Topology optimization of transient fluidic and thermal devices

Theulings, M.J.B.

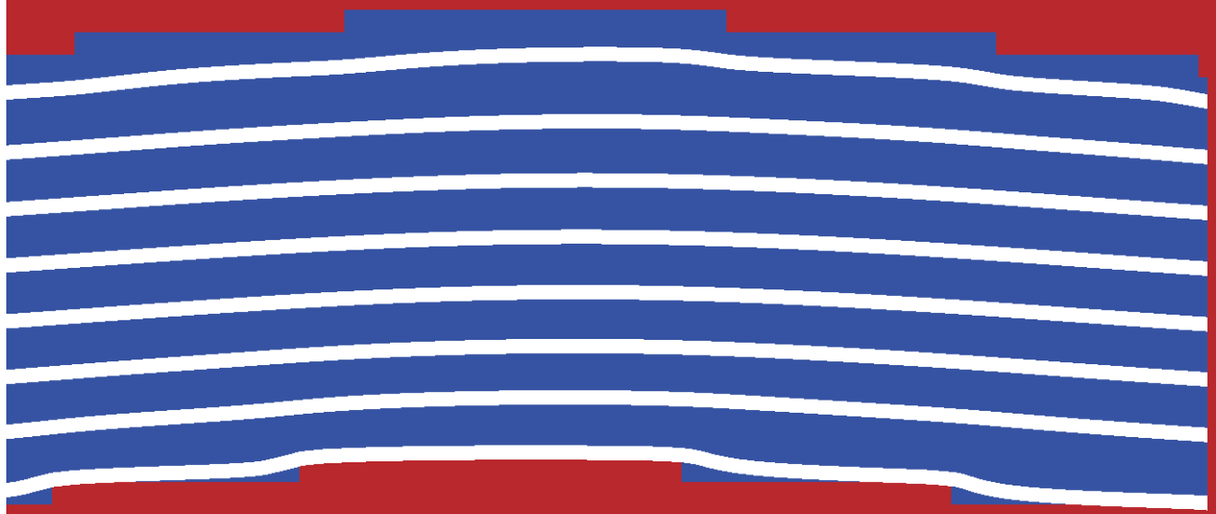
DOI
10.4233/uuid:d89aac66-a799-4309-926b-fe373c431936

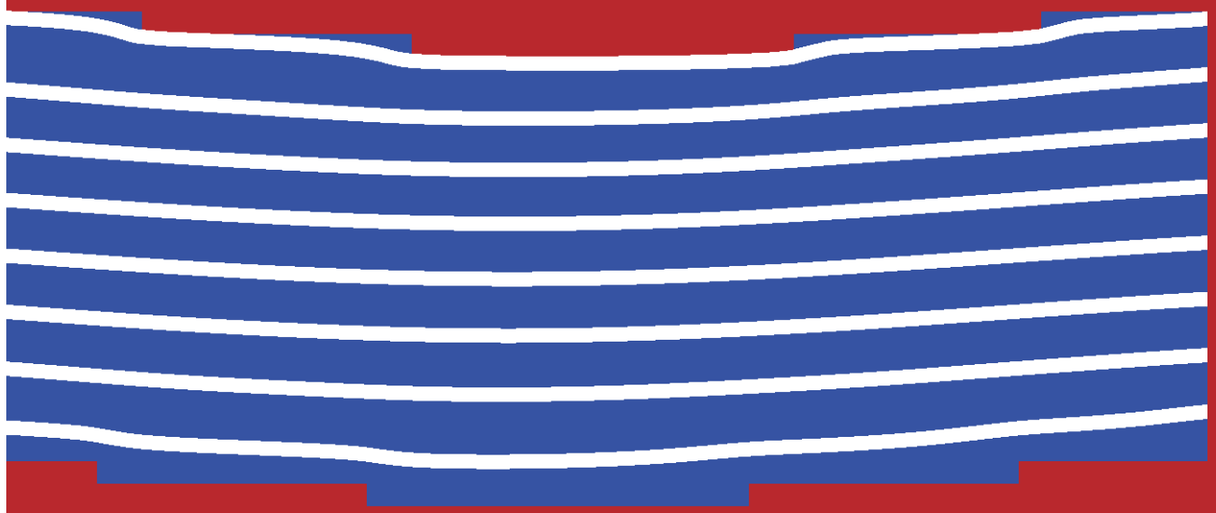
Publication date
2025

Document Version
Final published version

Citation (APA)
Theulings, M. J. B. (2025). *Topology optimization of transient fluidic and thermal devices*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:d89aac66-a799-4309-926b-fe373c431936

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

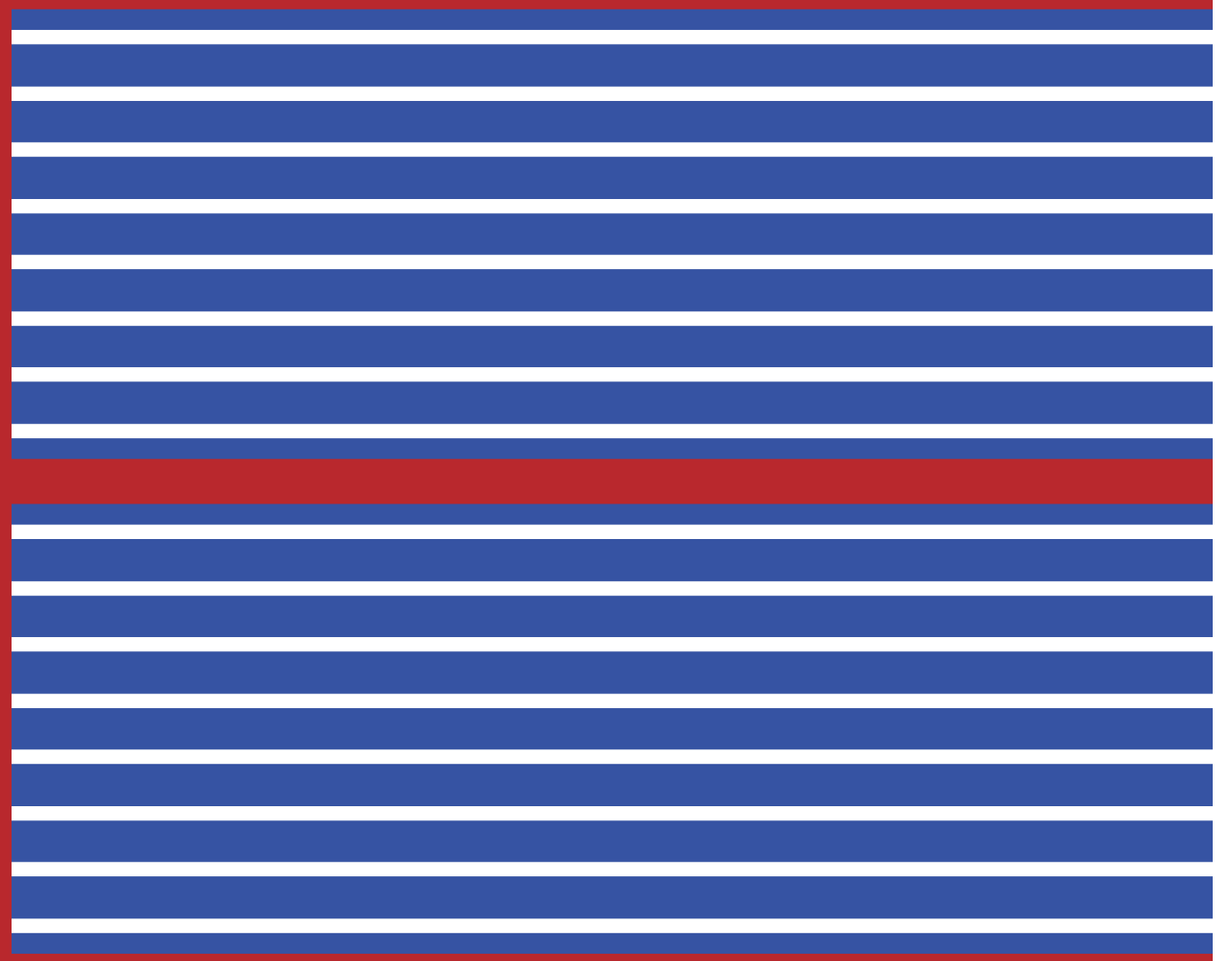




Topology optimization of transient fluidic and thermal devices

Maarten J.B. Theulings

Topology optimization of transient fluidic and thermal devices



Maarten J.B. Theulings

TOPOLOGY OPTIMIZATION OF TRANSIENT FLUIDIC AND THERMAL DEVICES

TOPOLOGY OPTIMIZATION OF TRANSIENT FLUIDIC AND THERMAL DEVICES

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof.dr.ir. T.H.J.J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op maandag 16 juni 2025 om 12:30 uur

door

Maarten Jan Bernard THEULINGS

Werktuigkundig Ingenieur,
Technische Universiteit Delft, Nederland,
geboren te Gouda, Nederland.

Dit proefschrift is goedgekeurd door de promotoren.

Samenstelling promotiecommissie bestaat uit:

Rector Magnificus,	voorzitter
Prof. dr. ir. M. Langelaar,	Technische Universiteit Delft, promotor
Dr. ir. L. F. P. Noel,	Technische Universiteit Delft, copromotor
Onafhankelijke leden:	
Prof. dr. ir. B. J. Boersma	Technische Universiteit Delft
Prof. dr. ir. J. K. Guest,	Johns Hopkins University, United States
Dr. J. Alexandersen,	University of Southern Denmark , Denmark
Dr. R. Picelli,	University of São Paulo, Brasil
Prof. dr. ir. R.A.W.M. Henkes,	Technische Universiteit Delft, reservelid
Overige leden:	
Drs. R. Maas	Koninklijk Nederlands Lucht- en Ruimtevaartcentrum

Drs. R. Maas heeft in belangrijke mate aan de totstandkoming van het proefschrift bijgedragen.

Keywords: Topology optimization, Transient, Thermal, Flow

Printed by: ProefschriftMaken, The Netherlands

Front & Back: Maarten Theulings

ISBN 978-94-6510-701-1

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

SUMMARY

The design of high-performing fluid and thermal devices is crucial for many aerospace applications such as heat exchangers and flow manifolds. These systems often operate under transient conditions, adding another layer of complexity to their design. Conventional design principles have limitations as they depend on engineers to propose the principal structure.

In this dissertation, topology optimization (TO) is investigated as a method to design transient flow and thermal devices. To date, the use of TO for transient flow or thermal problems remains limited to experts in the field. Performing successful optimization heavily relies on the tuning of model and optimization parameters. By systematically investigating and improving the algorithms, their parameters and their characteristics, this thesis provides engineers with guidelines for their use.

For TO of flow problems, where fluid and solid are usually governed by the same equation, the main challenge is the development of an appropriate approach to inhibit flow in the solid. Penalization approaches are often used to sufficiently reduce the flow in the solid, such that accurate flow solutions are found in the fluid. However, excessive penalization can cause early convergence to inferior local optima. Therefore, a balance between flow solution accuracy and design convergence has to be found. The models used for TO of flow problems are investigated using the Volume-Averaged Navier-Stokes (VANS) equations. This study shows that the commonly used Navier-Stokes with Darcy Penalization (NSDP) equations are a simplification of the VANS equations.

To appropriately inhibit flow in the solid, an order analysis is performed on the momentum equations, such that the flow reduction in the solid can be predicted. When using the Darcy penalization, a reliable prediction of the flow reduction is only possible in areas where viscous forces are dominant. In areas with dominant inertial forces, the Forchheimer penalization is needed. The novel Darcy with filtered Forchheimer penalization, which relies on a filtered velocity field, is introduced. This new approach is able to reliably find accurate flow solutions and predict the flow reduction in the solid.

Furthermore, two approaches are given to find accurate flow solutions without increasing the tendency to converge to ill-performing local optima. Both approaches rely on allowing for relatively large flow through porous areas in intermediate designs, while strongly inhibiting the flow in solid areas in converged designs. When larger flow magnitudes are allowed in porous areas, the flow solution is inaccurate but large design changes are observed, conversely, when flow is significantly inhibited in solid areas, the flow solution is accurate yet only small design changes are observed. The first approach relies on the pressure penalization, which has a negligible effect on the flow reduction in gray areas with intermediate design variables. In converged solid areas, the pressure penalization reduces the effect of the pressure gradient on the flow, leading to improved flow reduction. The second approach is a continuation on the flow penalization. A low penalization with large flow through solid areas is used in the earlier stages of optimiza-

tion, while a large penalization with accurate flow is used in the later stages. The continuation approach is made possible by the reliable and problem-independent prediction of the flow reduction, which is used to define appropriate penalization magnitudes for large/low flow reduction.

Lastly, for TO of transient problems, the sensitivity computation may have prohibitively large memory requirements. To tackle this issue, we analyze two state-of-the-art algorithms. The Checkpointing algorithm reduces memory requirements while increasing the computational cost, and the Local-in-Time algorithm reduces memory requirements while introducing sensitivity errors, but does not increase the computational cost. To achieve a better balance between computational cost and sensitivity errors, a hybrid Checkpointing/Local-in-Time algorithm is proposed. To further reduce the computational time, the Parallel-Local-in-Time algorithm is proposed to parallelize computations in the temporal instead of the spatial domain. Finally, guidelines are given to select an appropriate algorithm and find a good compromise between memory requirements, computational cost, and sensitivity errors.

By analyzing approaches for TO of transient flow or thermal problems in detail, and providing new methods and guidelines, this thesis contributes to improving the ease of use of TO. Moreover, the presented approaches are expected to extend to TO problems involving other types of physics.

SAMENVATTING

Het ontwerp van vloeistof- en thermische apparaten is van belang voor veel lucht- en ruimtevaarttoepassingen, zoals warmtewisselaars en stromingssystemen. Deze systemen werken vaak onder transiënte omstandigheden, wat een extra laag complexiteit toevoegt aan hun ontwerp. Conventionele ontwerpprincipes hebben beperkingen, aangezien ze afhankelijk zijn van een ingenieur om het basisconcept te bepalen.

In dit proefschrift wordt topologie-optimalisatie (TO) onderzocht als een methode voor het ontwerpen van transiënte stroming en thermische apparaten. Tot nu toe gebruiken alleen experts TO voor transiënte (warmte)stromingsproblemen. Het succesvol uitvoeren van optimalisatie is sterk afhankelijk van de afstelling van algoritmische parameters. Door de algoritmen, hun parameters en hun kenmerken systematisch te onderzoeken, biedt dit proefschrift richtlijnen voor het gebruik van de algoritmen.

Voor TO van stromingsproblemen, waar de vloeistof en het vaste materiaal vaak met dezelfde vergelijking worden beschreven, is de belangrijkste uitdaging het ontwikkelen van een geschikt model dat stroming in het vaste materiaal verhindert. Penalisatietechnieken worden vaak gebruikt om de stroming in het vaste materiaal voldoende te verhinderen, zodat nauwkeurige stromingsoplossingen in de vloeistof worden gevonden. Overmatige penalisatie kan echter leiden tot convergentie naar inferieure lokale optima. Daarom moet een balans worden gevonden tussen de nauwkeurigheid van de stromingsoplossing en de ontwerpconvergentie. De modellen voor TO van stromingsproblemen worden onderzocht met behulp van de volume-gemiddelde Navier-Stokes (VANS) vergelijkingen. Deze studie toont aan dat de veelgebruikte Navier-Stokes met Darcy-Penalisatie (NSDP) vergelijkingen een vereenvoudiging zijn van de VANS-vergelijkingen.

Om de stroming in het vaste materiaal op een gepaste manier te verhinderen wordt een orde-analyse uitgevoerd op de momentum vergelijkingen, zodat de stromingsreductie in het vaste materiaal kan worden voorspeld. Met alleen de Darcy-penalisatie is een betrouwbare voorspelling van de stromingsreductie alleen mogelijk in gebieden waar viskeuze krachten dominant zijn. In gebieden met dominante inertiële krachten is de Forchheimer-penalisatie nodig. De nieuwe Darcy met gefilterde Forchheimer-penalisatie, die afhankelijk is van een gefilterd stromingsveld, wordt geïntroduceerd. Deze nieuwe benadering is in staat om betrouwbaar stromingsoplossingen te vinden, en de stromingsreductie in het vaste materiaal te voorspellen.

Daarnaast worden twee technieken gepresenteerd om nauwkeurige stromingsoplossingen te vinden zonder dat de neiging om naar inferieure lokale optima te convergeren vergroot wordt. Beide technieken zijn gebaseerd op relatief grote stroming door poreuze gebieden in tussentijdse ontwerpen, terwijl de stroming in het vaste materiaal in geconvergeerde ontwerpen sterk wordt verhinderd. Wanneer veel stroming door poreuze gebieden wordt toegestaan is de stromingsoplossing onnauwkeurig, maar verandert het ontwerp gemakkelijk. Wanneer de stroming in het vaste materiaal aanzienlijk wordt ver-

hinderd is de stromingsoplossing nauwkeurig, maar verandert het ontwerp niet gemakkelijk. De eerste techniek bereikt dit met behulp van een verminderde drukgradiënt, die een verwaarloosbare bijdrage levert in grijze gebieden, terwijl de invloed van de drukgradiënt op de stroming in het vaste domein wordt verminderd. De tweede techniek is een geleidelijk toenemende stromingspenalisatie: aan het begin van de optimalisatie wordt weinig penalisatie met veel stroming door het vaste materiaal gebruikt, en aan het einde een hoge penalisatie met nauwkeurige stromingsoplossing. Deze techniek wordt mogelijk gemaakt door de betrouwbare en probleem-onafhankelijke voorspelling van stromingsreductie, die wordt gebruikt om de penalisatiehoogte te definiëren voor hoge/lage stromingsreductie.

Tot slot kunnen in TO van transiënte problemen de berekening van afgeleiden leiden tot belemmerend grote geheugeneisen. Om dit probleem aan te pakken analyseren we twee state-of-the-art algoritmen. Het Checkpointing-algoritme vermindert de geheugenbehoefte terwijl het de rekenkosten verhoogt, en het Local-in-Time-algoritme vermindert de geheugeneisen terwijl het fouten in de afgeleiden introduceert, maar de rekenkosten niet verhoogt. Om een beter evenwicht te bereiken tussen rekenkosten en nauwkeurigheid van de afgeleiden, wordt een hybride Checkpointing/Local-in-Time-algoritme voorgesteld. Om de rekentijd verder te verminderen, wordt het Parallel-Local-in-Time-algoritme voorgesteld om berekeningen in het temporele in plaats van het ruimtelijke domein te parallelliseren. Ten slotte worden richtlijnen gegeven om een geschikt algoritme te selecteren en een goed evenwicht te vinden tussen geheugenbehoefte, rekenkosten en nauwkeurigheid van de afgeleiden.

Door de methodes voor TO van transiënte (warmte)stromingsproblemen in detail te analyseren en nieuwe methoden en richtlijnen aan te bieden, draagt dit proefschrift bij aan de verbetering van het gebruiksgemak van TO. De verwachting is bovendien dat de gepresenteerde benaderingen uitbreidbaar zijn naar TO-problemen waar andere soorten fysica een rol spelen.

CONTENTS

1

INTRODUCTION

1.1. BACKGROUND

Flow and thermal phenomena play an important role in many engineering problems found in aerospace applications. For instance, electrification of planes is presently an important research direction for the reduction of carbon emissions in aviation industry. During takeoff high electric loads cause significant heat generation in batteries and well-designed heat exchangers are required to cool the system. Additionally, the heat generation during takeoff is of a sudden nature and transient effects have to be taken into account in designing an appropriate cooling system. Other examples are the aerodynamic heat generation in space vehicles during reentry, or drag minimization and lift maximization of flying objects. Moreover, since additional weight also increases the energy required for aerospace vehicles to ascend, all these thermal and flow systems should add a minimum amount of weight. In an industry as large as aerospace, it is of importance to find high-performing solutions to such transient fluidic and thermal problems.

Basic design principles for transient fluidic and thermal problems often suffice, but have their limitations. The most common types of heat exchangers and heat sinks such as double-pipe, shell-and-tube, plate heat exchangers, and pin or finned heat sinks are easy and cheap to manufacture but show relatively low performance. Increasing the performance is often done through sizing or shape optimization. Shape optimization is also extensively used in designing airfoils for maximum lift and minimum drag. These approaches are limited as they can only find the optimal shape for a given design. Although these methods are useful, they are not able to generate novel design concepts which remains the sole domain of the human engineer.

1.2. RESEARCH CHALLENGES

To improve the performance of transient fluidic and thermal devices and create innovative designs, novel approaches are required. A well-known high-performing approach to generate designs is density-based topology optimization (TO) (Bendsøe & Sigmund, 2004). Using TO, an optimal material distribution within a design domain can be deter-

mined numerically, often leading to non-intuitive high-performance devices. Originally, TO was introduced to design lightweight but stiff structures. However, the applications of TO have branched out into many other fields, as shown by the review papers on fluid flow TO by Alexandersen and Andreasen (2020) and on TO for heat transfer by Dbouk (2017). TO is thus a promising tool for designing novel devices for transient fluidic and thermal problems.

Although TO shows a lot of potential, its practical use in industry remains relatively rare. In the author's opinion, setting up the model and algorithm appropriately is one of the major challenges of current methods for TO. Before a structure can be successfully optimized, model and optimization parameters often need to be tuned. First, the engineer has to select interpolation functions for the material parameters and filters to regularize the design. Secondly, the engineer selects appropriate parameters to control the shape of these functions and filters. Parameters are often subject to continuation approaches which ensure design flexibility in earlier stages of the optimization procedure, and a crisp solid/fluid design in later stages. The tuning of these parameters and strategies is often a time intensive process, that relies heavily on the experience of the engineer. To spread the usage of TO, it is important to provide informed guidelines on how to approach transient thermal and flow problems, and to develop formulations that allow for a predictable, systematic approach.

Some areas of transient thermal and fluid TO remain underdeveloped. For density-based TO of flow problems, the accuracy of the flow solution and numerical stability of the optimization algorithm are often conflicting (Kreissl & Maute, 2012). Reliable models which balance these properties are needed to find optimal topologies. Moreover, when Reynolds numbers increase and inertia becomes dominant, finding reliable parameter settings for flow TO becomes even more challenging. Although many real-life problems are time dependent, transient TO is scarcely addressed for flow problems (Alexandersen & Andreasen, 2020). There are two main reasons behind the scarcity of work on this topic. Firstly, frameworks for static TO are not readily extended to transient TO (Kristiansen & Aage, 2022). Secondly, sensitivity analysis for transient TO drastically increases time and memory requirements, which can be reduced to some extent using specialized algorithms (Griewank, 1992; Yamaleev et al., 2010). However, reducing memory requirements comes at the cost of increased computational time or introducing errors in the optimization procedure.

1.3. RESEARCH AIMS

The main goals of this thesis revolve around the aforementioned challenges in density-based TO of transient flow or thermal problems. We address three specific obstacles that hinder the use of TO for aerospace problems, and aim to:

- **Improve the understanding of flow models for density-based TO:** To improve their usability, we examine models for solid/fluid TO in Chapter 2. The aim is to derive a generalized model based on the Volume Averaged Navier-Stokes (VANS) equations for porous flow, and inspect its adaptation for solid/fluid TO. In fact, most common models used in density-based TO constitute a simplification of the VANS model. By investigating the VANS model, we find new insight into its use for

solid/fluid TO, which can be leveraged for our second aim.

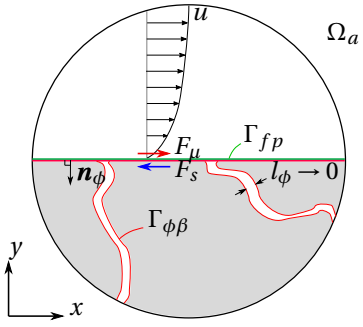
- **Construct a reliable approach for density-based TO of flow problems:** Parameter tuning is often a time consuming task in TO of flow problems. We aim to provide users with an intuitive and informed parameter selection and continuation strategy in Chapters 2 and 3. The strategy should be less likely to converge to inferior local optima than the state of the art, while maintaining accuracy of the solution. Moreover, the approach in Chapter 3 should be reliable for both viscous and inertia dominated flows.

- **Balance time and memory requirements in TO of transient thermal and flow problems.** Reducing memory requirements in TO of transient problems comes at the cost of increased computational time or errors in the optimization procedure. To find a trade-off, three state-of-the-art methods, and their extension into two novel approaches, are investigated in Chapter 4. In one of the novel methods, we parallelize by decomposing the time domain instead of the spatial domain. Selecting the appropriate algorithm is problem dependent. We provide guidelines to choose an appropriate algorithm for reducing time and/or memory requirements in TO of transient problems.

Beside these specific contributions, we aim for this work to illustrate a more general approach to reliably construct models for density-based TO. The approach should be systematic and result in a predictable behavior of the optimization procedure. Moreover, the approach should give insight into a TO problem, such that ease of use is improved. In Chapter 5, we assess the achievement of the three challenges to improve the ease of use, and reflect on the general approach presented in this work.

2

APPROACHES FOR LAMINAR FLOW TOPOLOGY OPTIMIZATION



In this chapter, methods to adapt the Navier-Stokes equations for TO are critically examined. Volume averaging is used to construct a set of flow equations for density-based TO. Insight is given into the balance between model accuracy and optimization convergence behavior, resulting in guidelines for appropriate parameter selection.

This chapter is based on the publication in Structural and Multidisciplinary Optimization 66(6), Theulings et al. (2023).

Towards improved porous models for solid/fluid topology optimization

2

Abstract *Modeling of fluid flows in density-based topology optimization forms a longstanding challenge. Methods based on the Navier-Stokes equations with Darcy penalization (NSDP equations) are widely used in fluid topology optimization. These methods use porous materials with low permeability to represent the solid domain. Consequently, they suffer from flow leakage in certain areas. In this work, the governing equations for solid/fluid density-based topology optimization are reevaluated and reinterpreted. The governing equations are constructed using the volume averaged Navier-Stokes (VANS) equations, well known in the field of porous flow modeling. Subsequently, we simplify, interpret and discretize the VANS equations in the context of solid/fluid topology optimization, and analytically derive lower bounds on the Darcy penalization to sufficiently prevent flow leakage. Based on both the NSDP and VANS equations, two flow solvers are constructed using the Finite Volume method. Their precision and the lower bound on the Darcy penalization are investigated. Subsequently, the solvers are used to optimize flow channels for minimal pressure drop, and the resulting designs and convergence behavior are compared. The optimization procedure using the VANS equations is found to show less tendency to converge to inferior local optima for more precise flow solutions and is less sensitive to its parameter selection.*

2.1. INTRODUCTION

Optimization of flow related problems is a challenging yet highly relevant subject. Topology optimization has been successfully applied to such problems as can be found in the extensive literature survey by Alexandersen and Andreasen (2020). In fluidic topology optimization, the two most popular approaches are density-based and level-set based optimization. In the first work on density-based fluidic topology optimization by Borrvall and Petersson (2003) the distinction between the fluid and solid parts of the design domain is introduced using an inverse permeability. They optimize 2D channel flow between two plates where in the solid domain the two plates are close to eachother, resulting in low permeability and limited flow. Low permeability is modeled by adding a high penalization on the flow. In the fluid domain the two plates are further apart, permeability is high and only a low penalization on the flow is added. Design variables control a penalization on the flow, and thus influencing permeability of the domain. This approach leads to a set of governing equations that combines the Darcy flow problem with the Stokes equations and is only suitable for low Reynolds flow. Subsequently, Gersborg-Hansen et al. (2005) extend this framework from Stokes to Navier-Stokes flow by including inertia terms. Furthermore, they note that the reference to flow between two plates can be dropped and replaced by flow through a porous medium modeled by a Brinkman-type model (Brinkman, 1949). We will refer to the resulting set of equations

as the Navier-Stokes equations with Darcy penalization (NSDP equations).

Kreissl and Maute (2012) found that density-based optimization using the NSDP equations required specific stabilization procedures and solutions suffered from erroneous *"pressure diffusion"* trough the solid domains. Kreissl and Maute (2012) use pressure diffusion to refer to pressure gradients in the solid domain which drive erroneous flow in these domains. In this work we will refer to this effect as *"flow leakage"*, as we will argue that fluid pressure gradients in the porous "solid" domain should be expected and flow through this domain is representative of both flow and pressure field errors in the fluid domain. As a solution to these density-based optimization problems, Kreissl and Maute (2012) propose to use level-set based optimization using X-FEM. In contrast, level-set based fluid optimization allows for crisp solid/fluid boundaries and rigorously diminishes spurious flow through solid parts of the design domain. Provided, if combined with a suitable modeling approach (Kreissl and Maute (2012) use X-FEM). There is however no such thing as a free lunch as resulting designs may become more dependent on the initial guess of the level-set function, as shown by Allaire et al. (2004). However, topological derivatives can be used to measure the effect of adding solid islands in the fluid domains, alleviating the influence of the initial design on the optimum (Challis & Guest, 2009; Guillaume & Idris, 2004). A disadvantage of current topological derivatives is that only solid material can be added in the flow domain, but the effect of creating a channel between two separate flow domains cannot be assessed using topological derivatives to the best of the authors' knowledge. The pressure gradients and flow leakage in the solid design domain in density-based optimization can thus also be seen as an advantage. They distribute sensitivities throughout the entire design domain and carry information on the effect of creating a channel between two separate flow domains as noted by Alexandersen and Andreasen (2020). However, similar to level set-based methods, density-based methods will also converge to local optima dependent on initial design and optimization parameters. Although the more restricted design modifications of the level-set approach result in a stronger dependence on the initial design.

Methods which blend features of level set-based and density-based approaches are proposed by H. Li et al. (2022), Picelli et al. (2022) and Behrou et al. (2019). To attain sensitivity information in the solid domain H. Li et al. (2022) optimize a topology based on a level set function while representing the solid domain as highly impermeable porous material. They thus do not enforce no flow penetration through the solid/fluid interface but inhibit solid domain flow using a penalization. The biggest advantage of this method is the fact that a body-fitted mesh may be used to accurately capture surface effects and no continuation approach is required for the flow penalization. However, using the level set approach other optimization parameters are introduced which also influence design convergence. Picelli et al. (2022) represent the topology using a crisp interface and explicitly prescribe no-penetration conditions on the solid/fluid interface such that no flow is present in the solid domain. Moreover, sensitivities in the fluid domain are computed by assuming porous material and a flow penalization may be introduced in the fluid domain (but in practice porous regions are not present in the flow computation). Subsequently, spatial filtering is used to populate the solid regions with sensitivities. However, the extrapolation of sensitivities into the solid regions is dependent on a spatial filtering radius and therefore parts of the solid domain will not contain

any sensitivities. Similarly, Behrou et al. (2019) use a density based model and remove flow in the solid domain by adaptive removal/addition of elements with a density below a certain threshold value. Both these last two methods have the additional advantage of reducing the computational cost of solving the physics. However, a disadvantage of these methods is that sensitivity information is also removed in the large solid areas. Consequently, the effect of adding channels in solid domains cannot be measured, creating difficulties for the optimizer to add these kind of design features. Both level-set and density-based flow optimization thus have their advantages and disadvantages. In this paper we use density-based fluidic topology optimization as we prefer its natural ability to add islands to the fluid domain and create channels in the solid domain. Furthermore, we aim to improve the flow model such that errors caused by flow leakage are reduced.

Several authors have already studied variations on the porous flow model to increase its precision and reduce flow leakage. A mixed formulation of the Darcy-Stokes equations is implemented by Guest and Prévost (2006), and they note the similarity between their flow model and the Brinkman equation for flow through multiple scale porous media. Contrary to Borrvall and Petersson (2003) who find some intermediate density elements at the solid/fluid boundaries of their optimal designs due to a density filter which is applied to prevent convergence to inferior local optima, Guest and Prévost (2006) find crisp solid/fluid optimal designs. They argue that their methods automatically converge to discrete valued designs and apply a continuation strategy on the maximum flow penalization and a procedure of smoothing/projecting the design variables to prevent convergence to inferior local optima. Furthermore, they note the possibility to use the new method to optimize porous/fluid structures. However, the mixed Darcy-Stokes equations only consider Stokes flow in the fluid domain and do not include the inertial effects in the Navier-Stokes equations. Philippi and Jin (2015) and Alonso and Silva (2022) extend the standard NSDP equations by including a Forchheimer permeability tensor besides the Darcy penalization to penalize flows. The Forchheimer permeability tensor, derived in detail by Whitaker (1996), adds a porous drag at higher Reynolds numbers which scales quadratically with flow velocities. Alonso and Silva (2022) found improved designs when the Forchheimer tensor was included.

Further extensions to the NSDP equations can be found in the optimization of permeable microstructures. Governing equations in these optimizations allow for flow through the porous domain and are interesting to investigate from the perspective of solid/fluid optimization where the porous domain approaches a solid. A unit cell within a larger periodic porous medium is optimized in micro scale using a standard Brinkman type model by Guest and Prévost (2007). Subsequently, a more refined Darcy penalization is computed for the NSDP equations using homogenization techniques similar to the techniques which will be used in this work. Takezawa et al. (2020) extend this work by also computing the Forchheimer permeability tensor. Furthermore, Michaël et al. (2020) use the method of *volume averaging* to derive state equations for the modeling and optimization of spatially varying porous media. Volume averaging is a well-known technique in the field of flow modeling, used to construct refined porous flow models. Moreover, by including a high flow penalization in the porous domain, distinct solid/fluid designs are found in the results computed by Michaël et al. (2020). Extended and more accurate porous flow models have thus been investigated in the context of topology optimization.

However, these techniques have not been applied to the construction and interpretation of flow models particularly for solid/fluid topology optimization.

In most previous topology optimization studies, porous flow models based on the Darcy or Brinkman equations were used to define solid/fluid parts in the design domain. Robustly decreasing flow leakage and finding correct interpolation functions for material properties remains a challenge in fluidic topology optimization (Alexandersen & Andreasen, 2020). However, as discussed in the previous paragraphs, more extensive porous flow models exist and can be used in topology optimization (Alonso & Silva, 2022; Michaël et al., 2020). To improve robustness of the optimization procedure and gain a better understanding of the equations for fluidic topology optimization, we approach the porous flow model from a more general viewpoint. An extended set of governing equations is investigated which we interpret and discretize specifically for solid/fluid topology optimization. Particularly, we will use the concept of volume averaging to derive the volume averaged Navier-Stokes (VANS) equations following Whitaker (1969, 1996). By closely examining and interpreting the VANS equations, we aim to improve two of the challenges set by Alexandersen and Andreasen (2020): improve "*precision of solution and/or optimality*", improve "*parameter robustness and algorithmic stability*".

In topology optimization we want to divide a design domain into a fluid and solid (impermeable porous) part using a single set of continuous governing equations to represent the flow everywhere within the design domain. The governing set of equations should thus be able to accurately capture both flow near the solid/fluid interface as well as in the solid and fluid domains. Accurately modeling of flows near porous/fluid interfaces has long been a subject of research, and is relevant for optimized flow structures where the porous/fluid interface represents a solid/fluid interface. A boundary condition was proposed by Beavers and Joseph (1967) to account for a jump in stress at the interface and to couple the Stokes to Poiseuille flow in a porous/fluid channel. Ochoa-Tapia and Whitaker (1995) propose a momentum jump condition for the interface based on the VANS equations to couple the Brinkman flow model to the Stokes flow model. Many authors discussed the nature of the jump in stress at the porous/fluid interface, continuity of stress, velocity and pressure, and appropriate governing equations/boundary condition (Goyeau et al., 2003; Nield, 1991; Vafai & Kim, 1995; Valdés-Parada et al., 2007). More recent studies by Breugem and Boersma (2005) and Hernandez-Rodriguez et al. (2022) compared pore scale simulations with volume averaged simulations and found matching results when the VANS equations were used. Furthermore, Hernandez-Rodriguez et al. (2022) confirm the necessity to include the Brinkman corrections to accurately model flow at the porous/fluid interface. In this work we will draw inspiration from these papers to be able to accurately capture stresses at the solid/fluid interface, and use volume averaged equations to improve solution precision.

Summarizing, the main contributions of this paper are the following:

1. For a solid/fluid topology optimization problem a refined flow model is constructed by investigating the limit case of the VANS equations where porous material represents a solid. This method improves design convergence for optima with similarly precise flow solutions as optima obtained using the NSDP equations.

2. Lower bounds on the Darcy penalization will be derived analytically such that flow

leakage in the solid domain is limited, improving parameter robustness of the optimization problem.

This paper is structured as follows: Section 2.2 gives an introduction to volume averaging techniques for self-containedness, shows the VANS equations and compares them to the standard NSDP equations. Section 2.3 presents the discretization of the VANS equations, and the interpolation function used to dicretize the Darcy penalization. Subsequently, we derive lower bounds on the Darcy penalization in Section 2.4, and make an *a priori* estimation of flow leakage. In Section 2.5 the optimization problem and a method for adjoint sensitivity computations are presented. In Section 2.6 we compare precision of the flow solution based on the VANS and NSDP equations for a range of Darcy penalizations and Reynolds numbers. Section 2.7 performs structural optimization using both the VANS and NSDP equations and compares the resulting structures in terms of precision and objective. Finally, in Section 2.8, we draw conclusions on the use of the VANS equations for topology optimization and identify subjects for further research.

2.2. THE VOLUME AVERAGED NAVIER-STOKES EQUATIONS

In this section a recap of the derivation of the VANS equations following Ochoa-Tapia and Whitaker (1995) and Whitaker (1996) is given, such that we are able to appropriately implement them in topology optimization. Subsequently, the limits of the VANS equations and their suitability for solid/fluid optimization are investigated. Moreover, the VANS equations are simplified and used to derive the NSDP equations.

2.2.1. A SHORT INTRODUCTION TO THE VOLUME AVERAGE

The aim of fluidic topology optimization is to divide design domain Ω into a solid and a fluid domain such that an optimal material layout is found for a certain objective and set of constraints. For density-based solid/fluid optimization, we simulate the solid domain as an impermeable porous domain and simulate flow using the VANS equations. The concept of a volume average is introduced using the porous and fluid domains Ω_p and Ω_f respectively, as shown in Figure 2.1. The domains have interface $\Gamma_{fp} = \overline{\Omega}_f \cap \overline{\Omega}_p$, where $\overline{\square}$ denotes the closure of a domain. For each $\boldsymbol{x}$ an averaging domain Ω_a is defined as depicted in Figure 2.2. The averaging domain is centered around $\boldsymbol{x}$ where vector $\boldsymbol{y}$ points to locations within the averaging domain. Furthermore, it has volume V and contains solid β and fluid ϕ. Consequently, the domain can be split into its fluid part (Ω_ϕ with volume V_ϕ) and solid part (Ω_β with volume V_β) as $\Omega_a = \Omega_\phi \cup \Omega_\beta$, and the interface between these domains is defined as $\Gamma_{\phi\beta} = \overline{\Omega}_\phi \cap \overline{\Omega}_\beta$. To find the average of property Ψ in fluid phase ϕ at coordinates $\boldsymbol{x}$ the intrinsic ($\langle\Psi\rangle^{i\phi}_{\boldsymbol{x}}$) and superficial ($\langle\Psi\rangle^{s\phi}_{\boldsymbol{x}}$) volume averages are defined as:

$$\langle\Psi\rangle^{i\phi} = \frac{1}{V_\phi}\int_{\Omega_\phi}\Psi(\boldsymbol{x}+\boldsymbol{y})d\Omega, \qquad \langle\Psi\rangle^{s\phi} = \frac{1}{V}\int_{\Omega_\phi}\Psi(\boldsymbol{x}+\boldsymbol{y})d\Omega, \tag{2.1}$$

where $\boldsymbol{y}$ is used to integrate over fluid domain Ω_ϕ centered around $\boldsymbol{x}$. Both the intrinsic and superficial averages are thus field quantities dependent on coordinate $\boldsymbol{x}$, for convenience the subscript $\boldsymbol{x}$ is omitted in the remainder of this work. Using the superficial

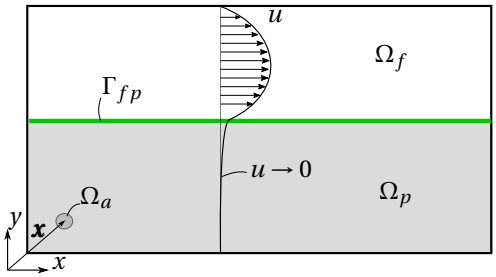


Figure 2.1: Design domain Ω divided into a fluid domain Ω_f and a porous domain Ω_p, both domains are connected at boundary $\Gamma_{fp} = \overline{\Omega}_f \cap \overline{\Omega}_p$. Centered around all $\boldsymbol{x}$ an averaging domain Ω_a is defined as illustrated in Figure 2.2.

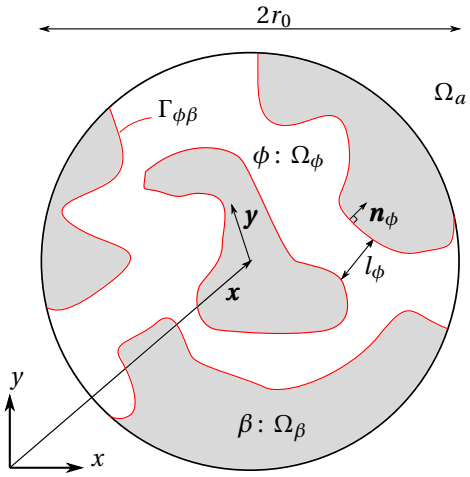


Figure 2.2: An averaging volume centered around $\boldsymbol{x}$, containing solid phase β and fluid phase ϕ. The averaging domain Ω_a can be divided into two parts as $\Omega_a = \Omega_\phi \cup \Omega_\beta$, where the interface between the two parts is $\Gamma_{\phi\beta} = \overline{\Omega}_\phi \cup \overline{\Omega}_\beta$ at which a normal $\boldsymbol{n}_\phi$ is defined pointing to phase β. The porous microstructure has characteristic length l_ϕ, and the averaging domain has characteristic length r_0. Vector $\boldsymbol{y}$ is used to integrate over an averaging domain fixed at location $\boldsymbol{x}$.

volume average the volume fraction of phase ϕ is defined as:

$$\alpha_\phi = \langle 1 \rangle^{s\phi} = \frac{1}{V}\int_{\Omega_\phi} 1 d\Omega = \frac{V_\phi}{V}, \tag{2.2}$$

which is used to relate the two averages as $\langle \Psi \rangle^{s\phi} = \alpha_\phi \langle \Psi \rangle^{i\phi}$. The difference in interpretation between the superficial and intrinsic averages can be explained using the volume fraction. The superficial average represents the bulk average and should generally converge to zero as $\alpha_\phi \to 0$, whereas the intrinsic average represents the pore scale average within the fluid domain and does not necessarily converge to zero as $\alpha_\phi \to 0$. For example, if within a certain averaging volume the fluid has constant pressure $p = p_c$ and the volume fraction approaches zero, the intrinsic average will be $\langle p \rangle^{i\phi} = p_c$, but the superficial average will also approach zero $\langle p \rangle^{s\phi} = \alpha_\phi \langle p \rangle^{i\phi} \to 0$.

Subsequently, some useful mathematical relations are defined. The volume average of a gradient can be simplified using the averaging theorem (Howes & Whitaker, 1985):

$$\langle \nabla \Psi \rangle^{s\phi} = \nabla \langle \Psi \rangle^{s\phi} + \frac{1}{V}\int_{\Gamma_{\phi\beta}} \Psi \boldsymbol{n}_\phi d\Gamma_{\phi\beta}, \tag{2.3}$$

where $\Gamma_{\phi\beta}$ is the interface between phases ϕ and β within Ω_a and $\boldsymbol{n}_\phi$ is the unit normal pointing outward of phase ϕ on this interface, as shown in Figure 2.2. Moreover, the averaging theorem in combination with the definition of the volume fraction can be used to prove that:

$$\nabla \alpha_\phi = -\frac{1}{V}\int_{\Gamma_{\phi\beta}} \boldsymbol{n}_\phi d\Gamma_{\phi\beta}. \tag{2.4}$$

Finally, we assume that fluid field quantities can be split into their averaged and a deviational part as:

$$\Psi = \langle \Psi \rangle^{i\phi} + \tilde{\Psi}, \tag{2.5}$$

where $\tilde{\Psi}$ is the deviation from the average which is small compared to $\langle \Psi \rangle^{i\phi}$. Furthermore, for the averages we assume that:

$$\begin{aligned} \langle \langle \Psi \rangle^{i\phi} \rangle^{i\phi} &= \langle \Psi \rangle^{i\phi}, \\ \langle \langle \Psi \rangle^{i\phi} \rangle^{s\phi} &= \langle 1 \rangle^{s\phi} \langle \Psi \rangle^{i\phi} = \alpha_\phi \langle \Psi \rangle^{i\phi}. \end{aligned} \tag{2.6}$$

For these approximations to be valid for the pressure or velocity of a fluid, separation of scales is required (Whitaker, 1969). If l_ϕ and r_0 are the characteristic length of the porous microstructure and averaging volume, respectively, as shown in Figure 2.2, separation of scales for quantity Ψ requires that:

$$l_\phi \ll r_0 \ll L_\Psi, \tag{2.7}$$

where L_Ψ is a characteristic length of property Ψ defined by:

$$\mathcal{O}\left(\frac{\partial^2 \langle \Psi \rangle^{i\phi}}{\partial x_i \partial x_j}\right) = \mathcal{O}\left(\frac{\langle \Psi \rangle^{i\phi}}{L_\Psi^2}\right). \tag{2.8}$$

These conditions ensure that Equation 2.6 holds and may furthermore be used to define the average of the deviation as:

$$\langle \tilde{\Psi} \rangle^{i\phi} = 0. \tag{2.9}$$

Using these tools and properties, the VANS equations can be defined in a meaningful manner, where only relatively simple closure relations are required to solve the resulting equations. We furthermore note that the introduced concepts for volume averaging show many similarities with the homogenization concepts for topology optimization by Hassani and Hinton (1998), and the averaging concepts for turbulent flow as shown by Alfonsi (2009).

2.2.2. DERIVATION OF THE VANS EQUATIONS

In this section, the general VANS equations are presented, such that they can be interpreted and simplified for topology optimization in the coming sections. Whitaker (1996) and Ochoa-Tapia and Whitaker (1995) derive the VANS equations starting from the incompressible Navier-Stokes equations, consisting of the continuity equation and momentum equations respectively:

$$\begin{aligned} \nabla \cdot \mathbf{v} &= 0, \\ \rho \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) &= -\nabla p + \mu \nabla^2 \mathbf{v}, \end{aligned} \tag{2.10}$$

where ρ and μ are the density and kinematic viscosity, p the fluid pressure and $\mathbf{v}^\intercal = [u, v, w]$ is the velocity field where u, v, w are the velocities in Cartesian coordinate directions x, y, z, respectively. Subsequently, the VANS equations are derived by taking the superficial volume average of the Navier-Stokes equations:

$$\begin{aligned} \langle \nabla \cdot \mathbf{v} \rangle^{s\phi} &= \langle 0 \rangle^{s\phi} = 0, \\ \langle \rho \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) \rangle^{s\phi} &= \langle -\nabla p + \mu \nabla^2 \mathbf{v} \rangle^{s\phi}. \end{aligned} \tag{2.11}$$

A derivation of the averaged continuity equation is found in (Ochoa-Tapia & Whitaker, 1995) and shown in Appendix 2.A, resulting in:

$$\nabla \cdot \langle \mathbf{v} \rangle^{s\phi} = 0. \tag{2.12}$$

The superficial velocity field is thus solenoidal and is divergence-free in contrast to the intrinsic velocity field which has to satisfy:

$$\nabla \cdot \langle \mathbf{v} \rangle^{s\phi} = \nabla \cdot \left(\alpha_\phi \langle \mathbf{v} \rangle^{i\phi} \right) = \nabla \cdot \langle \mathbf{v} \rangle^{i\phi} \alpha_\phi + \langle \mathbf{v} \rangle^{i\phi} \cdot \nabla \alpha_\phi = 0, \tag{2.13}$$

where we used the relation between superficial and intrinsic averages $\langle \mathbf{v} \rangle^{s\phi} = \alpha_\phi \langle \mathbf{v} \rangle^{i\phi}$. We prefer to solve for the superficial flow field as this will simplify the required solution procedure. Subsequently, we focus our attention on the averaging of the more complex momentum equations. For the derivation of the left-hand side of the averaged momentum equation we refer the reader to (Whitaker, 1996), and for the derivation of the right-

hand side we refer to (Ochoa-Tapia & Whitaker, 1995), resulting in:

$$\begin{aligned}&\rho\left(\frac{\partial\langle\mathbf{v}\rangle^{s\phi}}{\partial t}+\langle\mathbf{v}\rangle^{s\phi}\cdot\nabla\frac{\langle\mathbf{v}\rangle^{s\phi}}{\alpha_\phi}+\nabla\cdot\langle\tilde{\mathbf{v}}\tilde{\mathbf{v}}\rangle^{s\phi}\right)\\&=-\alpha_\phi\nabla\langle p\rangle^{i\phi}+\mu\nabla^2\langle\mathbf{v}\rangle^{s\phi}-\mu\nabla\alpha_\phi\cdot\nabla\frac{\langle\mathbf{v}\rangle^{s\phi}}{\alpha_\phi}+\frac{1}{V}\int_{\Gamma_{\phi\beta}}\left(-\tilde{p}+\mu\nabla\tilde{\mathbf{v}}\right)\cdot\boldsymbol{n}_\phi d\Gamma_{\phi\beta},\end{aligned}\tag{2.14}$$

where separation of scales is already used to simplify the equations. Whereas the *superficial* flow average is used, the VANS equations use the *intrinsic* pressure average for reasons explained in Section 2.2.3. For the interested reader, an example of the derivation of the averaged continuity equation and viscous forces is given in Appendix 2.A. In the coming section, an interpretation and simplification of the VANS equations will be given. The simplification and interpretation will follow results from literature, and are aimed at building a better understanding of the VANS equations in the context of topology optimization.

2.2.3. INTERPRETATION OF THE VANS EQUATIONS FOR SOLID/FLUID TOPOLOGY OPTIMIZATION

The VANS equations have many terms which have different physical origins. Closely inspecting these terms helps in using the VANS equations for topology optimization. Firstly, we focus on terms containing deviational velocities ($\tilde{\mathbf{v}}$) and pressures ($\tilde{p}$), as we do not want to solve for them and want to remove them from the equations. The boundary integral in Equation 2.14 is investigated:

$$\frac{1}{V}\int_{\Gamma_{\phi\beta}}\left(-\tilde{p}+\mu\nabla\tilde{\mathbf{v}}\right)\cdot\boldsymbol{n}_\phi d\Gamma_{\phi\beta}.\tag{2.15}$$

In (Whitaker, 1996) this term is referred to as the *surface filter*, as it filters information from the microscale solid/fluid interface to the averaged scale. To form a closure relation to interpret this term, Whitaker (1996) notes that at interface $\Gamma_{\phi\beta}$ the velocity $\mathbf{v}=\langle\mathbf{v}\rangle^{i\phi}+\tilde{\mathbf{v}}=\mathbf{0}$ and thus:

$$\tilde{\mathbf{v}}=-\langle\mathbf{v}\rangle^{i\phi}\text{ at }\Gamma_{\phi\beta}.\tag{2.16}$$

Subsequently, Whitaker (1996) uses this idea to construct a relation between the deviational and averaged quantities:

$$\begin{aligned}\tilde{p}&=\mu\boldsymbol{m}\cdot\langle\mathbf{v}\rangle^{i\phi},\\\tilde{\mathbf{v}}&=\boldsymbol{M}\langle\mathbf{v}\rangle^{i\phi},\end{aligned}\tag{2.17}$$

where $\boldsymbol{m}$ and $\boldsymbol{M}$ are a vector and matrix respectively, used to relate averaged to deviational quantities and close the VANS equations. By inserting these approximations in the surface filter, it may be simplified as:

$$\begin{aligned}&\frac{1}{V}\int_{\Gamma_{\phi\beta}}\left(-\tilde{p}+\mu\nabla\tilde{\mathbf{v}}\right)\cdot\boldsymbol{n}_\phi d\Gamma_{\phi\beta}\\&=\frac{\mu}{V}\int_{\Gamma_{\phi\beta}}\left(-\boldsymbol{m}+\nabla\cdot\boldsymbol{M}\right)\boldsymbol{n}_\phi^{\mathrm{T}}d\Gamma_{\phi\beta}\cdot\langle\mathbf{v}\rangle^{i\phi}=-\mu\boldsymbol{K}\langle\mathbf{v}\rangle^{i\phi}=-\mu\kappa\boldsymbol{I}\frac{\langle\mathbf{v}\rangle^{s\phi}}{\alpha_\phi},\end{aligned}\tag{2.18}$$

where the intrinsically averaged velocity is substituted with the superficially averaged velocity. Whitaker (1996) goes into great detail to define $\boldsymbol{K}$, but for the purpose of this work we recognize it as the *Darcy's law permeability tensor*. Furthermore, the permeability tensor is simplified as $\boldsymbol{K} = \kappa \boldsymbol{I}$ by assuming isotropy of the porous medium and the tensor is recognized as the penalization used in the NSDP equations for topology optimization to inhibit flow through solid domains. Moreover, Whitaker (1996) defines the order of magnitude for the Darcy permeability tensor as:

$$\frac{1}{V}\int_{\Gamma_{\phi\beta}} (-\boldsymbol{m} + \nabla \cdot \boldsymbol{M})\, \boldsymbol{n}_\phi^\intercal d\Gamma_{\phi\beta} = \kappa = \mathcal{O}\left(\frac{\alpha_\phi}{l_\phi^2}\right) + \mathcal{O}\left(\frac{\alpha_\phi \rho \langle \mathbf{v} \rangle^{i\phi}}{\mu l_\phi}\right). \tag{2.19}$$

Choosing the correct magnitude for κ, such that flow through the solid domain is inhibited sufficiently in an optimization procedure, remains a challenge. However, in Section 2.4 an order analysis will be used to derive lower bounds on $\mathcal{O}(\kappa)$ which will result in lower bounds of similar form as Equation 2.19.

Another interesting term in the momentum equation is the so called *volume filter* (Whitaker, 1996):

$$\rho \nabla \langle \tilde{\mathbf{v}} \tilde{\mathbf{v}} \rangle^{s\phi}, \tag{2.20}$$

which filters information from the flow on microscale to the macroscale. This is actually similar to the *Reynolds stress tensor* used in the Reynolds Averaged Navier Stokes (RANS) equations for turbulent flow modeling (Alfonsi, 2009). In this work we will neglect this term as we assume the deviational velocities to be small, and remain within the laminar flow regime:

$$\rho \nabla \langle \tilde{\mathbf{v}} \tilde{\mathbf{v}} \rangle^{s\phi} \approx 0. \tag{2.21}$$

Moreover, adding this term for RANS optimization might not be straightforward. In the RANS equations velocity fluctuations $\tilde{\mathbf{v}}$ stem from local eddies which are filtered out via a time average, whereas in the laminar VANS equations velocity fluctuations $\tilde{\mathbf{v}}$ stem from flow interaction at the pore scale solid/fluid interface.

Subsequently, we investigate the viscous terms in the averaged momentum equations:

$$\mu \nabla^2 \langle \mathbf{v} \rangle^{s\phi} - \mu \nabla \alpha_\phi \cdot \nabla \frac{\langle \mathbf{v} \rangle^{s\phi}}{\alpha_\phi}, \tag{2.22}$$

where the first part is called the *Brinkman correction* and the second part the *second Brinkman correction* (Brinkman, 1949; Ochoa-Tapia & Whitaker, 1995). In an optimized solid/fluid design flow in "solid" porous regions is generally limited and $\nabla^2 \langle \mathbf{v} \rangle^{s\phi} \to 0$. The Brinkman correction is thus mainly important in the fluid regions. In contrast, the second Brinkman correction is mainly important in boundary regions where large gradients in volume fraction $\nabla \alpha_\phi$ are found. In solid/fluid topology optimization the aim is to create crisp 0-1 designs where a porous "solid" region of low permeability ($\alpha_\phi \to 0$) is adjacent to a fluid region ($\alpha_\phi = 1$). In the boundary regions the second Brinkman correction should thus be included. However, according to Whitaker (1986) a solid wall should not be approximated using the second Brinkman correction as the length scale of the averaging volume r_0 becomes of the same order as the length scales of α_ϕ and $\langle \mathbf{v} \rangle^{i\phi}$ and we cannot adhere to the separation of scales. One of the solutions to this problem is to

use a two domain approach in which separate governing equations are defined for the homogeneous fluid and solid domains. Subsequently, these equations are coupled via a jump condition on the solid/fluid interface concerning velocities and shear stresses (Angot et al., 2017; Beavers & Joseph, 1967; Ochoa-Tapia & Whitaker, 1995). However, Breugem and Boersma (2005) and Hernandez-Rodriguez et al. (2022) show that flow along a porous wall can be accurately simulated using the VANS equations including the second Brinkman correction by using a correct interpretation of the results and a correct definition of the permeability tensor. As we want to optimize the solid/fluid layout within the design domain and do not know the solid/fluid domains *a priori*, we prefer to use only the VANS equations and not use a two domain approach.

A physical interpretation of the second Brinkman correction for solid/fluid topology optimization is given using Figure 2.3. In the derivation of the VANS equations in Appendix 2.A the correction shows up as a simplification of a surface integral over the microscale solid/fluid interface $\Gamma_{\phi\beta}$:

$$\frac{\mu}{V}\int_{\Gamma_{\phi\beta}} \nabla\langle\mathbf{v}\rangle^{i\phi}\boldsymbol{n}_\phi d\Gamma = -\mu\nabla\alpha_\phi\cdot\nabla\langle\mathbf{v}\rangle^{i\phi} = -\mu\nabla\alpha_\phi\cdot\nabla\frac{\langle\mathbf{v}\rangle^{s\phi}}{\alpha_\phi}. \tag{2.23}$$

In Figure 2.3 we show an averaging volume on porous/fluid interface Γ_{fp} where the porous material approaches a solid $l_\phi \to 0$. In the "solid" porous domain flow speeds and gradients within the pores are negligible with respect to flow speeds within the fluid domain. Within the averaging volume we may thus neglect the surface integral over porous domain boundaries $\Gamma_{\phi\beta}\setminus\Gamma_{fp}$ and simplify the boundary integral as:

$$\frac{\mu}{V}\int_{\Gamma_{\phi\beta}} \nabla\langle\mathbf{v}\rangle^{i\phi}\boldsymbol{n}_\phi d\Gamma = \frac{\mu}{V}\int_{\Gamma_{fp}\cap\Gamma_{\phi\beta}} \nabla\langle\mathbf{v}\rangle^{i\phi}\boldsymbol{n}_\phi d\Gamma. \tag{2.24}$$

Moreover, if the formulation is taken to its limits where the porous material is a solid and $l_\phi = 0$, it is clear that $\Gamma_{\phi\beta} = \Gamma_{fp}$ and Equation 2.24 holds. Furthermore, the momentum equation (and its volume average) represent an equilibrium between mass flow acceleration and stresses on an small domain of fluid. Subsequently, the boundary integral over $\Gamma_{fp}\cap\Gamma_{\phi\beta}$ is interpreted as an average of the viscous stresses ($\mu\nabla\langle\mathbf{v}\rangle^{i\phi}$) on the solid/fluid interface. Within the averaging domain, these stresses are supported by the solid material at the interface. Supporting a part of these stresses by a rigid solid material thus reduces mass flow acceleration and consequently flow. The second Brinkman correction thus represents the support of fluid domain viscous stresses by the solid material at the porous/fluid interface. Moreover, if these fluid domain viscous stresses would not be supported by the solid material, they would have to be be supported by the fluid in the porous domain. The second Brinkman correction can thus be said to remove fluid domain viscous forces from flow in the porous domain.

