indicate that the nodes are fixed in y-direction only, while the black sphere indicates that this node is fixed in all three directions. Left: Reference configuration. Right: Final configuration at time* $t = 50$ *ms for the case* $\alpha_{\max} = 1$. *Herein, the black arrows illustrate the reaction force.*

Here again, we want to be able to observe the dynamics of the simulation and thus track the position and the velocity of three points over time. These are roughly $P_1 = (0, 25, 6.5)$ (dashed lines), $P_2 = (1.66, 35, 1.66)$ (dotted lines), $P_3 = (1.8, 50, 0)$ (dash-dot lines). Furthermore, the final pressure at nodal positions is extracted. The location of the three

points together with the way the geometry is divided into finite elements is illustrated in Figure 6.21.

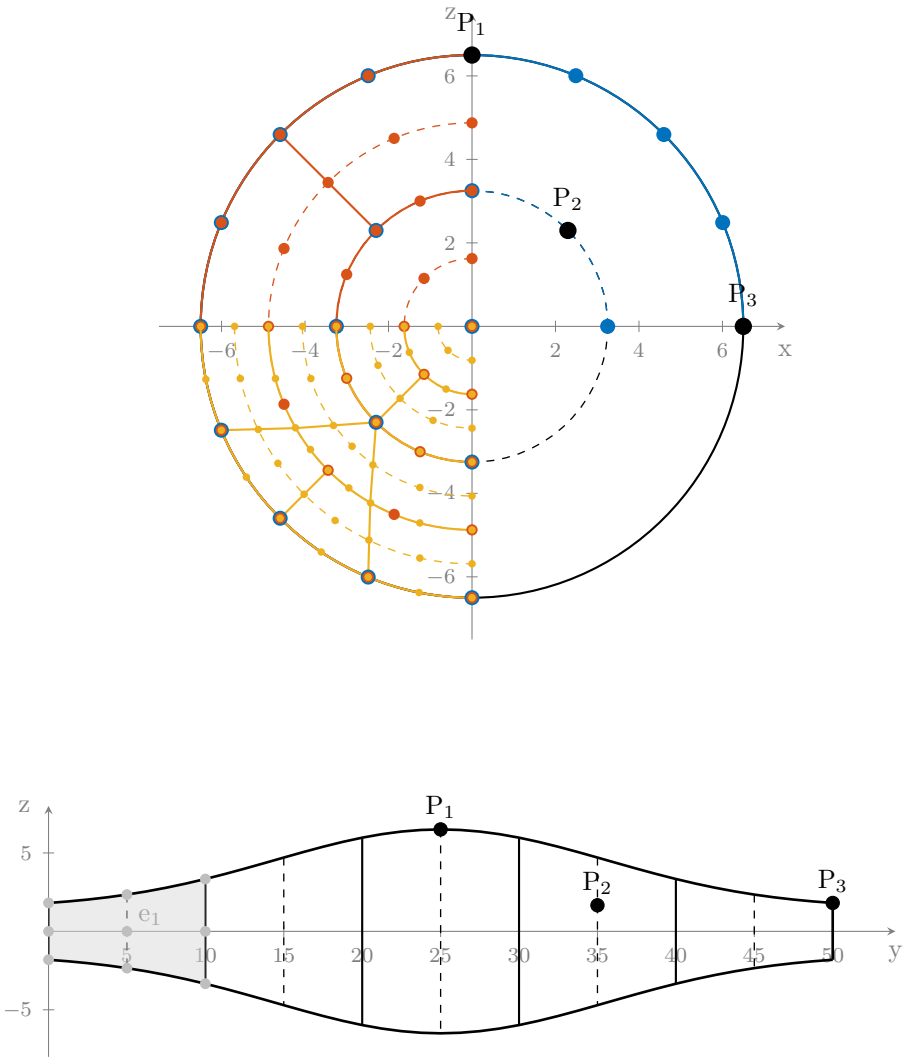


Figure 6.21: *Sketch of Example 3B: The division into finite elements and the location of the three points* $P_1 = (0, 25, 6.5)$, $P_2 = (1.66, 35, 1.66)$, $P_3 = (1.8, 50, 0)$ *chosen for evaluation purpose. For the spatial discretisation the muscle is cut into slices in its y-direction (see bottom). Depending on the chosen number of layers, the number of elements per slice varies (see top). If one layer is chosen (blue quadrant), each slice is built up of* $1 \times 4 = 4$ *elements, when choosing two layers (red quadrant), each slice is built up of* $3 \times 4 = 12$ *elements, and when choosing four layers (yellow quadrant), each slice is built up of* $10 \times 4 = 40$ *elements.*
Note that the depicted nodal positions are merely a rough location and thus not all the nodes of a coarser mesh necessarily have an equivalent in a finer discretisation.

Similar to Example 3A, BCxact, the contraction of the muscle happens relatively quickly after the activation starts. This is also the instant when the peak velocity is reached. For this scenario the peak velocity is roughly 0.7 mm/ms (c.f. Figure 6.22 right). The maximum velocity values for this test case are already included in Table 6.3 above. Figure 6.22 also shows that in the final state, the pressure at the ends, where the muscle is

clamped (e.g. representing the attachment of a tendon to a bone in reality) is negative, while inside in the middle of the muscle, the pressure is positive.

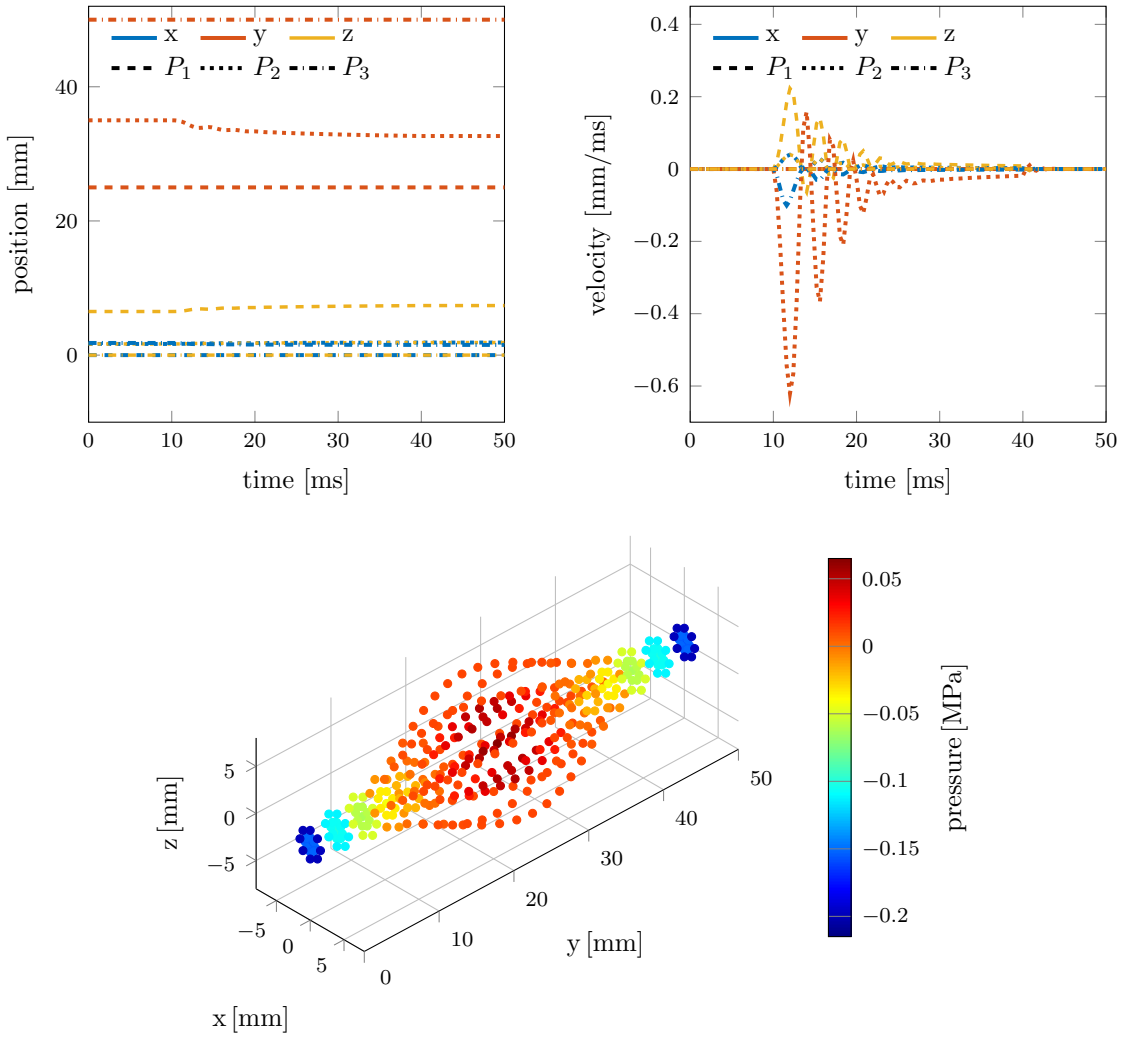


Figure 6.22: *Example 3B, results for the case* $(20 \times 12) = 240$ *elements and* $\alpha_{\max} = 1$*: Position (top left) and velocity (top right) of the three points* P_1*,* P_2 *and* P_3 *in* x*-,* y*- and* z*-direction over time and final pressure distribution (bottom).*

6.2 Observations and issues with the testing examples

For each example (Ex2, Ex3), several simulations were performed, for which not only the discretisation but also the maximum force and maximum activation (serving as parameters) were varied. First of all, those were conducted for investigations of the FOM, e.g. correct implementation, appropriate solution scheme, behaviour in stronger or weaker dynamics conditions, and a mesh convergence study. Secondly, varying the maximum force and activation, these simulations can already serve as a collection of training data (offline phase) for the final goal of this work, i.e. for obtaining a ROM. During these simulations of the FOM, different issues or difficulties already occurred.
As major factors that influence the stability of the FOM, (i) the spatial discretisation dx,

(ii) the temporal discretisation $\mathrm{d}t$, (iii) the (maximum) number of Jacobian updates in the NEWTON iteration, and (iv) the parameter range were identified.

			# converged simulations, $\mathrm{d}x =$				
	parameter range	# sim	40	20	10	5	2.5
Ex2 BC1	$\mu_3 \in [-10, 100]\,\mathrm{e}{-3}\,\mathrm{MPa}$	10	10	10	10	10	10
Ex2 BC2	$\mu_3 \in [-10, 20]\,\mathrm{e}{-3}\,\mathrm{MPa}$	8	8	8	8	8	**7**
Ex2 BC3	$\mu_3 \in [-10, 50]\,\mathrm{e}{-3}\,\mathrm{MPa}$	6	6	6	6	6	**4**
Ex3A BCxz	$\mu_3 \in [5, 45]\,\mathrm{e}{-3}\,\mathrm{MPa}$	5	5	5	5	**3**	–
Ex3A BCxact	$\alpha_{\max} \in [0.2, 1.0]$	5	5	5	5	5	–

Table 6.4: *The details of the conducted simulations with testing Examples 2 and 3. The entry "–" means that the simulations were not conducted.*
Remark: The details for Ex3B with the fusiform muscle geometry are not listed here, as the discretisation is different and there were no convergence issues for the chosen activations and discretisations.

Every scenario was solved by means of the implicit EULER scheme together with a NEWTON iteration as already roughly described in Section 3.3.2. As the evaluation of the analytical Jacobian (c.f. Equation (3.50)) is the most expensive calculation in the solution process, a slightly modified version of the NEWTON algorithm was applied. Therein, the Jacobian is only updated if the number of NEWTON iterations for the time step at hand exceeds 15, or if for the implicit function $\left\| \boldsymbol{f}(\bar{\boldsymbol{x}}_{i+1}^{(k)}) \right\|_{\bar{\boldsymbol{H}}} > 0.75 \cdot \left\| \boldsymbol{f}(\bar{\boldsymbol{x}}_{i+1}^{(k-1)}) \right\|_{\bar{\boldsymbol{H}}}$ holds. This is a slightly extended/modified variant of an inexact NEWTON method introduced by Shamanskii [43], referred to as SHAMANSKII-NEWTON-RAPHSON scheme. Furthermore, we performed a LU-decomposition (MATLAB function `lu()`) of the calculated Jacobian ($\boldsymbol{Jf} = \boldsymbol{LU}$) and solved for the increment (c.f. Equation (3.48)) by using MATLAB's backslash operator ($\Delta\bar{\boldsymbol{x}} = \boldsymbol{U}\backslash(\boldsymbol{L}\backslash\boldsymbol{f})$). The NEWTON tolerance was set to $1\,\mathrm{e}{-10}$ and the maximum number of NEWTON iterations per time step to 100. If nothing else is stated, we allow for a maximum of 20 Jacobian updates for a single simulation, as we observed that when more updates were needed, that simulation did not converge at all, not even when increasing this number. Certainly, there are more advanced solvers (e.g. MATLAB's solver `ode15i`) than the implicit EULER method. However, we chose to stick to the self implemented scheme described above in order to exactly know what is happening during the solution process and to have a comparable simulation time for the ROM in relation to the FOM.
Table 6.4 summarises the conducted simulations using this approach for Examples 2 and 3. As can be seen, convergence problems occurred in Example 2 for the finest spatial discretisation of $\mathrm{d}x = 2.5$. For BC2 the simulation with $\mu_3 = 20\,\mathrm{e}{-3}\,\mathrm{MPa}$ and for BC3 the simulations with $\mu_3 = 40\,\mathrm{e}{-3}\,\mathrm{MPa}$ and $\mu_3 = 50\,\mathrm{e}{-3}\,\mathrm{MPa}$ did not converge, while those cases did not cause any problems for the coarser discretisations $\mathrm{d}x \geq 5$ (factors (i) and (iv)). For Examples 3A, the maximum number of Jacobian updates (factor (iii)) was increased to 50 for discretisations $\mathrm{d}x \leq 10$ as one means to overcome convergence issues for those cases. Unfortunately, for BCxz and $\mathrm{d}x = 5$ this was still not sufficient and the simulations with $\mu_3 = 35\,\mathrm{e}{-3}\,\mathrm{MPa}$ and $\mu_3 = 45\,\mathrm{e}{-3}\,\mathrm{MPa}$ did not converge. Having the

CFL condition* in mind, we tried to decrease the time step size $\mathrm{d}t$ (factor (ii)) as an additional means. This was successful in the case of BCxz, $\mathrm{d}x = 5$. With a decreased time step size of $\mathrm{d}t = 0.025$, all simulations converged. Unfortunately neither increasing the maximum number of Jacobian updates, nor decreasing the time step size were means to achieve convergence in the failed cases of Example 2.
Tables 6.1 and 6.3 summarised the maximum velocities in x-, y- and z-direction together with $\max(v_x + v_y + v_z)$ for Examples 2 and 3. One can see that the maximum value for $(v_x + v_y + v_z)$ is 6.0 for all simulations. Calculating the COURANT number C for all chosen spatial discretisations $\mathrm{d}x = 40, 20, 10, 5, 2.5\,\mathrm{mm}$ and a temporal discretisation of $\mathrm{d}t = 0.1\,\mathrm{ms}$ leads to

$$\mathrm{C}_{40} = 0.0025 \cdot 6.0 = 0.015\,, \quad \mathrm{C}_{20} = 0.005 \cdot 6.0 = 0.03\,, \quad \mathrm{C}_{10} = 0.01 \cdot 6.0 = 0.06\,,$$
$$\mathrm{C}_{5} = 0.02 \cdot 6.0 = 0.12\,, \quad \mathrm{C}_{2.5} = 0.04 \cdot 6.0 = 0.24\,.$$

For all cases the values are (a lot) smaller than 1. This means, the CFL condition is fulfilled and indicates that the convergence problems cannot (mainly) be a result from the chosen ratio between the spatial and temporal discretisation.
Furthermore, in Examples 1, the finest spatial discretisation yields $\mathrm{d}t/\mathrm{d}x = 4$ with values for $\max(v_x+v_y+v_z)$ around 0.2, i.e. a COURANT number quite close to 1 and nevertheless those cases are converging. Thus, one possible explanation for the convergence problems might lie in the still missing viscous damping effects. While the oscillations are hardly present in Examples 1 with their small dimensions (i.e. negligible inertia effects), they are quite pronounced in some testing cases of Examples 2 and 3. Adding an appropriate viscous damping contribution in the material model (c.f. Section 3.1) might thus lead to a distinct equilibrium state and help to overcome the convergence issues.

6.3 Convergence study

To further analyse the FOM, we perform a mesh convergence study on the examples introduced in Section 6.1. The analysis is conducted for all three examples to investigate the effects of both, the transition from a quasi-static to a dynamic simulation and the increasing complexity from a simple and passive to a more complex and active material behaviour.
To keep the notation as straightforward and clear as possible, we will not distinguish (i.e. will not introduce an additional variable) between the vector of a discrete solution for a single time step and a matrix containing the discrete solution vectors for all time steps. For Examples 1, where only the final converged state is of interest for the comparison and we do not consider the velocity, this means

$$\tilde{\boldsymbol{x}} := \tilde{\boldsymbol{x}}(t_{n_t}) = \begin{pmatrix} \boldsymbol{u}(t_{n_t}) \\ \boldsymbol{w}(t_{n_t}) \end{pmatrix} \in \mathbb{R}^{\tilde{d}\times 1}\,. \tag{6.28}$$

While for Examples 2 and 3, where also the states in between the reference and final

*COURANT-FRIEDRICHS-LEWY (CFL) condition: For the three-dimensional case with an equidistant grid, the COURANT number C is calculated as $\mathrm{C} := \frac{\mathrm{d}t}{\mathrm{d}x}(v_x + v_y + v_z)$. Typically, for explicit solvers, $\mathrm{C} \leq 1$ needs to hold, while for implicit solvers also larger values for C might be tolerated (c.f. [17]).

configuration are of interest, the solution $\bar{\boldsymbol{x}}$ means

$$\bar{\boldsymbol{x}} := [\bar{\boldsymbol{x}}(t_0), \dots, \bar{\boldsymbol{x}}(t_{n_t})] = \left[\begin{pmatrix} \boldsymbol{u}(t_0) \\ \boldsymbol{v}(t_0) \\ \boldsymbol{w}(t_0) \end{pmatrix}, \dots, \begin{pmatrix} \boldsymbol{u}(t_{n_t}) \\ \boldsymbol{v}(t_{n_t}) \\ \boldsymbol{w}(t_{n_t}) \end{pmatrix} \right] \in \mathbb{R}^{\bar{d} \times (n_t+1)}, \tag{6.29}$$

whereas the solution for a single time step t_i, $i \in \{0, \dots, n_t\}$, is described using MATLAB notation for addressing the i^{th} column

$$\bar{\boldsymbol{x}}(:, t_i) := \bar{\boldsymbol{x}}(t_i) \in \mathbb{R}^{\bar{d} \times 1}. \tag{6.30}$$

Moreover, for Examples 1, the discretisation errors are expressed in the discrete $\mathcal{L}^2$-norm $\|\cdot\|_{\mathcal{L}^2}$, while Examples 2 and 3 use the discrete energy norm $\|\cdot\|_{\bar{\boldsymbol{H}}}$ based on the discrete inner product as it was derived in Section 3.3.1.

6.3.1 Quasi-static examples with an analytical solution

Here, the final converged state $\tilde{\boldsymbol{x}} = (\boldsymbol{u}, \boldsymbol{w})^T \in \mathbb{R}^{\tilde{d}}$ for four different mesh sizes is compared with the analytical solution derived in Section 6.1.1 on the nodal positions $\tilde{\boldsymbol{x}}^a := (\boldsymbol{u}^a, \boldsymbol{w}^a)^T \in \mathbb{R}^{\tilde{d}}$. The chosen spatial discretisations together with the resulting number of elements and dof for the Examples 1 are listed in Tables 6.5 and 6.6.

dx [mm]	elements	u/v-dof ($3N$)	w-dof (N_p)	total dof
0.2	5	297	24	$2 \times 297 + 24 = 618$
0.1	40	1 575	99	$2 \times 1\,575 + 99 = 3\,249$
0.05	320	9 963	525	$2 \times 9\,963 + 525 = 20\,451$
0.025	2 560	70 227	3 321	$2 \times 70\,227 + 3\,321 = 143\,775$

Table 6.5: *The chosen discretisations and resulting number of elements and dof for the convergence analysis of the quasi-static examples with an analytical solution.*

	Example 1A		Example 1B		Example 1C	
dx [mm]	u/v-dof	system dof	u/v-dof	system dof	u/v-dof	system dof
0.2	286	596	198	420	231	486
0.1	1 548	3 195	1 260	2 619	1 365	2 829
0.05	9 880	20 285	8 856	18 237	9 225	18 975
0.025	69 936	143 193	66 096	135 513	67 473	138 267

Table 6.6: *As different zero-*DIRICHLET *boundary conditions are applied, this results in a different number of total system dof for Examples 1A, 1B and 1C. However, note that the* DIRICHLET *dof are reinserted in the state vectors for the error computation.*

We computed the errors $\varepsilon^{(\star)}$ and relative errors, $\varepsilon^{(\star)}_{\text{rel}}$, $(\star) \in \{\tilde{\boldsymbol{x}}, \boldsymbol{u}, \boldsymbol{w}\}$, in the discrete

$\mathcal{L}^2$-norm, meaning e.g.

$$\varepsilon^{\tilde{\boldsymbol{x}}}_{\mathcal{L}^2} := \left\|\tilde{\boldsymbol{x}}^a - \tilde{\boldsymbol{x}}\right\|_{\mathcal{L}^2} \approx \sqrt{\mathrm{d}x^3} \cdot \left\|\tilde{\boldsymbol{x}}^a - \tilde{\boldsymbol{x}}\right\|_2 = \sqrt{\mathrm{d}x^3} \cdot \sqrt{\sum_{k=1}^{\tilde{d}} \left(\tilde{x}^a_k - \tilde{x}_k\right)^2} \tag{6.31}$$

and

$$\varepsilon^{\tilde{\boldsymbol{x}}}_{\mathrm{rel},\mathcal{L}^2} := \frac{\left\|\tilde{\boldsymbol{x}}^a - \tilde{\boldsymbol{x}}\right\|_{\mathcal{L}^2}}{\left\|\tilde{\boldsymbol{x}}^a\right\|_{\mathcal{L}^2}} = \sqrt{\sum_{i=k}^{\tilde{d}} \left(\tilde{x}^a_k - \tilde{x}_k\right)^2} \Bigg/ \sqrt{\sum_{i=k}^{\tilde{d}} \left(\tilde{x}^a_k\right)^2} \quad . \tag{6.32}$$

For each example, Figures 6.23–6.28 below contain six plots. The first row shows the error decays for the whole state $\tilde{\boldsymbol{x}}$ as well as separately for the position dof $\boldsymbol{u}$ and the pressure dof $\boldsymbol{w}$. Additionally, the second row shows the errors of the position in x-, y- and z-direction separately. The reason for this separation is the fact that the errors in the different type of dof sometimes vary in several orders magnitude and therefore, the discretisation error of the whole state $\tilde{\boldsymbol{x}}$ only captures the behaviour of the component with the largest error. Furthermore, each plot contains the linear (dashed) and quadratic (dash-dotted) slope for comparison.
The first observation that can be made is, that the discretisation errors in Example 1A are several orders of magnitude higher than the discretisation errors obtained in Examples 1B and 1C. While the absolute and relative discretisation errors in Example 1A are in the range $1\,\mathrm{e}{-4}$ to $1\,\mathrm{e}{-8}$, they are between $1\,\mathrm{e}{-10}$ to $1\,\mathrm{e}{-14}$ and $1\,\mathrm{e}{-18}$ (i.e. already machine precision) for Examples 1B and 1C, respectively. As expected, for each of the examples, the error component that dominates the overall discretisation error is the one in that direction where the largest displacements occur, e.g. the x-direction in the uniaxial extension Example 1A. Furthermore, for Examples 1A and 1C hardly any slope is visible, i.e. the error only decreases very little when refining the mesh. This is most pronounced for the relative errors in Example 1A (c.f. Figure 6.24), where the error plots are basically horizontal lines. For Example 1B (c.f. Figures 6.25 and 6.26), all error decays are in between the linear and the quadratic slope. Recalling that the three analytical solutions derived in Section 6.1.1 are linear in the position and constant in the pressure field, they lie in the Ansatz space of the chosen $\mathbb{Q}_2^{(27)} - \mathbb{Q}_1$ elements, which therefore should theoretically be capable of exactly representing the solutions with a single element already. However, we compare a time-converged dynamic solution with the quasi-static solution and therefore error propagation, due to errors arising from the time discretisation and from the Newton iteration has an additional effect. This might be one explanation for the rather marginal slopes.

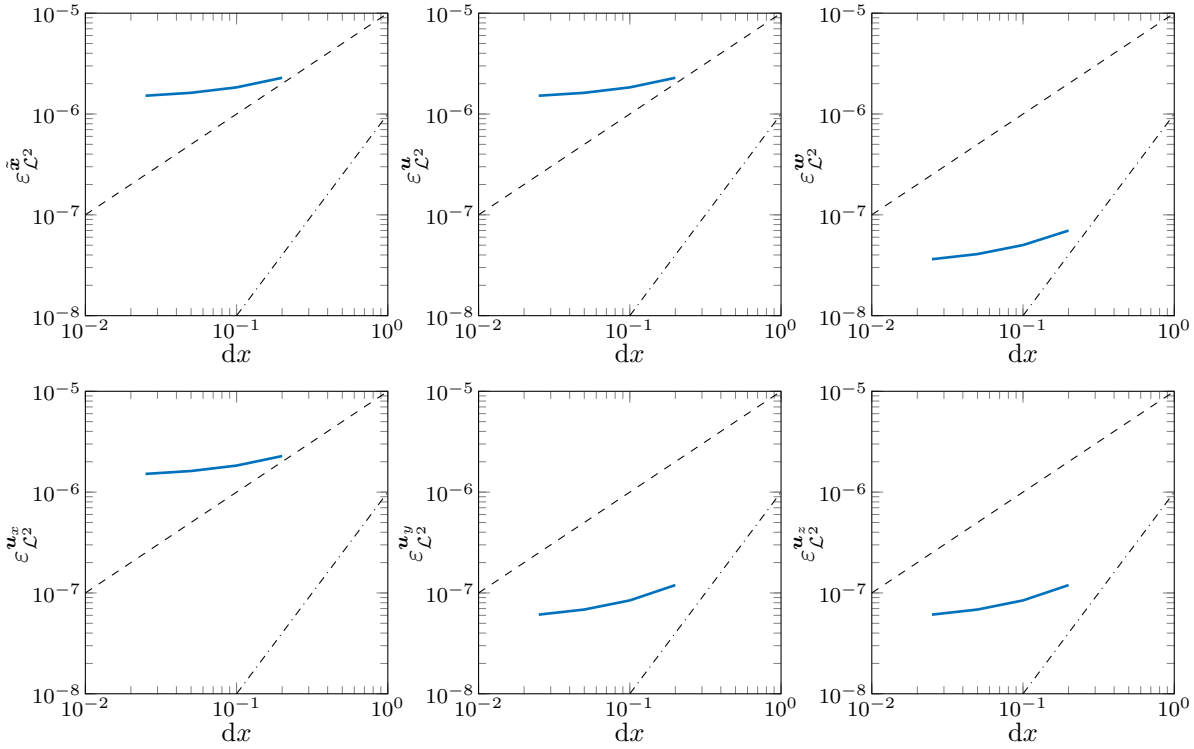


Figure 6.23: *Absolute errors in the discrete $\mathcal{L}_2$-norm for Example 1A.*

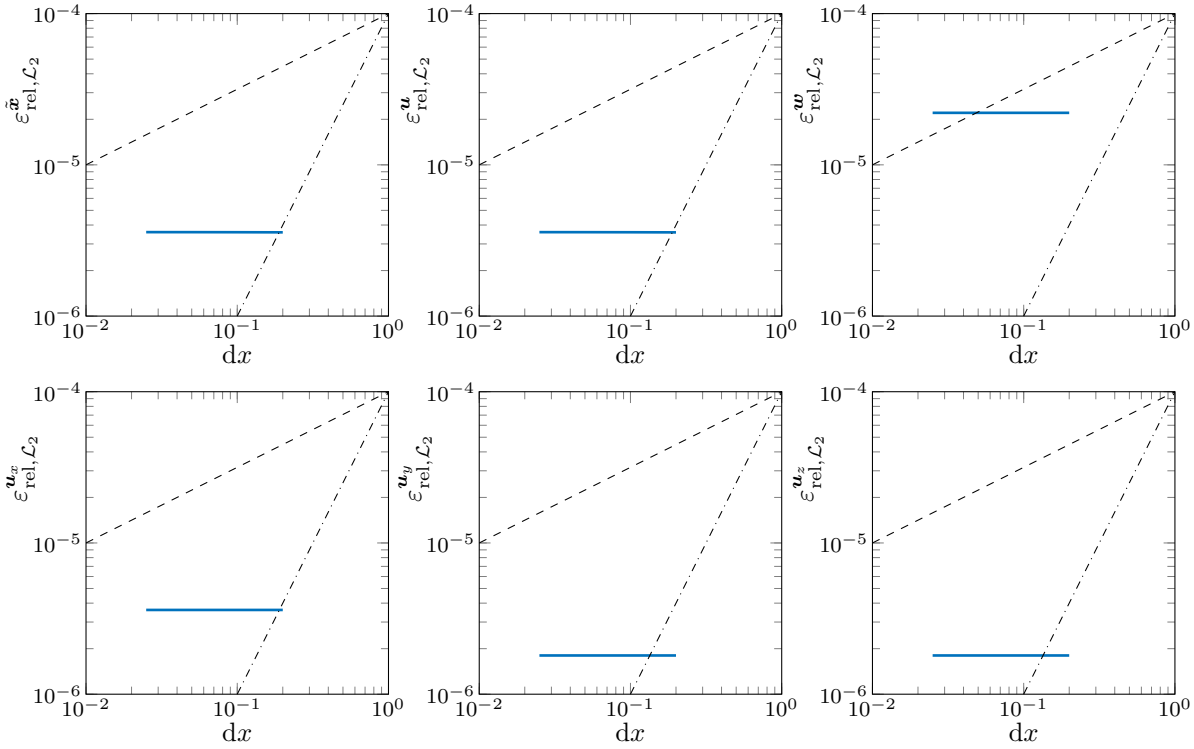


Figure 6.24: *Relative errors in the discrete $\mathcal{L}_2$-norm for Example 1A.*

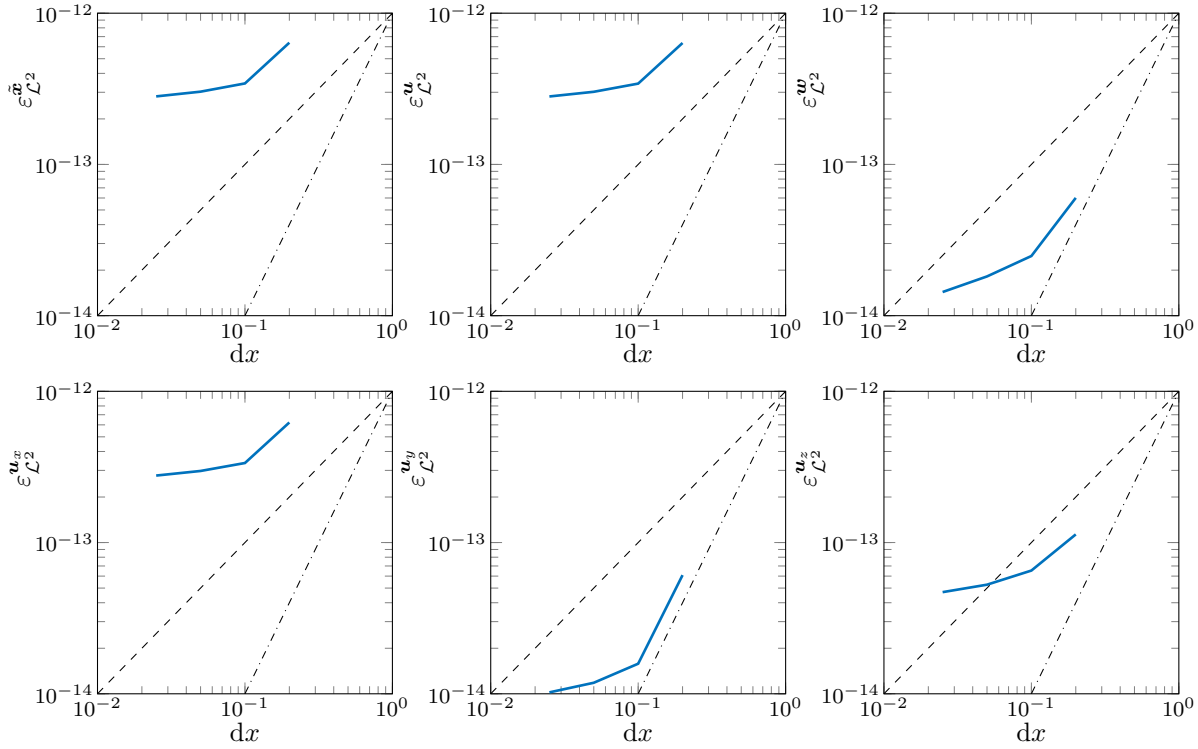


Figure 6.25: *Absolute errors in the discrete $\mathcal{L}_2$-norm for Example 1B.*

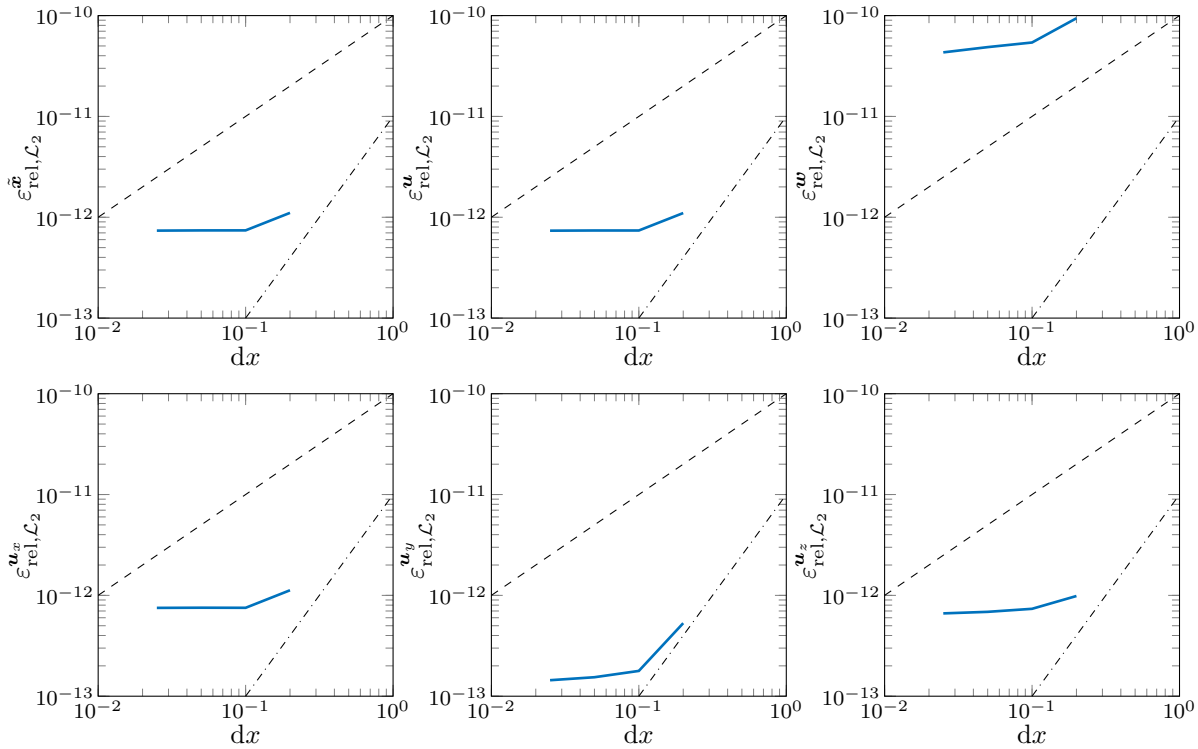


Figure 6.26: *Relative errors in the discrete $\mathcal{L}_2$-norm for Example 1B.*

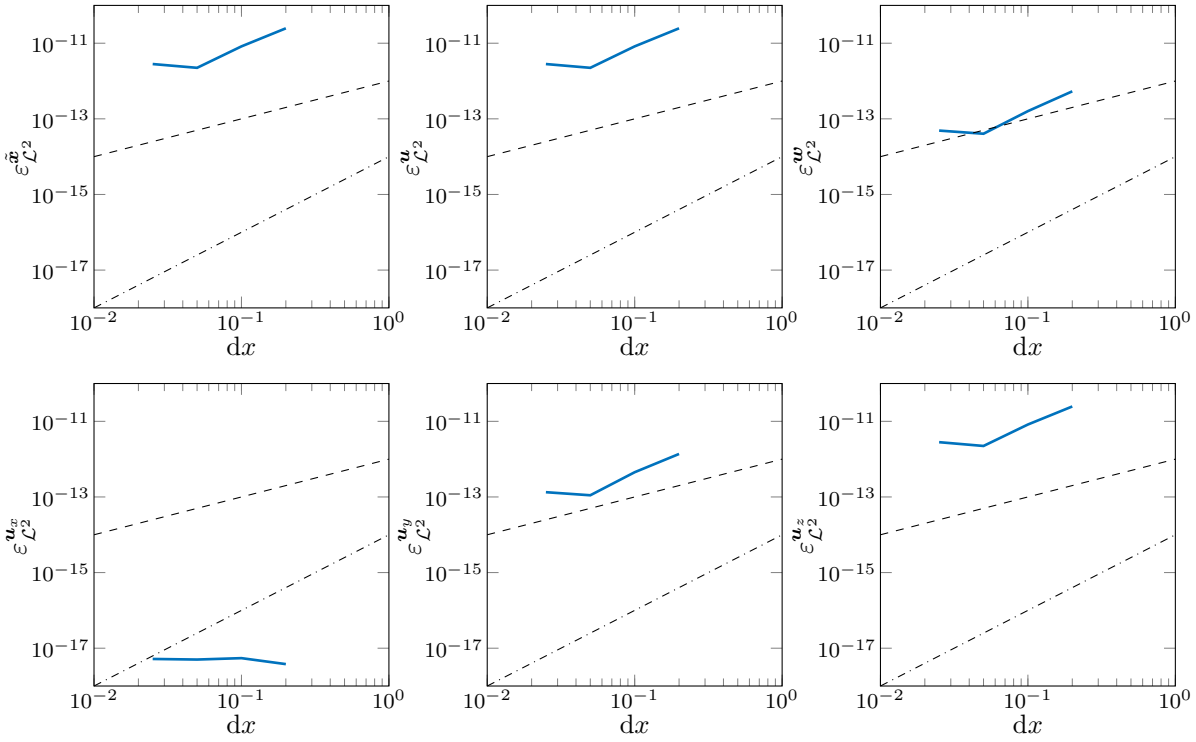


Figure 6.27: *Absolute errors in the discrete $\mathcal{L}_2$-norm for Example 1C.*

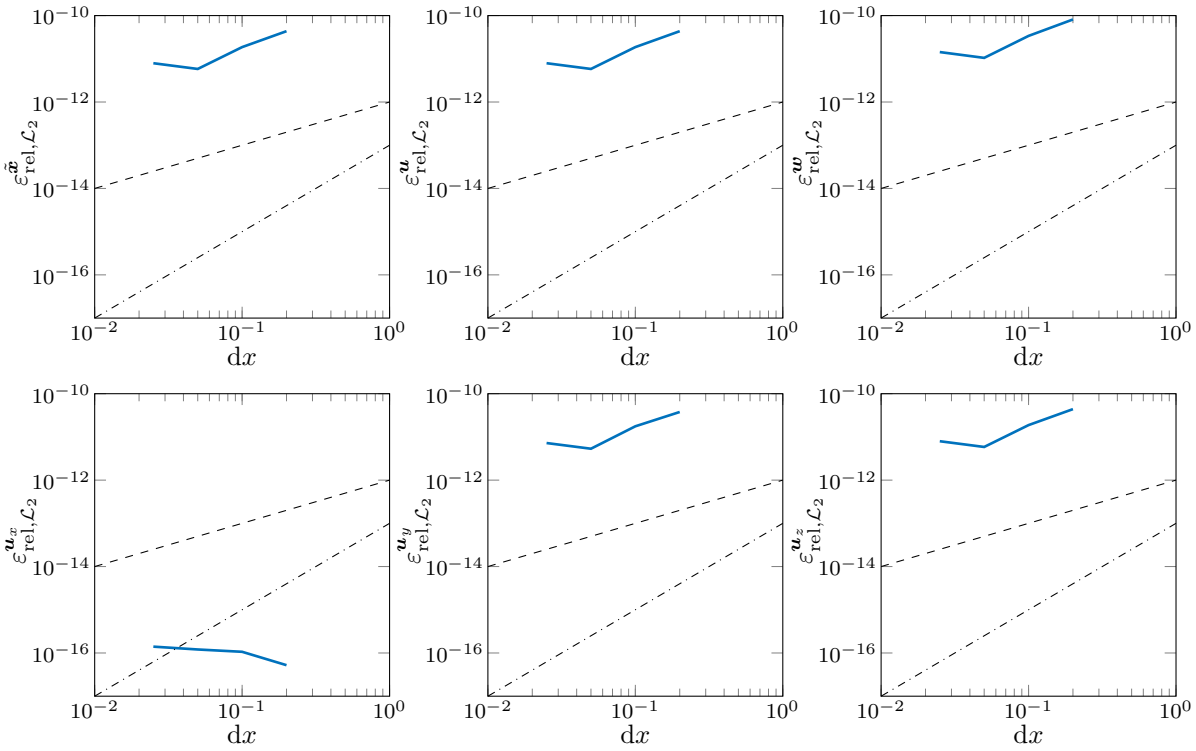


Figure 6.28: *Relative errors in the discrete $\mathcal{L}_2$-norm for Example 1C.*

6.3.2 Dynamic example with a simple material law

For Example 2 an approximate discretisation error is calculated. This is due to the lack of an analytical solution for these cases. Instead, the errors are calculated with respect to the solution computed on the finest mesh, which is considered the true solution. The chosen spatial discretisations together with the resulting number of elements and dof for the Examples 2 are listed in Table 6.7. The time step size was set to $\mathrm{d}t = 0.1\,\mathrm{ms}$ for every simulation.