Subsequently, we interpret the inertial term on the left-hand side of Equation 2.14:

$$\rho\langle\mathbf{v}\rangle^{s\phi}\cdot\nabla\frac{\langle\mathbf{v}\rangle^{s\phi}}{\alpha_\phi} = \langle\mathbf{v}\rangle^{s\phi}\cdot\nabla\left(\rho\langle\mathbf{v}\rangle^{i\phi}\right), \tag{2.25}$$

where we simplified $\langle\mathbf{v}\rangle^{s\phi}/\alpha_\phi = \langle\mathbf{v}\rangle^{i\phi}$. The inertial term on the right-hand side represents the advection of microscale momentum $\rho\langle\mathbf{v}\rangle^{i\phi}$ by bulk flow $\langle\mathbf{v}\rangle^{s\phi}$. The supposed

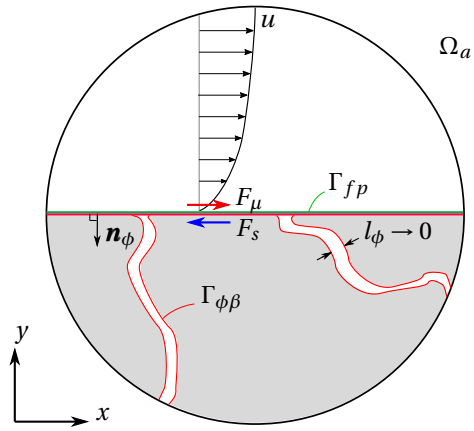


Figure 2.3: An illustration of the effect of the second Brinkman correction on porous/fluid interface Γ_{fp}. Within the pores flow magnitude is small with respect to flow magnitude in the fluid domain. Fluid domain viscous forces are subsequently mainly supported by the solid material at Γ_{fp}.

advantage of this term in topology optimization is illustrated on a converged solid/fluid design. In topology optimization, the solid domain is defined by a low and constant volume fraction $\alpha_\phi = \underline{\alpha} \ll 1$, resulting in:

$$\rho \langle \mathbf{v} \rangle^{s\phi} \cdot \nabla \frac{\langle \mathbf{v} \rangle^{s\phi}}{\alpha_\phi} = \frac{\rho}{\underline{\alpha}} \langle \mathbf{v} \rangle^{s\phi} \cdot \nabla \langle \mathbf{v} \rangle^{s\phi}. \tag{2.26}$$

Comparing the right-hand side of Equation 2.26 to the standard inertial term in the Navier-Stokes equations $\rho \mathbf{v} \cdot \nabla \mathbf{v}$ (as in Equation 2.10), we notice that the density is divided by $\underline{\alpha}$. The division by the volume fraction is interpreted as a scaling of density in the solid domain $\rho^s = \rho / \underline{\alpha} \gg \rho$, where "density" thus increases relative to the fluid domain where $\alpha_\phi = 1$. As larger masses require higher forces to accelerate, this should reduce flow in the solid domain. However, in Section 2.4 we will argue and in Section 2.6 we will find that the effect of this term on flow reduction is small when solid/fluid laminar flow designs are evaluated. Nonetheless, we will keep this term in our optimization model as the work by Alonso and Silva (2022) who added the Forchheimer penalization besides the Darcy penalization suggests that a quadratic flow penalization improves designs found at higher Reynolds numbers. In this sense, the inertia term is interpreted as a quadratic flow penalization which passes information on inertial effects to the sensitivities.

Finally, the new pressure term on the right-hand side of Equation 2.14 is interpreted:

$$-\alpha_\phi \nabla \langle p \rangle^{i\phi}. \tag{2.27}$$

For the pressure field the intrinsic average is used, as the superficial pressure average

$(\langle p\rangle^{i\phi} = \langle p\rangle^{s\phi}/\alpha_\phi)$ leads to more complex equations:

$$-\alpha_\phi \nabla\langle p\rangle^{i\phi} = -\alpha_\phi \nabla\left(\frac{\langle p\rangle^{s\phi}}{\alpha_\phi}\right) = -\nabla\langle p\rangle^{s\phi} + \frac{\langle p\rangle^{s\phi}}{\alpha_\phi}\nabla\alpha_\phi. \tag{2.28}$$

Furthermore, in the standard momentum equation (as shown in Equation 2.10), the pressure gradient $-\nabla p$ represents the volumetric pressure forces acting on a parcel of fluid. However, in the VANS equations these forces are multiplied by the fluid volume fraction $-\alpha_\phi \nabla\langle p\rangle^{i\phi}$. As the solid domain in topology optimization is defined as the domain where $\alpha_\phi = \underline{\alpha} \ll 1$, pressure forces on a fluid parcel in this domain are reduced and $\underline{\alpha}\nabla\langle p\rangle^{i\phi} \ll \nabla\langle p\rangle^{i\phi}$. Consequently, flow leakage caused by large pressure gradients in the solid domain is reduced. Moreover, if we assume that these pressure gradients are the main cause of flow leakage, the pressure penalization directly inhibits these errors in converged solid domains where $\alpha_\phi \to 0$ while it does not add much penalization in intermediate designs with a lot of "gray" material where $\alpha_\phi \approx 0.5$. In fact, the pressure penalization will allow us to use a lower maximum Darcy penalization than the one required by the NSDP equations as will be shown in Section 2.6. The advantage of this lowered penalization is that intermediate designs containing a lot of gray material will be penalized less and the optimizer will be less restricted than when the NSDP equations are used with a larger maximum penalization.

Implementing the simplifications in Equations 2.18 and 2.21, the VANS momentum equation is simplified to:

$$\rho\left(\frac{\partial\langle\mathbf{v}\rangle^{s\phi}}{\partial t} + \langle\mathbf{v}\rangle^{s\phi}\cdot\nabla\frac{\langle\mathbf{v}\rangle^{s\phi}}{\alpha_\phi}\right) = -\alpha_\phi\nabla\langle p\rangle^{i\phi} + \mu\nabla^2\langle\mathbf{v}\rangle^{s\phi} - \mu\nabla\alpha_\phi\cdot\nabla\frac{\langle\mathbf{v}\rangle^{s\phi}}{\alpha_\phi} - \frac{\mu\kappa\langle\mathbf{v}\rangle^{s\phi}}{\alpha_\phi}, \tag{2.29}$$

where we thus used superficial velocity averages and intrinsic pressure averages. Furthermore, as might be obvious at this stage, we aim to use the VANS equations for topology optimization where volume fraction α_ϕ is used as a design variable.

2.2.4. COMPARISON TO STANDARD PRACTICE

The VANS equations show many similarities to the NSDP equations often used for fluid topology optimization:

$$\begin{aligned}\rho\left(\frac{\partial\mathbf{v}}{\partial t} + \mathbf{v}\cdot\nabla\mathbf{v}\right) &= -\nabla p + \mu\nabla^2\mathbf{v} - \mu\kappa\mathbf{v},\\ \nabla\cdot\mathbf{v} &= 0,\end{aligned} \tag{2.30}$$

where Darcy penalization κ is a function of the design variables. In the solid domain flow is inhibited by setting a large κ resulting in a high flow penalization $-\mu\kappa\mathbf{v}$, while in the fluid domain $\kappa = 0$ and Equation 2.30 collapses to the standard Navier-Stokes equation (Equation 2.10). If the NSDP equations are a simplification of the VANS equations, we can deduce that superficial velocity averages are used in the NSDP equations as the continuity equation is the same as the left-hand side of Equation 2.13. Differences between the VANS and NSDP equations are found in the momentum equations. Rewriting the VANS (Equation 2.29) to the NSDP (Equation 2.30) equations can thus be done by assuming the velocity and pressure in the NSDP equations to be the superficial and intrinsic averages respectively $(\langle\mathbf{v}\rangle^{s\phi} = \mathbf{v}, \langle p\rangle^{i\phi} = p)$, and:

1. $-\cancel{\mu \nabla \alpha_\phi \cdot \nabla \frac{\langle \mathbf{v} \rangle^{s\phi}}{\alpha_\phi}}^{0} = 0$: The second Brinkman correction is not included in the NSDP equations, and the support of fluid domain viscous stresses at the solid/fluid interface is removed.

2. $\rho \langle \mathbf{v} \rangle^{s\phi} \cdot \nabla \frac{\langle \mathbf{v} \rangle^{s\phi}}{\cancel{\alpha_\phi}^{1}} = \rho \mathbf{v} \cdot \nabla \mathbf{v}$: Volume fraction α_ϕ is removed from the inertia term, and its flow leakage reducing effect as illustrated by Equation 2.26 and its influence on the sensitivities are removed.

3. $-\cancel{\alpha_\phi}^{1} \nabla \langle p \rangle^{i\phi} = \nabla \langle p \rangle^{i\phi}$: Flow leakage due to high pressure gradients in the solid domain is worsened as the volume fraction is removed from the pressure term.

4. $-\frac{\mu \kappa}{\cancel{\alpha_\phi}^{1}} \langle \mathbf{v} \rangle^{s\phi} = \mu \kappa \mathbf{v}$: In the Darcy penalization, the division by α_ϕ is removed. However, in Section 2.3.1 we will show that using certain interpolation functions $\kappa(\alpha_\phi)$, the Darcy penalization in the VANS and NSDP equations can be similar or the same.

The NSDP equations are thus a simplification of the VANS equations where we hypothesize that the VANS equations will be able to more precisely describe flow in an optimized solid/fluid topology. For the remainder of this work we will simplify notation by dropping the brackets and superscripts from the averaged variables and assuming that $\mathbf{v}$ is the superficial velocity average, p the intrinsic pressure average and α the fluid volume fraction.

2.3. DISCRETIZATION OF THE VANS EQUATIONS

To use the VANS Equations in topology optimization, volume fraction α is used as design variable. A well known method to solve and discretize computational fluid dynamics (CFD) is the finite volume (FV) method, which is often preferred due to its natural ability to conserve mass and momentum. As the topology optimization community originated from the structural analysis community where the Finite Element Method (FEM) is the standard, most fluidic optimization papers use FEM. However, the structured square meshes often used in topology optimization are highly suited for discretization using the FV method, and fast solution algorithms exist for these kind of problems.

2.3.1. DISCRETIZED MOMENTUM EQUATION

The VANS and NSDP equations are discretized using the FV method and solved using the Semi-Implicit Method for Pressure Linked Equations (SIMPLE) algorithm as described by Versteeg and Malalasekera (2007). The NSDP equations are discretized following Versteeg and Malalasekera (2007), with an exception for the Darcy penalization which is discretized in Section 2.3.1. In this section we first show the VANS discretization as several terms in the VANS equations are not standard in a CFD analysis. Furthermore, in this work we only consider 2D problems where we neglected flow w in the z-direction.

The VANS equations are discretized on an equidistant staggered grid as shown in Figure 2.4, where different control volumes are used for the continuity (p-control), u-momentum and v-momentum equations. Subsequently, Equation 2.29 is split into u/v-

momentum equations as:

$$\begin{aligned}\rho \mathbf{v}\cdot\nabla\frac{u}{\alpha} &= -\alpha p_{,x} + \mu\nabla^2 u - \mu\nabla\alpha\cdot\nabla\frac{u}{\alpha} - \mu\kappa\frac{u}{\alpha},\\ \rho \mathbf{v}\cdot\nabla\frac{v}{\alpha} &= -\alpha p_{,y} + \mu\nabla^2 v - \mu\nabla\alpha\cdot\nabla\frac{v}{\alpha} - \mu\kappa\frac{v}{\alpha},\end{aligned} \tag{2.31}$$

where we assumed solutions to be static ($\partial u/\partial t = \partial v/\partial t = 0$), and spatial derivatives are

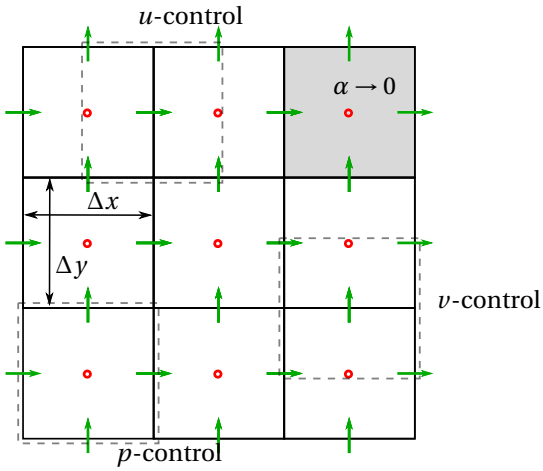


Figure 2.4: The staggered equidistant grid used to discretize the VANS equations using the FV method. velocity DOFs (green arrows) are located at the cell faces, while pressure DOFs and design variables are located at the cell centers (red dots). Design variables α are attached to the cell centers and represent a constant volume fraction within the corresponding cell, as illustrated by the upper right cell which is solid and where $\alpha \to 0$. Different control volumes are defined for the u, v momentum equations, and the continuity (p) equation. At the u-control volume all neighboring DOFS contributing to the momentum equation are depicted in Figure 2.5.

written as $p_{,x} = \partial p/\partial x$ and $p_{,y} = \partial p/\partial y$. Only the discretization of the u-momentum equations is described since the discretization of the v-momentum equation is analogous. To discretize the u-momentum equation we use control volume (CV) Ω_u, as shown in Figure 2.5. The CV has boundary $\Gamma_u = \overline{\Omega}_u \setminus \Omega_u = \Gamma^N \cup \Gamma^E \cup \Gamma^S \cup \Gamma^W$, where the superscripts denote north, east, south and west boundaries. Design variables representing volume fractions within the cells are attached to the red pressure nodes in Figure 2.5, we however interpolate these variables on the north/south edges of the CV and at the center of the CV (at DOF u^P in Figure 2.5):

$$\begin{aligned}\alpha^{CN} &= \frac{\alpha^E + \alpha^W + \alpha^{NE} + \alpha^{NW}}{4} \text{ on } \Gamma^N,\\ \alpha^{CS} &= \frac{\alpha^E + \alpha^W + \alpha^{SE} + \alpha^{SW}}{4} \text{ on } \Gamma^S,\\ \alpha^P &= \frac{\alpha^E + \alpha^W}{2} \text{ at } u^P,\end{aligned} \tag{2.32}$$

where all design variables on the right-hand sides can be found in Figure 2.5. Further-

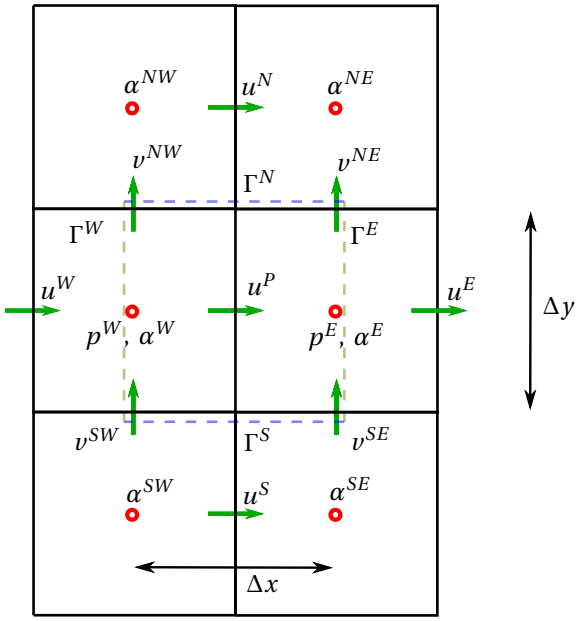


Figure 2.5: Control volume Ω_u for the u-momentum equation. We denote the relevant DOFs and design variabels with respect to the center DOF u^P. The boundary of the control volume is divided into horizontal boundaries Γ^N and Γ^S with length Δx, and vertical boundaries Γ^E and Γ^W with length Δy.

more, v-velocities on the north/south edges are interpolated as:

$$\bar{v}^N = \frac{v^{NE} + v^{NW}}{2} \text{ on } \Gamma^N, \qquad \bar{v}^S = \frac{v^{SE} + v^{SW}}{2} \text{ on } \Gamma^S. \tag{2.33}$$

To discretize the u-momentum equation it is integrated over its control volume:

$$\int_{\Omega_u} \rho \mathbf{v} \cdot \nabla \frac{u}{\alpha} d\Omega = \int_{\Omega_u} \left(-\alpha p_{,x} + \mu \nabla^2 u - \mu \nabla \alpha \cdot \nabla \frac{u}{\alpha} - \mu \kappa \frac{u}{\alpha} \right) d\Omega. \tag{2.34}$$

In the subsequent sections first the inertial pressure and viscous terms will be discretized after which more attention is given to the second Brinkman correction and Darcy penalization, finally the discretization is finished by discretizing the continuity equation.

DISCRETIZED INERTIAL, PRESSURE AND VISCOUS TERMS

The inertial term is discretized by applying the divergence theorem on the CV:

$$\int_{\Omega_u} \rho \mathbf{v} \cdot \nabla \frac{u}{\alpha} d\Omega = \int_{\Omega_u} \rho \nabla \cdot \left(\mathbf{v} \frac{u}{\alpha} \right) d\Omega_u = \rho \int_{\Gamma_u} \mathbf{v} \frac{u}{\alpha} \cdot \boldsymbol{n}_u d\Gamma_u, \tag{2.35}$$

where we used the continuity equation ($\nabla \cdot \mathbf{v} = 0$) to rewrite $\mathbf{v} \cdot \nabla \frac{u}{\alpha} = \nabla \cdot \left(\mathbf{v} \frac{u}{\alpha} \right)$ and $\boldsymbol{n}_u$ is the unit normal pointing outward of Ω_u on Γ_u. We discretize the inertial terms using

approximations on the north, south and east, west boundaries, respectively:

$$\int_{\Gamma_u} \rho \mathbf{v} \frac{u}{\alpha} \cdot \boldsymbol{n}_u d\Gamma_u = \rho \Delta x \Bigg(\overbrace{\bar{v}^N \frac{u^N + u^P}{2\alpha^{CN}}}^{\Gamma^N} - \overbrace{\bar{v}^S \frac{u^S + u^P}{2\alpha^{CS}}}^{\Gamma^S} \Bigg) + \rho \Delta y \Bigg(\overbrace{\frac{(u^E + u^P)^2}{4\alpha^E}}^{\Gamma^E} - \overbrace{\frac{(u^W + u^P)^2}{4\alpha^W}}^{\Gamma^W} \Bigg), \tag{2.36}$$

where Δx and Δy are the horizontal and vertical lengths of the control volume as in Figure 2.5. In the discretized term, intrinsic momentum $\rho u/\alpha$ is advected through the boundaries by superficial flow average $\mathbf{v}$. If momentum is to be conserved, advection of inertial momentum through a CV boundary should be consistent for all adjacent CV's. The intrinsic momentum $\rho u/\alpha$ and superficial flow $\mathbf{v}$ on a certain boundary should thus be the same for both elements adjacent to the boundary. As $\mathbf{v}$ and u are already interpolated consistently for all elements on the boundaries, volume fractions α^{CN} and α^{CS} are also interpolated consistently in Equation 2.32 at the north and south boundaries.

The pressure term is discretized by approximating the gradient in pressure and α at the center of the CV:

$$\int_{\Omega_u} \alpha p_{,x} d\Omega_u = \Delta x \Delta y \alpha^P \frac{p^E - p^W}{\Delta x} = \Delta y \alpha^P \left(p^E - p^W \right). \tag{2.37}$$

To discretize the first Brinkman correction, the divergence theorem is used:

$$\int_{\Omega_u} \mu \nabla^2 u d\Omega = \mu \int_{\Omega_u} \nabla \cdot (\nabla u) \, d\Omega = \mu \int_{\Gamma_u} \nabla u \cdot \boldsymbol{n}_u d\Gamma. \tag{2.38}$$

Subsequently, flow gradients are approximated on the north, south and east, west boundaries as:

$$\mu \int_{\Gamma_u} (\nabla u) \cdot \boldsymbol{n}_u d\Gamma = \mu \Delta x \Bigg(\overbrace{\frac{u^N - u^P}{\Delta y}}^{\Gamma^N} - \overbrace{\frac{u^P - u^S}{\Delta y}}^{\Gamma^S} \Bigg) + \mu \Delta y \Bigg(\overbrace{\frac{u^E - u^P}{\Delta x}}^{\Gamma^E} - \overbrace{\frac{u^P - u^W}{\Delta x}}^{\Gamma^W} \Bigg). \tag{2.39}$$

Most of the discretization techniques used until this point are fairly common and can be found in (Versteeg & Malalasekera, 2007). However, in most common methods fluid volume fraction α is not included, and in its inclusion and interpolation we deviate from most standard discrete CFD solvers.

SECOND BRINKMAN CORRECTION ON A WALL ORTHOGONAL TO THE FLOW

Special attention is given to the second Brinkman correction as it is not common in a standard FV discretization. The second Brinkman correction in Equation 2.22 depends on the gradient in volume fraction $\nabla\alpha$, and is used to support fluid domain viscous forces as explained in Section 2.2.3. The second Brinkman correction is investigated for a converged design where there are distinct regions of solid ($\alpha \to 0$) and fluid ($\alpha = 1$) material. In such a solid/fluid design, the gradient $\nabla\alpha = [\alpha_{,x} \; \alpha_{,y}]^\intercal$ is only present on the porous/fluid interface (the solid wall), and is a vector normal to the interface pointing to the fluid domain. The two parts of the gradient $\alpha_{,x}$ and $\alpha_{,y}$ are investigated separately. In fact, the case where only $\alpha_{,y} = 0$ represents a vertical wall orthogonal to u as shown

in Figure 2.6, and the case where only $\alpha_{,x} = 0$ represents a horizontal wall parallel to u as shown in Figure 2.7. The correction is thus split into a vertical and horizontal component:

$$B_2 = -\mu \int_{\Omega_u} \nabla\alpha \cdot \nabla \frac{u}{\alpha} d\Omega = -\mu \int_{\Omega_u} \left(\alpha_{,x} \left(\frac{u}{\alpha} \right)_{,x} + \alpha_{,y} \left(\frac{u}{\alpha} \right)_{,y} \right) d\Omega. \tag{2.40}$$

Firstly, the correction is constructed for the vertical wall in Figure 2.6, orthogonal to

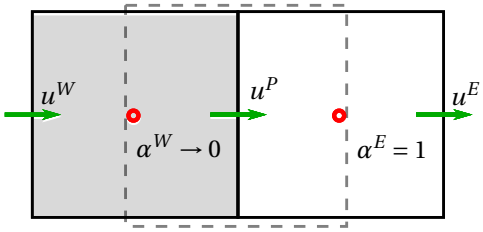


Figure 2.6: The relevant DOFs and design variables for the discretization of the second Brinkman correction in the CV for u^P. In this example, the elements to the left are solid ($\alpha^W \to 0$), the elements to the right are fluid ($\alpha^E = 1$), and $\alpha_{,y} = 0$ resulting in a vertical wall.

flow u in x-direction. The volume fraction is only dependent on x and the gradient in x-direction at the center of the CV is approximated as:

$$\alpha_{,x} = \frac{\alpha^E - \alpha^W}{\Delta x}, \tag{2.41}$$

where α^E, α^W are the east and west volume fractions as in Figure 2.6. Subsequently, the gradient in u/α at the center of the CV is approximated as:

$$\left(\frac{u}{\alpha} \right)_{,x} = \frac{1}{\alpha} \left(u_{,x} - \frac{u}{\alpha} \alpha_{,x} \right) = \frac{1}{\alpha^P} \left(\frac{u^E - u^W}{2\Delta x} - \frac{u^P}{\alpha^P} \frac{\alpha^E - \alpha^W}{\Delta x} \right), \tag{2.42}$$

where α^P is the volume fraction at u^P as in Equation 2.32. Using Equations 2.41 and 2.42, the second Brinkman correction for a vertical wall orthogonal to the flow direction is discretized as:

$$\begin{aligned} B_o &= -\mu \int_{\Omega_u} \left(\alpha_{,x} \left(\frac{u}{\alpha} \right)_{,x} + \cancelto{0}{\alpha_{,y}} \left(\frac{u}{\alpha} \right)_{,y} \right) d\Omega \\ &= -\mu \frac{\Delta y}{\Delta x} \frac{\alpha^E - \alpha^W}{2\alpha^P} (u^E - u^W) + u^P \mu \frac{\Delta y}{\Delta x} \left(\frac{\alpha^E - \alpha^W}{\alpha^P} \right)^2. \end{aligned} \tag{2.43}$$

The last term in the second Brinkman correction works in the opposite direction of the Darcy penalization $-\mu\kappa u/\alpha$ in Equation 2.34. This term will be neglected as it is assumed to be small compared to the Darcy penalization which will be discretized in Equation 2.60 and whose lower bound will be defined in Section 2.4, resulting in:

$$B_o = -\mu \frac{\Delta y}{\Delta x} \frac{\alpha^E - \alpha^W}{2\alpha^P} (u^E - u^W). \tag{2.44}$$

We investigate the correction for the vertical wall in Figure 2.6 where $\alpha^E = 1$, $\alpha^W \to 0$ resulting in $\alpha^P \approx 0.5$ and:

$$B_o^E \approx -\mu \frac{\Delta y}{\Delta x}(u^E - u^W). \tag{2.45}$$

If the correction is combined with the viscous forces on east boundary Γ^E in Equation 2.39:

$$F_\mu^E = \mu \frac{\Delta y}{\Delta x}(u^E - u^W), \tag{2.46}$$

we find that the second Brinkman correction removes these forces from the control volume as $B_o^E + F_\mu^E \approx 0$. The second Brinkman correction thus removes the viscous forces due to fluid flow to the east where no porous material is present ($\alpha^E = 1$). This is exactly the goal of adding it as these forces are in fact supported by the solid material in the porous domain as explained in Section 2.2.3.

SECOND BRINKMAN CORRECTION ON A WALL PARALLEL TO THE FLOW

In Section 2.3.1 the second Brinkman correction for flow orthogonal to a wall is investigated and explicitly discretized. However, horizontal walls parallel to u also contribute to the second Brinkman correction. As an example, we investigating the horizontal wall in Figure 2.7 where $\alpha^P = 1$ and $\alpha^{PN} \to 0$. A mismatch in the exact location of the wall is found. If the wall is interpreted as the boundary where flow stagnates, this leads to an interface at either $y = \Delta y$ ($u^N \to 0$) or at $y = \Delta y/2$ ($v^{NE} \to 0$ and $v^{SE} \to 0$). Furthermore, for approximating gradients $u_{,y}$ on Γ^N in Equation 2.39, we approximated the velocity profile as:

$$u(y) = u^P + \frac{u^N - u^P}{\Delta y} y, \tag{2.47}$$

resulting in a flow of $u(y = \Delta y/2) = \frac{u^P + u^N}{2}$ at north edge Γ^N, which coincides with the wall where flow should be stagnant. In this case, a better approximation of the flow at

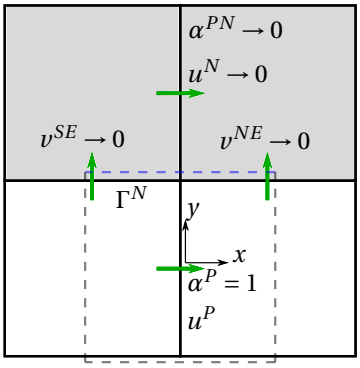


Figure 2.7: The relevant DOFs and design variables for the discretization of the second Brinkman correction in the CV for u^P. In this example, the elements to the north are solid ($\alpha^{PN} \to 0$), the elements in the CV are fluid ($\alpha^P = 1$), and $\alpha_{,x} = 0$ resulting in a horizontal wall which coincides with the north edge of the CV Γ^N.

Γ^N would be u^N as it should tend to zero. If opposed to Figure 2.7 $\alpha^P \to 0$ and $\alpha^{PN} = 1$ and the porous domain is to the south of the wall at Γ^N instead of to the north, it follows that $u^P \to 0$ is a better approximation of the velocity at the wall at Γ^N. Based on these requirements a new velocity interpolation is constructed on $0 \leq y < \frac{\Delta y}{2}$:

$$u(y) = u^P + (1 + \alpha^P - \alpha^{PN}) \frac{u^N - u^P}{\Delta y} y. \tag{2.48}$$

which has a complementary interpolation on $\frac{\Delta y}{2} < y \leq \Delta y$:

$$u(y) = u^N + (1 + \alpha^{PN} - \alpha^P) \frac{u^N - u^P}{\Delta y} (y - \Delta y). \tag{2.49}$$

A plot of the resulting flow fields is shown in Figure 2.8. When $\alpha^P = \alpha^{PN}$ no wall is present at Γ^N and Equations 2.48 and 2.49 collapse into Equation 2.47. Moreover, as α^P and α^{PN} remain continuous variables, we are able to continuously introduce solid walls in a topology optimization procedure. Furthermore, the new flow fields for walls at

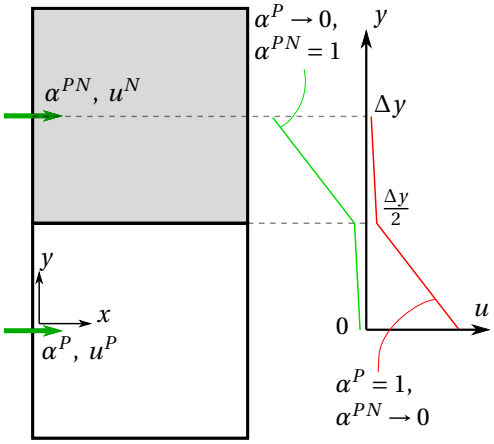


Figure 2.8: Two different flow fields for a vertical wall. Either the top elements are solid ($\alpha^{PN} \to 0$) and the flow at $\Delta y/2$ is approximated as $u^N \to 0$, or the bottom elements are solid ($\alpha^P \to 0$) and the flow at $\Delta y/2$ is approximated as $u^P \to 0$.

Γ^N is accompanied with a flow field for walls at Γ^S on $0 \geq y > -\frac{\Delta y}{2}$:

$$u(y) = u^P + (1 + \alpha^P - \alpha^{PS}) \frac{u^P - u^S}{\Delta y} y. \tag{2.50}$$

Using these interpolation functions, the correct gradients at the boundaries are computed as:

$$\begin{aligned} u^c_{,y} &= (1+\alpha^P-\alpha^{PN})\frac{u^N-u^P}{\Delta y}, \text{ on } \Gamma^N, \\ u^c_{,y} &= (1+\alpha^P-\alpha^{PS})\frac{u^P-u^S}{\Delta y}, \text{ on } \Gamma^S. \end{aligned} \tag{2.51}$$

The new gradient at the north edge contains a standard linear part (also found in Equation 2.39 at Γ^N):

$$u^l_{,y} = \frac{u^N-u^P}{\Delta y}, \text{ on } \Gamma^N, \tag{2.52}$$

as well as an update (found by subtracting the gradient in Equation 2.52 from the gradient in Equation 2.51):

$$\Delta u_{,y} = u^c_{,y} - u^l_{,y} = (\alpha^P-\alpha^{PN})\frac{u^N-u^P}{\Delta y}, \text{ on } \Gamma^N. \tag{2.53}$$

Subsequently, we define the correct viscous force at the north edge in Equation 2.39 by using $u^c_{,y} = u^l_{,y} + \Delta u_{,y}$ as:

$$\begin{aligned} &\mu\int_{\Gamma^N}(\nabla u)\cdot \boldsymbol{n}_u d\Gamma = \mu\int_{\Gamma^N}(u^l_{,y}+\Delta u_{,y})d\Gamma \\ &= \mu\Delta x\frac{u^N-u^P}{\Delta y} + \mu\Delta x(\alpha^P-\alpha^{PN})\frac{u^N-u^P}{\Delta y}, \end{aligned} \tag{2.54}$$

from which we extract an update to the standard viscous forces by subtracting the viscous force at the north edge in Equation 2.39 from Equation 2.54:

$$B^N_p = \mu\Delta x(\alpha^P-\alpha^{PN})\frac{u^N-u^P}{\Delta y}. \tag{2.55}$$

Moreover, we find the update to have precisely the function of the Second Brinkman correction as described in Section 2.2.3. When we are investigating flow in the porous domain ($\alpha^P \to 0$) with large viscous forces due to flow in a fluid domain above ($\alpha^{PN} = 1$), the update becomes:

$$B^N_p \approx -\mu\Delta x\frac{u^N-u^P}{\Delta y}, \tag{2.56}$$

which can be subtracted from the standard viscous force at Γ^N in Equation 2.39 (which is the viscous force due to $u^l_{,y}$):

$$F^N_\mu = \mu\Delta x\frac{u^N-u^P}{\Delta y}, \tag{2.57}$$

to find that for this porous control volume the viscous forces at the north edge become $B^N_p + F^N_\mu \approx 0$. The function of the second Brinkman correction is to remove viscous forces

in the fluid domain from the solid domain which is exactly what the update does. Consequently, we apply the second Brinkman correction for flow parallel to a wall using the updated velocity gradients in Equation 2.51 instead of explicitly discretizing Equation 2.40. Furthermore, the full correction for flow parallel to a wall is found by following a similar procedure on the south edge as:

$$B_p = \mu\Delta x(\alpha^P - \alpha^{PN})\frac{u^N - u^P}{\Delta y} + \mu\Delta x(\alpha^P - \alpha^{PS})\frac{u^P - u^S}{\Delta y}. \tag{2.58}$$

Subsequently, the complete second Brinkman correction can be defined by combining Equations 2.44 and 2.58 as $B_2 = B_o + B_p$.

DISCRETIZED DARCY PENALIZATION

The final term which is discretized in the volume averaged momentum equation is the Darcy penalization:

$$-\frac{\mu\kappa(\alpha)u}{\alpha} \equiv -K(\alpha)u, \tag{2.59}$$

where we introduce $K(\alpha)$ as the design dependent penalization interpolation which will be used to compare different types of interpolation functions and maximum penalization in the solid domain ($\overline{K} = K(\alpha = 0)$). The same discretization of the Darcy penalization will be used for both the VANS as well as the NSDP equations. Lower and upper bounds on κ are defined as $\overline{\kappa} \geq \kappa \geq \underline{\kappa} = 0$, which we relate to the volume fraction $0 < \alpha < 1$ via a linear interpolation as $\kappa(\alpha) = (1-\alpha)\overline{\kappa}$. As one of the aims of the discretization is to introduce the least amount of tunable parameters for optimization, a linear interpolation is used to reduce complexities and parameters in the resulting discrete flow model. Subsequently, we approximate the penalization at the center of the CV by discretizing as:

$$-\int_{\Omega_u} \mu\frac{\kappa(\alpha)}{\alpha}u d\Omega = -\Delta x\Delta y\mu\overline{\kappa}\frac{1-\alpha^P}{\alpha^P}u^P. \tag{2.60}$$

In the limit case where $\alpha^P = 0$, i.e. fully solid, this would result in an infinitely large penalization which is computationally infeasible. To prevent this we add a lower bound $\underline{\alpha} \ll 1$ on the volume fraction as $\tilde{\alpha} = \underline{\alpha} + (1-\underline{\alpha})\alpha$ and use $\tilde{\alpha}$ in the discretization such that $\alpha^P \geq \underline{\alpha}$.

In the work by Borrvall and Petersson (2003) and many subsequent papers on fluid topology optimization the Darcy penalization is interpolated as:

$$K_{Da}(\alpha) = \overline{K}_{Da}\frac{\tilde{q}(1-\alpha)}{\tilde{q}+\alpha}, \tag{2.61}$$

where $\tilde{q}$ is a parameter which is used to control convexity, generally by setting it as $0 \leq \tilde{q} \leq 1$. This convex function ensures that the penalization on intermediate designs where $\alpha \approx 0.5$ is not too severe, as a severe penalization on intermediate designs generally tends to pull the designs into local optima. In this work, no additional interpolation functions or filters are applied to $\tilde{\kappa}$, besides the linear interpolation $\kappa(\tilde{\alpha}) = (1-\tilde{\alpha})\overline{\kappa}$. However, we may examine the function in Equation 2.60 as an interpolation function:

$$K_{Su}(\tilde{\alpha}) = \mu\bar{\kappa}\frac{1-\tilde{\alpha}}{\tilde{\alpha}}, \tag{2.62}$$

which is in fact the same function as the one proposed by Evgrafov (2005). Furthermore, when $\tilde{\alpha} = \underline{\alpha} + (1-\underline{\alpha})\alpha$ is substituted into Equation 2.62, we find:

$$\begin{aligned} K_{Su}(\tilde{\alpha}(\alpha)) &= \mu\bar{\kappa}\frac{1-\underline{\alpha}-(1-\underline{\alpha})\alpha}{\underline{\alpha}+(1-\underline{\alpha})\alpha}, \\ K_{Su}(\alpha) &\approx \mu\bar{\kappa}\frac{1-\alpha}{\underline{\alpha}+\alpha}, \end{aligned} \tag{2.63}$$

where we assumed $\underline{\alpha} \ll 1$ and $1-\underline{\alpha} \approx 1$, and note that using this interpolation function we find a maximum penalization in the solid domain of:

$$\overline{K}_{Su} = K_{Su}(\alpha = 0) \approx \frac{\mu\bar{\kappa}}{\underline{\alpha}}. \tag{2.64}$$

Moreover, comparing Equations 2.63 and 2.61 we note that they scale as $\underline{\alpha} \cdot K_{Su}(\alpha) = K_{Da}(\alpha)$ if $\underline{\alpha} = \tilde{q} \ll 1$ and $\mu\bar{\kappa} = \overline{K}_{Da}$, as shown in Figure 2.9. Under these assumptions, interpolation function $K_{Su}(\alpha)$ thus has the same shape as $K_{Da}(\alpha)$ but increases the overall Darcy interpolation as $K_{Su}(\alpha) = K_{Da}(\alpha)/\underline{\alpha} \gg K_{Da}(\alpha)$. A more severe penalization on intermediate designs where $\alpha \approx 0.5$ is thus imposed using the discretization in equation 2.60, which is however balanced by defining a precise lower bound on $\bar{\kappa}$ to sufficiently penalize flow in the solid domain in Section 2.4.

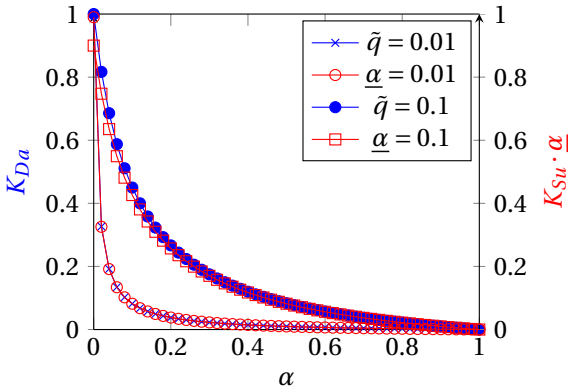


Figure 2.9: Plot of the interpolation function $K_{Da}(\alpha)$ in Equation 2.61 and scaled interpolation function $K_{Su}(\alpha) \cdot \underline{\alpha}$ in Equation 2.63 for $\mu\bar{\kappa} = \overline{K}_{Da} = 1$.

2.3.2. DISCRETIZED CONTINUITY EQUATION

The continuity equation is needed to close the equations and is thus discretized on control volume Ω_p^c in Figure 2.10 with boundary $\Gamma_p^c = \overline{\Omega}_p^c \setminus \Omega_p^c = \Gamma_p^N \cup \Gamma_p^E \cup \Gamma_p^S \cup \Gamma_p^W$. The continuity equation is integrated over the control volume and the divergence theorem is applied such that:

$$\int_{\Omega_p^c} \nabla \cdot \mathbf{v} d\Omega = \int_{\Gamma_p^c} \mathbf{v} \cdot \boldsymbol{n} d\Gamma = \overbrace{\Delta x v_p^N}^{\Gamma_p^N} - \overbrace{\Delta x v_p^S}^{\Gamma_p^S} + \overbrace{\Delta y u_p^E}^{\Gamma_p^E} - \overbrace{\Delta y u_p^W}^{\Gamma_p^W}. \tag{2.65}$$

The continuity equation and its discretization are the same for both the VANS and NSDP equations.

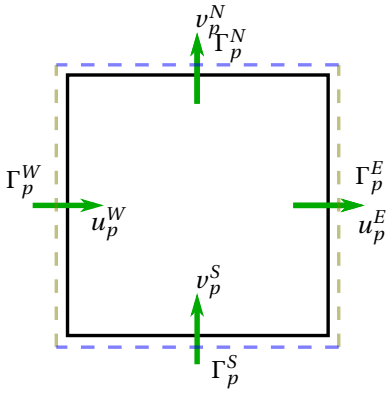


Figure 2.10: The control volume Ω_p^c for the continuity equation with relevant DOFs, and boundary $\Gamma_p^c = \overline{\Omega}_p^c \setminus \Omega_p^c = \Gamma_p^N \cup \Gamma_p^E \cup \Gamma_p^S \cup \Gamma_p^W$.

2.4. THE DARCY PENALIZATION

The question of choosing the correct $\bar{\kappa}$ is often a difficult one: setting it too low results in spurious flow through the solid domain while setting it too high may cause ill-convergence of the optimization procedure (Kreissl & Maute, 2012). As a guideline the maximum penalization is often related to the Darcy number Da (Olesen et al., 2006):

$$Da = \frac{\mu}{\overline{K} L^2}, \tag{2.66}$$

where L is a characteristic length scale of the system. In porous flow modeling, the Darcy number represents the permeability of a porous medium and a low Darcy number is related to impermeable porous structures. Subsequently, it is stated that impermeable "solids" have low Darcy numbers $Da \leq 10^{-5}$ which is used to define the inverse permeability of the solid $\overline{K} = \mu L^{-2} Da^{-1}$, and flow in the porous solid domain is penalized using Darcy penalization:

$$-\overline{K} \cdot \mathbf{v}. \tag{2.67}$$

However, this leaves the question which length scale L to use in a changing topology. Using an inlet diameter may result in significantly lower $\overline{K}$ than using the diameter of a narrow channel generated by the optimization procedure. Moreover, Darcy number Da is often either not low enough causing much flow leakage or too low resulting in inferior local optima and subsequently requires some tuning before optimization. We thus aim to define a lower bound on $\bar{\kappa}$ to penalize flow in the solid domain sufficiently. In Sections 2.4.1 and 2.4.2 the bounds will be derived for the VANS and NSDP equations respectively, after which Section 2.4.3 will give an overview and discussion of all derived bounds.

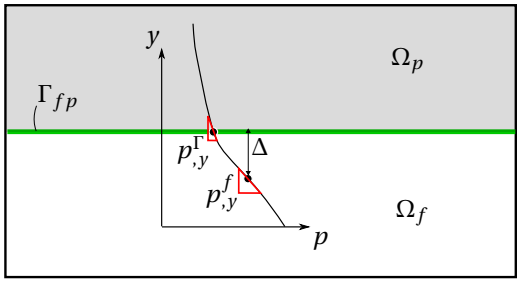


Figure 2.11: The intrinsic pressure field orthogonal to a horizontal wall which is at least C^1-continuous. For $\Delta \to 0$ the pressure gradient in the fluid ($p^f_{,y}$) will thus approach the pressure gradient at the porous interface $p^{\Gamma}_{,y} = p^f_{,y}$ at Γ_{fp}

To define the lower bounds on the penalization, the horizontal solid/fluid interface Γ_{fp} in Figure 2.11 is investigated. Although a horizontal interface is used, the actual orientation of the interface is irrelevant to the derivation. On the interface, the pressure field is examined, where we remind the reader that we are actually dealing with the intrinsic pressure average $\langle p \rangle^{i\phi}$. If we assume that $\langle p \rangle^{i\phi}$ is a good representation and substitute of the actual pore scale pressure p^{pore}, as stated by Equation 2.5, we may relate properties of the averaged and pore scale pressures. Of the pore scale pressure field, we know that it is at least C^1-continuous, and assume this to also be the case for intrinsic pressure average $\langle p \rangle^{i\phi}$. Subsequently, the pressure gradient orthogonal to the interface ($p_{,y}$ for Γ_{fp} in Figure 2.11) is assumed to be at least C^0-continuous. In the remaining text we will return to the convention of writing intrinsic pressure average $\langle p \rangle^{i\phi}$ as p, and superscripts $\square^{\Gamma}$ and $\square^{f}$ will be used to denote porous quantities on the interface and quantities in the fluid domain respectively. Due to the continuity, the pressure gradient at porous fluid interface Γ_{fp} ($p^{\Gamma}_{,y}$), and the gradient approaching Γ_{fp} from the fluid domain ($p^f_{,y}$) should thus be continuous at the interface ($p^{\Gamma}_{,y} = p^f_{,y}$ at Γ_{fp}). Consequently, the order of these terms at Γ_{fp} should be equal $\mathcal{O}\left(p^f_{,y}\right) = \mathcal{O}\left(p^{\Gamma}_{,y}\right)$. An order of magnitude analysis will be performed on these terms to derive a lower bound on the order of magnitude of $\bar{\kappa}$. To derive the bound, we aim to ensure no flow penetration at the solid/fluid interface Γ_{fp}.

2.4.1. BOUNDS ON THE DARCY PENALIZATION FOR THE VANS EQUATIONS

To derive the VANS bounds, firstly the VANS v-momentum equation found in Equation 2.31 is used to define a general equation for $p_{,y}$:

$$p_{,y} = -\rho \frac{\mathbf{v}}{\alpha} \cdot \nabla \frac{v}{\alpha} + \frac{\mu}{\alpha} \nabla^2 v - \frac{\mu}{\alpha} \nabla \alpha \cdot \nabla \frac{v}{\alpha} - \frac{\mu \kappa v}{\alpha^2}, \tag{2.68}$$

where Equation 2.31 is reordered and divided by α as it contains the pressure penalization $\alpha p_{,y}$. Subsequently, we make the following assumptions to define $p^{\Gamma}_{,y}$:

1. The penalization at the interface is interpolate as $\kappa^\Gamma = \bar{\kappa}(1-\alpha^\Gamma)$.

2. We assume the second Brinkman correction removes fluid domain viscous forces from the interface as explained and shown in Sections 2.2.3, 2.3.1, and thus neglect it and any contribution of fluid domain flow (v^f) to the viscous forces at the interface.

Which results in a pressure gradient at the interface as:

$$p^\Gamma_{,y} = -\rho \frac{\mathbf{v}^\Gamma}{\alpha^\Gamma} \cdot \nabla \frac{v^\Gamma}{\alpha^\Gamma} + \frac{\mu}{\alpha^\Gamma} \nabla^2 v^\Gamma - \frac{\mu \bar{\kappa}(1-\alpha^\Gamma) v^\Gamma}{\alpha^{\Gamma 2}}, \tag{2.69}$$

Subsequently, we make the following assumptions to define $p^f_{,y}$:

1. No flow penalization is applied in the fluid domain and $\kappa^f = 0$.

2. Fluid domain volume fraction $\alpha^f = 1$ is constant and thus $\nabla \alpha^f = 0$.

Which results in a pressure gradient in the fluid domain as:

$$p^f_{,y} = -\rho \mathbf{v}^f \cdot \nabla v^f + \mu \nabla^2 v^f, \tag{2.70}$$

The magnitudes of the different terms in Equations 2.69 and 2.70 are related by performing an order analysis on the discretized equations. In the order analysis we approximate the magnitude of gradients as $\mathcal{O}(\nabla\Psi) = \mathcal{O}\left(\Delta\Psi/\Delta x + \Delta\Psi/\Delta y\right) = \mathcal{O}(\Delta\Psi/h)$ where $h \approx \Delta x \approx \Delta y$ is the element size. The magnitude of the gradient in the inertial term in Equation 2.69 is thus approximated as:

$$\begin{aligned} &\mathcal{O}\left(\nabla \frac{v^\Gamma}{\alpha^\Gamma}\right) = \mathcal{O}\left(\frac{\nabla(v^\Gamma)}{\alpha^\Gamma} - \frac{v^\Gamma}{\alpha^{\Gamma 2}} \nabla \alpha^\Gamma\right) \\ &= \mathcal{O}\left(\frac{\Delta v^\Gamma}{h\alpha^\Gamma} - \frac{v^\Gamma}{\alpha^{\Gamma 2}} \frac{\Delta \alpha^\Gamma}{h}\right) = \mathcal{O}\left(\frac{v^\Gamma}{h\alpha^\Gamma}\left(1 - \frac{\Delta\alpha^\Gamma}{\alpha^\Gamma}\right)\right) = \mathcal{O}\left(\frac{v^\Gamma}{h\alpha^\Gamma}\left(1 - \frac{1}{\alpha^\Gamma}\right)\right), \end{aligned} \tag{2.71}$$

where we used $\Delta\alpha^\Gamma = 1$ as we are investigating the porous/fluid (0/1) interface and approximate the flow magnitude using the flow itself $\mathcal{O}\left(\Delta v^\Gamma\right) = \mathcal{O}\left(v^\Gamma\right)$. Subsequently, the magnitude of the flow velocity is approximated as $\mathcal{O}\left(\mathbf{v}^\Gamma\right) = \mathcal{O}\left(|\,\mathbf{v}^\Gamma\,|\right) \approx \mathcal{O}\left(u^\Gamma + v^\Gamma\right)$, such that the magnitude of the inertial term in Equation 2.69 can be approximated as:

$$\mathcal{O}\left(\rho \frac{\mathbf{v}^\Gamma}{\alpha^\Gamma} \cdot \nabla \frac{v^\Gamma}{\alpha}\right) = \mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^\Gamma\,|\,v^\Gamma}{\alpha^\Gamma h}\left(1 - \frac{1}{\alpha^\Gamma}\right)\right). \tag{2.72}$$

Moreover, as there are no difficult terms present in the Darcy penalization in Equation 2.69, its order is estimated as:

$$\mathcal{O}\left(\frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^{\Gamma 2}}\right). \tag{2.73}$$

Diffusive terms are approximated as $\mathcal{O}\left(\nabla^2\Psi\right)=\mathcal{O}\left(\Delta\Psi/\Delta x^2+\Delta\Psi/\Delta y^2\right)=\mathcal{O}\left(\Delta\Psi/h^2\right)$ such that the magnitudes of the viscous terms in Equations 2.69 and 2.70 are approximated as:

$$\mathcal{O}\left(\frac{\mu}{\alpha^\Gamma}\nabla^2 v^\Gamma\right)=\mathcal{O}\left(\frac{\mu v^\Gamma}{\alpha^\Gamma h^2}\right), \qquad \mathcal{O}\left(\mu\nabla^2 v^f\right)=\mathcal{O}\left(\frac{\mu v^f}{h^2}\right), \tag{2.74}$$

where we used $\mathcal{O}\left(\Delta v^\Gamma\right)=\mathcal{O}\left(v^\Gamma\right)$. Lastly, the inertial term in Equation 2.70 is estimated along the same lines as Equation 2.72:

$$\mathcal{O}\left(\rho\mathbf{v}^f\nabla v^f\right)=\mathcal{O}\left(\frac{\rho\mid\mathbf{v}^f\mid v^f}{h}\right), \tag{2.75}$$

where $\mathcal{O}\left(\mid\mathbf{v}^f\mid\right)\approx\mathcal{O}\left(u^f+v^f\right)$. The magnitudes of the flow at the porous interface/in the fluid domain are thus related by approximating the order of the pressure gradients using Equations 2.72, 2.73, 2.74, 2.75:

$$\begin{aligned}\mathcal{O}\left(p^\Gamma_{,y}\right)&=\mathcal{O}\left(-\frac{\rho\mid\mathbf{v}^\Gamma\mid v^\Gamma}{\alpha^\Gamma h}\left(1-\frac{1}{\alpha^\Gamma}\right)+\frac{\mu v^\Gamma}{\alpha^\Gamma h^2}-\frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^{\Gamma 2}}\right),\\ \mathcal{O}\left(p^f_{,y}\right)&=\mathcal{O}\left(-\frac{\rho\mid\mathbf{v}^f\mid v^f}{h}+\frac{\mu v^f}{h^2}\right),\end{aligned} \tag{2.76}$$

and by using continuity requirement $\mathcal{O}\left(p^f_{,y}\right)=\mathcal{O}\left(p^\Gamma_{,y}\right)$:

$$\mathcal{O}\left(-\frac{\rho\mid\mathbf{v}^\Gamma\mid v^\Gamma}{\alpha^\Gamma h}\left(1-\frac{1}{\alpha^\Gamma}\right)+\frac{\mu v^\Gamma}{\alpha^\Gamma h^2}-\frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^{\Gamma 2}}\right)=\mathcal{O}\left(-\frac{\rho\mid\mathbf{v}^f\mid v^f}{h}+\frac{\mu v^f}{h^2}\right). \tag{2.77}$$

To extract bounds on the penalization, an elemental Reynolds number is introduced to measure the respective relevance of the inertial and viscous forces as:

$$Re^e=\frac{\rho\mid\mathbf{v}^f\mid h}{\mu}, \tag{2.78}$$

where we make the important note that it is dependent on mesh size h and not on characteristic length L. Two main cases are examined based on whether element scale viscous ($Re^e\ll 1$) or inertial ($Re^e\gg 1$) forces are dominant. Subsequently, bounds on the penalization will be constructed by aiming to stop flow from penetrating into the solid domain through the solid/fluid interface. We will quantify flow penetrating the interface relative to the flow in the fluid domain via flow reduction v^Γ/v^f. If a relatively small amount of flow penetrates the interface, the order of the flow reduction becomes:

$$\mathcal{O}\left(\frac{v^\Gamma}{v^f}\right)<1, \tag{2.79}$$

which will be used to derive bounds on $\mathcal{O}\left(\bar{\kappa}\right)$.

VANS BOUNDS: DOMINANT VISCOUS FORCES

If the viscous term is dominant ($Re^e \ll 1$), the inertial terms are neglected and Equation 2.77 is simplified as:

$$\mathcal{O}\left(\frac{\mu v^\Gamma}{\alpha^\Gamma h^2} - \frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^{\Gamma 2}}\right) = \mathcal{O}\left(\frac{\mu v^f}{h^2}\right), \tag{2.80}$$

where either the first or the second term on the left-hand side is dominant. If the first term $(\mu v^\Gamma)/(\alpha^\Gamma h^2)$ is dominant the main mechanism for flow reduction is the pressure penalization which causes the division by α^Γ in Equation 2.68. If the second term $(\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma)/(\alpha^{\Gamma 2})$ is dominant the main mechanism for flow reduction is the Darcy penalization. We measure the relative dominance by dividing the first term with the second term:

$$r_V^l = \left|\frac{\mu v^\Gamma}{\alpha^\Gamma h^2}\frac{\alpha^{\Gamma 2}}{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}\right| = \left|\frac{\alpha^\Gamma}{h^2\bar{\kappa}(1-\alpha^\Gamma)}\right|, \tag{2.81}$$

and examine the two scenarios where either $r_V^l > 1$ or $r_V^l < 1$:

- **Dominant Darcy penalization** ($r_V^l < 1$):

  If the second term on the left-hand side of Equation 2.80 is dominant, the order analysis reduces to:

  $$\mathcal{O}\left(\frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^{\Gamma 2}}\right) = \mathcal{O}\left(\frac{\mu v^f}{h^2}\right), \tag{2.82}$$

  which is rewritten to find the flow reduction at the interface:

  $$\mathcal{O}\left(\frac{v^\Gamma}{v^f}\right) = \mathcal{O}\left(\frac{\alpha^{\Gamma 2}}{h^2\bar{\kappa}(1-\alpha^\Gamma)}\right) < 1. \tag{2.83}$$

  The bound on the flow reduction is subsequently rewritten to find a lower bound on $\bar{\kappa}$ for $Re^e \ll 1$ and $r_V^l < 1$:

  $$\mathcal{O}(\bar{\kappa}) > \mathcal{O}\left(\frac{\alpha^{\Gamma 2}}{h^2(1-\alpha^\Gamma)}\right). \tag{2.84}$$

- **Dominant pressure penalization** ($r_V^l > 1$):

  If the first term on the right-hand side of Equation 2.80 is dominant, the order analysis reduces to:

  $$\mathcal{O}\left(\frac{\mu v^\Gamma}{\alpha^\Gamma h^2}\right) = \mathcal{O}\left(\frac{\mu v^f}{h^2}\right), \tag{2.85}$$

  which can be rewritten to find the flow reduction as:

  $$\mathcal{O}\left(\frac{v^\Gamma}{v^f}\right) = \mathcal{O}\left(\alpha^\Gamma\right). \tag{2.86}$$

  Flow is thus automatically reduced for $\alpha^\Gamma \ll 1$ ($\mathcal{O}\left(\alpha^\Gamma\right) < 1$) if the first term on the left-hand side of Equation 2.80 is dominant.

VANS BOUNDS: DOMINANT INERTIAL FORCES

In the second case, inertial terms are dominant ($Re^e \gg 1$) and the viscous forces can be neglected, resulting in:

$$\mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^\Gamma\,|\,v^\Gamma}{\alpha^\Gamma h}\left(1-\frac{1}{\alpha^\Gamma}\right)+\frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^{\Gamma 2}}\right)=\mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^f\,|\,v^f}{h}\right), \tag{2.87}$$

where either the first or second term on the left-hand side is dominant. If the first term:

$$\frac{\rho\,|\,\mathbf{v}^\Gamma\,|\,v^\Gamma}{\alpha^\Gamma h}\left(1-\frac{1}{\alpha^\Gamma}\right) \tag{2.88}$$

is dominant, the main mechanisms for flow reduction are the pressure penalization which causes the first division by α^Γ in Equation 2.68 and the inertial penalization which results in the $(1-1/\alpha^\Gamma)$ term as can be seen in Equation 2.71. If the second term $(\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma)/(\alpha^{\Gamma 2})$ is dominant, the main mechanism for flow reduction is again the Darcy penalization. We measure the relative dominance by dividing the first term with the second term:

$$r_V^h=\left|\frac{\rho\,|\,\mathbf{v}^\Gamma\,|\,v^\Gamma}{\alpha^\Gamma h}\left(1-\frac{1}{\alpha^\Gamma}\right)\frac{\alpha^{\Gamma 2}}{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}\right|=\left|\frac{\rho\,|\,\mathbf{v}^\Gamma\,|}{h\mu\bar{\kappa}}\right|. \tag{2.89}$$

and again examine two scenarios where either $r_V^h>1$ or $r_V^h<1$:

- **Dominant Darcy penalization** ($r_V^h<1$):

  If the second term is dominant the order analysis reduces to:

  $$\mathcal{O}\left(\frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^{\Gamma 2}}\right)=\mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^f\,|\,v^f}{h}\right), \tag{2.90}$$

  which can be rewritten to find the flow reduction at the interface:

  $$\mathcal{O}\left(\frac{v^\Gamma}{v^f}\right)=\mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^f\,|\,\alpha^{\Gamma 2}}{h\mu\bar{\kappa}(1-\alpha^\Gamma)}\right)=\mathcal{O}\left(\frac{Re^e\alpha^{\Gamma 2}}{h^2\bar{\kappa}(1-\alpha^\Gamma)}\right)<1, \tag{2.91}$$

  from which we can extract a bound on the penalization for $Re^e \gg 1$ and $r_V^h<1$ as:

  $$\mathcal{O}(\bar{\kappa})>\mathcal{O}\left(\frac{Re^e\alpha^{\Gamma 2}}{h^2(1-\alpha^\Gamma)}\right). \tag{2.92}$$

- **Dominant pressure and inertial penalizations** ($r_V^h>1$):

  If however the first term on the left-hand side of Equation 2.87 is dominant the order analysis reduces to:

  $$\mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^\Gamma\,|\,v^\Gamma}{\alpha^\Gamma h}\left(1-\frac{1}{\alpha^\Gamma}\right)\right)=\mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^f\,|\,v^f}{h}\right). \tag{2.93}$$

The equation is subsequently rewritten to find that for $\alpha^\Gamma \ll 1$ flow at the interface is automatically lower than flow in the fluid domain:

$$\mathcal{O}\left(\frac{|\mathbf{v}^\Gamma|v^\Gamma}{|\mathbf{v}^f|v^f}\right) \approx \mathcal{O}\left(\frac{{v^\Gamma}^2}{{v^f}^2}\right) = \mathcal{O}\left(\frac{\alpha^\Gamma}{1-\frac{1}{\alpha^\Gamma}}\right) < 1. \tag{2.94}$$

For both Equations 2.86 and 2.94 we assumed that $\alpha^\Gamma \ll 1$ to achieve some flow reduction. In practice however, we use $\alpha^\Gamma = \alpha^P \approx 0.5$ which is just below 1. We will examine these assumptions and resulting errors in Section 2.4.4.