$\mathrm{d}x$ [mm]	elements	u/v-dof ($3N$)	w-dof (N_p)	total dof
40	1	81	8	$2\times 81 + 8 = 170$
20	8	375	27	$2\times 375 + 27 = 777$
10	64	2 187	125	$2\times 2\,187 + 125 = 4\,499$
5	512	14 739	729	$2\times 14\,739 + 729 = 30\,207$
2.5	4 096	107 811	4 913	$2\times 107\,811 + 4\,913 = 220\,535$

Table 6.7: *The chosen discretisations and resulting number of elements and dof for the convergence analysis of Example 2.*

For $(\star) \in \{\bar{\boldsymbol{x}}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}\}$, let $(\star)^f$ be the solution on the finest mesh ($\mathrm{d}x = 2.5$ or $\mathrm{d}x = 5$) and $(\star)^c$ the solution on the coarse mesh $\mathrm{d}x = 40,\ 20,\ 10\,(,\ 5)$ respectively*. For each time step t_k, $k = 0, ..., n_t$, we compute the errors $\varepsilon^{(\star)}(t_k)$ in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$, e.g.

$$\begin{aligned}\varepsilon^{\bar{\boldsymbol{x}}}_{\boldsymbol{H}}(t_k) &:= \left\|\bar{\boldsymbol{x}}^f(:,t_k) - \bar{\boldsymbol{x}}^c(:,t_k)\right\|_{\bar{\boldsymbol{H}}} \\ &= \sqrt{\left[\bar{\boldsymbol{x}}^f(:,t_k) - \bar{\boldsymbol{x}}^c(:,t_k)\right]^T \bar{\boldsymbol{H}}\left[\bar{\boldsymbol{x}}^f(:,t_k) - \bar{\boldsymbol{x}}^c(:,t_k)\right]}\quad .\end{aligned} \tag{6.33}$$

Subsequently, the overall absolute discretisation error is computed as the mean over all time steps

$$\varepsilon^{(\star)}_{\boldsymbol{H}} := \frac{1}{n_t+1}\sum_{k=0}^{n_t} \varepsilon^{(\star)}_{\boldsymbol{H}}(t_k)\,. \tag{6.34}$$

The computation of the relative errors, $\varepsilon^{(\star)}_{\mathrm{rel}}(t_k)$ for each time step could not be done as straight forward as in the Examples 1 due to zero velocity and/or zero pressure coefficients for some time steps (especially in the beginning of the simulation). Therefore, we choose

$$\varepsilon^{\bar{\boldsymbol{x}}}_{\mathrm{rel},\boldsymbol{H}}(t_k) := \frac{\left\|\bar{\boldsymbol{x}}^f(:,t_k) - \bar{\boldsymbol{x}}^c(:,t_k)\right\|_{\bar{\boldsymbol{H}}}}{\|\bar{\boldsymbol{x}}^f(:,t_k)\|_{\bar{\boldsymbol{H}}}}\,,\quad \varepsilon^{\boldsymbol{u}}_{\mathrm{rel},\boldsymbol{H}}(t_k) := \frac{\left\|\boldsymbol{u}^f(:,t_k) - \boldsymbol{u}^c(:,t_k)\right\|_{\boldsymbol{H_u}}}{\|\boldsymbol{u}^f(:,t_k)\|_{\boldsymbol{H_u}}}\,,$$

*Note the minor imprecision in the notation here for $(\star)^f$, which doesn't make a difference between the solution on the finest grid with all its dof and with the ones that correlate with the nodal positions for solutions on the coarser meshes, used for the calculation of the absolute differences.

i.e. as usual for the complete state dof vector $\bar{\boldsymbol{x}}$ and position dof vector $\boldsymbol{u}$, and deviant

$$\varepsilon^{\boldsymbol{v}}_{\mathrm{rel},\boldsymbol{H}}(t_k) := \begin{cases} \dfrac{\left\|\boldsymbol{v}^f(:,t_k)-\boldsymbol{v}^c(:,t_k)\right\|_{\boldsymbol{H_u}}}{\max\left(\left\|\boldsymbol{v}^f(:,t_k)\right\|_{\boldsymbol{H_u}},\|\boldsymbol{v}^c(:,t_k)\|_{\boldsymbol{H_u}}\right)}\,, & \text{if } \max\left(\left\|\boldsymbol{v}^f(:,t_k)\right\|_{\boldsymbol{H_u}},\|\boldsymbol{v}^c(:,t_k)\|_{\boldsymbol{H_u}}\right) \neq 0 \\ 0\,, & \text{if } \max\left(\left\|\boldsymbol{v}^f(:,t_k)\right\|_{\boldsymbol{H_u}},\|\boldsymbol{v}^c(:,t_k)\|_{\boldsymbol{H_u}}\right) = 0 \end{cases}$$

for the velocity dof $\boldsymbol{v}$ and

$$\varepsilon^{\boldsymbol{w}}_{\mathrm{rel},\boldsymbol{H}}(t_k) := \begin{cases} \dfrac{\left\|\boldsymbol{w}^f(:,t_k)-\boldsymbol{w}^c(:,t_k)\right\|_{\boldsymbol{H_p}}}{\max\left(\left\|\boldsymbol{w}^f(:,t_k)\right\|_{\boldsymbol{H_p}},\|\boldsymbol{w}^c(:,t_k)\|_{\boldsymbol{H_p}}\right)}\,, & \text{if } \max\left(\left\|\boldsymbol{w}^f(:,t_k)\right\|_{\boldsymbol{H_p}},\|\boldsymbol{w}^c(:,t_k)\|_{\boldsymbol{H_p}}\right) \neq 0 \\ 0\,, & \text{if } \max\left(\left\|\boldsymbol{w}^f(:,t_k)\right\|_{\boldsymbol{H_p}},\|\boldsymbol{w}^c(:,t_k)\|_{\boldsymbol{H_p}}\right) = 0 \end{cases}$$

for the pressure dof $\boldsymbol{w}$. The overall relative errors $\varepsilon^{(\star)}_{\mathrm{rel},\boldsymbol{H}}$ are computed in a similar manner as the absolute errors (c.f. (6.34)). However with the difference that only the non-zero entries are considered, i.e.

$$\varepsilon^{(\star)}_{\mathrm{rel},\boldsymbol{H}} := \frac{1}{\mathrm{nnz}} \sum_{k,nz} \varepsilon^{(\star)}_{\mathrm{rel},\boldsymbol{H}}(t_k)\,. \tag{6.35}$$

Again, the discretisation errors were calculated separately for the different dof types and the different spatial directions. Therefore, for each boundary condition type of Example 2, Figures 6.29–6.34 contain ten axes. The left column of each figure contains the errors for the whole state $\bar{\boldsymbol{x}}$ and the errors in the pressure dof $\boldsymbol{w}$. The middle column contains the errors in the overall position coefficient vector $\boldsymbol{u}$ in the top row, and additionally, in the rows below, the position errors separated in x-, y- and z-direction. The same is done in the right column, merely for the velocity coefficient vector $\boldsymbol{v}$ in this case. Furthermore, each axis contains several plots, since the simulations were performed for the different parameters listed in Table 6.4.
For those Examples 2 now, which were deliberately chosen with a sufficient complexity, the results show the expected behaviour of a quadratic asymptotic convergence rate in the $\boldsymbol{u}$ and $\boldsymbol{v}$ dof and a linear asymptotic convergence rate in the pressure dof $\boldsymbol{w}$. This is the case for the absolute as well as the relative errors. Note that the discretisation errors for an applied compression and traction force with the same absolute value are very similar and thus the two curves cannot be distinguished and it might look as if a plot is missing, which is not the case. While the discretisation error increases with increasing dynamics, the asymptotic convergence rates remain unchanged, as they should. Additionally, one can observe that the absolute errors in $\boldsymbol{v}$ are smallest for all three boundary condition types. This is reasonable, since mainly small velocities occur during the simulations. For the relative discretisation errors in $\boldsymbol{v}$, naturally the opposite is the case due to the division by small norms. As the absolute values for the pressure dof $\boldsymbol{w}$ are smaller in magnitude than the ones for $\boldsymbol{u}$ and $\boldsymbol{v}$, they do not influence the slope of the overall state $\bar{\boldsymbol{x}}$, which is dominated by the $\boldsymbol{u}$ and $\boldsymbol{v}$ errors. Further, for the separately calculated errors in x-, y- and z-direction, one can observe that there is no significant difference in the values as well as the slopes among them.

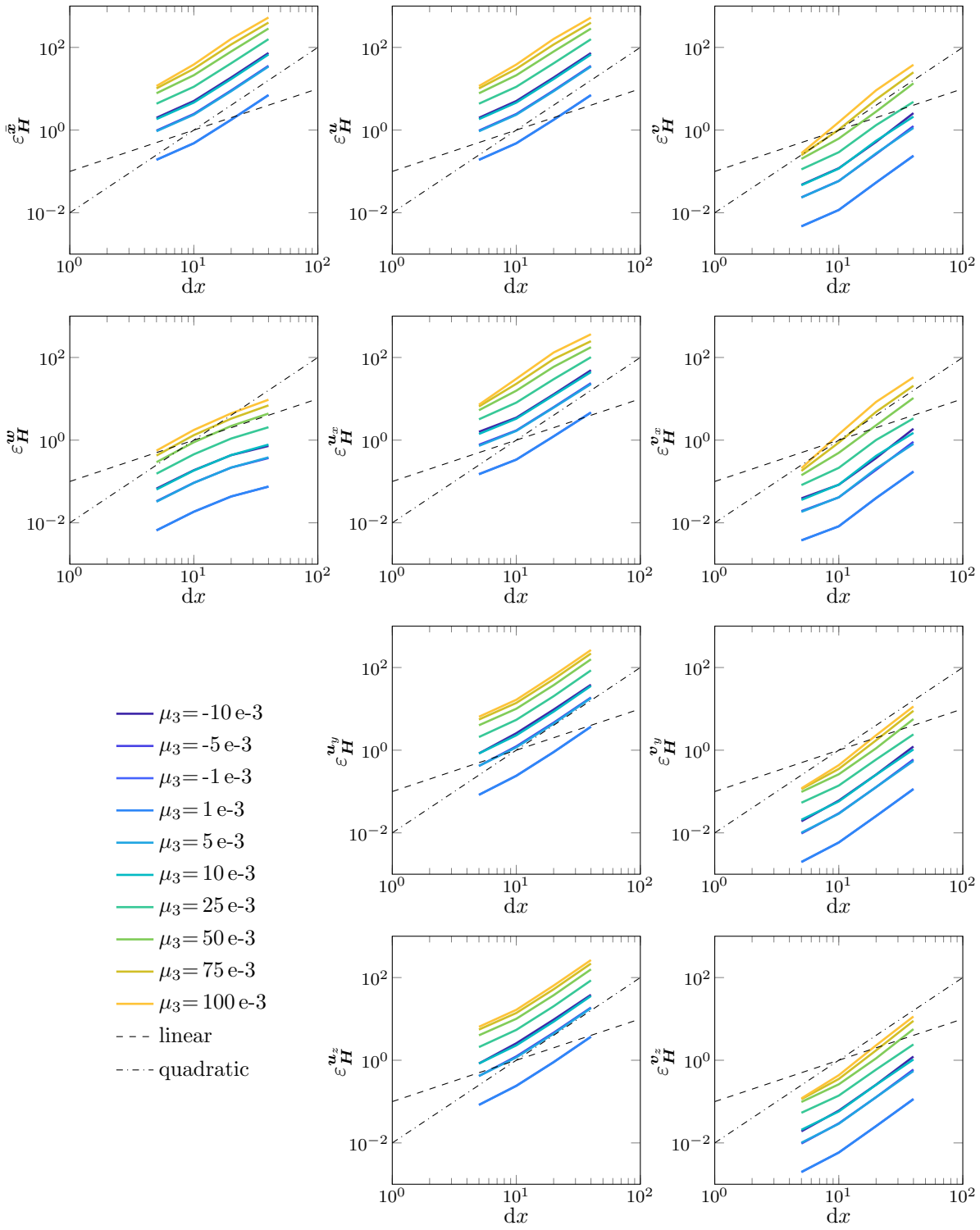


Figure 6.29: *Absolute errors in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$ for BC1.*
Each of the axes shows the results from ten simulations performed with a different final magnitude of the applied, linearly increasing compression/traction force μ_3 [MPa] to the right end together with a linear (dashed line) and a quadratic (dash-dotted line) slope.

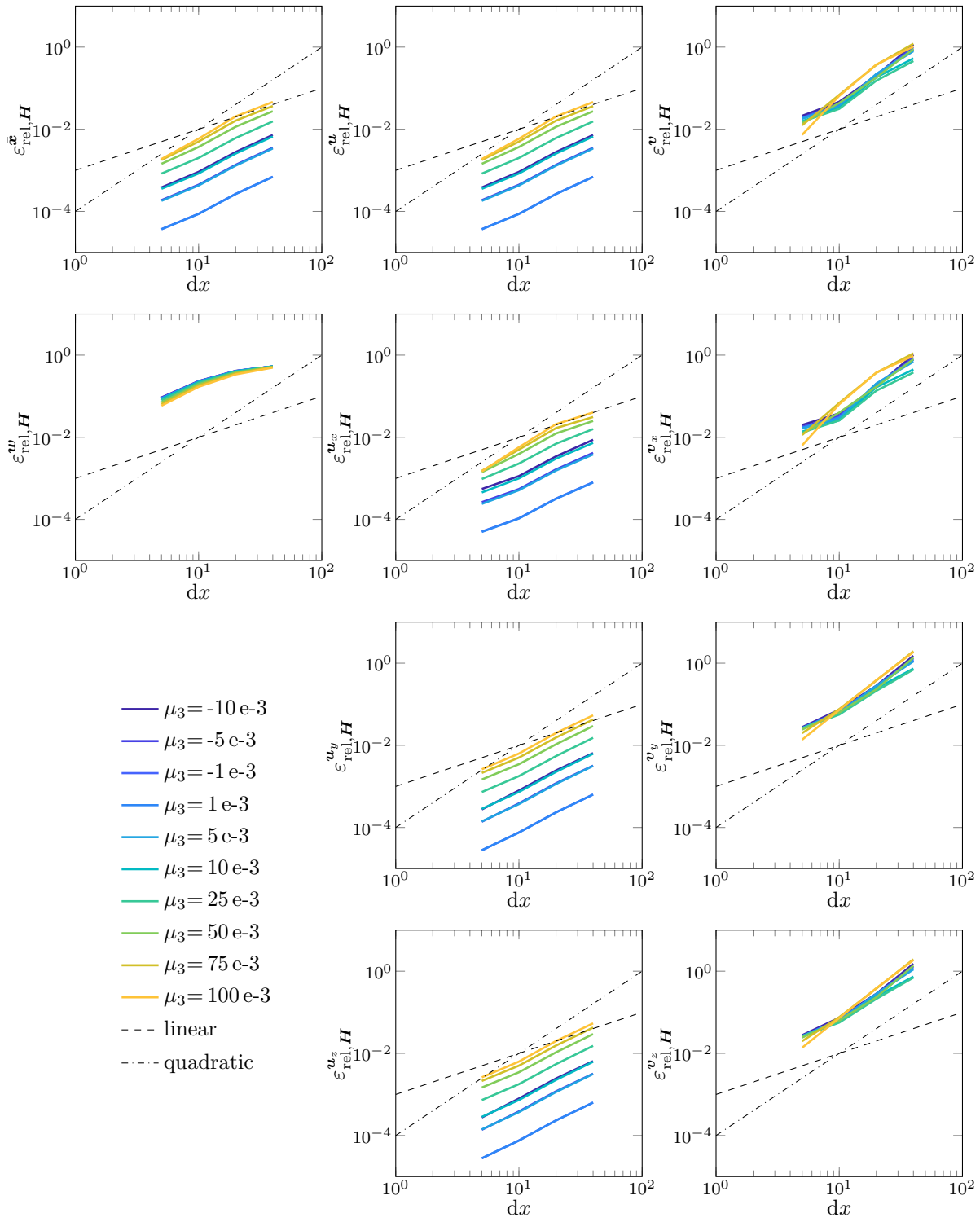


Figure 6.30: *Relative errors in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$ for BC1.*
Each of the axes shows the results from ten simulations performed with a different final magnitude of the applied, linearly increasing compression/traction force μ_3 [MPa] to the right end together with a linear (dashed line) and a quadratic (dash-dotted line) slope.

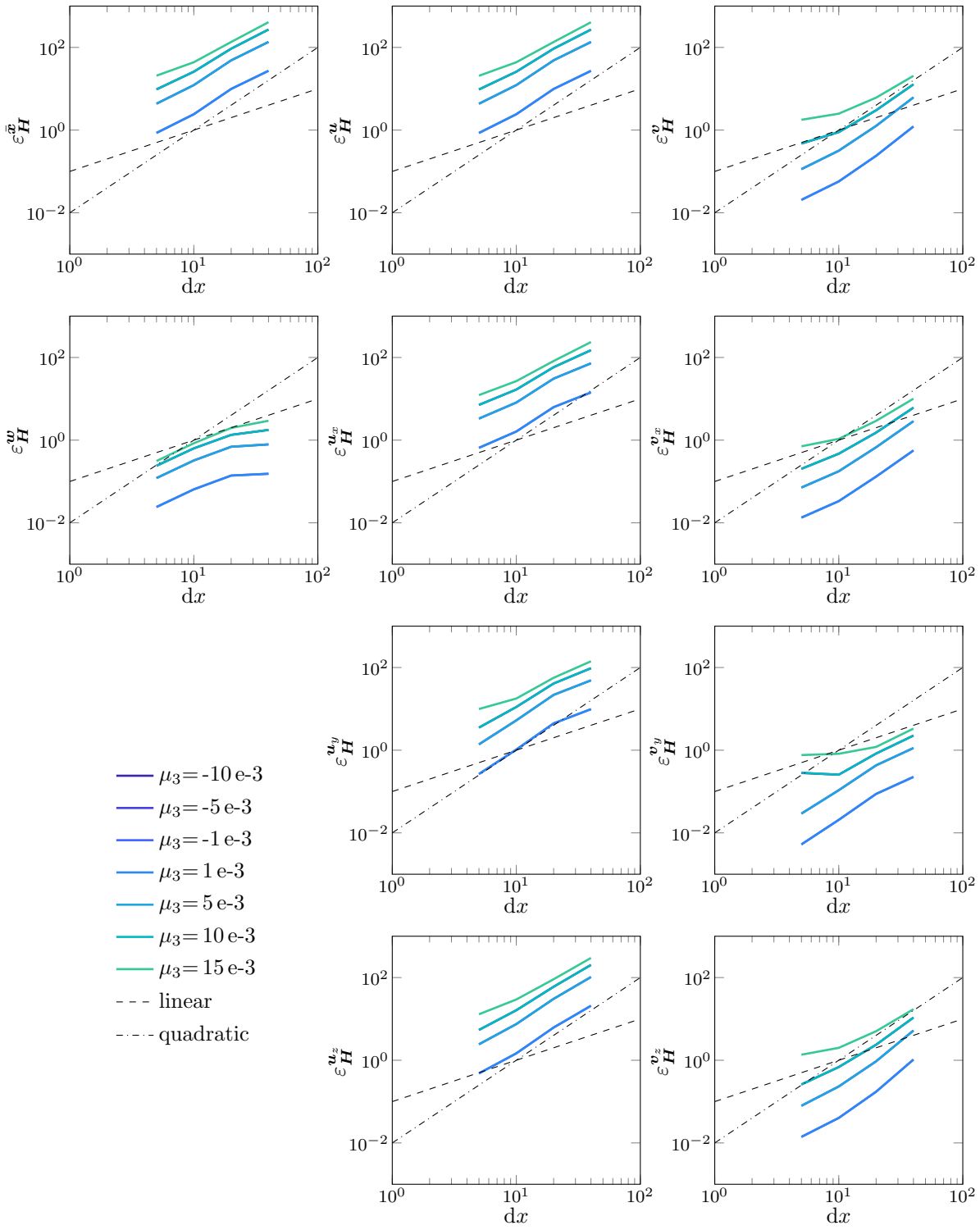


Figure 6.31: *Absolute errors in the discrete energy norm* $\|\cdot\|_{\boldsymbol{H}}$ *for BC2.*
Each of the axes shows the results from seven simulations performed with a different final magnitude of the applied, linearly increasing compression/traction force μ_3 *[MPa] to the right end together with a linear (dashed line) and a quadratic (dash-dotted line) slope. Note the missing curve for* $\mu_3 = 20\,e{-}3$ *due to the convergence issues described in* Section 6.2.

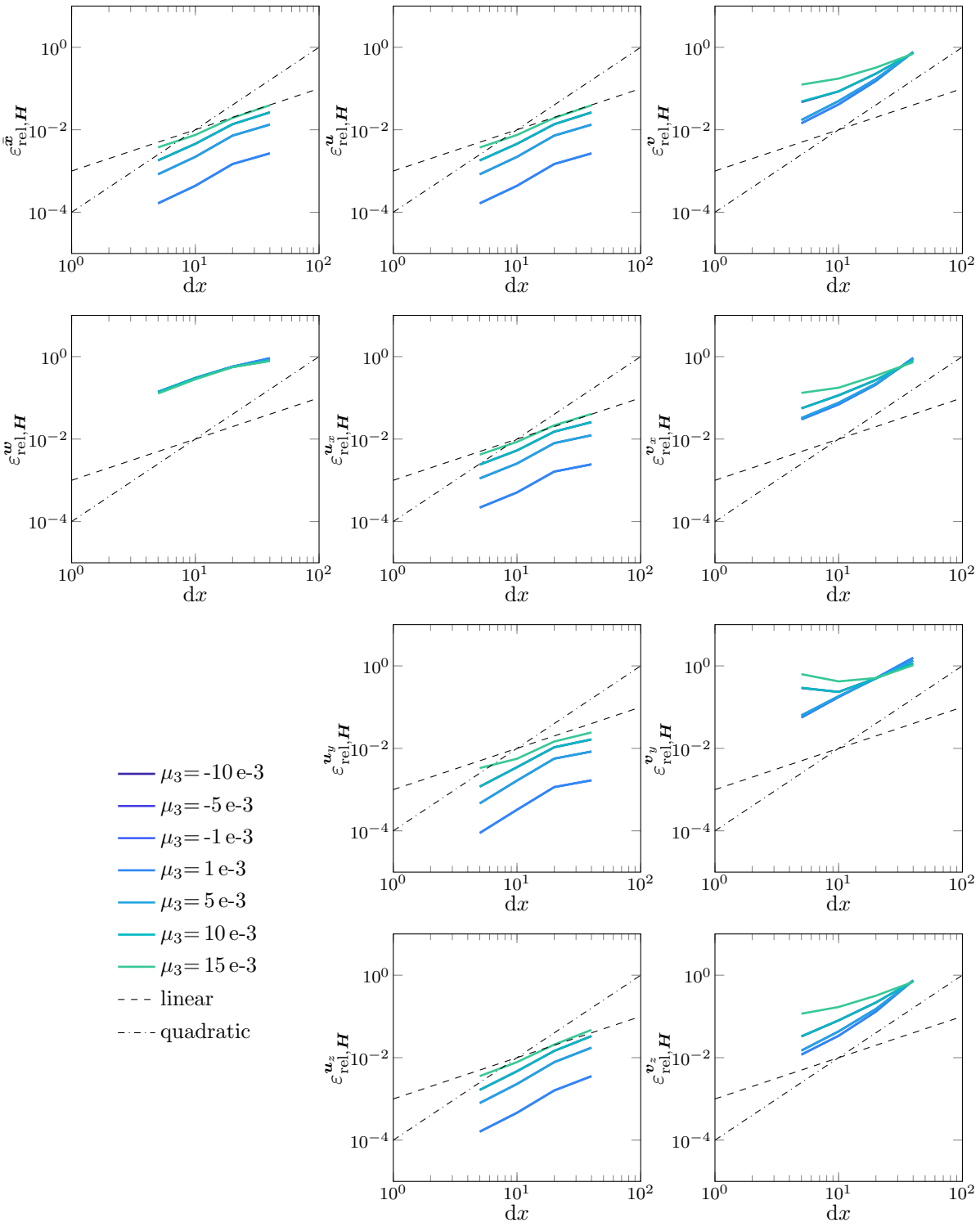


Figure 6.32: *Relative errors in the discrete energy norm* $\|\cdot\|_{\boldsymbol{H}}$ *for BC2.*
Each of the axes shows the results from seven simulations performed with a different final magnitude of the applied, linearly increasing compression/traction force μ_3 *[MPa] to the right end together with a linear (dashed line) and a quadratic (dash-dotted line) slope. Note the missing curve for* $\mu_3 = 20\,e{-}3$ *due to the convergence issues described in* Section 6.2.

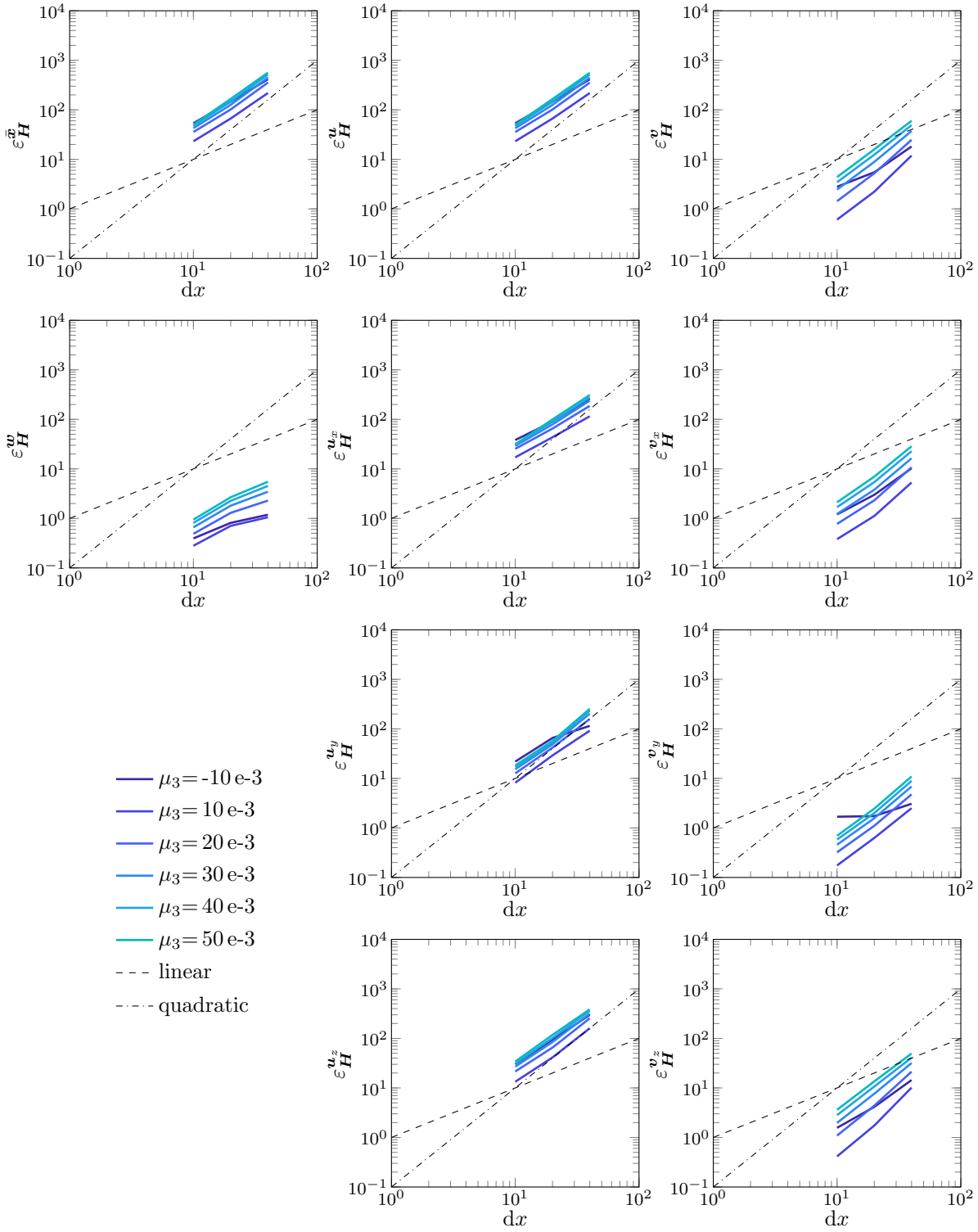


Figure 6.33: *Absolute errors in the discrete energy norm* $\|\cdot\|_{\boldsymbol{H}}$ *for BC3.*
Each of the axes shows the results from six simulations performed with a different final magnitude of the applied, linearly increasing compression/traction force μ_3 *[MPa] to the right end together with a linear (dashed line) and a quadratic (dash-dotted line) slope. Note that these plots were generated with* $\mathrm{d}x = 5$ *as fine solution due to convergence issues for some parameters in the simulation with* $\mathrm{d}x$*=2.5 (c.f.* Section 6.2*).*

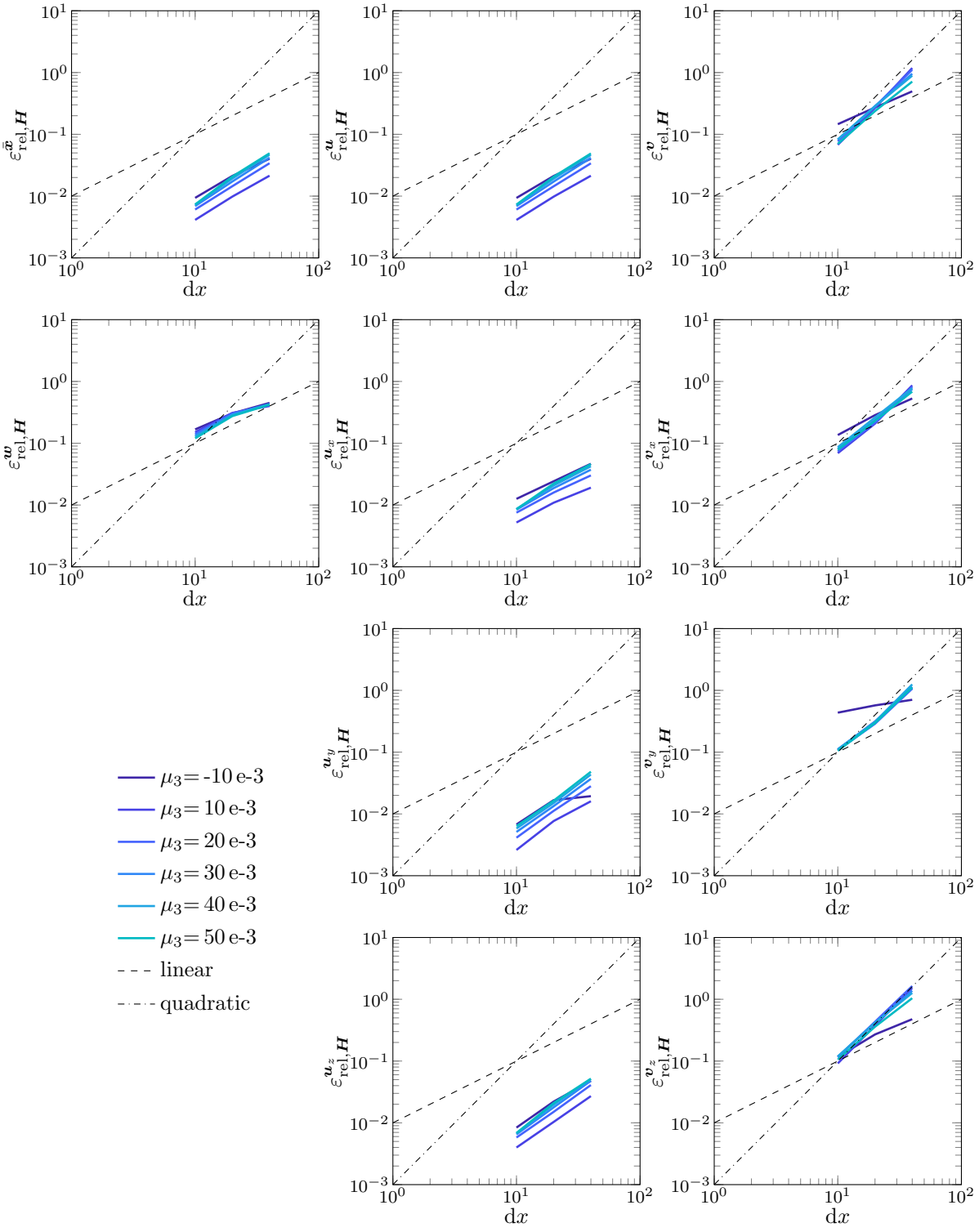


Figure 6.34: *Relative errors in the discrete energy norm* $\|\cdot\|_{\boldsymbol{H}}$ *for BC3. Each of the axes shows the results from six simulations performed with a different final magnitude of the applied, linearly increasing compression/traction force* μ_3 *[MPa] to the right end together with a linear (dashed line) and a quadratic (dash-dotted line) slope. Note that these plots were generated with* $\mathrm{d}x = 5$ *as fine solution due to convergence issues for some parameters in the simulation with* $\mathrm{d}x$*=2.5 (c.f.* Section 6.2*).*

Lastly, a comment on the magnitude of the discretisation errors: For the absolute errors, they are in the range of 1 e+1 for the finest discretisation even, which at first thought seems relatively large. However, thinking about the way the errors are calculated and how fine the spatial and temporal discretisations are, one could say that requiring each node to be at exactly the same position as the corresponding node for every single time step, might be a too strict error measure. After all, for such dynamic simulations including oscillations, one could argue that it is sufficient to capture the overall behaviour correctly.

6.3.3 Example with realistic skeletal muscle material behaviour

For Examples 3, an approximate discretisation error is calculated in the same way as for Examples 2. For the cubic geometry, i.e. Example 3A, BCxz and BCxact, the same spatial discretisations as for Examples 2 were chosen (c.f. Table 6.7). However, due to time constraints and the observed convergence issues in Examples 2 with $\mathrm{d}x = 2.5$, here with a finest spatial discretisation of $\mathrm{d}x = 5$. For the idealised fusiform geometry in Example 3B, the number of elements and dof are different due to the different meshing strategy as explained in Figure 6.21. Those are listed in Table 6.8. The time step size was set to $\mathrm{d}t = 0.1\,\mathrm{ms}$ for every simulation, except for the BCxz scenario, where it had to be decreased to $\mathrm{d}t = 0.025\,\mathrm{ms}$ (c.f. Section 6.2).

elements	u/v-dof ($3N$)	w-dof (N_p)	total dof
$5 \times 4 = 20$	825	54	$2 \times 825 + 54 = 1\,704$
$10 \times 4 = 40$	1 575	99	$2 \times 1\,575 + 99 = 3\,249$
$20 \times 12 = 240$	7 011	357	$2 \times 7\,011 + 357 = 14\,379$
$20 \times 40 = 800$	21 771	1 029	$2 \times 21\,771 + 1\,029 = 44\,571$
$40 \times 40 = 1\,600$	43 011	2 009	$2 \times 43\,011 + 2\,009 = 88\,031$

Table 6.8: *The chosen discretisations and resulting number of elements and dof for the convergence analysis of Example 3B. The first factor in the multiplication for the number of elements stands for the number of slices, while the second one results from the chosen number of layers (1 layer corresponds to 4 elements in each cross section slice, 2 layers to 12 and 4 layers to 40).*

For Example 3B with the fusiform muscle geometry, not every node of the coarser meshes always has a matching node in the finest mesh. This is due to the meshing strategy (c.f. Figure 6.21), which for a different number of layers recalculates the nodal positions instead of e.g. simply inserting additional nodes to the already existing ones from coarser meshes. This entailed difficulties in the error computation for the convergence study, which were overcome by simply including only those nodes in the error computation that had a matching node in the finest mesh. For the meshes resulting from one layer, i.e. the 20 elements mesh and the 40 elements mesh, 84 % of the nodes had a matching one in the 1 600 elements mesh (231 out of 275 and 441 out of 525 nodes, respectively), while for the 240 elements mesh resulting from two layers, 1 845 out of 2 337 nodes are found in the finest mesh. This corresponds to 79 % and was considered to be sufficient for producing meaningful results.

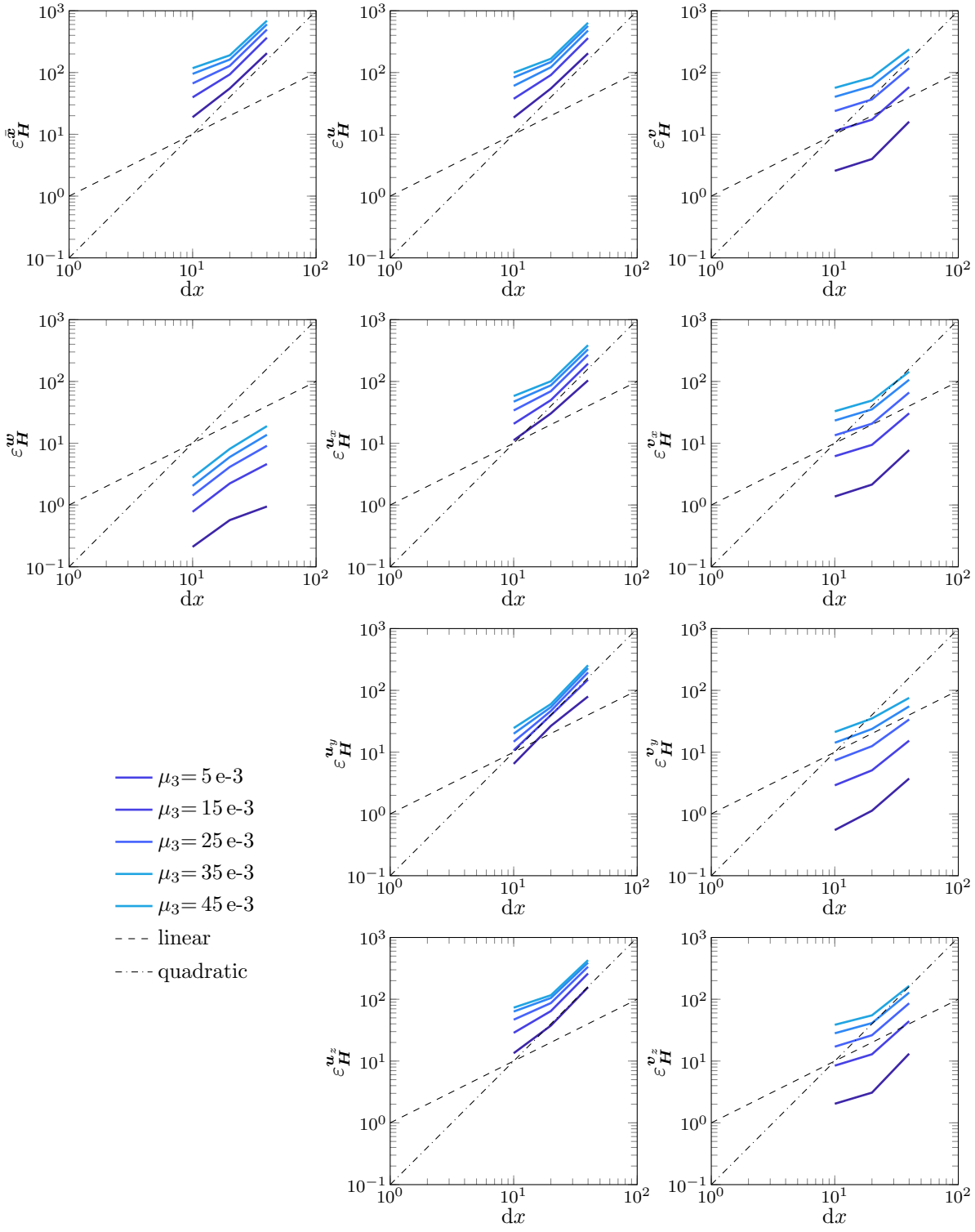


Figure 6.35: *Absolute errors in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$ for Example 3A, BCxz. Each of the plots shows the results from five simulations performed with a different final magnitude of the applied, linearly increasing traction force μ_3 [MPa] to the right end together with a linear (dashed line) and a quadratic (dash-dotted line) slope.*

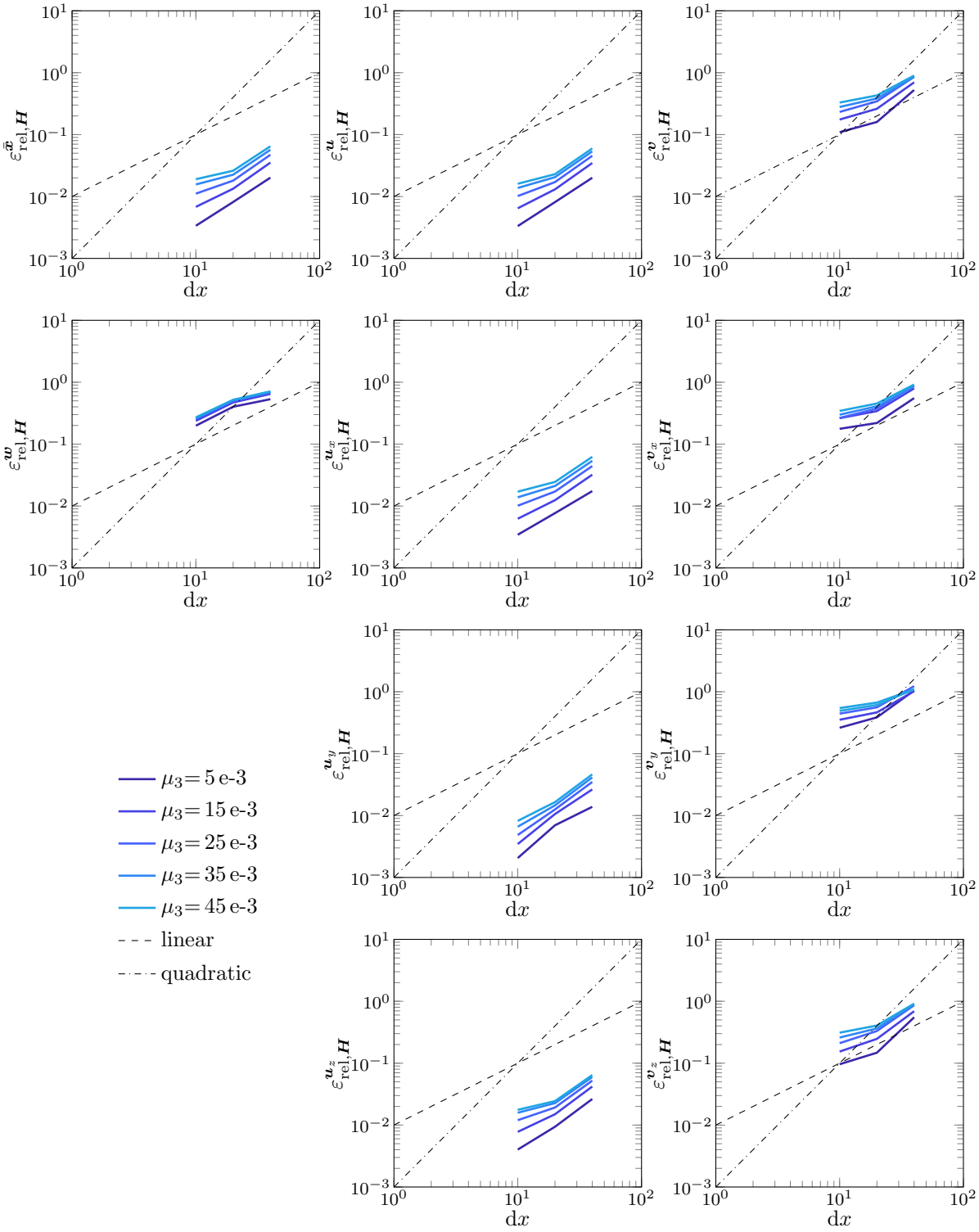


Figure 6.36: *Relative errors in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$ for Example 3A, BCxz. Each of the plots shows the results from five simulations performed with a different final magnitude of the applied, linearly increasing traction force μ_3 [MPa] to the right end together with a linear (dashed line) and a quadratic (dash-dotted line) slope.*

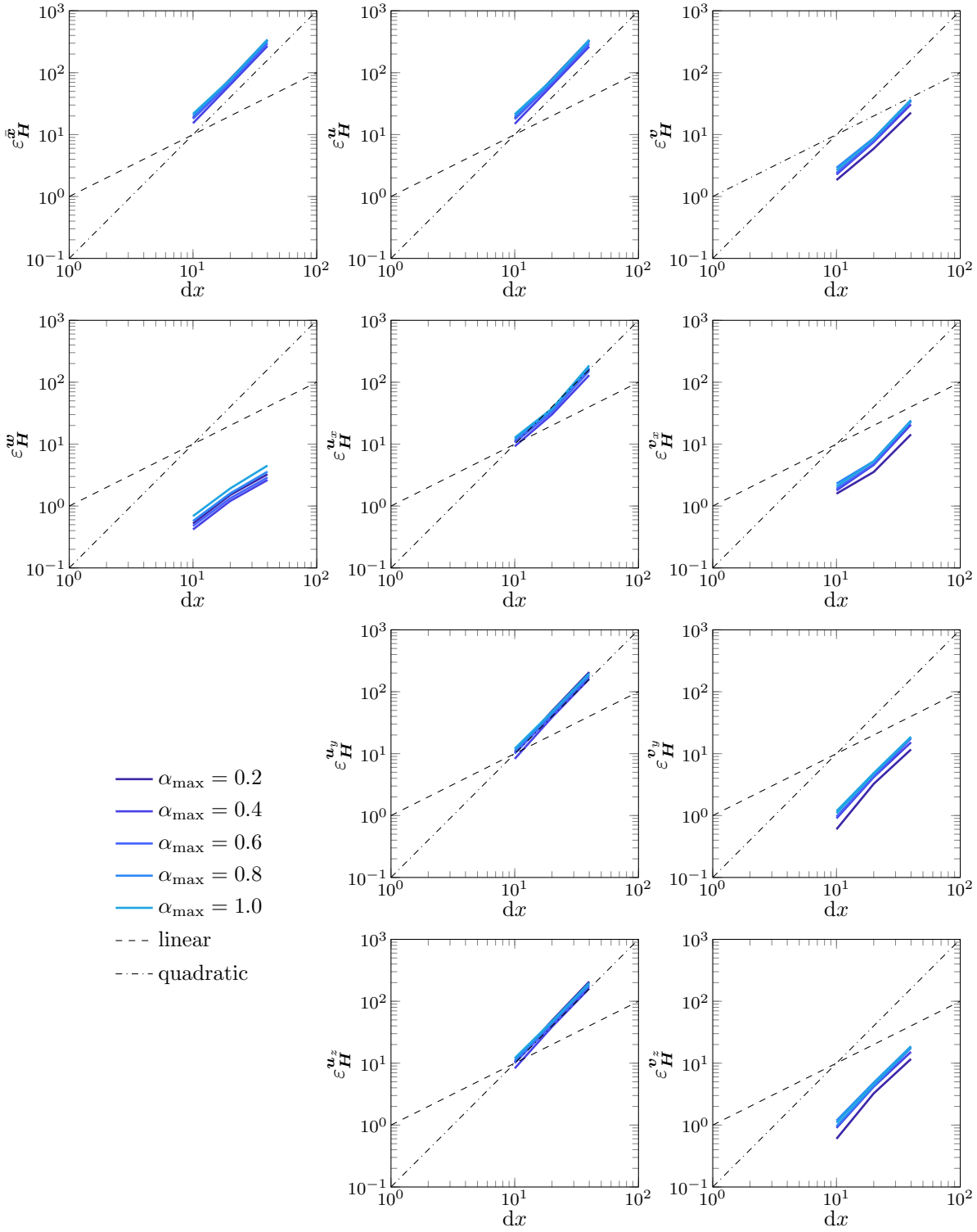


Figure 6.37: *Absolute errors in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$ for Example 3A, BCxact. Each of the plots shows the results from five simulations performed with a different final activation $\alpha_{\max}$ together with a linear (dashed line) and a quadratic (dash-dotted line) slope.*