2.4.2. BOUNDS ON THE DARCY PENALIZATION FOR THE NSDP EQUATIONS

For the NSDP equations we can follow a similar procedure as for the VANS equations. In the fluid domain $p^f_{,y}$ is defined as in Equation 2.70. In the porous domain $p^\Gamma_{,y}$ is defined using Equation 2.30 as:

$$p^\Gamma_{,y} = -\rho\mathbf{v}^\Gamma \cdot \nabla v^\Gamma + \mu\nabla^2 v^\Gamma - \frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^\Gamma}, \tag{2.95}$$

where κ^Γ is interpolated using $K_{Su}(\alpha^\Gamma)$ in Equation 2.62. The orthogonal pressure gradient $p_{,y}$ is again assumed to be continuous, and an order analysis is performed resulting in an equation similar to Equation 2.77:

$$\mathcal{O}\left(-\frac{\rho|\mathbf{v}^\Gamma|v^\Gamma}{h} + \frac{\mu v^\Gamma}{h^2} - \frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^\Gamma}\right) = \mathcal{O}\left(-\frac{\rho|\mathbf{v}^f|v^f}{h} + \frac{\mu v^f}{h^2}\right). \tag{2.96}$$

NSDP BOUNDS: DOMINANT VISCOUS FORCES

If $Re^e \ll 1$ and viscous forces are dominant inertial forces are neglected such that Equation 2.96 reduces to:

$$\mathcal{O}\left(\frac{\mu v^\Gamma}{h^2} - \frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^\Gamma}\right) = \mathcal{O}\left(\frac{\mu v^f}{h^2}\right). \tag{2.97}$$

We first examine the case where the first term on the left-hand side is dominant, resulting in:

$$\mathcal{O}\left(\frac{\mu v^\Gamma}{h^2}\right) = \mathcal{O}\left(\frac{\mu v^f}{h^2}\right), \tag{2.98}$$

which can be rewritten to find that no flow reduction takes place:

$$\mathcal{O}\left(\frac{v^\Gamma}{v^f}\right) = \mathcal{O}(1). \tag{2.99}$$

The only mechanism for flow reduction in the NSDP equations is thus the Darcy penalization and we assume that when an effective Darcy penalization is applied $\mathcal{O}(v^\Gamma) < \mathcal{O}(v^f)$ and we may neglect the first term on the left-hand side of Equation 2.97:

$$\mathcal{O}\left(-\frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^\Gamma}\right) = \mathcal{O}\left(\frac{\mu v^f}{h^2}\right). \tag{2.100}$$

After which the flow reduction:

$$\mathcal{O}\left(\frac{v^\Gamma}{v^f}\right) = \mathcal{O}\left(\frac{\alpha^\Gamma}{h^2\bar{\kappa}(1-\alpha^\Gamma)}\right) < 1, \tag{2.101}$$

is used to derive a lower bound on the penalization as:

$$\mathcal{O}(\bar{\kappa}) > \mathcal{O}\left(\frac{\alpha^\Gamma}{h^2(1-\alpha^\Gamma)}\right). \tag{2.102}$$

NSDP BOUNDS: DOMINANT INERTIAL FORCES

If $Re^e \gg 1$ and inertial forces are dominant, the order analysis reduces to:

$$\mathcal{O}\left(-\frac{\rho\,|\,\mathbf{v}^\Gamma\,|\,v^\Gamma}{h} - \frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^\Gamma}\right) = \mathcal{O}\left(-\frac{\rho\,|\,\mathbf{v}^f\,|\,v^f}{h}\right), \tag{2.103}$$

As was the case for low Re^e, the only mechanism for flow reduction is the Darcy penalization and we neglect the first term on the left-hand side by assuming that $\mathcal{O}\left(|\,\mathbf{v}^\Gamma\,|\,v^\Gamma\right) < \mathcal{O}\left(|\,\mathbf{v}^f\,|\,v^f\right)$, such that the inertial term at the interface can be neglected:

$$\mathcal{O}\left(\frac{\mu\bar{\kappa}(1-\alpha^\Gamma)v^\Gamma}{\alpha^\Gamma}\right) = \mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^f\,|\,v^f}{h}\right). \tag{2.104}$$

Subsequently, the analysis is again rewritten to find the order of flow reduction:

$$\mathcal{O}\left(\frac{v^\Gamma}{v^f}\right) = \mathcal{O}\left(\frac{\rho\,|\,\mathbf{v}^f\,|\,\alpha^\Gamma}{h\mu\bar{\kappa}(1-\alpha^\Gamma)}\right) = \mathcal{O}\left(\frac{Re^e\alpha^\Gamma}{\bar{\kappa}h^2(1-\alpha^\Gamma)}\right) < 1. \tag{2.105}$$

After which a lower bound on the Darcy penalization is defined as:

$$\mathcal{O}(\bar{\kappa}) > \mathcal{O}\left(\frac{Re^e\alpha^\Gamma}{h^2(1-\alpha^\Gamma)}\right). \tag{2.106}$$

For the NSDP equations we thus derive bounds on the penalization under the assumption that flow reduction is a fact ($\mathcal{O}\left(v^\Gamma\right) < \mathcal{O}\left(v^f\right)$ and $\mathcal{O}\left(|\,\mathbf{v}^\Gamma\,|\,v^\Gamma\right) < \mathcal{O}\left(|\,\mathbf{v}^f\,|\,v^f\right)$). These assumptions however only hold when the appropriate penalization's from Equations 2.105 and 2.102 are used. If these appropriate penalizations are not used there is no other mechanism for flow reduction and errors due to flow leakage will become large.

2.4.3. OVERVIEW AND DISCUSSION OF BOUNDS ON THE PENALIZATION

Both the VANS and NSDP equations thus have bounds on the penalization dependent on the elemental Reynolds number as defined in Equation 2.78. Moreover, the VANS equations may have an additional dependence on measurements r_V^l and r_V^h defined in Equations 2.81 and 2.89 which measure the dominant mechanism for flow reduction. We will first define bounds on the penalization assuming that the Darcy penalization is the dominant mechanism for flow reduction ($r_V^l < 1$ and $r_V^h < 1$) resulting in the VANS bounds in Equations 2.84 and 2.92. In Section 2.4.4 we will come back to this assumption

and show that although although the Darcy penalization is dominant the pressure penalization also plays a significant role in flow reduction. Subsequently, the VANS bounds in Equations 2.84 and 2.92 and NSDP bounds in Equations 2.102 and 2.106 are simplified by using the elemental Reynolds number Re^e and defining inverse elemental surface area:

$$H^e = \frac{1}{h^2}, \tag{2.107}$$

resulting in the lower bounds on $\mathcal{O}(\bar{\kappa})$ as summarized in Table 2.1. In practice for ele-

	$Re^e \ll 1$	$Re^e \gg 1$
NSDP	$\mathcal{O}\left(\frac{\alpha^\Gamma}{1-\alpha^\Gamma} H^e\right)$	$\mathcal{O}\left(\frac{\alpha^\Gamma}{1-\alpha^\Gamma} H^e Re^e\right)$
VANS	$\mathcal{O}\left(\frac{\alpha^{\Gamma 2}}{1-\alpha^\Gamma} H^e\right)$	$\mathcal{O}\left(\frac{\alpha^{\Gamma 2}}{1-\alpha^\Gamma} H^e Re^e\right)$

Table 2.1: The lower bounds on $\mathcal{O}(\bar{\kappa})$ at boundary Γ_{fp} defined using an inverse elemental surface area and elemental Reynolds number.

ments at the interface volume fraction $\alpha^\Gamma = \alpha^P \approx 0.5$ is used. In Table 2.1 the magnitudes depend on $\alpha^\Gamma = 0.5$ as $\alpha^\Gamma/(1-\alpha^\Gamma) = 1$ or ${\alpha^\Gamma}^2/(1-\alpha^\Gamma) = 0.5$. For the order of magnitude the dependence on α^Γ can thus be neglected resulting in the same penalization for the VANS and NSDP equations. Moreover, we require $\bar{\kappa}$ to be an order of magnitude higher than the values in Table 2.1, and specify the increase in magnitude using 10^q, where q is a small whole number (generally $q = 0$, $q = 1$ or $q = 2$). The resulting values which we implement for $\bar{\kappa}$ can be found in Table 2.2.

	$Re^e \leq 1$	$Re^e > 1$
VANS/NSDP	$10^q H^e$	$10^q H^e Re^e$

Table 2.2: Definition for $\bar{\kappa}$ in a practical application defined using inverse elemental surface area H^e and elemental Reynolds number Re^e. The power q is used to increase the magnitude of the penalization, and is generally set as $q = 0$, $q = 1$ or $q = 2$.

To relate the maximum Darcy penalization found in this work to common practice we rewrite it as:

$$-\mu \frac{\kappa(\alpha)}{\alpha} \cdot \mathbf{v} = -K_h(\alpha) \cdot \mathbf{v} \tag{2.108}$$

The maximum penalization in the solid domain at $\alpha = \underline{\alpha}$ and $\kappa = \bar{\kappa}$ for $Re^e \leq 1$ is subsequently found as:

$$\overline{K}_h = \mu \frac{\bar{\kappa}}{\underline{\alpha}} = \mu \frac{10^q H^e}{\underline{\alpha}} = \mu \frac{10^q}{h^2 \underline{\alpha}} \gg \frac{\mu}{h^2}, \tag{2.109}$$

and for $Re^e > 1$ as:

$$\begin{aligned}\overline{K}_h &= \mu\frac{\bar{\kappa}}{\underline{\alpha}} = \mu\frac{10^q H^e Re^e}{\underline{\alpha}} = \mu\frac{10^q}{h^2\underline{\alpha}}\frac{\rho\,|\,\mathbf{v}^f\,|\,h}{\mu} \\ &= \frac{10^q\rho\,|\,\mathbf{v}^f\,|}{h\underline{\alpha}} \gg \frac{\rho\,|\,\mathbf{v}^f\,|}{h},\end{aligned} \tag{2.110}$$

where we treat $10^q/\underline{\alpha}$ as a factor which scales the maximum penalization similar to the factor $1/Da \gg 1$ which scales the commonly used maximum penalization by Olesen et al. (2006):

$$\overline{K} = \frac{\mu}{L^2 Da} \gg \frac{\mu}{L^2}. \tag{2.111}$$

Comparing the maximum penalization for $Re^e \leq 1$ in Equation 2.109 to the common penalization in Equation 2.111 they seem similar. However, whereas the common penalization is dependent on characteristic length L which may change for changing topologies, our new penalization is dependent on mesh size h which allows us to accurately predict errors as will be discussed in Section 2.4.4 and shown in Section 2.6. Moreover, the bounds for the Darcy penalization are also dependent on an elemental Reynolds number. This is not completely new as Kondoh et al. (2012) and Alexandersen et al. (2013) already implement a Reynolds dependent penalization. However, contrary to those works, the penalization in this work is dependent on an elemental Reynolds number and h instead of the global Reynolds number:

$$Re = \frac{\rho V L}{\mu}. \tag{2.112}$$

Whereas a global Reynolds number is computed using reference length L, the elemental Reynolds number in Equation 2.78 is dependent on h which is often much lower ($h \ll L$) resulting in much lower elemental Reynolds numbers ($Re^e \ll Re$). Moreover, the penalization definitions by Kondoh et al. (2012) and Alexandersen et al. (2013) are defined for non-dimensional Navier-Stokes equations which impacts their interpretation and comparision to common practice as discussed in Appendix 2.B.

We note that the elemental Reynolds number and thus the Darcy penalization remain dependent on an *a priori* estimate of $|\,\mathbf{v}^f\,|$ which may cause problems in changing topologies when this estimate is erroneous. A solution to this problem could be a penalization dependent on actual local flow magnitude $|\,\mathbf{v}\,|$. However, as this requires the absolute flow magnitude this would introduce discontinuities in the gradients of the model used. Furthermore, this approach would be similar to the Forchheimer penalization introduced by Alonso and Silva (2022) who deal with discontinuous gradients by using automatic differentiation.

2.4.4. A PRIORI ERROR ESTIMATION

Using the bounds on the penalization as provided in the previous section, we may estimate flow leakage in the porous domain for low and high Reynolds flow. For low $Re^e \leq 1$ we set $\bar{\kappa} = 10^q H^e = 10^q h^{-2}$, which can be substituted into the estimated flow reduction

at the interface in Equations 2.101 and 2.83, respectively:

$$\begin{aligned}\mathscr{O}\left(\frac{v^{\Gamma}}{v^{f}}\right)_{\text{NSDP}} &= \mathscr{O}\left(\frac{\alpha^{\Gamma}}{1-\alpha^{\Gamma}}10^{-q}\right) = \mathscr{O}\left(10^{-q}\right),\\ \mathscr{O}\left(\frac{v^{\Gamma}}{v^{f}}\right)_{\text{VANS}} &= \mathscr{O}\left(\frac{\alpha^{\Gamma 2}}{1-\alpha^{\Gamma}}10^{-q}\right) = \mathscr{O}\left(0.5\cdot 10^{-q}\right),\end{aligned} \tag{2.113}$$

when $\alpha^{\Gamma} \approx 0.5$ at the solid/fluid interface. For the same value of q, flow at the interface should thus be reduced by a similar factor in the VANS as well as the NSDP equations. However, we speculate that the same flow reduction may also be used in the solid domain where $\alpha \approx \underline{\alpha} \ll 1$:

$$\begin{aligned}\mathscr{O}\left(\frac{v^{s}}{v^{f}}\right)_{\text{NSDP}} &= \mathscr{O}\left(\frac{\underline{\alpha}}{1-\underline{\alpha}}10^{-q}\right) = \mathscr{O}\left(\underline{\alpha}\cdot 10^{-q}\right),\\ \mathscr{O}\left(\frac{v^{s}}{v^{f}}\right)_{\text{VANS}} &= \mathscr{O}\left(\frac{\underline{\alpha}^{2}}{1-\underline{\alpha}}10^{-q}\right) = \mathscr{O}\left(\underline{\alpha}^{2}\cdot 10^{-q}\right),\end{aligned} \tag{2.114}$$

for the NSDP and VANS equations, respectively, where v^s represents the flow in the solid domain. Flow in the solid domain computed using the VANS equations may thus be decreased by an additional factor $\underline{\alpha}$ with respect to the NSDP equations for low Reynolds flow. This stronger suppression of flow leakage originates from the penalization of the pressure gradient $\alpha\nabla p$ in the VANS equations which added an extra division by α in the definition of $p^{\Gamma}_{,y}$ in Equation 2.68. Furthermore, the estimates for flow reduction at the interface have been derived in Sections 2.4.1 and 2.4.2, while the estimates in the porous domain are a speculation on the extensibility of Equation 2.113. The extensibility of flow reduction in the porous domain is based on the idea that the pressure gradient should not only be continuous across the fluid domain and porous/fluid interface, but also across the porous/fluid interface and porous domain. Moreover, for high $Re^e > 1$ we set $\bar{\kappa} = 10^q H^e Re^e = 10^q Re^e h^{-2}$, which can be substituted into Equations 2.91 and 2.105 respectively resulting in the exact same flow reductions as in Equations 2.113 and 2.114.

If an insufficient penalization is chosen for the NSDP equations ($q < 0$) flow leakage may introduce significant errors. However, for the VANS equations, there is still the possibility that $r^l_V > 1$ or $r^h_V > 1$ resulting in the Darcy penalization not being the dominant mechanism for flow reduction. The dominant mechanism for flow reduction is investigated by substituting $\alpha^{\Gamma} \approx 0.5$ and $\bar{\kappa}$ in Equations 2.81 and 2.89:

$$\begin{aligned}r^l_V &= \left|\frac{\alpha^{\Gamma}}{h^2\bar{\kappa}(1-\alpha^{\Gamma})}\right| \approx \left|\frac{0.5}{h^2 10^q h^{-2} 0.5}\right| = 10^{-q},\\ r^h_V &= \left|\frac{\rho\,|\,\mathbf{v}^{\Gamma}\,|}{h\mu\bar{\kappa}}\right| = \left|\frac{\rho\,|\,\mathbf{v}^{\Gamma}\,|}{h\mu}\,\frac{h\mu}{10^q\rho\,|\,\mathbf{v}^{f}\,|}\right| = \left|\frac{|\,\mathbf{v}^{\Gamma}\,|}{|\,\mathbf{v}^{f}\,|}10^{-q}\right|,\end{aligned} \tag{2.115}$$

for $Re^e < 1$ ($\bar{\kappa} = 10^q h^{-2}$) and $Re^e > 1$ ($\bar{\kappa} = 10^q Re^e H^e = 10^q \rho\left|\mathbf{v}^f\right| h^{-1}\mu^{-1}$) respectively. In the low elemental Reynolds case, the dominant mechanism for flow reduction is thus completely determined by q. If $q < 0$ and as a result $r^l_V > 1$ we do not satisfy the condition for flow reduction through the Darcy penalization in Equation 2.102, but still manage

some flow reduction through the condition in Equation 2.86:

$$\mathscr{O}\left(\frac{\nu^{\Gamma}}{\nu^{f}}\right)=\mathscr{O}\left(\alpha^{\Gamma}\right). \tag{2.116}$$

This will not result in much flow reduction at the solid/fluid interface where $\alpha^{\Gamma}\approx 0.5$. However, within the solid domain where $\alpha=\underline{\alpha}\ll 1$ this flow reduction mechanism might have significant effects. To ensure sufficient flow penalization, in this work we will use $q\geq 0$. If $Re^{e}>1$, the problem is more complicated as the dominant mechanism for flow reduction depends on the flow reduction itself. However, if we substitute the flow reduction ($\left|\mathbf{v}^{\Gamma}\right|/\left|\mathbf{v}^{f}\right|\approx\nu^{\Gamma}/\nu^{f}$) due to the Darcy penalization in Equations 2.113 into r_V^h in Equation 2.115 we find:

$$r_V^h=\left|\frac{|\mathbf{v}^{\Gamma}|}{|\mathbf{v}^{f}|}10^{-q}\right|=0.5\cdot 10^{-2q}. \tag{2.117}$$

Similar to the low Reynolds case, if $q<0$ it thus follows that the Darcy penalization is not the dominant flow reducing mechanism as $r_V^h>1$. Moreover, flow at the interface is not reduced as substituting $\alpha^{\Gamma}\approx 0.5$ into the flow reduction for $r_V^h>1$ in Equation 2.94 results in:

$$\mathscr{O}\left(\frac{\nu^{\Gamma 2}}{\nu^{f 2}}\right)=\mathscr{O}\left(\frac{\alpha^{\Gamma}}{1-\frac{1}{\alpha^{\Gamma}}}\right)=\mathscr{O}\left(1\right). \tag{2.118}$$

However, if we again speculate on the extensibility of these formulations to the porous domain where $\alpha=\underline{\alpha}\ll 1$, we find a significant solid domain flow reduction of:

$$\mathscr{O}\left(\frac{\nu^{s2}}{\nu^{f 2}}\right)=\mathscr{O}\left(\frac{\underline{\alpha}}{1-\frac{1}{\underline{\alpha}}}\right)\approx\mathscr{O}\left(\underline{\alpha}^{2}\right). \tag{2.119}$$

Finally, we make a note on the error in pressure, which is more difficult to estimate. In the solid domain, a non-zero pressure field will be present. However, the **intrinsic** average of the pressure field ($\langle p\rangle^{i\phi}$) is computed while the **superficial** average of the velocity field ($\langle\mathbf{v}\rangle^{s\phi}$) is computed. When no fluid is pumped into the porous domain where $\alpha\rightarrow 0$, velocity $\langle\mathbf{v}\rangle^{s\phi}$ naturally converges to zero, while $\langle p\rangle^{i\phi}$ does not necessarily converge to zero as it represents the pore scale average as explained in Section 2.2. Non-zero intrinsic pressure fields in the solid domain should thus be expected and should not be treated as erroneous. Furthermore, for both the VANS and NSDP momentum equations, the pressure gradient at every point within the fluid domain can be written as solely a function of velocity: $\nabla p=f(\mathbf{v})$, if material properties ρ, μ, $\bar{\kappa}$ and $\alpha=1$ are taken as constant. If the correct velocity field is found in the fluid domain, it follows that the correct pressure field is also computed, respective to a reference pressure in the fluid domain. As pressure gradients in the solid domain are expected, and errors in pressure are harder to quantify, we mainly focus on flow leakage as a representation of precision of the solution.

2.5. OPTIMIZATION PROBLEM AND ADJOINT SENSITIVITY ANALYSIS

Using the discretization and bounds on the Brinkman penalization the optimization problem is defined as:

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & f(\boldsymbol{u},\boldsymbol{v},\boldsymbol{p},\boldsymbol{\alpha}) \\ \text{subject to} \quad & \boldsymbol{R}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{p},\boldsymbol{\alpha}) = 0, \\ & \frac{\sum_{i=1}^{N} \alpha_i}{N} - V_f \leq 0, \end{aligned}$$

where N is the number of discrete design DOFs α_i, $\boldsymbol{R}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{p},\boldsymbol{\alpha})$ is a column containing all discretized equilibrium equations, $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{p}$, $\boldsymbol{\alpha}$ contain the discrete velocities pressures and design variables, the desired maximum fluid volume fraction is V_f and $f(\boldsymbol{u},\boldsymbol{v},\boldsymbol{p},\boldsymbol{\alpha})$ is the objective to be minimized. No additional filters such as a blurring filter or Heaviside projection are applied to the design as they are not necessary to regularize the optimization problems in this paper. The MMA algorithm by Svanberg (1987, 2004) is used to perform the optimization. Computing the sensitivities required by the MMA algorithm can be a cumbersome task in non-linear fluid problems. The main problem lies in the computation of the Jacobian matrix for the solution of the adjoint equations. In this work we use the (MATLAB, 2019) symbolic toolbox to construct discrete momentum and continuity equations, such that we are able to create functions for the Jacobian matrices before running the optimization. In essence, a similar approach as by Dilgen et al. (2018) is implemented, where automatic differentiation is used to compute the Jacobian of the residuals.

To construct the adjoint sensitivities, columns containing discrete functions for the u, v-momentum and continuity equations are defined as $\boldsymbol{R}_u(\boldsymbol{u},\boldsymbol{v},\boldsymbol{p},\boldsymbol{\alpha})$, $\boldsymbol{R}_v(\boldsymbol{u},\boldsymbol{v},\boldsymbol{p},\boldsymbol{\alpha})$, $\boldsymbol{R}_p(\boldsymbol{u},\boldsymbol{v},\boldsymbol{\alpha})$ respectively. An element of $\boldsymbol{R}_u(\boldsymbol{u},\boldsymbol{v},\boldsymbol{p},\boldsymbol{\alpha})$ is thus associated with DOF u^P and the stencil as in Figure 2.5, where both u^P and the stencil are mapped to the global mesh in Figure 2.4. Boundary conditions are applied via ghost nodes as described by Versteeg and Malalasekera (2007) and are added to the columns containing the discrete equations. All equations are gathered as $\boldsymbol{R}^\intercal = [\boldsymbol{R}_u^\intercal, \boldsymbol{R}_v^\intercal, \boldsymbol{R}_p^\intercal]$ and $\boldsymbol{U} = [\boldsymbol{u}^\intercal, \boldsymbol{v}^\intercal, \boldsymbol{p}^\intercal]$. Furthermore, objective $f(\boldsymbol{u},\mathbf{v},\boldsymbol{p},\boldsymbol{\alpha})$ is defined and used to construct the augmented objective:

$$F = f + \boldsymbol{\lambda}^\intercal \boldsymbol{R}, \tag{2.120}$$

where $\boldsymbol{\lambda}$ contains the adjoint multipliers. The sensitivities are subsequently defined as:

$$\frac{dF}{d\boldsymbol{\alpha}} = \frac{\partial f}{\partial \boldsymbol{\alpha}} + \boldsymbol{\lambda}^\intercal \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\alpha}} + \left(\frac{\partial f}{\partial \boldsymbol{U}} + \boldsymbol{\lambda}^\intercal \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} \right) \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{\alpha}}, \tag{2.121}$$

where first, the adjoint equations are solved:

$$\boldsymbol{\lambda}^\intercal = -\frac{\partial f}{\partial \boldsymbol{U}} \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}^{-1}, \tag{2.122}$$

after which the sensitivities are computed as:

$$\frac{dF}{d\boldsymbol{\alpha}} = \frac{\partial f}{\partial \boldsymbol{\alpha}} + \boldsymbol{\lambda}^\intercal \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\alpha}}. \tag{2.123}$$

The difficult part in solving the adjoint equations is to define $\partial \boldsymbol{R}/\partial \boldsymbol{\alpha}$ and the Jacobian:

$$\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} = \begin{bmatrix} \frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{u}} & \frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{v}} & \frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{R}_v}{\partial \boldsymbol{u}} & \frac{\partial \boldsymbol{R}_v}{\partial \boldsymbol{v}} & \frac{\partial \boldsymbol{R}_v}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{R}_p}{\partial \boldsymbol{u}} & \frac{\partial \boldsymbol{R}_p}{\partial \boldsymbol{v}} & 0 \end{bmatrix}, \tag{2.124}$$

as it is never explicitly formed in the SIMPLE solution algorithm. We are however able to assemble the Jacobian after solving for flow and pressure fields, by using a symbolic coding toolbox. For example, element R_u^i of $\boldsymbol{R}_u$ contains the discretized u-momentum equation in terms of the symbolic stencil variables in Figure 2.5 $\boldsymbol{U}^s = [u^P, u^N, ..., v^{NE}, ...\alpha^E, ...]^\intercal$, and derivatives can be computed using symbolic differentiation $\partial R_u^i/\partial \boldsymbol{U}^s$. As the derivatives for one stencil are the same for every element R_u^i, we only perform the symbolic differentiation for one stencil and let the symbolic coding toolbox automatically construct a vector function from $\partial R_u^i/\partial \boldsymbol{U}^s$ which takes vectors of DOFs $\boldsymbol{U}^s$ and returns vectors of $\partial R_u^i/\partial \boldsymbol{U}^s$. Both $\boldsymbol{U}^s$ and $\partial R_u^i/\partial \boldsymbol{U}^s$ can be mapped to the global mesh. The subsequent assembly procedure into $\partial \boldsymbol{R}/\partial \boldsymbol{U}$ is coded by hand and the same procedure is performed for $\partial \boldsymbol{R}/\partial \boldsymbol{\alpha}$. For this particular implementation, a symbolic toolbox which can compute derivatives and construct vector functions from symbolic equations is thus required. To confirm the adjoint sensitivities used in this work they are verified using complex step finite difference sensitivities in Appendix 2.C.

2.6. PRECISION OF THE VANS AND NSDP EQUATIONS

A precise flow solution is essential for finding precise optima when optimizing fluid problems. The precision of the NSDP and VANS solvers is therefore investigated for several flow problems, Reynolds numbers, minimal volume fraction $\underline{\alpha}$ and penalizations $\bar{\kappa}$. As precision is of most importance in the optimal black/white designs, only these designs are investigated. In Appendix 2.D we examine the effect of the second Brinkman correction and argue that flow leakage is a good measure for overall solution accuracy. Consequently, we will use flow leakage to asses solution precision in this section and the remainder of this work. To confirm the lower bound on $\bar{\kappa}$ derived in Section 2.4 a sweep on power q is performed for low elemental Reynolds numbers ($Re^e \leq 1$) in Section 2.6.1 and moderate elemental Reynolds flow ($Re^e > 1$) in Section 2.6.2. The results in these sections will be used to choose optimization settings for q and $\underline{\alpha}$ such that flow is sufficiently penalized in the optimization problems in Section 2.7. Flow is sufficiently penalized when we expect errors $e^{\mathbf{v}} \approx 10^{-1}$ and $u_l < 10^{-2}$. Error $e^{\mathbf{v}} \approx 10^{-1}$ is chosen relatively high on purpose as it represents a flow error in two cases and influences convergence behavior of the optimization procedure. The error is mainly caused by flow at those locations in the design where $\alpha^P \approx 0.5$, which are found at either the solid/fluid interface, or in the intermediate gray design during optimization where $\alpha \approx 0.5$. If flow in the intermediate gray design is penalized too much, the optimizer can be pushed into an inferior local optimum from which it may be hard to escape. Moreover, when choosing the optimization settings we put more emphasis on $u_l < 10^{-2}$ as in Appendix 2.D flow leakage is shown to correlate to errors in pressure drop (which we will optimize for in Section 2.7). Furthermore, in Section 2.6.3 the penalization for a range of Reynolds numbers is investigated.

2.6.1. LOW ELEMENTAL REYNOLDS NUMBERS

Firstly, a sweep on the penalization is performed for low Re^e (and low Re) using the problem depicted in Figure 2.12 and the parameters given in Table 2.3. We examine a

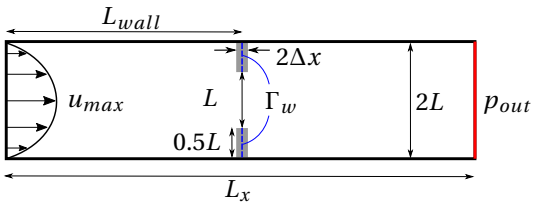


Figure 2.12: A 2D channel with parabolic inflow applied at the left inlet and constant pressure applied at the right outlet. At L_{wall} two small porous solid walls of thickness $2\Delta x$ are inserted to inhibit flow. For different mesh sizes the problem will thus slightly vary as the wall thickness changes. For low Reynolds flow the wall affects the pressure distribution to the left and is placed at the center, while for moderate Reynolds flow the wall has large wakes to the right and is placed closer to the inlet.

μ	ρ	u_{max}	p_{out}	L	L_{wall}	L_x
1 [Pas]	1 [kgm^{-3}]	1 [ms^{-1}]	1 [Pa]	1 [m]	$4L$	$8L$

Table 2.3: The material and problem parameters for the flow problem in Figure 2.13, where low Reynolds flow is investigated for low Re^e and varying q, $\underline{\alpha}$.

channel of height $2L$ where flow is obstructed by two solid porous walls of two elements thick. A parabolic flow profile with maximum velocity u_{max} is prescribed at the inlet and pressure at the outlet is fixed at p_{out}. Furthermore, as equidistant meshes are used where $\Delta x = \Delta y = h$ and four mesh refinements are examined where $h = 0.1L$, $h = 0.05L$, $h = 0.025L$, $h = 0.0125L$, the elemental Reynolds number can be computed as:

$$Re^e = \frac{\rho \left|\mathbf{v}^f\right| h}{\mu} = \frac{\rho u_{max} h}{\mu} = h, \tag{2.125}$$

where $Re^e = h < 1$ and we approximate $\left|\mathbf{v}^f\right| \approx u_{max}$. Since $Re^e = h < 1$, the maximum $\bar{\kappa}$ is computed following Table 2.2 as:

$$\bar{\kappa} = 10^q H^e = \frac{10^q}{h^2}, \tag{2.126}$$

Furthermore, using the parameters in Table 2.3, the Reynolds number is computed as

$$Re = \frac{\rho u_{max} L}{\mu} = 1. \tag{2.127}$$

To investigate the *a priori* error estimates in Section 2.4.4 the precision of the model is measured using a spurious flow error:

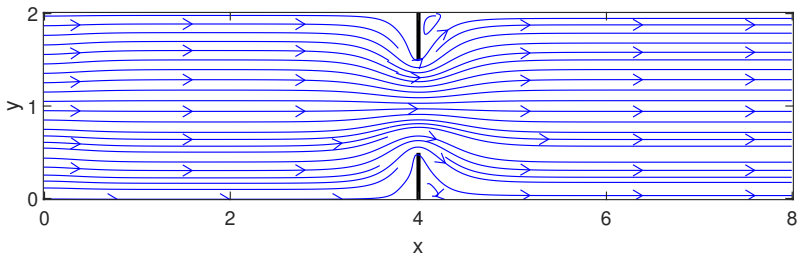
$$e^{\mathbf{v}} = \frac{1}{u_{max}} \sqrt{\sum_{i \in I^p} u_i^2 + \sum_{i \in I^p} v_i^2}, \tag{2.128}$$

where $I^p = \{i \mid (x_i, y_i) \in \Omega_p\}$ denotes the indices i of all DOFs u_i, v_i in Ω_p and in this example the porous domain Ω_p consists of the two obstructing walls. Error $e^{\mathbf{v}}$ measures the norm of flow in the porous domain and at the porous/fluid interface, and normalizes it using a measure for flow in the fluid domain (u_{max}). We expect flow at the porous/fluid interface to be dominant in the norm, and $e^{\mathbf{v}}$ can thus be seen as a measure for flow reduction v^{Γ}/v^{f}. Besides error $e^{\mathbf{v}}$ an error which represents the flow leakage in the solid domain is defined as:

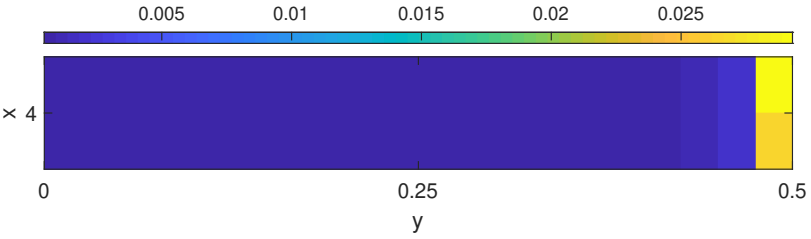
$$u_l = \frac{1}{L u_{max}} \int_{\Gamma_w} |\, u \,|\, d\Gamma, \tag{2.129}$$

where Γ_w is the center line in the two walls as shown in Figure 2.12. As u_l represents the leakage in the porous wall where $\alpha = \underline{\alpha}$ and is normalized by u_{max}, we use it as a measure of flow reduction in the solid domain v^s/v^f. Both $e^{\mathbf{v}}$ and u_l can thus be compared against the error estimates in Section 2.4.4.

LOW ELEMENTAL REYNOLDS RESULTS



(a) The flow solution.



(b) The magnitude of the erroneous flow in the bottom wall rotated 90 degrees to the right. Note the large contribution to the flow error at the wall tip to the right which causes the large errors $e^{\mathbf{v}}$.

Figure 2.13: A flow solution for the problem in Figure 2.12, computed using the VANS equations and the parameters in Table 2.3 and an erroneous flow plot. For this particular solution $q = 1$, $h = 0.025$ and $\underline{\alpha} = 10^{-1}$ were used.

Using these parameters the errors in Figure 2.14 and the flow field in Figure 2.13 are found. As shown in Figures 2.14a and 2.14b, we generally find that $\mathcal{O}\left(e^{\mathbf{v}}\right) = \mathcal{O}\left(10^{-q}\right)$

when $q \geq 0$ which confirms the bounds on $\tilde{\kappa}$ in Section 2.4 and expected flow reduction v^{Γ}/v^{f} in Equation 2.113. Furthermore, for the NSDP equations in Figure 2.14d $\mathcal{O}(u_l) = \mathcal{O}\left(\underline{\alpha}10^{-q}\right)$ and for the VANS equations in Figure 2.14c $\mathcal{O}(u_l) = \mathcal{O}\left(\underline{\alpha}^2 10^{-q}\right)$, which confirms the estimated flow reduction for u^s/u^f in Equation 2.114. When $q < 0$ the constraints derived in Section 2.4 are not satisfied, and relatively large errors $e^{\mathbf{v}}$ are found as predicted in Equation 2.116. As expected, larger errors $e^{\mathbf{v}}$ than u_l are found. The larger $e^{\mathbf{v}}$ is mainly caused by spurious flow through the tips of the walls as shown in Figure 2.13. Since the spurious flow mainly crosses the corners of the walls but not Γ_w, it only shows up in the computation of $e^{\mathbf{v}}$ and not in the computation of u_l. Furthermore, as the wall is two elements thick, the different h lead to slightly different problems with slightly different solutions, but no qualitatively different behavior between the solutions is observed for $h < 0.1$. For $h = 0.1$ the mesh is too coarse and is not able to facilitate the vortices at the base of the walls as observed in Figure 2.13. We note that for $q \leq 0$ the solution procedure becomes less stable and shows longer convergence times.

LOW ELEMENTAL REYNOLDS OPTIMIZATION PARAMETER SELECTION

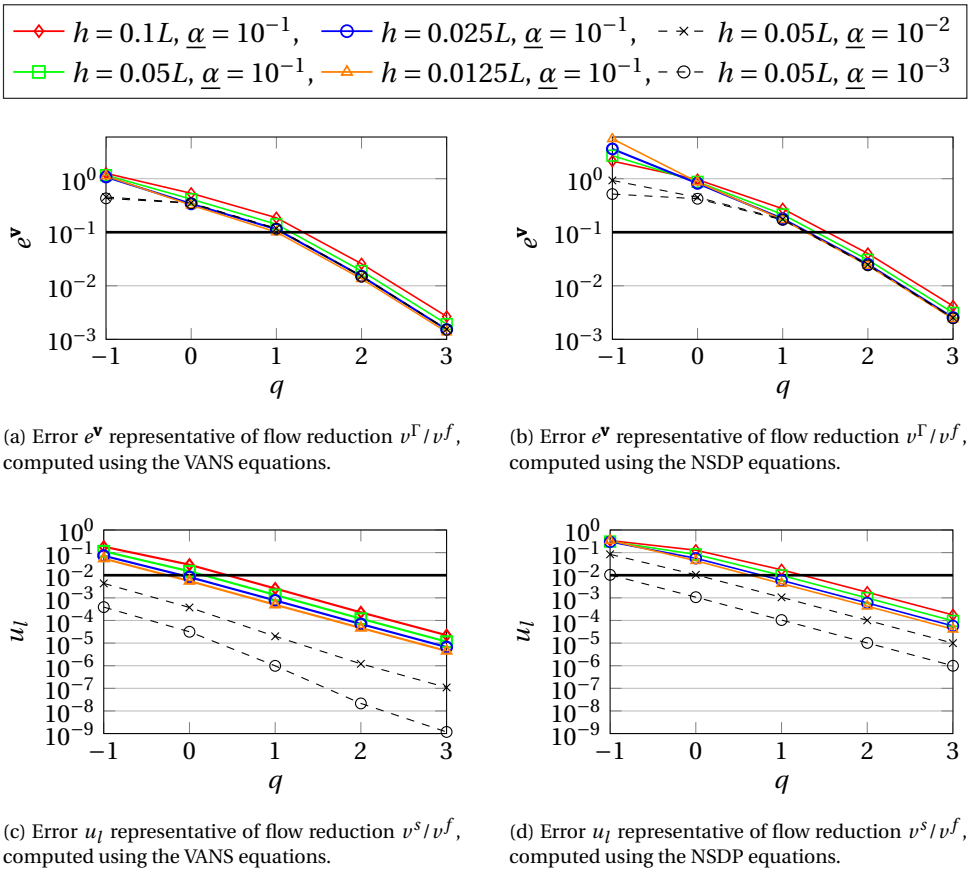
Subsequently, the errors in Figure 2.14 are used to select appropriate optimization values for q and $\underline{\alpha}$ such that $e^{\mathbf{v}} \approx 10^{-1}$ and $u_l < 10^{-2}$. For the VANS equations we use $q = 1$ as for this setting $e^{\mathbf{v}} \approx 10^{-1}$ in Figure 2.14a. We note that the errors are actually slightly higher but choose $q = 1$ as $q = 2$ would result in $e^{\mathbf{v}} \approx 10^{-2}$ and possibly a too severe penalization on the intermediate design where $\alpha \approx 0.5$ and convergence to inferior local optima. Moreover, for the selection of $\underline{\alpha}$ we note that when $\underline{\alpha} < 10^{-2}$ design convergence of the optimization problems in Section 2.7 was often ill behaved for both the VANS and NSDP equations. In Figure 2.14c for $q = 1$ we find $u_l \approx 10^{-3} < 10^{-2}$ when $\underline{\alpha} = 10^{-1}$ for the VANS equations. To select q for the NSDP equation both $q = 1$ and $q = 2$ are viable options for $e^{\mathbf{v}} \approx 10^{-1}$ in Figure 2.14b. If we select $q = 1$, we require $\underline{\alpha} = 10^{-2}$ to achieve $u_l \approx 10^{-3} < 10^{-2}$ for the NSDP equations in Figure 2.14d. However, if we select $q = 2$, $\underline{\alpha} = 10^{-1}$ is sufficient to achieve $u_l \approx 10^{-3} < 10^{-2}$ for the NSDP equations in Figure 2.14d. For the NSDP equations there are thus two options for appropriate parameter settings although we suspect the parameters using $q = 2$ might put a too severe penalization on intermediate designs. All parameter setting are summarized in Table 2.4 where we use a superscript to denote the parameter setting of a certain model.

	VANS$^{(a)}$	NSDP$^{(a)}$	NSDP$^{(b)}$
q	1	1	2
$\underline{\alpha}$	10^{-1}	10^{-2}	10^{-1}

Table 2.4: Parameter settings to achieve sufficient flow penalization such that $e^{\mathbf{v}} \approx 10^{-1}$ and $u_l < 10^{-2}$ during low Reynolds optimization for the VANS and NSDP equations. Two options (NSDP$^{(a)}$ and NSDP$^{(b)}$) are viable for the NSDP equations.

The selected penalization parameters can be compared against common practice. Using the definition of the maximum Darcy penalization in Equation 2.111 we find:

$$\overline{K} = \frac{\mu}{L^2 Da} = 10^5, \tag{2.130}$$



(a) Error $e^{\mathbf{v}}$ representative of flow reduction v^{Γ}/v^{f}, computed using the VANS equations.

(b) Error $e^{\mathbf{v}}$ representative of flow reduction v^{Γ}/v^{f}, computed using the NSDP equations.

(c) Error u_l representative of flow reduction v^{s}/v^{f}, computed using the VANS equations.

(d) Error u_l representative of flow reduction v^{s}/v^{f}, computed using the NSDP equations.

Figure 2.14: The errors $e^{\mathbf{v}}$ and u_l for **low** Re^e in the problem as illustrated in Figure 2.12 using the parameters from Table 2.3 computed using varying h and $\underline{\alpha}$. The horizontal solid black lines indicate the maximum allowable errors $e^{\mathbf{v}} \approx 10^{-1}$ and $u_l < 10^{-2}$ for optimization, and are used to select appropriate optimization parameters q and $\underline{\alpha}$.

where we used $Da = 10^{-5}$ as recommended by Olesen et al. (2006), a parameter which often requires a lot of tuning. On the contrary, since we use $\bar{\kappa}$ as defined in Equation 2.126 a maximum penalization is found using Equation 2.109 as:

$$\overline{K}_h = \mu \frac{10^q}{h^2 \underline{\alpha}}. \tag{2.131}$$

Consequently, using the optimization parameters in Table 2.4 we find the maximum penalizations:

$$\begin{aligned} \overline{K}_h^{VANS} &= \mu \frac{10^q}{h^2 \underline{\alpha}} = 10^2 \frac{\mu}{h^2}, \\ \overline{K}_h^{NSDP} &= \mu \frac{10^q}{h^2 \underline{\alpha}} = 10^3 \frac{\mu}{h^2}, \end{aligned} \tag{2.132}$$

where the parameters for NSDP$^{(a)}$ and NSDP$^{(b)}$ result in the same maximum penalization $\overline{K}_h^{NSDP}$ and the resulting penalization values for the four different mesh sizes can be found in Table 2.5. Even though the maximum penalization values for each model in Table 2.5 span two (almost three) orders of magnitude for varying h, similar error magnitudes are found for the same settings of q and $\underline{\alpha}$ in Figure 2.14. Moreover, whereas using the common penalization definition in Equation 2.130 ($\overline{K} = 10^5$) some tuning would be required to find the appropriate setting for the NSDP equations, using our new definition we are able to precisely define $\overline{K}_h = 6.4 \cdot 10^6$ to achieve the desired error magnitude for $h = 0.0125$.

h [m]	0.1	0.05	0.025	0.0125
$\overline{K}_h^{VANS}$ $[\frac{\mathrm{N\,s}}{\mathrm{m}^4}]$	10^4	$4 \cdot 10^4$	$1.6 \cdot 10^5$	$6.4 \cdot 10^5$
$\overline{K}_h^{NSDP}$ $[\frac{\mathrm{N\,s}}{\mathrm{m}^4}]$	10^5	$4 \cdot 10^5$	$1.6 \cdot 10^6$	$6.4 \cdot 10^6$

Table 2.5: The Darcy penalization from Equation 2.132 computed using the material parameters in Table 2.3 and optimization parameters in Table 2.4. Both optimization parameter setting for NSDP$^{(a)}$ and NSDP$^{(b)}$ in Table 2.4 result in the same maximum penalization $\overline{K}_h^{NSDP}$.

2.6.2. MODERATE ELEMENTAL REYNOLDS NUMBERS

Secondly, a sweep on the penalization is performed for moderate Re^e (and moderate Re) using the new parameters given in Table 2.6. The walls are shifted to the left ($L_{wall} = L$) to account for large wakes. Using these parameters, the Reynolds number is computed

μ	ρ	u_{max}	p_{out}	L	L_{wall}	L_x
$5 \cdot 10^{-3}$ [Pas]	1 [kgm^{-3}]	1 [ms^{-1}]	1 [Pa]	1 [m]	L	$8L$

Table 2.6: The material and problem parameters for the flow problem in Figure 2.15, when moderate Reynolds flow is investigated for varying q and $\underline{\alpha}$.

as:

$$Re = \frac{\rho u_{max} L}{\mu} = 2 \cdot 10^2. \tag{2.133}$$

Furthermore, as the elemental Reynolds number is directly proportional to the mesh size $Re^e \propto h$ only the relatively large mesh sizes of $h = 0.1L$, $h = 0.05L$ and $h = 0.025L$ are investigated. Subsequently, the elemental Reynolds number is computed by approximating $\left|\mathbf{v}^f\right| \approx u_{max}$ as:

$$Re^e = \frac{\rho u_{max} h}{\mu} = 2h \cdot 10^2, \tag{2.134}$$

resulting in $Re^e = 20$, $Re^e = 10$ and $Re^e = 5$ for the decreasing element sizes respectively. As $Re^e > 1$ the definition of $\bar{\kappa}$ changes following Table 2.2 as:

$$\bar{\kappa} = 10^q H^e Re^e = 10^q \frac{\rho u_{max}}{h\mu}. \tag{2.135}$$

MODERATE ELEMENTAL REYNOLDS RESULTS

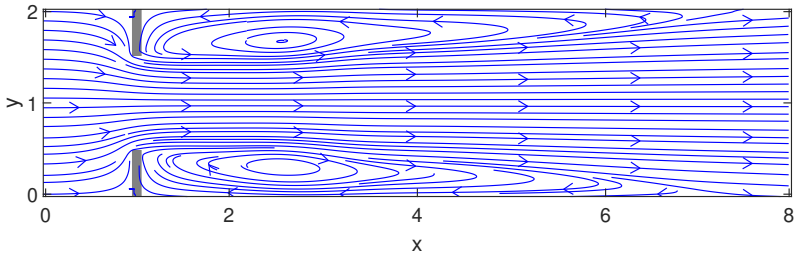


Figure 2.15: The resulting flow field for the problem in Figure 2.12, computed using the VANS equations and the parameters in Table 2.6. For the computation of this particular flow field $q = 3$ and $\underline{\alpha} = 10^{-2}$ were used, and for all other settings (VANS and NSDP) similar flow fields were found.

Using these parameters, the errors in Figure 2.16 and flow profile in Figure 2.15 are found. For $q = -1$, flow in the porous walls showed a tendency to oscillate and not stabilize and for $q < -1$ flow in the porous walls does not stabilize. This problem is more prominently present in the VANS equations than in the NSDP equations. In Figures 2.16a and 2.16b we generally find $\mathcal{O}\left(e^{\mathbf{v}}\right) = \mathcal{O}\left(10^{-q}\right)$ for both the VANS and NSDP equations, confirming the bounds on $\bar{\kappa}$ derived in Section 2.4 and expected flow reduction v^{Γ}/v^{f} in Equation 2.113. Furthermore, in Figure 2.16d we find $\mathcal{O}\left(u_l\right) = \mathcal{O}\left(\underline{\alpha}10^{-q}\right)$ for the NSDP equations as expected in Equation 2.114. Contrarily, for the VANS equations u_l in Figure 2.16c behaves less regular. For $\underline{\alpha} = 10^{-1}$, the error behaves regular as $\mathcal{O}\left(u_l\right) = \mathcal{O}\left(\underline{\alpha}^2 10^{-q}\right)$ as predicted by Equation 2.114. However, for $\underline{\alpha} < 10^{-1}$ errors behave less regular and are bounded from above as $\mathcal{O}\left(u_l\right) \leq \mathcal{O}\left(\underline{\alpha}10^{-q}\right)$. Nonetheless, the error follows the prediction of $\mathcal{O}\left(u_l\right) = \mathcal{O}\left(\underline{\alpha}^2 10^{-q}\right)$, but only for $q > 1$ when $\underline{\alpha} = 10^{-2}$ and for $q > 2$ when $\underline{\alpha} = 10^{-3}$. However, comparing errors between the VANS and NSDP equations in Figure 2.16, we find that the VANS equations generally result in errors of lower or similar magnitude.

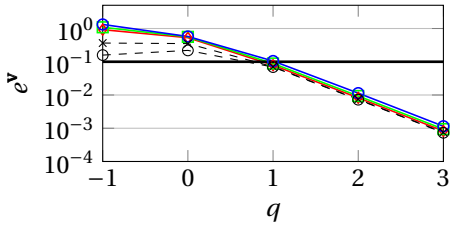
MODERATE ELEMENTAL REYNOLDS OPTIMIZATION PARAMETER SELECTION

Subsequently, the errors in Figure 2.16 are used to find appropriate setting for q and $\underline{\alpha}$ such that $e^{\mathbf{v}} \approx 10^{-1}$ and $u_l < 10^{-2}$. For the VANS equations we use $q = 1$ as for this setting $e^{\mathbf{v}} \approx 10^{-1}$ in Figure 2.16a. Moreover, in Figure 2.16c we find $u_l \approx 10^{-3} < 10^{-2}$ for $q = 1$ and $\underline{\alpha} = 10^{-1}$ which we will thus use for optimization using the VANS equations. For the NSDP equations we also use $q = 1$ as for this setting $e^{\mathbf{v}} \approx 10^{-1}$ in Figure 2.16b. Contrary to the low Reynolds NSDP optimization, the moderate Reynolds NSDP optimization has only one appropriate setting for q. Furthermore, for $q = 1$ and $\underline{\alpha} = 10^{-2}$ we find $u_l \approx 10^{-3} < 10^{-2}$ in Figure 2.16d which we will thus adopt for optimization using the NSDP equations. Parameter settings for moderate Reynolds optimization can be found in Table 2.7.

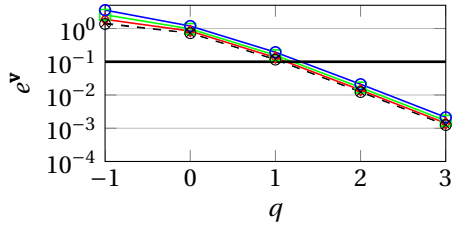
Furthermore, using $\bar{\kappa}$ as in Equation 2.135 results in the maximum Darcy penalization in Equation 2.110 of:

$$\overline{K}_h = \frac{10^q \rho \left|\mathbf{v}^f\right|}{h\underline{\alpha}} = \frac{10^q \rho u_{max}}{h\underline{\alpha}}. \tag{2.136}$$

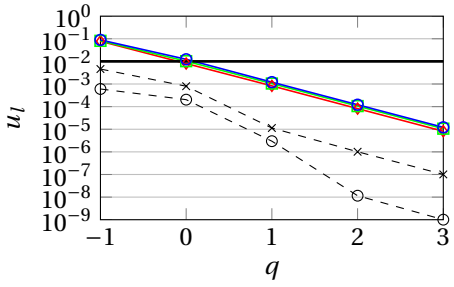
$h = 0.1L$, $\underline{\alpha} = 10^{-1}$, $Re^e = 20$, $h = 0.05L$, $\underline{\alpha} = 10^{-2}$, $Re^e = 10$
$h = 0.05L$, $\underline{\alpha} = 10^{-1}$, $Re^e = 10$, $h = 0.05L$, $\underline{\alpha} = 10^{-3}$, $Re^e = 10$
$h = 0.025L$, $\underline{\alpha} = 10^{-1}$, $Re^e = 5$,



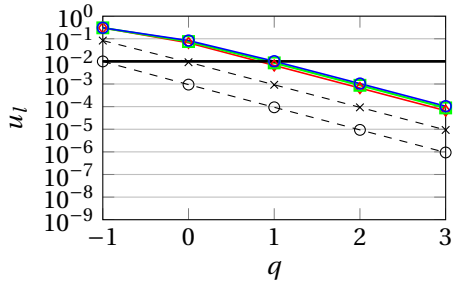
(a) Error $e^{\mathbf{v}}$ representative of flow reduction v^{Γ}/v^f, computed using the VANS equations.



(b) Error $e^{\mathbf{v}}$ representative of flow reduction v^{Γ}/v^f, computed using the NSDP equations.



(c) Error u_l representative of flow reduction v^s/v^f, computed using the VANS equations.



(d) Error u_l representative of flow reduction v^s/v^f, computed using the NSDP equations.

Figure 2.16: The errors $e^{\mathbf{v}}$ and u_l for the **moderate** Reynolds problem as illustrated in Figure 2.12 using the parameters from Table 2.3 computed using varying h and $\underline{\alpha}$. The horizontal solid black lines indicate the maximum allowable errors $e^{\mathbf{v}} \approx 10^{-1}$ and $u_l < 10^{-2}$ for optimization, and are used to select appropriate optimization parameters q and $\underline{\alpha}$.

Consequently, using the optimization parameters in Table 2.7 we find the maximum penalizations of:

$$\begin{aligned} \overline{K}_h^{VANS} &= \frac{10^q \rho u_{max}}{h\underline{\alpha}} = 10^2 \frac{\rho u_{max}}{h}, \\ \overline{K}_h^{NSDP} &= \frac{10^q \rho u_{max}}{h\underline{\alpha}} = 10^3 \frac{\rho u_{max}}{h}, \end{aligned} \tag{2.137}$$

resulting in the penalization values for varying h in Table 2.8. Using the common penalization definition in Equation 2.111 (with $Da = 10^{-5}$):

$$\overline{K} = \frac{\mu}{L^2 Da} = 5 \cdot 10^2, \tag{2.138}$$

would thus result in under penalization for the NSDP equations.

	VANS$^{(a)}$	NSDP$^{(a)}$
q	1	1
$\underline{\alpha}$	10^{-1}	10^{-2}

Table 2.7: Parameter settings to achieve sufficient flow penalization such that $e^{\mathbf{v}} \approx 10^{-1}$ and $u_l < 10^{-2}$ during moderate Reynolds optimization for the VANS and NSDP equations.

h [m]	0.1	0.05	0.025
$\overline{K}_h^{VANS}$ $[\frac{\mathrm{N\,s}}{\mathrm{m}^4}]$	10^3	$2\cdot10^3$	$4\cdot10^3$
$\overline{K}_h^{NSDP}$ $[\frac{\mathrm{N\,s}}{\mathrm{m}^4}]$	10^4	$2\cdot10^4$	$4\cdot10^4$

Table 2.8: The Darcy penalization from Equation 2.137 computed using the parameters in Table 2.6 and optimization parameters in table 2.7.

2.6.3. SWEEP ON THE ELEMENTAL REYNOLDS NUMBER

Comparing overall performance of the VANS and NSDP error convergence both sets of equations are found to reduce errors satisfactory for increasing q and decreasing $\underline{\alpha}$ and can thus be used to find precise optima in topology optimization. However, for higher Reynolds numbers an estimation of Re^e has to be made which depends on an *a priori* estimated flow velocity $\left|\mathbf{v}^f\right|$. Subsequently, when $Re^e > 1$ this estimation is used to set the appropriate $\bar{\kappa}$ as found in Table 2.2. In a changing topology Reynolds numbers may change and this estimate may be incorrect. The Reynolds dependence of the errors in the VANS and NSDP equations is thus investigated using the parameters in Table 2.9 where in contrast to Sections 2.6.1 and 2.6.2 we fix $q = 1$ and $h = 0.1 \cdot L$ but investigate for varying Reynolds number:

$$Re = \frac{\rho u_{max} L}{\mu}, \tag{2.139}$$

by changing the density as:

$$\rho = \frac{Re\mu}{u_{max} L} = Re, \tag{2.140}$$

Moreover, using the density and fixed mesh size $h = 0.1 \cdot L$ the elemental Reynolds number can be approximated as:

$$Re^e \approx \frac{\rho u_{max} h}{\mu} = \frac{Re\mu}{u_{max} L} \frac{u_{max} 0.1 \cdot L}{\mu} = 0.1 \cdot Re. \tag{2.141}$$

Using the bounds on $\bar{\kappa}$ as found in Table 2.2, maximum penalization values are found in Equations 2.109 and 2.110 as:

$$\begin{aligned} \overline{K}_h &= \mu \frac{10^q H^e}{\underline{\alpha}} = \frac{10^3}{\underline{\alpha}} \text{ for } Re^e \leq 1, \\ \overline{K}_h &= \frac{10^q H^e Re^e}{\underline{\alpha}} = \frac{10^3 Re^e}{\underline{\alpha}} \text{ for } Re^e > 1. \end{aligned} \tag{2.142}$$

μ	u_{max}	p_{out}	L	L_{wall}	L_x	q	h
1 [Pas]	1 [ms^{-1}]	1 [Pa]	1 [m]	L	$20L$	1	$0.1L$

Table 2.9: The material and problem parameters for the flow problem in Figure 2.15, for varying Reynolds numbers (and thus varying ρ).

Furthermore, we also investigate the case where the elemental Reynolds number is underestimated and we use the penalization for $Re^e \leq 1$ in Equation 2.142 ($\bar{\kappa} = 10^q H^e$) to compute errors for the cases where $Re^e > 1$. Using the resulting penalization values

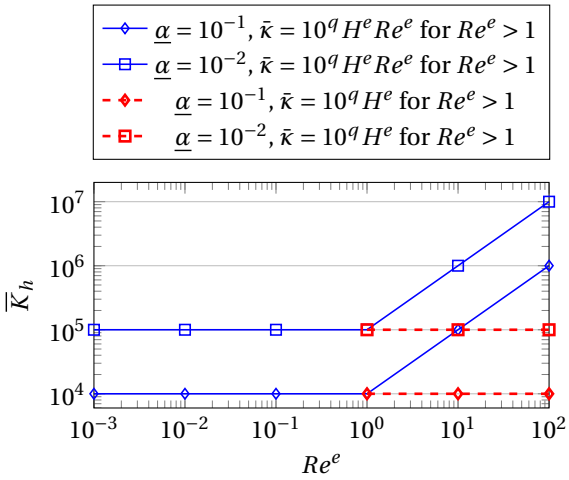
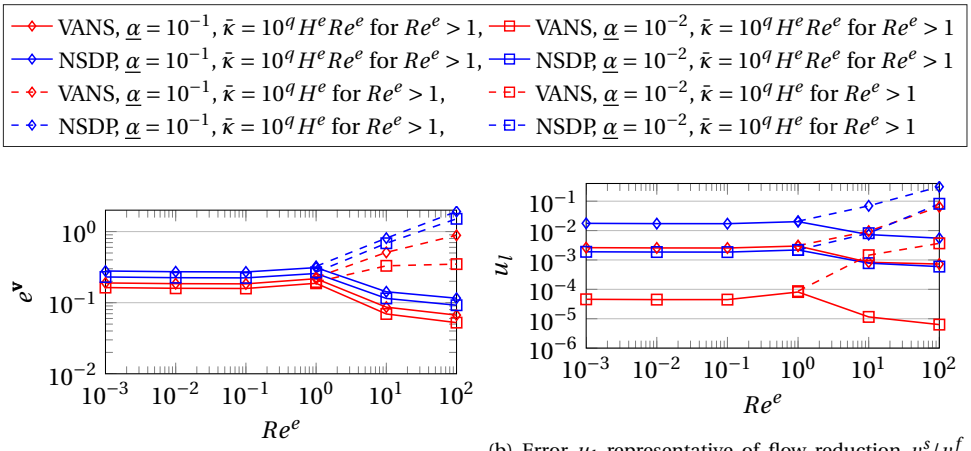


Figure 2.17: Maximum penalization values $\overline{K}_h$ computed using the Equation 2.142 and used in the computations for Figure 2.18. For $Re^e \leq 1$ we use $\bar{\kappa} = 10^q H^e$, while for $Re^e > 1$ we use two different definitions $\bar{\kappa}$.

found in Figure 2.17 leads to the errors shown in Figure 2.18. No Reynolds numbers larger than $Re = 10^3$ ($Re^e = 0.1 \cdot Re = 10^2$) were investigated as this would be within the turbulent flow regime which the flow solver is not suited for. When we use $\bar{\kappa} = 10^q H^e Re^e$ for $Re^e > 1$ both $e^{\mathbf{v}}$ and u_l decrease in Figure 2.18 but remain close to the expected order of magnitude as predicted in Section 2.4.4. However, when we divert from the penalization bounds derived in Section 2.4 and use a fixed $\bar{\kappa} = 10^q H^e$ for $Re^e > 1$, the order of magnitude of u_l in Figure 2.18 increases significantly for increasing Re^e. For $Re^e > 1$ a correct penalization is thus coupled to the elemental Reynolds number.