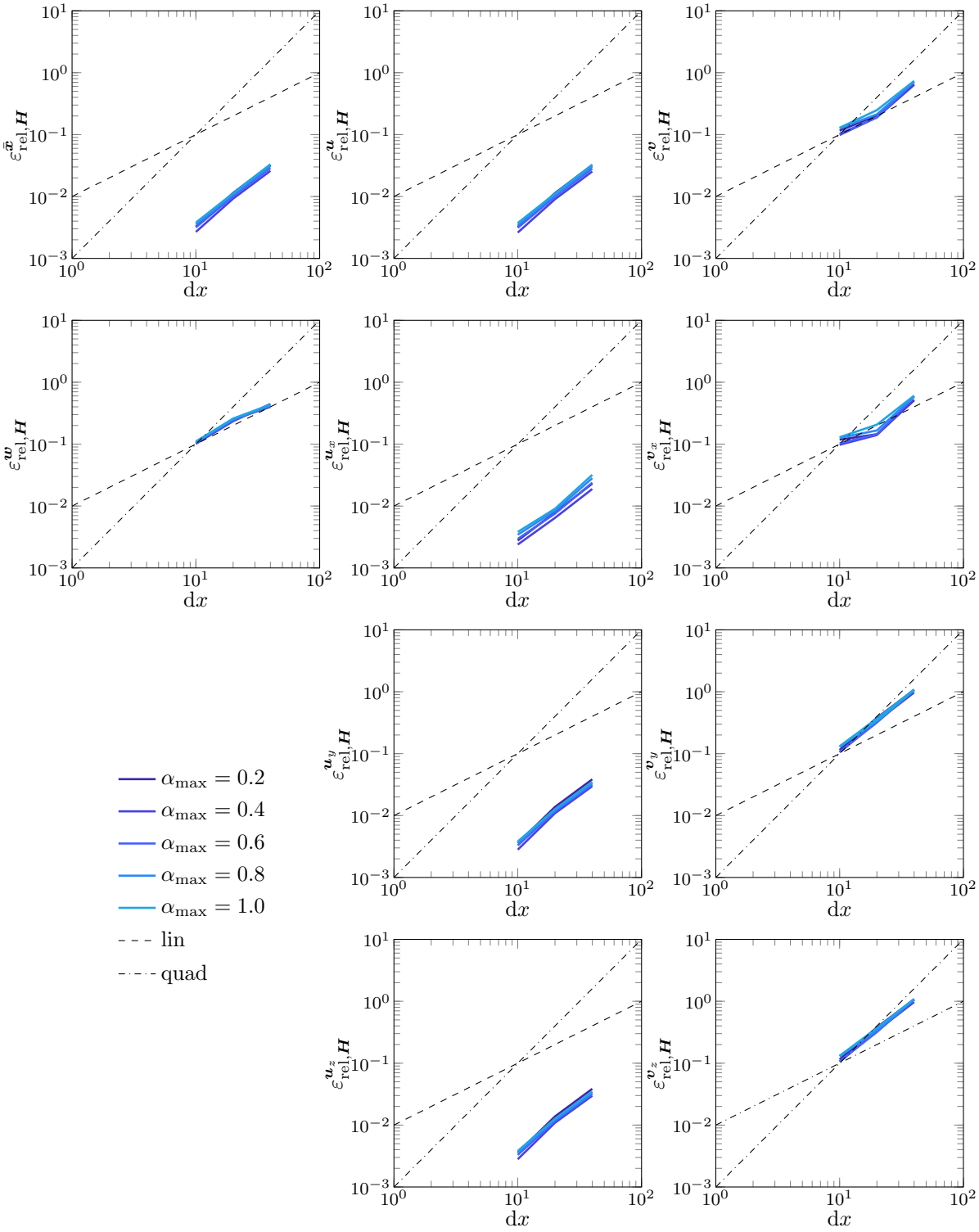


Figure 6.38: *Relative errors in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$ for Example 3A, BCxact. Each of the plots shows the results from five simulations performed with a different final activation $\alpha_{\max}$ together with a linear (dashed line) and a quadratic (dash-dotted line) slope.*

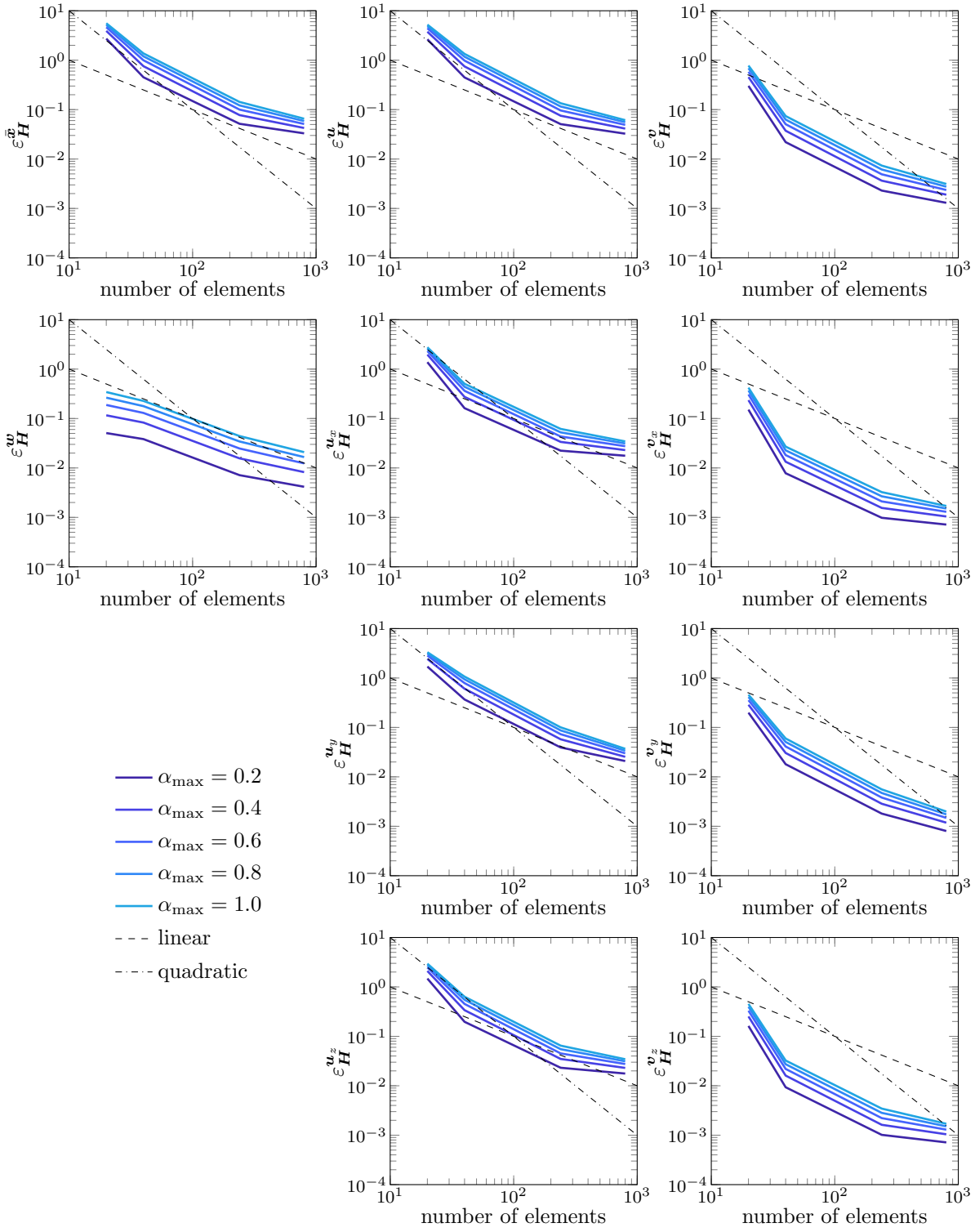


Figure 6.39: *Absolute errors in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$ for Example 3B. Each of the plots shows the results from five simulations performed with a different final activation $\alpha_{\max}$ together with a linear (dashed line) and a quadratic (dash-dotted line) slope.*

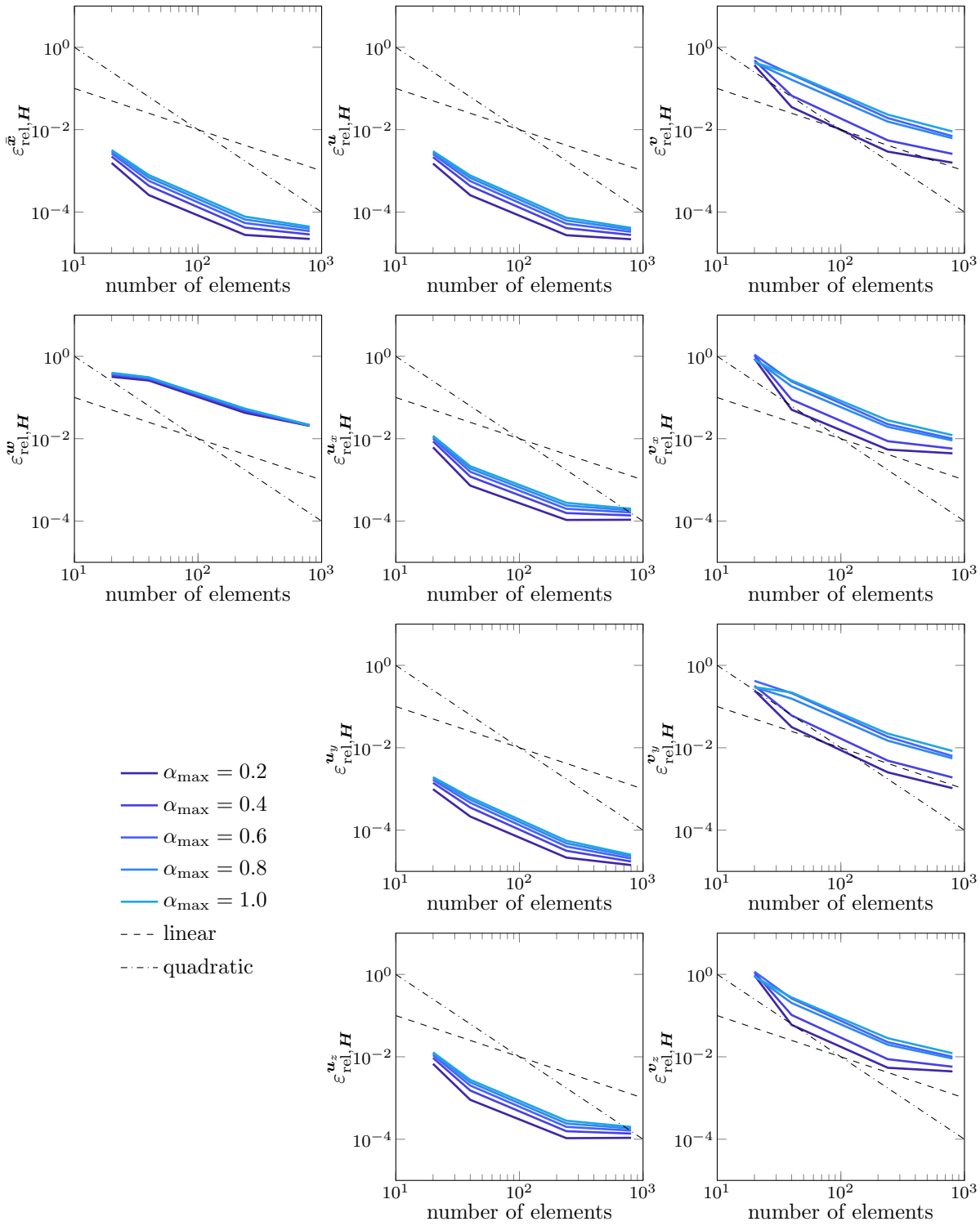


Figure 6.40: *Relative errors in the discrete energy norm $\|\cdot\|_{\boldsymbol{H}}$ for Example 3B. Each of the plots shows the results from five simulations performed with a different final activation $\alpha_{\max}$ together with a linear (dashed line) and a quadratic (dash-dotted line) slope.*

As the 800 elements mesh was constructed with four layers, i.e. the same number of layers as the finest mesh, and merely refined by duplicating the number of slices, here all nodes

had a corresponding node. Furthermore, the meshes for this realistic geometry have a non-uniform mesh size dx not only different in x-, y- and z-direction, but also varying along the y-axis due to the decreasing diameter of the muscle while keeping the number of elements in the cross section constant. Therefore, the absolute and relative errors are plotted against the number of elements instead against dx and thus the convergence plots show a decreasing slope here. Figures 6.35–6.38 show the results for Examples 3A, i.e. the ones, where the material complexity was increased. For the BCxz case, which, concerning the applied force boundary conditions, corresponds to the BC3 case of Example 2, the discretisation errors behave very similar to those cases. As anticipated, the errors in the $\boldsymbol{u}$ and $\boldsymbol{v}$ dof show a quadratic asymptotic convergence rate and the ones in the pressure dof $\boldsymbol{w}$ decay linearly. Merely the absolute values of the errors are around one order of magnitude higher for the more complex material in this Example 3A. For the BCxact case, which is the first test case with additional active material behaviour, the discretisation errors decrease very nicely and there is no significant difference whether the muscle was fully activated ($\alpha_{\max} = 1$) or just to a fraction.
Figures 6.39–6.40 show the results for Example 3B, i.e. the fusiform muscle geometry. While the discretisation errors, both the absolute and the relative one, in the pressure dof $\boldsymbol{w}$ again show a linear decay, the errors in the $\boldsymbol{u}$ and $\boldsymbol{v}$ dof show asymptotic convergence rates a little less than quadratic. This is still in a reasonable range and as visualised in Figure 6.21, the mesh is rather irregular containing some not really well defined elements, which explains the slight worsening. All other observations are similar to the ones made for the previous examples.

7 Analysis of different ROM

The purpose of this chapter is to find a suitable combination of projection spaces for the three different types of coefficient vectors $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$. This means that not only the calculation of the POD bases $\boldsymbol{V}_u$, $\boldsymbol{V}_v$, $\boldsymbol{V}_w$ themselves, but also the investigation on how they perform in combination and the choice of the size of each reduced spaces needs to be investigated. Suitable in this context shall mean that the obtained reduced-order model is not only faster than the full-order model, but also, or as a first aim more importantly even, at least as stable as the full-order model, i.e. converging for the chosen parameter ranges with a sufficient accuracy. These investigations are carried out in Sections 7.1–7.3 utilising Examples 1 and Examples 2.
Subsequently, in Section 7.4, the most satisfactory approach is applied to Examples 3 in order to examine its suitability for models with increasing complexity.

7.1 The influence of the combination of reduced position and velocity space

For all simulations in this section, training data $\boldsymbol{S} = [\boldsymbol{S}_u, \boldsymbol{S}_v, \boldsymbol{S}_w]^T \in \mathbb{R}^{\bar{d}\times n}$, $n := (n_t + 1) \cdot n_p$, is computed during the offline phase. The training parameter is the applied force, whose final magnitude, μ_3, varies in a chosen interval. If no further specification is made, the setup of the examples is the same as described in Chapter 6. Every POD basis is computed by means of a simple svd $(\boldsymbol{S}_\star)$, i.e. setting $\boldsymbol{\mathcal{A}} := \boldsymbol{I}$.
During the online phase, the reduced system is assembled and subsequently tested for the same parameter values as chosen in the offline phase, i.e. we are merely interested in reproducing the same test cases and compare the performance of different reduced models. This way, the solutions obtained by the FOM during the offline phase can serve as the true solution to compare with.
Similar to the discretisation error for the convergence analysis, the absolute error $\varepsilon^{(\star)}_{\mathcal{L}^2}$, $(\star) \in \{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}\}$ in the discrete $\mathcal{L}^2$-norm is computed. Let $(\star)^{\mathrm{F}}$ be the FE-solution and $(\star)^{\mathrm{R}}$ the corresponding solution obtained by the reduced model. For each time step t_k, $k = 0, ..., n_t$, we compute e.g.

$$\begin{aligned}\varepsilon^{\boldsymbol{u}}_{\mathcal{L}^2}(t_k) &:= \left\|\boldsymbol{u}^{\mathrm{F}}(:,t_k) - \boldsymbol{u}^{\mathrm{R}}(:,t_k)\right\|_{\mathcal{L}^2} \approx \sqrt{\mathrm{d}x^3} \cdot \left\|\boldsymbol{u}^{\mathrm{F}}(:,t_k) - \boldsymbol{u}^{\mathrm{R}}(:,t_k)\right\|_2 \\ &= \sqrt{\mathrm{d}x^3} \cdot \sqrt{\sum_{i=1}^{3N} \left(u_i^{\mathrm{F}}(:,t_k) - u_i^{\mathrm{R}}(:,t_k)\right)^2}\,, \end{aligned} \tag{7.1}$$

and subsequently

$$\varepsilon_{\mathcal{L}^2}^{(\star)} := \frac{1}{n_t+1}\sum_{k=0}^{n_t}\varepsilon_{\mathcal{L}^2}^{(\star)}(t_k)\,, \tag{7.2}$$

i.e. the mean over all time steps.

7.1.1 Results and problems when using the naive approach

In this section, Examples 1A, 1B and 1C (c.f. Section 6.1.1) are used for a first investigation of building a ROM. The intuitively obvious choice described in Section 5.2, of computing $\boldsymbol{V}_\star = \mathrm{svd}\,(\boldsymbol{S}_\star)$, $\star \in \{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}\}$, is investigated here. The results show that, unfortunately, obtaining a ROM is not as straight forward as one might have assumed or hoped for.

Offline phase - computation of training data

For the FOM, the discretisation $\mathrm{d}x = 0.05\,\mathrm{mm}$ resulting in 320 elements and roughly $(2 \times 9\,000 + 500 \approx) 20\,000$ system dof is chosen (c.f. Tables 6.5,6.6). The intervals/values of the final magnitude of the applied force and hence the dimensions of the obtained snapshot matrices are

1A: $\mu_3 \in \{-1, -0.1, 0.1, 1, 10, 100\}\,\mathrm{e}{-3}\,\mathrm{MPa}$, i.e. $n_p = 6$ logarithmically spaced values,
$\Longrightarrow \boldsymbol{S}_u, \boldsymbol{S}_v \in \mathbb{R}^{9\,880\times 1\,806}$, $\boldsymbol{S}_w \in \mathbb{R}^{525\times 1\,806}$,

1B: $\mu_3 \in \{-10, -1, -0.1, 0.1, 1, 10\}\,\mathrm{e}{-3}\,\mathrm{MPa}$, i.e. $n_p = 6$ logarithmically spaced values,
$\Longrightarrow \boldsymbol{S}_u, \boldsymbol{S}_v \in \mathbb{R}^{8\,856\times 1\,806}$, $\boldsymbol{S}_w \in \mathbb{R}^{525\times 1\,806}$,

1C: $\mu_3 \in \{-0.1, 0.1, 1, 10\}\,\mathrm{e}{-3}\,\mathrm{MPa}$, i.e. $n_p = 4$ logarithmically spaced values,
$\Longrightarrow \boldsymbol{S}_u, \boldsymbol{S}_v \in \mathbb{R}^{9\,225\times 1\,204}$, $\boldsymbol{S}_w \in \mathbb{R}^{525\times 1\,204}$.

Subsequently the SVD was performed on each of the snapshot matrices.

Offline phase - POD and investigation of POD bases

As explained in Section 4.2.2 (see Remark 4) the singular value decay usually serves as an a priori error estimate, which helps to choose an appropriate reduced model size. Figures 7.1–7.3 show the singular values for each of the three examples and each of the three coefficient types.
Additionally, it is interesting to look at the POD vectors, corresponding to the largest singular values, often referred to as the first POD modes. If the subspaces represented by the POD bases are suitable, the first modes should correspond to characteristic deformation states, as every deformation should be representable by a linear combination of those. Figures 7.4, 7.6, 7.8 show the m^{th} mode, $\mathrm{m} \in \{1, 2, 3\}$, for the SVD of the position coefficients $\boldsymbol{u}$, or more precisely,

$$\boldsymbol{u}_0 + \delta\,\boldsymbol{V}_u(:, \mathrm{m}) \tag{7.3}$$

is plotted in a three-dimensional plot for $\delta = \pm 10$.

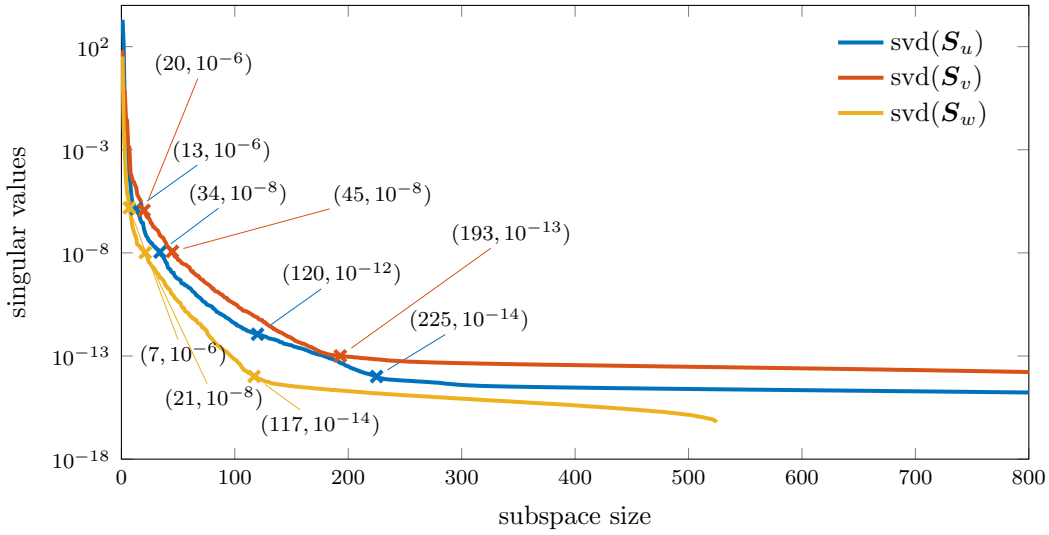


Figure 7.1: *The singular value decays of Example 1A. The plots for* $\boldsymbol{u}$ *and* $\boldsymbol{v}$ *are cut at the* 800^{th} *singular value. For each of the plots some characteristic positions are labelled with their respective values.*

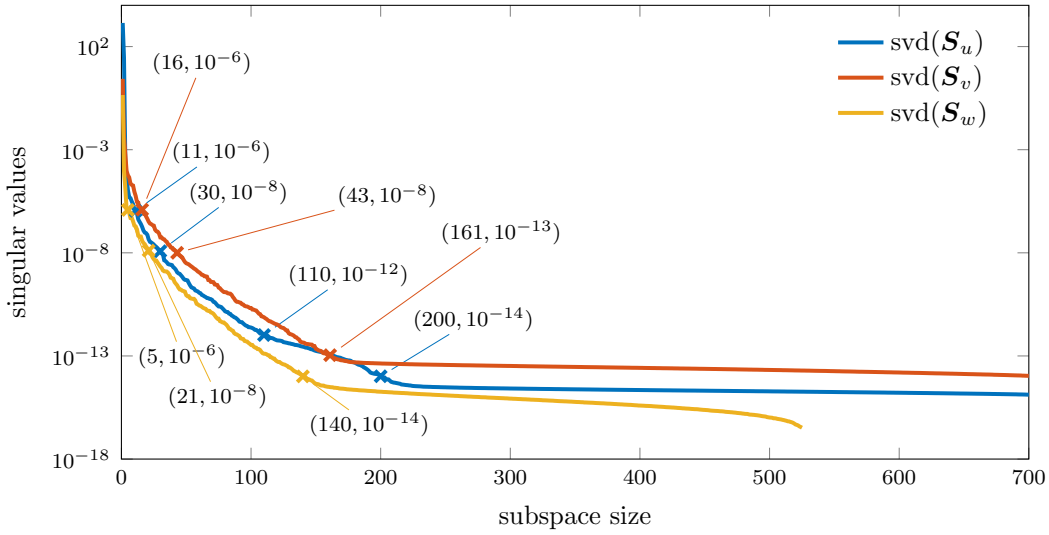


Figure 7.2: *The singular value decays of Example 1B. The plots for* $\boldsymbol{u}$ *and* $\boldsymbol{v}$ *are cut at the* 700^{th} *singular value. For each of the plots some characteristic positions are labelled with their respective values.*

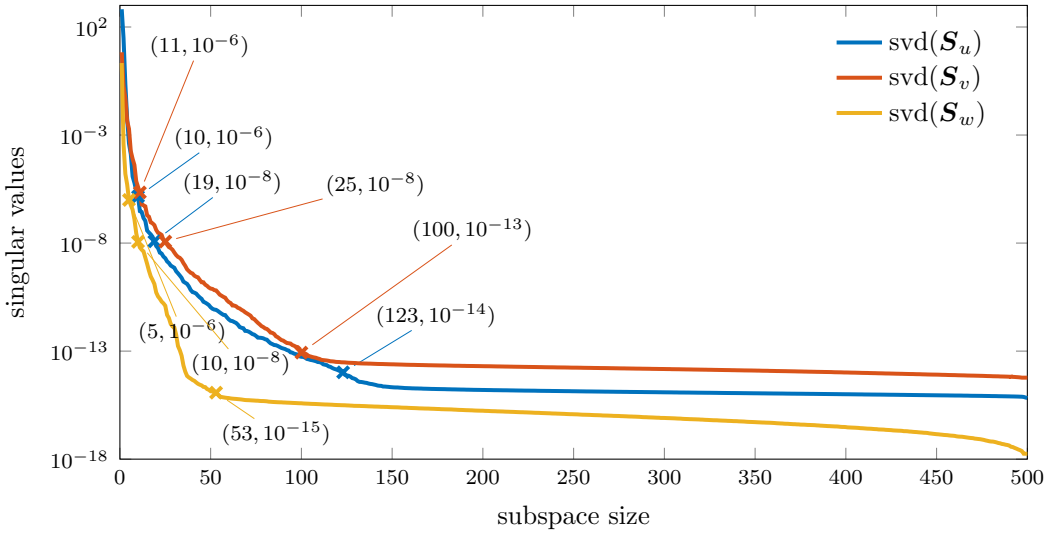


Figure 7.3: *The singular value decays of Example 1C. The plots are cut at the* 500^{th} *singular value. For each of the plots some characteristic positions are labelled with their respective values.*

Furthermore, Figures 7.5, 7.7, 7.9 show the first three pressure modes for each of the examples. As for these simulated cases, the pressure is (at least roughly) constant over the domain for each time step, the first POD mode should represent a constant pressure state. This is the case for all the three examples. Besides, at least from this point of view, further pressure modes should not be necessary for an accurate reduced simulation.

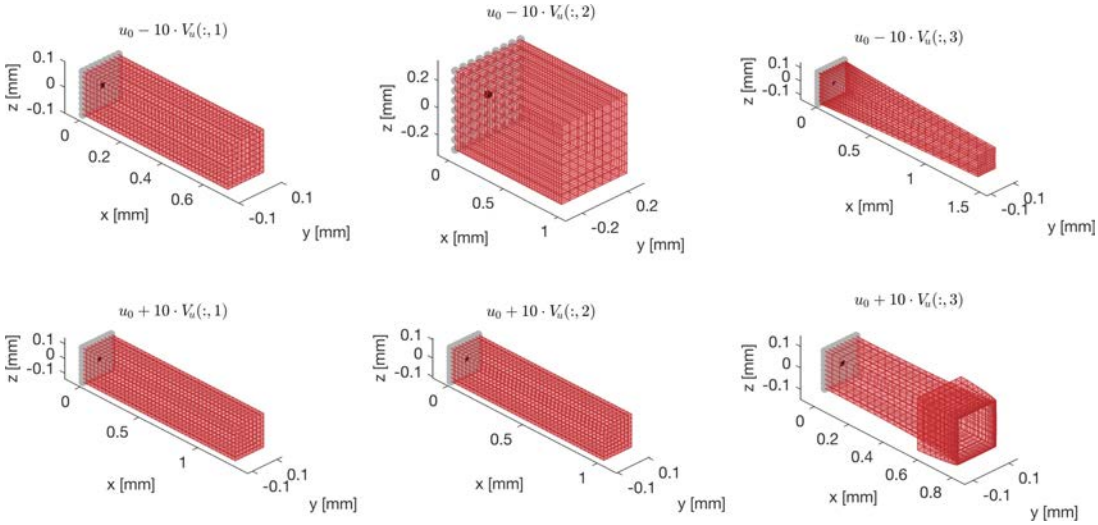


Figure 7.4: *The first deformation POD modes of Example 1A:*
Each column contains one of the three first deformation modes $\boldsymbol{V}_u(:, \mathrm{m})$, $\mathrm{m} = 1, 2, 3$, *added to the initial configuration* $\boldsymbol{u}_0$. *In the top row multiplied by* $\delta = -10$, *in the bottom row multiplied by* $\delta = +10$.

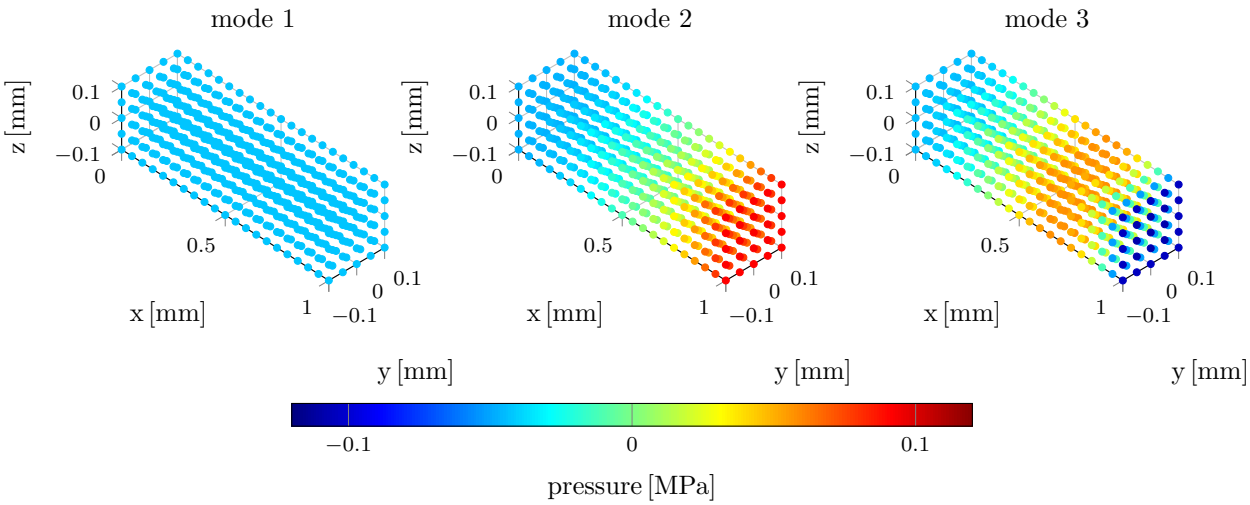


Figure 7.5: *The first three pressure POD modes* $\boldsymbol{V}_w(:, \mathrm{m})$, $\mathrm{m} = 1, 2, 3$, *of Example 1A at each node in the reference configuration.*

Especially for the uniaxial extension case, Example 1A, the first two modes of $\boldsymbol{V}_u$ nicely show the two main deformation modes. While mode 1 leaves the cross section fixed and merely stretches or shortens the body in x-direction, mode 2 keeps the length at 1 mm and only affects the cross sectional area. Every possible deformation state for this simple example should be representable by a linear combination of those first two modes. Furthermore, one can see that mode 3 is already representing an unusual deformation state and thus, at least from this point of view, should not be necessary to be included in the POD basis. Similar observations can be made for Examples 1B and 1C, i.e. the simple and pure shear deformation. While the first two modes still capture characteristic deformation states, the third and further modes show a rather uncharacteristic state.

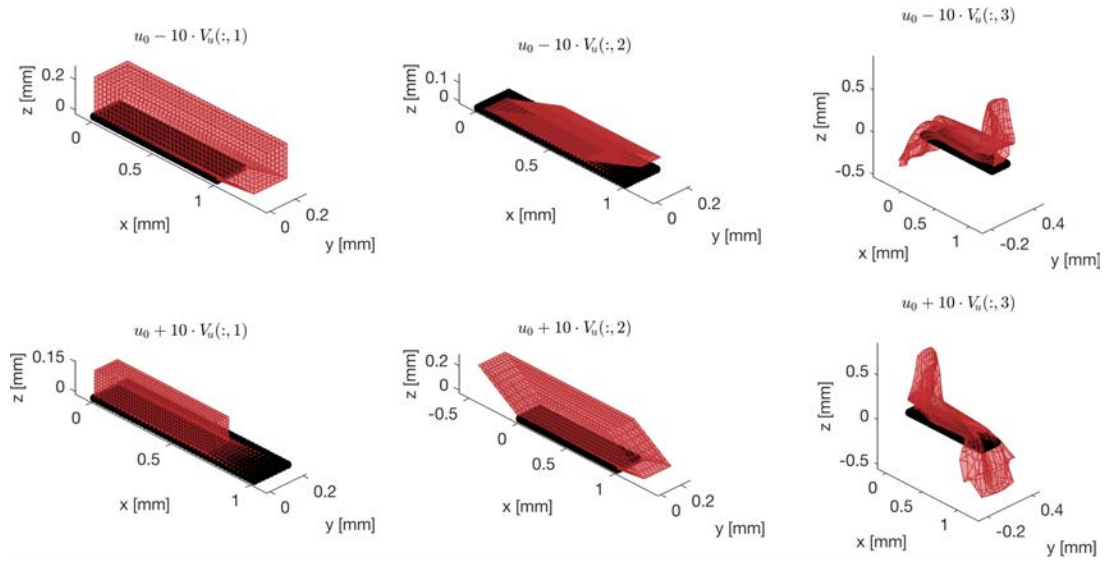


Figure 7.6: *The first deformation POD modes of Example 1B:*
Each column contains one of the three first deformation modes $\boldsymbol{V}_u(:, \mathrm{m})$, $\mathrm{m} = 1, 2, 3$, *added to the initial configuration* $\boldsymbol{u}_0$. *In the top row multiplied by* $\delta = -10$, *in the bottom row multiplied by* $\delta = +10$.

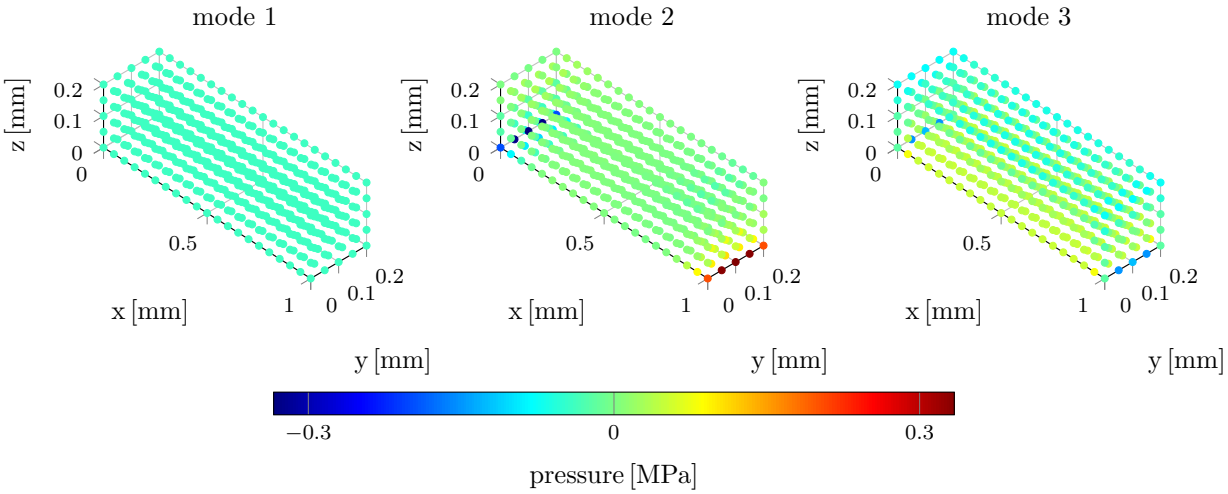


Figure 7.7: *The first three pressure POD modes* $\boldsymbol{V}_w(:, \mathrm{m})$, $\mathrm{m} = 1, 2, 3$, *of Example 1B at each node in the reference configuration.*

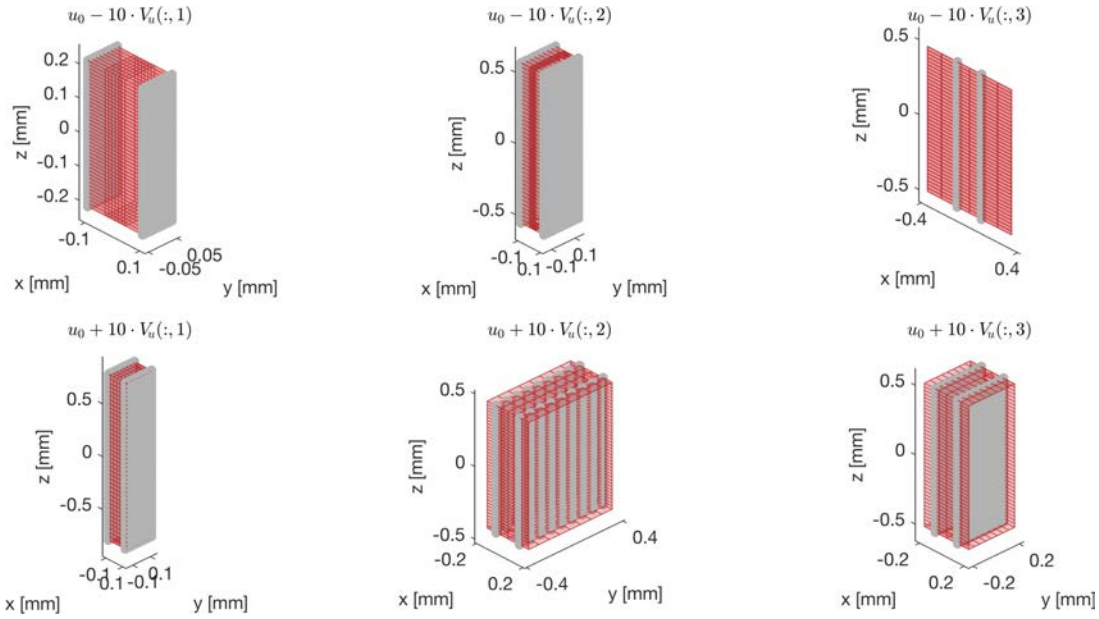


Figure 7.8: *The first deformation POD modes of Example 1C:*
Each column contains one of the three first deformation modes $\boldsymbol{V}_u(:, \mathrm{m})$, $\mathrm{m} = 1, 2, 3$, *added to the initial configuration* $\boldsymbol{u}_0$. *In the top row multiplied by* $\delta = -10$, *in the bottom row multiplied by* $\delta = +10$.

Similar to the visualisation of the pressure modes, Figures 7.10–7.12 show the first three velocity modes plotted over the reference domain. For visualisation purpose, the velocity modes were split into their components in x-, y- and z- direction.

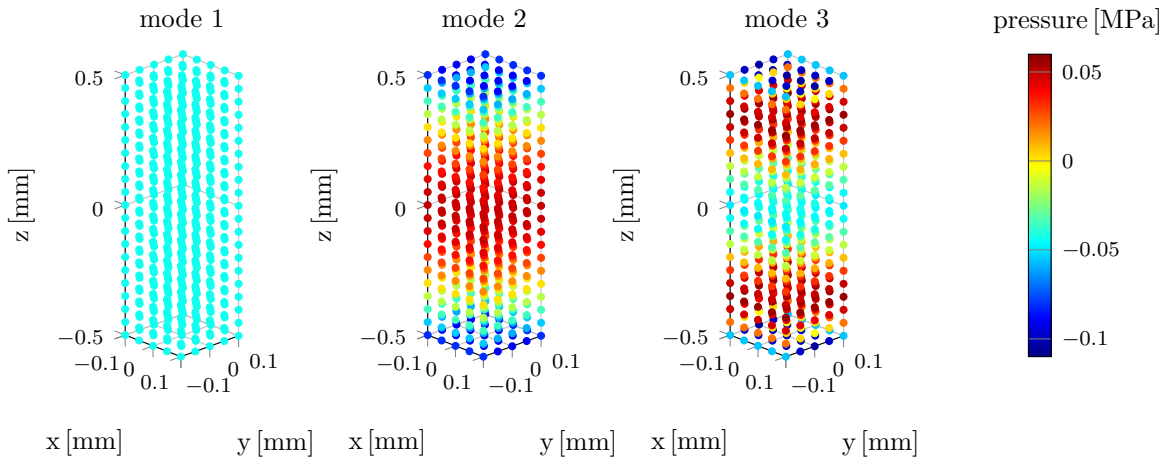


Figure 7.9: *The first three pressure POD modes* $\boldsymbol{V}_w(:, \mathrm{m})$, $\mathrm{m} = 1, 2, 3$, *of Example 1C at each node in the reference configuration.*

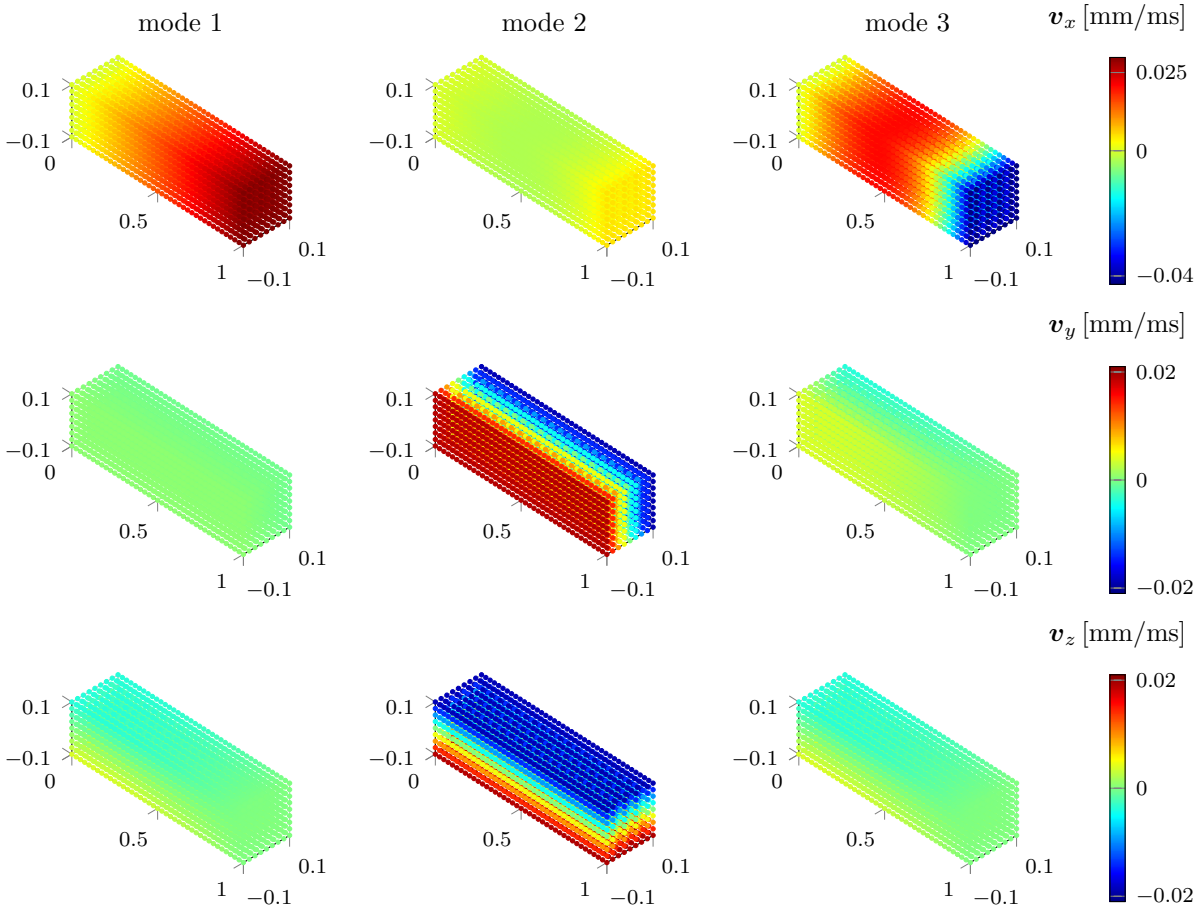


Figure 7.10: *The first three velocity POD modes* $\boldsymbol{V}_v(:, \mathrm{m})$, $\mathrm{m} = 1, 2, 3$, *of Example 1A separated into velocity in x-, y- and z- direction, plotted at the nodes in the reference configuration. (The axes labels are the same as in the previous figures of this example. They are removed on purpose for reasons of saving space.)*

For Example 1A, mode 1 in x-direction shows the largest velocity at the right end, where the force is applied. This makes sense as well as e.g. mode 3 in x-direction, which could represent the swinging back after the final force is applied, as it shows a negative velocity at the nodes of the right end. Moreover, the velocity POD modes in y- and $z-$ direction are the same, simply rotated 90 degrees around the x-axis, correctly reproducing the symmetry of the problem. Modes 2 nicely visualise the change in cross sectional area, where the middle nodes do not move, i.e. have zero velocity, while the nodes to the positive and negative y-/z-direction have negative and positive velocities respectively.