2.7. TOPOLOGY OPTIMIZATION EXAMPLES USING THE VANS AND NSDP EQUATIONS

To verify the applicability of the VANS equations to topology optimization firstly a flow problem is optimized for low and moderate Reynolds flow, after which we investigate optimization under moderate Reynolds flow in more detail. The problems are inspired by the flow around a bend problem as defined by Kreissl and Maute (2012), and the two



(a) Error $e^{\mathbf{v}}$ representative of flow reduction v^Γ / v^f computed using $q = 1$ for a range of Reynolds numbe

(b) Error u_l representative of flow reduction v^s / v^f computed using $q = 1$ for a range of Reynolds numbers.

Figure 2.18: The errors $e^{\mathbf{v}}$ and u_l for a range of Reynolds numbers computed using the parameters from Table 2.9 resulting in the penalization values in Figure 2.17.

channel flow problem as defined by Olesen et al. (2006).

2.7.1. INITIAL OPTIMIZATION INVESTIGATION

To push the flow penalization within an optimization to its limits a flow around a two element thick porous solid wall as shown in Figure 2.19 is optimized. The inlet and outlet are separated from the design domain by short pipes to allow for an accurate description of the boundary conditions. On inlet Γ_{in} a parabolic velocity profile with maximum velocity u_{max} is prescribed and on outlet Γ_{out} static reference pressure p_{out} is prescribed. The objective of the optimization procedure is to minimize pressure drop:

$$f = \int_{\Gamma_{in}} p d\Gamma - \int_{\Gamma_{out}} p d\Gamma, \tag{2.143}$$

under a volume constraint of $V_f = 0.5$. In fact, minimizing pressure drop is equivalent to minimizing fluid energy dissipation. The optimization problem is initialized using a completely fluid design domain ($\alpha = 1$) and the optimizer thus determines where to introduce solid material. Given the pressure drop objective, adding material will generally increase pressure drop and the first few design iterations objective values will increase. Furthermore, viscosity μ is determined using the Reynolds number as:

$$\mu = \frac{\rho u_{max} L}{Re}, \tag{2.144}$$

and the structure is optimized for $Re = 0.2$ and $Re = 200$ resulting in $Re^e = 0.01$ and $Re^e = 10$ respectively using the parameters in Table 2.10. Subsequently, maximum penalization $\bar{\kappa}$ is determined using q, H^e and Re^e as in Table 2.2, and we select q and $\underline{\alpha}$ following Tables 2.4 and 2.7 for low ($Re^e = 0.01$) and moderate ($Re^e = 10$) Reynolds optimization respectively. In Table 2.10 we show the penalization in the intermediate design

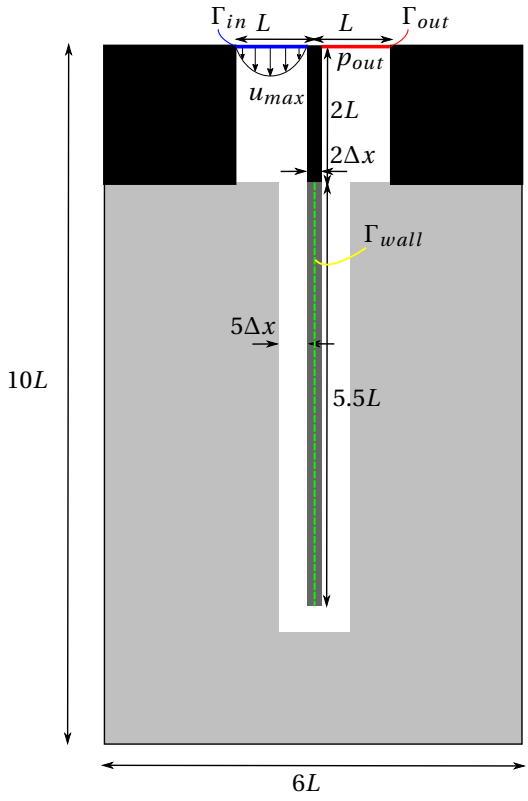


Figure 2.19: An optimization problem which minimizes pressure drop for flow around a thin wall of two elements thick. The thin wall (dark gray) is modeled using a porous formulation and its density is fixed at $\underline{\alpha}$, to challenge the flow model the thin wall is surrounded by a fluid non design domain (white) of five elements thick. A parabolic flow is prescribed on inlet Γ_{in} and a constant pressure on outlet Γ_{out}. The inlet/outlet are separated from the design domain by fluid pipes (white) surrounded by solid material where no flow is present (black). The optimization procedure is initialized using a completely fluid design domain (light gray).

where $\alpha \approx 0.5$ and thus $-\mu\bar{\kappa}(1-\alpha)/\alpha \approx -\mu\bar{\kappa}$ and the penalization in the solid domain $\overline{K}_h$. It can be observed that the selected parameters result in a maximal penalization $\overline{K}_h$ which spans two orders of magnitude for both the VANS and NSDP equations respectively. Kreissl and Maute (2012) find appropriate maximum penalization values for the problem on which our problem is inspired of $\overline{K} = 2.5 \cdot 10^6$ for $Re = 0.1$ and $\overline{K} = 2.5 \cdot 10^4$ for $Re = 10$. We thus use similar order of magnitude penalizations for the low and moderate Reynolds NSDP equations. However, our moderate Reynolds penalization could be expected to be one order higher than the one by Kreissl and Maute (2012) as our moderate Reynolds number is one order higher ($Re = 200$ instead of $Re = 10$) and the elemental Reynolds number should therefore also be one order higher. Since our penalization scales with elemental Reynolds number, the penalization by Kreissl and Maute (2012) could be extrapolated to $Re = 200$ to find a penalization of magnitude 10^5 which is one

ρ	u_{max}	p_{out}	L	h	H^e	V_f
1 [kgm^{-3}]	1 [ms^{-1}]	1 [Pa]	1 [m]	0.05	400 [m^{-2}]	0.5 [m]

	Re, Re^e	q	$\underline{\alpha}$	$\mu\bar{\kappa}$	$\overline{K}_h$
NSDP$^{(a)}$	0.2, 0.01	1	10^{-2}	$2\cdot10^4$	$2\cdot10^6$
NSDP$^{(b)}$	0.2, 0.01	2	10^{-1}	$2\cdot10^5$	$2\cdot10^6$
NSDP$^{(a)}$	200, 10	1	10^{-2}	$2\cdot10^2$	$2\cdot10^4$
VANS$^{(a)}$	0.2, 0.01	1	10^{-1}	$2\cdot10^4$	$2\cdot10^5$
VANS$^{(a)}$	200, 10	1	10^{-1}	$2\cdot10^2$	$2\cdot10^3$

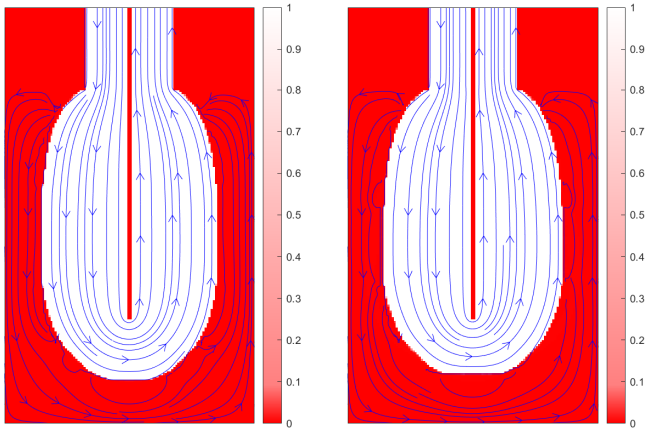
Table 2.10: The material and problem parameters for the optimization problem in Figure 2.19. The structure is optimized for low and moderate Reynolds flow, resulting in two different Re^e. For $Re^e = 0.01$ and $Re^e = 10$ values for q and $\underline{\alpha}$ are selected following Tables 2.4 and 2.7 respectively and $\bar{\kappa}$ is computed following Table 2.2, resulting the maximal penalizations $\overline{K}_h$ computed using Equations 2.131 and 2.136. We show $\mu\bar{\kappa}$ as this is the term used to penalize intermediate designs where $\alpha \approx 0.5$.

order higher than the one in Table 2.10 for $Re = 200$ NSDP optimization.

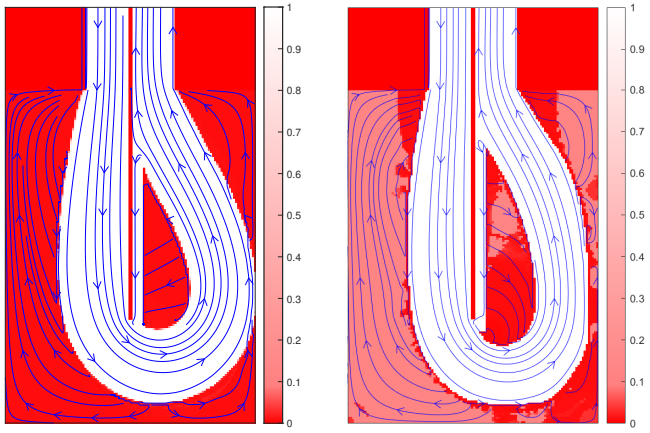
Optimized designs can be found in Figures 2.20 and 2.21. Furthermore, error $e^{\mathbf{v}}$ is computed by constructing porous domain Ω_p using all elements where $\alpha < 0.5$, and u_l as defined in Equation 2.129 is computed using Γ_{wall} as defined in Figure 2.19. Both errors and the optimized objectives f^* can be found in Table 2.11. The VANS equations

	VANS$^{(a)}$	NSDP$^{(a)}$	NSDP$^{(b)}$	VANS$^{(a)}$	NSDP$^{(a)}$
	$Re = 0.2$ $q = 1$ $\underline{\alpha} = 10^{-1}$	$Re = 0.2$ $q = 1$ $\underline{\alpha} = 10^{-2}$	$Re = 0.2$ $q = 2$ $\underline{\alpha} = 10^{-1}$	$Re = 200$ $q = 1$ $\underline{\alpha} = 10^{-1}$	$Re = 200$ $q = 1$ $\underline{\alpha} = 10^{-2}$
$e^{\mathbf{v}}$	$7.70\cdot10^{-2}$	$8.70\cdot10^{-2}$	$2.28\cdot10^{-2}$	$2.56\cdot10^{-2}$	$3.07\cdot10^{-2}$
u_l	$3.60\cdot10^{-3}$	$3.28\cdot10^{-3}$	$3.59\cdot10^{-3}$	$9.24\cdot10^{-4}$	$8.62\cdot10^{-4}$
f^*	278.2	274.0	293.4	0.4177	0.4529
	VANS^{R}	VANS^{R}	VANS^{R}	VANS^{R}	VANS^{R}
$e^{\mathbf{v}}$	$8.29\cdot10^{-8}$	$8.52\cdot10^{-8}$	$1.10\cdot10^{-8}$	$1.98\cdot10^{-8}$	$2.46\cdot10^{-8}$
u_l	$3.40\cdot10^{-13}$	$3.56\cdot10^{-13}$	$4.08\cdot10^{-14}$	$8.43\cdot10^{-14}$	$9.00\cdot10^{-14}$
f^*	280.6	282.8	302.4	0.4209	0.4676

Table 2.11: The optimized objectives and errors for the problem defined in Figure 2.19 and Table 2.10 with optimized designs in Figures 2.20 and 2.21. VANS^{R} solutions are computed using $\underline{\alpha} = 10^{-3}$ and $q = 7$.



(a) VANS$^{(a)}$, $Re = 0.2$, $q = 1$, $\underline{\alpha} = 10^{-1}$ (b) NSDP$^{(a)}$, $Re = 0.2$, $q = 1$, $\underline{\alpha} = 10^{-2}$



(c) VANS$^{(a)}$, $Re = 200$, $q = 1$, $\underline{\alpha} = 10^{-1}$ (d) NSDP$^{(a)}$, $Re = 200$, $q = 1$, $\underline{\alpha} = 10^{-2}$

Figure 2.20: The optimal design and flow fields for the problem in Figure 2.19 using the parameters in Table 2.10. Although flow through the solid (gray) material is plotted, spurious solid flow remains low as shown by the errors in Table 2.11.

are used to compute reference flow and pressure solutions (VANS^{R} solutions), objectives and errors. To achieve accurate reference results we set $\underline{\alpha} = 10^{-3}$ and $q = 7$, and post process the designs into crisp solid fluid distributions where α is set to zero for $\alpha < 0.5$ and to one for $\alpha \geq 0.5$. The convergence of the objectives is shown in Figure 2.22 where

the objective is normalized using the objective value at the first design iteration f_1. In principle, all optimization procedures ran for 300 iterations. However, if the design did not stabilize after 300 iterations the optimizer is allowed to optimize for an additional 200 or 400 iterations depending on the convergence behavior resulting in a total of 500 or 700 optimization iterations respectively. In general errors for the VANS and NSDP

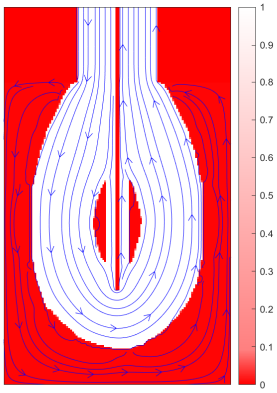
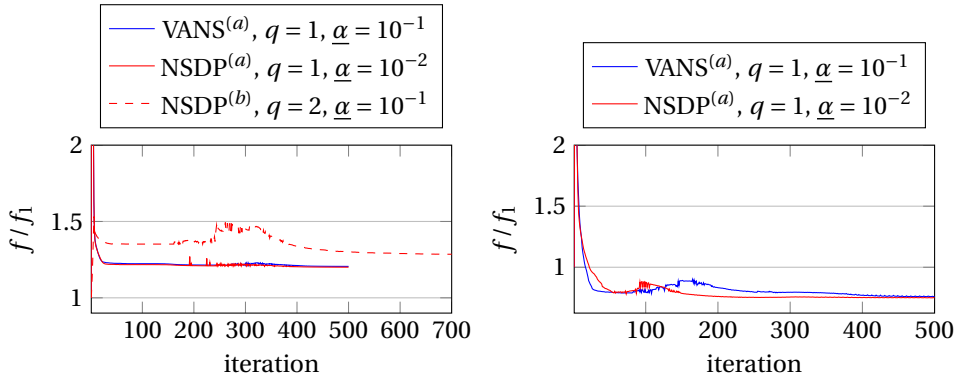


Figure 2.21: The inferior local optimum for the problem in Figure 2.19, computed using the NSDP$^{(b)}$ parameter set from Table 2.10 for $Re = 0.2$. spurious flow and objective values can be found in Table 2.11.

optimized designs in Table 2.11 are comparable. However, reference objectives of the VANS based optimal designs are generally lower than those of the NSDP based optimal designs.

ANALYSIS OF THE RESULTS

A noticeably different topology is found for $Re = 0.2$ when using the NSDP$^{(b)}$ model in Figure 2.21 than using the VANS$^{(a)}$ or NSDP$^{(a)}$ models in Figures 2.20a and 2.20b. Moreover, the NSDP$^{(b)}$ based optimization shows a longer and less regular convergence in Figure 2.22a and finds the worst performing optimum as shown in Table 2.11. This convergence behavior is tied to the two solid islands beside the inner wall in Figure 2.21 and the fact that $q = 2$ for NSDP$^{(b)}$ instead of $q = 1$ for NSDP$^{(a)}$, causing a greater $\mu\bar{\kappa}$ for the NSDP$^{(b)}$ model as can be found in Table 2.10. In the fist few design iterations, the optimizer adds gray material ($\alpha \approx 0.5$) to the fluid design domain to improve the design. However, using $q = 2$ this gray material is over-penalized as flow is reduced as $\mathcal{O}\left(10^{-q}\right)$ as derived in Equation 2.113. Over-penalizing flow causes much energy dissipation in the gray domain which is inefficient for minimizing pressure drop, as a consequence these gray areas are quickly converted to solid domains ($\alpha \to 0$) such that flow through them and thus fluid energy dissipation is minimized. Moreover, in Figures 2.23a and 2.23b intermediate designs for the VANS$^{(a)}$ and NSDP$^{(a)}$, $Re = 0.2$ optimization can be found which also show some intermediate material islands beside the wall. However, as the VANS$^{(a)}$ and NSDP$^{(a)}$ based optimizations use $q = 1$ they do not over-penalize these intermediate designs and are able to escape this inferior local optimum. Furthermore,

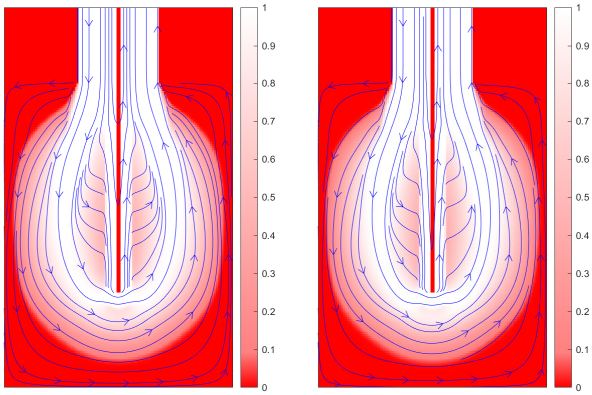


(a) $Re = 0.2$ (b) $Re = 200$

Figure 2.22: The convergence of the designs in Figures 2.20 and 2.21 with errors and objectives in Table 2.11. Objective values are normalized using the objective in the first design iteration f_1. During the first few design iterations objective values drastically increase by the addition of solid material to the fluid design domain, however normalized objective values are cut of at 2 to be able to inspect the convergence behavior

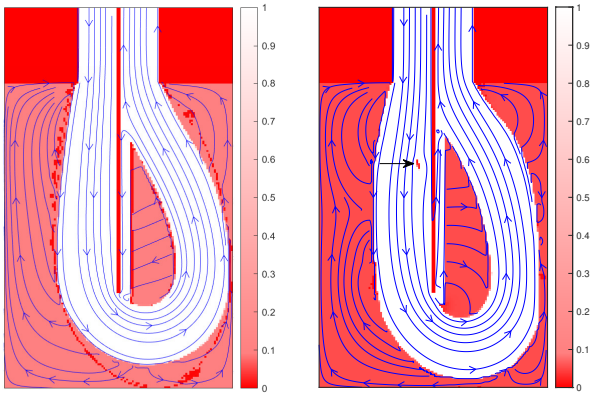
the convergence plot for the NSDP$^{(b)}$ parameter set in Figure 2.22a shows a large bump starting around iteration 250. This bump is caused by the optimization process decreasing the islands by pushing the boundaries inward during iterations 250-400. When the boundaries are pushed inward elements are not instantly switched from solid to fluid but move through some grayscale values $\alpha \approx 0.5$. In these grayscale areas flow speeds and thus fluid energy dissipation are increased and sensitivities change due to the non-linear nature of the Navier-Stokes equations. Due to the increased fluid energy dissipation the pressure drop objective increases, and due to the changed sensitivity information the design is further disturbed. The small oscillations around iterations 200-400 in objective for the VANS$^{(a)}$ and NSDP$^{(a)}$ convergence in Figure 2.22a are caused by a similar effect. During these iterations the designs and objective are quite close to the optimal design and objective, but small tweaks to the boundaries continue to be made which causes some boundary elements to become gray increasing fluid energy dissipation and pressure drop. A final remark is made on the fact that for $Re = 0.2$ the optimized pressure drop f^* is close to the reference VANS^R pressure drop for the VANS equations but large differences are observed between the NSDP and VANS^R pressure drop in Table 2.11. As shown in Section 2.3.1 for flow parallel to a wall the interpretation of the solid/fluid interface is off by $h/2$, causing lowered pressure drop through a channel for the NSDP model as confirmed in Appendix 2.D. However, the VANS equations correct for this error via the second Brinkman correction.

Comparing the optimal designs for $Re = 200$ in Figures 2.20c and 2.20d and reference objectives in Table 2.11, the moderate Reynolds NSDP$^{(a)}$ based optimization also seems to converge to an inferior local optimum. Convergence to the inferior local optimum may be caused by an overestimation of the Reynolds and consequently elemental Reynolds numbers. Both Re and Re^e are computed using the maximum velocity at the inlet u_{max}. However, within the design domain flow channels generally widen and



(a) VANS$^{(a)}$, iteration 4, $Re = 0.2$, $q = 1$, $\underline{\alpha} = 10^{-1}$.

(b) NSDP$^{(a)}$, iteration 4, $Re = 0.2$, $q = 1$, $\underline{\alpha} = 10^{-2}$.



(c) VANS$^{(a)}$, iteration 150, $Re = 200$, $q = 1$, $\underline{\alpha} = 10^{-1}$.

(d) NSDP$^{(a)}$, iteration 80, $Re = 200$, $q = 1$, $\underline{\alpha} = 10^{-2}$, a small solid island can be found at the arrow tip.

Figure 2.23: Intermediate designs and flow fields for the optimal design and flow fields found in Figure 2.20.

flow speed is decreased resulting in an overestimation of elemental Reynolds number and thus penalization $\bar{\kappa} = 10^q H^e Re^e$. For similar reasons as for the $Re = 0.2$ NSDP$^{(b)}$ convergence, this over-penalization causes the $Re = 200$ NSDP$^{(a)}$ optimization to converge to an inferior local optimum. However, the $Re = 200$ VANS$^{(a)}$ optimization uses the same $\bar{\kappa}$ as can be seen in Table 2.10 and finds an optimum containing similar error $e^{\mathbf{v}}$ as the $Re = 200$ NSDP$^{(a)}$ optimization as can be seen in Table 2.11. For similar errors the

VANS$^{(a)}$ model thus shows improved convergence behavior over the NSDP$^{(a)}$ case. Furthermore, the NSDP$^{(a)}$ $Re = 200$ convergence in Figure 2.22b shows oscillations and an increase in objective value starting around iteration 80. These disturbances are caused by the small solid island to the left of the wall as shown in Figure 2.23d disappearing. The small solid island itself increases pressure drop locally as flow moves around it. However, removing it causes a non-linear reaction of the flow in the remainder of the channel and consequently total pressure drop to increase. Moreover, the VANS$^{(a)}$ convergence in Figure 2.22b shows a longer range of oscillations and increasing objective during iterations 80-200, which is caused by the porous material in the intermediate designs as shown in Figure 2.23c and the non-linearity of the Navier-Stokes equations. While the design changes it evolves through some gray material states where flow is less penalized causing more flow through the porous domain which increases power dissipation and thus pressure drop, similar to the convergence for the NSDP$^{(b)}$ based optimization. Furthermore while small changes at the boundaries may seem to benefit the objective looking at the linear sensitivities, non-linear effects on the flow actually increase the objective.

Penalizing intermediate designs too much by setting $q > 1$ resulting in large $\bar{\kappa}$ may thus cause convergence to inferior local optima as shown in Figure 2.21. However, if an appropriate penalization is used ($q = 1$) low Reynolds optimization problems converge nicely using both the VANS$^{(a)}$ (with $\underline{\alpha} = 10^{-1}$) and NSDP$^{(a)}$ (with $\underline{\alpha} = 10^{-2}$) models. However, for moderate Reynolds optimization estimating the elemental Reynolds number introduces uncertainties in the parameter settings which may again cause convergence to inferior local optima, a property which will be examined in more detail in the next section.

TUNING THE NSDP PARAMETERS FOR MODERATE REYNOLDS OPTIMIZATION

For the $Re = 0.2$ NSDP$^{(a)}$ based optimization, lower q (and thus lower $\bar{\kappa}$) resulted in an improved design. As the $Re = 200$ NSDP$^{(a)}$ based optimization also converges to a local optimum, lowering the maximum penalization may cause improved convergence behavior. We thus investigate optimal designs and convergence for lower q (lowering $\bar{\kappa}$ and $\overline{K}$), increased $\underline{\alpha}$ (keeping $\bar{\kappa}$ constant but lowering $\overline{K}$) or both, resulting in the optimization parameters as in Table 2.12 (the same material parameters as in Table 2.10 are used). Using $\underline{\alpha} = 10^{-2}$ and lowered $q = 0$ (NSDP$^{(c)}$) results in the inferior local optimum in Fig-

	Re, Re^e	q	$\underline{\alpha}$	$\mu\bar{\kappa}$	$\overline{K}_h$
NSDP$^{(c)}$	200, 10	0	10^{-2}	$2 \cdot 10^1$	$2 \cdot 10^3$
NSDP$^{(d)}$	200, 10	1	10^{-1}	$2 \cdot 10^2$	$2 \cdot 10^3$
NSDP$^{(e)}$	200, 10	0	10^{-1}	$2 \cdot 10^1$	$2 \cdot 10^2$

Table 2.12: The tweaked optimization parameters for the problem in Figure 2.19. Optimized designs can be found in Figure 2.24 and the resulting objective values and errors in Table 2.13.

ure 2.24a. Although it converged to a distinct solid/fluid design and the reference objective value decreased to $f^* = 0.4699$ as found in Table 2.13, it still performed worse than the VANS optimum in Table 2.11 with $f^* = 0.4209$. Furthermore, we note that the VANS$^{\text{R}}$

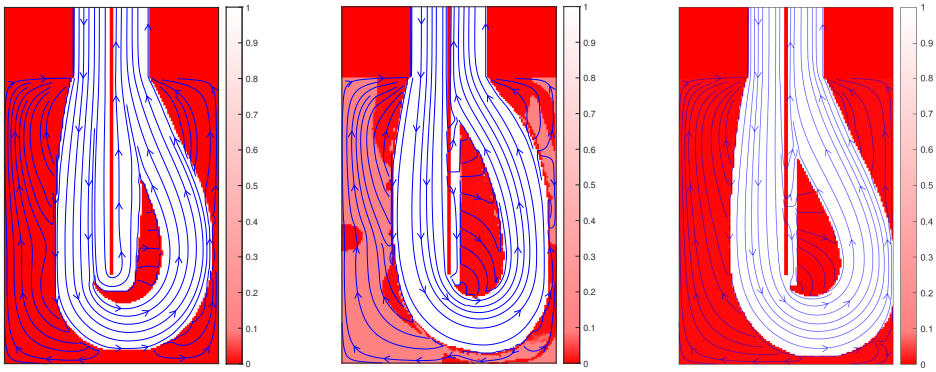
	NSDP$^{(c)}$	NSDP$^{(d)}$	NSDP$^{(e)}$
	$q = 0$	$q = 1$	$q = 0$
	$\underline{\alpha} = 10^{-2}$	$\underline{\alpha} = 10^{-1}$	$\underline{\alpha} = 10^{-1}$
$e^{\mathbf{v}}$	$3.85 \cdot 10^{-1}$	$4.39 \cdot 10^{-2}$	$2.63 \cdot 10^{-1}$
u_l	$4.51 \cdot 10^{-3}$	$7.84 \cdot 10^{-3}$	$4.53 \cdot 10^{-2}$
f^*	0.4514	0.4525	0.4166
	VANS$^{\mathrm{R}}$	VANS$^{\mathrm{R}}$	VANS$^{\mathrm{R}}$
$e^{\mathbf{v}}$	$1.045 \cdot 10^{-1}$	$2.40 \cdot 10^{-8}$	$1.96 \cdot 10^{-7}$
u_l	$5.20 \cdot 10^{-13}$	$9.13 \cdot 10^{-14}$	$6.21 \cdot 10^{-13}$
f^*	0.4699	0.4678	0.4312

Table 2.13: The optimized objectives and errors for the problem defined in Figure 2.19 using the material parameters in Table 2.10 and optimization parameters in Table 2.12. Optimized designs can be found in Figure 2.20.

reference error for the NSDP$^{(c)}$ based design in Table 2.13 is $e^{\mathbf{v}} = 1.045 \cdot 10^{-1}$ which is caused by the small tip on the left of the solid island in Figure 2.24a on which flow impinges at high speeds. Subsequently, using $q = 1$ and increased $\underline{\alpha} = 10^{-1}$ (NSDP$^{(d)}$) results in the design found in Figure 2.24b which suffers from the same problems as the initially optimized design in Figure 2.20d and is an inferior local optimum with $f^* = 0.4678$ as found in Table 2.13. However, lowering $q = 0$ and increasing $\underline{\alpha} = 10^{-1}$ (NSDP$^{(e)}$) results in the converged discrete solid/fluid design in Figure 2.24c with a reference objective of $f^* = 0.4312$ which is only 2.45% worse than the VANS$^{(a)}$ based reference objective in Table 2.11. Improved convergence behavior however came at the cost of increased errors as $e^{\mathbf{v}}$ increased by one order to $e^{\mathbf{v}} = 2.63 \cdot 10^{-1}$ and u_l increased by two orders to $4.53 \cdot 10^{-2}$ compared to the $Re = 200$ errors in Table 2.11. Furthermore, convergence of the objectives can be found in Figure 2.25 where the NSDP$^{(e)}$ based optimization shows the most stable convergence behavior. Solution precision is thus traded for convergence behavior when using the NSDP equations, whereas the VANS model is able to attain precise solutions and good objective convergence for moderate Reynolds optimization.

2.7.2. SOLUTION PRECISION VERSUS DESIGN CONVERGENCE

In the previous section we have shown that by reducing the Darcy penalization and increasing errors design convergence can be improved for the NSDP equations. In this sections we study the balance between solution precision and design convergence and establish that the VANS equations attain better convergence behavior for lower errors than the NSDP equations. We optimize the problem as shown in Figure 2.26, which is inspired by one of the problems by Olesen et al. (2006) and use the material and problem parameters as shown in Table 2.14. For the NSDP equations all parameter settings except those in NSDP$^{(b)}$ (with $q = 2$) are investigated. Furthermore, beside the VANS$^{(a)}$



(a) NSDP$^{(c)}$, $q = 0$, $\underline{\alpha} = 10^{-2}$ (b) NSDP$^{(d)}$, $q = 1$, $\underline{\alpha} = 10^{-1}$ (c) NSDP$^{(e)}$, $q = 0$, $\underline{\alpha} = 10^{-1}$

Figure 2.24: Optimal designs for $Re = 200$ computed using the NSDP equations, the optimization parameters in Table 2.12 and material parameters in Table 2.10.

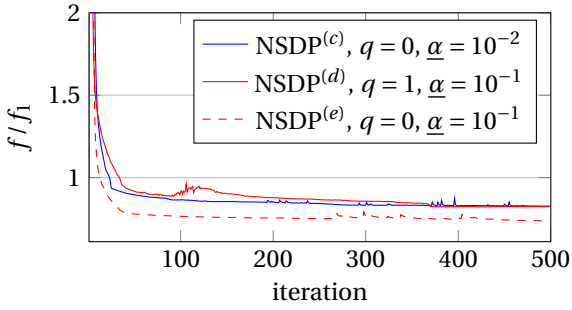


Figure 2.25: The convergence of the designs in Figure 2.24 with errors and objectives in Table 2.13. Objective values are normalized using the objective at the first design iteration f_1. During the first few design iterations objective values drastically increase by the addition of solid material to the fluid design domain, however normalized objective values are cut of at 2 to be able to inspect the convergence behavior.

parameters which are the same as used in the previous sections, we also use the VANS$^{(b)}$ parameters with lowered $q = 0$ to investigate if this also leads to improved designs for the VANS equations. We minimize pressure drop for flow through two channels which flow in opposite direction using maximum fluid volume fraction $V_f = 0.4$. Inspecting the optimized results in Olesen et al. (2006), the optimum is expected to consists of two separate channels of constant height L which would result in a pressure drop of:

$$\Delta p = \int_{\Gamma_{in}} p d\Gamma - \int_{\Gamma_{out}} p d\Gamma = 112 \frac{\mu u_{max}}{L}, \tag{2.145}$$

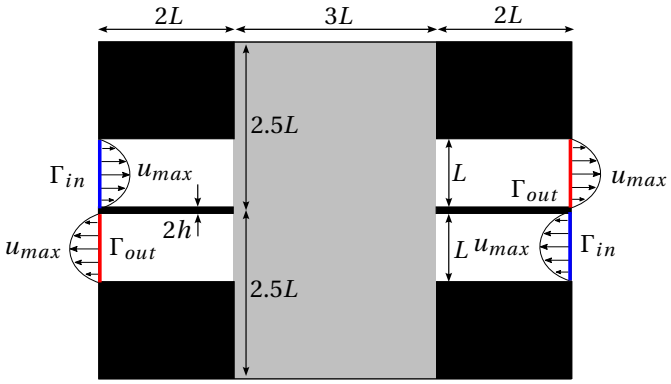


Figure 2.26: An optimization problem which minimizes pressure drop for flow through 2 channels. On the black thin wall at the inlet/outlet no slip and no penetration conditions are explicitly applied. Parabolic flow profiles are applied at all inlets Γ_{in} and outlets Γ_{out}. The optimization procedure is initialized using a completely fluid design domain (light gray).

ρ	μ	u_{max}	L	h	H^e	Re, Re^e	V_f
1 [kgm^{-3}]	1/180 [Pas]	1 [ms^{-1}]	1 [m]	0.05 [m]	400 [m^{-2}]	180, 9	0.4

	q	$\underline{\alpha}$	$\mu\bar{\kappa}$	$\overline{K}_h$
NSDP$^{(a)}$	1	10^{-2}	$2\cdot10^2$	$2\cdot10^4$
NSDP$^{(d)}$	1	10^{-1}	$2\cdot10^2$	$2\cdot10^3$
NSDP$^{(c)}$	0	10^{-2}	$2\cdot10^1$	$2\cdot10^3$
NSDP$^{(e)}$	0	10^{-1}	$2\cdot10^1$	$2\cdot10^2$
VANS$^{(a)}$	1	10^{-1}	$2\cdot10^2$	$2\cdot10^3$
VANS$^{(b)}$	0	10^{-1}	$2\cdot10^1$	$2\cdot10^2$

Table 2.14: The material and problem parameters for the optimization problem in Figure 2.26. for the NSDP equations all previously used parameter sets except for NSDP$^{(b)}$ are investigated. For the VANS equations we use the standard parameter set (VANS$^{(a)}$) but also investigate the case where $q = 0$ resulting in the VANS$^{(b)}$ parameter set.

where we assumed constant parabolic flow throughout the two channels. We thus normalize the pressure drop objective as:

$$f = \left(\int_{\Gamma_{in}} p d\Gamma - \int_{\Gamma_{out}} p d\Gamma \right) / \Delta p. \tag{2.146}$$

However, for this objective to be achieved the design would have to include a two element thick horizontal wall through the center of the domain. Excessive flow leakage and consequent errors in pressure drop as found in Appendix 2.D might thus be a problem for this optimization problem, and this may lead to alternative solutions.

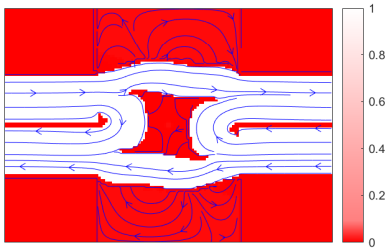
ANALYSIS OF THE RESULTS

Inspecting the optimized designs in Figure 2.27 we find that none of the optimization procedures converged to the theoretical optimum which is confirmed by the objective values in Table 2.15 for which $f^* > 1$. However, the VANS$^{(b)}$ based design in Figure 2.27f comes close to the theoretical optimum and finds a reference objective of $f^* = 1.028$ as found in Table 2.15. Increasing $q = 1$ for the VANS$^{(a)}$ based design however pushes the optimizer in a local optimum which splits the flow of all four inlets/outlets and subsequently finds an inferior local optimum which performs worse as $f^* = 1.269$. The spurious flow errors are however quite similar with $e^{\mathbf{v}} = 2.59 \cdot 10^{-1}$ for the VANS$^{(b)}$ based design and only slightly lower $e^{\mathbf{v}} = 1.00 \cdot 10^{-1}$ for the VANS$^{(a)}$ design. Nonetheless, flow errors of $e^{\mathbf{v}} = \mathcal{O}(-q) = \mathcal{O}(-1)$ are expected for the VANS$^{(a)}$ design as predicted in Equations 2.113, and in this particular design are mainly caused by the thin features at the upper right and lower left of the center island. Moreover, as these kind of thin features which guide the flow require sufficient penalization to be found by the optimizer, the VANS$^{(a)}$ based design is unlikely to be found by the VANS$^{(b)}$ based optimization procedure. In addition flow error $e^{\mathbf{v}}$ for the VANS$^{(b)}$ based design is quite low due to the objective of pressure drop minimization. If spurious flow is large, much energy is dissipated by flow through the solid domain and the optimizer thus tends to reduce spurious flow if possible. Furthermore, the convergence behavior for both VANS based designs in Figure 2.29 shows both designs converge relatively smoothly.

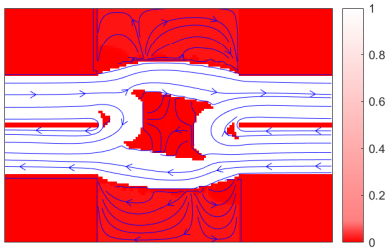
	NSDP$^{(a)}$	NSDP$^{(d)}$	NSDP$^{(c)}$	NSDP$^{(e)}$	VANS$^{(a)}$	VANS$^{(b)}$
	$q = 1$ $\underline{\alpha} = 10^{-2}$	$q = 1$ $\underline{\alpha} = 10^{-1}$	$q = 0$ $\underline{\alpha} = 10^{-2}$	$q = 0$ $\underline{\alpha} = 10^{-1}$	$q = 1$ $\underline{\alpha} = 10^{-1}$	$q = 0$ $\underline{\alpha} = 10^{-1}$
$e^{\mathbf{v}}$	$1.20 \cdot 10^{-1}$	$1.54 \cdot 10^{-1}$	$3.70 \cdot 10^{-1}$	$2.74 \cdot 10^{0}$	$1.00 \cdot 10^{-1}$	$2.59 \cdot 10^{-1}$
f^*	1.295	1.376	1.059	2.045	1.274	1.020
	VANS$^{\mathrm{R}}$	VANS$^{\mathrm{R}}$	VANS$^{\mathrm{R}}$	VANS$^{\mathrm{R}}$	VANS$^{\mathrm{R}}$	VANS$^{\mathrm{R}}$
$e^{\mathbf{v}}$	$7.98 \cdot 10^{-8}$	$8.64 \cdot 10^{-8}$	$3.705 \cdot 10^{-8}$	$1.97 \cdot 10^{-6}$	$7.62 \cdot 10^{-8}$	$2.99 \cdot 10^{-8}$
f^*	1.363	1.445	1.130	2.493	1.269	1.028

Table 2.15: The optimized objectives and errors for the problem defined in Figure 2.26 using the parameters in Table 2.14. Optimized designs can be found in Figure 2.27.

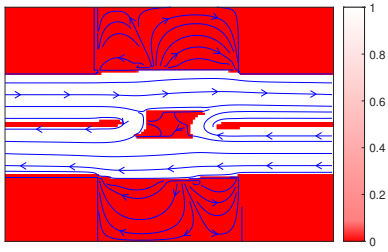
Subsequently, we investigate the NSDP$^{(a)}$ and NSDP$^{(d)}$ based designs which use $q = 1$. Both designs in Figures 2.27a and 2.27b are similar to the VANS$^{(a)}$ design in Figure 2.27e, although they perform worse in terms of objective as shown in Table 2.15. The decreased objective is mainly caused by the small solid islands at the tip of the thin walls separating the inlets and outlets. Similar small solid islands can be found in the design at design iteration 20 of the VANS$^{(a)}$ optimization procedure as shown in Figure 2.28. However, whereas the NSDP$^{(a)}$ and NSDP$^{(d)}$ based designs solidify the solid islands, using the VANS$^{(a)}$ model the islands are slowly removed over iterations 20-40 resulting in the design in Figure 2.27e.



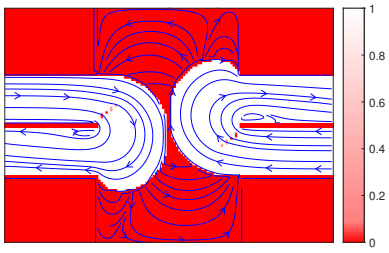
(a) NSDP$^{(a)}$, $q = 1$, $\underline{\alpha} = 10^{-2}$



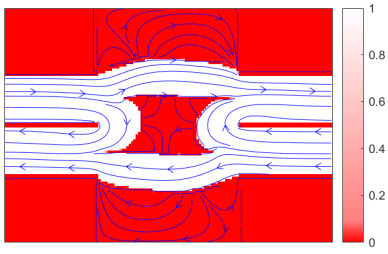
(b) NSDP$^{(d)}$, $q = 1$, $\underline{\alpha} = 10^{-1}$



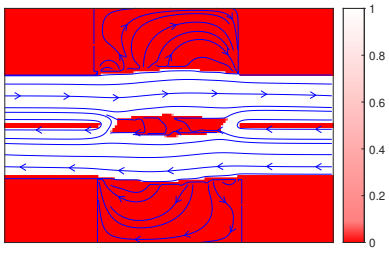
(c) NSDP$^{(c)}$, $q = 0$, $\underline{\alpha} = 10^{-2}$



(d) NSDP$^{(e)}$, $q = 0$, $\underline{\alpha} = 10^{-1}$



(e) VANS$^{(a)}$, $q = 1$, $\underline{\alpha} = 10^{-1}$



(f) VANS$^{(b)}$, $q = 0$, $\underline{\alpha} = 10^{-1}$

Figure 2.27: The optimal design and flow fields for the problem in Figure 2.26 using the parameters in Table 2.14. Although flow through the solid (red) material is plotted, spurious solid flow remains low as shown by the errors in Table 2.15.

Comparing the NSDP based designs which use $q = 0$, the NSDP$^{(c)}$ based design in Figure 2.27c seems an intermediate design between the VANS$^{(a)}$ and VANS$^{(b)}$ based designs in Figures 2.27e and 2.27f which is confirmed by the reference objective value of $f^* = 1.130$ in Table 2.15. The NSDP$^{(c)}$ based design seems to get stuck in an inferior local optimum and is found to contain less thin features than the VANS$^{(a)}$ based design in Figure 2.27e as it is unable to sufficiently penalize flow in these features. Moreover, the NSDP$^{(e)}$ based design in Figure 2.27d shows a completely different topology and

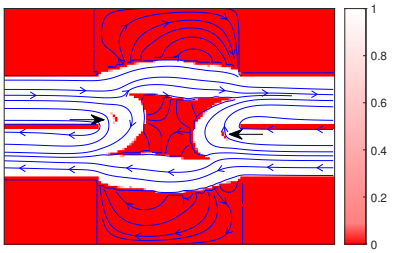


Figure 2.28: The intermediate design at iteration 20 for the problem in Figure 2.26 computed using the VANS$^{(a)}$ model. Note the two solid islands at the arrow tips which are slowly removed over design iterations 20-40 resulting in the optimal design in Figure 2.27e.

performs the worst with a reference objective of $f^* = 2.493$ as found in Table 2.15. Additionally, a large difference between optimized ($f^* = 2.045$) and reference objective is found, which is caused by the small porous islands in the fluid channels. These porous islands slow down the flow right before it bends around the thin wall, and thus smooth the change in direction of the flow but also increase error $e^{\mathbf{v}} = 2.74$. As the maximum penalization is low for the NSDP$^{(e)}$ based model it prefers to smooth the flow at the start of the bend even if more energy is dissipated and pressure drop is increased by flow through the porous material. However, post processing the design and computing the objective using a more accurate model increases the objective value by 21.9% from $f^* = 2.045$ to $f^* = 2.493$. Using the NSDP$^{(e)}$ based optimization procedure thus leads to the worst design containing the biggest errors for the problem in this section while it resulted in the best design attainable by the NSDP equations in Section 2.7.1 as seen in Figure 2.24c.

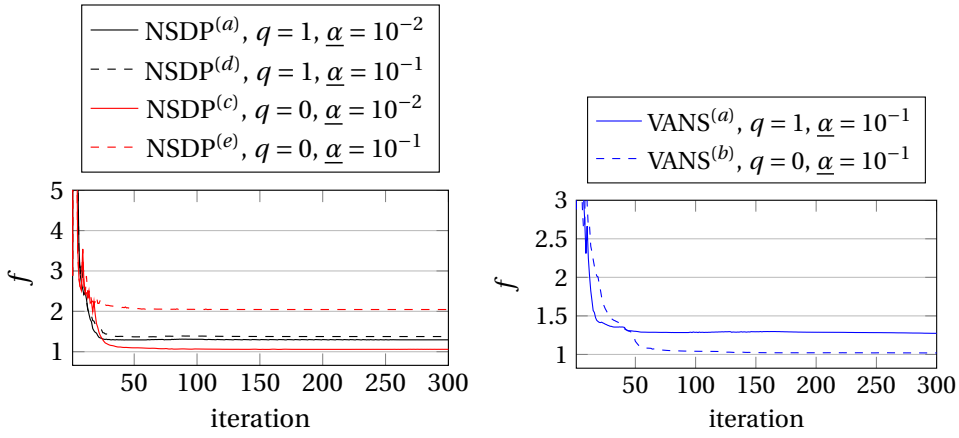


Figure 2.29: The convergence of the designs in Figure 2.27 with errors and objectives in Table 2.15. During the first few design iterations objective values drastically increase by the addition of solid material to the fluid design domain, however objective values are cut of at 5 and 3 to be able to inspect the convergence behavior.

FINDINGS

The preceding numerical experiments indicate that increasing solution precision (higher q and lower $\underline{\alpha}$) may lead to convergence to inferior local optima for both VANS and NSDP based moderate Reynolds optimization procedures. However, the VANS equations generally show better design convergence for higher flow penalization and solution precision and require less tuning of the optimization parameters. For low Reynolds optimization both VANS and NSDP based optimization procedures show good convergence behavior for equally precise flow solutions when appropriate optimization parameters are selected (NSDP$^{(a)}$ and VANS$^{(a)}$). Furthermore, in our framework parameter q is used to select an appropriate penalization at intermediate designs where $\alpha \approx 0.5$, and $\underline{\alpha}$ is used to increase flow inhibition and solution precision in the solid domain where $\alpha = \underline{\alpha}$ as shown in Section 2.4.4. Moreover, as shown in Section 2.3.1 parameter $\underline{\alpha}$ can be compared to parameter $\tilde{q}$ which is often tuned to set convexity of the Darcy interpolation and lower penalization in intermediate designs as discussed by Borrvall and Petersson (2003). However, in our approach we precisely define the penalization in intermediate designs $\bar{\kappa}$ using the derivations in Section 2.4. The approach thus differs slightly as we precisely set the penalization in intermediate density areas using q and increase maximum penalization $\overline{K}$ by lowering $\underline{\alpha}$, instead of setting the penalization in solid density areas and lowering the penalization in intermediate density areas using a convex interpolation.

2.8. CONCLUSIONS AND RECOMMENDATIONS

In this work we have introduced the VANS (Volume Averaged Navier-Stokes) equations for solid/fluid topology optimization and shown their applicability and advantages. Using volume averaging we were able to create a theoretically consistent framework for introducing design variables in the Navier-Stokes equations. The NSDP (Navier-Stokes with Darcy Penalization) equations often used in topology optimization are shown to be a simplification of the VANS equations. Moreover, two main improvements for solid/fluid topology optimization are found:

1. Lower bounds on the Darcy penalization are theoretically derived such that for both the VANS and NSDP equations flow is sufficiently penalized in the solid domain while keeping flow penalization in intermediate gray designs at a minimum to prevent convergence to inferior local optima.

2. Compared to the NSDP equations, the VANS equations are shown to require less parameter tuning and display improved design convergence for similarly accurate flow solutions.

Furthermore, instead of relating the appropriate Darcy penalization to problem specific parameters, it is related to the mesh size and an elemental Reynolds number, reducing the amount of required parameter tuning. However, the lower bounds are not exact, and for changing topologies (and mainly for changing elemental Reynolds numbers) flow leakage may remain hard to estimate *a priori*. A solution to this problem might be the addition of the Forchheimer penalization which scales quadratically with flow speeds and thus may not require an estimate of flow speeds and elemental Reynolds number to find

an appropriate magnitude for flow reduction. Moreover, viability of the lower bounds in other computational frameworks (such as the Finite Element Method instead of FV) remains an open question. We thus recommend performing a representative study on a range of mesh sizes and elemental Reynolds numbers as shown in Section 2.6 for implementation of the lower bounds in other computational frameworks. However, when the bounds are confirmed no representative studies should be required for individual optimization problems containing different mesh sizes and/or elemental Reynolds numbers.

Another opportunity for further research is to apply volume averaging techniques to other physics. A logical next step would be their application to turbulent flow optimization as the Reynolds average often used for turbulence modeling shows many similarities to the volume average. Besides turbulent optimization thermal and even mechanical optimization models could be investigated using volume averaging techniques, such that physically consistent models and interpretations can be constructed. Furthermore, in this work the second Brinkman correction is interpreted as the forces in the solid material which support the fluid domain viscous stresses at the solid/fluid interface. It could thus be included in solid/fluid interaction optimization to more accurately couple the fluid domain forces to the solid domain. Another field which could benefit from the averaging techniques in this paper could be the pseudo-3D topography optimization as discussed by Alexandersen (2022). In this work a slowly varying distance between two plates is optimized and an augmentation of the conservation equation needs to be introduced to attain accurate solutions. We note the possibility of approaching this problem using superficially averaged flow between the plates such that the continuity equation does not need to be changed, as was the case in this work.

2.A. DERIVATION OF THE VANS EQUATIONS

To give some more insight into the derivation of the VANS equations the volume averaged continuity equations and viscous terms will be derived in this appendix. For a more detailed and complete derivation of the VANS equations we refer the reader to the works of Whitaker (1996), and Ochoa-Tapia and Whitaker (1995). Firstly, we derive the volume averaged continuity equation by applying the averaging theorem from Equation 2.3 to pull the divergence operator out of the average:

$$\langle \nabla \cdot \mathbf{v} \rangle^{s\phi} = \nabla \cdot \langle \mathbf{v} \rangle^{s\phi} + \frac{1}{V} \int_{\Gamma_{\phi\beta}} \mathbf{v} \cdot \boldsymbol{n}_\phi \, d\Gamma = 0. \tag{2.147}$$

Using the no-penetration condition ($\mathbf{v} \cdot \boldsymbol{n}_\phi = 0$) at the solid/fluid boundary $\Gamma_{\phi\beta}$, the boundary integral can be removed:

$$\nabla \cdot \langle \mathbf{v} \rangle^{s\phi} + \frac{1}{V} \int_{\Gamma_{\phi\beta}} \cancelto{0}{\mathbf{v} \cdot \boldsymbol{n}_\phi} \, d\Gamma = \nabla \cdot \langle \mathbf{v} \rangle^{s\phi} = 0, \tag{2.148}$$

resulting in the averaged continuity equation.

The derivation of the averaged viscous term ($\langle \mu \nabla^2 \mathbf{v} \rangle^{s\phi} = \langle \mu \nabla \cdot \nabla \mathbf{v} \rangle^{s\phi}$) is slightly more complex and will result in the first/second Brinkman corrections and a part of the Darcy

penalization. Firstly, the averaging theorem from Equation 2.3 is used to pull the divergence operator out of the average:

$$\langle \mu \nabla \cdot \nabla \mathbf{v} \rangle^{s\phi} = \mu \nabla \cdot \langle \nabla \mathbf{v} \rangle^{s\phi} + \frac{\mu}{V} \int_{\phi\beta} \nabla \mathbf{v} \cdot \boldsymbol{n}_\phi d\Gamma, \tag{2.149}$$

where we assumed μ to be constant and pulled it out of the averaging operators. The first term on the right-hand side of Equation 2.149 can again be simplified using the averaging theorem resulting in the first Brinkman correction:

$$\mu \nabla \cdot \langle \nabla \mathbf{v} \rangle^{s\phi} = \mu \nabla \cdot \nabla \langle \mathbf{v} \rangle^{s\phi} + \nabla \cdot \frac{1}{V} \int_{\phi\beta} \cancelto{\mathbf{0}}{\mathbf{v}} \boldsymbol{n}_\phi^\intercal d\Gamma = \mu \nabla \cdot \nabla \langle \mathbf{v} \rangle^{s\phi}, \tag{2.150}$$

where we used the no-slip and no-penetration conditions at soid/fluid boundary $\Gamma_{\phi\beta}$ to assume that $\mathbf{v} = \mathbf{0}$. Subsequently, for the second term on the right-hand side of Equation 2.149 we assume separation of scales as shown in Equation 2.5 and assume the velocity can be split into its intrinsic volume average ($\langle \mathbf{v} \rangle^{i\phi}$) and deviational part ($\tilde{\mathbf{v}}$). The divergence of the velocity within an averaging volume can thus be rewritten as:

$$\nabla \mathbf{v} = \nabla \left(\langle \mathbf{v} \rangle^{i\phi} + \tilde{\mathbf{v}} \right) = \nabla \langle \mathbf{v} \rangle^{i\phi} + \nabla \tilde{\mathbf{v}}. \tag{2.151}$$

Furthermore, by assuming the averaged divergence $\nabla \langle \mathbf{v} \rangle^{i\phi}$ to be constant within the averaging domain, it can be removed from the boundary integral:

$$\begin{aligned} \frac{\mu}{V} \int_{\phi\beta} \nabla \mathbf{v} \cdot \boldsymbol{n}_\phi d\Gamma &= \frac{\mu}{V} \int_{\phi\beta} \left(\nabla \langle \mathbf{v} \rangle^{i\phi} + \nabla \tilde{\mathbf{v}} \right) \cdot \boldsymbol{n}_\phi d\Gamma \\ &= \frac{\mu}{V} \nabla \langle \mathbf{v} \rangle^{i\phi} \cdot \int_{\phi\beta} \boldsymbol{n}_\phi d\Gamma + \frac{\mu}{V} \int_{\phi\beta} \nabla \tilde{\mathbf{v}} \cdot \boldsymbol{n}_\phi d\Gamma. \end{aligned} \tag{2.152}$$

Subsequently, we use the fact that the gradient of the volume fraction can be rewritten using Equation 2.4 as $\nabla \alpha_\phi = -\frac{1}{V} \int_{\Gamma_{\phi\beta}} \boldsymbol{n}_\phi d\Gamma$, and simplify the boundary integral as:

$$\begin{aligned} \frac{\mu}{V} \nabla \langle \mathbf{v} \rangle^{i\phi} \cdot \int_{\phi\beta} \boldsymbol{n}_\phi d\Gamma + \frac{\mu}{V} \int_{\phi\beta} \nabla \tilde{\mathbf{v}} \cdot \boldsymbol{n}_\phi d\Gamma &= -\mu \nabla \langle \mathbf{v} \rangle^{i\phi} \cdot \nabla \alpha_\phi + \frac{\mu}{V} \int_{\phi\beta} \nabla \tilde{\mathbf{v}} \cdot \boldsymbol{n}_\phi d\Gamma \\ &= -\mu \nabla \alpha_\phi \cdot \frac{\langle \mathbf{v} \rangle^{s\phi}}{\alpha_\phi} + \frac{\mu}{V} \int_{\phi\beta} \nabla \tilde{\mathbf{v}} \cdot \boldsymbol{n}_\phi, \end{aligned} \tag{2.153}$$

where the superficial volume average is introduced ($\langle \mathbf{v} \rangle^{i\phi} = \langle \mathbf{v} \rangle^{s\phi} / \alpha_\phi$) as this is used in the final VANS equations. Gathering all terms in Equations 2.150 and 2.153, the volume averaged viscous stresses are found as:

$$\langle \mu \nabla \cdot \nabla \mathbf{v} \rangle^{s\phi} = \mu \nabla \cdot \nabla \langle \mathbf{v} \rangle^{s\phi} - \mu \nabla \alpha_\phi \cdot \frac{\langle \mathbf{v} \rangle^{s\phi}}{\alpha_\phi} + \frac{\mu}{V} \int_{\phi\beta} \nabla \tilde{\mathbf{v}} \cdot \boldsymbol{n}_\phi, \tag{2.154}$$

where the first term is the first Brinkman correction, the second term is the second Brinkman correction and the third term (in combination with a term containing devotional pressures) will be simplified as the Darcy penalization.