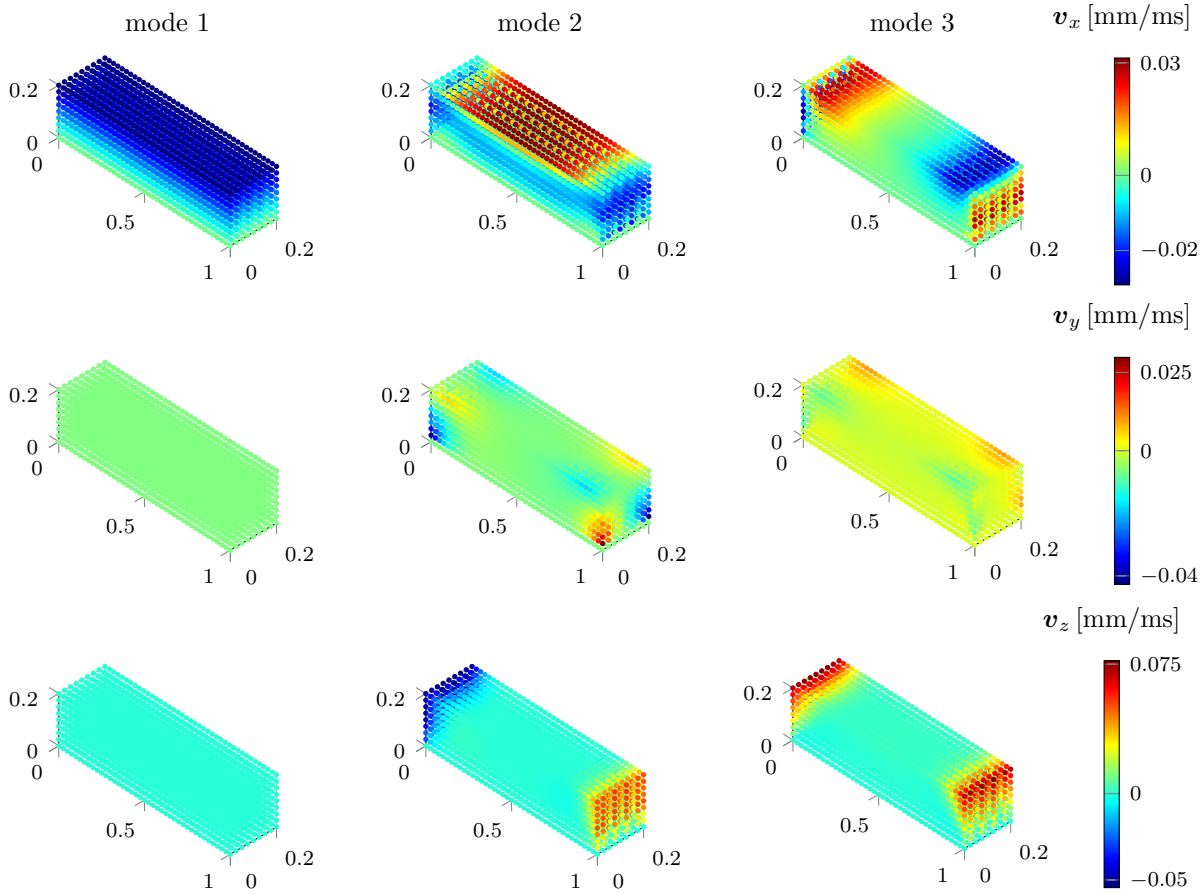


Figure 7.11: *The first three velocity POD modes $\boldsymbol{V}_v(:,\mathrm{m})$, $\mathrm{m} = 1,2,3$, of Example 1B separated into velocity in x-, y- and z- direction, plotted at the nodes in the reference configuration. (The axes labels are the same as in the previous figures of this example. They are removed on purpose for reasons of saving space.)*

For Example 1B, the main characteristic feature, which is nicely represented by the first velocity POD modes, can be observed for mode 1 in x-direction, where the upper half of the nodes move faster than the lower ones, which are close to the zero DIRICHLET boundary condition. The fact that no significant deformations and thus velocities should occur in y- and z- direction here, is correctly captured by modes 1 for both directions.

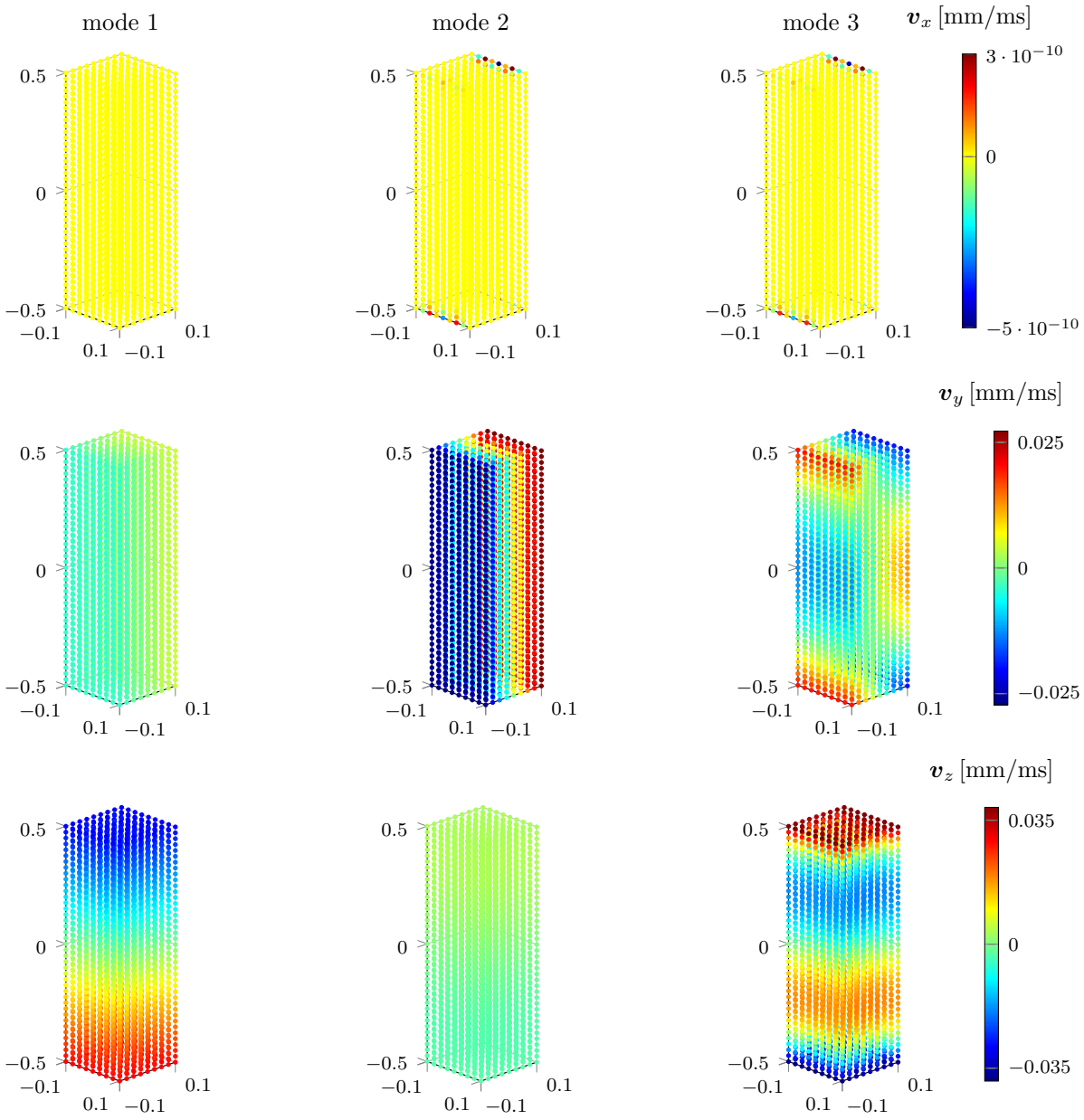


Figure 7.12: *The first three velocity POD modes* $\boldsymbol{V}_v(:, \mathrm{m})$, $\mathrm{m} = 1, 2, 3$, *of Example 1C separated into velocity in* x*-,* y*- and* z*- direction, plotted at the nodes in the reference configuration. (The axes labels are the same as in the previous figures of this example. They are removed on purpose for reasons of saving space.)*

Lastly, for Example 1C, all three velocity POD modes in x-direction are zero. This correctly represents the zero DIRICHLET boundary condition applied to all nodes of the x-faces in that direction. Similar to the uniaxial extension case, Example 1A, mode 2 in y-direction and mode 1 in z-direction nicely show the shrinking and extension respectively, i.e. have velocities in opposite directions.

Online phase - building and simulating the ROM

In order to investigate the reduced model(s) that can be assembled through the POD basis computations described above, several sizes r_u, r_v, $r_p \in \mathbb{N}$ of the reduced spaces and combinations thereof were tested. The two main foci here were (i) on whether a reduced simulation converged or not and (ii) on the resulting error in the displacements, velocities and pressure coefficients computed as specified in Equations (7.1)–(7.2).
Based on observations of the singular value decays c.f. Figures 7.1–7.3. and roughly cutting off each of them at singular values around $1\,\mathrm{e}{-6}$ to $1\,\mathrm{e}{-8}$, values and combinations as listed in Table 7.1 were chosen for a first investigation.

	Ex1A	Ex1B	Ex1C
r_u	$16:2:30$ (8)	$14:2:26$ (7)	$10:2:22$ (7)
r_v	$10:2:26$ (9)	$10:2:24$ (8)	$8:2:24$ (9)
r_p	$6:2:10$ (3)	$5:2:11$ (4)	$3:2:7$ (3)
n_p	6	6	4
# simulations	$216 \times 6 = 1\,296$	$224 \times 6 = 1\,344$	$189 \times 4 = 756$
# non / converged	132 / 1 164	165 / 1 179	26 / 730

Table 7.1: *The ROM simulations performed with Examples 1. For the intervals of the reduced space sizes,* MATLAB *notation is used.*

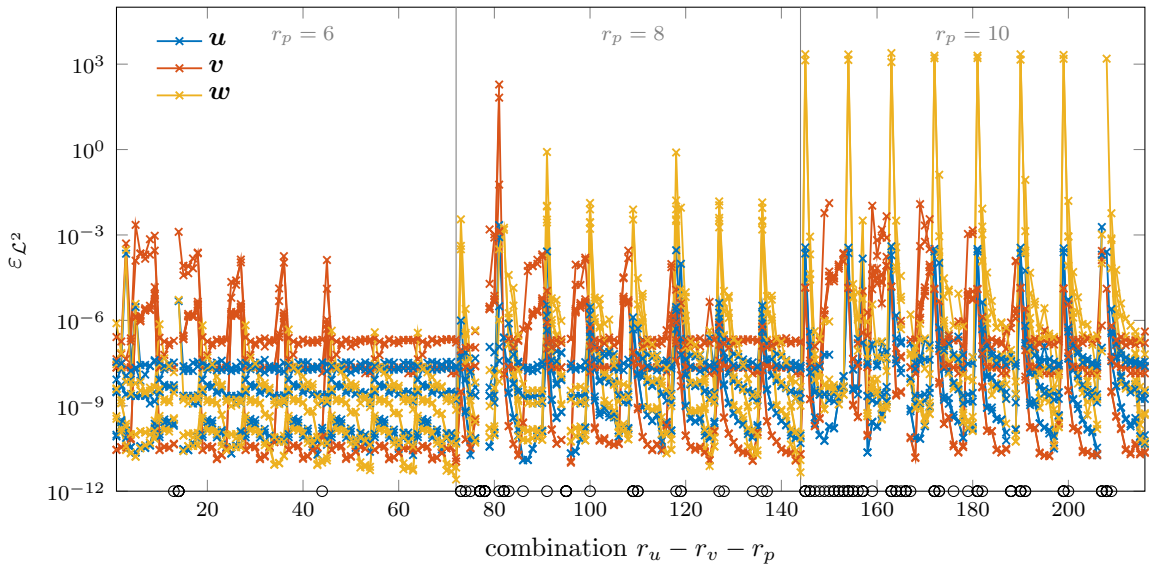


Figure 7.13: *Absolute errors and non-converged simulations of the ROM simulations performed with Example 1A.*

For each example and for each of the $r_u - r_v - r_p$ reduced size combination, a reduced model was built and the simulation performed for each of the n_p parameters from the training data set. Figures 7.13–7.15 show the resulting absolute $\mathcal{L}^2$-errors plotted over the size combinations. This is done separately for $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ dof and for each of the n_p parameters. Additionally, the black circles on the x-axis indicate those combinations, where the ROM did not converge for one or even several parameters. The sequence of the

$r_u - r_v$ combinations in one of the r_p-intervals (which are indicated by the grey vertical lines) is such that first, r_u is kept at its value, while r_v is increased, then, r_u is increased and r_v starts from the beginning of the interval (i.e. $r_u^1 - r_v^1$, $r_u^1 - r_v^2$, ..., $r_u^2 - r_v^1$, ..., $r_u^{\text{end}} - r_v^1$,..., $r_u^{\text{end}} - r_v^{\text{end}}$).

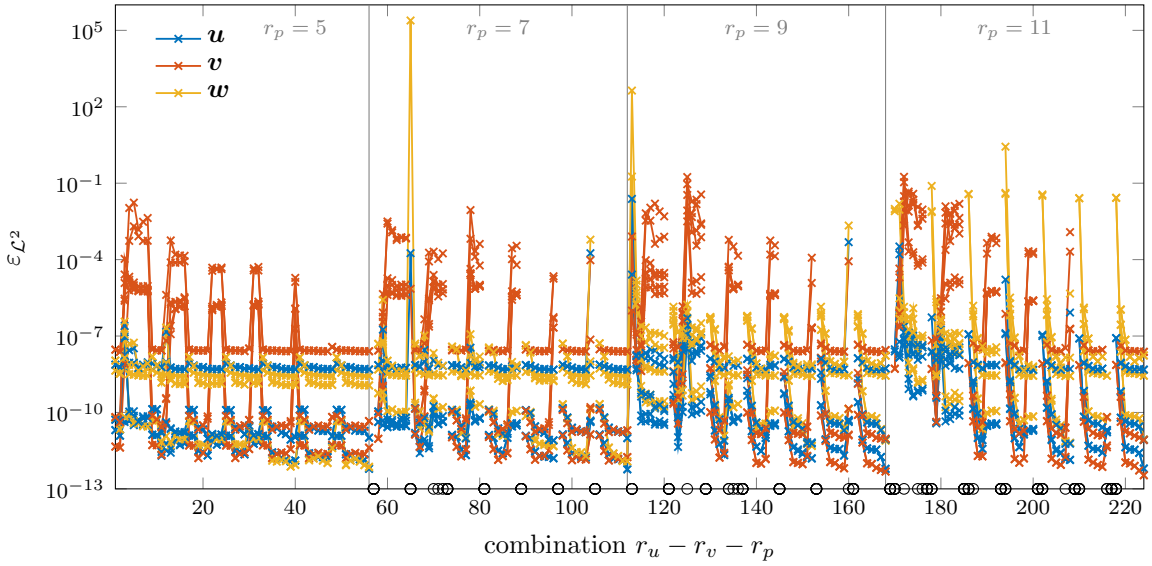


Figure 7.14: *Absolute errors and non-converged simulations of the ROM simulations performed with Example 1B.*

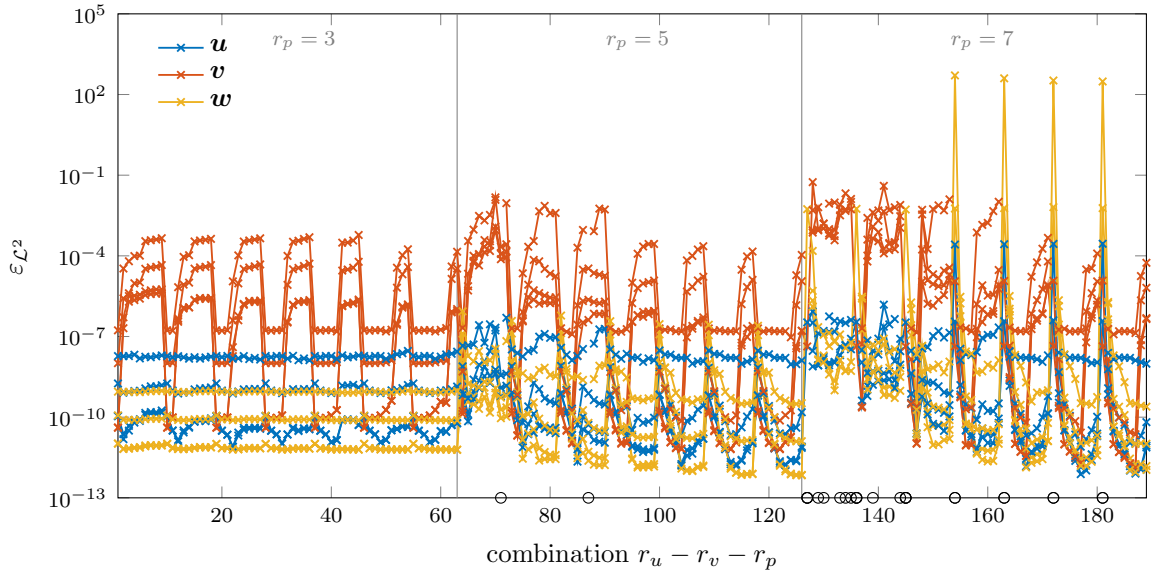


Figure 7.15: *Absolute errors and non-converged simulations of the ROM simulations performed with Example 1C.*

With respect to focus (i), whether a reduced simulation converged or not, one can observe that $5-15\,\%$ of the performed simulations did not converge. The exact numbers are listed in Table 7.1. For simple examples like those chosen here and for such small intervals of

chosen sizes, this performance is very unsatisfactory as one cannot always test several possible sizes and combinations thereof, when one is interested in a stable and fast model. Additionally, huge differences in the absolute $\mathcal{L}^2$-errors (focus (ii)) between the simulated size combinations are present. Merely changing e.g. the reduced size r_v by 2 can have an effect of increasing the error by more than 5 orders of magnitude. Generally, one can observe the tendency that surprisingly less pressure modes yield better convergence and smaller errors, especially in the $\boldsymbol{w}$ dof. The errors in the position dof $\boldsymbol{u}$ are the smallest from the three fields. However, as can be seen not only in Figures 7.13–7.15, but also in Tables 7.2–7.4, where the smallest and largest occurring errors in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ obtained with the ROM simulations are listed, there are large variations in the errors of up to 15 orders of magnitude. Thus, simply taking each of the singular value decays as a measure for the quality and accuracy of the respective ROM is not sufficient in the case of the incompressible skeletal muscle model. Other properties, indicators, and measures need to be found and investigated in order to have an a priori knowledge on what is essential to build a stable ROM.

		$\mu_3 = -1$	$\mu_3 = -0.1$	$\mu_3 = 0.1$	$\mu_3 = 1$	$\mu_3 = 10$	$\mu_3 = 100$
FOM	time	2 435	2 358	2 343	2 357	2 823	18 409
	time	1 928	1 880	1 877	1 933	2 092	15 154
ROM (best)	$\varepsilon^{\boldsymbol{u}}_{\mathcal{L}^2}$	1 e−9	1 e−11	1 e−11	1 e−9	1 e−9	1 e−8
	$\varepsilon^{\boldsymbol{v}}_{\mathcal{L}^2}$	1 e−8	1 e−11	1 e−11	1 e−8	1 e−8	1 e−7
	$\varepsilon^{\boldsymbol{w}}_{\mathcal{L}^2}$	1 e−11	1 e−12	1 e−12	1 e−11	1 e−10	1 e−9
	time	24 610	30 378	31 125	27 527	30 489	32 480
ROM (worst)	$\varepsilon^{\boldsymbol{u}}_{\mathcal{L}^2}$	1 e−3	1 e−4	1 e−4	1 e−3	1 e−4	1 e−4
	$\varepsilon^{\boldsymbol{v}}_{\mathcal{L}^2}$	1 e+2	1 e+1	1 e−4	1 e−2	1 e−2	1 e−3
	$\varepsilon^{\boldsymbol{w}}_{\mathcal{L}^2}$	1 e−1	1 e+3	1 e+3	1 e−1	1 e−2	1 e−4

Table 7.2: *Example 1A: The simulation times [sec] of the FOM simulations (μ_3 in e−3 MPa) compared to those obtained with the ROM simulations, where for the ROM simulations a best case and a worst case value from the size combinations is shown. Additionally, the smallest (best) and largest (worst) occurring errors in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ obtained with the ROM simulations are listed.*

Tables 7.2–7.4 additionally list the solution times for each of the three examples and each of the n_p parameters needed by the FOM simulation and a best case and a worst case of the ROM simulations. The overall time to solve the system of equations is mainly dominated by the necessary number of Jacobian updates, which is related to the condition of the linear system to solve during each NEWTON iteration. A badly conditioned reduced system seems to need more Jacobian updates, and as that evaluation depends on the dimension $\bar{d}$ of the full-order system, some ROM simulations take even longer than the original FOM simulation. Reducing the total system dof to around 0.2 % of the full system (from roughly 20 000 to ±40) yielded in the best cases simulation times around 80 % of the FOM, i.e. an insignificant speedup of 1.25. On the other hand, in the worst cases, the online simulation with a ROM could take 14 times as long as the same simulation with the FOM.

		$\mu_3 = -10$	$\mu_3 = -1$	$\mu_3 = -0.1$	$\mu_3 = 0.1$	$\mu_3 = 1$	$\mu_3 = 10$
FOM	time	2 597	2 160	2 053	2 032	2 176	2 579
ROM (best)	time	2 195	1 919	1 893	1 902	1 924	2 199
	$\varepsilon^{\boldsymbol{u}}_{\mathcal{L}^2}$	1 e−9	1 e−12	1 e−13	1 e−13	1 e−12	1 e−9
	$\varepsilon^{\boldsymbol{v}}_{\mathcal{L}^2}$	1 e−8	1 e−12	1 e−13	1 e−13	1 e−12	1 e−8
	$\varepsilon^{\boldsymbol{w}}_{\mathcal{L}^2}$	1 e−9	1 e−9	1 e−13	1 e−13	1 e−9	1 e−9
ROM (worst)	time	26 987	17 787	25 366	31 598	31 362	25 384
	$\varepsilon^{\boldsymbol{u}}_{\mathcal{L}^2}$	1 e−8	1 e−5	1 e−4	1 e−2	1 e−5	1 e−4
	$\varepsilon^{\boldsymbol{v}}_{\mathcal{L}^2}$	1 e−2	1 e−1	1 e−2	1 e−1	1 e−1	1 e−2
	$\varepsilon^{\boldsymbol{w}}_{\mathcal{L}^2}$	1 e−6	1 e−1	1 e+5	1 e+2	1 e+1	1 e−3

Table 7.3: *Example 1B: The simulation times [sec] of the FOM simulations (μ_3 in $e{-}3$ MPa) compared to those obtained with the ROM simulations, where for the ROM simulations a best case and a worst case value from the size combinations is shown. Additionally, the smallest (best) and largest (worst) occurring errors in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ obtained with the ROM simulations are listed.*

		$\mu_3 = -0.1$	$\mu_3 = 0.1$	$\mu_3 = 1$	$\mu_3 = 10$
FOM	time	2 503	2 506	2 704	9 189
ROM (best)	time	2 028	2 032	2 125	5 640
	$\varepsilon^{\boldsymbol{u}}_{\mathcal{L}^2}$	1 e−12	1 e−13	1 e−12	1 e−9
	$\varepsilon^{\boldsymbol{v}}_{\mathcal{L}^2}$	1 e−12	1 e−12	1 e−11	1 e−8
	$\varepsilon^{\boldsymbol{w}}_{\mathcal{L}^2}$	1 e−13	1 e−13	1 e−11	1 e−10
ROM (worst)	time	5 613	32 969	15 878	32 658
	$\varepsilon^{\boldsymbol{u}}_{\mathcal{L}^2}$	1 e−7	1 e−4	1 e−7	1 e−6
	$\varepsilon^{\boldsymbol{v}}_{\mathcal{L}^2}$	1 e−2	1 e−3	1 e−2	1 e−2
	$\varepsilon^{\boldsymbol{w}}_{\mathcal{L}^2}$	1 e−3	1 e+2	1 e−6	1 e−6

Table 7.4: *Example 1C: The simulation times [sec] of the FOM simulations (μ_3 in $e{-}3$ MPa) compared to those obtained with the ROM simulations, where for the ROM simulations a best case and a worst case value from the size combinations is shown. Additionally, the smallest (best) and largest (worst) occurring errors in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ obtained with the ROM simulations are listed.*

In order to further investigate this problem and to make use of the information extracted from visualising the POD modes, some even smaller reduced models starting with a number of total system dof of $2+1+1=4$ were created and tested in the same way. The chosen values and combinations are listed in Table 7.5.
As before, the resulting absolute $\mathcal{L}^2$-errors in the $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ dof are plotted separately over the size combinations $r_u - r_v - r_p$, together with black circles on the x-axis indicating those locations, where the ROM did not converge (c.f. Figures 7.16–7.18).

	Ex1A	Ex1B	Ex1C
r_u		$\longleftarrow$ 2 : 2 : 10 (5) $\longrightarrow$	
r_v		$\longleftarrow$ 1 : 2 : 9 (5) $\longrightarrow$	
r_p		$\longleftarrow$ 1 : 1 : 3 (3) $\longrightarrow$	
n_p	6	6	4
# simulations	$75 \times 6 = 450$	$75 \times 6 = 450$	$75 \times 4 = 300$
# non / converged	88 / 362	150 / 300	76 / 224

Table 7.5: *The second set of ROM simulations performed with Examples 1. Here, the same reduced sizes were chosen for each of the three examples.*

Unfortunately, also for these smaller combinations, many of the simulations did not converge, even though one would have expected the reduced spaces to be adequate by looking at the first POD modes. Table 7.5 lists the numbers of non-converged and converged simulations. In this chosen size range, even more, namely $20-35\,\%$ of the performed simulations did not converge. Again, it is conspicuous that with less pressure POD modes, more simulations converge. In Examples 1A and 1C this is the case even for all simulations performed with $r_p = 1$. But also among those, the differences in the absolute $\mathcal{L}^2$-errors are huge among different $r_u - r_v$ size combinations. Generally, one can observe that the errors of the best simulations with the smaller ROM are only insignificantly higher (by $1-3$ orders of magnitude) compared to the larger ROM.

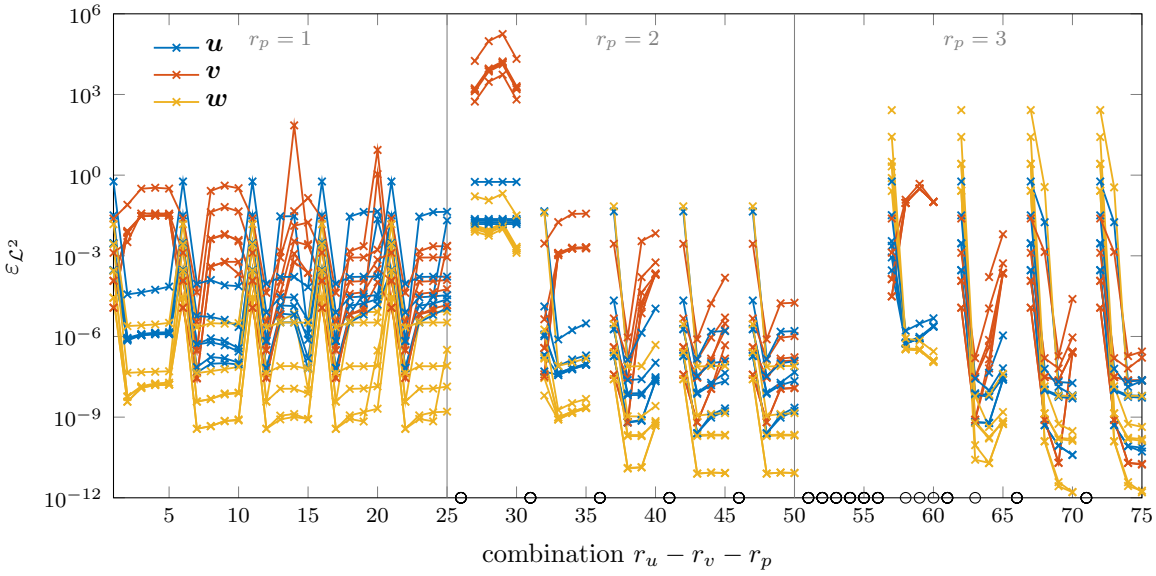


Figure 7.16: *Absolute errors and non-converged simulations of the smaller ROM simulations performed with Example 1A.*

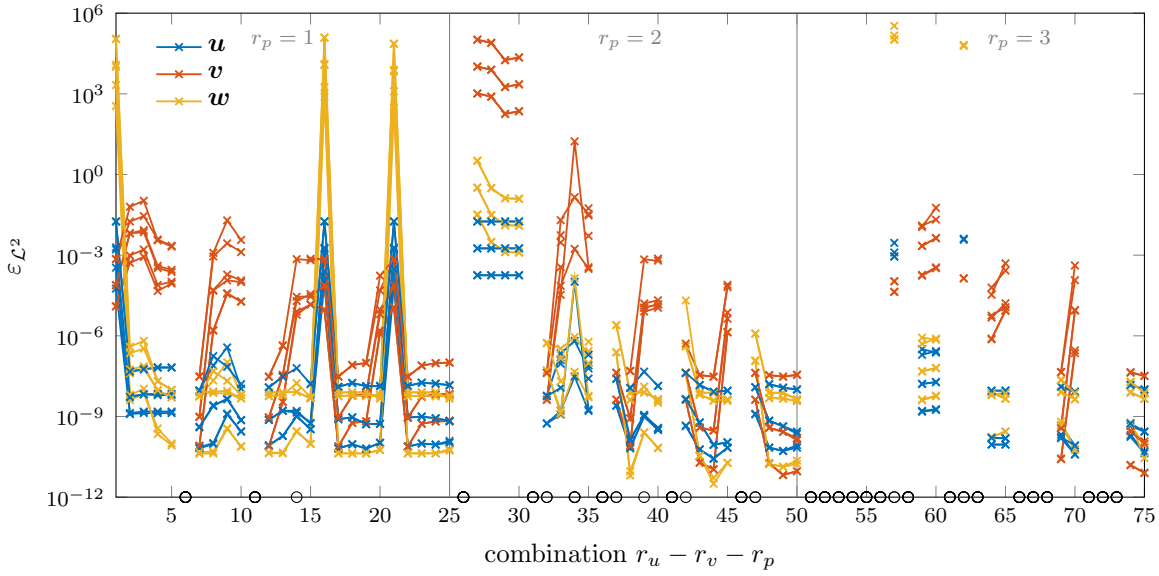


Figure 7.17: *Absolute errors and non-converged simulations of the smaller ROM simulations performed with Example 1B.*

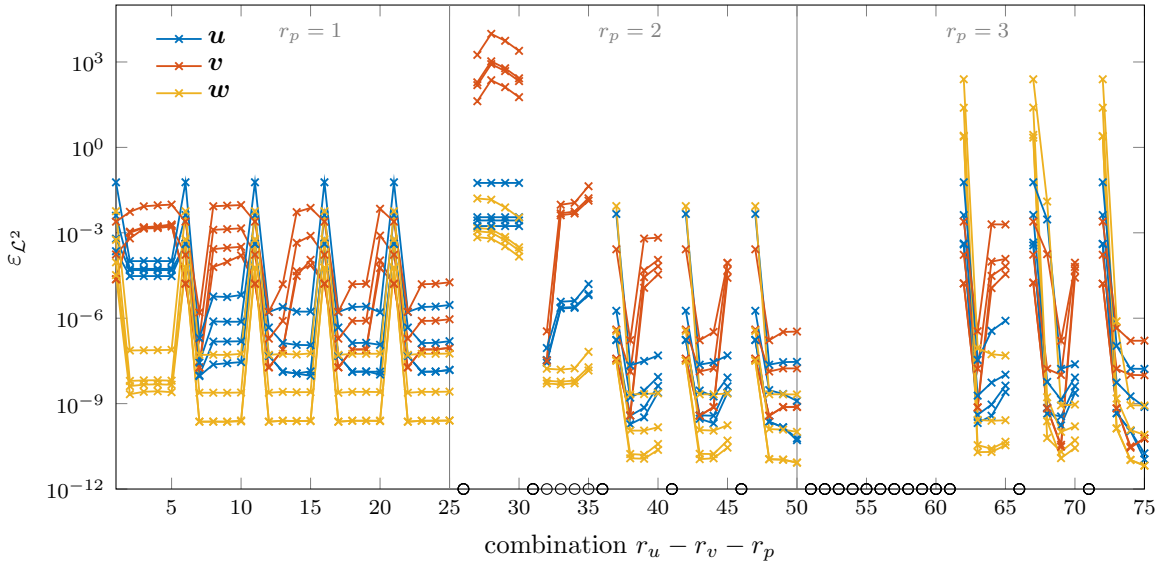


Figure 7.18: *Absolute errors and non-converged simulations of the smaller ROM simulations performed with Example 1C.*

7.1.2 Choice of velocity space with respect to position space

Based on the observations made in Examples 1A, 1B and 1C in the previous section, the choice of the POD bases $\boldsymbol{V}_u$ and $\boldsymbol{V}_v$ shall now be investigated further. Given a reduced space $\mathcal{V}_u^r$, the question of how to choose the reduced space $\mathcal{V}_v^r$, which already arose from the theoretical point of view in Section 5.2, appears now also from the simulation/experimental studies. Without going into investigations on how to best compute the POD basis $\boldsymbol{V}_u$, this section compares the performance of a ROM (a) that is constructed with $\boldsymbol{V}_v \neq \boldsymbol{V}_u$ with a ROM (b), where $\boldsymbol{V}_v = \boldsymbol{V}_u$ is chosen. The investigation was

made using Example 2, which will also be used throughout the rest of this chapter. More specifically, now, only the BC1 case the way it was set up in Section 6.1.2 is used.

Offline phase - computation of training data and POD

Using the FOM with spatial discretisation $\mathrm{d}x = 5\,\mathrm{mm}$, which means 512 elements and $(2 \times 14\,739 + 729 =)\,30\,207$ total dof (c.f. Table 6.7), choosing $n_p = 10$ values for the final magnitude of the applied force in the offline phase, namely

$$\mu_3 \in \{-10, -5, -1, 1, 5, 10, 25, 50, 75, 100\}\ \mathrm{e}-3\,\mathrm{MPa} \tag{7.4}$$

(see also the convergence study in Section 6.3.2), and accounting for the zero DIRICHLET BC, the dimensions of the obtained snapshot matrices are

$$\boldsymbol{S}_u, \boldsymbol{S}_v \in \mathbb{R}^{13\,872 \times 10\,010}\,, \quad \boldsymbol{S}_w \in \mathbb{R}^{729 \times 10\,010}\,. \tag{7.5}$$

Figure 7.19 shows the singular value decays of the subsequently performed $\mathrm{svd}\,(\boldsymbol{S}_\star)$, $\star \in \{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}\}$, on each of the three snapshot matrices.

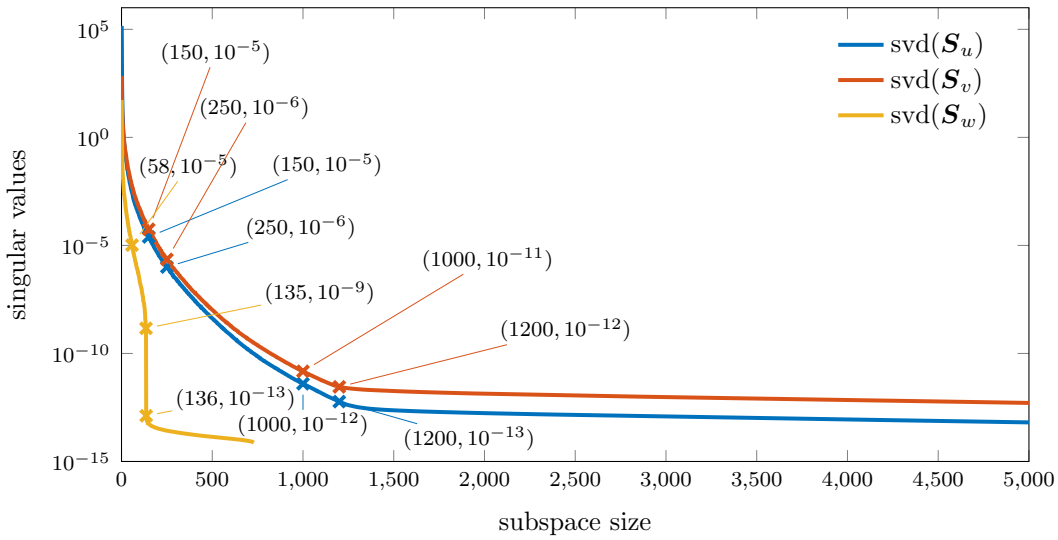


Figure 7.19: *The singular value decays of Example 2, BC1. The plots for $\boldsymbol{u}$ and $\boldsymbol{v}$ are cut at the 5000$^{\text{th}}$ singular value. For each of the plots some characteristic positions are labelled with their respective values.*

For the $\boldsymbol{u}$ and $\boldsymbol{v}$ dof, values of magnitude $1\,\mathrm{e}{-5}$ are reached around the 150$^{\text{th}}$ singular value. After the 1200$^{\text{th}}$ singular value, where values of magnitude $1\,\mathrm{e}{-13}$ for $\boldsymbol{u}$ and magnitude $1\,\mathrm{e}{-12}$ for $\boldsymbol{v}$, are reached, no significant decay occurs. This corresponds to less than 10 % of the original full size. The singular value decay for the $\boldsymbol{w}$ dof shows a steep decay of 4 orders of magnitude between the 135$^{\text{th}}$ and the 136$^{\text{th}}$ singular value from $1\,\mathrm{e}{-9}$ to $1\,\mathrm{e}{-13}$. This corresponds to less than 20 % of the original full size of the pressure space.

Online phase - building and simulating the ROM

In the online phase, two types of reduced models were built:

(a) $\boldsymbol{V}_u := \text{svd}(\boldsymbol{S}_u)$, $\boldsymbol{V}_v := \text{svd}(\boldsymbol{S}_v)$, $\boldsymbol{V}_w := \text{svd}(\boldsymbol{S}_w)$, and

(b) $\boldsymbol{V}_u := \text{svd}(\boldsymbol{S}_u) =: \boldsymbol{V}_v$, $\boldsymbol{V}_w := \text{svd}(\boldsymbol{S}_w)$.

Both ROM types were always built with the same reduced size $r_u = r_v$ for the position and the velocity space. Then, in order to reduce the number of computations, for each ROM three out of the ten training cases, $\mu_3 = -10, 10, 100\,\text{e}{-3}\,\text{MPa}$, were tested for convergence. Table 7.6 summarises the results.

r_p	$r_u = r_v$							
	150		250		1 000		1 200	
	(a)	(b)	(a)	(b)	(a)	(b)	(a)	(b)
136	–	–	–	–	–	3	–	3
140	–	–	–	–	–	3	–	3

Table 7.6: *For both ROM types,* $(4\times2 =)\,8$ *size combinations were tested for the three parameter values. Here, the number of converged simulations is listed, where '–' means, none converged.*

As none of the ROM of type (a) converged, while for type (b) at least both larger ones converged for all the tested parameter values, the conclusion of this investigation is clear: One has to choose the same reduced space for the velocity as for the position field, i.e. use the same POD basis $\boldsymbol{V}_v = \boldsymbol{V}_u$.

7.1.3 Examination of the ratio between the reduced sizes of position and velocity space

The investigations of both previous sections still leaves the open question of how to choose the size r_v with respect to the size r_u. Furthermore, it is not clear, which ratio between the size r_p of the pressure space and the sizes r_u and r_v of the displacement and the velocity spaces needs to be maintained for obtaining a stable ROM. In this section, the focus lies on answering the first question, while the second one is investigated further in Section 7.3.
Here, for all test cases, we compute the simple SVD with $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ of the two snapshot matrices $\boldsymbol{S}_u$ and $\boldsymbol{S}_w$, i.e. $\text{svd}\,(\boldsymbol{S}_\star)$, $\star \in \{\boldsymbol{u}, \boldsymbol{w}\}$, and set $\boldsymbol{V}_u := \text{svd}(\boldsymbol{S}_u) =: \boldsymbol{V}_v$, $\boldsymbol{V}_w := \text{svd}(\boldsymbol{S}_w)$.

Offline phase - computation of training data and POD

To avoid accidental observations, which might only be valid for, or occur in, a specific scenario, the investigation is performed with all the three boundary condition cases of Example 2. As FOM the finite element model with discretisation $\text{d}x = 5\,\text{mm}$, which means 512 elements and $(2 \times 14\,739 + 729 =)\,30\,207$ total dof (c.f. Table 6.7) is used. The n_p values for the final magnitude of the applied force in the offline phase were chosen the same as for the convergence analysis in Section 6.3.2:

BC1: $\mu_3 \in \{-10, -5, -1, 1, 5, 10, 25, 50, 75, 100\}$ e-3 MPa, i.e. $n_p = 10$,

BC2: $\mu_3 \in \{-10, -5, -1, 1, 5, 10, 15, 20\}$ e-3 MPa, i.e. $n_p = 8$,

BC3: $\mu_3 \in \{-10, 0, 10, 20, 30, 40, 50\}$ e-3 MPa, i.e. $n_p = 7$.

Accounting for the zero DIRICHLET boundary conditions, the dimensions of the obtained snapshot matrices are:

BC1: $\boldsymbol{S}_u(, \boldsymbol{S}_v) \in \mathbb{R}^{13\,872 \times 10\,010}$, $\boldsymbol{S}_w \in \mathbb{R}^{729 \times 10\,010}$,

BC2: $\boldsymbol{S}_u(, \boldsymbol{S}_v) \in \mathbb{R}^{13\,872 \times 8\,008}$, $\boldsymbol{S}_w \in \mathbb{R}^{729 \times 8\,008}$,

BC3: $\boldsymbol{S}_u(, \boldsymbol{S}_v) \in \mathbb{R}^{13\,872 \times 7\,007}$, $\boldsymbol{S}_w \in \mathbb{R}^{729 \times 7\,007}$.

Figures 7.19 (see previous section), 7.20 and 7.21 show the singular value decays of the subsequently performed SVD on each of the three snapshot matrices for BC1, BC2 and BC3 respectively. Note that $\boldsymbol{S}_v$ is only included for completeness at this point and that the POD bases obtained from the singular value decomposition of these training data sets are not used for the assembly of the ROM for this investigation.
Comparing the singular value decays of BC1, BC2 and BC3, one can generally observe faster decays in all three dof types, i.e. steeper curves, for BC1 than for BC2 and BC3. For the pressure data, $\boldsymbol{S}_w$, the difference is very pronounced. While for the BC1 case, the kink occurs at the 136$^{\text{th}}$ singular value with a magnitude of 10^{-13}, this magnitude is reached a lot later, around the 500$^{\text{th}}$ singular value, for the BC2 and BC3 case and a kink followed by a gradual decline occurs only at the very end. Considering the larger number of dof in the position data, $\boldsymbol{S}_u$, the difference is not very significant. For BC1 a kink in the values occurs at the 1 200$^{\text{th}}$ singular value with a magnitude of 10^{-13}. BC2 and BC3 are again very similar. A slope of nearly 0 is reached around the 1 350$^{\text{th}}$ singular value with a magnitude of 10^{-11}.

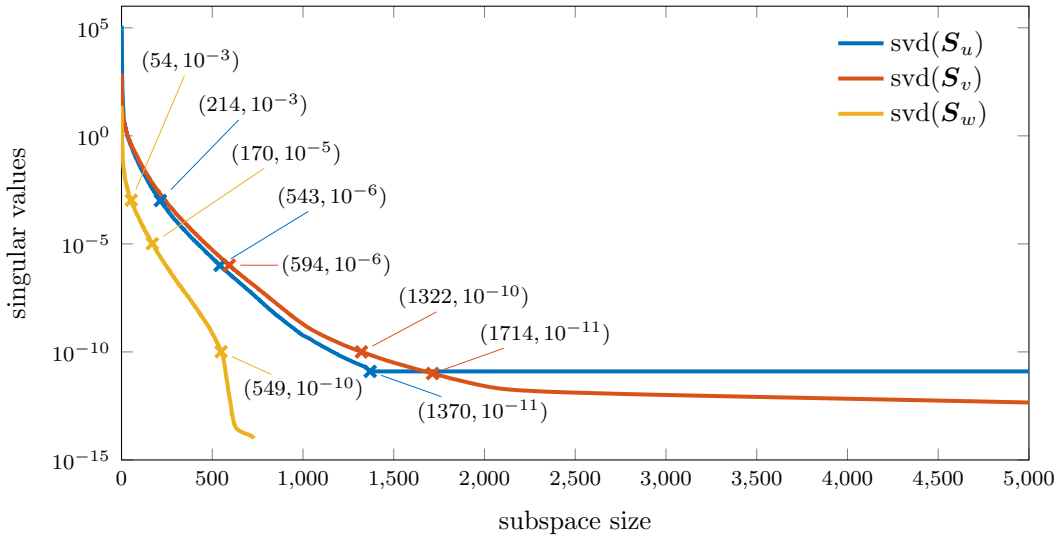


Figure 7.20: *The singular value decays of Example 2, BC2. The plots for $\boldsymbol{u}$ and $\boldsymbol{v}$ are cut at the 5000$^{\text{th}}$ singular value. For each of the plots some characteristic positions are labelled with their respective values.*

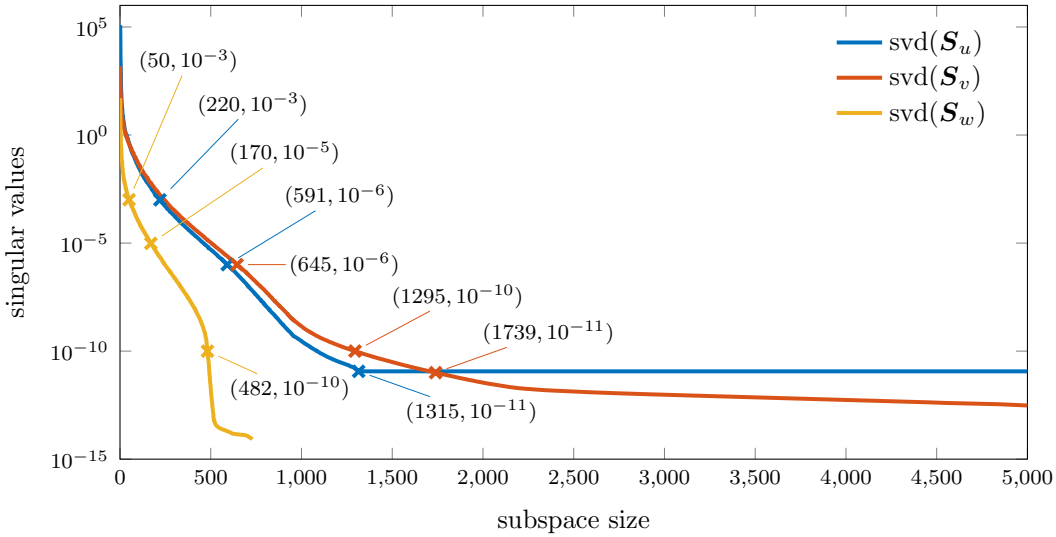


Figure 7.21: *The singular value decays of Example 2, BC3. The plots for $\boldsymbol{u}$ and $\boldsymbol{v}$ are cut at the 5000$^{\text{th}}$ singular value. For each of the plots some characteristic positions are labelled with their respective values.*

Online phase - building and simulating the ROM

Based on the above observations of the singular value decays (c.f. Figures 7.19, 7.20, 7.21), and roughly cutting off each of them at the most obvious kinks (with the exception of svd($\boldsymbol{S}_w$) for BC2 and BC3, where the gradual decline happens very late), values and combinations thereof as listed in Table 7.7 were chosen for a first investigation.