2.B. COMPARISON OF MAXIMUM PENALIZATION VALUES

In literature, different interpolation functions for Darcy interpolation $-K(\alpha)\cdot\mathbf{v}$ and different definitions for a maximum penalization $\overline{K}$ can be found. Similar to the maximum penalization for $Re^e > 1$ in this work, Reynolds dependent penalization values have been used. However, these different forms of penalization are often argued from non-dimensional Navier-Stokes equations and remain similar to the commonly used maximum penalization by Olesen et al. (2006):

$$\overline{K} = \frac{\mu}{L^2 Da} \gg \frac{\mu}{L^2}, \tag{2.155}$$

where $Da \ll 1$ is the Darcy number used to scale the penalization, and which is based on the Navier-Stokes Equations:

$$\rho\mathbf{v}\cdot\nabla\mathbf{v} - \mu\nabla\left(\nabla\mathbf{v} + \nabla\mathbf{v}^\intercal\right) + \nabla p + K_A(\alpha)\cdot\mathbf{v} = \boldsymbol{s}. \tag{2.156}$$

We examine the maximum penalization by Alexandersen et al. (2013):

$$\overline{K}_A = \frac{1}{ReDa} \gg \frac{1}{Re}, \tag{2.157}$$

where $Da \ll 1$ and the maximum penalization by Kondoh et al. (2012):

$$\overline{K}_K = \left(1 + \frac{1}{Re}\right)\chi \gg \left(1 + \frac{1}{Re}\right) \tag{2.158}$$

where $\chi \gg 1$. Both penalizations in Equations 2.157 and 2.158 are based on the non-dimensional Navier-Stokes equations:

$$\tilde{\mathbf{v}}\cdot\tilde{\nabla}\tilde{\mathbf{v}} - \frac{1}{Re}\tilde{\nabla}\left(\tilde{\nabla}\tilde{\mathbf{v}} + \tilde{\nabla}\tilde{\mathbf{v}}^\intercal\right) - \tilde{\nabla}\tilde{p} + K(\alpha)\cdot\tilde{\mathbf{v}} = \boldsymbol{s}, \tag{2.159}$$

where:

$$\begin{aligned} \tilde{\mathbf{v}} &= \frac{\mathbf{v}}{U}, \\ \tilde{p} &= \frac{p}{\rho U^2}, \\ \tilde{\boldsymbol{x}} &= \frac{\boldsymbol{x}}{L}, \\ \tilde{\nabla} &= L\nabla, \end{aligned} \tag{2.160}$$

are the non-dimensional velocity, pressure, coordinate vector and gradient operator respectively, based on characteristic length L and velocity U. We may however rewrite the non-dimensional Navier-Stokes equation into a form similar to Equation 2.156:

$$\rho^*\tilde{\mathbf{v}}\cdot\tilde{\nabla}\tilde{\mathbf{v}} - \mu^*\tilde{\nabla}\left(\tilde{\nabla}\tilde{\mathbf{v}} + \tilde{\nabla}\tilde{\mathbf{v}}^\intercal\right) - \tilde{\nabla}\tilde{p} + K(\alpha)\cdot\tilde{\mathbf{v}} = \boldsymbol{s}, \tag{2.161}$$

where $\rho^* \equiv 1$ and $\mu^* \equiv 1/Re$ can be regarded as a non-dimensional density and viscosity respectively and the problem has characteristic length and velocity $L^* \equiv 1$ and $U^* \equiv 1$ respectively. Subsequently, we rewrite the penalization by Alexandersen et al. (2013) as:

$$\overline{K}_A = \frac{1}{ReDa} = \frac{\mu^*}{Da} \gg \mu^* = \frac{\mu^*}{L^{*2}}, \tag{2.162}$$

since $1/L^{*2} = 1$ and we note that this form is the same as the one in Equation 2.155, which is a logical consequence of the fact that Alexandersen et al. (2013) argue their penalization from the dimensional Navier-Stokes equations. Moreover, the penalization by Kondoh et al. (2012) can be rewritten using the formulation in Equation 2.161 as:

$$\begin{aligned}\overline{K}_K &= \left(1 + \frac{1}{Re}\right)\chi = \left(\frac{\rho^* U^*}{L^*} + \frac{\mu^*}{L^{*2}}\right)\chi \\ &\gg \left(\frac{\rho^* U^*}{L^*} + \frac{\mu^*}{L^{*2}}\right).\end{aligned} \tag{2.163}$$

For high Reynolds numbers $\mu^* = 1/Re \ll 1$ and thus $\overline{K}_K \approx \chi(\rho^* U^*)/L^* \gg (\rho^* U^*)/L^*$, which shows similarities to our penalization for $Re^e > 1$ as defined in Equation 2.110:

$$\overline{K}_h = \frac{10^q \rho \,|\mathbf{v}^f|}{h\underline{\alpha}} \gg \frac{\rho \,|\mathbf{v}^f|}{h}. \tag{2.164}$$

However, $\overline{K}_h$ differs significantly from $\overline{K}_K$ in the fact that it scales with h instead of L^*. For low Reynolds numbers $\mu^* = 1/Re \gg 1$ and thus $\overline{K}_K \approx \chi\mu^*/L^{*2} \gg \mu^*/L^{*2}$, and we again retrieve a similar maximum penalization as in Equations 2.155 and 2.162

2.C. FINITE DIFFERENCE SENSITIVITY VERIFICATION

To investigate the validity of the adjoint sensitivity computation as presented in Section 2.5 a Finite Difference (FD) sensitivity verification is performed on the problem in Figure 2.30 using the parameters in Table 2.16. Two cases are examined, one where the porous "solid" domain consists of volume fraction $\underline{\alpha} = 0.1$ and one where it consists of $\underline{\alpha} = 0.5$. Sensitivities are thus verified for converged designs containing solid domains of volume fraction $\alpha = 0.1$ and for intermediate designs containing gray areas of volume fraction $\alpha = 0.5$. Furthermore, the (elemental) Reynolds number can be computed as:

$$\begin{aligned}Re &= \frac{\rho \overline{u} L}{\mu} = 10, \\ Re^e &= \frac{\rho \overline{u} h}{\mu} = 1,\end{aligned} \tag{2.165}$$

and we expect both viscous and inertial effects to be relevant for the sensitivity computation. The adjoint and FD sensitivities are compared using the objective of minimal

μ	ρ	$\overline{u}$	p_o	L	h	q	$\underline{\alpha}$	e
10^{-1} [Pa s]	1 [kg m^{-3}]	1 [m s^{-1}]	1 [Pa]	1 [m]	0.1 [m]	1	0.1, 0.5	10^{-5}

Table 2.16: The material and problem parameters for the flow problem in Figure 2.30, when the adjoint sensitivity analysis is compared to a Finite Difference sensitivity analysis.

pressure drop:

$$f = \int_{\Gamma_{in}} p d\Gamma - \int_{\Gamma_{out}} p d\Gamma, \tag{2.166}$$

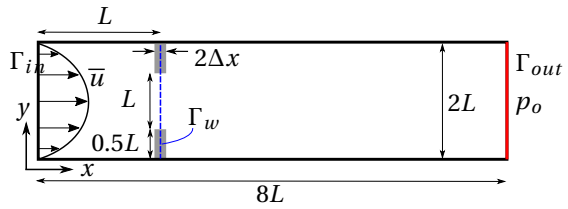


Figure 2.30: A 2D channel with parabolic inflow applied at the left inlet Γ_{in} and constant pressure applied at the right outlet Γ_{out}. At $x = L$ two small porous solid walls of thickness $2h$ ($\Delta x = \Delta y = h$) and volume fraction $\underline{\alpha}$ are inserted to inhibit flow. Sensitivity values are computed in the wall elements to the right of Γ_w at $x = L + h/2$.

where Γ_{in} is the flow inlet boundary and Γ_{out} is the pressure outlet in Figure 2.30. To verify the adjoint sensitivities complex step finite difference (CSFD) sensitivities (Martins et al., 2003) are computed. For the CSFD analysis the design variable for which the sensitivity is computed (s_k) is perturbed as:

$$\tilde{s}_k = s_k + e \cdot i, \tag{2.167}$$

where $\tilde{s}_k$ is the perturbed design variable, e is an offset as found in Table 2.16 and $i^2 = -1$ the imaginary number. Consequently, using the perturbed design variable the objective is computed as:

$$\tilde{f} = f_{Re} + f_{Im} \cdot i, \tag{2.168}$$

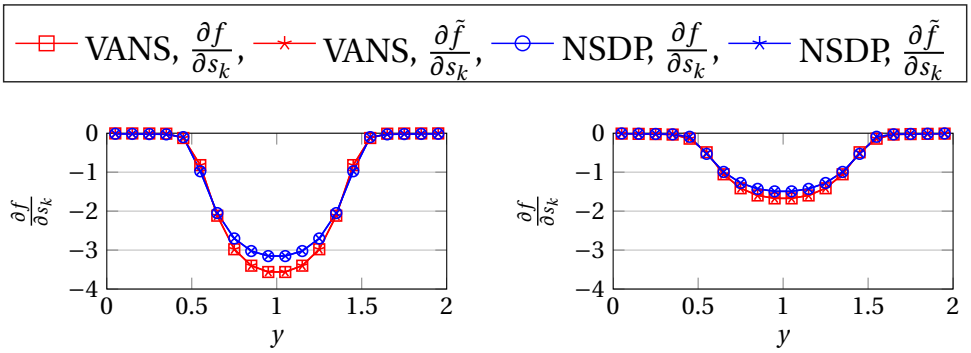
and if the offset is small enough ($e \ll 1$) the CSFD sensitivity can be computed as:

$$\frac{\partial \tilde{f}}{\partial s_k} = \frac{\mathrm{Im}(\tilde{f})}{e} + \mathcal{O}\left(e^2\right) = \frac{f_{Im}}{e} + \mathcal{O}\left(e^2\right), \tag{2.169}$$

where the FD sensitivity is thus accurate to the order $\mathcal{O}\left(e^2\right)$ and we approximate $\partial f/\partial s_k \approx \partial \tilde{f}/\partial s_k$. Using the problem shown in Figure 2.30, the adjoint and CSFD sensitivities are computed for all elements at $x = L + h/2$ such that sensitivities in the porous wall and in the fluid domain are computed, resulting in the sensitivities in Figures 2.31a and 2.31b. Moreover, relative errors in sensitivity are computed as:

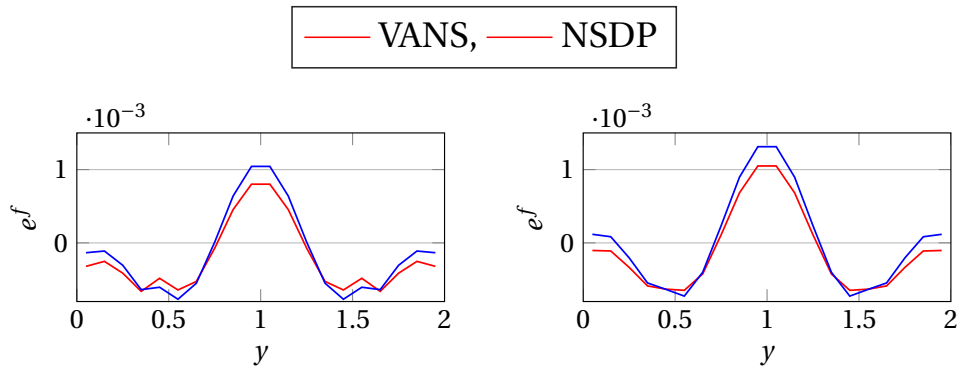
$$e^f = \frac{\frac{\partial \tilde{f}}{\partial s_k} - \frac{\partial f}{\partial s_k}}{\left|\frac{\partial \tilde{f}}{\partial s_k}\right|}, \tag{2.170}$$

where $\partial f/\partial s_k$ is the adjoint sensitivity and the errors can be found in Figures 2.31c and 2.31d. Visually the sensitivities in Figures 2.31a and 2.31b are the same, but a small relative error of magnitude 10^{-3} can be found in Figures 2.31c and 2.31d. The small error is caused by the numerical accuracy of the solution procedure as the velocity and pressure fields are updated until the relative change in fields is lower than 10^{-4}. Lowering this value also lowers the error in sensitivity.



(a) The adjoint ($\partial f/\partial s_k$) and CSFD ($\partial \tilde{f}/\partial s_k$) sensitivities for $\underline{\alpha} = 0.1$.

(b) The adjoint ($\partial f/\partial s_k$) and CSFD ($\partial \tilde{f}/\partial s_k$) sensitivities for $\underline{\alpha} = 0.5$.



(c) The error in adjoint sensitivity relative to the CSFD sensitivity computed using Equation 2.170 for $\underline{\alpha} = 0.1$.

(d) The error in adjoint sensitivity relative to the CSFD sensitivity computed using Equation 2.170 for $\underline{\alpha} = 0.5$.

Figure 2.31: The adjoint/CSFD sensitivities and resulting errors computed using Figure 2.30 and the parameters from Table 2.16.

2.D. FLOW LEAKAGE AND THE EFFECT OF THE SECOND BRINKMAN CORRECTION

To investigate flow leakage and the effect of the second Brinkman correction a simple channel of height $L = 2r$ as in Figure 2.32 is investigated. On the inlet and outlet respectively velocity $\mathbf{v}_{in} = [u_{in}, 0]^\intercal$ and relative pressure p_{out} are prescribed. A porous "*solid*" wall of two elements thick is inserted at $y = r$ and $y = -r$. Close to the inlet and outlet no-slip and no penetration conditions are prescribed on the wall to ensure a correct application of the boundary conditions. Some spurious flow through the porous walls is expected as we prescribe a constant relative pressure of $-10 \cdot p_{out}$ on the upper and lower boundaries and leave them open. Consequently, flow should develop into a parabolic

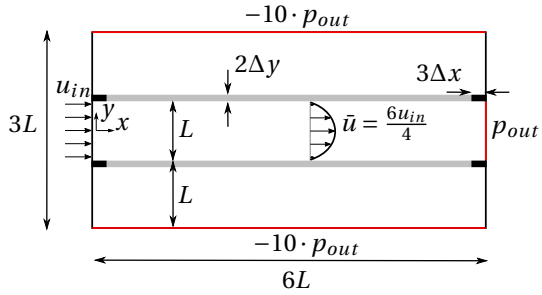


Figure 2.32: A 2D channel with constant inflow applied at the inlet and constant pressure applied at the outlet. Upper and lower walls at $y = \pm r$ are made of highly impermeable solids (gray) except for small solid walls (black) at the inlet/outlet on which no-slip and no penetration conditions are explicitly prescribed to ensure a correct application of the boundary conditions. The upper and lower boundaries are open and a constant relative pressure of $-10 \cdot p_{out}$ is applied on them.

μ	ρ	u_{in}	p_{out}	L	$\Delta x, \Delta y$	q	$\underline{\alpha}$
10^{-1} [Pas]	1 [kgm^{-3}]	1 [ms^{-1}]	1 [Pa]	1 [m]	$L/40$	1,2	10^{-1}

Table 2.17: The material and problem parameters for the flow problem in Figure 2.32, where q is used to compute $\bar{\kappa}$ using the equations from Table 2.2.

profile as:

$$u_r(y) = \left(1 - \frac{y^2}{r^2}\right) u_{max}, \tag{2.171}$$

where the maximum flow velocity can be computed from continuity as $u_{max} = 6u_{in}/4$. At $x = 5L$ reference solution $u_r(y)$ is used to compute flow error:

$$e^u_{5L} = \sqrt{\frac{\sum_{i \in I_{5L}} (u_r(y_i) - u_i)^2}{\sum_{i \in I_{5L}} (u_r(y_i))^2}} \cdot 100\%, \tag{2.172}$$

where index i is related to discrete DOF u_i at coordinates (x_i, y_i), and $I_{5L} = \{i \mid x_i = 5L \cup r > y_i > -r\}$. Furthermore, exact pressure drop $p^R_{,x} = -2\mu u_{max}/r^2$ is computed and used to define an error in pressure drop at $(x, 0)$:

$$e^{\Delta p}_x = \sqrt{\frac{(p^R_{,x} - p_{x,x})^2}{p^{R\,2}_{,x}}} \cdot 100\%, \tag{2.173}$$

where $p_{x,x}$ is the pressure drop computed using finite difference on the discrete solution at $(x, 0)$. Finally, flow leakage through porous walls is computed by numerical integration of the flow through the wall:

$$v_l = \frac{1}{Lu_{in}} \int_0^{6L} |\, v \,|\, dx \cdot 100\%, \tag{2.174}$$

at $y = r + \Delta y$. Subsequently, the parameters in Table 2.17 are used and the elemental Reynolds number is estimated as:

$$Re^e = \frac{\rho \mid \mathbf{v}^f \mid h}{\mu} = \frac{\rho u_{in} h}{\mu} = 0.25 \leq 1, \tag{2.175}$$

where $h = \Delta x = \Delta y$, and we approximate $\mid \mathbf{v}^f \mid \approx u_{in}$ using the inlet velocity. For both the NSDP and VANS equations the penalization was thus computed following Table 2.2 as:

$$\bar{\kappa} = \frac{10^q}{h^2}. \tag{2.176}$$

	e^u_{5L}	$e^{\Delta p}_{4L}$	$e^{\Delta p}_{5L}$	v_l
VANS, $q = 1$	4.34%	3.53%	4.07%	4.56%
NSDP, $q = 1$	32.8%	36.3%	40.2%	36.6%
NSDP, $q = 2$	5.64%	9.43%	9.86%	3.93%

Table 2.18: The errors in flow profile and pressure drop for the problem in Figure 2.32 with results in Figure 2.33.

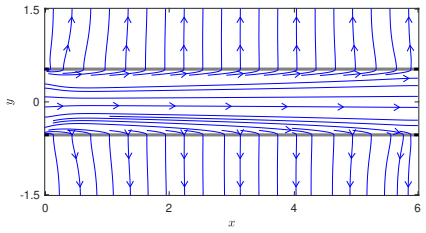


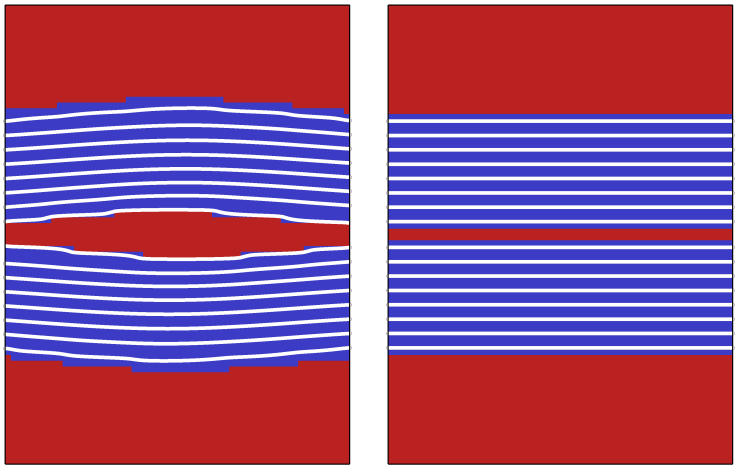
Figure 2.33: The flow solution using the NSDP equations for the problem in Figure 2.32 where $q = 1$ was used. The illustration shows flow lines at the inlet/outlet and top/bottom pressure boundaries, in gray the porous walls are shown.

In Table 2.18 the errors computed using this problem setup are presented. Major flow leakage is observed for the NSDP equations when $q = 1$, as illustrated in Figure 2.33 and by the error $v_l = 36.6\%$. However, flow leakage is significantly reduced to 3.93% using $q = 2$. Furthermore, as expected in Equation 2.114, flow leakage is reduced by a factor $\underline{\alpha}$ using the VANS equations (from $v_l = 36.6\%$ to $v_l = 4.56\%$). Another notable difference in error between the NSDP and VANS equations is the difference in $e^{\Delta p}_x$, which has two causes. Firstly, in section 2.3.1 it was observed that for flow parallel to a wall the interpretation of the solid/fluid interface might be off by $h/2$. The VANS equations correct for this error via the second Brinkman correction while the NSDP equations do not. In the NSDP discretization, the upper and lower wall are thus shifted by $h/2$ and the channel has erroneous height $L + h = 2\tilde{r}$. As $\tilde{r} > r$ increased, we find an erroneous decreased pressure drop $p_{,x} = -2\mu u_{max}\tilde{r}^2$. Secondly, we notice that more flow leakage

v_l causes larger errors in pressure drop $e_x^{\Delta p}$. For larger v_l, more flow is lost through the lower and upper walls. Consequently, the parabolic flow profile in the pipe flattens which reduces viscous forces and pressure drop. Furthermore, as flow is moving to the right, more flow is lost through the upper and lower wall and the parabolic profile flattens even more. Consequently, pressure drop is erroneously reduced when moving to the right and we find $e_{5L}^{\Delta p} > e_{4L}^{\Delta p}$ in all examples. Errors in pressure distribution are thus correlated to flow leakage, and for the remainder of this work we will mainly focus on flow leakage as a representation of precision of the solution.

3

MODERATE REYNOLDS FLOW TOPOLOGY OPTIMIZATION



In this chapter, methods for density-based TO of flow problems with moderate Reynolds numbers are critically examined. While the Darcy penalization inhibits viscous dominated flow, the Forchheimer penalization is used to inhibit inertia dominated flow. A reliable parameter selection strategy and a continuation approach, which balance accuracy of the flow solution and optimization convergence, are derived.

This chapter is based on the publication in Computer Methods in Applied Mechanics and Engineering 443, 118027, Theulings et al. (2025).

Reducing parameter tuning in topology optimization of flow problems using a Darcy and Forchheimer penalization

Abstract *In density-based topology optimization of flow problems, flow in the solid domain is generally inhibited using a penalization approach. Setting an appropriate maximum magnitude for the penalization traditionally requires manual tuning to find an acceptable compromise between flow solution accuracy and design convergence. In this work, three penalization approaches are examined, the Darcy (D), the Darcy with Forchheimer (DF), and the newly proposed Darcy with filtered Forchheimer (DFF) approach. Parameter tuning is reduced by analytically deriving an appropriate penalization magnitude for accuracy of the flow solution. The state-of-the-art D and DF approaches are improved by developing the novel DFF approach, based on a spatial average of the velocity magnitude. In comparison, the parameter selection in the DFF approach is more reliable, as accuracy of the flow solution and objective convexity are more predictable. Moreover, a continuation approach on the maximum penalization magnitude is derived by numerical inspection of the convexity of the pressure drop response. Using two-dimensional optimization benchmarks, the DFF approach reliably finds accurate flow solutions and is less prone to converge to inferior local optima.*

3.1. INTRODUCTION

Designing flow structures is of importance for many engineering problems, see for example, the design of tesla-type turbine devices (Alonso & Silva, 2022), microfluidic mixers (Andreasen et al., 2009), or drag minimization and lift maximization (Kondoh et al., 2012). Such problems often involve moderate to high Reynolds flow, which is difficult to design for due to the highly nonlinear flow equations. A tool to design flow structures is density-based Topology Optimization (TO). In flow TO, we intend to find an optimal phase distribution to separate a design domain into distinct solid and fluid parts. This is commonly accomplished by introducing the Darcy penalization to inhibit flow in the solid domain (Borrvall & Petersson, 2003; Gersborg-Hansen et al., 2005; Olesen et al., 2006). High penalizations are present in the solid domain, low or no penalizations are used in the fluid domain. A continuous penalization interpolation is used between the solid and the fluid domain, such that gradient-based optimizers can be used. This approach can be seen as an optimization by penalty method (Bruns, 2007), where the zero velocity constraint in the solid domain is enforced using a penalty term. In this approach, a proper selection of the magnitude of the penalization is crucial. It is well known that the penalization should be high enough to sufficiently inhibit flow in the solid domain, but small enough to ensure numerical stability. Often, finding the correct penalty requires manual tuning (Kreissl & Maute, 2012), which is time consuming and requires experience with TO. In this work we focus on the flow penalization in TO of

moderate Reynolds flow problems, and aim to improve parameter robustness and algorithmic stability.

The most common approach to select an appropriate magnitude for the Darcy penalization is presented by Olesen et al. (2006). The maximum penalization is determined based on a Darcy number and a characteristic length L. As a characteristic length the inlet diameter is often used, and in porous flow modeling, the Darcy number relates viscous and porous friction forces. However, parameter tuning remains necessary to select a Darcy number, such that flow is sufficiently reduced in the solid domain. On top of a user-defined Darcy number, Kondoh et al. (2012) include the Reynolds number to select the maximum penalty magnitude, and recent work confirms that the relation between flow reduction and penalization depends on the Reynolds number for inertia dominated flows (Alexandersen, 2023). However, complex geometries with locally varying flow velocities, length scales, and Reynolds numbers may be found using TO. A Darcy penalization based on a *single* estimation of the Reynolds number is thus unable to appropriately penalize flow in all parts of a design.

As the Darcy penalization is linearly dependent on the velocity magnitude, including the Reynolds number in the penalization results in a penalization which depends quadratically on velocity magnitude. A direct way to include a quadratic dependency on the flow magnitude in the penalty is through the Forchheimer penalization (Whitaker, 1996). Alonso and Silva (2022) find improved designs when using the Forchheimer penalization in addition to the Darcy penalization. However, selecting the appropriate magnitude for the Forchheimer penalization also requires manual tuning. Often, the maximum penalization magnitude is selected based on the physical interpretation of the Forchheimer term as a friction term for flow through porous media (X. Li et al., 2024; Philippi & Jin, 2015; Tian et al., 2024). So far, no critical analysis on the relation between Forchheimer penalization and accuracy of the flow solution has been performed.

Flow solutions are obtained using discretization. Bruns (2007) selects a penalty which is a couple of magnitudes larger than the largest diagonal stiffness matrix value, which suggests a relation between discretization and penalization magnitude. Recent work by Theulings et al. (2023) and by Abdelhamid and Czekanski (2023) suggests that the penalization should depend on the mesh size. This is supported by Jensen (2018), who finds increasingly thin features using mesh adaptation for increasingly high penalization magnitudes.

Another problem generally associated with the penalty approach is the convergence to ill-performing local optima. Design convergence is influenced by the maximum penalization, the penalty interpolation, and the initial design. Borrvall and Petersson (2003) show that a linear interpolation of the Darcy penalization results in optimal discrete solid/fluid designs without intermediate gray areas for Stokes flow problems. However, to escape ill-performing local optima, a penalization interpolation function which lowers the penalty for intermediate *gray* design variables is used in the earlier design iterations. Gersborg-Hansen et al. (2005) note that a lower penalization for gray design variables increases the convexity of the objective response, but also increases the amount of gray in the optimal design. A continuation approach which starts with a low penalization for gray design variables to escape ill-performing local optima, and ends with a higher penalization for gray design variables is recommended. Olesen et al. (2006) im-

prove the convergence behavior of the design by first optimizing using a low maximum flow penalization, allowing the optimizer to escape ill-performing local optima but finding designs with inaccurate flow solutions. Subsequently, the design is further optimized using a higher maximum penalization, resulting in designs with accurate flow solutions. Moreover, initial designs, dominated by flow penalization, are found to show a larger tendency to converge to ill-performing local optima. The relation between the choice of interpolation function and the objective behavior is analyzed on a simplified problem where a solid/fluid boundary is slightly modified for a fluid structure interaction problem in (Lundgaard et al., 2018). Results suggest that the objective should respond monotonically to design updates for the problem to be well-posed. Besides the convergence of the design, the convergence of the flow solution should be taken into account. While using a high flow penalization increases the accuracy of the flow solution, it can affect the stability of the flow solver (Kreissl et al., 2011). The penalization approach should thus be included in the stabilization approach of the flow solver (Alexandersen et al., 2014).

In this work, we aim to formulate a reliable approach for moderate Reynolds flow TO, which exhibits four desirable traits:

1. Parameter tuning is reduced.

2. The flow solution in the optimized design is accurate.

3. The optimization procedure does not show a tendency to converge to inferior local optima.

4. The flow solver remains stable over changing designs during the optimization process.

To reduce parameter tuning, we closely inspect three different penalization approaches in Section 3.2, and derive appropriate parameters for the magnitude of the flow penalization. We inspect the common approach using the Darcy penalization, and introduce two new approaches which additionally include the Forchheimer penalization. In Section 3.3, we discuss the implementation and the stabilization approach for the flow solution. Numerical analyses are performed using the finite element method implemented in COMSOL Multiphysics® v.6.1. (n.d.), as the necessary capabilities are readily available and as the use of a commercial software will further promote the use of presented techniques outside academia. In Section 3.4, we investigate the accuracy of the flow solution and the robustness of our parameter definition for varying Reynolds numbers. Subsequently, in Section 3.4.2, we inspect the convexity of the pressure drop objective, and derive a novel continuation strategy. In Section 3.5, the different approaches are compared in terms of accuracy of the optimized solution and of tendency to converge to inferior local optima for two different TO problems. We evaluate our novel method with respect to the proposed four traits identified for a reliable approach in the discussion in Section 3.6 and conclude in Section 3.7.

3.2. PENALIZATION IN LAMINAR FLOW TOPOLOGY OPTIMIZATION

In this section, we introduce the incompressible Navier-Stokes equations used to model laminar flows. To make them suitable for density-based TO and represent fluid and solid, we add penalization terms. Several approaches including Darcy and/or Forchheimer penalizations are explored. To derive an appropriate penalization magnitude, we use a dimensional analysis similar to Theulings et al. (2023). In Section 3.2.2, the novel approach to the dimensional analysis will lead to the same results found by Theulings et al. (2023), confirming its validity. In Section 3.2.3, we introduce the Forchheimer penalization similar to Alonso and Silva (2022) and use our method to derive novel settings for the penalization magnitudes of this term. Finally, in Section 3.2.4, we introduce a novel approach based on the Forchheimer penalization and a filtered velocity magnitude and derive the associated appropriate penalization magnitudes.

3.2.1. INCOMPRESSIBLE NAVIER-STOKES EQUATIONS FOR DENSITY-BASED TO

For laminar flow problems, the steady-state incompressible Navier-Stokes equations consist of: i) the momentum equation, which represents the forces acting on an infinitesimal volume of fluid, and ii) the continuity equation, which ensures that no fluid mass is created or destroyed. They are given in residual form as:

$$\begin{aligned} \boldsymbol{R}_{\mathbf{v}}(\mathbf{v},p) &= -\rho\nabla\mathbf{v}\cdot\mathbf{v} - \nabla p + \nabla\cdot\left(\mu\left(\nabla\mathbf{v}+\nabla\mathbf{v}^{\top}\right)\right) = \mathbf{0}, \\ R_p(\mathbf{v}) &= \nabla\cdot\mathbf{v} = 0, \end{aligned} \tag{3.1}$$

where $\mathbf{v}^{\top} = [u, v]$ is the velocity vector with u and v, the velocities in x and y direction, p the pressure field, ρ the fluid density, and μ the dynamic viscosity. In the momentum equation, we find the inertial force $\rho\nabla\mathbf{v}\cdot\mathbf{v}$, the pressure force $-\nabla p$, and the viscous force $\nabla\cdot\left(\mu\left(\nabla\mathbf{v}+\nabla\mathbf{v}^{\top}\right)\right)$.

For density-based TO, a design variable α, representing the fluid volume fraction, is used to distinguish between solid and fluid parts of the design domain. We continuously interpolate between the solid domain, $\alpha = 0$, and the fluid domain, $\alpha = 1$. In the solid domain, we aim to inhibit the flow by counteracting the forces in the momentum equation. The most common approach to adapt the Navier-Stokes equations to density-based TO is through the Darcy penalization (Borrvall & Petersson, 2003; Gersborg-Hansen et al., 2005; Olesen et al., 2006), here referred to as the D approach, which adds a force proportional to and in the opposite direction of the velocity to the momentum equation:

$$\boldsymbol{R}_{\mathbf{v}}(\mathbf{v},p) = -\rho\nabla\mathbf{v}\cdot\mathbf{v} - \nabla p + \nabla\cdot\left(\mu\left(\nabla\mathbf{v}+\nabla\mathbf{v}^{\top}\right)\right) \overbrace{-D_1(\alpha)\mathbf{v}}^{\text{Darcy penalization}} = \mathbf{0}. \tag{3.2}$$

where $D_1(\alpha)$ is a design dependent interpolation which inhibits flow in the solid domain using a high penalization $D_1(\alpha = 0) = \overline{D}_1$. The flow is governed by the standard momentum equation and $D_1(\alpha = 1) = 0$ in the fluid domain. The challenge in this adaptation lies in the selection of an appropriate maximum value $\overline{D}_1$ and of an adequate interpolation function $D_1(\alpha)$. The maximum penalization $\overline{D}_1$ should be large enough to sufficiently

penalize the flow in the solid domain. However, it should not be chosen too large, as such a choice would lead to a deterioration of the convergence of the forward solution due to ill-conditioning of the system equations, as well as a premature convergence to ill-performing local optima.

To define an appropriate magnitude for the penalization, we follow a similar approach to our earlier work (Theulings et al., 2023) and investigate the local flow reduction. Flow reduction is defined as the ratio v^s/v^f between v^s and v^f, the flow magnitudes in neighboring solid and fluid domains, respectively. We intend to define the maximum penalization $\overline{D}_1$ such that we can accurately predict $v^s/v^f = 10^{-q}$, where the user-defined parameter q indicates the order of magnitude by which the velocity in the solid and fluid domain differ. Subsequently, we can select the parameter q and the interpolation function $D_1(\alpha)$ to achieve a compromise between accuracy and convergence.

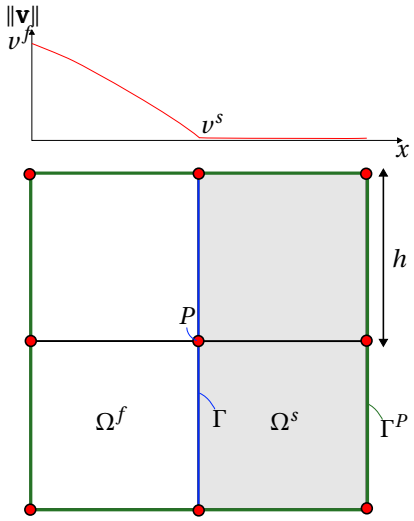


Figure 3.1: A four element domain Ω^P, with two fluid elements ($\alpha = 1$) to the left in Ω^f (white) and two solid elements ($\alpha = 0$) to the right in Ω^s (gray), such that $\Omega^P = \Omega^f \cup \Omega^s$ with boundary $\Gamma^P = \overline{\Omega^P} \setminus \Omega^P$. At the solid/fluid interface $\Gamma = \Omega^f \cap \Omega^s$, node P is selected for the analysis of flow leakage. On top the behavior of the flow magnitude over the elements is illustrated. In the fluid domain Ω^f we find a flow magnitude $\|\mathbf{v}\| = v^f$ which decreases towards $\|\mathbf{v}\| = v^s$ at the solid/fluid interface Γ after which it stagnates at a magnitude v^s in the solid domain Ω^s. We assume a regular mesh with elements of size h.

The flow reduction should be appropriate in the smallest scale represented by our model, i.e., the expected flow reduction is achieved in the thinnest design features. In this work, we use the finite element method to discretize the Navier-Stokes equations. We derive penalization magnitudes in the context of this discretization, although the proposed derivation is not limited to the finite element method and can be applied for other weighted residual methods, such as the finite volume method. To estimate the flow reduction, we consider a small part of the design domain Ω^P, discretized with four square elements of size h, as shown in Figure 3.1. We define a fluid domain Ω^f, made of two fluid elements, and a solid domain Ω^s, made of two solid elements. The interface

between fluid and solid is defined as $\Gamma = \Omega^f \cap \Omega^s$. We aim to define a penalization such that $v^s / v^f = 10^{-q}$, where the velocity magnitude v^f represents the maximum magnitude in fluid elements neighboring solid elements with a velocity magnitude v^s.

The discretized residual is used to compare the velocity magnitudes v^s and v^f. From the discretization in Figure 3.1, we select the center Node P on the interface to which test function $\phi^P(\boldsymbol{x})$ is attached, with $\boldsymbol{x}^\top = [x, y]$ the spatial coordinate vector. In the finite element discretization, the test function spans only the four elements in $\Omega^P = \Omega^f \cup \Omega^s$ which are attached to Node P and are zero outside of Ω^P and on the boundary $\Gamma^P = \overline{\Omega^P} \setminus \Omega^P$. The weighted residuals attached to Node P in domain Ω^P, are subsequently defined using the discretized velocity and pressure fields $\mathbf{v}^h(\boldsymbol{x})$ and $p^h(\boldsymbol{x})$:

$$\begin{bmatrix} R_u^P \\ R_v^P \end{bmatrix} = \int_{\Omega^P} \phi^P \boldsymbol{R}_\mathbf{v}(\mathbf{v}^h, p^h) d\Omega = \int_{\Omega^f} \phi^P \boldsymbol{R}_\mathbf{v}(\mathbf{v}^h, p^h) d\Omega + \int_{\Omega^s} \phi^P \boldsymbol{R}_\mathbf{v}(\mathbf{v}^h, p^h) d\Omega = \mathbf{0}, \tag{3.3}$$

where we defined the discretized residuals R_u^P and R_v^P, in x and y direction respectively, using the same scalar test function ϕ^P. In the finite element method, weighted residuals of the momentum equation are commonly defined per element and subsequently assembled in the complete residual $\boldsymbol{R}_\mathbf{v}^h = \mathbf{0}$, discretized on the N_d nodes in the mesh. For this analysis, we investigate all contributions to Node P at once, associated with the complete residual as $\boldsymbol{R}_\mathbf{v}^{h\top} = [R_u^1, R_v^1, ..., R_u^P, R_v^P, ..., R_u^{N_d}, R_v^{N_d}] = \mathbf{0}$.

Equation 3.3 presents the weighted residual of the momentum equation at Node P in the solid/fluid domain Ω^P. For the discretized residual to be zero, we require the discretized solid and fluid domain terms to be in equilibrium. Using Equation 3.3, we will be able to compare the velocity magnitudes v^f and v^s in the neighboring fluid and solid elements.

We use the fact that for Equation 3.3 to hold, the fluid and solid domain terms on the right-hand side are equal in magnitude and opposite in direction:

$$\left| \int_{\Omega^s} \phi^P \boldsymbol{R}_\mathbf{v}(\mathbf{v}^h, p^h) d\Omega \right|_2 = \left| \int_{\Omega^f} \phi^P \boldsymbol{R}_\mathbf{v}(\mathbf{v}^h, p^h) d\Omega \right|_2, \tag{3.4}$$

where $|\square|_2$ is the L^2-norm. We proceed to approximate the terms in Equation 3.4 using approximate fluid and solid domain velocity magnitudes v^f and v^s. First, we simplify the required analysis. If $\left|\boldsymbol{R}_\mathbf{v}(\mathbf{v}^h, p^h)\right|_2 = C$ is constant, the left- and right-hand side in Equation 3.4 become $\left|\int_{\Omega^s} \phi^P d\Omega\right|_2 C$ and $\left|\int_{\Omega^f} \phi^P d\Omega\right|_2 C$, respectively. We use the test function ϕ^P to inspect the residual at Node P, but do not want to approximate its value and aim for the derivation to be general and independent on a specific test function. We thus normalize Equation 3.4 as:

$$\frac{\left|\int_{\Omega^s} \phi^P \boldsymbol{R}_\mathbf{v}(\mathbf{v}^h, p^h) d\Omega\right|_2}{\left|\int_{\Omega^s} \phi^P d\Omega\right|_2} = \frac{\left|\int_{\Omega^f} \phi^P \boldsymbol{R}(\mathbf{v}^h, p^h)_\mathbf{v} d\Omega\right|_2}{\left|\int_{\Omega^f} \phi^P d\Omega\right|_2}. \tag{3.5}$$

We note that due to the symmetry of ϕ^P over Γ and the structured square mesh, $\left|\int_{\Omega^s} \phi^P d\Omega\right|_2 = \left|\int_{\Omega^f} \phi^P d\Omega\right|_2$. If a different unstructured mesh is used, this normalization does not hold and the difference in element size and shape in the fluid and solid domains has to be taken into account. Subsequently, we define a notation for approximating the

orders of magnitude of the residuals as:

$$\mathcal{O}^s\left(\boldsymbol{R}_{\mathbf{v}}\right) \approx \frac{\left|\int_{\Omega^s} \phi^P \boldsymbol{R}_{\mathbf{v}}(\mathbf{v}^h, p^h) d\Omega\right|_2}{\left|\int_{\Omega^s} \phi^P d\Omega\right|_2}, \qquad \mathcal{O}^f\left(\boldsymbol{R}_{\mathbf{v}}\right) \approx \frac{\left|\int_{\Omega^f} \phi^P \boldsymbol{R}_{\mathbf{v}}(\mathbf{v}^h, p^h) d\Omega\right|_2}{\left|\int_{\Omega^f} \phi^P d\Omega\right|_2}, \tag{3.6}$$

which can be approximated and compared as $\mathcal{O}^f\left(\boldsymbol{R}_{\mathbf{v}}\right) \approx \mathcal{O}^s\left(\boldsymbol{R}_{\mathbf{v}}\right)$.

In this paper, we use the following rules for computing approximate orders of magnitude $\mathcal{O}^s(\bullet)$ and $\mathcal{O}^f(\bullet)$. For an arbitrary vector field $\boldsymbol{\Psi}(\boldsymbol{x})$ in Ω^f and Ω^s, we define:

$$\mathcal{O}^s\left(\boldsymbol{\Psi}\right) = \Psi^s, \qquad \mathcal{O}^f\left(\boldsymbol{\Psi}\right) = \Psi^f, \tag{3.7}$$

where we approximate $\Psi^s \approx |\boldsymbol{\Psi}|_2 \in \Omega^s$ and $\Psi^f \approx |\boldsymbol{\Psi}|_2 \in \Omega^f$. The largest gradient is assumed to be dependent on the element size h such that:

$$\mathcal{O}^s\left(\nabla\boldsymbol{\Psi}\right) \approx \Delta\Psi^s / h, \qquad \mathcal{O}^f\left(\nabla\boldsymbol{\Psi}\right) \approx \Delta\Psi^f / h, \tag{3.8}$$

where $\Delta\Psi^s$, $\Delta\Psi^f$ are estimates of the *maximum* change of $|\boldsymbol{\Psi}|_2$ in Ω^s and Ω^f, respectively. We emphasize that this assumption on the magnitude of gradients is essential, and can only be made because we investigate the discretized residual in Equation 3.3. Using an analysis of the residual in Equation 3.2, gradients would be related to an overall length scale L related to the design, which would lead to the common approach relating the penalization magnitude to a Darcy number Da and L (Olesen et al., 2006), which has limitations as shown by Theulings et al., 2023. Finally, we assume that only one term in either the fluid and solid domain is dominant:

$$\mathcal{O}^s\left(\boldsymbol{A}+\boldsymbol{B}\right) \approx \max\left(\mathcal{O}^s\left(\boldsymbol{A}\right), \mathcal{O}^s\left(\boldsymbol{B}\right)\right), \qquad \mathcal{O}^f\left(\boldsymbol{A}+\boldsymbol{B}\right) \approx \max\left(\mathcal{O}^f\left(\boldsymbol{A}\right), \mathcal{O}^f\left(\boldsymbol{B}\right)\right). \tag{3.9}$$

To summarize, we use an estimation of $\mathcal{O}^f\left(\boldsymbol{R}_{\mathbf{v}}\right) \approx \mathcal{O}^s\left(\boldsymbol{R}_{\mathbf{v}}\right)$ to derive an approximation for v^s / v^f. This is done by examining each term in $\boldsymbol{R}_{\mathbf{v}}(\mathbf{v}^h, p^h)$, as presented in Equation 3.2, individually. Subsequently, the approximation is used to define a penalization such that we achieve a desired flow reduction $v^s / v^f = 10^{-q}$.

3.2.2. THE DARCY APPROACH

First, we examine how to appropriately use the D approach and study its limitations. We start by investigating the magnitude of the terms in the fluid domain Ω^f. The magnitude of the velocity gradient is approximated using the plot in Figure 3.1. We find a velocity magnitude of $\left|\mathbf{v}^h\right|_2 = v^f$ on the left edge of Ω^f and of $\left|\mathbf{v}^h\right|_2 = v^s \ll v^f$ on the right edge of Ω^f at the solid/fluid interface Γ. In Ω^f we assume that any change in velocity magnitude between neighboring nodes in the fluid domain is lower than between fluid and neighboring solid/fluid interface nodes. Consequenlty, the maximum change in velocity magnitude is approximated as $\Delta v^f = v^f - v^s \approx v^f$. As square elements are used, the maximum velocity gradient is found in the direction normal to the fluid/solid interface Γ, and is approximated as $\mathcal{O}^f\left(\nabla\mathbf{v}^h\right) \approx \Delta v^f / h \approx v^f / h$. We note that for distorted elements

with high aspect ratios, this assumption may not hold. The magnitude of the inertial and viscous terms are consequently approximated as:

$$\mathscr{O}^f\Big(-\rho\nabla\mathbf{v}^h\cdot\mathbf{v}^h+\nabla\cdot\Big(\mu\Big(\nabla\mathbf{v}^h+\nabla\mathbf{v}^{h\top}\Big)\Big)\Big)\approx\max\left(\rho\frac{v^{f\,2}}{h},\mu\frac{v^f}{h^2}\right). \tag{3.10}$$

Subsequently, the magnitude of the pressure term is approximated as:

$$\mathscr{O}^f\Big(\nabla p^h\Big)\approx\frac{\Delta p^f}{h}, \tag{3.11}$$

where Δp^f is an estimate of the maximum change in pressure in Ω^f. In the fluid domain the magnitude of the Darcy penalization is zero, $D_1(\alpha=1)=0$. Finally, the magnitude of the terms in the fluid domain can be estimated as:

$$\mathscr{O}^f\Big(\boldsymbol{R}_\mathbf{v}\Big)\approx\max\left(\rho\frac{v^{f\,2}}{h},\,\mu\frac{v^f}{h^2},\,\frac{\Delta p^f}{h}\right). \tag{3.12}$$

The magnitude of the velocity gradient in the solid domain is estimated using the change in velocity Δv^s as $\mathscr{O}^s\Big(\nabla\mathbf{v}^h\Big)\approx\Delta v^s/h$, which is used to approximate the inertial and viscous terms:

$$\mathscr{O}^s\Big(-\rho\nabla\mathbf{v}^h\cdot\mathbf{v}^h+\nabla\cdot\Big(\mu\Big(\nabla\mathbf{v}^h+\nabla\mathbf{v}^{h\top}\Big)\Big)\Big)\approx\max\left(\rho\frac{v^s\Delta v^s}{h},\mu\frac{\Delta v^s}{h^2}\right). \tag{3.13}$$

Similar to the fluid domain, we approximate the pressure term as:

$$\mathscr{O}^s\Big(\nabla p^h\Big)\approx\frac{\Delta p^s}{h}, \tag{3.14}$$

In the solid domain, the magnitude of the Darcy penalization is maximal, $D_1(\alpha=0)=\overline{D}_1$, and is estimated as:

$$\mathscr{O}^s\Big(D_1(\alpha)\mathbf{v}^h\Big)\approx\overline{D}_1 v^s. \tag{3.15}$$

Gathering terms from Equations 3.13, 3.14, 3.15, the magnitude of the terms in the solid domain is estimated as:

$$\mathscr{O}^s\Big(\boldsymbol{R}_\mathbf{v}\Big)\approx\max\left(\rho\frac{v^s\Delta v^s}{h},\frac{\Delta p^s}{h},\mu\frac{\Delta v^s}{h^2},\overline{D}_1 v^s\right). \tag{3.16}$$

As we assume that the terms in the fluid and solid domain are in equilibrium $\mathscr{O}^f\Big(\boldsymbol{R}_\mathbf{v}\Big)\approx\mathscr{O}^s\Big(\boldsymbol{R}_\mathbf{v}\Big)$:

$$\max\left(\rho\frac{v^{f\,2}}{h},\frac{\Delta p^f}{h},\mu\frac{v^f}{h^2}\right)\approx\max\left(\rho\frac{v^s\Delta v^s}{h},\frac{\Delta p^s}{h},\mu\frac{\Delta v^s}{h^2},\overline{D}_1 v^s\right). \tag{3.17}$$

Our approach is to investigate each term on the left-hand side assuming it is dominant. Subsequently, we investigate how the right-hand side counteracts these terms and how the velocity magnitudes are related. A first working assumption is that if an appropriate

penalization is applied then $v^f \gg v^s > \Delta v^s$. Since the inertial and viscous terms scale the same with respect to ρ, μ and h in the fluid and solid domains, we neglect the solid domain inertial and viscous terms as they cannot counteract the fluid domain terms. Reducing Equation 3.17 to:

$$\max\left(\rho\frac{{v^f}^2}{h}, \frac{\Delta p^f}{h}, \mu\frac{v^f}{h^2}\right) \approx \max\left(\frac{\Delta p^s}{h}, \overline{D}_1 v^s\right). \tag{3.18}$$

Moreover, in (Theulings et al., 2023) we derived an appropriate penalization based on the assumption that the pressure field is C^1-continuous and $\mathscr{O}^s\left(\nabla p^h\right) \approx \mathscr{O}^f\left(\nabla p^h\right)$. In (C. Wu & Zhang, 2024) and from experience, we find this assumption to hold across a solid/fluid interface. Assuming that if dominant, the pressure gradients are in equilibrium, we neglect them in the order analysis:

$$\max\left(\rho\frac{{v^f}^2}{h}, \mu\frac{v^f}{h^2}\right) \approx \overline{D}_1 v^s. \tag{3.19}$$

In Equation 3.19, two terms might be dominant in the fluid domain. We need to determine which one to select an appropriate maximum penalization $\overline{D}_1$. Following (Theulings et al., 2023), we select the dominant term based on the elemental Reynolds number, the ratio between elemental inertial and viscous term magnitudes:

$$Re_e^f = \frac{\rho\frac{{v^f}^2}{h}}{\mu\frac{v^f}{h^2}} = \frac{\rho v^f h}{\mu}. \tag{3.20}$$

The fluid domain velocity magnitude v^f varies throughout the design domain and is not known when selecting $\overline{D}_1$. An estimation of the magnitude $\tilde{v}^f \approx v^f$ is thus needed to evaluate the elemental Reynolds number:

$$\widetilde{Re}_e^f \approx \frac{\rho \tilde{v}^f h}{\mu} \tag{3.21}$$

Generally, we estimate $\tilde{v}^f$ using the maximum velocity magnitude at the inlet or outlet. For low elemental Reynolds number, $\widetilde{Re}_e^f \leq 1$, viscous terms are dominant and Equation 3.19 reduces to $\mu v^f / h^2 \approx \overline{D}_1 v^s$, such that we can compute the flow reduction as:

$$\frac{v^s}{v^f} \approx \frac{\mu}{h^2\overline{D}_1} \approx 10^{-q}. \tag{3.22}$$

Subsequently, we find a maximum penalization for the desired flow reduction:

$$\overline{D}_1 = 10^q \frac{\mu}{h^2}. \tag{3.23}$$

To summarize, we first used the fact that $\mathscr{O}^f\left(\boldsymbol{R}_\mathbf{v}\right) \approx \mathscr{O}^s\left(\boldsymbol{R}_\mathbf{v}\right)$ to find an expression for the ratio of velocity magnitudes v^s / v^f. Subsequently, $\overline{D}_1$ is defined to ensure that this ratio

approximately takes a desired value, i.e., $v^s/v^f \approx 10^{-q}$. A similar method will be used to define all other penalization magnitudes.

For low elemental Reynolds numbers, $\widetilde{Re}_e^f \leq 1$, the definition of the maximum Darcy penalization is straightforward. However, for larger elemental Reynolds numbers, $\widetilde{Re}_e^f > 1$, the inertial term is dominant and Equation 3.19 reduces to $\rho {v^f}^2/h \approx \overline{D}_1 v^s$, such that flow reduction is computed as:

$$\frac{v^s}{v^f} \approx \frac{\rho v^f}{h\overline{D}_1} \approx 10^{-q}, \tag{3.24}$$

which holds for:

$$\overline{D}_1 = 10^q \frac{\rho v^f}{h} \approx 10^q \frac{\rho \tilde{v}^f}{h} = 10^q \frac{\mu}{h^2} \widetilde{Re}_e^f. \tag{3.25}$$

The estimation of the velocity magnitude $\tilde{v}^f$ influences the selection of $\overline{D}_1$ in two ways: first in selecting whether inertial or viscous terms are dominant in $\widetilde{Re}_e^f$, and second when inertial terms are dominant in selecting the penalization $\overline{D}_1$. An inaccurate estimation of the velocity $\tilde{v}^f$ influences the flow reduction, as shown by Theulings et al., 2023. Therefore, in this work, we aim to construct a penalization that does not require an *a priori* estimation of the velocity.

3.2.3. THE DARCY WITH FORCHHEIMER APPROACH

To circumvent the estimation of the velocity magnitude $\tilde{v}^f$, we add the Forchheimer penalization (Whitaker, 1996) to the momentum equation, resulting in the Darcy with Forchheimer (DF) approach. The Forchheimer penalization depends quadratically on the velocity and is introduced beside the Darcy penalization as:

$$\boldsymbol{R}_{\mathbf{v}}(\mathbf{v},p) = -\rho\nabla\mathbf{v}\cdot\mathbf{v} - \nabla p + \nabla\cdot\left(\mu\left(\nabla\mathbf{v}+\nabla\mathbf{v}^\top\right)\right) \overbrace{-D_2(\alpha)\mathbf{v}}^{\text{Darcy penalization}} \underbrace{-F_2(\alpha)\left|\mathbf{v}\right|_2\mathbf{v}}_{\text{Forchheimer penalization}} = \mathbf{0}. \tag{3.26}$$

Both penalizations are set to zero $D_2(\alpha=1) = F_2(\alpha=1) = 0$ in the fluid domain and to their maximum value $D_2(\alpha=0) = \overline{D}_2$, $F_2(\alpha=0) = \overline{F}_2$ in the solid domain. To add the Forchheimer term to the order analysis performed in Section 3.2.2, we estimate its magnitude in the solid domain as:

$$\mathscr{O}^s\left(F_2(\alpha)\left|\mathbf{v}^h\right|_2\mathbf{v}^h\right) \approx \overline{F}_2\left(v^s\right)^2, \tag{3.27}$$

whereas its magnitude is set to zero in the fluid domain. Subsequently, we introduce the magnitude of the Forchheimer penalization to the right-hand side of Equation 3.19:

$$\max\left(\rho\frac{{v^f}^2}{h}, \mu\frac{v^f}{h^2}\right) \approx \max\left(\overline{D}_2 v^s, \overline{F}_2\left(v^s\right)^2\right). \tag{3.28}$$

We aim for the Forchheimer penalization to counteract inertial terms, as both scale quadratically with velocity, and for the Darcy penalization to counteract viscous terms,

as both scale linearly with velocity. Neglecting the viscous ($Re_e^f \geq 1$) or inertial ($Re_e^f < 1$) terms, we find:

$$Re_e^f \geq 1 : \rho \frac{{v^f}^2}{h} \approx \overline{F}_2 \left(v^s\right)^2, \tag{3.29}$$

$$Re_e^f < 1 : \mu \frac{v^f}{h^2} \approx \overline{D}_2 v^s, \tag{3.30}$$

from which we derive the desired flow reduction and consequent maximum penalization magnitude as:

$$Re_e^f \geq 1 : \frac{v^s}{v^f} \approx \sqrt{\frac{\rho}{h\overline{F}_2}} \approx 10^{-q}, \qquad \overline{F}_2 = 10^{2q} \frac{\rho}{h}, \tag{3.31}$$

$$Re_e^f < 1 : \frac{v^s}{v^f} \approx \frac{\mu}{h^2 \overline{D}_2} \approx 10^{-q}, \qquad \overline{D}_2 = 10^{q} \frac{\mu}{h^2}. \tag{3.32}$$

We find that the Forchheimer magnitude scales similarly to the Darcy magnitude in the D approach when inertial terms are dominant $\widetilde{Re}_e^f \geq 1$ in Equation 3.25, but with a factor 10^{2q} instead of 10^q. Moreover, we emphasize that, contrary to the D approach, both terms in Equations 3.31 and 3.32 do not require an *a priori* velocity estimation.

3.2.4. THE DARCY WITH FILTERED FORCHHEIMER APPROACH

The DF approach solves the problem of estimating $\tilde{v}^f$, but we find inconsistencies when performing numerical analysis in the assumptions made in the derivation of $\overline{F}_2$ and $\overline{D}_2$. We assumed that when inertial (resp. viscous) terms are dominant in the fluid, the Forchheimer (resp. Darcy) term is dominant in the solid. However, this assumption might not hold. To compare dominant terms, we define a solid domain elemental Reynolds number Re_e^s by inspecting the definition of Re_e^f. The fluid domain element Reynolds number in Equation 3.20 was defined as the ratio between inertial and viscous term magnitudes, which is equivalent to dividing the Equation 3.29 by Equation 3.30:

$$Re_e^f = \frac{\rho v^f h}{\mu} \approx \frac{\overline{F}_2 v^s}{\overline{D}_2}. \tag{3.33}$$

Subsequently, we define the solid domain elemental Reynolds number by substituting the maximum values for the Forchheimer/Darcy penalization in Equation 3.33:

$$Re_e^s = \frac{\overline{F}_2 v^s}{\overline{D}_2} = 10^q \frac{\rho v^s h}{\mu}, \tag{3.34}$$

Using this definition, it should hold that $Re_e^f \approx Re_e^s$, i.e., the elemental Reynolds number should be continuous as it is of similar magnitude in neighboring solid and fluid domains. In the ideal case when $v^s = v^f 10^{-q}$ this holds as $Re_e^s = 10^q(\rho v^s h)/\mu = (\rho v^f h)/\mu$. However, it might happen that $Re_e^s \geq 1$, while $Re_e^f < 1$, or vice versa. This mainly occurs during the convergence of the state solution when we have not yet converged to

a solution where $v^s/v^f = 10^{-q}$. When this occurs, Re_e^s and penalizations may abruptly change, resulting in unstable convergence behavior. Consequently, in our numerical analysis in Section 3.4, we find the DF approach to have less predictable solutions and a less predictable objective convexity. Moreover, in Appendix 3.C jumps in the elemental Reynolds number are shown to lead to convergence problems of the forward solution. An alternative penalization approach, which ensures a more continuous elemental Reynolds number, is needed.

To solve this issue, we aim to define penalizations such that $Re_e^s = Re_e^f = \rho v^f h/\mu$, and Re_e^s should thus be dependent on the fluid domain velocities v^f. However, no field containing information on fluid domain velocities is present in the solid domain. To pull information from the fluid domain to the solid domain, we define the *filtered* velocity magnitude U by applying the PDE filter, as described for the design field by Lazarov and Sigmund (2011), to the velocity magnitude:

$$-R^2\nabla^2 U + U = |\mathbf{v}|_2\,. \tag{3.35}$$

Information about the local flow velocity is distributed over a domain with a radius of N elements using $R = Nh/(2\sqrt{3})$, such that we may estimate $\mathscr{O}^s\left(U^h\right) \approx v^f$. Although the filtered velocity magnitude U will have a significant magnitude in the solid domain, which does not appropriately represent the physics, it can be used to appropriately penalize and decrease the actual magnitude of $\mathbf{v} = [u, v]^\top$ in the solid domain. The filtered velocity magnitude U thus exists in conjunction with the actual velocity and its magnitude $|\mathbf{v}|_2$, i.e., no additional flow is introduced in the solid domain. We introduce a *filtered* Forchheimer penalization, resulting in the Darcy with filtered Forchheimer (DFF) approach:

$$\boldsymbol{R}_{\mathbf{v}}(\mathbf{v},p) = -\rho\nabla\mathbf{v}\cdot\mathbf{v} - \nabla p + \nabla\cdot\left(\mu\left(\nabla\mathbf{v}+\nabla\mathbf{v}^\top\right)\right) \overbrace{-D_3(\alpha)\mathbf{v}}^{\text{Darcy penalization}} \underbrace{-F_3(\alpha)U\mathbf{v}}_{\text{filtered Forchheimer penalization}} = \mathbf{0}. \tag{3.36}$$

The magnitude of the filtered Forchheimer penalization is estimated as $\mathscr{O}^s\left(F_3(\alpha)U^h\mathbf{v}^h\right) \approx \overline{F}_3 v^f v^s$, which is introduced in Equation 3.19, to find:

$$\max\left(\rho\frac{v^{f\,2}}{h}, \mu\frac{v^f}{h^2}\right) \approx \max\left(\overline{D}_3 v^s, \overline{F}_3 v^f v^s\right). \tag{3.37}$$

The Darcy penalization remains unchanged with respect to the DF approach and the maximum penalization $\overline{D}_3 = 10^q\mu/h^2$ found in Equation 3.32 is used. When $Re_e^f \geq 1$ and inertial terms are dominant, we aim for the filtered Forchheimer penalization to inhibit the flow. We find $\rho\frac{v^{f\,2}}{h} \approx \overline{F}_3 v^f v^s$, and a flow reduction with resulting maximum penalization:

$$\frac{v^s}{v^f} \approx \frac{\rho}{h\overline{F}_3} \approx 10^{-q}, \qquad \overline{F}_3 = 10^q\frac{\rho}{h}, \tag{3.38}$$

which is a factor 10^q lower than the maximum $\overline{F}_2$ for the DF approach in Equation 3.31. Moreover, computing the solid domain elemental Reynolds number using the filtered Forchheimer penalization, we find:

$$Re_e^s = \frac{\overline{F}_3 v^f v^s}{\overline{D}_3 v^s} = \frac{\rho v^f h}{\mu} = Re_e^f. \tag{3.39}$$

Using the filtered Forchheimer penalization, we ensure a continuous elemental Reynolds number and appropriate penalization magnitudes for viscous and inertial terms.

3.2.5. OVERVIEW OF PENALIZATION APPROACHES

Three approaches for penalizing the flow in the solid domain are discussed in this paper: the Darcy, the Darcy with Forchheimer, and the Darcy with filtered Forchheimer approach, defined respectively as:

$$\text{D:} -\rho\nabla\mathbf{v}\cdot\mathbf{v} - \nabla p + \nabla\cdot\left(\mu\left(\nabla\mathbf{v}+\nabla\mathbf{v}^\top\right)\right) - D_1(\alpha)\mathbf{v} = \mathbf{0} \tag{3.40}$$

$$\text{DF:} -\rho\nabla\mathbf{v}\cdot\mathbf{v} - \nabla p + \nabla\cdot\left(\mu\left(\nabla\mathbf{v}+\nabla\mathbf{v}^\top\right)\right) - D_2(\alpha)\mathbf{v} - F_2(\alpha)|\mathbf{v}|_2\mathbf{v} = \mathbf{0}, \tag{3.41}$$

$$\text{DFF:} -\rho\nabla\mathbf{v}\cdot\mathbf{v} - \nabla p + \nabla\cdot\left(\mu\left(\nabla\mathbf{v}+\nabla\mathbf{v}^\top\right)\right) - D_3(\alpha)\mathbf{v} - F_3(\alpha)U\mathbf{v} = \mathbf{0}. \tag{3.42}$$

Subsequently, maximum penalizations are defined such that the flow reduction at the fluid/solid interface can be approximated as $v^s/v^f = 10^{-q}$. For the D approach, an appropriate penalization depends on an estimate of the fluid velocity magnitude and consequent elemental Reynolds number in Equation 3.21. The maximum penalization values derived in previous sections can be found in Table 3.1.

	D: $\widetilde{Re}_e^f \leq 1$	D: $\widetilde{Re}_e^f > 1$	DF	DFF
Darcy	$\overline{D}_1 = 10^q \frac{\mu}{h^2}$	$\overline{D}_1 = 10^q \frac{\mu}{h^2}\widetilde{Re}_e^f$	$\overline{D}_2 = 10^q \frac{\mu}{h^2}$	$\overline{D}_3 = 10^q \frac{\mu}{h^2}$
Forchheimer	-	-	$\overline{F}_2 = 10^{2q}\frac{\rho}{h}$	$\overline{F}_3 = 10^q \frac{\rho}{h}$

Table 3.1: The derived appropriate settings for the maximum penalization in the solid domain at $\alpha = 0$ such that the flow reduction can be estimated as $v^s/v^f = 10^{-q}$.

3.2.6. INTERPOLATION FUNCTION AND POST-PROCESSING APPROACH

So far, this section has mainly focused on the flow reduction at crisp fluid/solid interfaces. However, since gradient-based optimization with continuous design variables is used, we interpolate the penalization $D(\alpha)$ and $F(\alpha)$ for $0 \leq \alpha \leq 1$, and fluid/solid interfaces generally exhibit intermediate values of α. In this work, we use the interpolation function presented by Borrvall and Petersson (2003) and shown in Figure 3.2:

$$D(\alpha) = \overline{D}\frac{\hat{p}(1-\alpha)}{\hat{p}+\alpha}, \tag{3.43}$$

where the parameter $\hat{p}$ was originally introduced in a two-step continuation approach to control the level of gray in the optimized designs. The layout is first optimized using

$\hat{p} = 0.01$ to make the response surface of the objective more convex and allow the optimizer to escape ill-performing local optima. Secondly, the resulting designs are further optimized using $\hat{p} = 0.1$ to obtain a discrete valued solution. In Section 3.4.2, the relation between the interpolation function and convexity of the pressure drop objective will be further investigated.

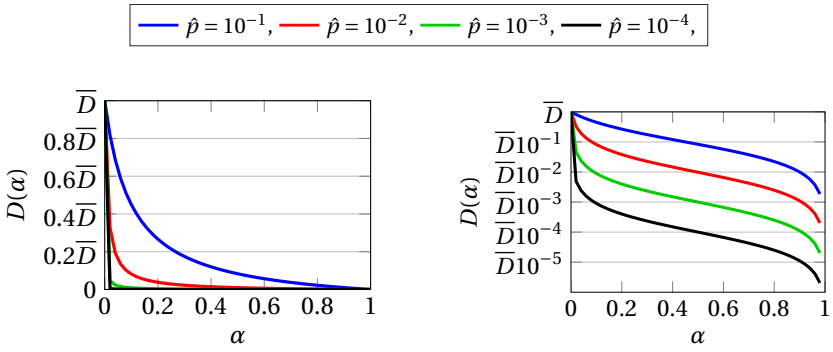


Figure 3.2: The interpolation function presented in Equation 3.2 in linear and logarithmic scales.

Our approach for finding appropriate interpolation functions is to examine a fluid/solid interface, where the solid domain in Figure 3.1 is replaced by a porous domain with $\alpha \approx 0.5$. We note that Gersborg-Hansen et al. (2005) already interpret the interpolation as a function which decreases the penalization in the gray areas to improve the convexity of the objective response. However, using our prediction of the flow reduction, a more reliable derivation of the required parameter $\hat{p}$ can be obtained. We investigate the penalization achieved for these interface areas by observing that the interpolation lowers the magnitude of the penalization determined by 10^q in Table 3.1, and consequently increases the predicted flow. In all approaches, the Darcy penalization is interpolated by using $\hat{p} = 10^{-\hat{q}}$ in Equation 3.43, as:

$$D(\alpha) = \overline{D}\frac{10^{-\hat{q}}(1-\alpha)}{10^{-\hat{q}}+\alpha}, \tag{3.44}$$

such that $D(\alpha = 0.5) \approx 10^{-\hat{q}}\overline{D}$, decreasing the Darcy penalization by $10^{-\hat{q}}$ as shown in the logarithmic plot in Figure 3.2, resulting in more porous domain flow as $v^s/v^f = 10^{\hat{q}-q}$.

For the Forchheimer penalization, a different interpolation function is used for the DF and the DFF approach where we substitute $\hat{p} = 10^{-2\hat{q}}$ and $\hat{p} = 10^{-\hat{q}}$, respectively:

$$F_2(\alpha) = \overline{F}_2\frac{10^{-2\hat{q}}(1-\alpha)}{10^{-2\hat{q}}+\alpha}, \qquad F_3(\alpha) = \overline{F}_3\frac{10^{-\hat{q}}(1-\alpha)}{10^{-\hat{q}}+\alpha}, \tag{3.45}$$

such that at the fluid/solid interface $F_2(\alpha = 0.5) \approx 10^{-2\hat{q}}\overline{F}_2$ and $F_3(\alpha = 0.5) \approx 10^{-\hat{q}}\overline{F}_3$, decreasing the penalization by $10^{-2\hat{q}}$ and $10^{-\hat{q}}$, respectively. In Table 3.1, the penalizations for the DF and DFF approach scale respectively with 10^{2q} and 10^q, both resulting in a flow reduction of $v^s/v^f = 10^{-q}$. The lower parameter $\hat{p} = 10^{-2\hat{q}}$ for the DF approach is thus needed to ensure that the flow in the porous domain scales similarly for the DF and DFF approaches as $v^s/v^f = 10^{\hat{q}-q}$.

Using the predicted flow reduction in porous areas, we derive an improved threshold on α for the post-processing of optimized results. Common approaches for post-processing define the solid domain as the areas where $\alpha < \alpha_t = 0.5$. However, using our prediction of the flow reduction, we define the solid domain as the domain where a specific flow reduction is achieved and derive a specific value α_t dependent on both q and $\hat{q}$. For a specific flow reduction of 10^{-r}, we require an interpolated Darcy magnitude of $D(\alpha_t) = \overline{D}10^{r-q}$, and the solid domain is defined as those areas where:

$$\frac{10^{-\hat{q}}(1-\alpha)}{10^{-\hat{q}}+\alpha} > 10^{r-q}, \tag{3.46}$$

which can be used to threshold the design using the interpolation function itself, or be rewritten as a threshold for α in the solid domain:

$$\alpha_t < \frac{1-10^{r-q}}{1+10^{r-q+\hat{q}}}. \tag{3.47}$$

The same threshold for α_t is found using the interpolation function in Equation 3.45 for the Forchheimer penalization in the DF approach. For the remainder of this work, we threshold the designs using $r = 1$ such that the solid domain is defined as those areas where $v^s / v^f < 0.1$.

3.3. NUMERICAL IMPLEMENTATION

For the analysis and optimization of the flow problems, we use COMSOL Multiphysics® v.6.1. (n.d.). Since the implementation is done by the multiphysics software, we take a birds-eye view of the model. However, it remains important to make informed choices of the settings within the software.

3.3.1. DISCRETIZATION AND OPTIMIZATION APPROACH

Shape functions, stabilization approaches, and solution procedures have to be specified for the flow field, pressure field, and filtered flow field. All fields are discretized using the finite element method on the same quadrilateral mesh with square elements of size h. For the flow field we use quadratic while for the pressure field we use linear interpolation functions. The filtered velocity field is handled similarly to the velocity field with quadratic interpolation functions. Streamline diffusion (Hauke, 2001; Hauke & Hughes, 1994) is used to stabilize the solution when convection is dominating the flow. In our changing topology, small islands or sharp corners may appear, therefore, to stabilize the solution around these features, additional diffusion is added by applying crosswind diffusion (Hauke & Hughes, 1994; Hughes & Mallet, 1986). We note that streamline and crosswind diffusion slightly lower the accuracy of the flow solution, although not significantly in comparison to errors related to flow leakage. However, the use of these stabilization terms promote a smooth convergence of the flow solution.

Finally, the design is represented using a constant volume fraction α in each element in the mesh, and no filter is applied on the design variables. Furthermore, we use the Method of Moving Asymptotes (MMA) approach with an optimality tolerance of 10^{-3} and a constraint penalty factor of 10^2. The relatively low penalty factor allows for more

design flexibility while satisfying the constraints within an acceptable tolerance. To compute sensitivities, the adjoint approach is used.

3.3.2. STABILIZATION OF THE STATE SOLUTION

To solve the state equations reliably for moderate Reynolds numbers, a pseudo time-stepping scheme is used (*CFD Module User's Guide v. 6.1.* 2022), and a transient problem is solved until steady state is reached:

$$\boldsymbol{R}_v = -\rho \frac{\mathbf{v}^n - \mathbf{v}^{n-1}}{\Delta \tilde{t}} - \rho \nabla \mathbf{v}^n \cdot \mathbf{v}^n - \nabla p^n + \nabla \cdot \left(\mu \left(\nabla \mathbf{v}^n + \nabla \mathbf{v}^{n\top} \right) \right) - \boldsymbol{f}(\mathbf{v}^n) = \mathbf{0}, \tag{3.48}$$

where the superscript n denotes the iteration in the pseudo time-stepping scheme and $\boldsymbol{f}$ represents the selected penalization. Time steps are computed based on the local Courant–Friedrichs–Lewy (CFL) number as $\Delta \tilde{t} = CFL_{loc} \Delta \tilde{t}_r$, where the reference time step is defined as:

$$\Delta \tilde{t}_r = \frac{h}{|\mathbf{v}^n|_2}. \tag{3.49}$$

The local CFL number CFL_{loc} is determined by a PID regulator, and $|\mathbf{v}^n|_2$ the local velocity magnitude. This approach assumes that inertial terms are dominant on the element scale and are the limiting factor for a stable time step. However, using this approach, some solutions diverge in the forward solve during optimization. Upon close inspection, divergence happens for higher $q \geq 2$ and when an update in the design causes a new solid element to appear in the fluid domain. To speed up computations, we initialize the flow solution in the current design using the flow solution from the previous design. We found this may cause large non realistic flow speeds and consequently diverging flow solutions in the newly introduced solid elements.

We solve the stability issue by investigating the assumption of dominant inertia in penalized elements. In Section 3.2.2, we assumed that when inertia is dominant in the fluid domain, the penalization terms are dominant in the solid domain. As a consequence, the penalization term in should be dominant over the inertia terms in the solid domain. An appropriate time step in the solid domain should thus be dependent on the penalization magnitude.

To define an appropriate time step, we investigate the order of magnitude of the transient and the penalization term in the solid domain:

$$\mathcal{O}^s\left(\rho \frac{\mathbf{v}^n - \mathbf{v}^{n-1}}{\Delta \tilde{t}}\right) = \rho \frac{v^s}{\Delta \tilde{t}_f}, \qquad \mathcal{O}^s\left(\boldsymbol{f}\right) = \left|\boldsymbol{f}\right|_2 = f^s, \tag{3.50}$$

where we compute $v^s = |\mathbf{v}^n|_2$ at each time step, and define the magnitude of the penalization in the differing approaches as:

$$\text{D: } f^s = D_1(\alpha) v^s \tag{3.51}$$

$$\text{DF: } f^s = \left(D_2(\alpha) + F_2(\alpha) v^s\right) v^s, \tag{3.52}$$

$$\text{DFF: } f^s = \left(D_3(\alpha) + F_3(\alpha) U\right) v^s. \tag{3.53}$$

We use interpolated values $D(\alpha)$ and $F(\alpha)$ as instabilities may already occur in gray elements with relatively high penalization values. Subsequently, we assume the penalization is dominant and limits the maximum time step $(\rho v^s)/\Delta\tilde{t}_f = f^s$:

$$\text{D: } \Delta\tilde{t}_f = \frac{\rho}{D_1(\alpha)} \tag{3.54}$$

$$\text{DF: } \Delta\tilde{t}_f = \frac{\rho}{D_2(\alpha) + F_2(\alpha)v^s}, \tag{3.55}$$

$$\text{DFF: } \Delta\tilde{t}_f = \frac{\rho}{D_3(\alpha) + F_3(\alpha)U}. \tag{3.56}$$

In the fluid domain where $\alpha = 1$ and $\boldsymbol{f} = \boldsymbol{0}$, the inertial term remains dominant. Since the time step is computed per element and inversely proportional to either the inertia or penalization term, we use the smallest computed time step in our solution procedure.

$$\Delta\tilde{t} = CFL_{loc} \min\left(\Delta\tilde{t}_r, \Delta\tilde{t}_f\right). \tag{3.57}$$

To regularize the time step and speed up the computation the local CFL number is included in the time step definition. Using this approach, we found the state solutions to converge more reliably as shown in Sections 3.4 and 3.5.

We note that computational time is generally reduced when a direct steady-state solver is used instead of the pseudo time-stepping scheme. However, when Reynolds numbers increase, the presented pseudo time-stepping approach is often necessary for the state-solution to converge. Pseudo time-stepping performs more gradual updates of the flow and pressure fields, and generally needs more iterations to converge. This results in a larger computational cost, but a more smooth and reliable convergence behavior.

3.4. MODEL INVESTIGATION

In Section 3.2, we described three penalization approaches that should lead to a flow reduction of $v^s/v^f = 10^{-q}$. In Section 3.4.1, we verify the predicted flow reduction. We expect the D approach to be less predictable due to the estimation of the fluid domain velocity magnitude and the DF and DFF approaches to achieve a more predictable flow reduction. Additionally, in Appendix 3.A we investigate the effect of element size h on flow reduction. In Section 3.4.2, we examine the convergence behavior of the optimization by investigating the convexity of the objective function, here chosen as the pressure drop.

3.4.1. MODEL ACCURACY AND FLOW LEAKAGE

To investigate the flow leakage, we use the setup in Figure 3.3 and the geometry and material parameters in Table 3.2. Different inlet Reynolds numbers Re_{in} are investigated and we define the fluid density as:

$$\rho = \frac{Re_{in}\mu}{\overline{v}L}, \tag{3.58}$$

where $\overline{v}$ is the maximum inlet velocity and L the inlet diameter. For the D approach, we estimate the fluid domain flow speed using the maximum inlet velocity $\overline{v}$. For the DFF

approach, we filter the velocity over $N = 10$ elements, i.e., $R = Nh/(2\sqrt{3})$, based on the findings from Appendix 3.B on the effect of the filter radius on flow leakage. The choice of $N = 10$ is based on the maximum filter radius for which overpenalization does not occur.

To check the predictability of the flow leakage, we place solid obstacles with $\alpha = 0$ either in the center (Ω_i^c) or towards the edge (Ω_i^e) of all $i \in \{1:6\}$ channels. The design is constructed such that different elemental Reynolds numbers occur naturally in different parts of the domain. In Channel 6, low flow velocities lead to low elemental Reynolds numbers, while in Channel 1, larger flow velocities lead to relatively high elemental Reynolds numbers. Additionally, flow speeds and elemental Reynolds numbers near the channel walls are significantly lower than in the center of the channels. A unique estimation $\tilde{v}^f$ and thus $\widetilde{Re}_e^f$ for the penalization in the D approach does not exist as the velocity varies throughout the domain, a situation that often occurs during topology optimization.

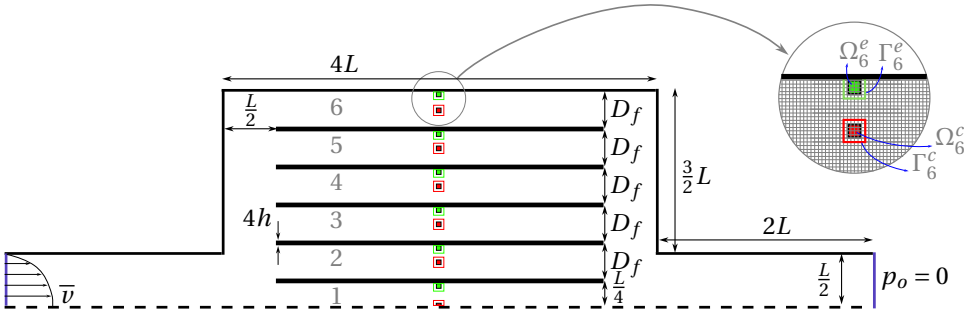


Figure 3.3: The setup to measure the accuracy of the predicted flow leakage. The design is symmetric over the dashed boundary. In Channel 1 to 6, obstacles with volume fraction $\alpha = 0$ are placed. We either add the green obstacles Ω_i^e at the edge or the red obstacles Ω_i^c in the center of the channels. Obstacles consist of a four by four element domain. To measure the flow leakage relative to the local flow magnitude, we define fluid edges Γ_i^e and Γ_i^c as the edges one element away from Ω_i^e and Ω_i^c, respectively.

L	D_f	h	μ	$\overline{v}$	$\tilde{v}^f$	N
1 [m]	$\frac{7L}{20}$	$\frac{L}{80}$	1[Pas]	10[m s^{-1}]	$\overline{v}$	10

Table 3.2: The parameters used to verify the predicted flow leakage using the setup in Figure 3.3.

We compute the flow leakage by comparing the average flow magnitude within obstacles Ω_i^e or Ω_i^c to the average flow speed on Γ_i^e or Γ_i^c, the fluid edges located one element away from the obstacles. Superscripts c and e denote center and edge obstacles, respectively. To average the flow speeds, we use the surface area for the obstacles A_Ω, and length of the center or edge boundaries L_{Γ^c} or L_{Γ^e}, respectively. The flow leakage is

computed as:

$$\epsilon_i^{\mathrm{e}} = \frac{v^s}{v^f} \approx \frac{\int_{\Omega_i^{\mathrm{e}}} |\mathbf{v}|_2 \, d\Omega}{\int_{\Gamma_i^{\mathrm{e}}} |\mathbf{v}|_2 \, d\Gamma} \frac{L_{\Gamma^e}}{A_\Omega}, \qquad \epsilon_i^{\mathrm{c}} = \frac{v^s}{v^f} \approx \frac{\int_{\Omega_i^{\mathrm{c}}} |\mathbf{v}|_2 \, d\Omega}{\int_{\Gamma_i^{\mathrm{c}}} |\mathbf{v}|_2 \, d\Gamma} \frac{L_{\Gamma^c}}{A_\Omega} \tag{3.59}$$

After computing the flow leakage from our solution, we check it against our prediction of $v^s / v^f = 10^{-q}$. The order of the flow reduction $\tilde{q}$ in the i-th channel can be computed when solving with a different $q \in 0, 1, 2, 3$, as:

$$\tilde{q}_{q,i}^e = -\log_{10}\left(\epsilon_i^e\right), \qquad \tilde{q}_{q,i}^c = -\log_{10}\left(\epsilon_i^c\right), \tag{3.60}$$

where subscript q denotes the user-defined parameter for the solution. The measured magnitude of the flow reduction is subsequently used to compute a mean error with respect to our prediction:

$$\xi^e = \frac{\sum_{q=1}^{3} \sum_{i=1}^{6} \left| q - \tilde{q}_{q,i}^e \right|_2}{18}, \qquad \xi^c = \frac{\sum_{q=1}^{3} \sum_{i=1}^{6} \left| q - \tilde{q}_{q,i}^c \right|_2}{18}. \tag{3.61}$$

We do not include results for $q < 1$ in the verification as the flow reduction associated to such values is too low to be accurate and all approaches show similar results. The analysis will be performed for moderate and relatively high inlet Reynolds numbers $Re = 100$ and $Re = 1000$ such that low ($Re_f^e < 1$) and high ($Re_f^e > 1$) elemental Reynolds numbers are present.

In Figure 3.4, the measured flow leakage for varying q can be found. We present only errors in Channels 1 and 6 as in these channels, the highest and lowest flow magnitudes and elemental Reynolds numbers are found, as shown in Figure 3.5. The first observation is that for $Re = 100$, the D approach is in good agreement with the prediction, while for $Re = 1000$, it overpenalizes flow leading to an increased flow reduction. This is caused by an erroneous estimation $\tilde{v}^f$ of the velocity magnitude. For $Re = 100$, the approximate elemental Reynolds number is low $\widetilde{Re}_e^f = 1.25$ and the estimation of $\tilde{v}^f$ does not significantly influence the penalization. For $Re = 1000$, the approximate elemental Reynolds number is larger, $\widetilde{Re}_e^f = 12.5$, and the magnitude of the penalization is largely dependent on the estimation of $\tilde{v}^f$. The estimation of the flow velocity is only valid in the center of Channel 1 where $\widetilde{Re}_e^f \approx Re_e^f$. For all other obstacles, the flow velocity is overestimated leading to excessive flow reduction.

	ξ^e, $Re = 100$	ξ^e, $Re = 1000$	ξ^c, $Re = 100$	ξ^c, $Re = 1000$
D	0.159	0.627	0.280	0.462
DF	0.142	0.086	0.217	0.102
DFF	0.188	0.137	0.299	0.176

Table 3.3: The error in flow reduction for the center obstacles (ξ^c), edge obstacles (ξ^e) for moderate ($Re_{in} = 100$, $Re_e^f \leq 1$) and relatively high ($Re_{in} = 1000$, $Re_e^f > 1$) Reynolds numbers. Examining the errors, the DF approach is found to be the most accurate followed by the DFF and D approaches.

As reported in Table 3.3 by the errors in prediction ξ, both the DF and DFF approaches produce more predictable errors, whereas errors in the D approach spike for

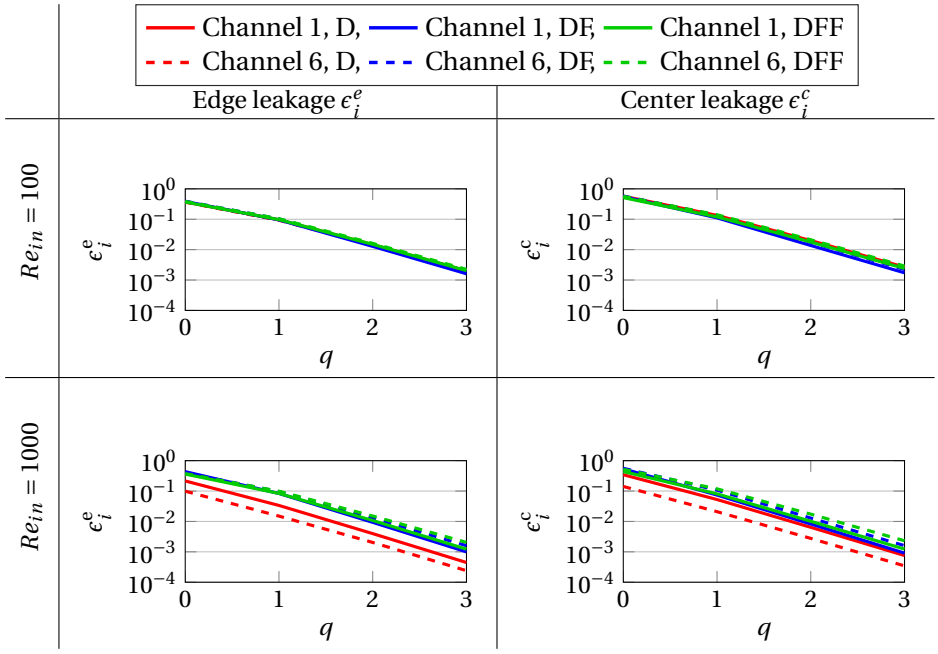
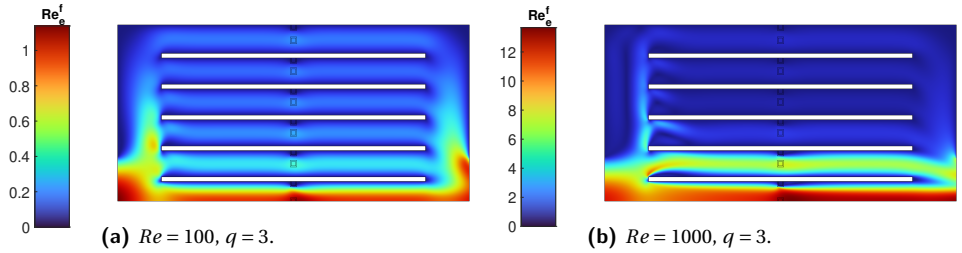


Figure 3.4: Trend in the flow leakage for increasing q in the problem in Figure 3.3 with parameters in Table 3.2. Errors for Channel 1 and 6 are shown as these channels present the highest and lowest flow magnitudes. Both the DF and DFF approaches predict errors well as 10^{-q}. The D approach predicts errors well for low but not for high Reynolds numbers.



(a) $Re = 100$, $q = 3$. **(b)** $Re = 1000$, $q = 3$.

Figure 3.5: Elemental Reynolds number Re_e^f, computed using the D approach. Only edge obstacles Ω_i^e are introduced. Using $\tilde{v}^f = \overline{v}$, we obtain an approximate elemental Reynolds number of $\widetilde{Re}_e^f = 1.25$ ($Re = 100$) and $\widetilde{Re}_e^f = 12.5$ ($Re = 1000$).

$Re = 1000$. In all cases, the DF approach is found to be the most accurate. However, the DF approach shows convergence problems, even with the stabilization procedure in Section 3.3.2 and generally takes more iterations to converge. For high Reynolds numbers, the D approach does not predict flow leakage accurately which will cause problems for the optimization procedure as examined in Section 3.4.2 and 3.5.

3.4.2. OBJECTIVE CONVEXITY AND CONTINUATION APPROACH

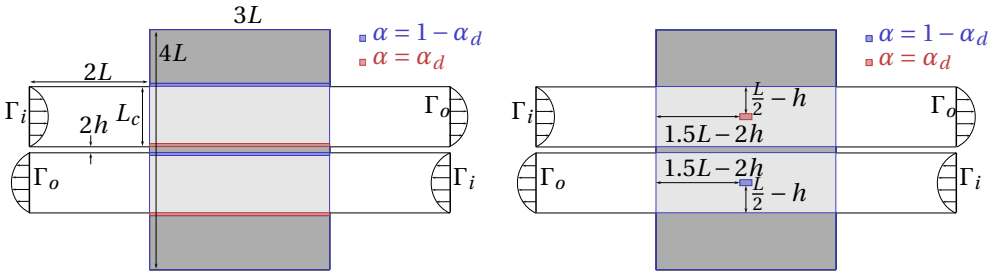
To use the D, DF, or DFF approach for TO effectively, a continuation strategy is derived by investigating the effect of design changes on the objective. In particular, we study how the magnitude of the flow reduction and the shape of the interpolation function for the Darcy and the Forchheimer penalization affect an objective function, here chosen as the pressure drop:

$$g_p = \int_{\Gamma_o} p d\Gamma - \int_{\Gamma_i} p d\Gamma, \tag{3.62}$$

where Γ_o and Γ_i are the flow outlets and inlets, respectively. To examine the interaction between objective and penalization, we use the problems described in Figure 3.6 with parameters in Table 3.4, and a Reynolds-dependent density:

$$\rho = \frac{Re_{in}\mu}{\overline{v}L_c}, \tag{3.63}$$

where L_c is the inlet diameter. To study the convexity of the objective response, we per-



(a) The viscous dominated design change.

(b) The inertia dominated design change.

Figure 3.6: The two-channel problems used to investigate the convexity of the objective. On the inlets Γ_i, a parabolic inflow is applied and on the outlets Γ_o, a parabolic outflow is applied. Both parabolic in- and outflows have maximum velocity $\overline{v}$. In the red and blue areas, the design is changed. While the light gray is fluid ($\alpha = 1$) and the dark gray solid ($\alpha = 0$), the red and blue areas are changed through gray ($\alpha \approx 0.5$) from solid to fluid or vice versa.

form a similar analysis as Lundgaard et al. (2018), who inspect the monotonicity of the objective response for fluid structure interaction problems. We impose a change in design variable from solid, $\alpha = 0$, through gray, $\alpha \approx 0.5$, to fluid, $\alpha = 1$, for certain areas of the design. A convex response presents lower values for gray than for crisp designs, while a concave response takes lower values for crisp than for gray designs. Therefore, a convex response leads to designs that more freely change than for a concave response. Based on this information, a continuation approach can be derived to allow for large design updates in the early optimization stages and to ensure a crisp 0/1 design in later stages.

Two design changes, regulated by parameter $0 \geq \alpha_d \geq 1$, are examined as shown in Figure 3.6. In Figure 3.6a, the channel walls are perturbed over one element to examine viscous dominated design changes. For $\alpha_d = 1$, the upper channel is straight while the lower channel is curved, and vice versa for $\alpha_d = 0$. In Figure 3.6b, a four-by-two-element-island is introduced in the center of the channel to examine inertia dominated

L	L_c	h	μ	$\overline{v}$	$\tilde{v}^f$
1 [m]	L	$\frac{L}{20}$	1[Pas]	1[ms^{-1}]	$\overline{v}$

Table 3.4: Parameters for the analysis of the objective convexity for the problem in Figure 3.6.

design changes. For $\alpha_d = 1$, an island is present only in the top channel, and for $\alpha_d = 0$, only in the bottom channel. The design changes are symmetric to ensure the overall volume fraction in the design domain remains constant, as most optimization procedures involve an active volume fraction constraint. Additionally, as the designs for $\alpha_d = 0$ and $\alpha_d = 1$ are the same when mirrored over the center wall, we find the same objective values for these designs, which allow us to visually observe whether the response is concave or convex.

We present the pressure drop objective g_p for different values of α_d in the two designs defined by Figures 3.6a and 3.6b using either $Re_{in} = 10$ or $Re_{in} = 500$ for, respectively, the DFF , DF, and D approach in Figures 3.7, 3.8, and 3.11. The penalization in the gray areas is lowered using the penalization interpolation approach described in Section 3.2.6 using $\hat{q} \in \{1,2\}$ and the maximum penalization magnitude is defined by by $q \in \{0,1,2,3\}$, which should lead to a flow reduction at the gray/fluid interface of $v^s/v^f = 10^{\hat{q}-q}$.

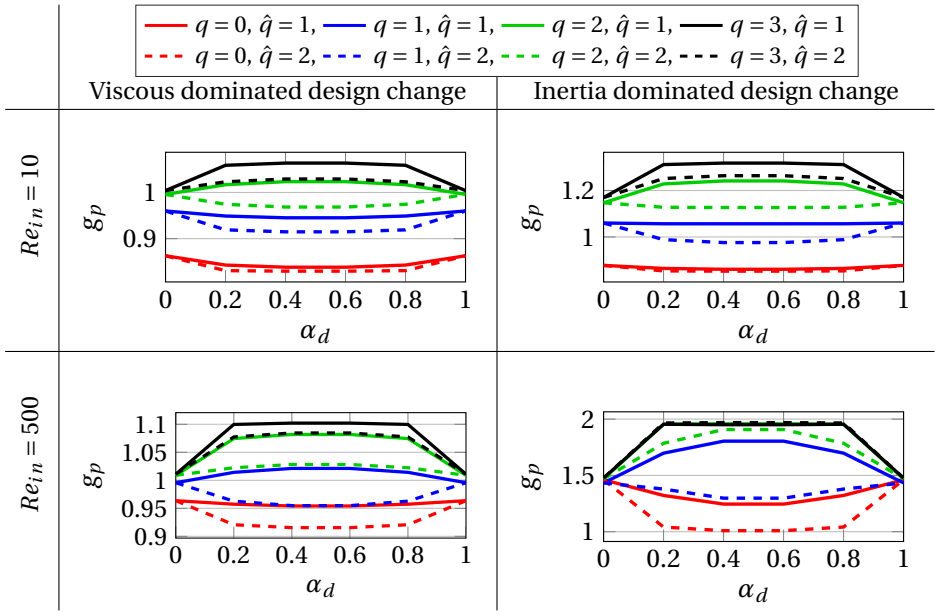


Figure 3.7: Convexity of the pressure drop for the DFF approach using a viscosity dominated (Figure 3.6a) or an inertia dominated design change (Figure 3.6b) at Reynolds number $Re_{in} = 10{,}500$.

First, we analyze the convexity of the objective using the DFF approach in Figure 3.7 and find a clear switch between convex and concave behavior. When the predicted flow reduction is $v^s/v^f = 10^{-q+\hat{q}} > 1$ (q<$\hat{q}$), the response is convex, when it is $v^s/v^f < 1$ (q>$\hat{q}$),

the response is concave, and when it is exactly $v^s/v^f = 1$ ($q = \hat{q}$), the response differs between the sub-figures and is undetermined. We note that for a predicted $v^s/v^f \geq 1$, the penalization is is not active and we measure $v^s/v^f \approx 1$. This behavior can be explained by the constant fluid volume associated to the design changes in Figure 3.6. When the average volume fraction remains constant and gray areas emerge in the design, two scenarios may occur. If the penalization in the gray areas is low enough such that they can be seen as a fluid domain. The total "fluid" domain is increased, which is generally associated with less pressure drop. If the penalization is high enough such that the gray areas can be seen as a solid domain. The "solid" domain is increased which generally leads to an increase in pressure drop.

Secondly, we analyze the convexity of the objective using the DF approach in Figure 3.8. We emphasize that following Section 3.2.6, a steeper interpolation is used for the Forchheimer than for the Darcy penalization. The DF approach does not present a consistent behavior as the DFF approach. For $Re_{in} = 10$, we find convex behavior for $q < \hat{q}$ and concave for $q > \hat{q}$, the same as the DFF approach. The response is thus convex for lower q and concave for higher q. However, for $Re_{in} = 500$ in the viscous dominated design change, we find different behavior. The response is concave for $q = 0$, convex for $q = 1$, and concave again for $q = 2$ and $q = 3$.

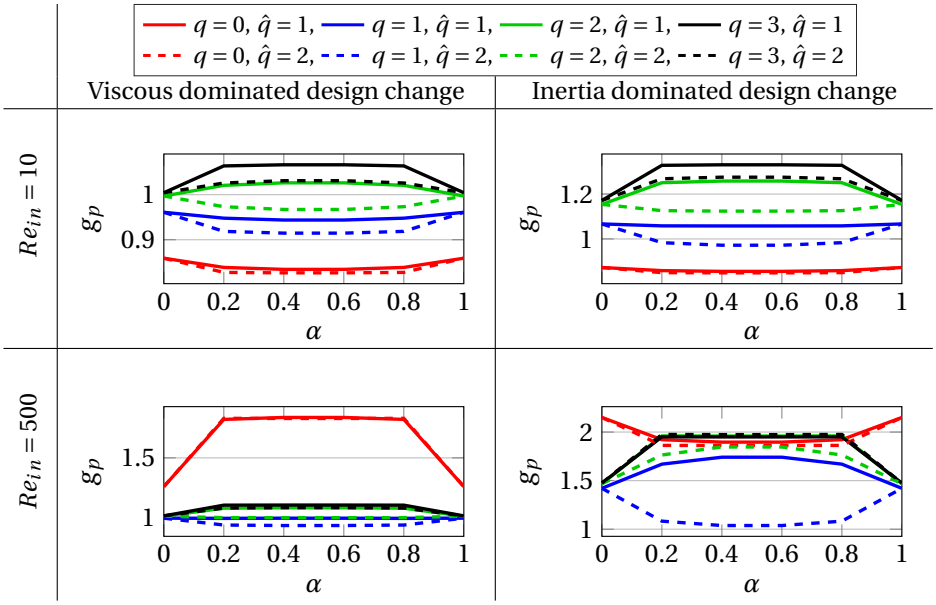
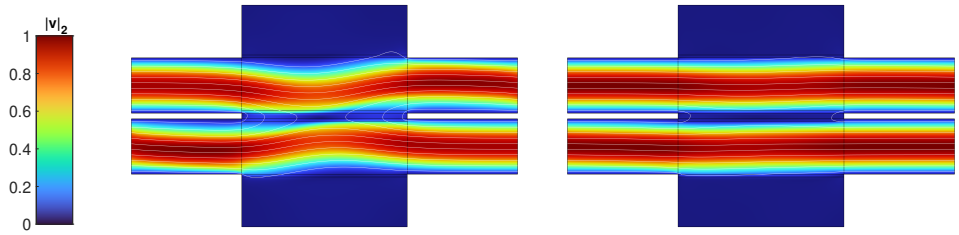


Figure 3.8: Convexity of the pressure drop for the DF approach using a viscosity dominated (Figure 3.6a) or an inertia dominated design change (Figure 3.6b) at Reynolds number $Re_{in} = 10,500$.

The difference in behavior between DF and DFF approaches can be understood from the flow profiles in Figures 3.9a and 3.9b. For $q = 0$, larger flow leakage through the center wall is observed for the DF approach compared to the DFF approach. Subsequently, the leakage causes the flow path to curve resulting in more pressure drop for the DF

approach. Consequently, when gray elements are introduced $0 < \alpha_d < 1$ at the center



(a) DF flow profile (b) DFF flow profile

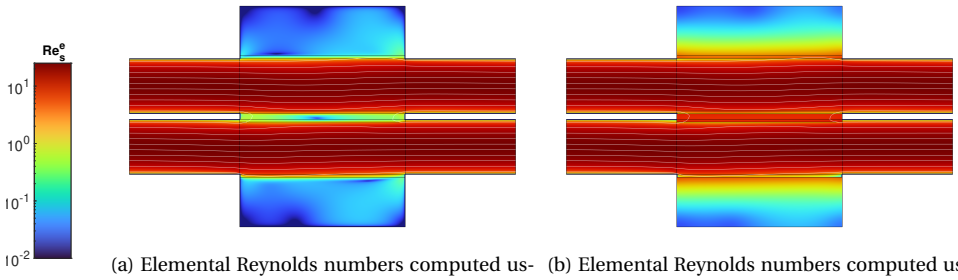
Figure 3.9: Flow profiles for viscous dominated design change (Figure 3.6a) for $q = 0$, $\hat{q} = 1$, $\alpha_d = 0$, and at $Re_{in} = 500$.

wall, the resulting penalization lowers results in more flow leaking through the wall. This further deteriorates the objective, causing the DF response to be concave. However, for $q = 1$, the flow profile for the DF approach is similar to the profile for the DFF approach, and the response becomes convex.

To explain why the DF approach does not reproduce a flow solution similar to the DFF approach for $q = 0$, we further examine the DFF flow solution. In Section 3.2.4, we predicted that a difference in elemental Reynolds numbers in the solid and fluid domains may cause an abrupt change in penalization in the DF approach. In Figure 3.10, the elemental Reynolds numbers are shown for the solution using $q = 0$ and $\alpha_d = 0$. The solid domain elemental Reynolds numbers are computed following Equation 3.34 for the DF approach in Figure 3.10a, and using Equation 3.39 for the DFF approach in Figure 3.10b. In both figures, the elemental Reynolds numbers are computed using the DFF flow solution. The DFF elemental Reynolds numbers in Figure 3.10b are of the same order of magnitude in the solid wall and in the channels. However, the DF elemental Reynolds numbers in Figure 3.10a are at least one order of magnitude lower in the solid wall than in the channels. Moreover, in the solid domain the elemental Reynolds numbers are smaller than 1, $Re_e^s < 1$, while in the fluid domain, they are larger than one, $Re_e^f > 1$. In the solid domain, the Darcy penalization, which should inhibit viscous forces, is dominant, while in the fluid, inertial forces are dominant. For this reason, the flow solution found using the DFF approach cannot be obtained using the DF approach.

It is important to note that the DF approach is not less accurate nor that the flow leakage is less predictable. The examined results are computed for $q = 0$, which results in less predictable flow leakage for all approaches. Additionally, due to the curved flow path through the center wall in the DF flow solution in Figure 3.9a, flow speeds close to the wall (v^f) are higher resulting in a similar flow leakage (v^s / v^f) for both approaches. In fact, these results illustrate that the response computed using the DF approach is less predictable than the one computed with the DFF approach.

Finally, we examine the convexity of the pressure drop for the D approach in Figure 3.11. Generally, the results are similar to those using the DFF approach, i.e., the response is convex for $q < \hat{q}$ and concave for $q > \hat{q}$. However, for $Re_{in} = 500$, $q = 0$, $\hat{q} = 1$ in the viscous dominated change and for $Re_{in} = 500$, $q = 1$, $\hat{q} = 2$ in the inertia



(a) Elemental Reynolds numbers computed using the definition for the DF approach ($Re_s^e = 10^q \frac{\rho|\mathbf{v}|_2 h}{\mu}$) in the solid domain.

(b) Elemental Reynolds numbers computed using the definition for the DFF approach ($Re_s^e = \frac{\rho U h}{\mu}$) in the solid domain.

Figure 3.10: Elemental Reynolds numbers for the viscous design change (Figure 3.6a) for $q = 0$, $\hat{q} = 1$, $\alpha_d = 0$, and $Re_{in} = 500$. All elemental Reynolds numbers are computed using the DFF flow solution in Figure 3.9b.

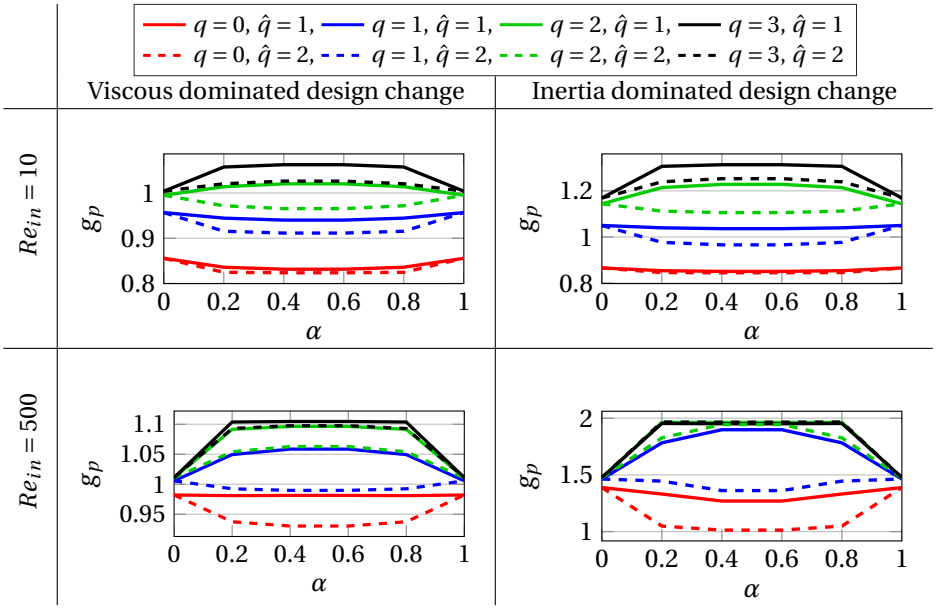


Figure 3.11: Convexity of the pressure drop for the D approach using a viscosity dominated (Figure 3.6a) or an inertia dominated design change (Figure 3.6b) at Reynolds number $Re_{in} = 10,500$.

dominated change, the response is neither completely concave or convex. This results from the estimation of the flow velocity, which leads to an elemental Reynolds number $\widetilde{Re}_e^f > 1$ and consequently to an overestimation of the required penalization in those areas where $Re_e^f < 1$. As we will show in Section 3.5, overpenalization can make the responses concave and leads to convergence to inferior local optima.

Using the results in this section, we create an informed continuation approach for optimization. In the initial stage of optimization, we are interested in the flexibility of the design while the accuracy is less important. In the later stage of optimization, we

want to guide the design to a solid/fluid 0/1 solution which accurately describes the flow and thus the objective and constraint functions. We therefore use a continuation on q and optimize using low q in the initial stage and high q in the later stages. Another approach to add more convexity in the response is to use larger $\hat{q}$, as seen in Figures 3.7, 3.8, and 3.11. However, in the authors' experience, adding a steeper slope to the interpolation function using $\hat{q} > 2$ generally deteriorates the convergence of the optimization procedure.

3.5. TOPOLOGY OPTIMIZATION

In this section, we compare the penalization approaches using optimization examples. Section 3.5.1 focuses on the ability of the approaches to escape ill-performing local optima. In Section 3.5.2, we investigate the relation between design convergence and continuation, and show the limitations of the D approach under varying estimations of velocity magnitude $\tilde{v}^f$.

Beside accuracy and convexity, another important property is the stability of an approach. Stability with respect to the convergence of the flow solution, and with respect to the convergence of the optimization process. In our experience with the DF approach, design updates were often larger and less stable. Moreover, the solution procedure for the flow and pressure fields suffers from convergence issues, even using the stabilization approach described in Section 3.3.2. We found that the initial flow conditions were more important for the flow/pressure fields to converge in the DF approach than in the D or DFF approaches. Using the D and the DFF approach, we could use the flow/pressure solution of the previous design as initial solution for the current design, resulting in much lower computational cost. Using the DF approach, we were often forced to reinitialize the flow/pressure fields resulting in large computational efforts. In Appendix 3.C, we further investigate instabilities in the DF approach. Although the DFF approach is less predictable with respect to flow leakage than the DF approach, it is preferred for its stability. For the remainder of this chapter we do not consider the DF approach.

3.5.1. DEALING WITH ILL-PERFORMING LOCAL OPTIMA

To examine the ability of the D and DFF approaches to escape inferior local optima, we use the problem introduced for the convexity analysis in Figure 3.6. The parameters are provided in Table 3.5 for $Re_{in} = 500$ using the Reynolds dependent density defined in Equation 3.63. The design domain is limited to the gray center areas and the inlet/outlet channels remain unchanged during the optimization. We expect optimized designs to consist of two straight channels with a parabolic flow profile. Using an inlet Reynolds number $Re_{in} = 500$, we expect the elemental Reynolds number Re_f^e to be larger than one in the center of the parabolic flow, and lower than one toward the channel walls.

L	h	μ	$\overline{v}$	$\tilde{v}^f$	N_i	q_{min}	q_{max}	Δq	$\hat{q}$	V_f
1 [m]	$\frac{L}{20}$	1[ms]	1[ms^{-1}]	$\overline{v}$	50	0	3	1	1, 2	$2L_c/4L$

Table 3.5: Parameters for the optimization of the problem in Figure 3.6.

In Section 3.4.2 we defined a general continuation approach which starts the optimization using a low q and ends using a high q. Specifically, we partition the optimization procedure into four stages of maximum $N_i = 50$ design iterations. Each part is terminated after the $N_i = 50$ iterations or when the largest change in a single design variable is less than 10^{-3}. We start the optimization using $q = q_{min} = 0$, and increase q after each stage by $\Delta q = 1$ until $q_{max} = 3$.

To examine the different approaches, we use a problem with a predictable optimum, i.e., two straight channels. This design would use $2L_c/4L$ of the design space and we set a volume constraint to:

$$g_v(\boldsymbol{\alpha}) = \frac{\sum_{i=1}^{N_\alpha} \alpha_i}{N_\alpha} - V_f \leq 0, \tag{3.64}$$

where we have N_α design variables α_i in the design domain Ω_d and $V_f = 2L_c/4L$. Two different inlet diameters and consequent volume constraints will be considered, $L_c = L$ and $L_c = L/2$. We normalize the pressure drop objective in Equation 3.62 with the pressure drop associated to the two straight channel design, i.e., $g_{p0} = 112\mu\overline{v}L/L_c$.

Another important choice is the initial design $\alpha = \alpha_0 \in \Omega_d$. We use two approaches and either start with a fully fluid ($\alpha_0 = 1$) or gray ($\alpha_0 = V_f$) design. Starting from a fully fluid design, largely violating the volume constraint, leads to large design updates that tend to deteriorate the convergence of the forward problem. We found this effect to be worsened by increasing $\hat{q}$. A gray design initially inhibits the flow, decreasing the inertia effects, and favoring convergence towards designs with reduced viscous energy dissipation.

Optimal designs, raw (g_p^*), and post-processed ($g_{p,ref}^*$) objective values for the problem using $L_c = L$ are shown in Figure 3.12, convergence history can be found in Figure 3.13. It is noticeable that using $\hat{q} = 1$ inferior local optima with curved channels are found. For the D approach, both initial designs lead to the inferior optimum, while for the DFF approach, the inferior optimum is only found using $\alpha_0 = 1$ and the superior straight channel optimum is found using $\alpha_0 = V_f$. Increasing $\hat{q}$ to 2 improves the convexity of the objective response and allows the optimizer to escape the local optimum.

In the convergence history in Figure 3.13, we find an increased convergence instability caused by using $\hat{q} = 2$ and $\alpha_0 = 1$. The objective of both the D and DFF approach show a large increase at iteration 4 caused by large design changes. When initializing using $\alpha_0 = 1$, the design violates the volume constraint resulting in relatively large design changes and objective fluctuations. Moreover, a spike in objective for the DFF approach using $\hat{q} = 2$ and $\alpha_0 = 1$ is observed at iteration 35. The volume fraction in the center wall, and the associated penalization, become too low, and flow leaks through the wall. As the volume fraction in the center wall, and consequently the penalization, is increased, the flow profile and objective stabilize. It should be noted that the DFF approach using $\hat{q} = 1$ and $\alpha_0 = V_f$ requires relatively few optimization iterations, see Figure 3.13, and generates the desired topology quickly. Later increase in q do not lead to topology changes but help improve the accuracy of state solution.

Although improved design convergence is found for the D approach when using a more convex interpolation function with $\hat{q} = 2$, this cannot be generalized to other optimization problems. The same problem is investigated but with an inlet diameter $L_c = L/2$, a maximum volume fraction $V_f = 2L_c/4L$, and an estimated elemental

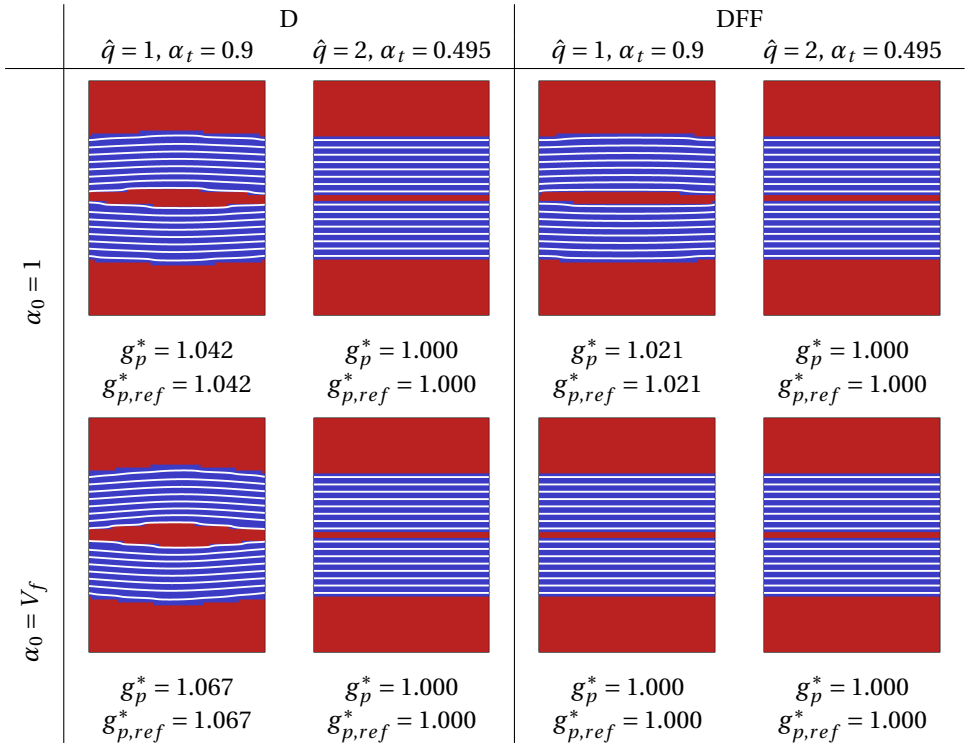


Figure 3.12: Optimized designs computed using $L_c = L$ and associated raw objective value g_p^* and post-processed objective value $g_{p,ref}^*$ following Section 3.2.6 using α_t. Only the solution in the gray design domain in Figure 3.6 is shown.

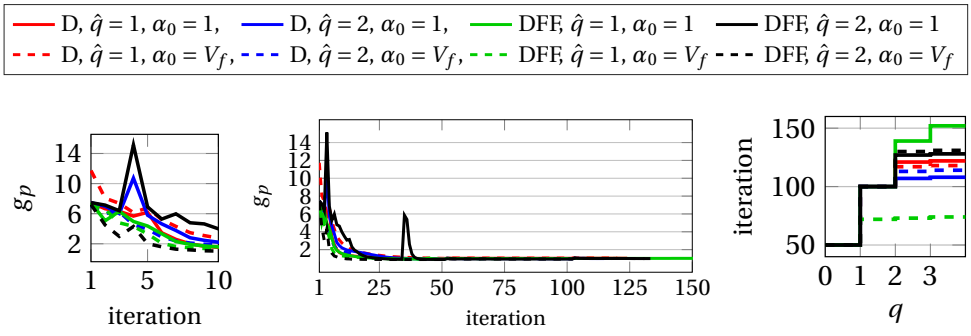


Figure 3.13: The convergence history for the problem in Figure 3.6 using $L_c = L$.

Reynolds number $\widetilde{Re}_e^f = 50$. Resulting designs, raw (g_p^*), and post-processed ($g_{p,ref}^*$) objective values are given in Figure 3.14, convergence history in Figure 3.15. For this slight variation of the problem, the D approach converges to significantly different and inferior optima. The larger $\widetilde{Re}_e^f = 50$ results in a large maximum Darcy penalization $\overline{D}_1$.

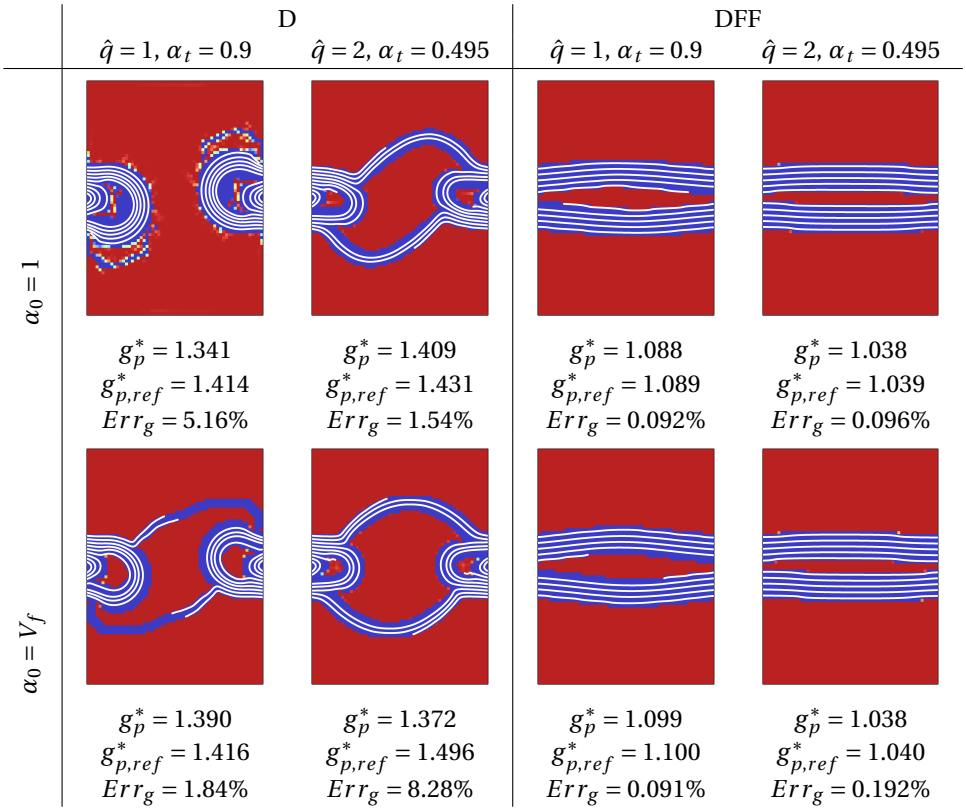


Figure 3.14: Optimized designs computed using $L_c = L/2$ and associated raw objective value g_p^* and post-processed objective value $g_{p,ref}^*$ following Section 3.2.6 using α_t. Only the solution in the gray design domain in Figure 3.6 is shown. Relative errors in objective are defined as $Err_g = (g_{p,ref}^* - g_p^*)/g_{p,ref}^*$.

As predicted in Section 3.4.2, the penalization is overestimated for the straight channel optimum which is dominated by viscous effects. Consequently, the objective response becomes concave and the optimization process converges to an inferior local optimum dominated by inertial effects. Moreover, the D approach designs do not converge to fully solid/fluid designs and present larger gray areas which lower the accuracy of the raw objective values with respect to the post-processed ones, as shown by the large errors $Err_g = (g_{p,ref}^* - g_p^*)/g_{p,ref}^*$ in Figure 3.14.

Comparing the DFF against the D results, it should be noted that while the DFF approach performs better, optimized designs still present bent channels which are sub-optimal. As can be seen in Figure 3.15, the convergence behavior of the DFF approach using $\hat{q} = 1$ is more stable and converges faster than the D approach. Large jumps and fluctuations are observed in the D objective after iteration 50 when we q is updated from 0 to 1. They are caused by the increase in q, which disturbs the objective by reducing the flow leakage through the solid domain, increasing the flow in the fluid domain and, consequently the pressure drop. To limit such disturbances a more gentle update of q is

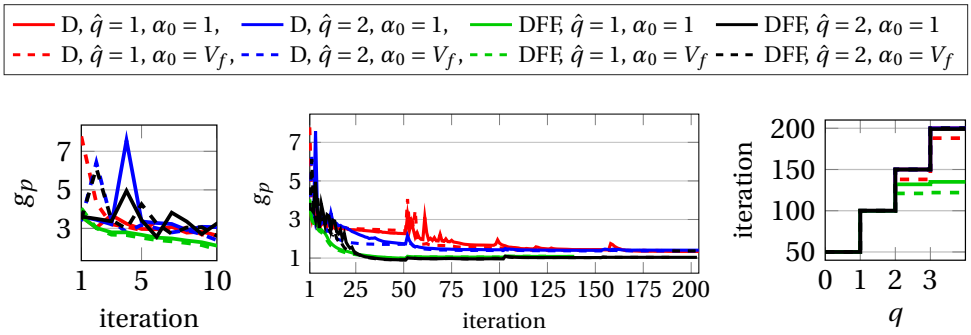


Figure 3.15: The convergence history for the problem in Figure 3.6 using $L_c = L/2$. Large fluctuations in objective are observed for the D approach after iteration 50 where we update q from 0 to 1.

used in Section 3.5.2. Large objective fluctuations are only present in the DFF approach for $\hat{q} = 2$ during the initial convergence when using $q = 0$ in the first 25 design iterations.

In this section, we compared the D and DFF approaches with respect to their ability to escape ill-performing local optima. Optimization methods should balance accuracy of the solution, design flexibility, and convergence. The main issue of the D approach is a conflict between accuracy and design flexibility. As shown by the relatively large errors in objective Err_g found using the D approach in Figure 3.14, a large Darcy penalization is needed for accuracy. However, a large penalization also results in the design converging to ill-performing local optima. This effect is less prevalent in the DFF approach which generally converges to better performing optima.

Another issue in both approaches is a conflict between design flexibility and convergence behavior. To avoid premature convergence to inferior local optima and promote design flexibility, the objective response is made more convex by lowering the penalization for gray design variables and using the higher $\hat{q} = 2$. However, the resulting interpolation function has a steep slope towards the maximum penalization, which often results in large design updates causing large changes in the flow solution and jumps in objective. Using a less steep interpolation function with $\hat{q} = 1$ results in smaller design updates, but a larger tendency to end up in ill-performing local optima.

3.5.2. FLOW INVERTER

In more practical optimization problems, design domains may be larger and flow velocities may vary more drastically. For example, a heat exchanger may present large velocities at the inlet, and relatively low velocities in many branching channels. To illustrate the benefits of achieving a predictable flow reduction in the DFF approach regardless of the local Re_e^f, we tackle a problem with inherently differing elemental Reynolds numbers. The problem is inspired by the flow inverter introduced by Gersborg-Hansen et al. (2005) and revisited recently by Alexandersen (2023). It is assumed that by inverting the flow, velocities locally increase, causing elemental Reynolds numbers to increase and vary throughout the design domain. In the problem shown in Figure 3.16, fluid generally flows from the inlet Γ_i, where a parabolic inflow with maximum velocity $\overline{v}$ is applied,

to the outlet Γ_o, where a constant static pressure ($p_o = 0$) is applied. However, we optimize for the maximum amount of inverted flow in x-direction $-u_p$ at the center of the domain, and minimize:

$$g_v = 1 + \frac{u_p}{u_{p,max}}, \tag{3.65}$$

where $u_{p,max}$ is set to ten times the inlet velocity $u_{p,max} = 10\overline{v}$. We add the volume constraint in Equation 3.64 and a constraint on the inlet pressure:

$$g_p = \frac{\frac{1}{L}\int_{\Gamma_i} p d\Gamma}{\overline{p}(\beta)} - 1 \leq 0, \tag{3.66}$$

where $\overline{p}$ is the maximum allowed pressure drop, dependent on the user-defined parameter β. We define the reference pressure drop $\overline{p}$ assuming a parabolic flow profile as:

$$\overline{p} = \frac{8\mu\overline{v}}{L^2}(1+\beta)5L, \tag{3.67}$$

where the pressure gradient defined as $\partial p/\partial x = 8\mu\overline{v}/L^2$ is multiplied by the length of inlet and outlet channel and we allow for β times the pressure drop in the gray design area of length $5L$. This pressure drop constraint is consistent with Alexandersen (2023). We use the parameters in Table 3.6 and a Reynolds dependent viscosity:

$$\mu = \frac{\rho\overline{v}L}{Re_{in}}. \tag{3.68}$$

As in this problem, a symmetric initial design tends to converge to an ill-performing local optimum (Alexandersen, 2023), the design is initialized using the non-symmetric design with a thin wall on the bottom of the channel in Figure 3.16. We first use $Re_{in} = 100$, $\beta = 30$, $h = L/50$ to compare the results to (Alexandersen, 2023) and investigate the relation between design evolution and continuation approach. In a second application, we will use $Re_{in} = 200$, $\beta = 60$, $h = L/40$ to examine the effect of different estimations of velocity magnitude $\tilde{v}^f$ in the D approach.

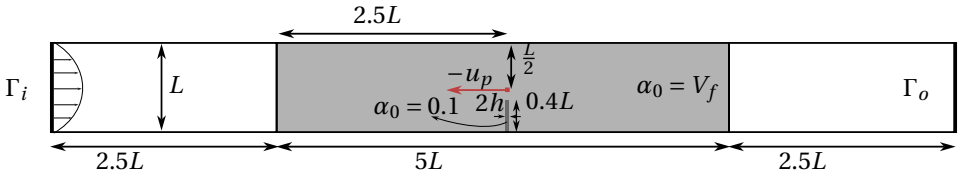


Figure 3.16: The flow inverter optimization problem.