	BC1	BC2	BC3
r_u	1 000 : 10 : 1 200	{1 320, 1 340, 1 520, 1 540}	
r_v	1 000 : 10 : 1 200	1 320 : 20 : 1 540	
r_p	140	170	
online μ_3 values (CPU time FOM)	−10 e−3 (5.6 hrs)	−10 e−3 (7.9 hrs)	20 e−3 (7.4 hrs)
	10 e−3 (5.1 hrs)	1 e−3 (4.2 hrs)	40 e−3 (9.3 hrs)
	100 e−3 (8.8 hrs)	20 e−3 (14.3 hrs)	50 e−3 (10.0 hrs)

Table 7.7: *The ROM simulations performed with Examples 2. The upper part of the table contains the chosen reduced sizes r_u, r_v and r_p. In the lower part, the μ_3 values chosen for the online computations and comparison with the FOM are listed. Here, the CPU time of the FOM simulation for that trajectory is added in parentheses.*

For each of the examples, $n_p = 3$ values (c.f. Table 7.7) out of the training data set for the applied maximum force were chosen for the online ROM simulations. Subsequently, the $\mathcal{L}^2$-errors, $\varepsilon_{\mathcal{L}^2}$, in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ with respect to the FOM were computed. The logarithm of the resulting errors for each simulated case are plotted in a 2-dimensional surface plot and

shown in Figures 7.22–7.24 together with the CPU time in percent of the FOM CPU time for each of the ROM simulations. For the absolute CPU times of the FOM see Table 7.7. In each of the three figures, the columns one to three contain the results for each of the three μ_3 values, while the first three rows show the errors in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ respectively and the last row shows the percentage CPU time.

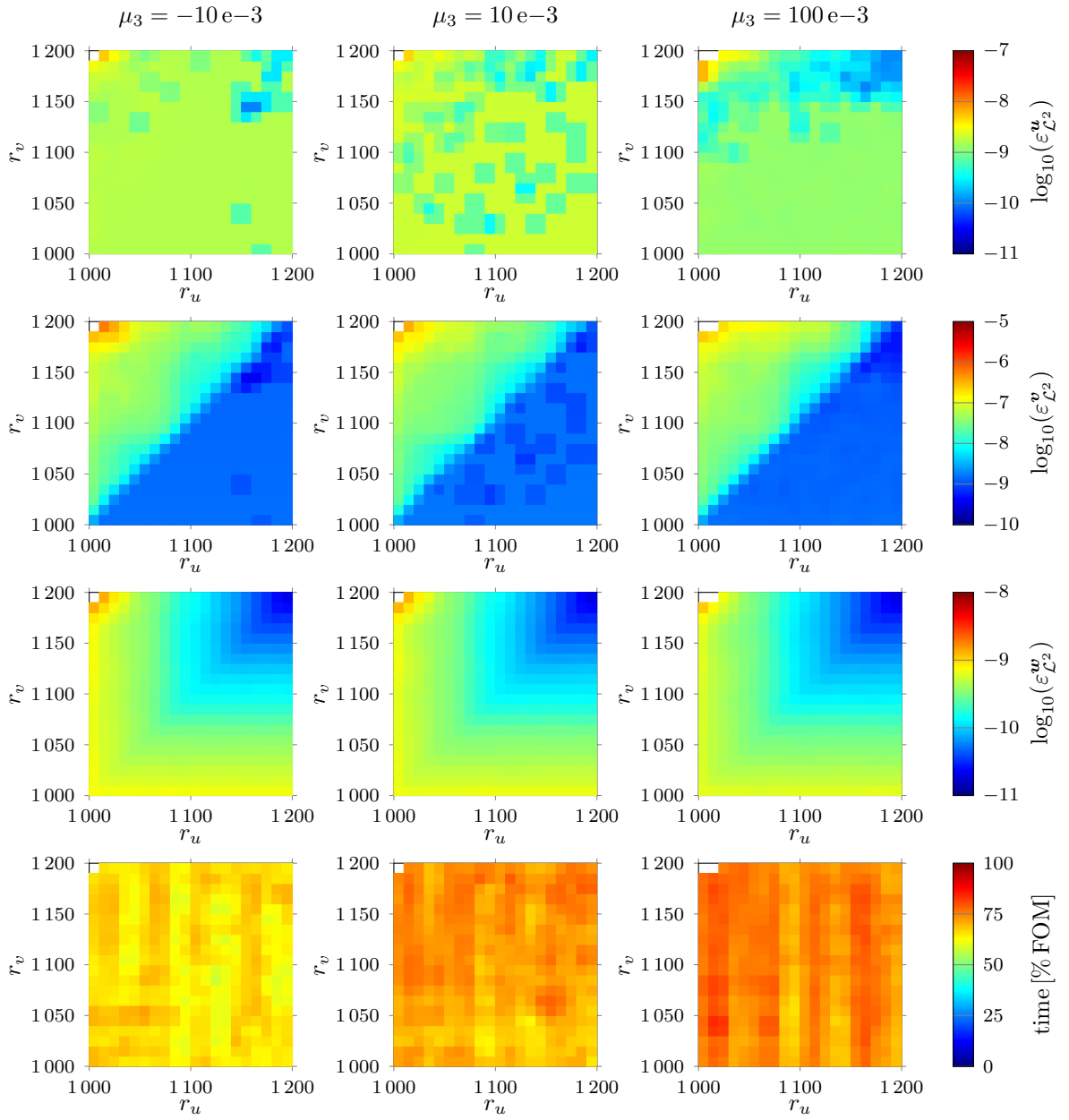


Figure 7.22: *The errors in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ and the percentage CPU time for each of the chosen μ_3 parameter values for the BC1 test of Example 2. Each surface plot contains the result for each of the tested $r_u - r_v$ combination with fixed size $r_p = 140$. Note here that the white squares in the top left corner of each surface plot mean that the simulation(s) performed with that size combination did not converge.*

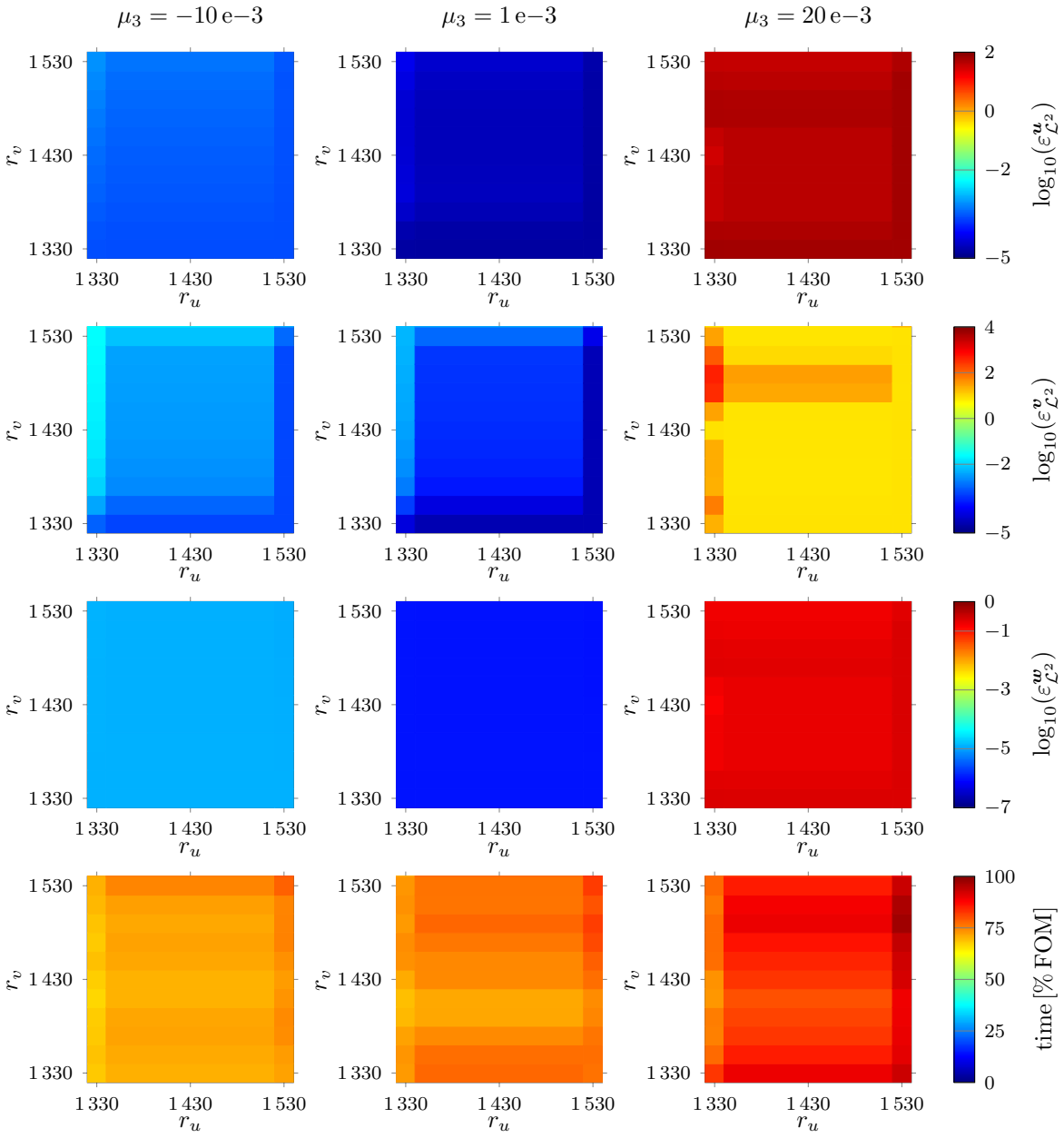


Figure 7.23: *The errors in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ and the percentage CPU time for each of the chosen μ_3 parameter values for the BC2 test of Example 2. Each surface plot contains the result for each of the tested $r_u - r_v$ combination with fixed size $r_p = 170$. Note here that due to time constraints, the twelve r_v sizes were tested with four r_u sizes only, which explains the constant values/colours in the middle part of the squares.*

The most obvious observation, a diagonal separating the cases $r_v > r_u$ and $r_v \leq r_u$, can be made in the $\mathcal{L}^2$-error in $\boldsymbol{v}$ for the BC1 case (c.f. Figure 7.22, row 2). While simulations performed with a ROM, where $r_v \leq r_u$, yield a small $\mathcal{L}^2$-error in the range of 1e$-$9, the $\mathcal{L}^2$-error increases around two to three orders of magnitude, when r_v is increased only slightly about 10, which corresponds to 1 %. Further increase of r_v to 20 % more reduced dof than r_u even results in a non-converging reduced simulation for this case. The same behaviour can be observed in BC2 and BC3. Here it merely doesn't become as obvious

due to the coarser grid of tested r_u sizes. For BC3, the same behaviour is also present in the $\mathcal{L}^2$-error in $\boldsymbol{u}$ (c.f. Figure 7.24, row 1). However, looking at the colour bar range, with only one order of magnitude, it is not as pronounced as for the $\mathcal{L}^2$-error in $\boldsymbol{v}$. Due to large differences in the $\mathcal{L}^2$-error in $\boldsymbol{u}$ among the different μ_3 values it is hard to identify a specific behaviour in the BC2 case.

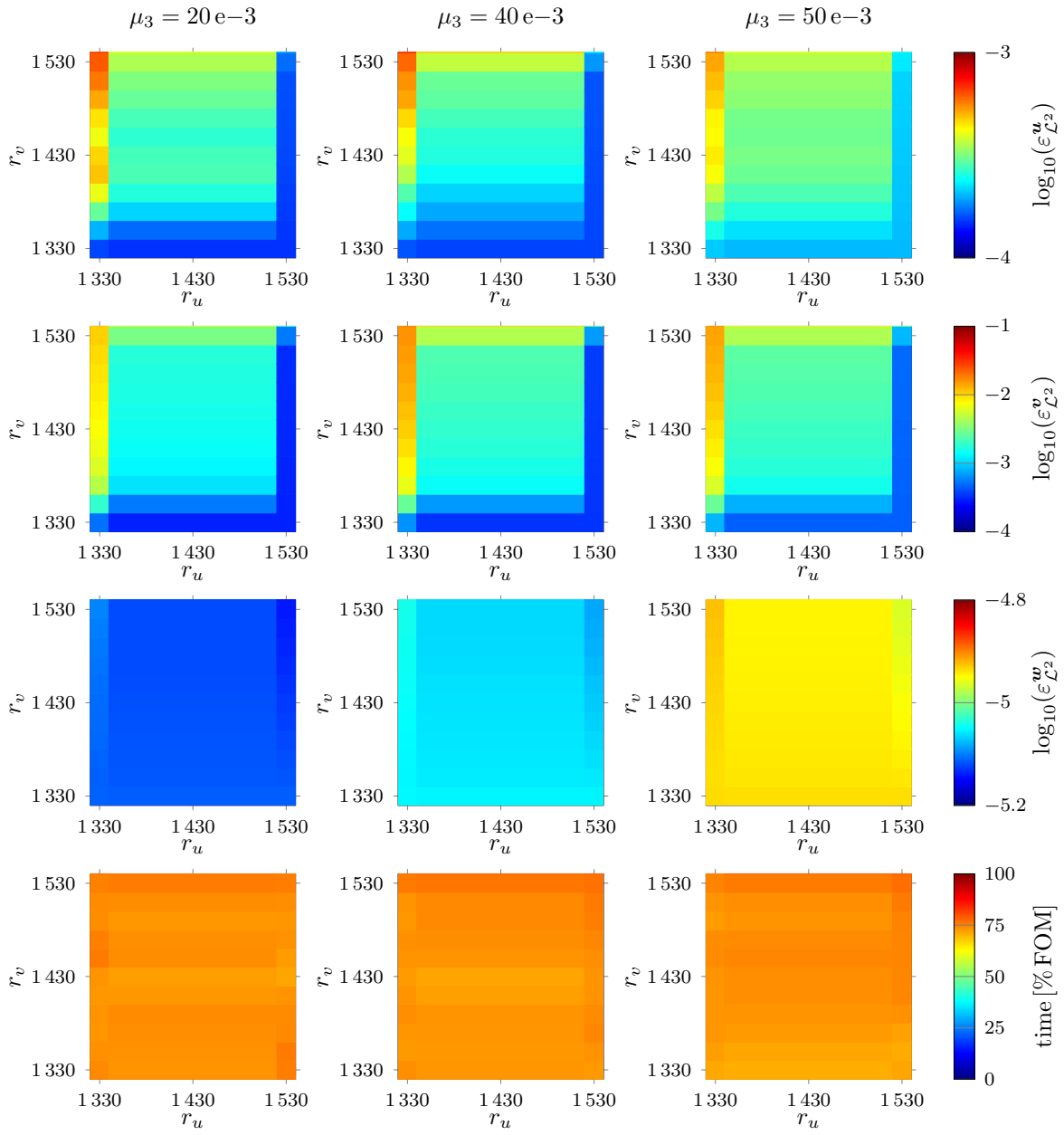


Figure 7.24: *The errors in $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ and the percentage CPU time for each of the chosen μ_3 parameter values for the BC3 test of Example 2. Each surface plot contains the result for each of the tested $r_u - r_v$ combination with fixed size $r_p = 170$. Note here that due to time constraints, the twelve r_v sizes were tested with four r_u sizes only, which explains the constant values/colours in the middle part of the squares.*

In the BC1 case the $\mathcal{L}^2$-error in $\boldsymbol{u}$ seems quite random at first glance. For $\mu_3 = 10\,\mathrm{e}{-3}$, unstructured patches of smaller errors are present and for $\mu_3 = 100\,\mathrm{e}{-3}$ the error even

improves when increasing r_v while keeping r_u constant, i.e. an opposite behaviour can be observed (at least within a certain interval). But for smaller r_u sizes this results in the aforementioned non-converging simulations, when r_v becomes too large. As for the $\mathcal{L}^2$-error in $\boldsymbol{w}$, one can clearly recognise in the BC1 case (c.f. Figure 7.22, row 3) that it decreases with increasing r_u (and r_v) size. This fact could also be inversely interpreted, stating that the $\mathcal{L}^2$-error in $\boldsymbol{w}$ decreases, when r_p decreases, which supports the observation from Section 7.1.1. There it was observed that more simulations converge when less pressure POD modes are accounted for. This rather counter-intuitive behaviour will be further investigated in Section 7.3. Knowing, what one is looking for, the same tendency, although not as strong (note the small colour bar range), can be seen in the surface plots of the $\mathcal{L}^2$-error in $\boldsymbol{w}$ for BC3. Lastly, one can say that no significant difference in the CPU times of the simulated ROM cases can be observed within the chosen size intervals. This is the same for all the three boundary condition cases and a good and important fact as it allows to focus on the error, which is a lot more susceptible to changes in the $r_u - r_v - r_p$ combination. For obvious reasons, there is no significant speedup yet, as the computation of the nonlinear right-hand side and the nonlinear Jacobian (c.f. Section 5.1, Equations (5.7) and (5.8)) still depend on the full dimension $\bar{d}$. The simulation times of the ROM are in the range of $60 - 80\,\%$ of the FOM CPU time.
Generally, as already remarked for the discretisation error in Section 6.3, measuring the error by comparing the position of each node at each time step, might be a relatively strict quality measure, especially for very fine spatial and temporal discretisations. This could be an explanation for the larger errors occurring throughout all size combinations of BC 3 (c.f. Figure 7.24), which is quite a dynamic simulation case with additional oscillations. Small temporal shifts might simply accumulate over the simulated time interval and lead to a large total error, while the overall behaviour might be captured very well. For the purpose of this section, i.e. as a measure to compare the quality of different $r_u - r_v$ size combinations, however, it is suitable. Nevertheless, one might have to think about other error measures, when the absolute value is of interest, e.g. one could define a certain output of interest and just calculate the error specifically for that. The reason behind the even larger errors of magnitude $1\,\mathrm{e}{+2}$ in the BC2, $\mu_3 = 20\,\mathrm{e}{-3}$ case (c.f. Figure 7.23), might lie in the fact that for this simulation, the mesh of the FOM becomes a little distorted towards the maximum applied force and thus an error computation with respect to the FOM could be problematic.
Furthermore, it should be mentioned that the results documented in this section were not the only tested size combinations. Keeping r_p as in the shown examples above and e.g. decreasing the r_u, r_v values to intervals around 250 and 500 for BC1 and BC2/BC3 respectively, most of the ROM built with these reduced sizes did not converge for any of the μ_3 values. Similarly, for the BC3, $\mu_3 = -10\,\mathrm{e}{-3}$ case, only very few ROM built with the above size combinations converged, and thus that parameter value was not suitable for visualising the conclusion of this section.
Certainly, there are many more size combinations that could be tested. However, that approach would be neither an efficient nor a satisfying strategy and one has to come up with additional rules in order to know how to assemble a stable and efficient ROM previously to the online phase. At least, from this section we can already conclude that, for the ratio between the number of reduced displacement dof, r_u, and the number of

reduced velocity dof, r_v,

$$r_v \leq r_u \tag{7.6}$$

has to hold, i.e. an upper bound for r_v is given by the chosen reduced size r_u. As for a lower bound, i.e. how much smaller than r_u one could choose r_v, and whether that is necessary after all for a potential speedup, this question remains open for now.

7.2 Investigating different ways of POD basis computation

The purpose of this section is to investigate the influence of the chosen POD bases $\boldsymbol{V}_u (= \boldsymbol{V}_v)$ and $\boldsymbol{V}_w$ on the efficiency and accuracy of the corresponding ROM. The question, which of the available training data $\boldsymbol{S}_u, \boldsymbol{S}_v$ should be used for the basis construction of the ROM was already raised in Section 5.2 (c.f. Equations (5.20) and (5.21)). Furthermore, the POD bases can be computed optimal in the $\mathcal{L}^2$-norm (i.e. $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$), or optimal in the $\mathcal{H}^1$-norm (i.e. $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$).
Therefore, Section 7.2.1 compares ROM obtained by means of setting

$$(\boldsymbol{V}_v =)\, \boldsymbol{V}_u = \boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}} \boldsymbol{S}_u\right) \in \mathbb{R}^{3N \times r_u} \qquad \text{with } \boldsymbol{\mathcal{A}} \in \{\boldsymbol{H_u}, \boldsymbol{I}_{3N}\}, \tag{7.7}$$

$$\text{and} \quad \boldsymbol{V}_w = \boldsymbol{\mathcal{A}}^{-\frac{1}{2}} \operatorname{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}} \boldsymbol{S}_w\right) \in \mathbb{R}^{N_p \times r_p} \qquad \text{with } \boldsymbol{\mathcal{A}} \in \{\boldsymbol{H}_p, \boldsymbol{I}_{N_p}\}. \tag{7.8}$$

Subsequently, Section 7.2.2 compares ROM, where all POD bases are constructed optimal in the $\mathcal{H}^1$-norm (i.e. $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$), but with different combinations of the available training data, i.e.

$$\boldsymbol{S}_{uv}^{\beta\gamma} := [\beta \boldsymbol{S}_u, \gamma \boldsymbol{S}_v] \in \mathbb{R}^{3N \times 2n} \quad \text{with } \beta, \gamma \in [0, 1]. \tag{7.9}$$

Specifically, we employ four different POD bases for $\boldsymbol{V}_v = \boldsymbol{V}_u$, namely $\boldsymbol{V}_{uv}^{10}$, $\boldsymbol{V}_{uv}^{01}$, $\boldsymbol{V}_{uv}^{11}$ and $\boldsymbol{V}_{uv}^{\bar{\beta}\bar{\gamma}}$ with $\bar{\beta} := \frac{1}{\max \boldsymbol{S}_u}$, $\bar{\gamma} := \frac{1}{\max \boldsymbol{S}_v}$.
As before, the comparison is drawn by calculating the CPU time in percent of the FOM and by computing the errors with respect to the FOM solutions $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$. Additionally, we compare the necessary number of reduced Jacobian updates, which could give a hint on the stability of the ROM. Here, in correlation to the POD bases construction, we calculate the absolute errors $\varepsilon_{\boldsymbol{H}}^{(\star)}$, $(\star) \in \{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}\}$, in the discrete $\mathcal{H}^1$-norm. Let $(\star)^{\mathrm{F}}$ be the FE-solution and $(\star)^{\mathrm{R}}$ the corresponding solution obtained by the reduced model. For each time step t_k, $k = 0, ..., n_t$, we compute

$$\begin{aligned} \varepsilon_{\boldsymbol{H}}^{(\star)}(t_k) &:= \left\| (\star)^{\mathrm{F}}(:, t_k) - (\star)^{\mathrm{R}}(:, t_k) \right\|_{\boldsymbol{H}_{(\star)}} \\ &= \sqrt{\left[(\star)^{\mathrm{F}}(:, t_k) - (\star)^{\mathrm{R}}(:, t_k)\right]^T \boldsymbol{H}_{(\star)} \left[(\star)^{\mathrm{F}}(:, t_k) - (\star)^{\mathrm{R}}(:, t_k)\right]}, \end{aligned} \tag{7.10}$$

and subsequently

$$\varepsilon_{\boldsymbol{H}}^{(\star)} := \frac{1}{n_t+1}\sum_{k=0}^{n_t}\varepsilon_{\boldsymbol{H}}^{(\star)}(t_k)\,, \tag{7.11}$$

i.e. the mean over all time steps. Note that the choice of comparing the absolute and not the relative errors is secondary, as the focus of this section is on the comparison between the different ROM and not on absolute error values. Besides, both errors turned out to be similar and show the same differences between the ROM.
As in Section 7.1.3, Example 2 with BC1, BC2 and BC3 and the discretisation $\mathrm{d}x = 5$ are chosen and the same training data is used. During the online phase, ROM built with 11 different reduced sizes for $r_u = r_v$ are simulated with two chosen parameters $\mu_3 \in \mathcal{P}$ for each of the boundary condition types. These are listed in Table 7.8 and utilised for both, the $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ versus $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ (Section 7.2.1) as well as the $\boldsymbol{V}_{uv}^{10}$, $\boldsymbol{V}_{uv}^{01}$, $\boldsymbol{V}_{uv}^{11}$ versus $\boldsymbol{V}_{uv}^{\bar{\beta}\bar{\gamma}}$ (Section 7.2.2) comparison.

	BC1	BC2	BC3
$r_u = r_v$	1 000 : 20 : 1 200	1 200 : 20 : 1 400	
r_p	140	170	
μ_3	$\{-10, 100\}$ e-3	$\{-10, 1\}$ e-3	$\{20, 50\}$ e-3

Table 7.8: *The size combinations for building the ROM and the performed simulations used for the comparisons in this section.*

7.2.1 The influence of the chosen norm

In order to investigate the effect of the chosen norm for the POD basis computation on the subsequently built ROM, two different POD bases are computed for each of the three boundary condition types. Training data $\boldsymbol{S}_u$ and $\boldsymbol{S}_w$ are utilised to construct the POD bases $\boldsymbol{V}_v = \boldsymbol{V}_u$ and $\boldsymbol{V}_w$.

Offline phase - POD and singular value decay

In the first case (solid lines), we set $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$, i.e. simply perform a singular value decomposition $\mathrm{svd}\left(\boldsymbol{S}_{(\star)}\right)$, $(\star) \in \{\boldsymbol{u}, \boldsymbol{w}\}$. In the second case (dashed lines), we set $\boldsymbol{\mathcal{A}} = \boldsymbol{H}_{(\star)}$, i.e. $\boldsymbol{V}_{(\star)} = \boldsymbol{H}_{(\star)}^{-\frac{1}{2}}\mathrm{svd}\left(\boldsymbol{H}_{(\star)}^{\frac{1}{2}}\boldsymbol{S}_{(\star)}\right)$, $(\star) \in \{\boldsymbol{u}, \boldsymbol{w}\}$. Figures 7.25–7.27 each show the four corresponding singular value decays for BC1, BC2 and BC3.
For all the three boundary condition types, the behaviour of the singular value decays is very similar for the cases $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ and $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$. Especially the change from a steep decay to a slope of (nearly) 0 occurs at similar singular values. This is the case for the SVD on the displacement as well as on the pressure training data. Merely the magnitude of the singular values is insignificantly higher for the $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ case, i.e. for the optimality in the $\mathcal{H}^1$-norm. For the singular values obtained from the BC1 displacement data, this difference is bigger than in all other cases. Here, for the smallest singular values, the difference is three orders of magnitude between the cases $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ and $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$. However, the plateau is reached faster, already at around the 900^{th} singular value.

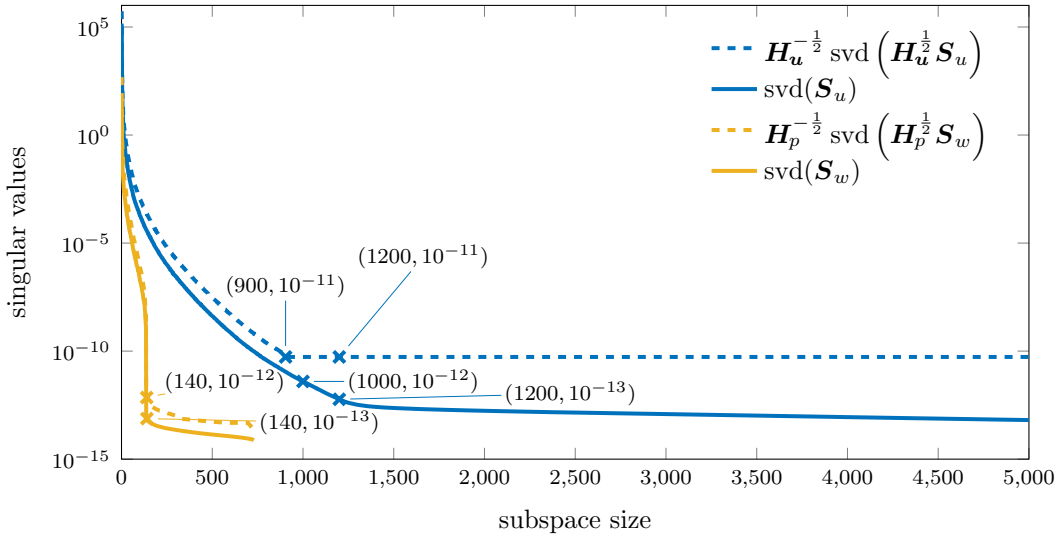


Figure 7.25: *Example 2, BC1:*
The comparison of the singular value decays obtained by means of $\boldsymbol{\mathcal{A}}^{-\frac{1}{2}}\,\mathrm{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}}\boldsymbol{S}_{(\star)}\right)$ *with* $\boldsymbol{\mathcal{A}} = \boldsymbol{H}_{(\star)}$ *(dashed lines) and* $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ *(solid lines) for* $(\star) \in \{\boldsymbol{u}, \boldsymbol{w}\}$. *The plots for* $\boldsymbol{u}$ *are cut at the* 5000^{th} *singular value.*

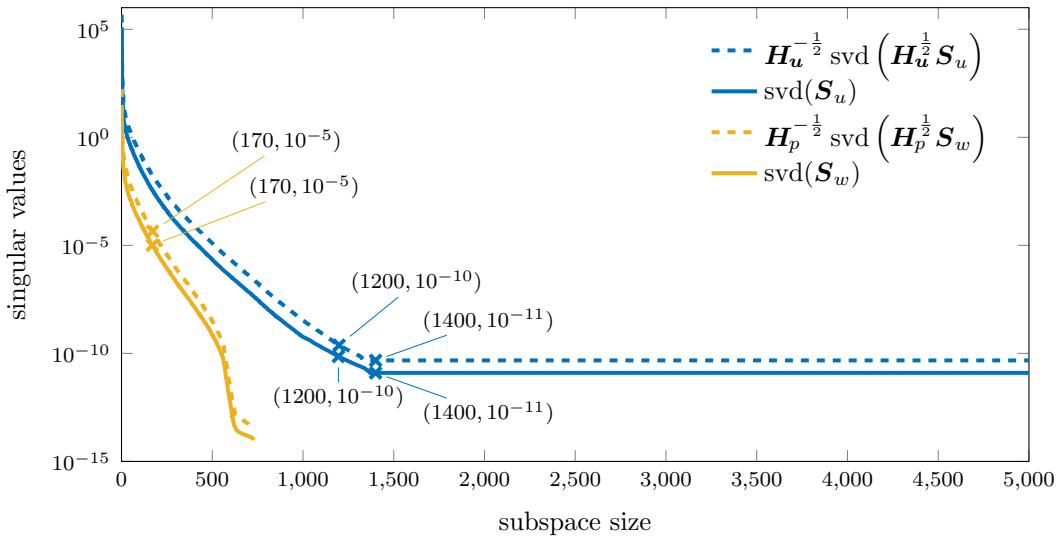


Figure 7.26: *Example 2, BC2:*
The comparison of the singular value decays obtained by means of $\boldsymbol{\mathcal{A}}^{-\frac{1}{2}}\,\mathrm{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}}\boldsymbol{S}_{(\star)}\right)$ *with* $\boldsymbol{\mathcal{A}} = \boldsymbol{H}_{(\star)}$ *(dashed lines) and* $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ *(solid lines) for* $(\star) \in \{\boldsymbol{u}, \boldsymbol{w}\}$. *The plots for* $\boldsymbol{u}$ *are cut at the* 5000^{th} *singular value.*

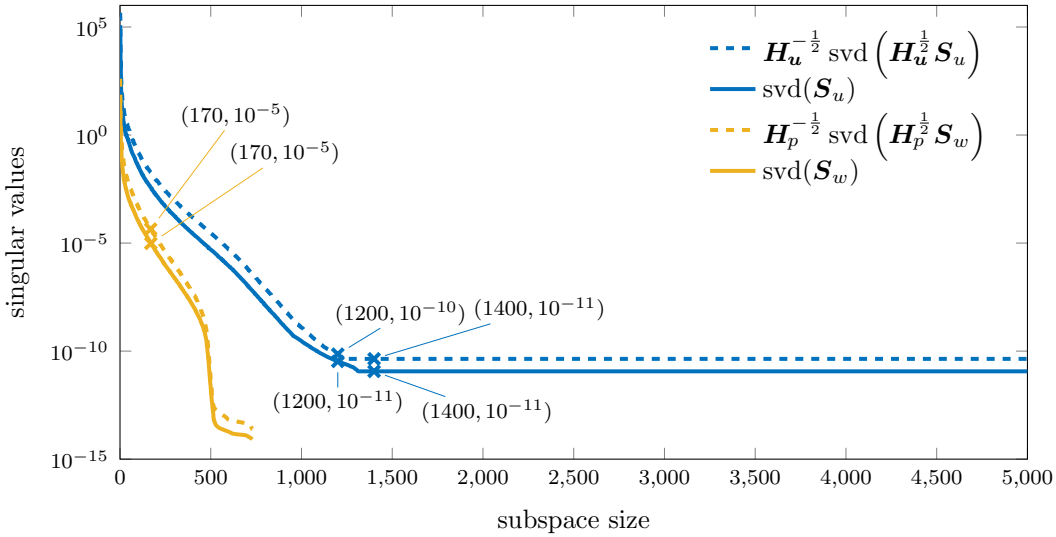


Figure 7.27: *Example 2, BC3:*
The comparison of the singular value decays obtained by means of $\boldsymbol{\mathcal{A}}^{-\frac{1}{2}}\,\mathrm{svd}\left(\boldsymbol{\mathcal{A}}^{\frac{1}{2}}\boldsymbol{S}_{(\star)}\right)$ *with* $\boldsymbol{\mathcal{A}} = \boldsymbol{H}_{(\star)}$ *(dashed lines) and* $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ *(solid lines) for* $(\star) \in \{\boldsymbol{u}, \boldsymbol{w}\}$. *The plots for* $\boldsymbol{u}$ *are cut at the* 5000^{th} *singular value.*

Online phase - building and simulating the ROM

Due to the similar singular value decays and for easy comparability, the reduced sizes $r_u = r_v$ and r_p (c.f. Table 7.8) were chosen equal for both ROM types and the intervals selected based on the results of Section 7.1.3. Then, for each simulation, the CPU time of the ROM simulation in percent of the corresponding FOM simulation is computed. Furthermore, the number of reduced Jacobian updates is extracted. These two values are considered as indicator for the efficiency and the stability of the ROM. The ROM simulation time obviously is dominated by the operations that still depend on the full dimension $\bar{d}$, i.e. on the number of RHS evaluations ($\boldsymbol{f}^r$) and the number of Jacobian ($\boldsymbol{J}\boldsymbol{f}^r$) updates. As it was observed that not only an increase in the dynamics of a simulation, but also an increase in the condition number of the (reduced) Jacobian resulted in an increasing number of Jacobian updates, this value is not only an indicator for the efficiency but also for the stability of the ROM. Aside from that, the reduced simulations also seemed to diverge when too many Jacobian updates were necessary. Thus, the maximum number of allowed reduced Jacobian updates was set to twice the number of necessary Jacobian updates in the FOM case. However, for the chosen reduced sizes intervals and combinations, every performed ROM simulation at least converged, which is already a considerable improvement on the previously performed tests.
As a measure for the accuracy of the different ROM, naturally, the errors of the ROM solution with respect to the FE solution were computed, again separately for the $\boldsymbol{u}$-, $\boldsymbol{v}$- and $\boldsymbol{w}$-dof. For the investigation at hand, the errors were calculated in both norms, the discrete $\mathcal{L}^2$-norm (c.f. Equations (7.1) and (7.2)) and the discrete $\mathcal{H}^1$-norm (c.f. Equations (7.10) and (7.11)). This decision was taken, as here the two types of POD bases were computed optimally in each norm and hence in order not to overreach one case or the other.

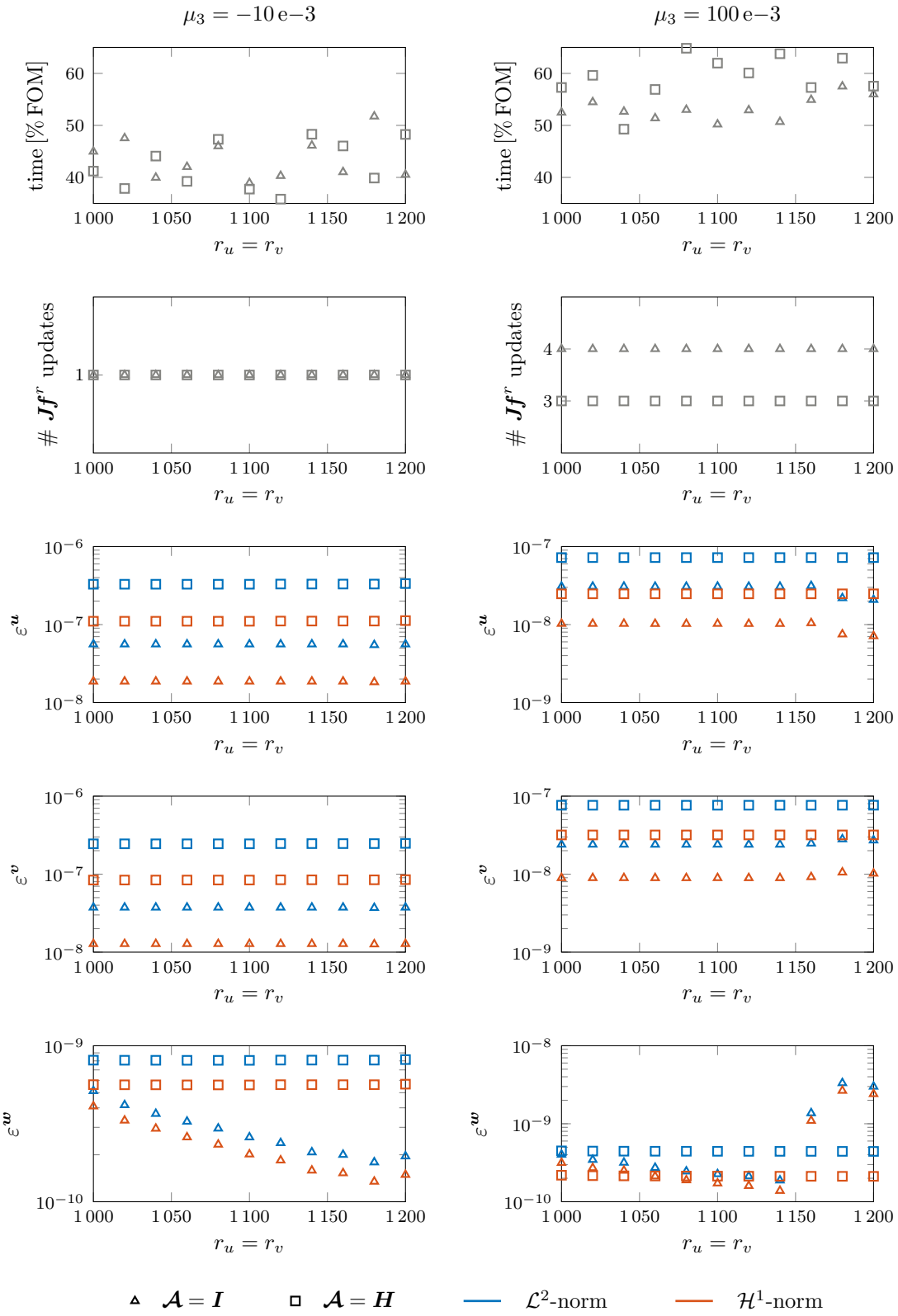


Figure 7.28: *Example 2, BC1 - influence of utilised norm:*
Each of the two columns contains the ROM simulation time in percent of the FOM simulation time, the necessary number of Jacobian updates and the errors in the $\boldsymbol{u}$-, $\boldsymbol{v}$- and $\boldsymbol{w}$-dof for each simulated reduced size for one of the two chosen parameter values μ_3 from the training data set $\mathcal{P}$.

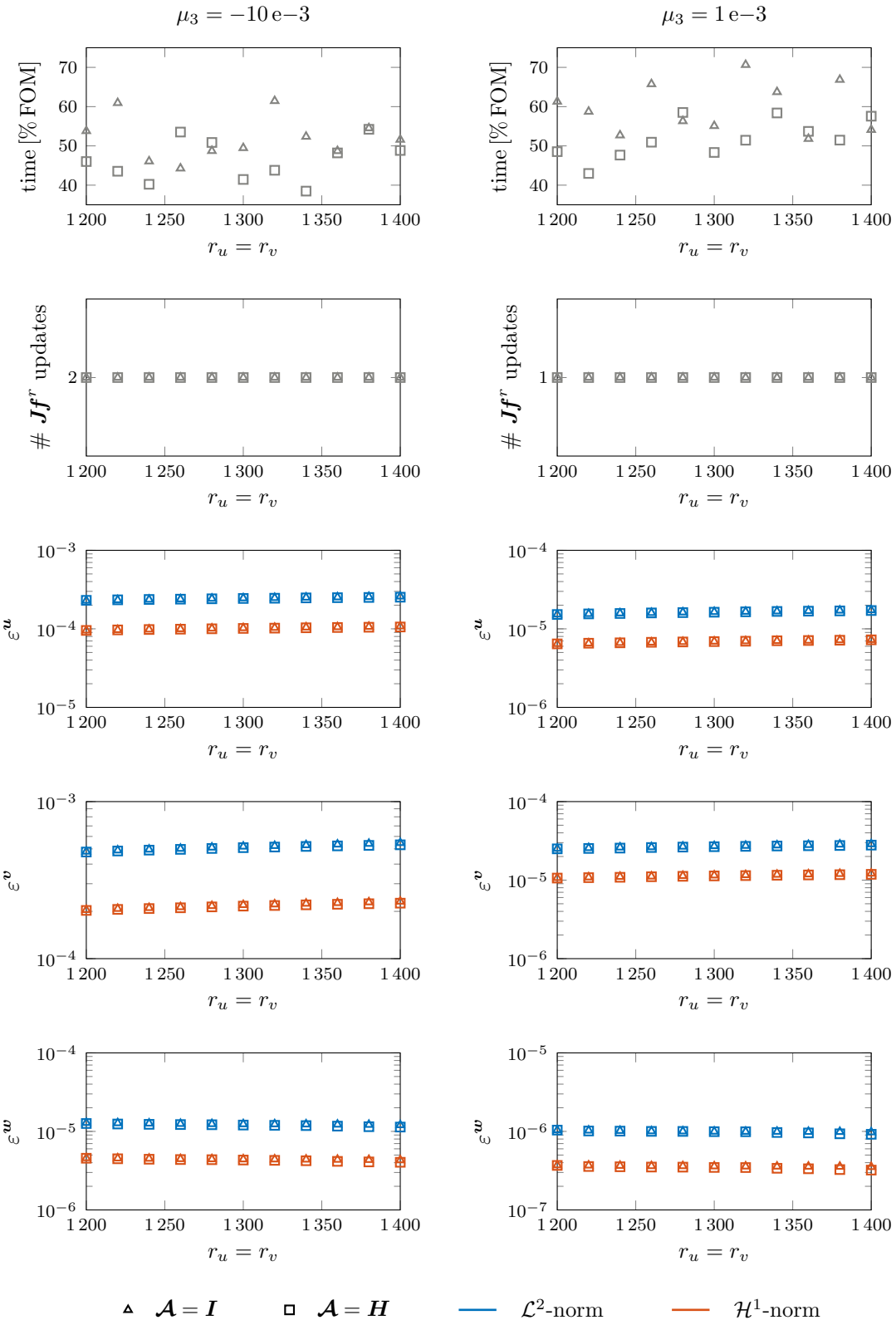


Figure 7.29: *Example 2, BC2 - influence of utilised norm:*
Each of the two columns contains the ROM simulation time in percent of the FOM simulation time, the necessary number of Jacobian updates and the errors in the $\boldsymbol{u}$-, $\boldsymbol{v}$- and $\boldsymbol{w}$-dof for each simulated reduced size for one of the two chosen parameter values μ_3 from the training data set $\mathcal{P}$.