L	ρ	$\overline{v}$	$\tilde{v}^f$	N_i	q_{min}	q_{max}	Δq	$\hat{q}$	V_f
1 [m]	1[kgm^{-3}]	1[ms^{-1}]	$\overline{v}$	20	0	2	$\frac{1}{3}$	1, 2	0.6

Table 3.6: Material and optimization parametersfor the flow inverter in Figure 3.16.

DESIGN EVOLUTION AND CONTINUATION

At the optimum, the pressure drop constraint is generally active and a change in q may drastically perturb it. We thus use a more gradual continuation scheme for q and increase its value from $q_{min} = 0$ to $q_{max} = 2$ by small increments of $\Delta q = 1/3$ triggered every $N_i = 20$ design updates, as shown in Table 3.6. It should be noted that a maximum value $q_{max} = 2$ is selected for this problem, as increasing q further leads to more accuracy of the flow solution, but no significant design changes. For higher q, the pressure response becomes concave and the design does not change much compared to the one found using lower values of q. Moreover, for the D approach and to a lesser extent the DFF approach, using $q = 3$ often causes the flow solution to become unstable and design updates more erratic. Once the maximum value for $q = q_{max}$ is set after 120 iterations, the optimization process is allowed to take 80 extra iterations to perform final shape changes to the design.

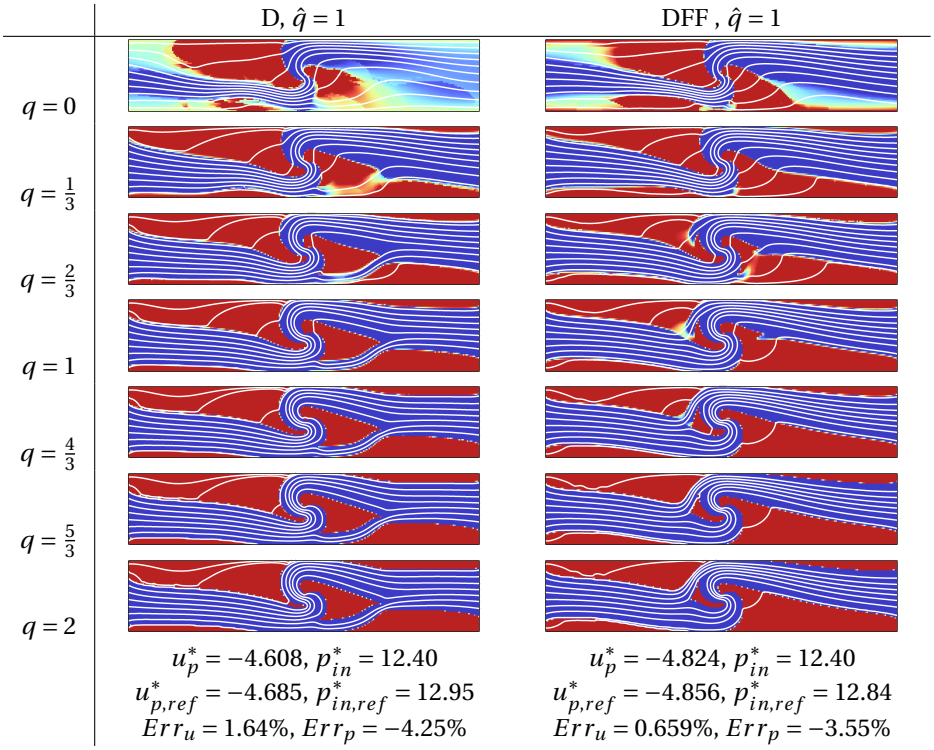


Figure 3.17: The flow inverter designs including streamlines for $Re_{in} = 100$, $h = L/50$ and $\beta = 30$ at the end of each continuation step for q optimized using $\hat{q} = 1$. Only the solution in the gray design domain in Figure 3.16 is shown. Inverted flow velocity and inlet pressure u_p^*/p_{in}^* at the optimum and their post-processed reference values $u_{p,ref}^*/p_{in,ref}^*$, computed using $\alpha_t = 0.45$ following Section 3.2.6, are given. Inlet pressures are constrained using $\overline{p} = 12.4$.

We inspect the optimized designs and their convergence behavior. The designs, inverted flow magnitudes and inlet pressures in the optimized density-based design (u_p^*,

p^*_{in}), and their reference values ($u^*_{p,ref}$, $p^*_{in,ref}$) computed using a post-processed design as described in Section 3.2.6, can be found in Figure 3.17 for $\hat{q} = 1$ and Figure 3.18 for $\hat{q} = 2$. We compute errors with respect to the post-processed reference designs as:

$$Err_u = \frac{u^*_{p,ref} - u^*_p}{u^*_{p,ref}} \qquad Err_p = \frac{p^*_{in} - p^*_{in,ref}}{p^*_{in,ref}},$$

where a negative (resp. positive) error deteriorates (resp. improves) the design. Using $\hat{q} = 1$, the D and DFF approaches find similar performing optima with low errors Err_u, Err_p. Comparing our designs to Alexandersen (2023), both the D and DFF approach find a similar topology. We note that our problem and continuation setup are different and we optimize using a higher maximum penalization than the one used in Alexandersen (2023).

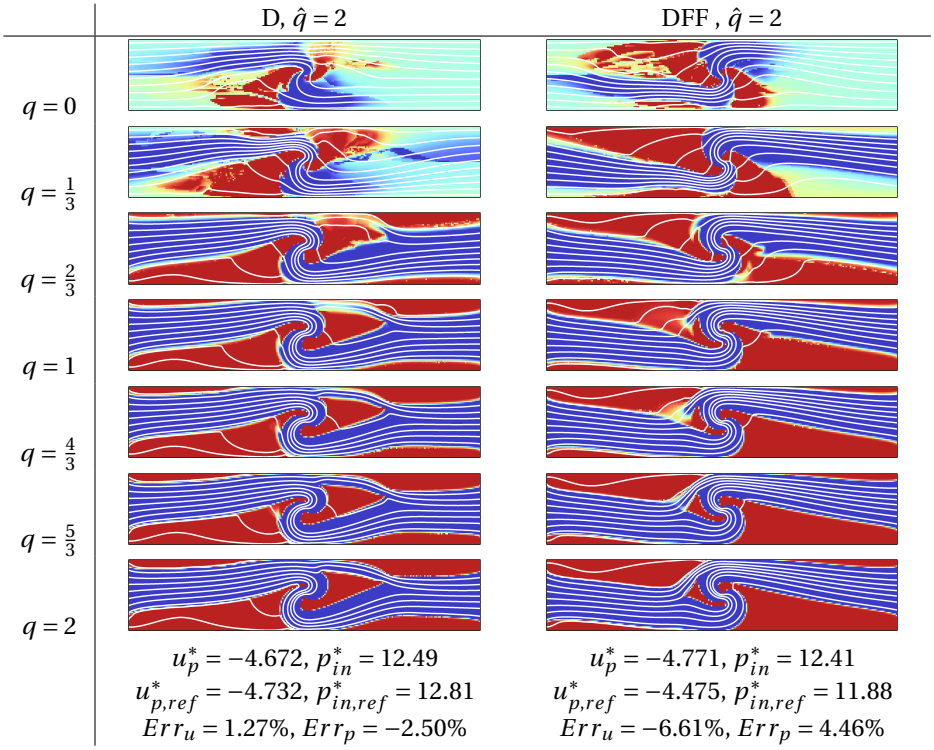


Figure 3.18: The flow inverter designs including streamlines for $Re_{in} = 100$, $h = L/50$ and $\beta = 30$ at the end of each continuation step for q optimized using $\hat{q} = 2$. Only the solution in the gray design domain in Figure 3.16 is shown. Inverted flow velocity and inlet pressure u^*_p/p^*_{in} at the optimum and their post-processed reference values $u^*_{p,ref}/p^*_{in,ref}$, computed using $\alpha_t = 0.082$ folowing Section 3.2.6, are given. Inlet pressures are constrained using $\overline{p} = 12.4$.

To examine the relation between the continuation and design evolution, we show the final design for each continuation step on q in Figure 3.17 for $\hat{q} = 1$ and in Figure 3.18 for

$\hat{q} = 2$. Both approaches find a final topology for lower q after which only shape changes occur for higher q. However, for $\hat{q} = 1$ respectively $\hat{q} = 2$, the D approach generates the distinct fluid/solid topology for $q = \frac{2}{3}$ respectively $q = 1$, and the DFF approach for $q = \frac{4}{3}$ respectively $q = \frac{5}{3}$. The DFF approach settles to its final topology for designs with higher penalization and thus more accurate flow solutions. Using the higher $\hat{q} = 2$ allows the optimizer to keep changing the topology at higher penalization values. However, the final designs in Figure 3.18 for $\hat{q} = 2$ contain more intermediate volume fraction elements at the boundaries. Porous solid/fluid interfaces are caused by the convexity of the pressure drop response, which was found in Section 3.4.2 to be undetermined for $\nu^s/\nu^f = 10^{\hat{q}-q} = 1$. We note that the optimal design found using the DFF approach with $\hat{q} = 1$ performs best.

Another distinction in design convergence is the fact that the DFF approach is able to introduce a fluid channel within a solid domain at later stages of the optimization procedure. For $q = \frac{1}{3}$, a solid domain is constructed in the top left half of the design, although significant flow is present in this porous solid domain. Over the subsequent iterations, the flow through the porous solid domain is inhibited, forcing the creation of new channels through these areas. The final topology is found at $q = 1$ and $q = \frac{4}{3}$ for $\hat{q} = 1$ and $\hat{q} = 2$, respectively. A rationale behind the evolving channels can be found by inspecting the pressure drop constraint in Figure 3.19. At each increase in q, less flow is allowed in the porous solid domain and forced back into the fluid domain, resulting in an increase in the inverted flow magnitude and pressure drop. To counter this effect, side channels bypassing the flow inversion are introduced in areas of large flow leakage in the porous solid domain. For the D approach, the design evolution is straightforward. After finding its first distinct fluid/solid topology, only shape and no topology changes are performed. We note that in Figure 3.18 for $\hat{q} = 2$, the D approach attempts to form a channel through the solid domain in the bottom left half of the design at $q = \frac{5}{3}$. However, the design change is too slow and increasing the penalization using $q = 2$ removes the channel from the design. The DFF approach is more flexible in the sense that even when the penalization and the consequent flow solution accuracy are increased, topology changes are more likely to occur.

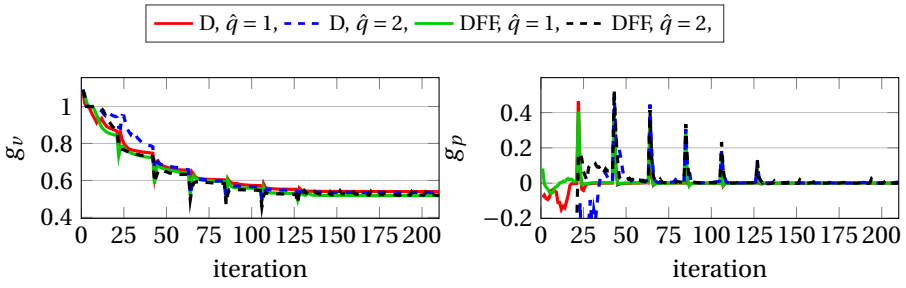


Figure 3.19: The objective and the inlet pressure constraint for the flow inverter design computed with $Re_{in} = 100$ and $h = L/50$. We observe fluctuations in objective and constraint at each increase of q in our continuation.

DESIGN CONVERGENCE FOR DIFFERING ESTIMATIONS OF THE ELEMENTAL REYNOLDS NUMBER

Although elemental Reynolds numbers were varying over the design domain in the previous example, they remained generally low. Using the maximum inlet velocity $v^f = \overline{v} = 1$ or the inverted flow magnitude $v^f = u_p \approx 5$, we find elemental Reynolds numbers of $Re_e^f = 2$ or $Re_e^f = 10$, respectively. To investigate the design convergence for higher elemental Reynolds numbers, we optimize the flow inverter for $Re_{in} = 200$ and $h = \frac{L}{40}$, which leads to an increase in pressure drop for similar inverted flow magnitudes. While the elemental Reynolds number at the inlet remain low, $Re_e^f = 5$, we allow for larger pressure by increasing β from 30 to 60, which should result in a higher elemental Reynolds number at the flow inversion of $Re_e^f \approx 25$.

	D^-	D	D^+
$\tilde{v}^f$	$0.1\overline{v}$	$\overline{v}$	$10\overline{v}$
$\widetilde{Re}_e^f$	0.5	5	50
$\overline{D}_1 \cdot 10^{-q}$	$10^q \frac{\mu}{h^2} = 8$	$10^q \frac{\mu}{h^2} Re_e^f = 40$	$10^q \frac{\mu}{h^2} Re_e^f = 400$

Table 3.7: The estimations for $\tilde{v}^f$ and consequent elemental Reynolds numbers and penalization magnitudes. Values are computed for the flow inverter using $Re = 200$ and $h = \frac{L}{40}$.

To investigate the effect of the penalization on local features where flow speeds and thus elemental Reynolds number vary, we examine the effect of selecting different estimations of $\tilde{v}^f$ and associated elemental Reynolds number $\widetilde{Re}_e^f$. Varying the estimated flow velocity allows us to investigate the effect of an erroneous estimation, as could be encountered in more complex optimization problems. As shown in Table 3.7, we study the D approach for $\tilde{v}^f = 0.1\overline{v}$, $\overline{v}$, and $10\overline{v}$, referred to as the D^-, D, and D^+ approaches, respectively. Using $\tilde{v}^f = 0.1\overline{v}$ and $\widetilde{Re}_e^f = 0.5$, the dominant forces are expected to be viscous, and we penalize using $\overline{D}_1 = 10^q \mu / h^2$. Using $\tilde{v}^f = 10\overline{v}$ and $\widetilde{Re}_e^f = 50$, the penalization magnitude dependends linearly on the elemental Reynolds number as $\overline{D}_1 = 10^q \widetilde{Re}_e^f \mu / h^2$ and is two times larger than a penalization computed using $\tilde{v}^f = u_p \approx 5$ and $\widetilde{Re}_e^f = 25$. The D^+ approach thus uses a fair approximation when the actual maximum velocity magnitude in the design domain is taken into account. To illustrate the advantage of the DFF approach which does not rely on $\tilde{v}^f$, we compare designs optimized using the DFF approach against those optimized using the D^-, D, and D^+ approaches.

We compare the optimized designs, objective, and constraint in Figure 3.20. The D and the DFF approaches show the best performance and find a similar design. The D^- and D^+ approaches yield ill-performing local optima associated with large errors Err_u and Err_p. In the D^- design, a volume fraction $\alpha \approx 0.95$ is found for $q = 2$ in the area shown by the green circle. In this area with flow magnitudes around $v^f \approx 1.5$, the penalization is too low to force the channel to become completely fluid. Moreover, due to the high amount of flow leaking through the solid domains, objective and constraint

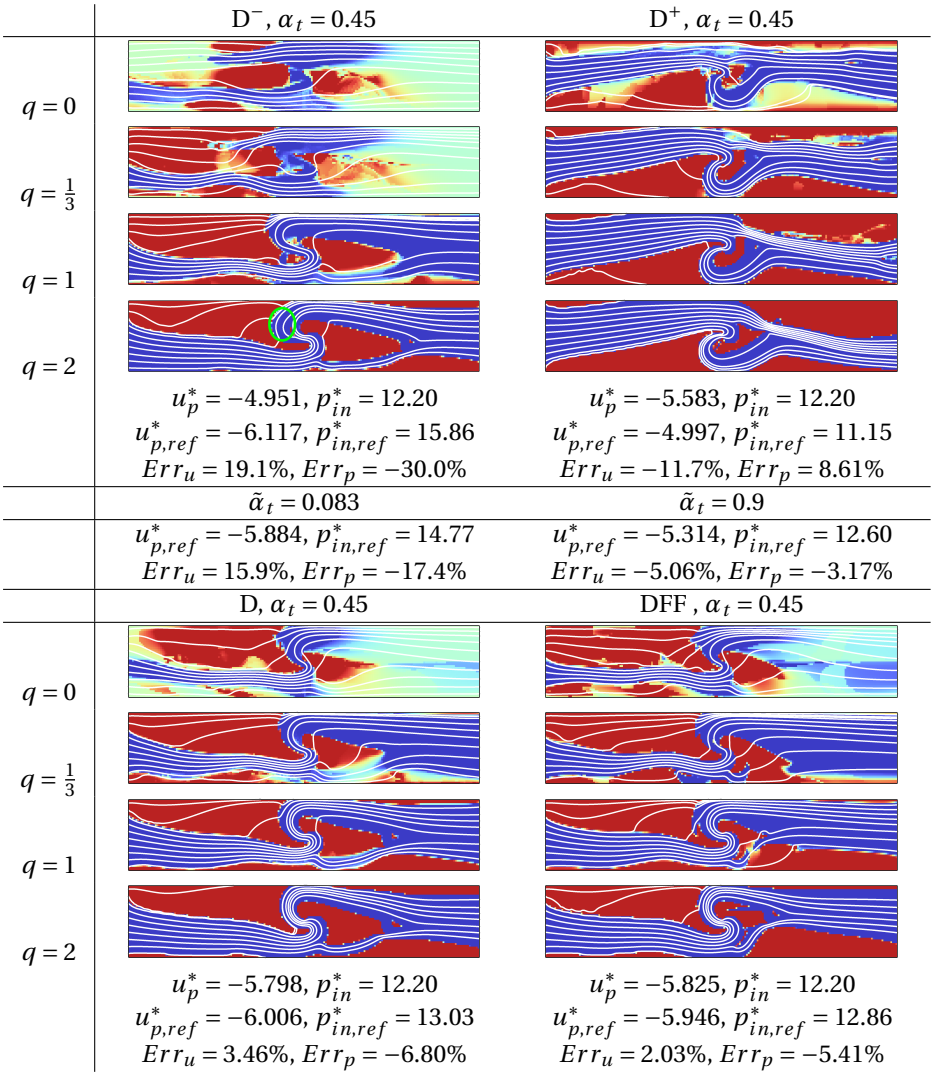


Figure 3.20: The flow inverter designs including streamlines for $Re_{in} = 200$, $h = L/40$ and $\beta = 60$ at the end of continuation steps for q optimized using $\hat{q} = 1$. Only the solution in the gray design domain in Figure 3.16 is shown. In the D$^-$ $q = 2$ design, porous elements of volume fraction $\alpha \approx 0.95$ are found in the area denoted by the green circle. Inverted flow velocity and inlet pressure u_p^*/p_{in}^* at the optimum and their post-processed reference values $u_{p,ref}^*/p_{in,ref}^*$ are given. For the D$^-$ and D$^+$ results, we additionally post-process using updated threshold values $\tilde{\alpha}_t$, based on a compensation of the erroneous penalization magnitude. Inlet pressures are constrained using $\overline{p} = 12.2$.

values have large errors with respect to the reference simulation, computed using the post-processed design as described in Section 3.2.6. The cause of the excessive flow leakage is examined in Figure 3.21 using the elemental Reynolds number in the optimized

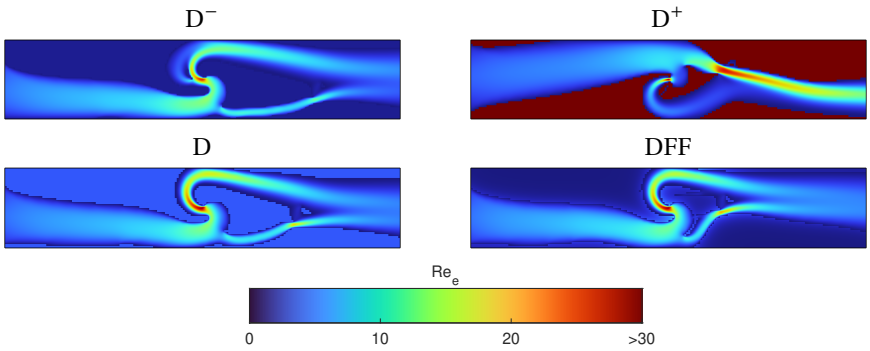


Figure 3.21: Elemental Reynolds numbers in the designs for $q = 2$ from Figure 3.20. In the fluid domain we plot $Re_e^f = (\rho|\mathbf{v}|_2 h)/\mu$. For the DFF approach we plot $Re_e^f = (\rho U h)/\mu$ in the solid domain. For the D, D$^-$, and D$^+$ approaches, we approximate and plot the constant $\widetilde{Re}_e^f$ as found in Table 3.7 in the solid domain. For D$^+$ approach, the solid domain elemental Reynolds number is $\widetilde{Re}_e^f = 50$, but the color-scale is limited to 30 for enhanced readability.

designs. For the D$^-$ approach, the elemental Reynolds numbers abruptly decrease at the solid/fluid interface, and the solid domain significantly underestimates the fluid domain elemental Reynolds numbers. Consequently, the penalization based on $\widetilde{Re}_e^f < 1$ assumes viscous terms are dominant, whereas the higher inertial terms are actually dominant in the fluid domain where $Re_e^f > 1$.

The D$^+$ designs in Figure 3.20 quickly converge to an inferior local optimum. Due to the high penalization, the design topology is identified using $q = 0$, and does not undergo any large modifications over subsequent continuation steps. The elemental Reynolds number is significantly overestimated in the solid domain as shown in Figure 3.21. Moreover, with respect to the post-processed design, the D$^+$ design presents large errors Err_u and Err_p, which have two origins. Firstly, the design contains small, but crucial, features of only one element in size, which are gray ($\alpha \approx 0.5$) but have a relatively large impact on the flow. The optimizer is thus misusing the high penalization in intermediate density elements to improve the objective. Secondly, when post-processing using the approach in Section 3.2.6, we assume that a correct penalization is used, such that $v^s/v^f < 10^{-1}$ in the solid domain where $\alpha < \alpha_t$. However, due to the overestimation of the elemental Reynolds number and consequent over-penalization, the prediction of v^s/v^f becomes inaccurate. Consequently, we are not able to threshold the design appropriately. In general, issues related to the under- or over-estimation of the velocity become more significant when considering designs with many branching flow channels and differing flow magnitudes.

As we expect post-processing to be inaccurate due to erroneous estimations of $\widetilde{Re}_e^f$, an adapted post-processing approach for the D$^-$ and D$^+$ designs is examined. In Equation 3.47, we determine the threshold value as $\alpha_t = (1 - 10^{r-q})/(1 + 10^{r-q+\hat{q}})$, using a flow reduction in the solid domain of $v^s/v^f < 10^{-r}$. This value is based on an appropriate penalization magnitude and estimation of $\widetilde{Re}_e^f$. In all designs, we find $u_p \approx 5$ and we

assume an appropriate estimation to be $\widetilde{Re}_e^f = 5$. For the D$^-$ approach, the penalization is thus five times too low. To compensate for this underpenalization, the threshold value should be chosen at a penalization value which is five times higher than the one found at α_t. This is found using $r = 1$ at a threshold of $\tilde{\alpha}_t = (1 - 5 \cdot 10^{r-q})/(1 + 5 \cdot 10^{r-q+\hat{q}}) = 0.083$. For the D$^+$ design, a similar analysis results in a threshold of $\tilde{\alpha}_t = 0.9$. Using these threshold values, we find the errors with respect to the post-processed design to decrease in Figure 3.20. This poses a challenge to post processing larger designs with many branching channels as an appropriate threshold value depends on a local elemental Reynolds number, and a unique threshold α_t may not exist.

The flexibility and stability of the design convergence are investigated by inspecting the objective and constraint histories in Figure 3.22. For the D$^-$ approach, we find the objective to converge slowly over the first iterations. This is caused by the fact that the penalization is not high enough to sufficiently guide the flow and impact the design. For the D$^+$ approach, the objective decreases relatively quickly. After the $q = \frac{1}{3}$ continuation step, no large design changes happen and the main reduction in objective is caused by a reduction of the flow through the porous solid areas. Moreover, due to the high penalization, we observe oscillatory behavior of the pressure constraint, especially in the first 30 iterations. The D and DFF approaches both converge smoothly, except at iterations where q is increased. The D$^-$ approach is unable to accurately resolve the physics to represent the design and the D$^+$ approach prematurely converges towards an ill-performing local optimum. This demonstrates that the tuning of the penalization in the D approach is both crucial and sensitive. Conversely, the DFF approach requires no velocity estimate and shows design flexibility while the optimal design remains accurate.

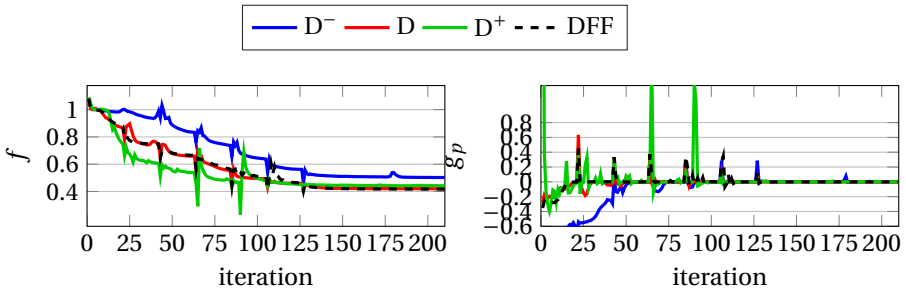


Figure 3.22: The objective and the inlet pressure constraint for the flow inverter design computed with $Re_{in} = 200$ and $h = L/40$.

DESIGN CONVERGENCE FOR VARYING MESH SIZES

One of the main findings in Section 3.2 is that an appropriate magnitude for both the Darcy and Forchheimer penalization is dependent on the element size h. In Section 3.4.1, we found the mesh dependent penalization to result in reliable predictions for the flow leakage and in Appendix 3.A we confirm these findings for further mesh refinement. In this section, we examine the effect of the mesh size on optimization. Kreissl and Maute (2012) find that increasing the penalization magnitude for improved accuracy may result in the optimizer to converge to inferior local optima. In Table 3.1, we

find the penalization to increase with decreasing element size h. We examine whether the behavior of the penalization with mesh refinement leads to a stronger tendency to converge to inferior local optima.

	D			DFF		
	$h = \frac{L}{10}$	$h = \frac{L}{40}$	$h = \frac{L}{150}$	$h = \frac{L}{10}$	$h = \frac{L}{40}$	$h = \frac{L}{150}$
$\widetilde{Re}_e^f$	10	2.5	0.66	-	-	-
$\overline{D} \cdot 10^{-q}$	10	40	225	1	16	225
$\overline{F} \cdot 10^{-q}$	-	-	-	10	40	150

Table 3.8: The approximate elemental Reynolds number $\widetilde{Re}_e^f$, and Darcy/Forchheimer penalization magnitudes $\overline{D}/\overline{F}$ for the differing mesh sizes used to optimize the flow inverter.

To examine the effect of the penalization, we optimize the flow inverter using $Re = 100$, $\beta = 30$, and $\hat{q} = 1$, such that the results in Figure 3.17 are reproduced. A coarser mesh with $h = L/10$ and a finer mesh with $h = L/150$ are examined, which results in the approximated elemental Reynolds numbers and penalization magnitudes shown in Table 3.8. Refining the mesh for the D approach, the Darcy penalization varies one order of magnitude. However, for the DFF approach, the Darcy penalization varies two orders of magnitude, whereas the Forchheimer penalization varies only one order of magnitude. We find significantly higher penalizations to be applied when using a smaller element size.

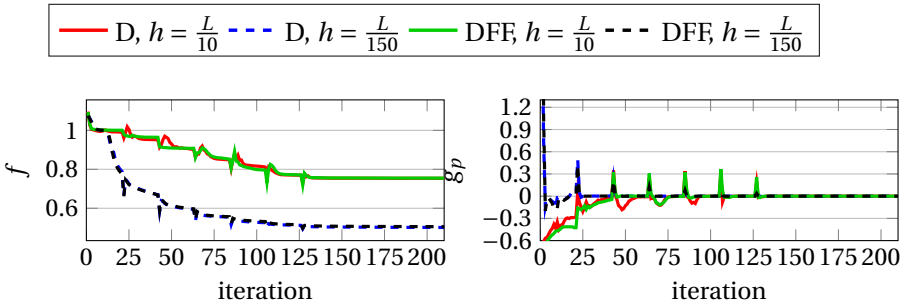


Figure 3.23: The objective and the inlet pressure constraint for the flow inverter design computed with $Re_{in} = 100$ for $h = L/10$ and $h = L/150$.

In Figure 3.24, we plot the design history for $h = L/10$ and $h = L/150$. For $q = 0$, large gray areas are present in the designs. Increasing q, these gray areas converge to either solid or fluid. The convergence history in Figure 3.23 shows more change in objective and constraint for low values of q in the earlier stages of optimization than for high values of q in the later stages. Using the coarser and finer mesh in Figures 3.24 and 3.23, the design convergence behaves similar to the convergence using $h = L/40$ as shown in Figures 3.17 and 3.19. Using the continuation approach presented in this chapter, the increased penalization due to mesh refinement thus does not significantly impact convergence behavior.

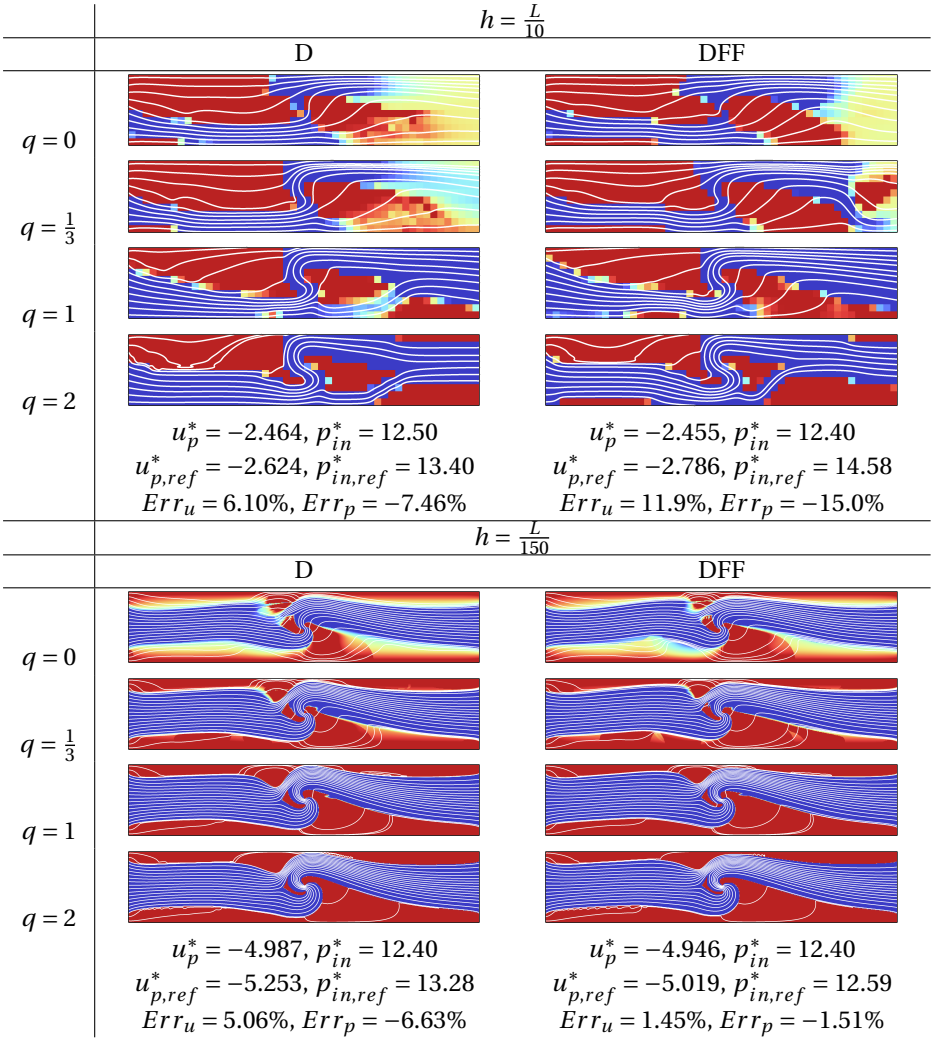


Figure 3.24: The flow inverter designs including streamlines for $Re_{in} = 100$ and $\beta = 60$ at the end of continuation steps for q optimized using $\hat{q} = 1$. Only the solution in the gray design domain in Figure 3.16 is shown. Solutions are computed for $h = \frac{L}{10}$ and $h = \frac{L}{150}$. Inverted flow velocity and inlet pressure u_p^*/p_{in}^* at the optimum and their post-processed reference values $u_{p,ref}^*/p_{in,ref}^*$ computed using $\alpha_t = 0.45$ following Section 3.2.6, are given. are given. Inlet pressures are constrained using $\overline{p} = 12.4$.

3.6. DISCUSSION

One of the main advantages of the Darcy with Filtered Forchheimer (DFF) approach is the reduced parameter tuning. While common approaches require trial-and-error to find an appropriate penalization magnitude, we select a penalization magnitude for a desired flow reduction of $v^s/v^f = 10^{-q}$ using Table 3.1. Moreover, we control convexity of the pressure drop response using both the penalization magnitude with q, and

the penalization interpolation with $\hat{q}$. The DFF approach does not require tuning based on simulation results, but allows engineers to make an informed selection of the appropriate penalization based on a desired balance between flow solution accuracy and objective convexity. In practice, we recommend using $\hat{q} = 1$ and control convexity of the pressure drop response using q. Subsequently, a continuation on q can be derived starting with $q < \hat{q}$, to create a convex pressure drop response, and ending with $q > \hat{q}$, to create a concave pressure drop response. We recommend to initialize the optimization using $q = 0$ and finalize using $q = 2$. While choosing $q > 2$ decreases the flow leakage, we generally found the pressure drop response to become too concave for effective design updates. The update on q should be done in small increments to avoid destabilizing the design process. We found that increasing q by $\Delta q = 1/3$ every $N_i = 20$ design iterations performed well for the relatively unstable flow inverter design. However, more efficient continuation strategies are suggested as a subject for future research.

As discussed in Appendix 3.B, the filter radius is defined in terms of number of elements N, which should not change for a different element size. Although an appropriate penalization can also be defined for a different number of elements N, tuning this parameter is not recommended. The main parameter determining the order of magnitude of the penalization is q. As a continuation on q is proposed, the required accuracy of the filtered velocity field is relaxed. A radius within $6 \leq N \leq 12$ elements will have enough accuracy for the continuation approach to work and the final design to be accurate.

The most common continuation strategy in flow TO is to start the optimization procedure with a highly convex penalization interpolation, i.e., equivalent to high $\hat{q}$ in this paper, and to finalize the optimization process using a less convex interpolation function, i.e., low $\hat{q}$ in this paper. We verified that the convexity of the pressure drop objective depends on the penalization magnitude in the gray areas of intermediate volume fraction. A highly convex interpolation function with a steep slope towards the maximum penalization has the same effect as the low maximum penalization used in this work, as both reduce the penalization in gray areas. However, using a steep slope in the penalization interpolation often causes design updates to become more erratic. A small change in design variable may have a large effect on the flow solution due to a sudden high penalization, which can drastically change the flow solution over design iterations due to the non-linear nature of the Navier-Stokes equations. For this reason, we prefer to use a continuation on the penalization magnitude instead of on the penalization interpolation. However, hybrid methods which apply a continuation on both the penalization interpolation and magnitude may be derived using the convexity analysis and the prediction of flow reduction in this paper.

As a first attempt at using the Forchheimer penalization previously introduced by Alonso and Silva (2022), the Darcy with Forchheimer (DF) approach was introduced which improves on previous work by using an order analysis for the appropriate penalization magnitude. Although the DF approach is found to more reliably predict the flow reduction than the DFF approach, it also suffers from unstable flow solutions. Unstable flow solutions are mainly caused by reusing the state solution over subsequent design iterations, as reusing the state solution significantly decreases the computational effort. In the authors' experience, when attempting optimization using the DF approach, more erratic design updates are encountered, increasing the tendency to find diverging flow so-

lutions when reusing the state solution. Furthermore, we note that the DF approach can be seen as a special case of the DFF approach where the velocity is averaged over a radius $R = 0$, and the penalization magnitude is increased. A hybrid between the DF and DFF approaches could thus be derived. Using a smaller filter radius R, the averaged flow magnitude will underestimate the fluid domain flow magnitude $U < v^f$, which may be compensated by using a higher maximum Forchheimer penalization $10^q \rho / h < \overline{F} < 10^{2q} \rho / h$. Using this approach, a novel method may be derived which is as stable as the DFF approach and finds accurate flow reductions as the DF approach.

The presented study on laminar moderate Reynolds flow TO is a first step towards improved understanding of turbulent high Reynolds flow TO. For future work, we recommend a similar procedure to derive an appropriate penalization strategy for turbulent flow TO using Reynolds Averaged Navier-Stokes (RANS) equations. The procedure should follow three steps: 1) A dimensional analysis on the discretized physics, similar to the one presented in Section 3.2. 2) The flow reduction and other turbulent boundary conditions at the solid/fluid interface are verified using an analysis similar to Section 3.4.1. 3) The convexity of the objective response is inspected using a method similar to Section 3.4.2. As turbulent flow inherently contains high Reynolds numbers, we expect the DFF approach to be required to appropriate penalize the RANS momentum equation. Moreover, a similar procedure to derive a robust TO approach may be performed for other problems. An example is TO problems involving thermo-fluid equations which often require tuning to find appropriate material interpolation functions. To this end, we introduced a more general analysis approach in Section 3.2.1 than the one presented in Theulings et al., 2023. While the analysis in Theulings et al., 2023 is specific for flow physics and relies on the continuous pressure gradient, the analysis in this work can be extended to penalization or interpolation approaches for different physics.

3.7. CONCLUSION

To derive a reliable penalization approach for moderate Reynolds flow TO, the Forchheimer penalization is crucial. While the flow in areas where viscosity is dominant, is inhibited using the Darcy penalization, to penalize the flow in areas where inertia is dominant, the Forchheimer penalization is used. The Darcy penalization alone cannot simultaneously penalize the flow in both areas appropriately.

A reliable penalization and continuation approach for density-based TO of laminar moderate Reynolds flow problems has been introduced and compared to the state-of-the-art. The novel DFF approach is based on a Forchheimer penalization dependent on a filtered velocity, and a continuation strategy with a predictable flow reduction in the solid domain. Moreover, the approach does not depend on a specific problem setup, as it can be used without additional tuning to optimize different inlet/outlet configurations with different Reynolds numbers, and different mesh sizes. We improve all four conditions, stated in Section 3.1, for a reliable approach as follows: 1) Continuing our previous work in Theulings et al. (2023), parameter tuning is reduced by deriving appropriate penalization magnitudes for a predictable flow reduction. 2) As the flow reduction is predictable in the DFF approach, we can guarantee its value and that the flow solution is accurate in the optimal design. 3) By analyzing the convexity of the pressure drop response, a continuation strategy is derived for both the D and the DFF approach which

mitigates the tendency to converge to ill-performing local optima. By starting the optimization procedure with a low penalization, the pressure drop response is convex and the design easily changes. Subsequently, the maximum penalization is increased making the pressure drop response concave and forcing the design into a discrete solid/fluid solution. 4) Although no thorough analysis is performed on the stability of the flow solution, we found relatively quickly and reliably converging flow solutions for both the D and the DFF approach.

3

3.A. MODEL ACCURACY FOR VARYING ELEMENT SIZE

In Section 3.4.1, we examined the accuracy of the predicted flow leakage. In this appendix, we further analyze the flow leakage for varying element size h. Since all penalization magnitudes in Table 3.1 scale with $1/h$ or $1/h^2$, significantly higher penalization magnitudes are required when using a smaller element size. A higher penalization magnitude is expected to result in lower overall velocity magnitudes in the solid domain. However, flow leakage is defined as the ratio of velocity magnitudes in neighboring fluid and solid elements. Decreasing the mesh size, we expect the flow leakage in solid elements neighboring fluid elements to remain $v^f/v^s = 10^{-q}$. We thus examine the interplay between element size, flow leakage, and velocity magnitude in the solid domain.

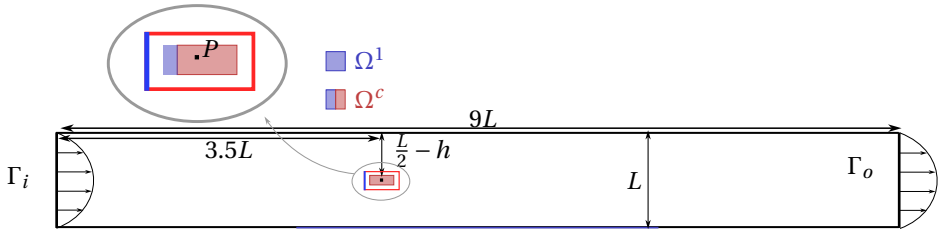


Figure 3.25: Benchmark design for testing the effect of different element sizes h on the flow leakage and absolute flow magnitude in the solid domain. The design consists of a straight flow channel with a solid island Ω^c introduced in the center of the domain. The island is $4h_0$ wide and $2h_0$ high, where $h_0 = L/12$ is the largest element size used. Flow leakage in the front column of elements at the left edge of Ω^c is measured using the one element thick blue domain Ω^1 and blue boundary Γ^1. Overall flow leakage is measured using the full island domain Ω^c (red and blue domains) with boundary Γ^c (red and blue boundary). The boundaries are separated by one fluid element from their associated domains. Moreover, to measure the absolute velocity magnitude, Point P at the center of the island is introduced.

L	μ	$\overline{v}$	$\tilde{v}^f$
1 [m]	1[Pas]	1[ms^{-1}]	$\overline{v}$

Table 3.9: The parameters used in Figure 3.25 to examine the relation between element size, flow leakage, and absolute flow magnitude.

For the analysis of flow leakage, we use the design in Figure 3.25 with the parameters in Table 3.9 and the Reynolds dependent density in Equation 3.58. We introduce a solid obstacle Ω^c in the center of the flow channel. The island has a width of $4h_0$ and a height

of $2h_0$, where $h_0 = L/12$ is the largest element size. When the mesh is refined and the size of the elements decreases, the island remains the same size and consists of more elements. Flow leakage ϵ is defined using Equation 3.59 and measured over the whole island as ϵ^c using Ω^c and Γ^c, or only in the front column of elements as ϵ^1 using Ω^1 and Γ^1. Additionally, we measure the velocity magnitude $|\mathbf{v}(\boldsymbol{x}_P)|_2$ in the center of the island at Point P. Three different element sizes $h = L/12,\ L/48,\ L/192$ are examined for two Reynolds numbers $Re_{in} = 50,\ 500$, leading to the penalization magnitudes in Table 3.10. We observe that the appropriate penalization magnitudes increase at least an order of magnitude for decreasing h.

	$Re = 50$					
	D			DFF		
	$h = \frac{L}{12}$	$h = \frac{L}{48}$	$h = \frac{L}{192}$	$h = \frac{L}{12}$	$h = \frac{L}{48}$	$h = \frac{L}{192}$
$\widetilde{Re}_e^f$	4.2	1.0	0.26	-	-	-
$\overline{D}10^{-q}$	$6\cdot10^2$	$2.4\cdot10^3$	$3.7\cdot10^4$	$1.4\cdot10^2$	$2.3\cdot10^3$	$3.7\cdot10^4$
$\overline{F}10^{-q}$	-	-	-	$6\cdot10^2$	$2.4\cdot10^3$	$9.6\cdot10^3$
	$Re = 500$					
	D			DFF		
	$h = \frac{L}{12}$	$h = \frac{L}{48}$	$h = \frac{L}{192}$	$h = \frac{L}{12}$	$h = \frac{L}{48}$	$h = \frac{L}{192}$
$\widetilde{Re}_e^f$	42	10	2.6	-	-	-
$\overline{D}10^{-q}$	$6\cdot10^3$	$2.4\cdot10^4$	$9.6\cdot10^4$	$1.4\cdot10^2$	$2.3\cdot10^3$	$3.7\cdot10^4$
$\overline{F}10^{-q}$	-	-	-	$6\cdot10^3$	$2.4\cdot10^4$	$9.6\cdot10^4$

Table 3.10: The penalization magnitudes for differing elements sizes h used for the problem in Figure 3.25.

Since the island is at the center of the domain, the approximation of $v^f \approx \overline{v}$ is expected to be appropriate and the D approach to be accurate. Figure 3.26 provides the flow leakage for $q = 0,1,2,3$ and shows $\epsilon^1 \approx v^s/v^f = 10^{-q}$ for both the D and DFF approach. The leakage in the solid elements neighboring fluid elements at the front of the island thus behaves as expected. When velocity magnitudes are compared between neighboring solid and fluid elements, an appropriate penalization is dependent on element size.

Although the expected leakage is recovered for ϵ^1, the overall leakage ϵ^c behaves differently. For smaller h, the overall leakage in Figure 3.26, is found to be too strict as $\epsilon^c \ll 10^{-q}$. For $h = L/192$, the island is 32 elements high and 64 elements wide and ϵ^c does not measure the leakage between neighboring fluid and solid elements. Flow leakage as defined in this chapter can only be appropriately examined when comparing velocity magnitudes in solid elements and neighbouring fluid elements.

At Point P, the velocity magnitude $|\mathbf{v}(\boldsymbol{x}_P)|_2$ in Figure 3.26 drastically decreases for

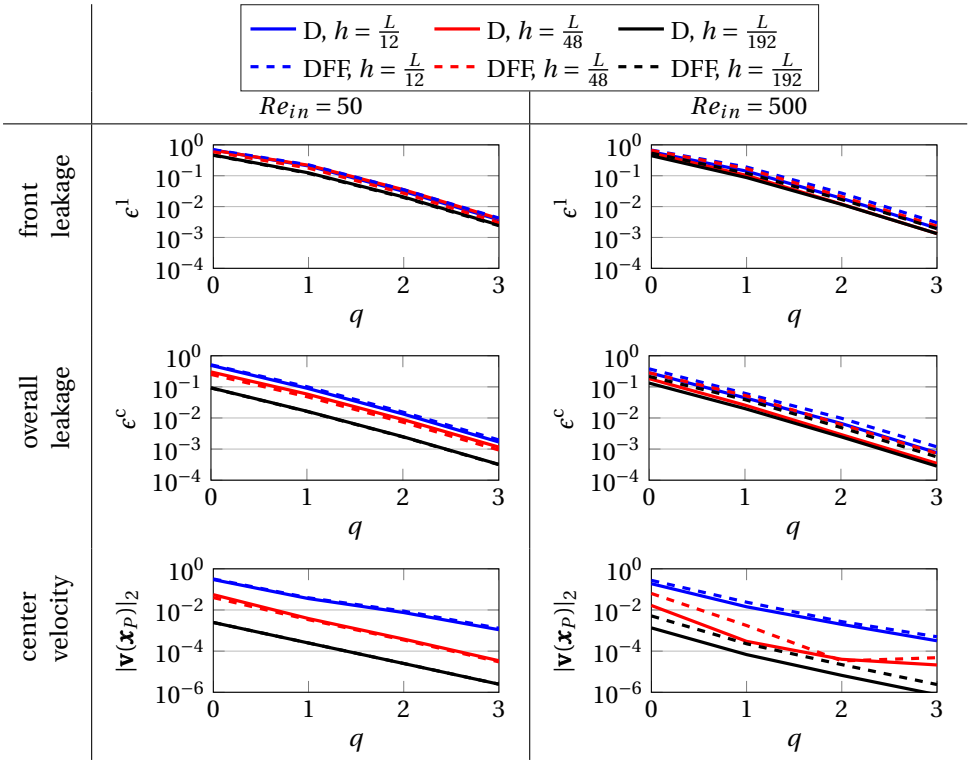


Figure 3.26: Trends in flow leakage and velocity magnitude for increasing q in the problem in Figure 3.25. For the front leakage, we find the expected error $\epsilon^1 = 10^{-q}$. We note that using $Re_{in} = 500$ and $h = L/48$, both the D and the DFF approach find a center velocity $|\mathbf{v}(\boldsymbol{x}_P)|_2$ of the same magnitude for $q = 2$ and $q = 3$. This is caused by the fact that $|\mathbf{v}(\boldsymbol{x}_P)|_2$ measures the magnitude at a discrete point. Although the averaged velocity magnitude around Point P decreases, at the Point it remains the same.

decreasing h. Several elements away from the fluid domain, the fluid forces can be neglected and the momentum equation can be simplified to $-\nabla p = (\overline{D} + \overline{F}U)\mathbf{v}$. As the pressure gradient has the same magnitude in the solid domain for different values of q and h, the velocity magnitude decreases with the magnitude of $\overline{D} + \overline{F}U$. For the D approach, the magnitude scales as $|\mathbf{v}(\boldsymbol{x}_P)|_2 \propto \overline{D}^{-1}$, while for the DFF approach as $|\mathbf{v}(\boldsymbol{x}_P)|_2 \propto (\overline{D} + \overline{F}U)^{-1}$. For $Re_{in} = 50$ and $h = L/192$, in both the D and DFF approach, the Darcy penalization is the same and is dominant for the DFF approach, resulting in the same $|\mathbf{v}(\boldsymbol{x}_P)|_2$. However, for $Re_{in} = 500$, the Darcy penalization is lower for the DFF approach and the flow in the solid domain is less restricted. To conclude, in both the D and DFF approach, the velocity magnitude in the solid domain, several elements removed from the fluid domain, may significantly decrease. However, when comparing neighboring fluid and solid domain elements, the flow reduction remains predictable for differing values of h.

3.B. FILTER SIZE FOR THE DFF APPROACH

We investigate the effect of the filter size R on the flow reduction in the DFF approach. A predictable flow reduction is desirable as the continuation approach in Section 3.5 allows for more leakage and design flexibility in the first design iterations, and less leakage resulting in accurate and crisp but less flexible designs in the final design iterations. We require the filter to be large enough such that the filtered velocity magnitude U in solid elements accurately represents the velocity magnitude in neighboring fluid elements, i.e., $U \approx v^f$. For $Re_e^f > 1$, underestimating the fluid velocity magnitude $U \ll v^f$, leads to an underestimated Forchheimer magnitude and results in increased flow leakage, $v^s/v^f > 10^{-q}$. Overestimation $U \gg v^f$ leads to an overestimated Forchheimer magnitude, resulting in reduced flow leakage, $v^s/v^f < 10^{-q}$. We intend to find the appropriate filter radius $R = Nh/(2\sqrt{3})$, which determines the distance over over which information is distributed. The radius is defined using N, the number of elements of size h in radius R (Lazarov & Sigmund, 2011).

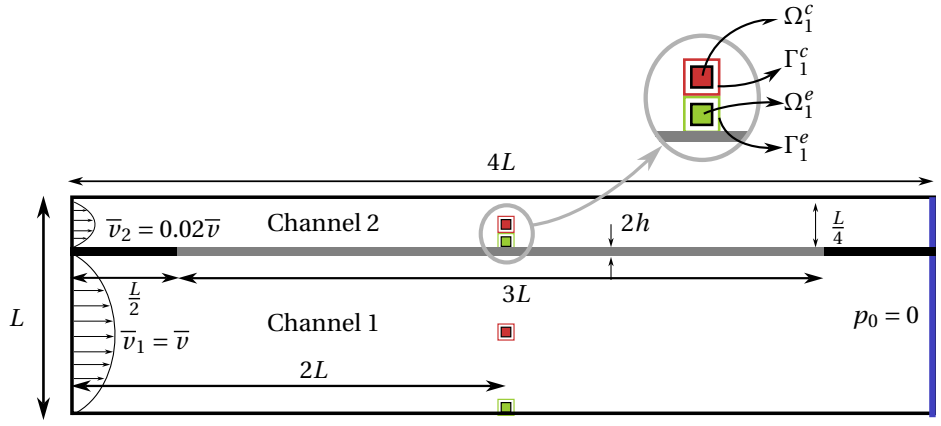


Figure 3.27: Design for testing the effect of different filter radii R on the predictability of the flow reduction in the DFF approach. The design is split into Channel 1 at the bottom with relatively large velocity magnitudes, and Channel 2 at the top with relatively low velocity magnitudes. The separating wall consists of a solid part (black), where velocity is fixed as $\mathbf{v} = \mathbf{0}$, and a density-based part (gray), where flow is inhibited such that $v^s \ll v^f$. At the midpoint of both channels, density-based obstacles of size $2h$ by $2h$ are placed at the edge (Ω_i^e in green) and in the center (Ω_i^c in red). Separated by one element from the obstacles, boundaries Γ_i^c and Γ_i^e are defined to measure the velocity magnitudes in neighboring fluid elements.

The design in Figure 3.27 is used to inspect flow leakage for several filter radii. Two channels separated by a two-element thick porous wall are investigated. In Channel 1 at the bottom, flow speeds are high with a maximum inlet velocity of $\overline{v}_1 = \overline{v}$, in Channel 2 at the top, they are low with $\overline{v}_2 = 0.02\overline{v}$. We expect U to overestimate v^f in the top channel when the filter radius is too large, as U will be influenced by the large flow magnitude in the bottom channel. The parameters in Table 3.11 are used and we inspect two different mesh sizes, $h = \frac{L}{40}$ and $\frac{L}{80}$. A Reynolds dependent density ρ, in Equation 3.58, is used. We emphasize that the different mesh size influences the filter radius as it is dependent on a number of elements N. We thus compute the flow solution for varying N, and expect to find the appropriate N, independent of the mesh size h.

L	Re_{in}	μ	$\overline{v}$
1 [m]	4000	1[Nsm^{-2}]	10[ms^{-1}]

Table 3.11: Parameters used to measure the effect of the filter radius in flow leakage for the design in Figure 3.27.

To inspect the flow leakage, we introduce obstacles in the flow at the edges and in the centers of the channels. Flow leakage is computed using Equation 3.59. Edge and center obstacle have domains Ω_i^e and Ω_i^c, both with area A_Ω, and boundary Γ_i^e and Γ_i^c with length L_Γ^e and L_Γ^c, respectively. Beside the leakage, we inspect the accuracy in filtered flow magnitude U with respect to the measured fluid velocity magnitude:

$$\widetilde{U}_i^e = \frac{U}{v^f} \approx \frac{\int_{\Omega_i^e} U d\Omega}{\int_{\Gamma_i^e} |\mathbf{v}|_2 \, d\Omega} \frac{L_\Gamma^e}{A_\Omega}, \qquad \widetilde{U}_i^c = \frac{U}{v^f} \approx \frac{\int_{\Omega_i^c} U d\Omega}{\int_{\Gamma_i^c} |\mathbf{v}|_2 \, d\Omega} \frac{L_\Gamma^c}{A_\Omega} \tag{3.69}$$

For accuracy close to 1, the filtered velocity magnitude U accurately represents the fluid velocity magnitude v^f and the filtered Forchheimer penalization should lead to the expected flow leakage of $v^s/v^f = 10^{-q}$.

To measure the effect of the filter radius on the filtered Forchheimer penalization, we require it to be dominant and thus $Re_e^f \approx Re_e^s > 1$. In Figure 3.28, we show elemental Reynolds numbers in the channels for the most accurate flow solution computed using $q = 3$, $N = 16$, and $h = L/80$. In the fluid ($\alpha = 1$) and the solid ($\alpha = 0$) domain, elemental Reynolds numbers are computed as:

$$Re_e^f = \frac{\rho |\mathbf{v}|_2 \, h}{\mu}, \qquad Re_e^s = \frac{\rho U h}{\mu}. \tag{3.70}$$

Consequently, in Channel 1 we find elemental Reynolds numbers of $Re_e^f \approx 34$ at the center and $Re_e^f \approx 7.4$ at the edge obstacles, and in Channel 2, of $Re_e^f \approx 0.60$ and $Re_e^f \approx 0.39$, respectively. In bottom Channel 1, erroneous estimations of $U \approx v^f$ cause an inappropriate penalization and less predictable flow leakage. In top Channel 2, underestimating $U < v^f$, resulting in $Re_e^s < Re_e^f < 1$, renders the Darcy penalization dominant, which should accurately penalize the dominant viscosity in the fluid domain. However, overestimating $U > v^f$ may cause $Re_e^s > 1$, resulting in a dominant Forchheimer penalization with a larger magnitude than the appropriate Darcy penalization.

In Figure 3.29, the measured leakage and filtered velocity accuracy for the center obstacles can be found. At the center island in Channel 1, the accuracy $\widetilde{U}_1^c$ tends to 1 for filter radii containing more elements, which leads to a predictable flow reduction $v^s/v^f = 10^{-q}$. In Channel 2, the accuracy in filtered flow magnitude is overestimated for $N > 8$. The filter radius is too large and the flow velocity in bottom Channel 1 starts to influence the filtered velocity magnitude in top Channel 2. The overestimation of local flow speeds subsequently causes the flow leakage to decrease. The overestimation in the filtered velocity magnitude is more pronounced for $h = L/40$. This is caused by the fact that the center island is six elements away from the bottom channel for $h = L/40$ and 11 elements for $h = L/80$. A filter radius with less elements N thus results in the filter penetrating the bottom channel earlier for $h = L/40$ than $h = L/80$.

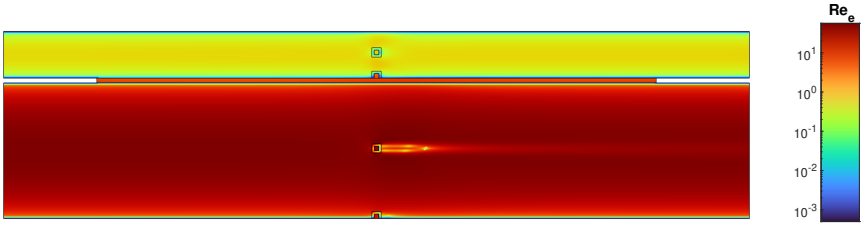


Figure 3.28: Elemental Reynolds numbers in the fluid and solid domains for $q = 3$, $N = 16$, and $h = L/80$.

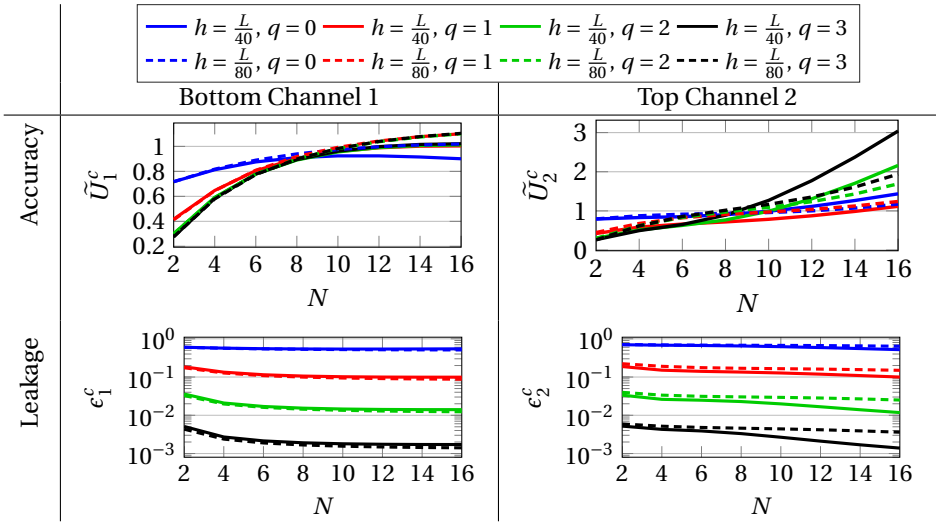


Figure 3.29: The flow leakage and filtered velocity magnitude accuracy for the center obstacles Ω_i^c.