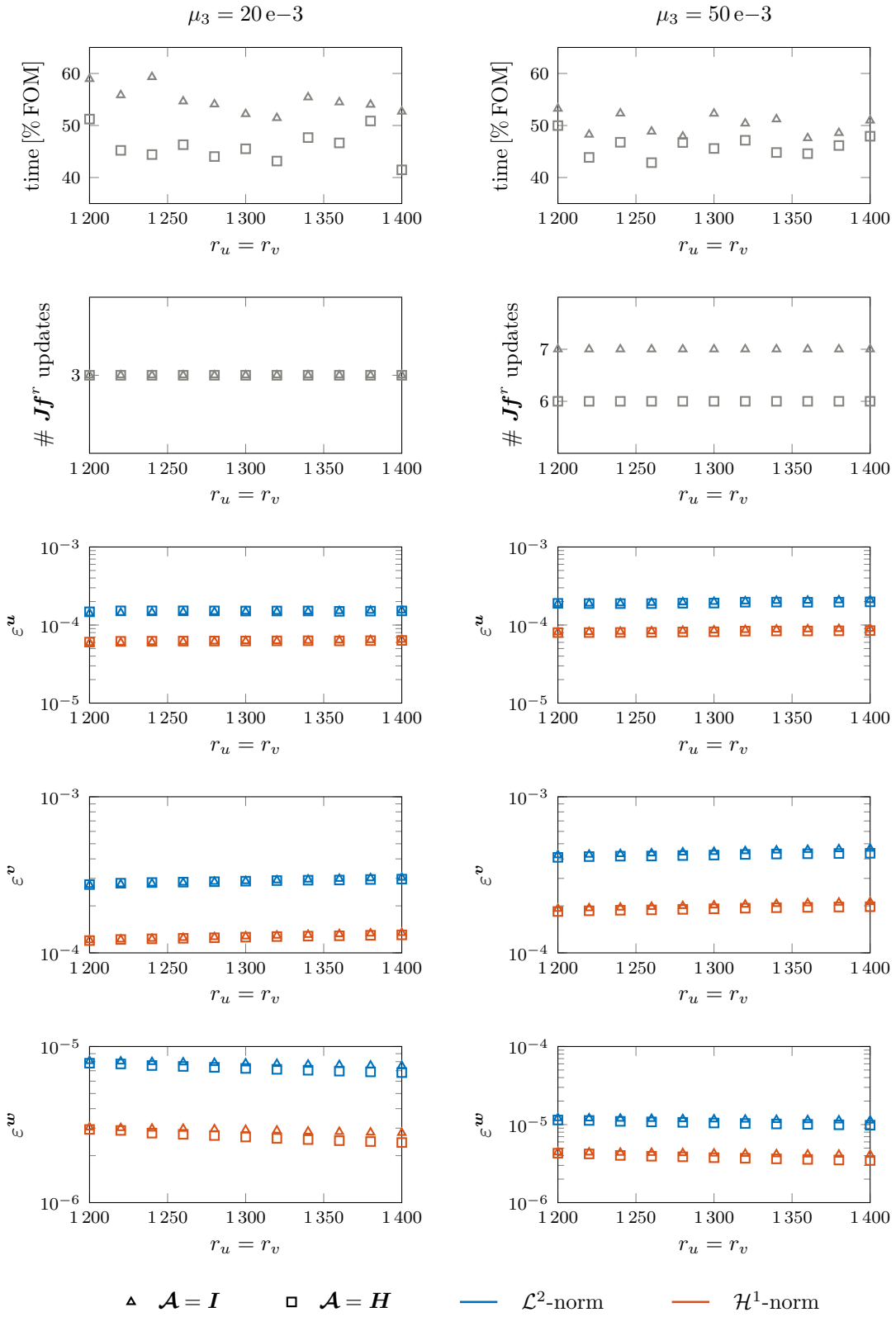


Figure 7.30: *Example 2, BC3 - influence of utilised norm:*
Each of the two columns contains the ROM simulation time in percent of the FOM simulation time, the necessary number of Jacobian updates and the errors in the $\boldsymbol{u}$-, $\boldsymbol{v}$- and $\boldsymbol{w}$-dof for each simulated reduced size for one of the two chosen parameter values μ_3 from the training data set $\mathcal{P}$.

The results for each of the three boundary conditions are displayed in Figures 7.28–7.30. Starting by assessing the accuracy of the two ROM types, one can generally say that there is no significant difference between the errors in the $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ and the $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ case. For BC2 and BC3, all three errors, $\varepsilon^{\boldsymbol{u}}$, $\varepsilon^{\boldsymbol{v}}$ and $\varepsilon^{\boldsymbol{w}}$, lie in the range of 10^{-6} to 10^{-3}. This is the case for the discrete $\mathcal{L}^2$-norm as well as the discrete $\mathcal{H}^1$-norm. Only if one has a very close look, one can observe that the errors in the $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ ROM are slightly better. The errors for the BC1 simulations are three orders of magnitude smaller, they lie in the range of 10^{-10} to 10^{-6}. Here, the difference between both ROM types seems more pronounced. It is more evident that the $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ ROM perform slightly better. However, as the range of the errors is three orders of magnitude smaller, this is merely a misleading observation. Thus, as the errors for both ROM types are of the same magnitude, the difference in the accuracy is negligible. While for BC2 and BC3 all the errors do not vary significantly between the different reduced sizes $r_u = r_v$, there are two noticeable features in the pressure error plots of the $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ ROM in the BC1 simulations. In the $\mu_3 = 10\,\mathrm{e}{-3}$ case, the error $\varepsilon^{\boldsymbol{w}}$ decreases uniformly with increasing size $r_u = r_v$. This effect has already been observed previously in Section 7.1.3 (c.f. e.g. Figure 7.22). An oppositional effect occurs in the more dynamic $\mu_3 = 100\,\mathrm{e}{-3}$ case. Here the error $\varepsilon^{\boldsymbol{w}}$ for the $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ type ROM abruptly jumps up one order of magnitude, between the reduced sizes $r_u = r_v = 1140$ and $r_u = r_v = 1160$ and increases even further afterwards. This observation could mean that ROM built with POD bases computed by means of setting $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ are less susceptible to (small) changes in the chosen reduced sizes and size ratios and thus, from this (stability) point of view, are preferable to the $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ type ROM.
Interpreting the time plots and the plots showing the number of reduced Jacobian updates, we can draw a conclusion regarding the efficiency and stability of the two ROM types. In four out of the six cases, the number of reduced Jacobian updates is equal, while in two cases, the ROM built by means of $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ needs one update less. These two cases represent more dynamic simulations, where stability seems to be more of an issue. Hence, this observation speaks in favour of the $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ ROM type. Furthermore, this factor should also be reflected in the CPU time needed for the reduced simulation. However, there the plots show some contradictory results. While for the BC2 and BC3 cases, most (39 out of 44) simulations performed with the $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ type ROM are faster than the corresponding $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ type ROM simulation, this is the opposite for the BC1 simulations, where only in 7 out of 22 simulations the $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ type ROM is faster than the corresponding $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ type ROM simulation. Nevertheless, considering all the simulations, this adds up to 46 out of 66 simulations, where the $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$ type ROM performs better than the $\boldsymbol{\mathcal{A}} = \boldsymbol{I}$ type ROM from the speedup point of view. Additionally, in each of the six simulated cases (three boundary condition types times two chosen parameters), the fastest result was achieved with a ROM built from a POD basis optimal in the $\mathcal{H}^1$-norm. Taking all these observations and interpretations into account, one can conclude that it is advisable to compute the POD bases optimal in the $\mathcal{H}^1$-norm, which is also employed from now on in all the following investigations.

7.2.2 The influence of the chosen training data

Based on the results of the previous section, every POD basis in this section is constructed optimally in the $\mathcal{H}^1$-norm. The pressure POD basis $\boldsymbol{V}_w = \boldsymbol{H}_p^{-\frac{1}{2}} \mathrm{svd}\left(\boldsymbol{H}_p^{\frac{1}{2}} \boldsymbol{S}_w\right)$ is combined

with four different bases $\boldsymbol{V}_u = \boldsymbol{V}_v$ for the position and velocity dof. The aim is to investigate the effect of the chosen training data on the performance of the ROM.

Offline phase - POD and singular value decay

At the beginning of Section 7.2, the chosen combinations of the available training data $\boldsymbol{S}_u$ and $\boldsymbol{S}_v$ for this investigation are specified. Again, we analyse the effects for all the three boundary condition cases. Figures 7.31–7.33 each show the four different singular value decays resulting from the corresponding training data combination together with the singular value decay for the pressure.

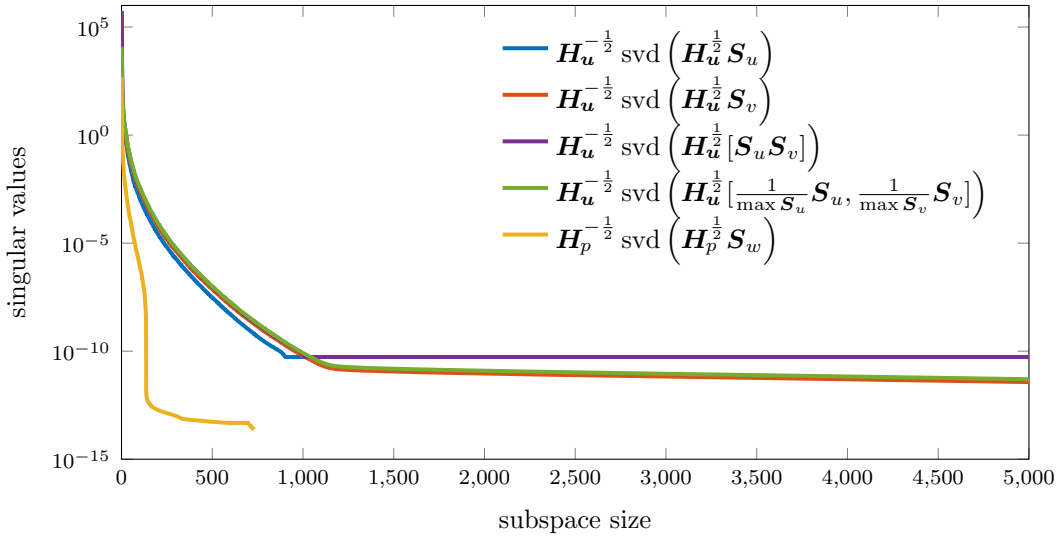


Figure 7.31: *Example 2, BC1: The singular value decays obtained by means of* $\boldsymbol{H}^{-\frac{1}{2}}\,svd\left(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{S}\right)$, *with* $\boldsymbol{S} \in \left\{\boldsymbol{S}_u, \boldsymbol{S}_v, [\boldsymbol{S}_u, \boldsymbol{S}_v], [\frac{1}{\max \boldsymbol{S}_u}\boldsymbol{S}_u, \frac{1}{\max \boldsymbol{S}_v}\boldsymbol{S}_v], \boldsymbol{S}_w\right\}$.

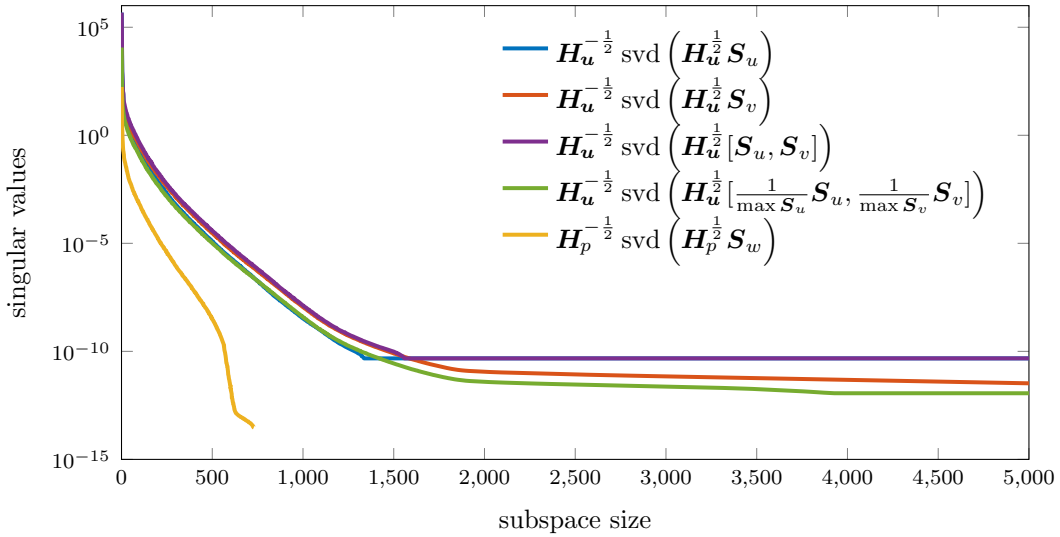


Figure 7.32: *Example 2, BC2: The singular value decays obtained by means of* $\boldsymbol{H}^{-\frac{1}{2}}\,svd\left(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{S}\right)$, *with* $\boldsymbol{S} \in \left\{\boldsymbol{S}_u, \boldsymbol{S}_v, [\boldsymbol{S}_u, \boldsymbol{S}_v], [\frac{1}{\max \boldsymbol{S}_u}\boldsymbol{S}_u, \frac{1}{\max \boldsymbol{S}_v}\boldsymbol{S}_v], \boldsymbol{S}_w\right\}$.

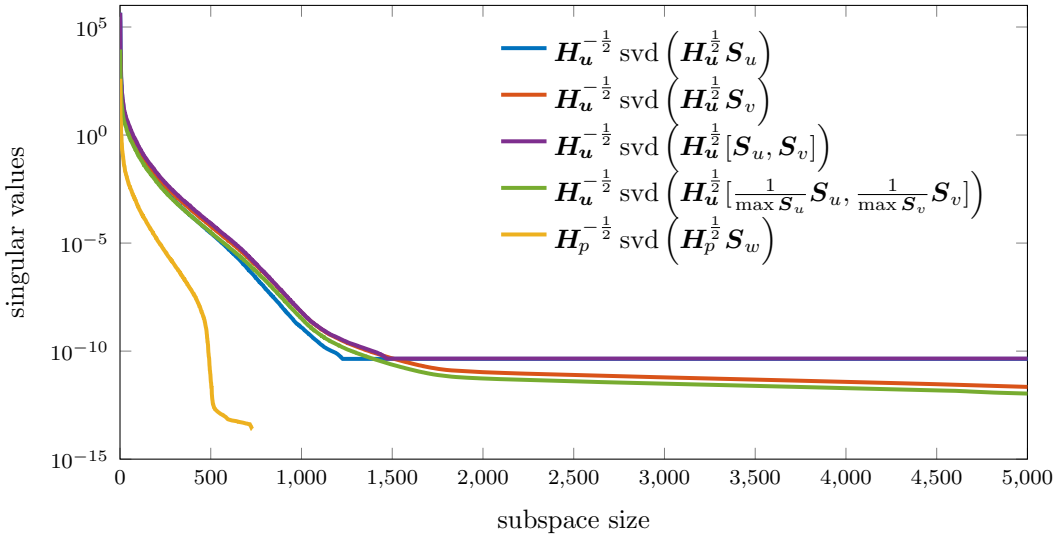


Figure 7.33: *Example 2, BC3: The singular value decays obtained by means of* $\boldsymbol{H}^{-\frac{1}{2}}$ *svd* $\left(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{S}\right)$*, with* $\boldsymbol{S} \in \left\{\boldsymbol{S}_u, \boldsymbol{S}_v, [\boldsymbol{S}_u, \boldsymbol{S}_v], [\frac{1}{\max \boldsymbol{S}_u}\boldsymbol{S}_u, \frac{1}{\max \boldsymbol{S}_v}\boldsymbol{S}_v], \boldsymbol{S}_w\right\}$.

For all the three boundary conditions, no significant differences exist among the respective singular value decays. The decays for $\boldsymbol{S} = \boldsymbol{S}_v$ and $\boldsymbol{S} = \left[\frac{1}{\max \boldsymbol{S}_u}\boldsymbol{S}_u, \frac{1}{\max \boldsymbol{S}_v}\boldsymbol{S}_v\right]$ are a little slower than the ones for $\boldsymbol{S} = \boldsymbol{S}_u$ and $\boldsymbol{S} = [\boldsymbol{S}_u, \boldsymbol{S}_v]$, but this effect is really insignificant.

Online phase - building and simulating the ROM

Based on the singular value decays, we again chose the reduced sizes $r_u = r_v$ and r_p as before (c.f. Table 7.8) for all four ROM types. Equivalent to the previous section, for each example and each parameter value μ_3, the CPU time of the ROM with respect to the corresponding FOM, the number of Jacobian updates and the discrete $\mathcal{H}^1$-errors $\varepsilon_{\boldsymbol{H}}^{(\star)}$, $(\star) \in \{\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}\}$, are extracted or computed for the 11 different reduced sizes $r_u = r_v$. The results are shown in Figures 7.34–7.36.
For the errors in $\boldsymbol{u}$ and $\boldsymbol{v}$ it makes no noteworthy difference, with which combination of the available training data the POD basis of the corresponding ROM was computed. This is the case for all the three boundary condition types. The errors range from 10^{-3} to 10^{-8}, depending on the boundary condition type and the parameter value. The accuracy in the pressure dof, i.e. the error $\varepsilon_{\boldsymbol{H}}^{w}$, is slightly more influenced by this choice. In the case $\boldsymbol{V}_{uv}^{10}$, the error is smaller than in the $\boldsymbol{V}_{uv}^{11}$ and $\boldsymbol{V}_{uv}^{\bar{\beta}\bar{\gamma}}$ cases for all reduced sizes. For BC2 and BC3, they still lie in the same order of magnitude and this marginal difference can be neglected. For BC1 however, especially the $\boldsymbol{V}_{uv}^{\bar{\beta}\bar{\gamma}}$ case performs worse. Here, the error for the first half of the chosen reduced sizes $r_u = r_v$ is one order of magnitude larger than in the $\boldsymbol{V}_{uv}^{10}$ ROM. The errors of the $\boldsymbol{V}_{uv}^{11}$ ROM lie in between. This result is rather surprising as one would expect a more accurate ROM the more information it contains. However, it seems like adding the information of the velocity training data $\boldsymbol{S}_v$ does not improve the ROM at all and maybe it even yields a worse performance.

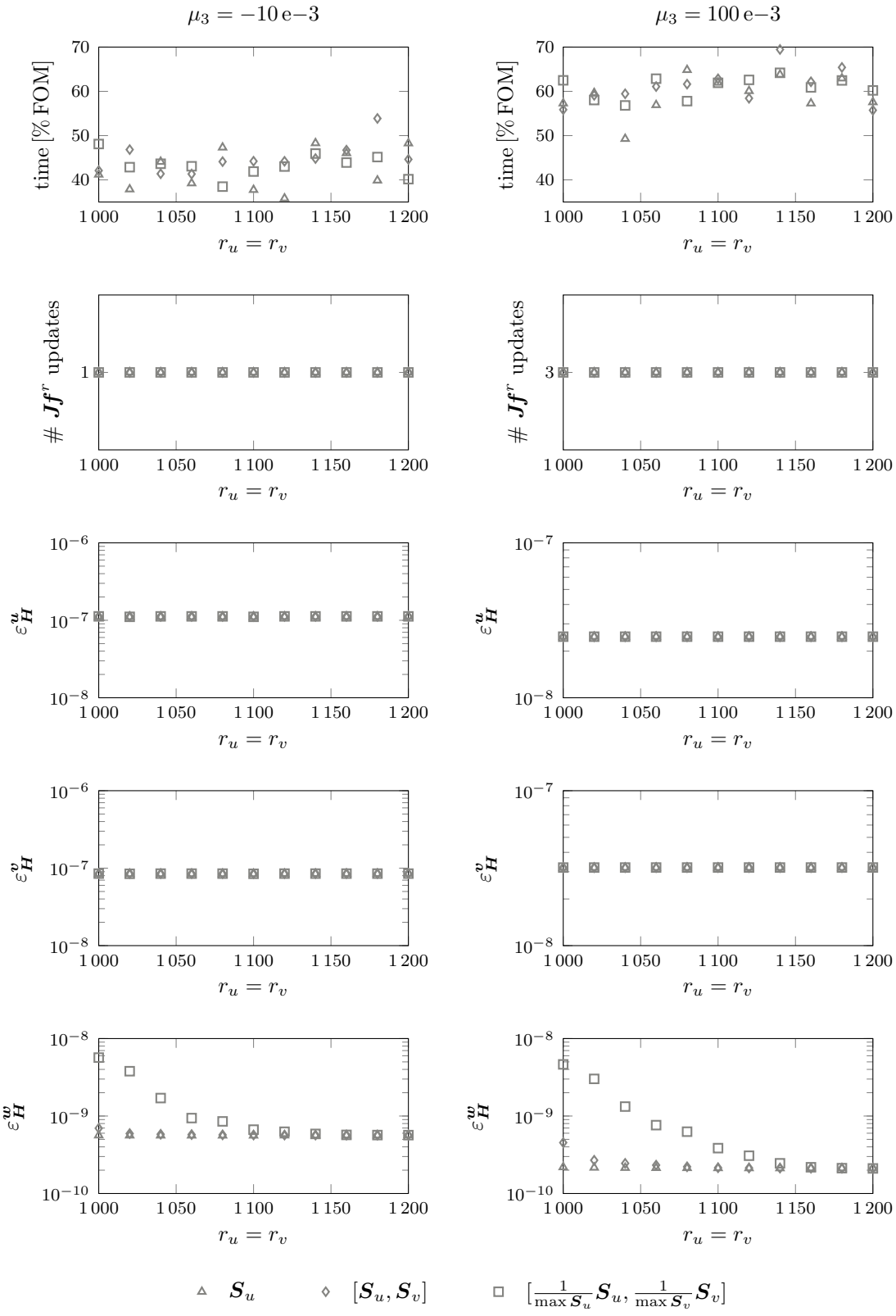


Figure 7.34: *Example 2, BC1 - influence of utilised training data:*
Each of the two columns contains the ROM simulation time in percent of the FOM simulation time, the necessary number of Jacobian updates and the $\mathcal{H}^1$-errors in the $\boldsymbol{u}$-, $\boldsymbol{v}$- and $\boldsymbol{w}$-dof for each simulated reduced size for one of the two chosen parameter values μ_3 from the training data set $\mathcal{P}$.

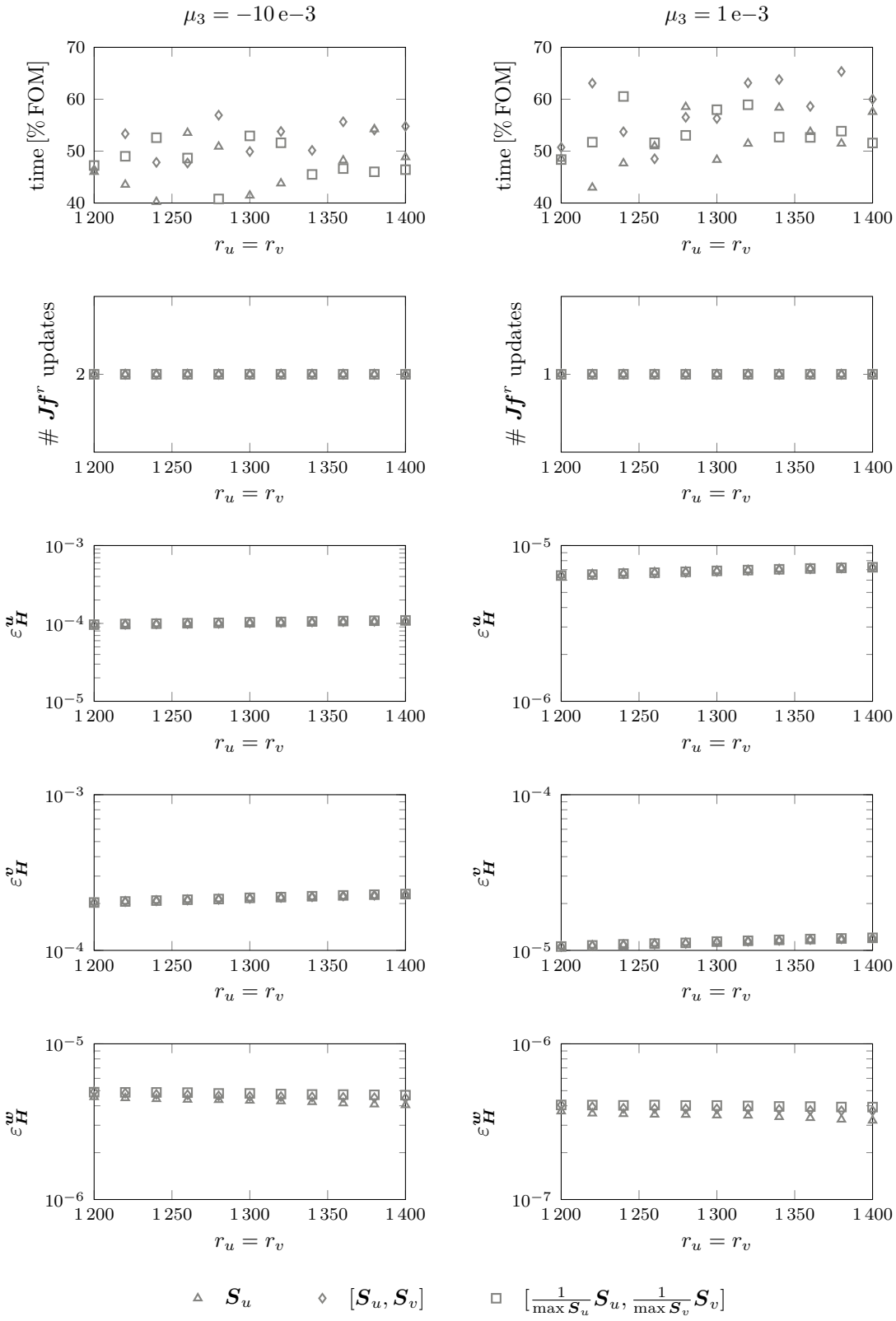


Figure 7.35: *Example 2, BC2 - influence of utilised training data:*
Each of the two columns contains the ROM simulation time in percent of the FOM simulation time, the necessary number of Jacobian updates and the $\mathcal{H}^1$-errors in the $\boldsymbol{u}$-, $\boldsymbol{v}$- and $\boldsymbol{w}$-dof for each simulated reduced size for one of the two chosen parameter values μ_3 from the training data set $\mathcal{P}$.

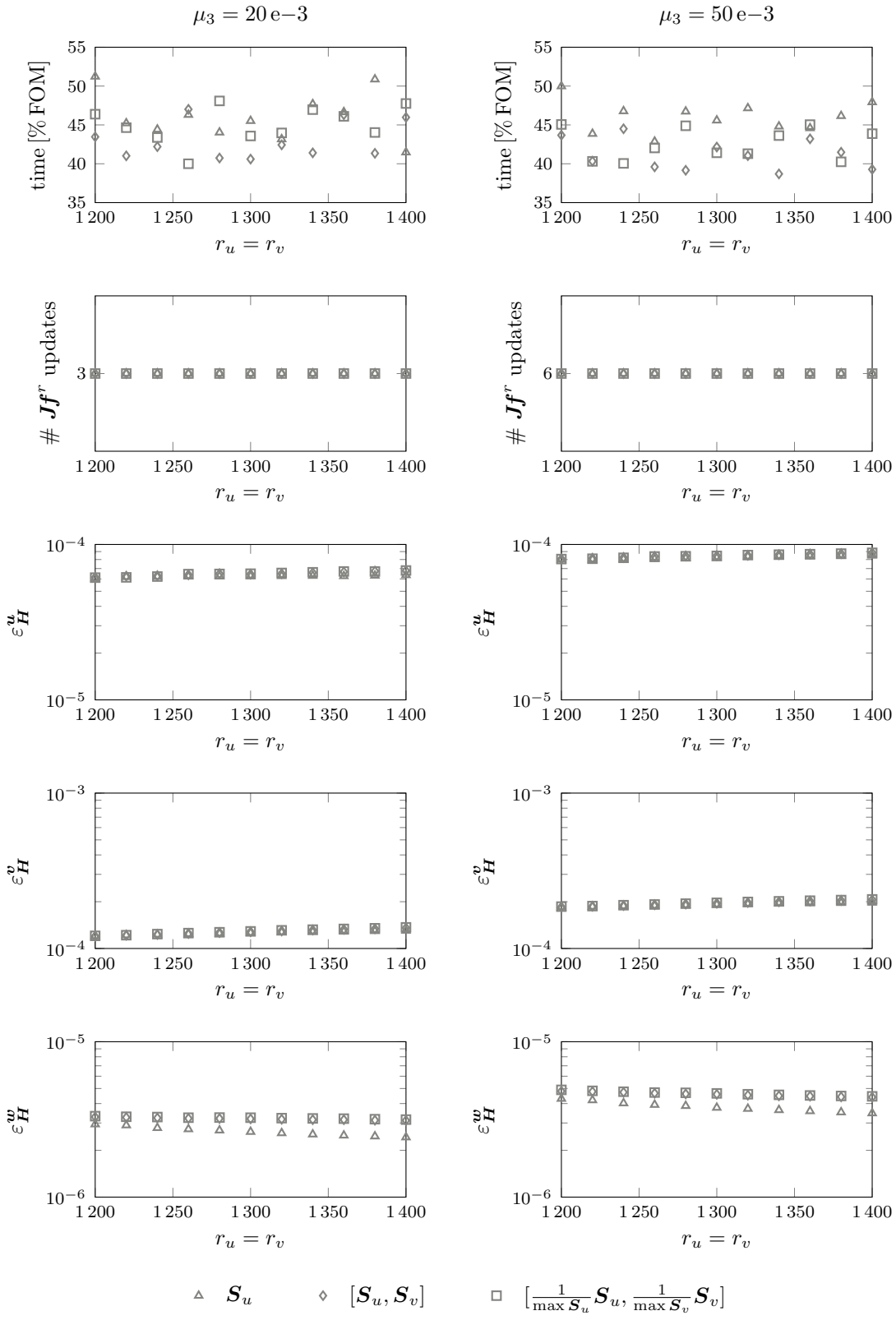


Figure 7.36: *Example 2, BC3 - influence of utilised training data:*
Each of the two columns contains the ROM simulation time in percent of the FOM simulation time, the necessary number of Jacobian updates and the $\mathcal{H}^1$-errors in the $\boldsymbol{u}$-, $\boldsymbol{v}$- and $\boldsymbol{w}$-dof for each simulated reduced size for one of the two chosen parameter values μ_3 from the training data set $\mathcal{P}$.

Evaluating the resulting times and number of reduced Jacobian updates yields no indication as to which of the ROM should be favoured. The number of Jacobian updates is the same for every single simulation. The simulation times all range roughly between $40 - 70\,\%$ of the FOM time and are rather randomly distributed, such that no tendency can be revealed. Counting how often each of the ROM types is the fastest also doesn't reveal any useful information, as it is fairly balanced among all cases ($\boldsymbol{V}_{uv}^{10}$: 20, $\boldsymbol{V}_{uv}^{11}$: 21, $\boldsymbol{V}_{uv}^{\bar{\beta}\bar{\gamma}}$: 23 times the fastest).
The thorough reader might have noticed the lacking results for the ROM built with the POD basis $\boldsymbol{V}_{uv}^{01}$. This is due to convergence issues with these ROM, for which not a single simulation, for none of the boundary condition types and none of the chosen reduced sizes, converged. As reason for this problem, the reduced initial condition $\boldsymbol{z}_{u_0} := \left(\boldsymbol{V}_{uv}^{01}\right)^{-1} \boldsymbol{u}_0$ (c.f. Equation (5.5)) and accordingly the reconstructed initial condition $\boldsymbol{u}_0 \approx \boldsymbol{V}_{uv}^{01} \boldsymbol{z}_{u_0}$, which is needed for the evaluation of the nonlinear parts, was identified. The reduced initial condition was computed using the MATLAB function `lsqminnorm()`, i.e. computing `zu0 = lsqminnorm(V, u0)`. Figure 7.37 shows the difference between the original initial condition $\boldsymbol{u}_0$ and the reconstructed initial condition $\boldsymbol{V}_{(\star)}\boldsymbol{z}_{u_0}$, $\boldsymbol{V}_{(\star)} \in \left\{\boldsymbol{V}_{uv}^{10}, \boldsymbol{V}_{uv}^{11}, \boldsymbol{V}_{uv}^{\bar{\beta}\bar{\gamma}}, \boldsymbol{V}_{uv}^{01}\right\}$ for each dof exemplarily for BC1.

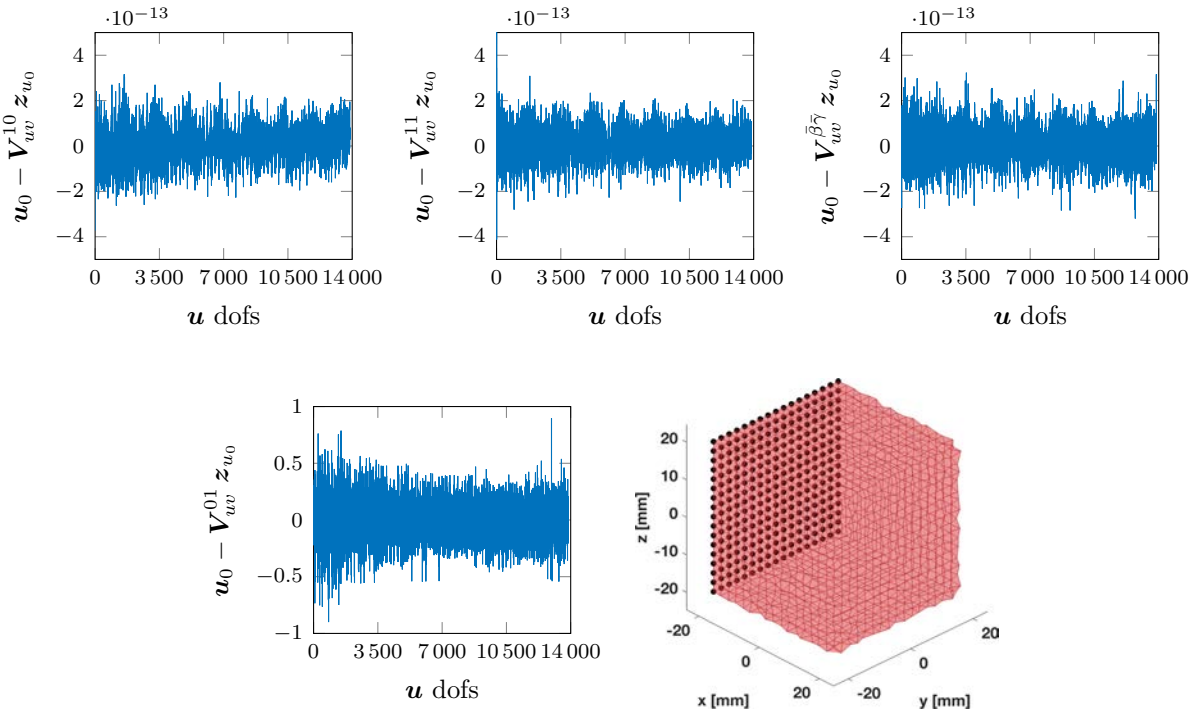


Figure 7.37: *The difference between the original and the reconstructed initial conditions plotted for each dof. Here exemplarily done for BC1 and for the four chosen POD bases. Additionally, on the bottom right, the reconstructed initial condition $\boldsymbol{V}_{uv}^{01}\boldsymbol{z}_{u_0}$ is plotted in the three-dimensional space.*

One can observe that this procedure works very well for the cases $\boldsymbol{V}_{uv}^{10}$, $\boldsymbol{V}_{uv}^{11}$ and $\boldsymbol{V}_{uv}^{\bar{\beta}\bar{\gamma}}$, where the differences are of magnitude 10^{-13} (see Figure 7.37 top row). For the $\boldsymbol{V}_{uv}^{01}$ case, though, the differences are in the interval $[-1, 1]$, thus far too big. Here, the reconstructed initial condition was additionally plotted in the three-dimensional space, where the problem

becomes even more obvious (see Figure 7.37 bottom right). As we have zero velocity initial conditions, no problems occur for those (reduced) dof, since $\boldsymbol{z}_{v_0}$ is also zero. Therefore, it seems like the POD basis $\boldsymbol{V}_{uv}^{01}$ is badly conditioned. To make sure, no coding mistake or similar is the reason for this problem, the orthonormality property was checked and as it should, it holds $\left(\boldsymbol{V}_{uv}^{01}\right)^T \boldsymbol{H_u}\, \boldsymbol{V}_{uv}^{01} = \boldsymbol{I}$. The idea of computing the reduced initial condition by

$$\boldsymbol{z}_{u_0} := \left(\boldsymbol{V}_{uv}^{01}\right)^T \boldsymbol{H_u}\, \boldsymbol{u}_0 \tag{7.12}$$

instead, yielded the same result and so did the MATLAB function `mldivide()`. This issue needs to be addressed in the future.
The hope of finding a difference between the other three ROM types from the initial condition plots, which could have helped in the decision process on how to combine the available training data, has unfortunately not been fulfilled. As the observed differences between the three ROM types were rather insignificant, and other factors seem to have a stronger influence on the performance of the ROM, we refrained from further investigations of weighting the training data $\boldsymbol{V}_{uv}^{\beta\gamma} = [\beta \boldsymbol{S}_u, \gamma \boldsymbol{S}_v]$ with more $\beta, \gamma \in [0, 1]$, at this point. Instead, aspects that seem to have a stronger influence, like the ratio $(r_u = r_v)/r_p$ between the reduced sizes for the position/velocity space and the reduced size for the pressure space, shall be investigated further in the following Section 7.3.
As conclusion of this section, we will from now on calculate the position (and velocity) POD basis by means of $\boldsymbol{H_u}^{-\frac{1}{2}} \text{svd}\left(\boldsymbol{H_u}^{\frac{1}{2}} \boldsymbol{S}_u\right)$, i.e. optimal in the $\mathcal{H}^1$-norm and simply using the position training data $\boldsymbol{S}_u$. The latter choice was made, since including the velocity training data $\boldsymbol{S}_v$ so far was not beneficial and additionally rather unpredictable effects might occur and thus complicate interpretation. Furthermore, the pressure POD basis is always computed through $\boldsymbol{H_p}^{-\frac{1}{2}} \text{svd}\left(\boldsymbol{H_p}^{\frac{1}{2}} \boldsymbol{S}_w\right)$.

7.3 The influence of the combination of reduced position and pressure space

This section further examines the effect of the combination of the reduced position (and velocity) space and pressure space on the performance of the ROM. For this purpose, Section 7.3.1 first investigates the influence of different ratios between the reduced size r_u and the reduced size r_p on stability, accuracy and efficiency of the ROM. Subsequently, in Section 7.3.2, the effect of enriching the position (and velocity) space with approximate supremizer solutions is investigated.
As in the previous section, Example 2 is utilised here with all the three boundary condition cases and the same training data (c.f. Section 7.1.3 - offline phase). For both investigations, ROM with four different reduced sizes r_p were built. The sizes r_p were chosen based on the kinks shown in the singular value decays (c.f. Figures 7.41–7.43) and smaller, i.e. already making use of the results from the previous sections. Ballarin et al. [4] give the advice to choose the number of supremizer modes, r_s, to enrich the position space with, smaller or equal to the number of pressure modes, i.e. $r_s \leq r_p$. Therefore, for each of the reduced size r_p, the ROM with a supremizer enriched position (and velocity) POD basis, were built choosing two different reduced sizes r_u combined with four different sizes $r_s \leq r_p$. In order

to appropriately compare the ROM obtained with and without supremizer enrichment, the reduced sizes for the pure position spaces, here named r_{us}, were chosen such that the obtained sizes of the overall ROM were equal, i.e. setting $r_{us} := r_u + r_s$. Tables 7.9 and Tables 7.10 list the chosen reduced sizes.

r_{us}	r_p			
	80	100	120	140
r_u	$\{920, 1\,120\}$	$\{900, 1\,100\}$	$\{880, 1\,080\}$	$\{860, 1\,060\}$
r_s	$\{20, 40, 60, 80\}$	$\{40, 60, 80, 100\}$	$\{60, 80, 100, 120\}$	$\{80, 100, 120, 140\}$
r_{us}	$\{940, 960, 980, 1\,000, 1\,140, 1\,160, 1\,180, 1\,200\}$			

Table 7.9: *Example 2, BC1:*
The size combinations for building the ROM used for the comparison of different ratios r_p/r_u on the one hand, and for investigating the effect of adding r_s supremizers to the position and velocity space on the other hand.

r_{us}	r_p			
	110	130	150	170
r_u	$\{1\,090, 1\,290\}$	$\{1\,070, 1\,270\}$	$\{1\,050, 1\,250\}$	$\{1\,030, 1\,230\}$
r_s	$\{50, 70, 90, 110\}$	$\{70, 90, 110, 130\}$	$\{90, 110, 130, 150\}$	$\{110, 130, 150, 170\}$
r_{us}	$\{1\,140, 1\,160, 1\,180, 1\,200, 1\,340, 1\,360, 1\,380, 1\,400\}$			

Table 7.10: *Example 2, BC2 and BC3:*
The size combinations for building the ROM used for the comparison of different ratios r_p/r_u on the one hand, and for investigating the effect of adding r_s supremizers to the position and velocity space on the other hand.

7.3.1 Examination of the ratio between the reduced sizes of position and pressure space

Figures 7.38–7.40 show the results obtained for BC1, BC2 and BC3, respectively. Each of the two columns contains the $\mathcal{H}^1$-errors in the $\boldsymbol{u}$-, $\boldsymbol{v}$- and $\boldsymbol{w}$-dof, the necessary number of reduced Jacobian updates and the ROM simulation time in percent of the FOM simulation time, for each simulated reduced size combination r_u with r_p for one of the two chosen parameter values μ_3 from the training data set $\mathcal{P}$. The largest reduced sizes r_u were, as before, chosen based on the kink in the singular value decays. Recalling the property of the remaining singular values as a priori error estimate, and theoretically being satisfied with less accurate results, further sizes were chosen smaller than that.
For all the three cases, one can observe that, at least within the chosen range, the reduced size r_u $(= r_v)$ has only an insignificant influence on the errors, which are obviously dominated by the choice of the size r_p of the reduced pressure space. Merely for the error in the pressure coefficients $\boldsymbol{w}$ of BC3 (c.f. Figure 7.40, row 3), one can see a minor positive effect, when increasing the number of reduces position modes included in the POD basis.

On the other hand, increasing the size r_p from roughly 10 % of the FOM size N_p to 20 % in the BC1 case and from 15 % of the FOM size N_p to 25 % in the BC2 and BC3 case, results in a decrease of all errors ($\boldsymbol{u}$,$\boldsymbol{v}$ and $\boldsymbol{w}$) around one to two orders of magnitude.

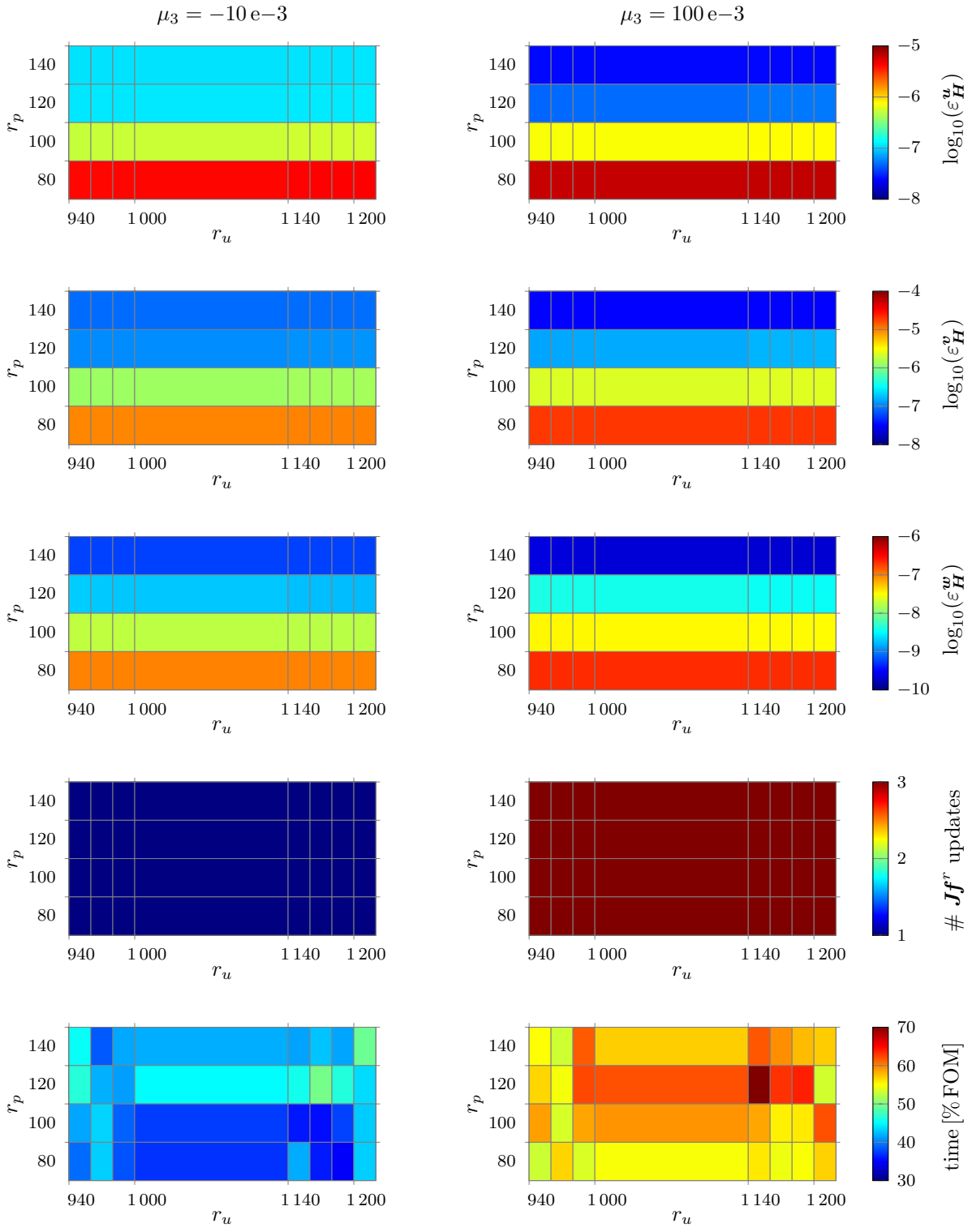


Figure 7.38: *Example 2, BC1: Results for ROM built with different ratios r_u/r_p.*

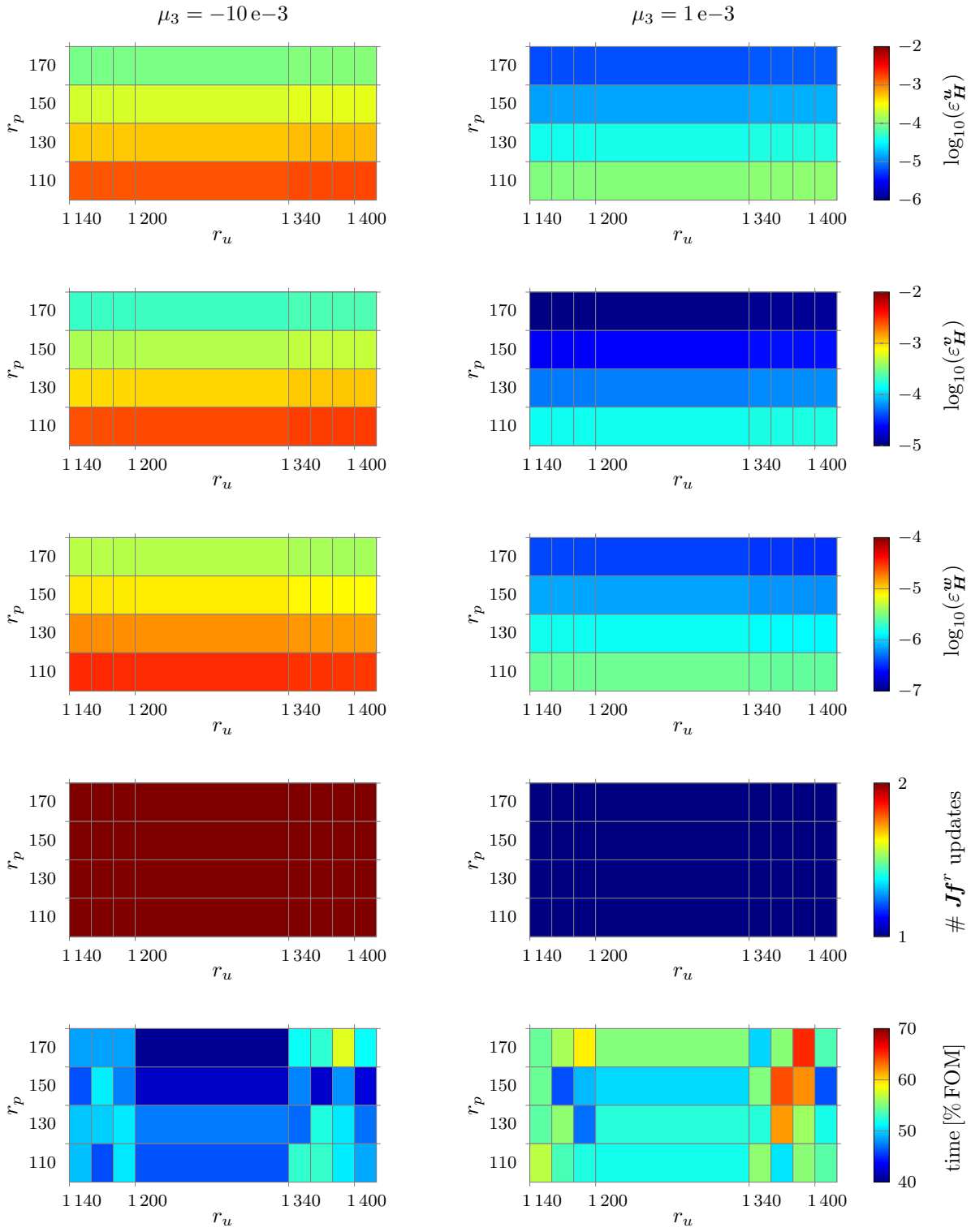


Figure 7.39: *Example 2, BC2: Results for ROM built with different ratios r_u/r_p.*

The number of Jacobian updates is the same for all simulations performed within the chosen intervals for r_u and r_p. Thus, this cannot give any advice (at least for the examples investigated in this section) on the ratio between the size of the reduced position and pressure space. Maybe for future work it is advisable to additionally look at e.g. a mean

condition number of the reduced Jacobians.
As for the differences in the simulation time of the reduced models, if at all, an opposite effect compared to the errors can be noted. For increasing sizes r_p, the simulation time tends to increase for most of the simulations. However, the time plots also contain a lot

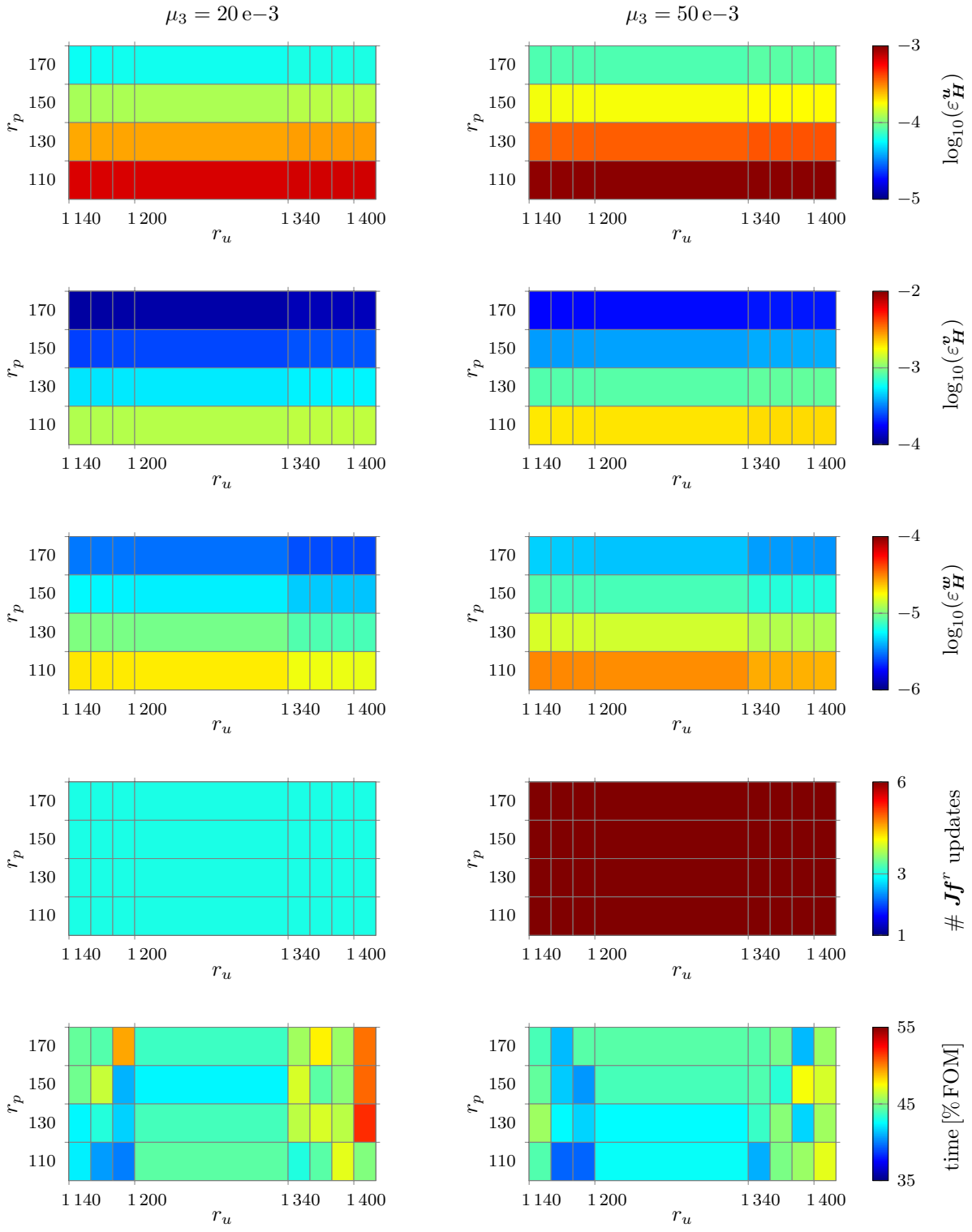


Figure 7.40: *Example 2, BC3: Results for ROM built with different ratios r_u/r_p.*

of random effects and no clear pattern can be recognised. To summarise, it seems impossible to give an advice or come up with a heuristic criterion on how to choose the reduced sizes r_u and r_p based on the results of this section.

7.3.2 The benefit of supremizer enrichment

Now, the influence of enriching the position space with approximate supremizer solutions shall be investigated. As described in Section 5.3, for this purpose, the offline phase needs to be extended by additionally computing approximate supremizer solutions and a POD basis $\boldsymbol{V}_s$. Tables 7.9 and 7.10 list the two chosen reduced sizes r_u for each of the four chosen sizes r_p. As described in Section 5.3.2 it is beneficial to construct the supremizer POD basis $\boldsymbol{V}_s$ orthonormal to the position POD basis $\boldsymbol{V}_u$. This means that prior to the computation of $\boldsymbol{V}_s$, one needs to specify the size of the space $\boldsymbol{V}_u$ that needs to be excluded. Here, it was necessary to compute eight different supremizer POD bases for each of the boundary condition cases. The corresponding singular value decays are shown in Figures 7.41–7.43.

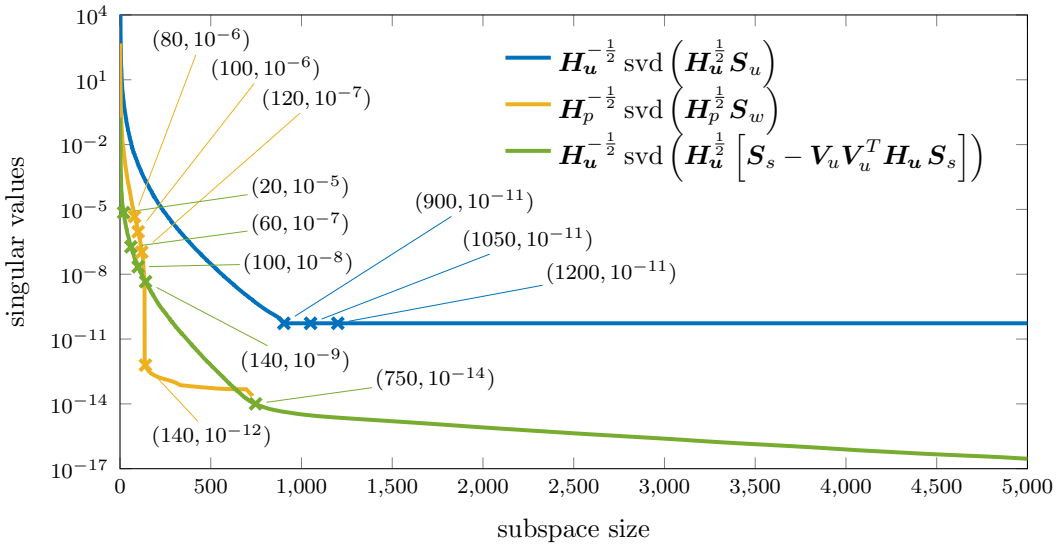


Figure 7.41: *Example 2, BC1: The singular value decays of the POD on the supremizer data $\boldsymbol{S}_s$ together with the (already previously plotted) singular value decays of the displacement and the pressure data. The supremizer data singular value decays are similar for each size r_u of the excluded space $\boldsymbol{V}_u$.*

Every ROM in this section, constructed with the chosen combination of POD bases and reduced sizes, converged. This is already a good result. The simulation times of the ROM range between $30-75\,\%$ of the corresponding FOM simulation time. This represents quite a wide range. While one might be satisfied with a ROM that yields results three times faster than the FOM, an insignificant speedup of 1.3 does not really compensate for a time-consuming offline phase. Even keeping in mind that the nonlinear components are not approximated yet.

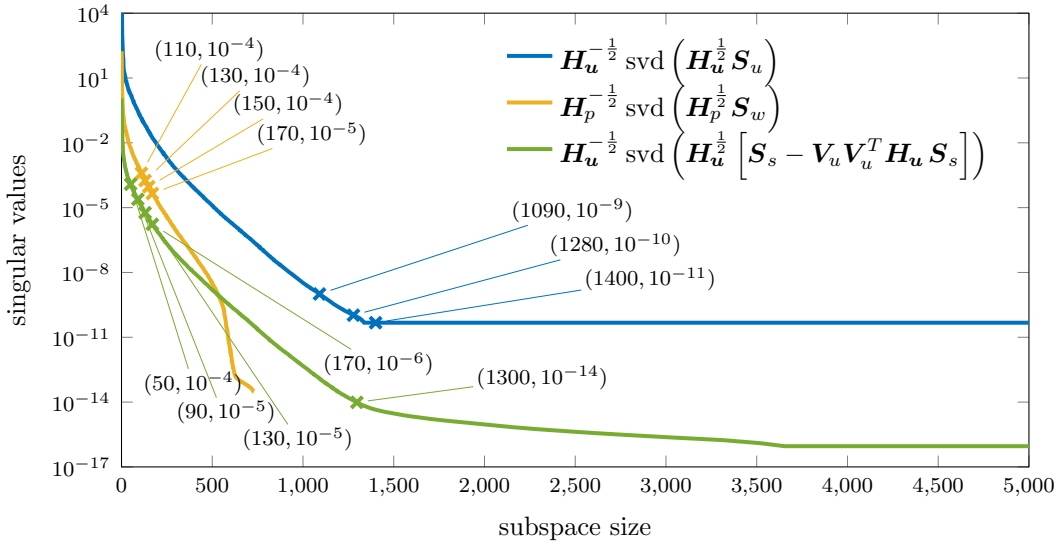


Figure 7.42: *Example 2, BC2: The singular value decays of the POD on the supremizer data $\boldsymbol{S}_s$ together with the (already previously plotted) singular value decays of the displacement and the pressure data. The supremizer data singular value decays are similar for each size r_u of the excluded space $\boldsymbol{V}_u$.*

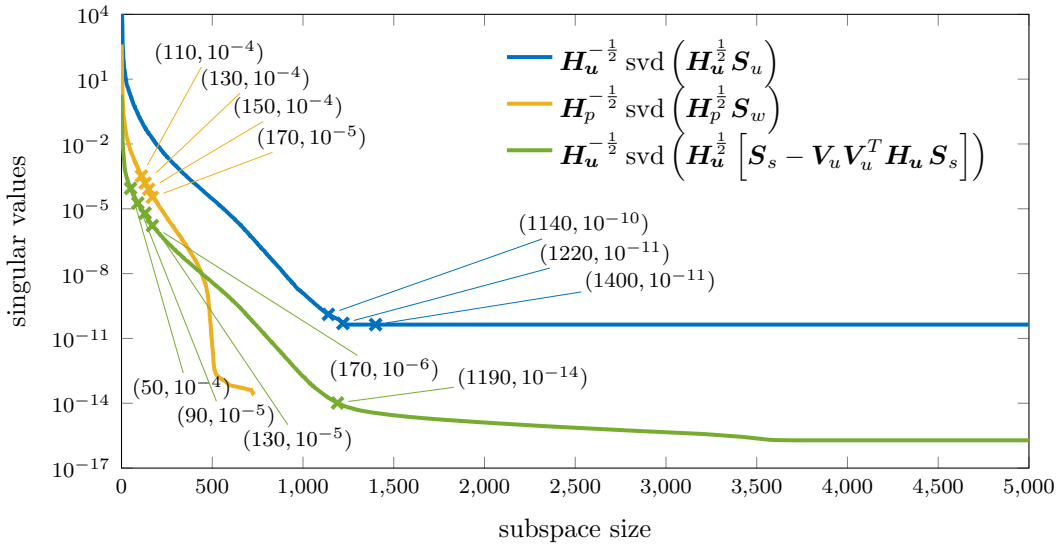


Figure 7.43: *Example 2, BC3: The singular value decays of the POD on the supremizer data $\boldsymbol{S}_s$ together with the (already previously plotted) singular value decays of the displacement and the pressure data. The supremizer data singular value decays are similar for each size r_u of the excluded space $\boldsymbol{V}_u$.*

Figures 7.44–7.46 show the results of the reduced simulations in more detail. As before, the simulation time in percent of the corresponding FOM simulation time, the number of reduced Jacobian updates and the errors in the $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ dof are plotted in each figure for the two chosen parameters $\mu_3 \in \mathcal{P}$ in two columns.

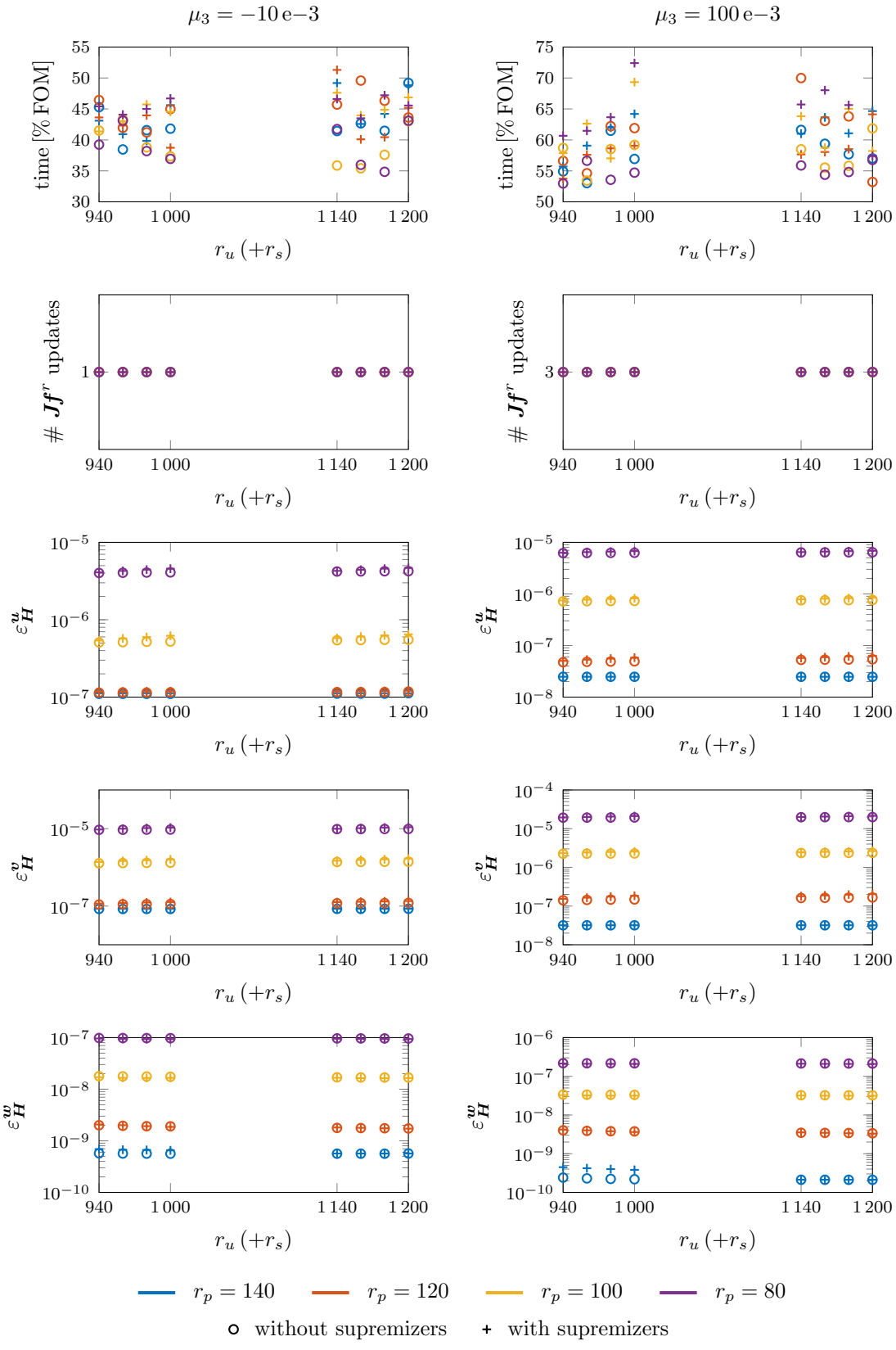


Figure 7.44: *Example 2, BC1:*
Comparing the performance of the ROM built with and without supremizer enrichment of the position POD basis for different reduced sizes.

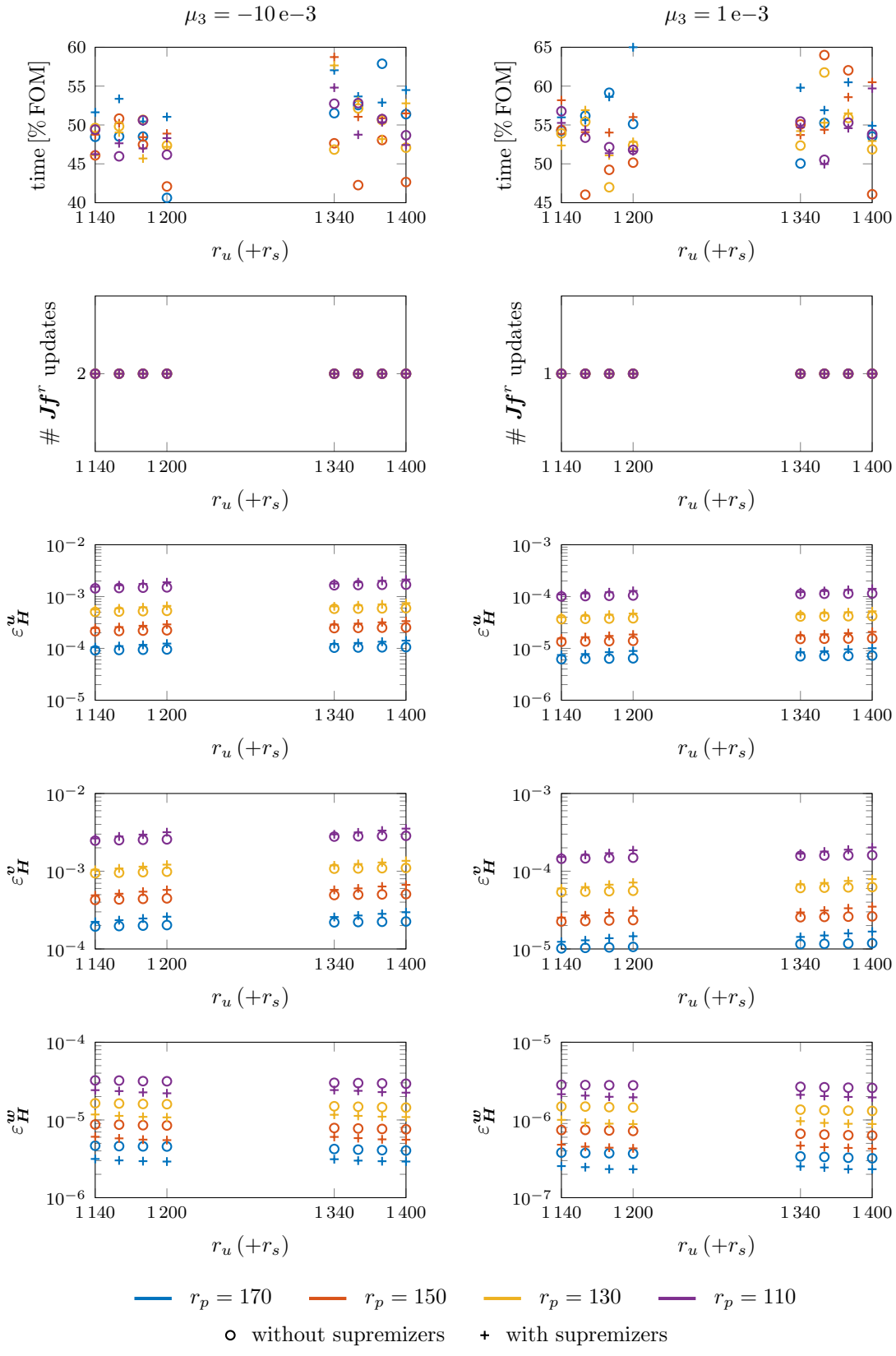


Figure 7.45: *Example 2, BC2:*
Comparing the performance of the ROM built with and without supremizer enrichment of the position POD basis for different reduced sizes.

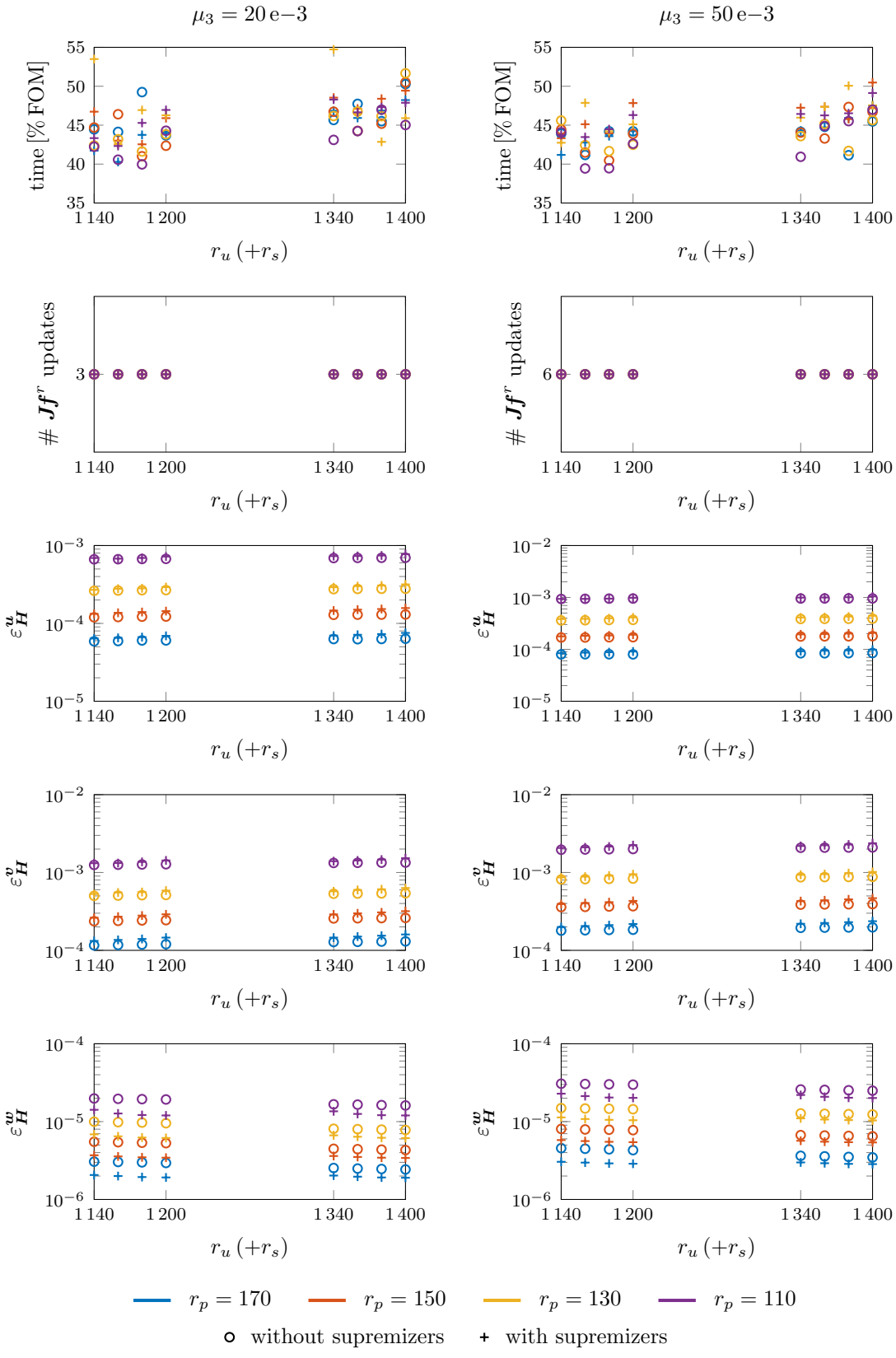


Figure 7.46: *Example 2, BC3:*
Comparing the performance of the ROM built with and without supremizer enrichment of the position POD basis for different reduced sizes.

To begin with, one can observe no significant difference between the simulations performed with and without the supremizer basis enrichment. Whether a ROM of overall size $2 \times r_{us} + r_p$ was built purely with POD modes from $\boldsymbol{V}_u$ or with a combination of POD modes from $\boldsymbol{V}_u$ and $\boldsymbol{V}_s$ has only little influence on the error. Taking a close look, one can identify that adding the supremizer modes slightly increases the error in the $\boldsymbol{u}$ and $\boldsymbol{v}$ dof, whereas the error in the $\boldsymbol{w}$ dof decreases. However, this effect is as expected and appears negligible. Again, it becomes obvious that the choice of the reduced size r_p of the pressure space dominates the error. This is the case for the position, velocity and pressure dof in each of the three scenarios that were tested.
The number of necessary Jacobian updates does not differ among the different ROM and thus cannot give any advice on which method to favour. The same holds for the simulation times, which are very randomly distributed again. While sometimes the ROM with supremizer basis enrichment is faster than the ROM of corresponding reduced size without added supremizers, the opposite is the case just as often.
An important fact to mention here is that ROM built with POD bases, where the supremizer POD basis $\boldsymbol{V}_s$ was not constructed orthonormal to the POD basis $\boldsymbol{V}_u$, did not converge. Thus, preserving the algebraic stability seems to be more important than improving the approximation stability. Again, one could suggest to look additionally at the condition numbers of the reduced Jacobians as additional means to decide which of the ROM to choose. Since no tendency could be revealed so far, both approaches will be applied in the next section for Examples 3, i.e. the models with increasing complexity.

7.4 Results for models with increasing complexity

In this section, the method that seemed to have proven most suitable, by looking at the results from all previous investigations, shall be applied to the Examples 3 in order to see how it performs if one goes towards the goal of the dynamic skeletal muscle model. Summarised, this means:

- Utilising training data $\boldsymbol{S}_u$, $\boldsymbol{S}_w$ to compute POD bases $\boldsymbol{V}_u$, $\boldsymbol{V}_w$ that are optimal in the inner product norm, i.e. setting $\boldsymbol{\mathcal{A}} = \boldsymbol{H}$.

- Looking at the corresponding singular value decays and choosing reduced sizes r_u and r_p that seem appropriate.

- Choosing the same approximation space and reduced dimension for the velocity dof, i.e. $\boldsymbol{V}_v = \boldsymbol{V}_u$ and $r_v = r_u$.

- Computation of approximate supremizer training data $\boldsymbol{S}_s$ based on the available training data and constructing a POD basis $\boldsymbol{V}_s$ that is orthonormal to $\boldsymbol{V}_u$ (already using the chosen size r_u).

- Construction of a ROM, either

  $\circ$: simply using the position POD basis $\boldsymbol{V}_u$, or

  $+$: enriching with supremizers and thus choosing $r_s \leq r_p$ and approximating with $(\boldsymbol{V}_u \, \boldsymbol{V}_s)$.

This approach is first applied to Example 3A with its increasing material complexity in Section 7.4.1 for the two different simulation scenarios. Then, in Section 7.4.2 it is applied to Example 3B with its more complex, fusiform shaped muscle geometry. Finally, in Section 7.4.3, the methods are examined for a case, where the online simulation is not contained in the training data used for the generation of the POD bases.

7.4.1 Increasing the material complexity

As FOM, Example 3A, i.e. the cubic geometry, with a discretisation $\mathrm{d}x = 5\,\mathrm{mm}$ is used. For recalling details on the setup, the reader is referred to Sections 6.1.3 and 6.3.3.

Offline phase - computation of training data

For the scenario BCxz, the training parameter is the applied traction force, i.e. μ_3. For the scenario BCxact, the maximum activation $\alpha_{\max}$ is the training parameter. $n_p = 5$ parameter values were chosen for the generation of training data in both cases,

$$\text{BCxz:} \quad \mu_3 \in \{5, 15, 25, 35, 45\}\,\text{e}{-3}\,\text{MPa}\,,$$

$$\text{BCxact:} \quad \alpha_{\max} \in \{0.2, 0.4, 0.6, 0.8, 1.0\}\,.$$

Accounting for the zero DIRICHLET boundary conditions and the four times finer time discretisation in the BCxz case, the dimensions of the obtained snapshot matrices are:

$$\text{BCxz:} \quad \boldsymbol{S}_u, \boldsymbol{S}_s \in \mathbb{R}^{13\,872 \times 20\,005}\,, \quad \boldsymbol{S}_w \in \mathbb{R}^{729 \times 20\,005}\,,$$

$$\text{BCxact:} \quad \boldsymbol{S}_u, \boldsymbol{S}_s \in \mathbb{R}^{13\,872 \times 5\,005}\,, \quad \boldsymbol{S}_w \in \mathbb{R}^{729 \times 5\,005}\,.$$

Figures 7.47 and 7.48 show the singular value decays of the subsequently performed SVD on each of the snapshot matrices.

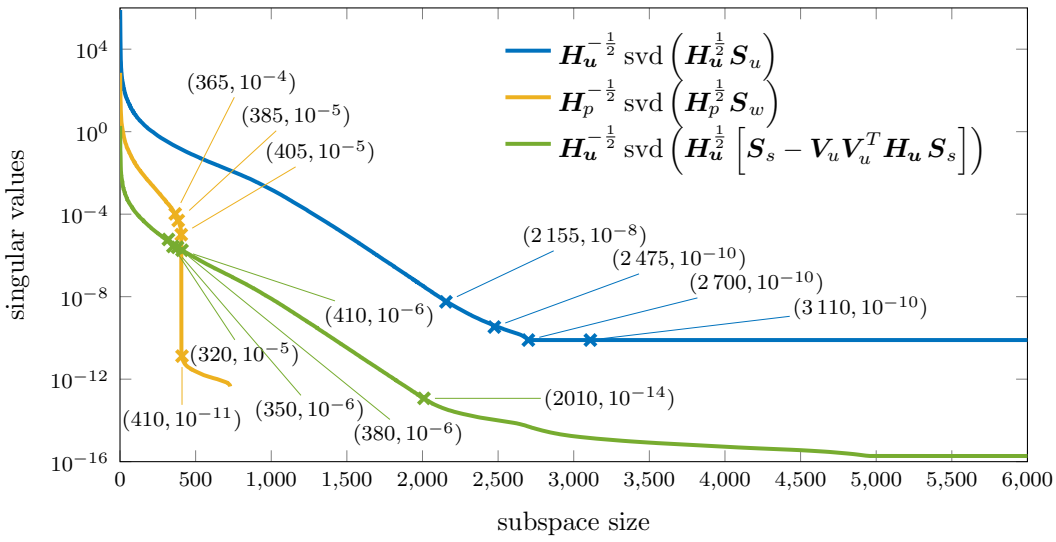


Figure 7.47: *Example 3A, BCxz:*
The singular value decays of the POD on the supremizer data $\boldsymbol{S}_s$ together with the singular value decays of the position and the pressure data. The supremizer data singular value decays are very similar for each size r_u (see below) of the excluded space $\boldsymbol{V}_u$, thus only one case is plotted here.

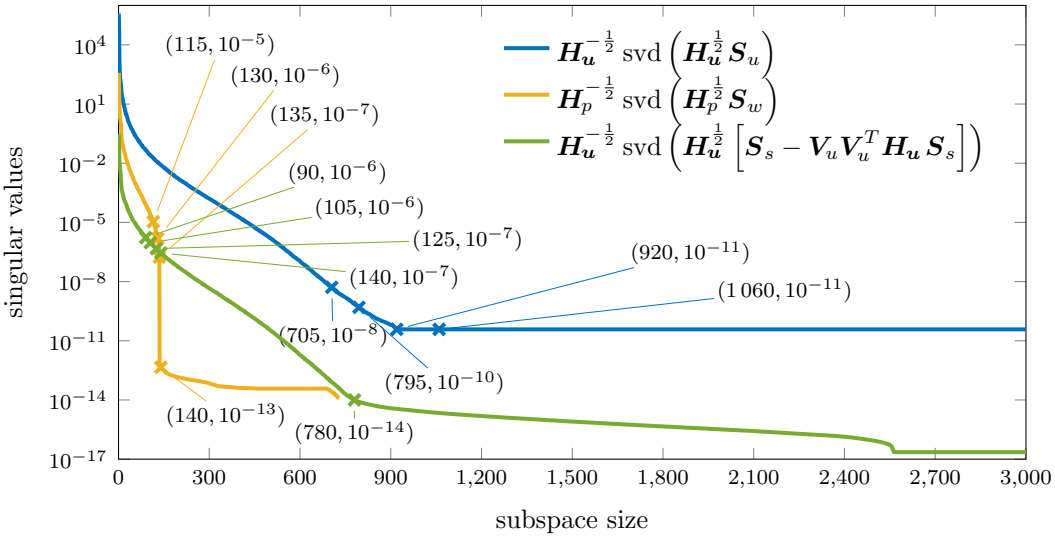


Figure 7.48: *Example 3A, BCxact:*
The singular value decays of the POD on the supremizer data $\boldsymbol{S}_s$ together with the singular value decays of the position and the pressure data. The supremizer data singular value decays are very similar for each size r_u (see below) of the excluded space $\boldsymbol{V}_u$, thus only one case is plotted here.

Online phase - building and simulating different ROM

The range for each of the reduced sizes r_u (r_{us}) and r_p for the different ROM was chosen based on the obtained singular value decay and the values are listed in Tables 7.11 and 7.12. As in the previous section, it was made sure that the sizes $r_{us} := r_u + r_s$ were set in such a way that a comparison of ROM with the same overall size is possible.
For the online simulation with the different ROM, again two parameters out of the training parameter set $\mathcal{P}$ were tested. Figures 7.49 and 7.50 show the results of all simulations performed with the various reduced size combinations for the case BCxz and BCxact respectively. As before, the simulation time in percent of the corresponding FOM simulation time, the number of reduced Jacobian updates and the errors in the $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ dof are plotted in each figure for the two chosen parameters $\mu_3 \in \mathcal{P}$ or $\alpha_{\max} \in \mathcal{P}$ in two columns.

r_{us}	r_p			
	365	385	405	410
r_u	$\{2\,155, 2\,745\}$	$\{2\,135, 2\,725\}$	$\{2\,115, 2\,705\}$	$\{2\,110, 2\,700\}$
r_s	$\{320, 340, 360, 365\}$	$\{340, 360, 380, 385\}$	$\{360, 380, 400, 405\}$	$\{365, 385, 405, 410\}$
r_{us}	$\{2\,475, 2\,495, 2\,515, 2\,520, 3\,065, 3\,085, 3\,105, 3\,110\}$			

Table 7.11: *Example 3A, BCxz:*
The size combinations for building the ROM used for the comparison of different ratios r_p/r_u on the one hand, and for investigating the effect of adding r_s supremizers to the position and velocity space on the other hand.

r_{us}	r_p			
	115	130	135	140
r_u	$\{705, 945\}$	$\{690, 930\}$	$\{685, 925\}$	$\{680, 920\}$
r_s	$\{90, 105, 110, 115\}$	$\{105, 120, 125, 130\}$	$\{110, 125, 130, 135\}$	$\{115, 130, 135, 140\}$
r_{us}	$\{795, 810, 815, 820, 1\,035, 1\,050, 1\,055, 1\,060\}$			

Table 7.12: *Example 3A, BCxact:*
The size combinations for building the ROM used for the comparison of different ratios r_p/r_u on the one hand, and for investigating the effect of adding r_s supremizers to the position and velocity space on the other hand.

For the BCxz case, the following observations can be made. First of all, and maybe most importantly, for the case without supremizer enrichment, 6 out of the 64 reduced simulations did not converge, while the ROM of corresponding size, but built with supremizer enriched position and velocity POD basis, did. This occurred for the two larger values r_p and the more dynamic load case $\mu_3 = 45\,\mathrm{e}{-3}\,\mathrm{MPa}$ in combination with some of the smaller reduced sizes r_u. This observation stands in high favour for enriching the ROM with approximate supremizer solutions. Additionally, this can be supported by the necessary number of Jacobian updates. For this more complex Example 3 now, this number varies a lot among the different ROM, and as shown in the second row of Figure 7.49, is not always, but frequently lower for the supremizer enriched ROM. Especially for the smaller ROM and the more dynamic load case $\mu_3 = 45\,\mathrm{e}{-3}\,\mathrm{MPa}$, this is evident. Here, for ROM built with supremizer enriched POD basis, the necessary number of updates is best between 15 and 25, while it is between 35 and 40 for all ROM built without considering supremizer solutions. Naturally, this is reflected in the solution time as well. The best, i.e. fastest simulations were performed with the supremizer enriched ROM. As for the errors, the observations from Section 7.3 are confirmed. Adding the supremizer modes slightly increases the error in the $\boldsymbol{u}$ and $\boldsymbol{v}$ dof, whereas the error in the $\boldsymbol{w}$ dof decreases. However, compared to the effect, which the reduced size r_p of the pressure space has on the error, this is again negligible.
For the BCxact case, generally one can observe that the differences among the ROM without and with supremizer enrichment are not very pronounced. They seem more random again, as for the Examples 2. However, they are also not contradictory to the statements made for Example 3, BCxz. A single inconstancy appears in the error in $\boldsymbol{w}$, which is also present in the BCxz case. Choosing r_p too large with respect to r_u has a negative effect on the error in $\boldsymbol{w}$ in the supremizer enriched ROM (see blue + in the last rows of Figures 7.49 and 7.50).

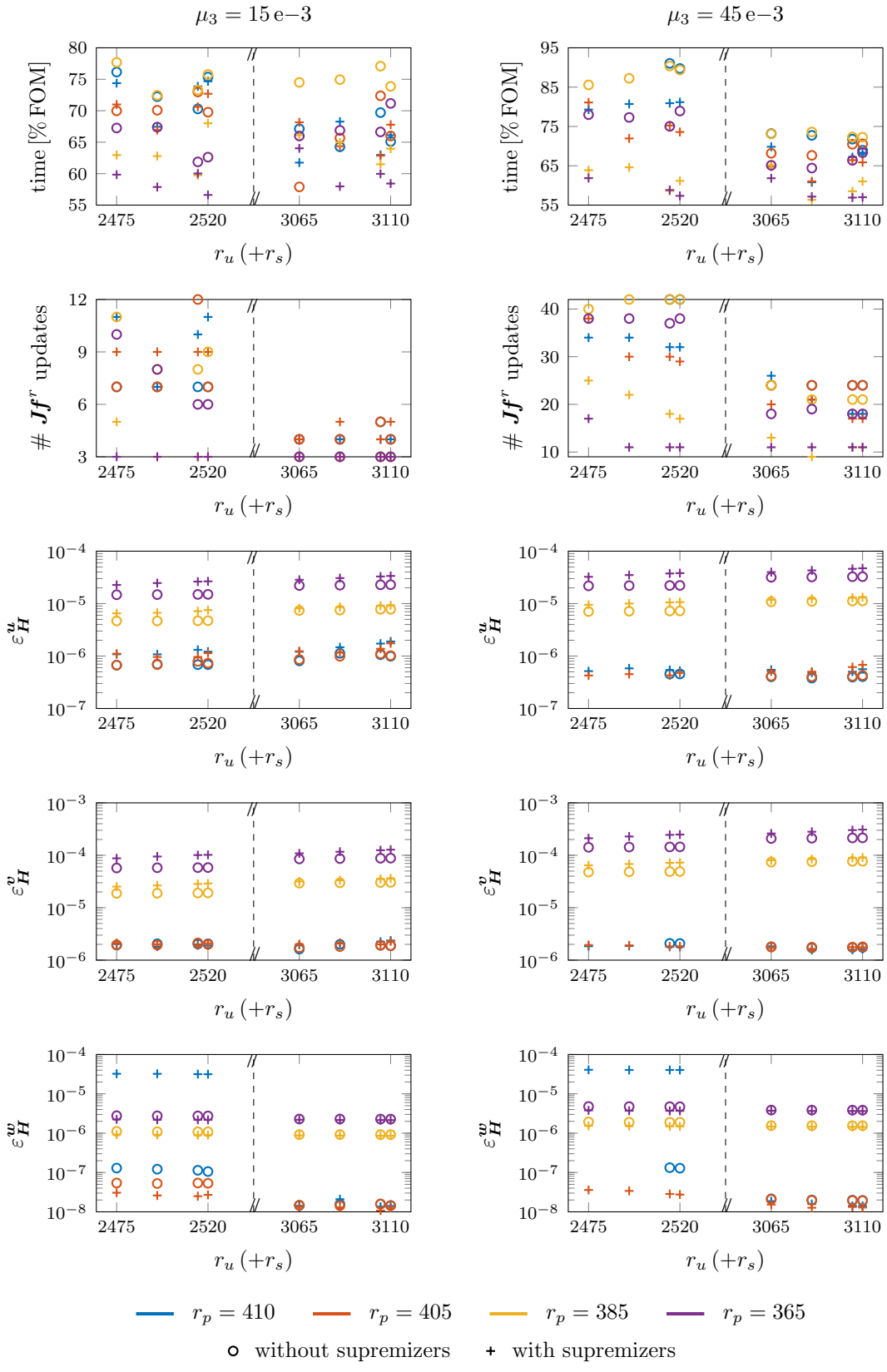


Figure 7.49: *Example 3A, BCxz:*
Comparing the performance of the ROM built with and without supremizer enrichment of the position POD basis for different reduced sizes.

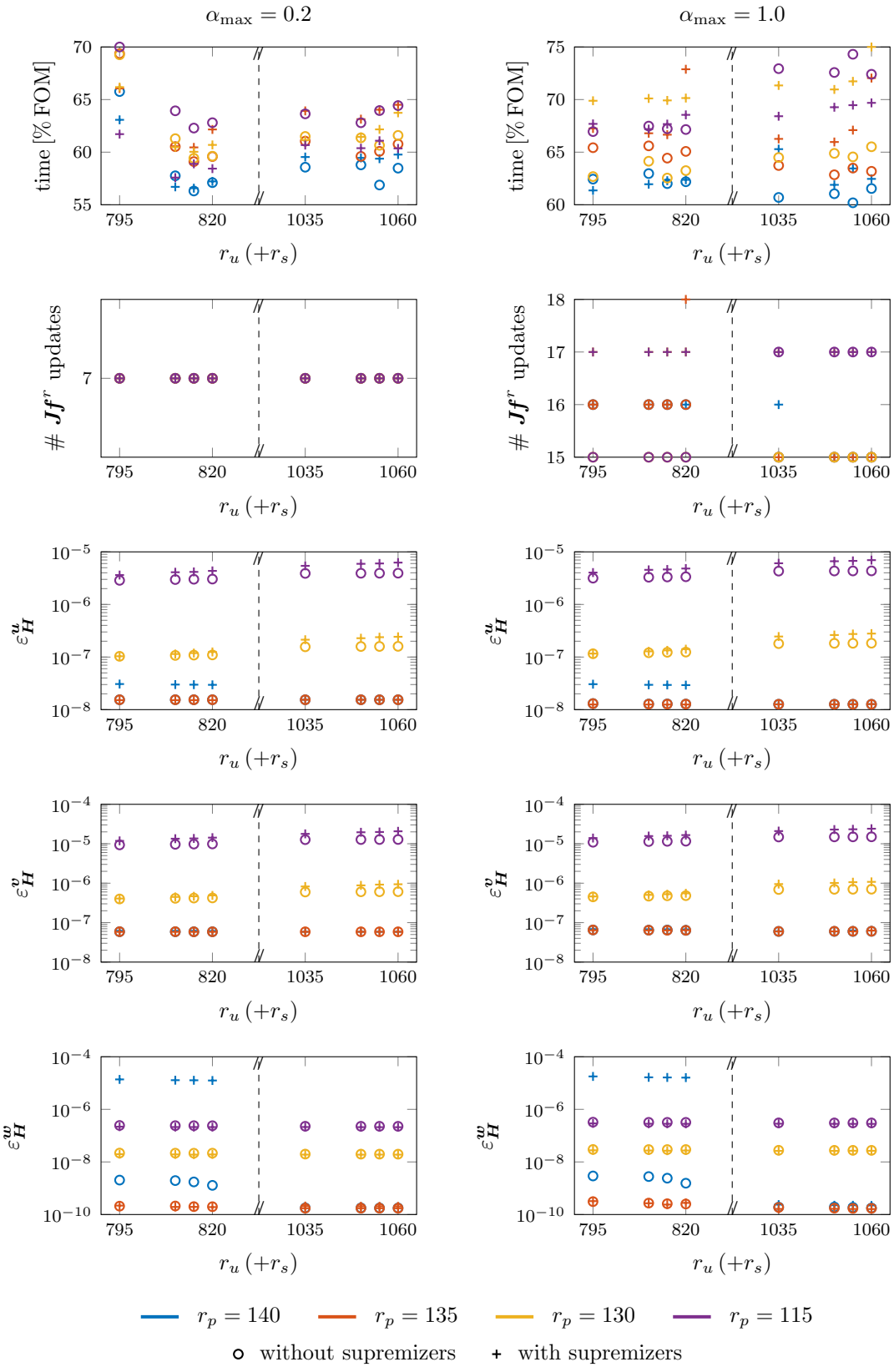


Figure 7.50: *Example 3A, BCxact:*
Comparing the performance of the ROM built with and without supremizer enrichment of the position POD basis for different reduced sizes.

7.4.2 Increasing the geometrical complexity

As FOM, Example 3B, i.e. the fusiform muscle geometry, discretised with 800 elements is used here. Details on the setup can be recalled by looking back into Sections 6.1.3 and 6.3.3.

Offline phase - computation of training data

In this scenario, the maximum activation $\alpha_{\max}$ is the training parameter. Again, $n_p = 5$ parameter values $\alpha_{\max} \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$ were chosen for the generation of training data. Accounting for the zero DIRICHLET boundary conditions and the shorter time interval for the simulation (only 50 ms here), the dimensions of the obtained snapshot matrices are $\boldsymbol{S}_u, \boldsymbol{S}_s \in \mathbb{R}^{21\,413 \times 2\,505}$, and $\boldsymbol{S}_w \in \mathbb{R}^{1\,029 \times 2\,505}$. Figure 7.51 shows the singular value decays of the subsequently performed SVD on each of the snapshot matrices.

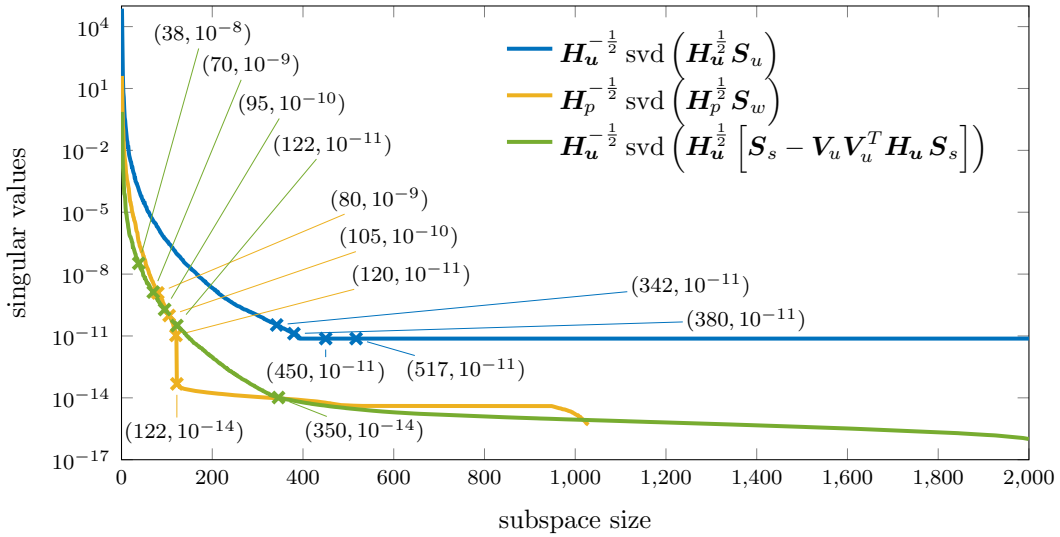


Figure 7.51: *Example 3B: The singular value decays of the POD on the supremizer data $\boldsymbol{S}_s$ together with the singular value decays of the position and the pressure data. The supremizer data singular value decays are very similar for each size r_u of the excluded space $\boldsymbol{V}_u$, thus only one case is plotted here.*

Online phase - building and simulating different ROM

Again, the range for each of the reduced sizes r_u (r_{us}) and r_p was chosen based on the obtained singular value decay. The values are listed in Table 7.13.
For the online simulation with the different ROM, again two parameters out of the training parameter set $\mathcal{P}$ were tested. Figure 7.52 shows the results of all simulations performed with the various reduced size combinations. Like in the activation case with the cubic geometry, no significant difference can be made between the ROM with and without supremizer enriched position and velocity basis. The errors in $\boldsymbol{u}$ and $\boldsymbol{v}$ for this case can be called the same. The error in $\boldsymbol{w}$ again is negligibly smaller in the ROM without supremizer enrichment.

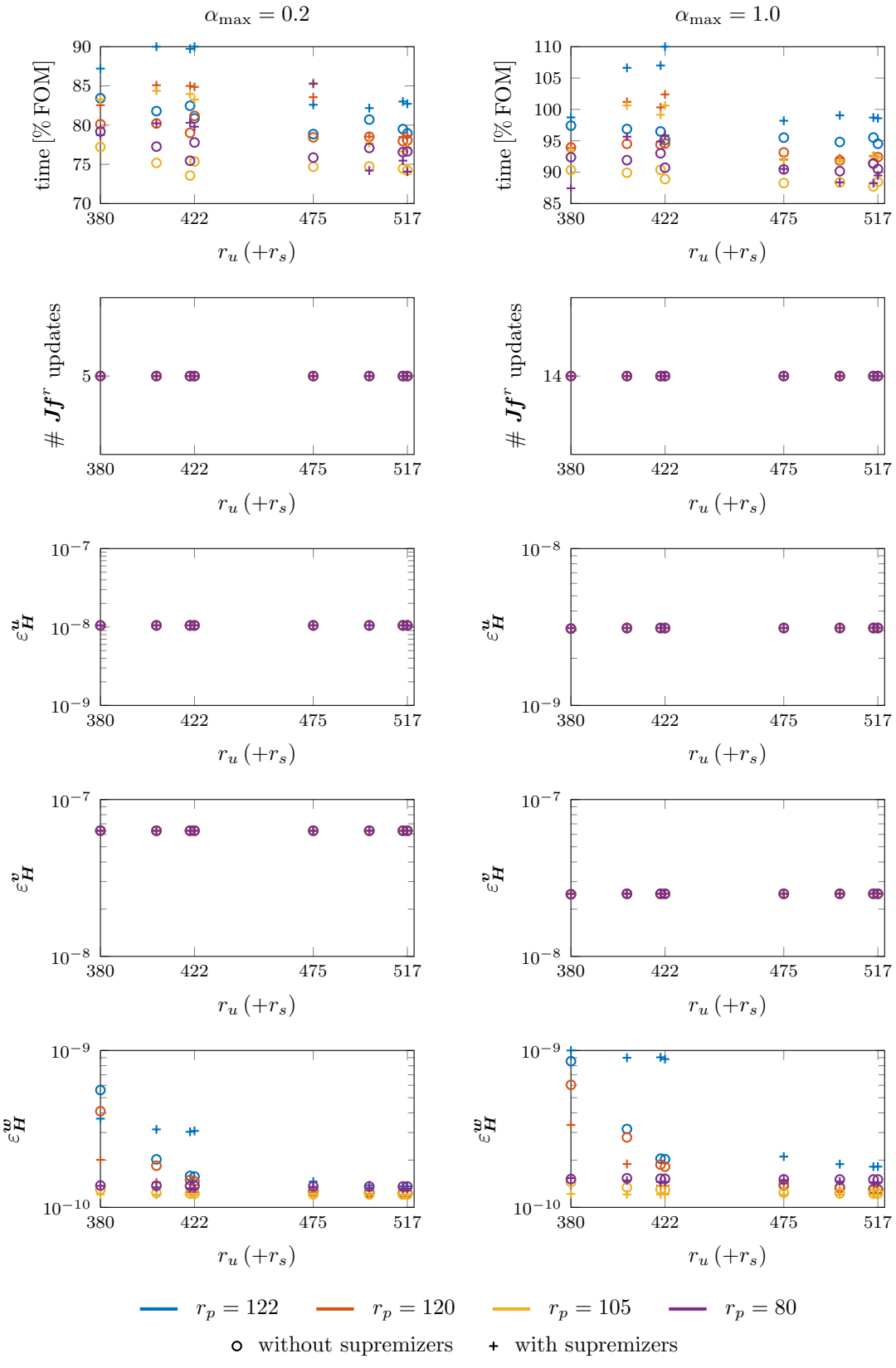


Figure 7.52: *Example 3B:*
Comparing the performance of the ROM built with and without supremizer enrichment of the position POD basis for different reduced sizes.

r_{us}	r_p			
	80	105	120	122
r_u	$\{342, 437\}$	$\{317, 412\}$	$\{302, 397\}$	$\{300, 395\}$
r_s	$\{38, 63, 78, 80\}$	$\{63, 88, 103, 105\}$	$\{78, 103, 118, 120\}$	$\{80, 105, 120, 122\}$
r_{us}	$\{380, 405, 420, 422, 475, 500, 515, 517\}$			

Table 7.13: *Example 3B: The size combinations for building the ROM used for the comparison of different ratios r_p/r_u on the one hand, and for investigating the effect of adding r_s supremizers to the position and velocity space on the other hand.*

While the number of necessary Jacobian updates does not vary among the ROM here, the simulation times range between 70 and 110 percent of the FOM simulation time. Especially the ROM with larger reduced size r_p and smaller r_u stand out negatively here. Thus, this example again confirms that the performance of the ROM is very sensitive concerning the chosen size of the pressure space with respect to the position space.

7.4.3 Simulating a scenario not included in the training data

The idea behind this test case is to have a selection of reduced-order muscles, which were e.g. trained with different boundary conditions, loading directions and activation states. Then, those could potentially be assembled in a multi-muscle and multi-body system to obtain an overall simulation, that is much faster than one performed with full-order muscle models. Example 3A, i.e. the cubic muscle geometry with the more complex material, with discretisation $\mathrm{d}x = 5\,\mathrm{mm}$ serves as FOM for this investigation.

Offline phase - computation of training data

During the offline phase, two different loading cases are simulated, each of them for several parameter values μ_3. Those are, in correspondence with the BC1 and BC2 case of Example 2, an applied traction force in x-direction (0°), referred to as BCx, and an applied traction force in z-direction (90°), referred to as BCz. Thus, the muscle is once subject to a uniaxial force and once exposed to a shear loading. In total, $n_p = 9$ simulations served as training data. Those are

BCx: $\mu_3 \in \{5, 10, 25, 50, 75\}\,\mathrm{e}{-3}\,\mathrm{MPa}$, i.e. 5 different values,

BCz: $\mu_3 \in \{1, 5, 10, 15\}\,\mathrm{e}{-3}\,\mathrm{MPa}$, i.e. 4 different values.

Accounting for the zero DIRICHLET boundary conditions, the dimensions of the obtained snapshot matrices are: $\boldsymbol{S}_u, \boldsymbol{S}_s \in \mathbb{R}^{13\,872 \times 9\,009}$, and $\boldsymbol{S}_w \in \mathbb{R}^{729 \times 9\,009}$. Figure 7.53 shows the singular value decays of the subsequently performed SVD on each of the snapshot matrices. Since during the online phase, the different ROM shall simulate the BCxz test case (i.e. with 45° an angle between the BCx and BCz case), the singular value decays obtained with Example 3A, BCxz are added in Figure 7.53 using dashed lines. Interestingly, the singular values of the position and the supremizer POD decay faster for the separate calculation of BCx and BCz.

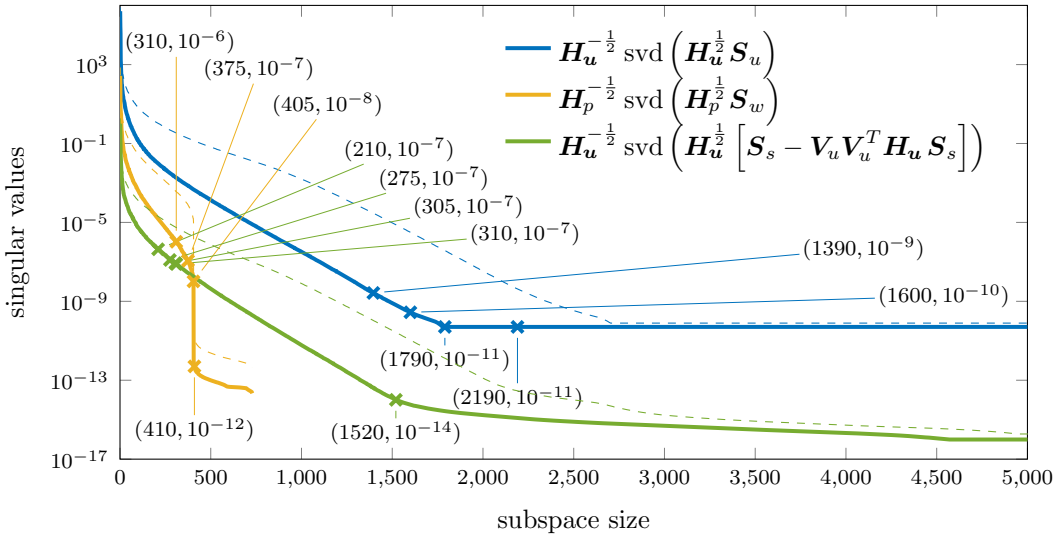


Figure 7.53: *Example 3A, BCx concatenated with BCz:*
The singular value decays of the POD on the supremizer data $\boldsymbol{S}_s$ together with the singular value decays of the position and the pressure data. The supremizer data singular value decays are very similar for each size r_u of the excluded space $\boldsymbol{V}_u$, thus only one case is plotted here. Additionally, the singular value decays for the Example 3A, BCxz are plotted with dashed lines for comparison, since this will be the online scenario to simulate with the constructed ROM.

Online phase - building and simulating different ROM

The range for each of the reduced sizes r_u (r_{us}) and r_p was chosen based on the respective singular value decay. Note that due to the faster decay a lot smaller values for $r_{us} = r_u + r_s$ were chosen here compared to the direct test case BCxz. The values are listed in Table 7.14.

r_{us}	r_p			
	310	375	405	410
r_u	$\{1\,390, 1\,880\}$	$\{1\,325, 1\,815\}$	$\{1\,295, 1\,785\}$	$\{1\,290, 1\,780\}$
r_s	$\{210, 275, 305, 310\}$	$\{275, 340, 370, 375\}$	$\{305, 370, 400, 405\}$	$\{310, 375, 405, 410\}$
r_{us}	$\{1\,600, 1\,665, 1\,695, 1\,700, 2\,090, 2\,155, 2\,185, 2\,190\}$			

Table 7.14: *Example 3A, BCx concatenated with BCz:*
The size combinations for building the ROM used for the comparison of different ratios r_p/r_u on the one hand, and for investigating the effect of adding r_s supremizers to the position and velocity space on the other hand.

The different ROM were tested for $\mu_3 \in \{15, 45\}\,\mathrm{e}{-3}\,\mathrm{MPa}$. The full solutions obtained during the offline phase of Example 3A, BCxz served as reference solutions here. Note that those were obtained with a smaller time step size $\mathrm{d}t = 0.025\,\mathrm{ms}$ and that the online simulation for the test scenario at hand was also conducted using this time step size, while for the generation of the separate training data BCx and BCz, the time step size $\mathrm{d}t = 0.1\,\mathrm{ms}$ was used.

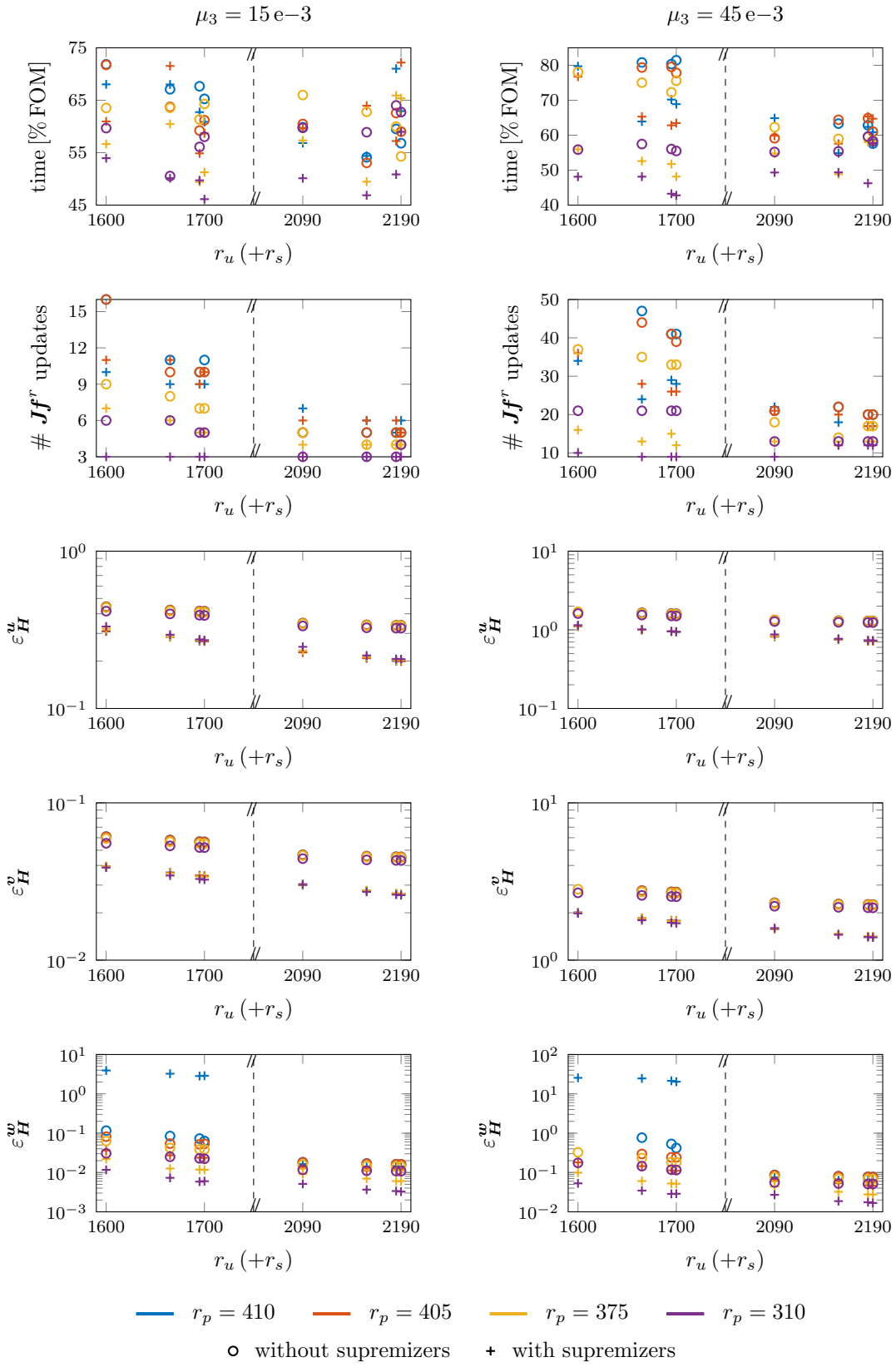


Figure 7.54: *Example 3A, BCxz simulated with a ROM built from training data BCx concatenated with BCz: Comparing the performance of the ROM built with and without supremizer enrichment of the position POD basis for different reduced sizes.*

As before for the direct BCxz scenario, some of the reduced simulations performed with a ROM without supremizer enrichment did not converge, while the ROM of corresponding size, but built with supremizer enriched position and velocity POD basis, did. Here it only occurred for 2 out of the 64 reduced simulations, more specifically for the two larger values r_p in combination with the smallest reduced sizes r_u and the more dynamic load case $\mu_3 = 45\,\mathrm{e}{-3}\,\mathrm{MPa}$. Figure 7.54 shows the results of the converged simulations.
For this example now, every chosen measure speaks in favour of the supremizer enriched ROM. Especially the ROM built with the two smaller reduced sizes r_p in combination with a supremizer enriched POD basis $\boldsymbol{V}_u$ perform better than the rest (see the yellow and purple + in Figure 7.54). The errors for each component $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ are smaller for solutions obtained with a supremizer enriched ROM. However, one should notice that the magnitude of all errors is a lot larger here than for the previous examples and ranges between $1\,\mathrm{e}{-3}$ and $1\,\mathrm{e}{+1}$. Maybe one should additionally look at the magnitude of the relative errors here. Checking the solutions by looking at the video over all time steps confirmed that something reasonable is computed, i.e. that the overall behaviour was captured correctly by the ROM. Thus, it could again mean that the comparison of each node at each single time step is simply an error measure, which is too strict. Comparing these ROM with the ones obtained in Example 3A, BCxz, one should also keep in mind the difference between the chosen range for the reduced size r_u, due to the difference in the singular value decays. While here we have $r_{us} \in [1\,000, 2\,190]$, for Example 3A, BCxz it was $r_{us} \in [2\,475, 3\,110]$. This might explain the difference in the errors of around 4 orders of magnitude. On the other hand, this observation somehow demonstrates again that the singular value decay, i.e. the sum of the remaining singular values, cannot appropriately serve as an a priori error measure for this three field muscle model. Looking at the number of necessary reduced Jacobian updates and the simulation times with respect to the FOM, one can also conclude that the supremizer enriched ROM outperform the ones simply using the position POD basis. The necessary number of Jacobian updates, e.g. for $r_p = 375$ in the $\mu_3 = 45\,\mathrm{e}{-3}\,\mathrm{MPa}$ case, is around 10 to 15 for the supremizer enriched ROM, and between 30 and 40 for other one. This is evidently also reflected in the simulation times.
Lastly, not only focusing on the comparison between ROM built with or without supremizer enriched bases, this example again shows the enormous effect of the ratio between the reduced position size r_u and the reduced pressure size r_p. Increasing the size r_u (with respect to the chosen size r_p) does not only yield an improvement in terms of the errors, but more significantly also in the number of Jacobian updates and the computation times. First of all, those are smaller for larger r_u, but additionally, there is not as much deviation, when varying the size r_p and between the ROM with and without supremizer enrichment. This could be an indication as to greater stability in this case.

8 Summary, discussion and outlook

Using continuum-mechanical models to predict the behaviour of the musculoskeletal system is a useful approach to enhance or support experimental studies. However, it is also a challenging and computationally expensive task. This dissertation makes a valuable contribution towards the aim of providing a stable and fast skeletal muscle model, applicable for complex simulations and investigations. Therefore, the structure of a three-dimensional, dynamic, incompressible and nonlinear skeletal muscle model was derived in detail and thoroughly investigated. Subsequently, the well-established methods of reduced basis approximation and proper orthogonal decomposition were customised to be suitable for the application on the highly complex skeletal muscle model. This way, a significant dimensional reduction of the governing system of differential algebraic equations to around 15 % of the original system size could be achieved, while preserving the original structure and stability. Furthermore, considering that approximating the nonlinear components still remains an open task, an acceptable speedup of nearly 2 was obtained.

The equations suitable for describing the mechanical behaviour of skeletal muscle were reviewed and a reasonably complex continuum-mechanical dynamic skeletal muscle model was set up. To that purpose a brief introduction into constitutive modelling of incompressible, transversely isotropic hyperelasticity was given. The governing equations were discretised in space and time, using the mixed Taylor-Hood finite elements and the implicit Euler scheme, respectively, to make the continuous problem accessible to numerical solution methods. The numerical implementation of the obtained differential algebraic equation system describing three fields, namely the position, the velocity and the hydrostatic pressure, was realised in the Matlab library KerMor. A reasonably stable full-order model (FOM) was obtained, which was investigated from the theoretical and computational point of view. The computational analysis by means of a mesh convergence study was carried out using three different examples with increasing material and geometrical complexity, which each were derived to adequately serve their specific purpose.
Subsequently, the technique of projection-based model order reduction (MOR) was reviewed and the proper orthogonal decomposition (POD) was considered to be an appropriate method to obtain the reduced subspaces. Since the full-order skeletal muscle model consists of three fields, the projection of the system onto a subspace or onto different subspaces is not straightforward. This was first investigated from a theoretical point of view and later demonstrated with computations performed with the derived examples. Therefore, different ROM were built and their performance with respect to stability, efficiency and accuracy were compared. Various combinations and sizes of subspaces, each of them again described by differently calculated POD bases, were tested. The results led to the following conclusions or recommendations for the construction of a stable reduced-order skeletal muscle model: (i) The velocity POD basis has to be chosen equal to the position POD basis, i.e. the same subspace for both fields is required in order to preserve the

structure of the FOM. (ii) For the reduced sizes of the velocity and the position space, $r_v \leq r_u$ has to hold. (iii) The POD bases should be computed optimal in the inner product norm. (iv) The stability of the ROM mainly depends on the ratio between the size of the reduced position space, r_u, and the reduced pressure space, r_p. (v) Enriching the position (and velocity) POD basis with approximate supremizer solutions proved to be beneficial to gain stability.
Following these recommendations, a ROM with a satisfying performance could be constructed. Nevertheless, especially the last two points are not analysed to complete satisfaction, since the performance of the differently constructed ROM was highly problem-specific, i.e. depending on the characteristics of the underlying FOM, and thus exact rules or heuristic criteria could not be established. This leads to an outlook on possible improvements and general suggestions for future work on this topic.

Starting with the full-order model, it would certainly be valuable to conduct a deeper, more problem-specific theoretical analysis of the differential algebraic equation system and resultant requirements on the solution procedure. For example, theoretical work on the inf-sup condition, often available for simpler quasi-static and linear problems, should be revised and written down thoroughly for this more complex problem. Also the employed numerical solution scheme leaves room for improvement. While some different approaches, like an index reduction of the differential algebraic equation system prior to solving it, were already tested and considered ineffective, further methods or algorithms, e.g. operator splitting methods, different time integration schemes, damped NEWTON schemes, or the work of Glowinski & Le Tallec [23], are certainly worth looking at and might reveal larger convergence regions. With respect to increasing the stability of the full-order skeletal muscle model, including the viscous damping term should also have a positive effect. A proper viscous damping term should decrease the oscillations that are clearly visible in the more dynamic test cases. As explained in Chapter 3, this has been omitted in the presented simulations for the sake of reducing the complexity and potential sources of errors. However, looking at the discretised equation system and the Jacobian, i.e. the linear solve, in particular, it should clearly be beneficial. This contribution shows that issues, already occurring in the FOM are inherited by the ROM and might become even more pronounced, which is why it should be emphasised that a profound understanding of the FOM before starting any model reduction procedure is inevitable.
With respect to building a reduced-order skeletal muscle model, this offers various interesting possibilities for future research, which can be divided roughly into the two aims of preserving stability and gaining computational speedup. The former has already been thoroughly investigated in this thesis. As already mentioned during the discussion of the obtained results, further analysing the condition number of the reduced Jacobian matrices for the different POD bases and combinations thereof might give more insight as to which option to favour. Moreover, this could provide an additional means to find rules or heuristic criteria how to choose the ratio between the size of the reduced position space, r_u, and the reduced pressure space, r_p. The application of a greedy snapshot selection, which automatically considers the importance of each snapshot $\boldsymbol{u}$ and $\boldsymbol{v}$ and selects based on certain criteria, which ones to include in the POD, would certainly prove beneficial. Enriching the position and velocity POD basis by approximate supremizers turned out to be a promising approach for improving the stability of the ROM. Again, this can be investigated further, also or especially from the theoretical point of view, based on the

suggested revision of the inf-sup condition for the FOM. Lastly, the issue of computing the reduced initial condition in the case of exclusively choosing the velocity training data $\boldsymbol{S}_v$ for the computation of the position and velocity POD basis, still needs to be solved. Maybe this problem arises due to the oscillations on the one hand and relatively small velocities on the other hand. If that was the case, adding the viscous damping term in the FOM might already offer a remedy by removing the oscillatory response. Additionally, this should be beneficial from a theoretical point of view. The reduced Jacobian contains the term $\boldsymbol{V}_u^T \left[-\boldsymbol{M} + \mathrm{d}t\boldsymbol{D}\right] \boldsymbol{V}_u$, which is close to the identity matrix (multiplied by some scalar factor) for a POD basis that was constructed to be orthonormal in the inner product norm. The latter aim, gaining computational speedup, is strongly related to the nonlinear operations, i.e. the evaluation of the reduced right-hand side function $\boldsymbol{g}^r$ and the reduced Jacobian $\boldsymbol{J}\boldsymbol{g}^r$. Those are still evaluated exactly in this work, thus depending on the dimension of the FOM and impeding a significant computational speedup. Their approximate evaluation, by additionally applying e.g. the ECSW or the DEIM in a customised form, is definitely required in future research. However, from experiences made so far, this additional task is only recommendable, once a stable projected ROM is available. Furthermore, it would also be extremely interesting to look into the structure of those operators in more detail, once one has a particular material model and simulation scenario in mind. Exactly knowing the features and the structure of such an assembled reduced nonlinear operator, one could exploit those to develop specific suitable hyperreduction methods. This could potentially be the microstructurally and homogenisation-based material model by Bleiler et al. [7], where for instance use could be made of the affinity assumption and a resulting additive decomposition of energy contributions of different microstructures.

Bibliography

[1] Antoulas, A. C. & Sorensen, D. C.: Approximation of large-scale dynamical systems: An overview. *International Journal of Applied Mathematics in Computer Science* **11** (2001), 1093–1121, URL http://matwbn.icm.edu.pl/ksiazki/amc/amc11/amc1155.pdf.

[2] Bader, E.; Kärcher, M.; Grepl, M. A. & Veroy, K.: Certified Reduced Basis Methods for Parametrized Distributed Elliptic Optimal Control Problems with Control Constraints. *SIAM Journal on Scientific Computing* (2016).

[3] Ball, J. M.: Convexity Conditions and Existence Theorems in Nonlinear Elasticity. *Archive of Rational Mechanics and Analysis* **63** (1976), 337 – 403.

[4] Ballarin, F.; Manzoni, A.; Quarteroni, A. & Rozza, G.: Supremizer stabilization of POD-Galerkin approximation of parametrized steady incompressible Navier-Stokes equations. *International Journal for Numerical Methods in Engineering* **102** (2015), 1136–1161.

[5] Bathe, K.: The inf-sup condition and its evaluation for mixed finite element methods. *Computers and Structures* **79** (2001), 243 – 252, URL http://web.mit.edu/kjb/www/Principal_Publications/The_Inf-Sup_Condition_and_its_Evaluation_for_Mixed_Finite_Element_Methods.pdf.

[6] Benzi, M.; Golub, G. H. & Liesen, J.: Numerical solution of saddle point problems. *Acta Numerica* (2005).

[7] Bleiler, C.; Ponte Castañeda, P. & Röhrle, O.: A microstructurally-based, multiscale, continuum-mechanical model for the passive behaviour of skeletal muscle tissue. *Journal of the Mechanical Behavior of Biomedical Materials* **97** (2019), 171 – 186.

[8] Bodine, S. C.; Roy, R. R.; Eldred, E. & Edgerton, V. R.: Maximal Force as a Function of Anatomical Features of Motor Units in the Cat Tibialis Anterior. *Journal of Neurophysiology* **57** (1987), URL https://www.physiology.org/doi/abs/10.1152/jn.1987.57.6.1730.

[9] Böl, M.; Ehret, A. E.; Leichensring, K.; Weichert, C. & Kruse, R.: On the anisotropy of skeletal muscle tissue under compression. *Acta Biomaterialia* **10** (2014), 3225 – 3234.

[10] Bonet, J. & Wood, R. D.: *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press 1997.

[11] Braess, D.: *Finite Elemente - Theorie, schnelle Löser und Anwendungen in der Elastizitätstheorie*. Springer 1992.

[12] Brenner, S. C. & Scott, L. R.: *The Mathematical Theory of Finite Element Methods*. Springer 1994.

[13] Brezzi, F.: Stability of Saddle-Points in Finite Dimensions. *Frontiers in Numerical Analysis* (2003).

[14] Carlberg, K.; Barone, M. & Antil, H.: Galerkin v. least-squares Petrov-Galerkin projection in nonlinear model reduction. *Journal of Computational Physics* **330** (2016), 693–734, URL http://www.sciencedirect.com/science/article/pii/S0021999116305319.

[15] Chagnon, G.; Rebouah, M. & Favier, D.: Hyperelastic Energy Densities for Soft Biological Tissues: A Review. *Journal of Elasticity* **120** (2015), 129 – 160, URL https://link.springer.com/article/10.1007%2Fs10659-014-9508-z.

[16] Chaturantabut, S. & Sorensen, D. C.: Nonlinear Model Reduction via Discrete Empirical Interpolation. *SIAM Journal on Scientific Computing* **32** (2010), 2737–2764, URL https://epubs.siam.org/doi/10.1137/090766498.

[17] Courant, R.; Friedrichs, K. & Lewy, H.: On the Partial Difference Equations of Mathematical Physics. *IBM Journal of Research and Development* **11** (1967), 215 – 234.

[18] Dao, T. T. & Ho Ba Tho, M.-C.: A Systematic Review of Continuum Modeling of Skeletal Muscles: Current Trends, Limitations, and Recommendations. *Hindawi Applied Bionics and Biomechanics* **2018** (2018), 1 – 17, URL https://doi.org/10.1155/2018/7631818.

[19] Demiray, H.: A note on the elasticity of soft biological tissues. *Journal of Biomechanics* **5** (1972), 309 – 311.

[20] Ern, A. & Guermond, J.-L.: *Theory and Practice of Finite Elements*, vol. 159 of *Applied Mathematical Sciences*. Springer 2004.

[21] Farhat, C.; Avery, P.; Chapman, T. & Cortial, J.: Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency. *International Journal for Numerical Methods in Engineering* **98** (2014), 625–662, URL https://onlinelibrary.wiley.com/doi/epdf/10.1002/nme.4668.

[22] Fung, Y. C.: *Biomechanics: Mechanical Properties of Living Tissues*. Springer New York 2010, 2nd edn.

[23] Glowinski, R. & Le Tallec, P.: Numerical Solution of Problems in Incompressible Finite Elasticity by Augmented Lagrangian Methods II. Three-Dimensional problems. *SIAM Journal on Applied Mathematics* **44** (1984), 710–733, URL https://www.jstor.org/stable/2101248.

[24] Guerses, E.: *Aspects of Energy Minimization in Solid Mechanics: Evolution of Inelastic Microstructures and Crack Propagation*. Dissertation, University of Stuttgart (2007).

[25] Haasdonk, B.: Convergence Rates of the POD-Greedy Method. *Mathematical Modelling and Numerical Analysis* **47** (2013), 859 – 873, URL https://www.esaim-m2an.org/articles/m2an/pdf/2013/03/m2an120045.pdf.

[26] Haasdonk, B. & Ohlberger, M.: Reduced Basis Method for Finite Volume Approximations of Parametrized Linear Evolution Equations. *Mathematical Modelling and Numerical Analysis* **42** (2008), 277 – 302, URL https://www.esaim-m2an.org/articles/m2an/pdf/2008/02/m2an0672.pdf.

[27] Helfenstein, J.; Jabareen, M.; Mazza, E. & Govindjee, S.: On non-physical response in models for fiber-reinforced hyperelastic materials. *International Journal of Solids and Structures* **47** (2010), 2056 – 2061.

[28] Hesthaven, J.; Rozza, G. & Stamm, B.: *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer 2016.

[29] Holzapfel, G. A.: *Nonlinear Solid Mechanics - A Continuum Approach for Engineering*. John Wiley & Sons, LTD 2000.

[30] Holzapfel, G. A.; Gasser, T. C. & Ogden, R. W.: A New Constitutive Framework for Arterial Wall Mechanics and a Comparative Study of Material Models. *Journal of elasticity and the physical science of solids* **61** (2000), 1 – 48, URL https://link.springer.com/article/10.1023%2FA%3A1010835316564.

[31] Hotelling, H.: Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology* (1933).

[32] Méndez, J. & Keys, A.: Density and composition of mammalian muscle. *Metabolism - Clinical and Experimental* **9** (1960), 184 – 188, URL https://eurekamag.com/research/024/450/024450136.php.

[33] Morrow, D. A.; Haut Donahue, T. L.; Odegard, G. M. & Kaufman, K. R.: Transversely isotropic tensile material properties of skeletal muscle tissue. *Journal of the Mechanical Behavior of Biomedical Material* **3** (2010), 124 – 129.

[34] Oden, J. T. & Le Tallec, P.: On the Existence of Hydrostatic Pressure in Regular Finite Deformations of Incompressible Hyperelastic Solids. In Sternberg, R. L. (ed.): *Nonlinear Partial Differential Equations in Engineering and Applied Science*, 1979, vol. 54.

[35] Ogden, R. W.: Elastic Deformations of Rubberlike Solids. *Mechanics of Solids* (1982), 499 – 537.

[36] Ogden, R. W.: *Non-linear Elastic Deformations*. Dover Civil and Mechanical Engineering 1997.

[37] Quarteroni, A.; Manzoni, A. & Negri, F.: *Reduced Basis Methods for Partial Differential Equations: An Introduction*, vol. 92 of *La Matematica per il 3+2*. Springer 2016.

[38] Reese, S. & Govindjee, S.: Theory of Finite Viscoelasticity and Numerical Aspects. *International Journal of Solids and Structures* **35** (1998), 3455 – 3482, URL https://doi.org/10.1016/S0020-7683(97)00217-5.

[39] Röhrle, O.; Yavuz, U.; Klotz, T.; Negro, F. & Heidlauf, T.: Multiscale modeling of the neuromuscular system: Coupling neurophysiology and skeletal muscle mechanics. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine* **11** (2019), 1 – 43, URL https://doi.org/10.1002/wsbm.1457.

[40] Rozza, G.; Huynh Phuong, D. B. & Manzoni, A.: Reduced basis approximation and a posteriori error estimation for Stokes flows in parametrized geometries: roles of the inf-sup stability constants. *Numerische Mathematik* **125** (2013), 115 – 152.

[41] Rozza, G. & Veroy, K.: On the stability of the reduced basis method for Stokes equations in parametrized domains. *Computational Methods in Applied Mechanical Engineering* **196** (2007), 1244 – 1260, URL https://www.sciencedirect.com/science/article/pii/S0045782506002969?via%3Dihub.

[42] Sansour, C.: On the physical assumptions underlying the volumetric-isochoric split and the case of anisotropy. *European Journal of Mechanics A/Solids* **27** (2008), 28 – 39.

[43] Shamanskii, V. E.: A Modification of Newton's Method. *Ukrainian Mathematical Journal* (1967), 118 – 122.

[44] Sirovich, L.: Turbulence and the Dynamics of Coherent Structure: I, ii and iii. *Quarterly of Applied Mathematics* **45** (1987), 561–590.

[45] Stenberg, R.: On some three-dimensional finite elements for incompressible media. *Computer Methods in Applied Mechanics and Engineering* **63** (1987), 261–269, URL https://doi.org/10.1016/0045-7825(87)90072-7.

[46] Takaza, M.; Moerman, J., K. M. Gindre; Lyons, G. & Simms, C. K.: The anisotropic mechanical behaviour of passive skeletal muscle tissue subjected to large tensile strain. *Journal of the Mechanical Behavior of Biomedical Materials* **17** (2013), 209 – 220.

[47] University of Stuttgart - Institute of Applied Mechanics: Chair of Continuum Mechanics: Vector and Tensor Calculus - An Introduction. Lecture Notes (2012).

[48] Van Loocke, M.; Lyons, G. & Simms, C. K.: Viscoelastic properties of passive skeletal muscle in compression: Stress-relaxation behaviour and constitutive modelling. *Journal of Biomechanics* **41** (2008), 1555 – 1566.

[49] Volkwein, S.: Proper Orthogonal Decomposition: Theory and Reduced-Order Modelling. Lecture Notes (2013), URL http://www.math.uni-konstanz.de/numerik/personen/volkwein/teaching/POD-Book.pdf.

[50] Wang, C. C. & Truesdell, C.: *Introduction to rational elasticity*, vol. 1. Springer 1973.

[51] Wirtz, D.: KerMor: Model order reduction for nonlinear dynamical systems and nonlinear approximation. Software (Last checked 10.12.2019), URL http://www.morepas.org/software/kermor/.

[52] Zohdi, T. I.: *A Finite Element Primer for Beginners - The Basics*. Springer 2018, 2nd edn., URL https://doi.org/10.1007/978-3-319-70428-9.

This thesis investigates the possibility to reduce the computational effort of a dynamic skeletal muscle model making use of model order reduction methods. For that purpose, a three-dimensional, nonlinear, dynamic skeletal muscle model based on the theory of incompressible finite hyperelasticity is introduced. After discretisation in space and time, using the mixed Taylor-Hood finite elements and the implicit Euler scheme, respectively, the obtained complex and high-dimensional differential algebraic equation system describing the three fields position, velocity and pressure, is investigated from a theoretical as well as computational point of view. Furthermore, the stability issues, encountered with a reduced-order model, built by projecting each field of the high-dimensional model onto a reduced subspace, are demonstrated. The reason for these problems is additionally investigated and confirmed from the theoretical perspective. In order to propose a suitable approach for obtaining a stable reduced order skeletal muscle model, the well-established technique of combining the reduced basis approximation with the proper orthogonal decomposition needs to be customised. The performance with respect to stability, efficiency and accuracy of different reduced-order models, built from various combinations and sizes of subspaces, each of them again constructed from differently calculated POD bases, with and without enrichment by approximate supremizer solutions, is compared.

M. Mordhorst

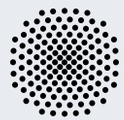
Towards a fast and stable dynamic skeletal muscle model

ISBN 978-3-946412-04-5

CBM-05 (2020)

Towards a fast and stable dynamic skeletal muscle model

Mylena Mordhorst



vorgelegt an der

Universität Stuttgart