For the edge obstacles, overestimation of fluid flow velocities due to large filter radii poses a larger problem, as shown in Figure 3.30. For the obstacle next to the solid bottom wall in Channel 1, increasing the filter radius past $N = 8$ elements results in an overestimation of the fluid flow velocity $\widetilde{U}_1^e > 1$. This causes the flow leakage to decrease below the expected value $v^s/v^f < 10^{-q}$. In top Channel 2, $\widetilde{U}_2^e$ increases even more drastically, but this does not cause a significantly larger flow reduction. When velocity magnitudes in the bottom channel dominate the filtered velocity in the top channel edge obstacle, the filtered velocity largely overestimates the local fluid velocity magnitude. However, flow in the edge obstacle is influenced by the bottom channel as flow passes through the porous wall into the edge obstacle. For the parabolic flow profile over the bottom inlet with maximum velocity $\overline{v}_1 = 10\ ms^{-1}$, the velocity magnitude at a distance h from the wall is theoretically 1.38 or 0.678 for $h = L/40$ or $h = L/80$, respectively. However, when we compute the average velocity magnitude around the edge obstacle in the top channel for $q = 3$, we find $0.29 > v^f > 0.19$ and $0.088 > v^f > 0.077$ for $h = L/40$ and

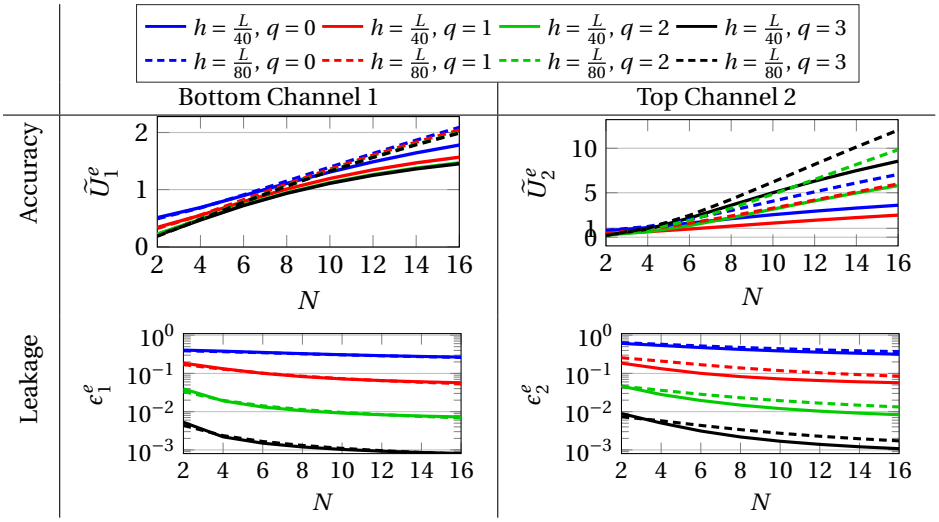


Figure 3.30: The flow leakage and filtered velocity magnitude accuracy for the edge obstacles Ω_i^e.

$h = L/80$, respectively. Forces in the Navier-Stokes equations scale with velocity magnitude. Fluid domain forces below the wall will thus be larger than those above the wall. Consequently, the flow through the wall and in the edge obstacle next to the wall will be mostly dependent on the velocity magnitude below the wall.

To determine the appropriate number of elements N in the filter, we compare the different flow leakage results. A larger N is expected to distribute sensitivities more equally between the fluid and solid domain and to improve design convergence. Small changes may drastically alter the nonlinear flow solution and objective, smoothing out sensitivities and making design changes more gradual can improve design convergence. We thus choose the largest N for which no significant overpenalization occurs. Moreover, in the choice of filter radius, the results in Figure 3.30 in top Channel 2 are neglected. The velocity magnitude in a solid domain is always determined by the largest velocity magnitude in the adjacent fluid domain. Consequently, overestimation of U only matters compared to the largest fluid domain velocity magnitude. Which is one of the major drawbacks of all penalization approaches presented in this paper. In two fluid areas separated by a thin solid domain, flow leaking from an area with high flow velocity to an area with low flow velocity significantly disturbs the flow in the latter area. In bottom Channel 1 in Figure 3.30, a small amount of overpenalization occurs for $N = 10$, in Figure 3.29, $N = 10$ results in an appropriate penalization. In this work, we thus use $N = 10$ to determine the filter radius for averaging the velocities.

3.C. INSTABILITY OF THE DARCY WITH FORCHHEIMER APPROACH

During optimization, we often found the DF approach to become unstable due to large fluctuations in design and the forward solution diverging. In this section, we investigate

the cause of this instability in the DF approach and motivate our preference for the DFF approach.

The forward solve is a computationally expensive and time consuming part of the optimization process. The overall optimization time can be significantly reduced by decreasing the time spent on the forward solve. In this work, we use the forward solution in the previous design as the initial guess for the solution in the current design. However, reusing the previous flow solution may lead to instabilities, particularly in the DF approach. When solid material is introduced in areas previously filled with fluid, the initial guess based on this previous design significantly overestimates the flow magnitude in the current design, which may lead to instabilities in the forward solve. Consequently, instabilities are amplified by the larger design fluctuations more often found when optimizing using the DF approach than using the DFF approach. Moreover, as shown in this section, for the DFF approach this does not lead to diverging flow solutions, contrary to the DF approach.

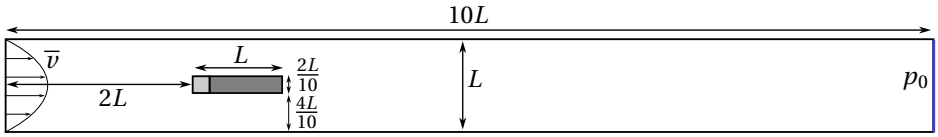


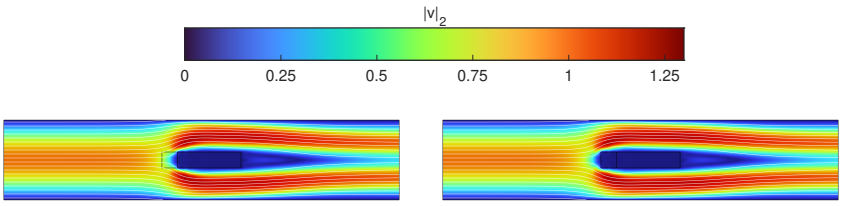
Figure 3.31: The fin in a channel used to investigate forward solve instabilities. In the fluid domain (white), no penalization is present and in the solid domain (dark gray), the maximum penalization is present. At the tip of the fin, we change the fluid volume fraction α and thus the penalization in the two-by-two element sized light gray domain.

To examine the effect of reusing the flow solution when sudden changes in design happen, we use the problem in Figure 3.31. A thin solid fin is inserted in the middle of a flow channel with parabolic flow inlet and constant pressure outlet. We use the parameters in Table 3.12 and the Reynolds-dependent density from Equation 3.58. The design is changed by extending the solid fin upstream, such that solid material is added in an area with large velocity magnitudes. We first solve the problem using the short fin, and subsequently reuse this flow solution to solve for the extended fin. We expect instabilities to be mainly caused by large elemental Reynolds numbers. A large $Re_{in} = 1000$ is thus used to ensure the large Re_e^s found in Figure 3.33a for the DF approach.

L	μ	$\overline{v}$	p_0	Re_{in}	q	h
1 [m]	1 [Nsm^{-2}]	1 [ms^{-1}]	0 [Pa]	1000	2	$\frac{L}{10}$

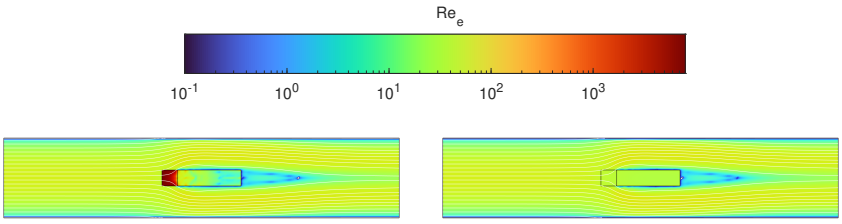
Table 3.12: Parameters used to investigate the instability of the DF approach using the problem in Figure 3.31.

While the forward solution of the extended fin in Figure 3.32 converges using the DFF approach, it diverges using the DF approach. When initializing using the flow solution of the problem without extended tip, the elemental Reynolds number for the DF approach in the extended tip is high relative to the elemental Reynolds number in the surrounding fluid, as shown in Figure 3.33. For the DFF approach the elemental Reynolds number varies more smoothly. As described in Section 3.2.4, the fluid domain elemental



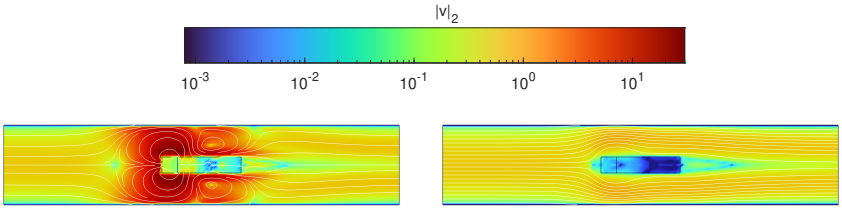
(a) Without extended fin. **(b)** With extended fin.

Figure 3.32: The flow magnitude $|\mathbf{v}|_2$ and flow lines found using the DFF approach in the first half of the channel.



(a) DF approach **(b)** DFF approach

Figure 3.33: Flow lines and elemental Reynolds number in the first forward iteration of the extended fin problem. In the fluid domain we plot $Re_e^f = (\rho\,|\mathbf{v}|_2\,h)/\mu$, while in the solid domain we plot $Re_e^s = 10^q(\rho\,|\mathbf{v}|_2\,h)/\mu$ for the DF approach and $Re_e^s = (\rho U h)/\mu$ for the DFF approach.



(a) DF solution. **(b)** DFF solution.

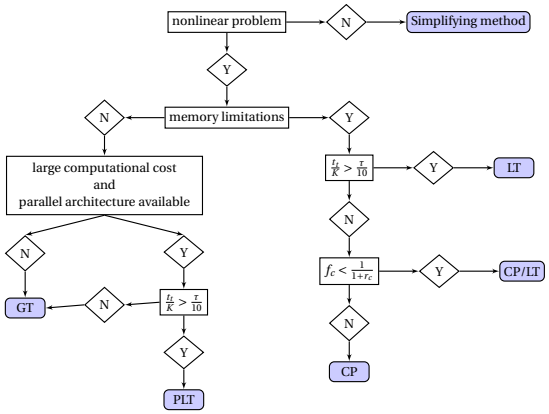
Figure 3.34: Velocity magnitude $|\mathbf{v}|_2$ and flow lines in the extended fin problem after performing one nonlinear solution iteration. Instead of decreasing the relatively high flow in the tip, the flow around the tip is significantly increased.

Reynolds number $Re_e^f = (\rho v^f h)/\mu$ should be similar to neighboring solid domain elemental Reynolds number $Re_e^s = 10^q(\rho v^s h)/\mu$. To compensate for the jump in elemental Reynolds number in the DF approach, the fluid domain elemental Reynolds number is increased by increasing the fluid domain velocity magnitude v^f in the first forward iteration, as shown in Figure 3.34a. Subsequently, the flow solution does not stabilize. The abrupt change in elemental Reynolds number, caused by reusing the state solution after a sudden design change, destabilizes the forward solution in the DF approach. Contrarily, using the DFF approach, the velocity magnitude is significantly decreased in the solid tip after the first forward iteration and velocity magnitudes in the surrounding fluid area

are not drastically increased, as shown in Figure 3.34b. This results in convergence of the forward solution even when initializing using an inaccurate initial guess in the DFF approach.

4

APPROACHES FOR TRANSIENT TOPOLOGY OPTIMIZATION



The limitations caused by memory requirements in gradient-based TO of transient problems are examined for thermal and flow problems. In addition, as TO of transient problems is time consuming, a novel approach for parallelization is introduced. Guidelines for selecting an appropriate algorithm for TO of transient problems are provided.

This chapter is based on the publication in International Journal for Numerical Methods in Engineering 125(14), Theulings et al. (2024).

Reducing time and memory requirements in topology optimization of transient problems

Abstract *In topology optimization of transient problems, memory requirements and computational costs often become prohibitively large due to the backward-in-time adjoint equations. Common approaches such as the Checkpointing (CP) and Local-in-Time (LT) algorithms reduce memory requirements by dividing the temporal domain into intervals and by computing sensitivities on one interval at a time. The CP algorithm reduces memory by recomputing state solutions instead of storing them. This leads to a significant increase in computational cost. The LT algorithm introduces approximations in the adjoint solution to reduce memory requirements and leads to a minimal increase in computational effort. However, we show that convergence can be hampered using the LT algorithm due to errors in approximate adjoints.*

To reduce memory and/or computational time, we present two novel algorithms. The Checkpointing/Local-in-Time (CP/LT) algorithm improves the convergence behavior of the LT algorithm at the cost of an increased computational time but remains more efficient than the CP algorithm. The Parallel-Local-in-Time (PLT) algorithm reduces the computational time through a temporal parallelization in which state and adjoint equations are solved simultaneously on multiple intervals. State and adjoint fields converge concurrently with the design. The effectiveness of each approach is illustrated with two-dimensional density-based topology optimization problems involving transient thermal or flow physics. Compared to the other discussed algorithms, we found a significant decrease in computational time for the PLT algorithm. Moreover, we show that under certain conditions, due to the use of approximations in the LT and PLT algorithms, they exhibit a bias towards designs with short characteristic times. Finally, based on the required memory reduction, computational cost, and convergence behavior of optimization problems, guidelines are provided for selecting the appropriate algorithms.

4.1. INTRODUCTION

Topology optimization is used to find the optimal material distribution for problems involving various types of physics such as mechanics, thermal, fluidics, magnetics, and many more. In the work by Alexandersen and Andreasen (2020), they found that much attention is given to static problems within the vast literature on topology optimization, while few works focus on solving transient optimization problems. However, there is often a need for considering transient effects, e.g., when designing thermally actuated compliant mechanisms (Y. Li et al., 2004), transient heat conducting devices (S. Wu et

al., 2019), or fluid pumps (Nørgaard et al., 2016).

Gradient-based algorithms such as the method of moving asymptotes (MMA) by Svanberg (1987) are often used to solve optimization problems such as topology optimization. To compute the required gradients, adjoint sensitivity calculations are commonly used due to their computational efficiency for problems with many design variables. For static problems, first the state equations pertaining to the optimization problems physics are solved. Secondly, the adjoint equations are solved resulting in the adjoint variables which are combined with the state solution to compute the sensitivities. A transient adjoint sensitivity computation follows the same procedure but has a larger computational cost and requires more memory because the state solutions are required to compute the sensitivities. During the *forward-in-time* solve of the state equations the solutions are thus stored for every discrete time step. Subsequently, the adjoint equations are solved and the adjoint solution is combined with the state solution to compute the sensitivities. However, as discussed by Haftka (1981), the adjoint equations are a terminal value problem and need to be solved *backward-in-time*. A transient adjoint sensitivity analysis may lead to prohibitively large memory requirements as the state solution needs to be stored for every time step, and to large computational cost since solving the transient adjoint equations is often as expensive as solving the transient state equations. The described algorithm for transient adjoint sensitivity computation will be referred to as the Global-in-Time (GT) algorithm as it solves the adjoint and state equations in the whole (global) time domain at once. In this paper we investigate algorithms to reduce memory requirements while keeping computational cost low.

A relatively straightforward method to reduce memory requirements for transient optimization problems is the method of equivalent static loads (ESLs). Experienced designers are able to make heuristic approximations or create simple computations of dynamic peak loads and consequently optimize dynamic structures using ESLs representing these peak loads (Kang et al., 2001). Advancements on these ad-hoc design practices have been made by Choi and Park (2002) by computing the ESLs of every displacement field computed in a transient analysis, where the ESLs at each time step is constructed such that it results in the exact displacement at that time step. Subsequently, these ESLs are used in a static optimization containing many loads. Memory requirements may become substantial computing and storing an ESL of the same size as the state vector at each time step. Transient problems with quickly decaying transient state solutions, referred to as *stiff* problems, may be optimized using ESLs as the transient part of the state response may be neglected. An example of such a problem would be the dynamic optimization of a system which is subject to high stiffness and relatively low inertia effects. Although ESLs are useful for many simple linear design applications, transient effects are ignored and when a complex transient system is optimized the ESL approach may become inaccurate. In these optimization problems the transient response cannot be neglected but due to a regularity of the input and linearity of the state equations, model order reduction techniques may be used to reduce memory and/or computational cost. Methods based on model order reduction, such as those examined by Hooijkamp and Keulen (2018), are able to circumvent the backward-in-time computation of the adjoint equations or reduce the amount of required storage (Qian, 2022). Frequency based modal reduction techniques have been used to reduce the amount of storage in

transient topology optimization but are limited by the computational cost of solving the eigenvalue problem (Q. Li et al., 2021), and have been shown by Zhao and Wang (2016) to become computationally worthwhile only when when many time steps are considered. However, when complex boundary conditions which change drastically over time or nonlinear systems are considered, constructing an accurate modal or proper orthogonal decomposition may prove cumbersome. For these systems, solving the complete state and adjoint solutions as in the GT algorithm is recommended. Characterizing the system to be optimized is crucial for the selection of the appropriate method for topology optimization of transient problems. In this work we focus on methods for topology optimization of transient problems which are nonlinear and/or subject to complex loads.

Several approaches tackling the large memory requirements of the GT algorithm have been proposed in literature. A well-known method to reduce data storage is the Checkpointing (CP) algorithm (Griewank, 1992; Griewank & Walther, 2000) in which the temporal domain is subdivided into multiple intervals. The algorithm consists of two steps. First, the forward state solution is computed but only stored at the interfaces between the intervals, i.e., the so-called checkpoints. Second, the state solution is computed and stored only for the final interval and the adjoint equation is propagated backward on this interval. Subsequently, the state solution on the final interval is removed from memory, the state solution on the second-to-last interval is computed again and stored on each time step and the adjoint variables are propagated further backward, and so on. Although the CP algorithm reduces memory requirements, it increases computational cost as the state equations are evaluated twice: for the computation of solutions on the checkpoints and for the backward propagation of the adjoint variables. The CP scheme was originally introduced by Griewank (1992) to find the sensitivity of iterative functions using automatic differentiation, where a binomial distribution of checkpoints was proposed which was proven to be optimal with respect to memory requirements by Grimm et al. (1996). Further developments of the CP algorithm by Heuveline and Walther (2006) and Q. Wang et al. (2009) have focused on problems where the number of time steps is not known *a priori*.

Another method to reduce memory requirements is the Local-in-Time (LT) algorithm proposed by Yamaleev et al. (2010). Similar to the CP algorithm, the LT algorithm divides the temporal domain into intervals but stores approximate adjoint solutions on the checkpoints instead of exact state solutions. The LT algorithm computes the sensitivity contribution of one interval at a time. To compute the sensitivity contribution of an interval, first the state equations on the interval are solved forward-in-time starting with an exact initial condition. Subsequently, the adjoint equations are solved backward-in-time starting from an approximate terminal adjoint solution for the interval. The sensitivity contribution is computed by combining adjoint and state solutions. Starting at the exact terminal state solution for the current interval the exact state solution can be computed in the next interval. Using an approximate terminal adjoint in the next interval, adjoint solutions can be evaluated and combined with the state solution to compute the sensitivities. To perform the sensitivity analysis, the full state solution needs to be stored for only one interval at a given time. Moreover, by updating the approximate adjoint solutions at the checkpoints in each design iteration, the LT algorithm is able to converge to an optimum while simultaneously converging the approximate adjoints to

the exact adjoints. However, since approximate solutions are used for the adjoint equations, the LT algorithm computes *approximate* sensitivities in contrast to the GT and CP algorithms which compute *exact* sensitivities. The main advantage of the LT algorithm is its computational cost. As the LT algorithm only solves the state equations once on every interval, it has a computational cost comparable to the GT algorithm and lower than the CP algorithm. Three dimensional topology optimization problems have been successfully tackled using the LT algorithm by Chen et al. (2017) and Yaji et al. (2018) but further research on the effect of approximate adjoint fields on the convergence behavior of the optimization process and the obtained optimal solution is needed.

Simultaneously reducing the error on approximate adjoint variables and converging to the optimal design share similarities with multiple shooting algorithms for optimal control (Carraro & Geiger, 2015). Such algorithms split the temporal domain into intervals to which are attached both approximate terminal adjoint and approximate initial state solutions. Subsequently, control problems are optimized by simultaneously solving for the control variables, approximate state, and approximate adjoint variables. Moreover, using approximate state and adjoint solutions, all temporal intervals can be decoupled and parallelized in time (Fang et al., 2022). In comparison, common practice for parallel speedup in topology optimization is to parallelize via domain decomposition (Borrvall & Petersson, 2002; Kristiansen & Aage, 2022). In domain decomposition, state equations are solved by splitting the spatial domain into several subdomains and performing computations on these domains in parallel (Mahdavi et al., 2006). A limitation in these methods is that adjacent domains share certain degrees of freedom (DOFs) and thus need to communicate with each other. If many subdomains and processors are used, a substantial amount of time is spent on communication, as shown by Aage et al. (2008) and Mahdavi et al. (2006). However, when parallelization via domain decomposition saturates, parallel-in-time algorithms often offer opportunities for further parallelization (Gander, 2015).

In this work we focus on methods which balance memory requirements and computational cost for topology optimization of transient problems which are nonlinear or subject to complex loads. We investigate the limitations of the CP and LT algorithms in terms of computational cost and convergence behavior, and propose two novel algorithms:

- the hybrid Checkpointing/Local-in-Time (CP/LT) algorithm which introduces an error measure and corrections for the LT algorithm,
- the Parallel-Local-in-Time (PLT) algorithm which parallelizes the optimization process by decomposing the time domain.

The new CP/LT algorithm addresses errors in the LT algorithm while keeping computational cost to a minimum. The PLT algorithm divides the temporal domain into parallel intervals and performs the computations on all these intervals at once while keeping computational overhead to a minimum by carrying out all parallel communication at once and only communicating between adjacent intervals. The computational cost, memory requirements, and limitations of the proposed algorithms are evaluated and compared to the state of the art on two-dimensional density-based topology optimiza-

tion problems: a transient heat conductor and a transient fluid pump. Both are discretized using the finite volume method (Versteeg & Malalasekera, 2007).

The remainder of this article is organized as follows. In Section 4.2, the exact methods for transient sensitivity analysis are introduced. A general approach for solving the transient state/adjoint equations and combining the solutions into the sensitivity computations is given and the CP algorithm is introduced. In Section 4.3, approximate algorithms for transient sensitivity analysis are introduced. Subsequently, in Section 4.3.1, we derive the LT algorithm from these general equations and identify some issues concerning stability and convergence. In Section 4.3.2, the novel CP/LT algorithm is proposed to address these stability and convergence issues. The final novel PLT algorithm which efficiently parallelizes the transient problem is developed in Section 4.3.3. The theoretical memory requirements and computational cost of the GT, CP, LT, CP/LT, and PLT algorithms are analyzed in Section 4.4, after which the algorithms are compared and evaluated using actual optimization examples in Section 4.5. Stability and convergence of the approximate algorithms are investigated in Section 4.5.1 using a thermal transient optimization problem and computational time is compared in Section 4.5.2 using a transient flow optimization problem. Finally, guidelines for algorithm selection are given in Section 4.6 and a discussion and conclusion on the results and algorithms for transient sensitivity analysis are provided in Section 4.7.

4.2. EXACT METHODS FOR SENSITIVITY ANALYSIS OF TRANSIENT PROBLEMS

In this work we focus on methods for topology optimization of transient problems which are nonlinear or subject to complex transient loads. Optimization is performed using gradient-based algorithms which require the sensitivities of the objective and constraints with respect to the design variables. We are thus interested in methods which aim to compute discrete adjoint sensitivities such as the GT, CP, and LT algorithms. In this section we investigate the GT and CP algorithms which compute exact discrete sensitivities.

4.2.1. DISCRETE TRANSIENT SENSITIVITIES AND THE GLOBAL-IN-TIME ALGORITHM

A generic topology optimization problem for transient physics on spatial domain $\vec{x} \in \Omega$ with boundary $\Gamma = \overline{\Omega} \setminus \Omega$, where $\overline{\Omega}$ is the closure of Ω, is defined as:

$$
\begin{array}{lll}
\underset{s(\vec{x})}{\text{minimize}} & F = \displaystyle\int_{t=0}^{t=t_t} \int_{\Omega} f\left(u(\vec{x},t), s(\vec{x})\right) d\Omega dt & \\
\text{subject to} & R\left(u(\vec{x},t), s(\vec{x}), t\right) = 0, & t \in [0,t_t],\ \vec{x} \in \Omega, \\
& R_{\Gamma}\left(u(\vec{x},t), \partial u(\vec{x},t)/\partial \vec{x}, s(\vec{x}), \vec{x}, t\right) = 0, & t \in [0,t_t],\ \vec{x} \in \Gamma, \\
& u\left(\vec{x}, t=0\right) = \hat{u}(\vec{x}), & \vec{x} \in \Omega, \\
& g_i\left(s(\vec{x})\right) \leq 0, & i \in I,\ \vec{x} \in \Omega,
\end{array}
\tag{4.1}
$$

where t_t is the terminal time, $u(\vec{x},t)$ is the time dependent state solution with $\hat{u}(\vec{x})$ as initial condition, $s(\vec{x})$ is the design field, $R(u(\vec{x},t),s(\vec{x}),t)$ is the Partial Differential Equation (PDE) constraint with boundary conditions $R_\Gamma(u,\partial u/\partial\vec{x},s,\vec{x},t)$ defined on boundary Γ, and $g_i(s(\vec{x}))$ is the i^{th} inequality constraint in set I. Subsequently the continuous optimization problem is discretized in space and time as:

$$\begin{aligned} \underset{\boldsymbol{s}}{\text{minimize}} \quad & F \approx \sum_{n=0}^{N} F_n(\boldsymbol{u}_n,\boldsymbol{s})\Delta t \\ \text{subject to} \quad & R_n(\boldsymbol{u}_{n-1},\boldsymbol{u}_n,\boldsymbol{s},t_n)=0, \quad n\in\{1,2\dots N\}, \\ & R_0(\boldsymbol{u}_0,\boldsymbol{s},t_0)=0, \\ & g_i(\boldsymbol{s})\le 0, \qquad i\in I \end{aligned} \tag{4.2}$$

where the full temporal domain $t\in[0,t_t]$ is divided into N time steps of length $\Delta t = t_t/N$, $\boldsymbol{u}_n$ is the column containing discretized state variables at time t_n, all discrete design variables are gathered in column $\boldsymbol{s}$, the continuous integral objective is discretized at time step n using $F_n\Delta t$, the column $R_n(\boldsymbol{u}_{n-1},\boldsymbol{u}_n,\boldsymbol{s})$ contains all discretized PDE constraints and boundary conditions at time t_n, $R_0(\boldsymbol{u}_0,\boldsymbol{s},t_0)$ contains the initial conditions, and $g_i(\boldsymbol{s})$ is the i^{th} inequality constraints on $\boldsymbol{s}$. Furthermore, the discretized time step $R_n(\boldsymbol{u}_{n-1},\boldsymbol{u}_n,\boldsymbol{s},t_n)$ accounts for several discrete integration techniques such as backward/forward Euler, Crank-Nicolson, and several of the Runge-Kutta methods. The equations derived in this work are thus valid for these methods which can be described using $R_n(\boldsymbol{u}_{n-1},\boldsymbol{u}_n,\boldsymbol{s},t_n)$. To compute the sensitivities of the PDE constrained objective, an augmented objective F^* is constructed,

$$F^* = F_0(\boldsymbol{u}_0,\boldsymbol{s})\Delta t + \boldsymbol{\lambda}_0^\intercal R_0(\boldsymbol{u}_0,\boldsymbol{s},t_0) + \sum_{n=1}^{N}\left(F_n(\boldsymbol{u}_n,\boldsymbol{s})\Delta t + \boldsymbol{\lambda}_n^\intercal R_n(\boldsymbol{u}_{n-1},\boldsymbol{u}_n,\boldsymbol{s},t_n)\right), \tag{4.3}$$

where the constraints are introduced using adjoint variables $\boldsymbol{\lambda}_n$ at time t_n. Subsequently, to construct the adjoint equations, the sensitivities are derived as:

$$\begin{aligned} \frac{dF^*}{d\boldsymbol{s}} = &\sum_{n=0}^{N-1}\left(\frac{\partial F_n}{\partial \boldsymbol{s}}\Delta t + \boldsymbol{\lambda}_n^\intercal\frac{\partial R_n}{\partial \boldsymbol{s}} + \left(\frac{\partial F_n}{\partial \boldsymbol{u}_n}\Delta t + \boldsymbol{\lambda}_n^\intercal\frac{\partial R_n}{\partial \boldsymbol{u}_n} + \boldsymbol{\lambda}_{n+1}^\intercal\frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n}\right)\frac{\partial \boldsymbol{u}_n}{\partial \boldsymbol{s}}\right) \\ &+ \frac{\partial F_N}{\partial \boldsymbol{s}}\Delta t + \boldsymbol{\lambda}_N^\intercal\frac{\partial R_N}{\partial \boldsymbol{s}} + \left(\frac{\partial F_N}{\partial \boldsymbol{u}_N}\Delta t + \boldsymbol{\lambda}_N^\intercal\frac{\partial R_N}{\partial \boldsymbol{u}_N}\right)\frac{\partial \boldsymbol{u}_N}{\partial \boldsymbol{s}}, \end{aligned} \tag{4.4}$$

where we drop the dependencies of R_n on $\boldsymbol{u}_{n-1}$, $\boldsymbol{u}_n$, $\boldsymbol{s}$, and t_n for brevity. To avoid the computation of matrix $\partial\boldsymbol{u}_n/\partial\boldsymbol{s}$, we set all sums multiplied by $\partial\boldsymbol{u}_n/\partial\boldsymbol{s}$ to zero, resulting in the adjoint equations:

$$n = N : \boldsymbol{\lambda}_N = -\frac{\partial R_N}{\partial \boldsymbol{u}_N}^{-\intercal}\frac{\partial F_N}{\partial \boldsymbol{u}_N}^{\intercal}\Delta t, \tag{4.5a}$$

$$n\in\{N-1,N-2,\dots 0\} : \boldsymbol{\lambda}_n = -\frac{\partial R_n}{\partial \boldsymbol{u}_n}^{-\intercal}\left(\frac{\partial F_n}{\partial \boldsymbol{u}_n}^{\intercal}\Delta t + \frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n}^{\intercal}\boldsymbol{\lambda}_{n+1}\right). \tag{4.5b}$$

Since the adjoint equations only have a terminal condition for $\boldsymbol{\lambda}_N$, they can only be solved backward-in-time starting at $t_N = t_t$ taking backward steps until $t_0 = 0$. A notable property of the backward-in-time adjoint equations is the fact that they resemble

the forward-in-time state equations. If we assume that R_n is a linear equation with respect to $\boldsymbol{u}_{n-1}$ and $\boldsymbol{u}_n$, the forward-in-time state equations may be written as the solution of:

$$n = 0 : \boldsymbol{u}_0 = \hat{\boldsymbol{u}}_0, \tag{4.6a}$$

$$n \in \{1, 2, \dots N\} : \boldsymbol{u}_n = -\frac{\partial R_n}{\partial \boldsymbol{u}_n}^{-1} \left(\boldsymbol{q}_n + \frac{\partial R_n}{\partial \boldsymbol{u}_{n-1}} \boldsymbol{u}_{n-1} \right), \tag{4.6b}$$

where $\hat{\boldsymbol{u}}_0$ are the discretized initial conditions and $\boldsymbol{q}_n$ is the part of R_n independent of the state variables. Since the backward-in-time adjoint equations mirror the forward-in-time state equations, we can assume that the same stability and convergence criteria hold for both equations. However, for nonlinear $R_n(\boldsymbol{u}_{n-1}, \boldsymbol{u}_n, \boldsymbol{s}, t_n)$ there is an important difference between the state and adjoint equations. As the adjoint equations are a linearized version of the state equations, solving the adjoint equations may take significantly less computational work and time than solving nonlinear state equations. At best both the state and adjoint equations are linear and require the same amount of computational work.

After solving the adjoint equations backward-in-time, the adjoint variables can be combined with the stored state equations to compute the sensitivities:

$$\frac{dF}{d\boldsymbol{s}} = \sum_{n=0}^{N} \left(\frac{\partial F_n}{\partial \boldsymbol{s}} \Delta t + \boldsymbol{\lambda}_n^{\mathsf{T}} \frac{\partial R_n}{\partial \boldsymbol{s}} \right). \tag{4.7}$$

To compute the sensitivities, two types of equations need to be solved; the state equations pertaining to the physics, and the adjoint equations. The most straightforward method to solve these equations is the GT algorithm which first solves the state equations forward-in-time and stores the state solution at every time step. Subsequently, the adjoint equations are solved backward-in-time using Equation 4.5 while simultaneously updating the sensitivities using Equation 4.7. As the sensitivities depend on $\partial R_n / \partial \boldsymbol{s}$ which in turn is dependent on $\boldsymbol{u}_n$ and $\boldsymbol{u}_{n-1}$, the state solution is thus required while propagating the adjoint solution backward to computed the sensitivities. The full state solution is thus stored during the forward solve and the main restriction in using the GT algorithm is the large memory requirement. For a problem with a state solution $\boldsymbol{u}_n$ of size m and involving N time steps, memory requirements M scale proportionally as $M \propto mN$. To compare the computational cost of the following algorithms in Sections 4.2.2 and 4.3 we evaluate the cost of the GT algorithm. We define the cost of solving one state step $R_n(\boldsymbol{u}_{n-1}, \boldsymbol{u}_n, \boldsymbol{s}, t_n)$ as c_s and the cost of solving one adjoint step using Equation 4.5 and consequently updating sensitivities using Equation 4.7 as c_a. The cost of the adjoint step and updating sensitivities is combined as this is a practical implementation in code. For highly nonlinear systems the cost of solving an adjoint step is much cheaper and we define ratio $r_c = c_a / c_s$. As the GT algorithm solves a state/adjoint step only once for all N time steps, the computational cost C scales proportionally as $C \propto Nc_s(1 + r_c)$.

4.2.2. THE CHECKPOINTING ALGORITHM

To reduce memory requirements the CP algorithm (Griewank, 1992; Griewank & Walther, 2000) may be used. The algorithm subdivides the temporal domain into K discrete intervals of length $\Delta\theta = t_t / K$, resulting in $K + 1$ checkpoints θ_k, where each θ_k is

assumed to correspond to one of the discrete times t_n but not all t_n have a corresponding θ_k. The K intervals are thus defined as $\Theta_k = [\theta_k, \theta_{k+1}]$, and contain all discrete times $\{n | t_n \in [\theta_k, \theta_{k+1}]\}$, as illustrated in Figure 4.1. The Subscript k is used to denote checkpoints and intervals. Furthermore, we use state/adjoint variables $\boldsymbol{U}_k / \boldsymbol{\Lambda}_k$ as the variables at checkpoint k where $\boldsymbol{U}_k = \boldsymbol{u}_n$ and $\boldsymbol{\Lambda}_k = \boldsymbol{\lambda}_n$ at $\theta_k = t_n$, respectively. These variables and subscript k are introduced to clearly describe and visualize the algorithms presented in this paper as will be shown for the CP algorithm in Figure 4.2. To compute sensitivi-

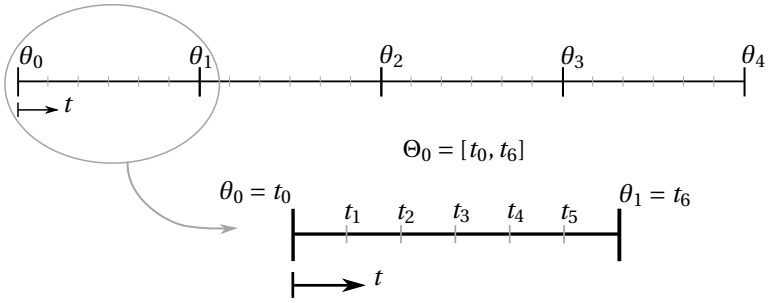


Figure 4.1: A temporal domain discretized using $N = 24$ time steps, and subdivided into $K = 4$ intervals $\Theta_k = [\theta_k, \theta_{k+1}]$, where θ_k are the temporal checkpoints. Each interval Θ_k contains time steps $\{n | t_n \in [\theta_k, \theta_{k+1}]\}$. For instance, the first interval Θ_0 is defined as $\Theta_0 = [t_0, t_6]$, and contains discrete time steps $n \in \{0, 1, 2, 3, 4, 5, 6\}$.

ties, the CP algorithm first computes the full forward state solution and stores only $\boldsymbol{U}_k$ at checkpoints θ_k. The full state solution is only stored for the last interval Θ_{K-1} and adjoint equations are subsequently solved and used to update the sensitivities, after which the state solution on the terminal interval can be removed from memory. Next, the adjoint solution is propagated further backward by recomputing and storing the state solution at the final to last interval Θ_{K-2} from the stored state solution $\boldsymbol{U}_{K-2}$ at checkpoint θ_{K-2}. Thus by continuously recomputing the state solution on the previous interval, and removing used state solutions from memory, exact sensitivities can be computed while reducing the memory requirements. A schematic of the CP algorithm can be found in Figure 4.2.

Depending on the number of intervals K, memory may be greatly reduced. If an initial state solution $\boldsymbol{U}_k$ of size m is stored for each of the K intervals except the first where it is found by solving $R_0(\boldsymbol{u}_0, \boldsymbol{s}, t_0)$, this requires the storage of $m(K-1)$ discrete state variables. Additionally, on each of the intervals we require the storage of the full state solution which requires a memory of mN/K discrete state variables. Memory requirements thus scale as $M \propto m(K-1) + mN/K$, and are reduced with respect to the GT algorithm. To reduce the computational overhead, we recommend to use the minimal number of intervals allowed by the memory limitations. Other approaches for optimal memory reduction such as the binomial distribution of checkpoints are also proposed in literature by Grimm et al. (1996).

The reduced memory in the CP algorithm comes at a higher computational cost as

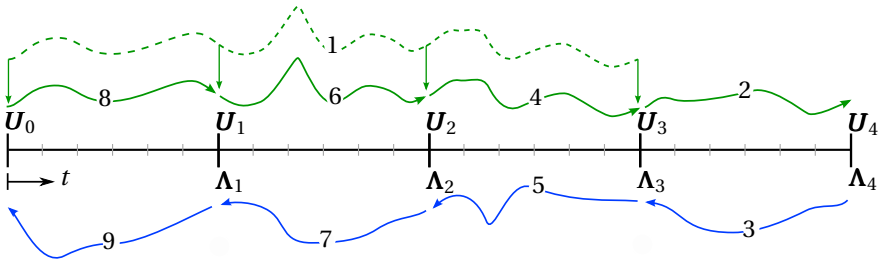


Figure 4.2: A schematic of the CP algorithm. The numbers represent the order of operations. The dashed green line represents the solving of the full forward state solution while only storing it at the checkpoints. A green arrow represents the computation and storage of state solutions on a complete interval. A blue arrow represents the backward computation of adjoint variables and update of sensitivities using the adjoint variables and stored state variables.

state solutions are recomputed on the first $K-1$ intervals of length N/K. Recomputation of the state solutions is associated to an additional cost proportional to $c_s(N/K)(K-1) = Nc_s(1-1/K)$. The computational cost of the CP algorithm is thus the cost of the GT algorithm ($C \propto Nc_s(1+r_c)$) with the addition of the recomputed state solutions, i.e., $C \propto Nc_s(2+r_c-1/K)$.

4.3. APPROXIMATE METHODS FOR SENSITIVITY ANALYSIS OF TRANSIENT PROBLEMS

Other approaches for the reduction of memory requirements and computational time are based on approximations of the state and/or adjoint variables. Generally, these methods rely on an iterative procedure to update approximate state and/or adjoint variables and the design until convergence. In a standard optimization procedure, design variables $\boldsymbol{s}^j$ at optimization iteration j are iteratively improved by computing sensitivities $dF^*/d\boldsymbol{s}^j$ and using a gradient-based optimizer. Exact solutions for all state and adjoint variables are computed. In the proposed approximate methods, state and/or adjoint equations are not satisfied at every iteration j. Approximate states $\tilde{\boldsymbol{U}}_k^j \approx \boldsymbol{U}_k^j$ and/or adjoint variables $\tilde{\boldsymbol{\Lambda}}_k^j \approx \boldsymbol{\Lambda}_k^j$ at checkpoints θ_k are updated using fixed point iterations. The idea is to let these variables converge to exact solutions simultaneously with the convergence of the design to the optimum, with the purpose of reducing memory requirements and/or computational time. Examples of such algorithms are the LT and multiple shooting type algorithms. In this section, we first introduce the LT algorithm and discuss its limitations. Subsequently, we propose some modifications to the LT algorithm to increase stability and introduce a novel multiple shooting type algorithm for topology optimization of transient problems.

4.3.1. THE LOCAL-IN-TIME ALGORITHM

To reduce memory requirements, the LT algorithm by Yamaleev et al. (2010) can be used. In the LT algorithm, adjoint variables are approximated and solved for iteratively. The algorithm computes the sensitivities successively on each interval Θ_k moving forward-in-time. Memory is reduced as the state solution is only stored on a single interval Θ_k at a time. However, errors are introduced as we approximate $\tilde{\boldsymbol{\Lambda}}_k^j \approx \boldsymbol{\Lambda}_k^j$ and solve for the exact adjoint over design iterations j. Furthermore, to initialize the LT algorithm approximate adjoint variables $\tilde{\boldsymbol{\Lambda}}_k^{j=0}$ are required and are set to $\tilde{\boldsymbol{\Lambda}}_k^{j=0} = \mathbf{0}$ as suggested by Yamaleev et al. (2010). The process to perform the complete sensitivity analysis is illustrated in Figure 4.3. The contribution of one interval Θ_k is computed following three steps:

1. **Computation and storage of the state solution** Firstly, the initial state solution $\boldsymbol{u}_n^j = \boldsymbol{U}_k^j$ at time $t_n = \theta_k$ is retrieved, which is either the initial value from $R_0(\boldsymbol{U}_0^j, \boldsymbol{s}, t_0) = \mathbf{0}$ at $k = 0$ or the terminal value $\boldsymbol{U}_k^j$ at time θ_k on the previous interval Θ_{k-1}. Starting from this initial state solution, the complete state solution is evaluated and stored forward-in-time until $\boldsymbol{U}_{k+1}^j$ at time θ_{k+1} is reached.

2. **Computation of approximate adjoint solution and sensitivities** We retrieve an approximate terminal adjoint variable $\tilde{\boldsymbol{\lambda}}_n^j = \tilde{\boldsymbol{\Lambda}}_{k+1}^j$ at time $t_n = \theta_{k+1}$ from memory, or when the terminal interval Θ_{K-1} is being investigated we compute the exact terminal adjoint $\boldsymbol{\lambda}_N^j = \boldsymbol{\Lambda}_K^j$ by solving Equation 4.5a. Subsequently, Equation 4.5b and the stored state solution $\boldsymbol{u}_n^j$ are used to propagate the approximate adjoint backward as:

$$\tilde{\boldsymbol{\lambda}}_n^j = -\left(\frac{\partial R_n}{\partial \boldsymbol{u}_n}\right)^{-\mathsf{T}} \left(\frac{\partial F_n}{\partial \boldsymbol{u}_n}\Delta t + \frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n}^{\mathsf{T}} \tilde{\boldsymbol{\lambda}}_{n+1}^j\right), \text{ for } n \in \{n | \theta_k + \Delta t \leq t_n < \theta_{k+1}\}, \tag{4.8}$$

   while simultaneously updating the sensitivities using Equation 4.7.

3. **Update and storage of approximate adjoint variables** After computing the sensitivities on the interval, we solve Equation 4.5b one final time to compute the approximate adjoint at the checkpoint at time $t_n = \theta_k$ for the next design iteration $j+1$, as:

$$\tilde{\boldsymbol{\lambda}}_n^{j+1} \approx -\left(\frac{\partial R_n}{\partial \boldsymbol{u}_n}\right)^{-\mathsf{T}} \left(\frac{\partial F_n}{\partial \boldsymbol{u}_n}\Delta t + \frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n}^{\mathsf{T}} \tilde{\boldsymbol{\lambda}}_{n+1}^j\right), \text{ for } n | t_n = \theta_k. \tag{4.9}$$

   The approximate adjoint $\tilde{\boldsymbol{\Lambda}}_k^{j+1} = \tilde{\boldsymbol{\lambda}}_n^{j+1}$ from Equation 4.9 is stored to be used as a terminal value at $t_n = \theta_k$ for interval Θ_{k-1} in the next design iteration $j+1$. We emphasize that within iteration j, adjoint vector $\tilde{\boldsymbol{\lambda}}_n^j = \tilde{\boldsymbol{\Lambda}}_k^j$ and not $\tilde{\boldsymbol{\lambda}}_n^{j+1} = \tilde{\boldsymbol{\Lambda}}_k^{j+1}$ at time $t_n = \theta_k$ is used to update the sensitivities using Equation 4.7.

After the evaluation of sensitivity contributions on interval Θ_k, the state solutions on the interval are removed from memory. Starting from the solution $\boldsymbol{u}_n^j = \boldsymbol{U}_{k+1}^j$, the sensitivities in domain Θ_{k+1} are computed as shown in the LT schematic in Figure 4.3. In the LT algorithm, approximate adjoint variables $\tilde{\boldsymbol{\lambda}}_n^j = \tilde{\boldsymbol{\Lambda}}_{k+1}^j$ are used to compute sensitivities on interval Θ_k. They are propagated backward-in-time and used to update $\tilde{\boldsymbol{\Lambda}}_k^{j+1} = G(\tilde{\boldsymbol{\Lambda}}_{k+1}^j)$

using Equations 4.8 and 4.9. Here, we define $G(\tilde{\mathbf{\Lambda}}^j_{k+1})$ as an operator which propagates adjoint variables backward-in-time over an interval as used in Figure 4.3, which we find to be a linear operator by inspecting Equations 4.8 and 4.9.

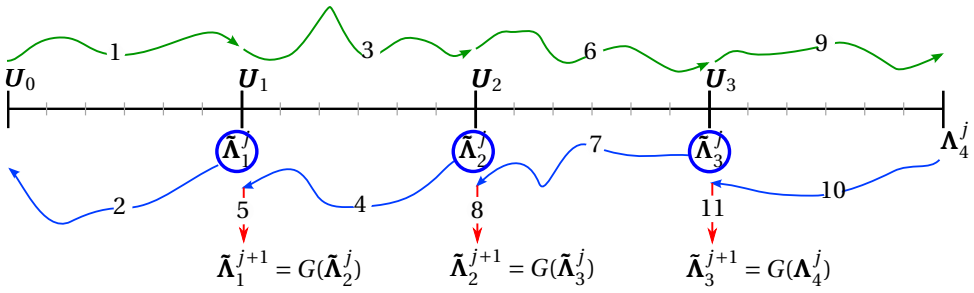


Figure 4.3: A schematic of the LT algorithm. The numbers represent the order of operations. A green arrow represents the computation and storage of state solutions on a certain interval. The blue arrow represents the computation of adjoint variables and update of sensitivities using the stored state and computed adjoint variables. Finally, the red arrow represents the update of the approximate adjoint variable.

It should be noted that the adjoint variables at time $t_n = t_N$ are always exact as they are computed using Equation 4.5a, and all adjoint solutions on terminal interval Θ_{K-1} are thus exact. Consequently, when adjoint variables have stabilized ($\tilde{\mathbf{\Lambda}}^j_k = \tilde{\mathbf{\Lambda}}^{j+1}_k = G(\tilde{\mathbf{\Lambda}}^j_{k+1})$), the exact variables from the terminal interval propagate backward and the approximate adjoint variables have converged to the exact variables found in the GT algorithm ($\tilde{\mathbf{\Lambda}}^j_k = \mathbf{\Lambda}^j_k$). The idea behind the LT algorithm is thus that approximate adjoint variables are improved by $\tilde{\mathbf{\Lambda}}^{j+1}_k = G(\tilde{\mathbf{\Lambda}}^j_{k+1})$ and converge to the exact values as the design converges to the optimum and stabilizes.

Memory requirements may be significantly reduced using the LT algorithm. Since the adjoint vector $\tilde{\mathbf{\Lambda}}_k$ and state vector $\boldsymbol{U}_k$ both have the same size m and the LT algorithm stores the approximate terminal adjoint variables on each of the K intervals except the last, $m(K-1)$ discrete approximate adjoint variables need to be stored. In addition, the complete state solution on one of the intervals at a time is stored which requires the storage of mN/K discrete state variables. The memory requirement of the LT algorithm thus scales as $M \propto m(K-1) + mN/K$, which is identical to the CP algorithm.

Whereas the CP algorithm reduces memory at the cost of increased computational time compared to the GT algorithm, the LT algorithm decreases memory requirements with no significant increase in computational time. As the state and adjoint solutions are solved only once per interval, the computational cost of the LT algorithm is exactly the same as the cost of the GT algorithm: $C \propto Nc_s(1+r_c)$. However, this does assume that the convergence of the optimization process using the LT algorithm is not negatively affected by the use of approximate adjoint solutions and sensitivity information.

STABILITY AND CONVERGENCE OF THE LOCAL-IN-TIME ALGORITHM

In the LT algorithm, it is assumed that as a design converges over multiple design iterations, the adjoint field will also converge to the exact adjoint field resulting in accurate sensitivities. Using these sensitivities, we ensure an accurate local minimum is found. The assumption is thus that upon stabilization of the design, the adjoint field stabilizes and $\boldsymbol{\Lambda}_k^j = \tilde{\boldsymbol{\Lambda}}_k^j = G(\tilde{\boldsymbol{\Lambda}}_{k+1}^j)$. To allow for both the design and adjoint fields to stabilize, two types of convergence are necessary: local convergence and global convergence. Local convergence relates to the decrease in the error of the approximate adjoint field on subdomain Θ_k. Assuming that, by backward-in-time propagation of $\tilde{\boldsymbol{\Lambda}}_k^{j+1} = G(\tilde{\boldsymbol{\Lambda}}_{k+1}^j)$, the error in $\tilde{\boldsymbol{\Lambda}}_k^{j+1}$ decreases with respect to the error in $\tilde{\boldsymbol{\Lambda}}_{k+1}^j$ on every interval Θ_k, we are able to deduce that upon a stable non-changing design $\boldsymbol{s}^j$ all approximation errors will reduce to zero. Global convergence relates to the simultaneous convergence of the design and approximate adjoint field.

Local convergence

Using a modal analysis of the backward-in-time adjoint equations, local convergence is investigated. We assume that the exact adjoint variables $\boldsymbol{\lambda}_n$ at time t_n can be expressed as the sum of the approximate adjoint variables and a correction $\Delta\boldsymbol{\lambda}_n$:

$$\boldsymbol{\lambda}_n = \tilde{\boldsymbol{\lambda}}_n + \Delta\boldsymbol{\lambda}_n. \tag{4.10}$$

Furthermore, we assume that the approximate adjoint variables approach the exact adjoint variables as found in the GT algorithm in a stable optimum such that $\tilde{\boldsymbol{\lambda}}_n \approx \boldsymbol{\lambda}_n$ and the adjoint correction approaches zero $\Delta\boldsymbol{\lambda}_n \to \mathbf{0}$. We thus investigate the evolution of the corrections by constructing a time stepping scheme for the corrections. In Equation 4.11a, the exact adjoint equation is repeated from Equation 4.5. The improved approximation $\boldsymbol{\lambda}_n^j = \tilde{\boldsymbol{\lambda}}_n^j + \Delta\boldsymbol{\lambda}_n^j$ is substituted into Equation 4.11a resulting in Equation 4.11b:

$$\boldsymbol{\lambda}_n^j = -\left(\frac{\partial R_n}{\partial \boldsymbol{u}_n^j}\right)^{-\mathsf{T}} \left(\frac{\partial F_n}{\partial \boldsymbol{u}_n^j}\Delta t + \frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n^j}^{\mathsf{T}} \boldsymbol{\lambda}_{n+1}^j\right), \tag{4.11a}$$

$$\tilde{\boldsymbol{\lambda}}_n^j + \Delta\boldsymbol{\lambda}_n^j = -\left(\frac{\partial R_n}{\partial \boldsymbol{u}_n^j}\right)^{-\mathsf{T}} \left(\frac{\partial F_n}{\partial \boldsymbol{u}_n^j}\Delta t + \frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n^j}^{\mathsf{T}} \tilde{\boldsymbol{\lambda}}_{n+1}^j + \frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n^j}^{\mathsf{T}} \Delta\boldsymbol{\lambda}_{n+1}^j\right). \tag{4.11b}$$

To form the update of the adjoint correction over one time step we subtract the update of the approximate adjoint in Equation 4.8 from Equation 4.11b, resulting in:

$$\Delta\boldsymbol{\lambda}_n^j = -\left(\frac{\partial R_n}{\partial \boldsymbol{u}_n^j}\right)^{-\mathsf{T}} \frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n^j}^{\mathsf{T}} \Delta\boldsymbol{\lambda}_{n+1}^j, \tag{4.12}$$

which propagates the correction one step backward. Subsequently, the adjoint correction can be propagated over an entire interval from time step $n = n_{k+1}$ at terminal time $t_{n_{k+1}} = \theta_{k+1}$ until time step $n = n_k$ at initial time $t_{n_k} = \theta_k$:

$$\Delta\boldsymbol{\Lambda}_k = \left(\prod_{n=n_k+1}^{n_{k+1}} -\left(\frac{\partial R_{n-1}}{\partial \boldsymbol{u}_{n-1}^j}\right)^{-\mathsf{T}} \frac{\partial R_n}{\partial \boldsymbol{u}_{n-1}^j}^{\mathsf{T}}\right) \Delta\boldsymbol{\Lambda}_{k+1} = G_c(\Delta\boldsymbol{\Lambda}_{k+1}), \tag{4.13}$$

for which we used $\Delta\boldsymbol{\Lambda}_k = \Delta\boldsymbol{\lambda}_n$ at $\theta_k = t_n$ and we define operator G_c which propagates the adjoint correction backward over an interval as in Equation 4.13. Operator G_c thus differs from operator G in the fact that G propagates an adjoint variable using Equations 4.8 and 4.9, and G_c propagates an adjoint correction using Equations 4.12 and 4.13. Further analysis of local stability is performed by assuming the state equations $R_n(\boldsymbol{u}_n, \boldsymbol{u}_{n-1}, \boldsymbol{s}, t_n)$ to be linear with respect to $\boldsymbol{u}_n$ and $\boldsymbol{u}_{n-1}$. Although stability criteria can be derived for nonlinear state equations, proving local stability for nonlinear systems is out of the scope of this work. We assume linear state equations R^{lin}, which are used to represent any linear time stepping scheme:

$$R_n^{lin}(\boldsymbol{u}_n, \boldsymbol{u}_{n-1}, \boldsymbol{s}, t_n) = \boldsymbol{u}_n - \boldsymbol{A}_n(\boldsymbol{s})\boldsymbol{u}_{n-1} - \boldsymbol{q}_n(\boldsymbol{s}, t_n) = \boldsymbol{0}, \tag{4.14}$$

4

where the state update matrix $\boldsymbol{A}_n(\boldsymbol{s})$ and the external load vector $\boldsymbol{q}_n(\boldsymbol{s}, t_n)$ are both independent of $\boldsymbol{u}_n$ and $\boldsymbol{u}_{n-1}$. We note that any linear time stepping scheme (such as Forward Euler, backward Euler or Crank-Nicolson) can be transformed into R^{lin} by simple matrix manipulations. Consequently, the state variables are updated following the linear time stepping scheme:

$$\boldsymbol{u}_n = \boldsymbol{A}_n(\boldsymbol{s})\boldsymbol{u}_{n-1} + \boldsymbol{q}_n(\boldsymbol{s}, t_n). \tag{4.15}$$

This linear time stepping scheme is only stable if all eigenvalues ϕ_i of matrix $\boldsymbol{A}_n(\boldsymbol{s})$ are bounded as $|\phi_i(\boldsymbol{K}_n)| < 1$. Returning to the propagation of the adjoint correction we substitute derivatives:

$$\frac{\partial R_{n-1}^{lin}}{\partial \boldsymbol{u}_{n-1}^j} = \boldsymbol{I}, \quad \frac{\partial R_n^{lin}}{\partial \boldsymbol{u}_{n-1}^j} = -\boldsymbol{A}_n, \tag{4.16}$$

into the correction update in Equation 4.13, and find:

$$\Delta\boldsymbol{\Lambda}_k = \left(\prod_{n=n_k+1}^{n_{k+1}} \boldsymbol{A}_n^\mathsf{T} \right) \Delta\boldsymbol{\Lambda}_{k+1}. \tag{4.17}$$

Consequently, the eigenvalues with which the adjoint errors are propagated backward are also smaller than one $|\phi_i(\boldsymbol{A}_n^\mathsf{T})| = |\phi_i(\boldsymbol{A}_n)| < 1$. We may conclude that if the discrete state equations for a linear transient problem are stable then the discrete adjoint equations will also be stable and the adjoint error $\Delta\boldsymbol{\Lambda}_k$ will decrease over an interval. Furthermore, for a larger decrease in error, the number of steps in an interval $n_{k+1} - n_k \gg 1$ should be large or the absolute eigenvalues $|\phi_i(\boldsymbol{A}_n)| \ll 1$ should be smaller. However, a fine spatial mesh and thus a large size of the state solution m may cause large memory usage and thus require a user with limited memory to use many short intervals $K > 1$, which is prohibitive for the local convergence of the LT algorithm.

Global convergence

One of the most important assumptions of the LT method is that the design stabilizes and converges, which allows for the adjoint field to converge to the exact adjoint solution over multiple design iterations. This is caused by local convergence and the terminal adjoint $\boldsymbol{\Lambda}_K^j = \boldsymbol{\lambda}_N^j$ which is exact by definition. When the design $\boldsymbol{s}^j$ and the approximate adjoints $\tilde{\boldsymbol{\Lambda}}_k^j$ stabilize, the correct terminal adjoint $\boldsymbol{\Lambda}_K^j$ propagates backward-in-time to all

other $\tilde{\mathbf{\Lambda}}_k^j$, which must also be exact due to local convergence. Stable design and adjoints thus lead to exact adjoints and consequently an exact optimum where the objective is stationary $dF^*/d\boldsymbol{s} = 0$.

If the design does not stabilize, the adjoint field will not converge to the exact field resulting in erroneous sensitivities, which in turn, can cause the design to destabilize. This type of stability is referred to as global stability in this paper. By changing the adjoint equations, state equations and resulting state solutions, a design update changes the adjoint solutions. The update of the design as $\boldsymbol{s}^{j+1} = \boldsymbol{s}^j + \delta\boldsymbol{s}^j$ is thus associated with a change in exact adjoint variables $\mathbf{\Lambda}_k^{j+1} = \mathbf{\Lambda}_k^j + \delta\mathbf{\Lambda}_k^j$. If we assume local convergence is satisfied, we expect the backward propagation of the approximate adjoint over an interval to result in the exact adjoint $G(\tilde{\mathbf{\Lambda}}_{k+1}^j) = \tilde{\mathbf{\Lambda}}_k^{j+1} \approx \mathbf{\Lambda}_k^j$, but only for the current design $\boldsymbol{s}^j$. By updating the design, we change the exact adjoint solution by $\delta\mathbf{\Lambda}_k^j$, which in turn, introduces an error in the approximate adjoints estimated using the exact adjoints in the previous design iteration:

$$\tilde{\mathbf{\Lambda}}_k^{j+1} \approx \mathbf{\Lambda}_k^j = \mathbf{\Lambda}_k^{j+1} - \delta\mathbf{\Lambda}_k^j. \tag{4.18}$$

Conversely, as the sensitivities are dependent on the approximate adjoint variables and as we aim to use gradient-based design updates, $\delta\boldsymbol{s}^{j+1}$ is dependent on $\tilde{\mathbf{\Lambda}}_k^{j+1} \approx \mathbf{\Lambda}_k^{j+1} - \delta\mathbf{\Lambda}_k^j$ and thus on $\delta\mathbf{\Lambda}_k^j$. For the optimization procedure to stabilize, we require the effect of $\delta\mathbf{\Lambda}_k^j$ on $\delta\boldsymbol{s}^{j+1}$ and subsequently the effect of $\delta\boldsymbol{s}^{j+1}$ on $\delta\mathbf{\Lambda}_k^{j+1}$ to decrease the change in adjoint variables, such that $|\delta\mathbf{\Lambda}_k^{j+1}| < |\delta\mathbf{\Lambda}_k^j|$. We call instabilities due to this feedback loop of adjoint and design changes *strong global convergence* issues. A formal proof of strong global convergence is out of the scope of this paper and is not further discussed.

Beside strong global convergence, we can make some observations using the similarity between state and adjoint equations leading to *weak global convergence* issues. As shown in Section 4.2.1, the state equations resemble the adjoint equations and we expect features with large characteristic times in the state equations to also be associated with large characteristic times in the adjoint equations. Large approximate characteristic times (Picioreanu et al., 2000) are associated with large eigenvalues and relatively large settling times (Åström & Murray, 2020) over which state solutions stabilize. Consequently, adjoint errors decay relatively slow in these features. Further examples of these features and the computation of characteristic times will be given in Sections 4.5.1 and 4.5.2. Features with short characteristic times will have faster settling times and eigenvalues leading to a stronger local convergence. Essentially, we expect that the correct sensitivities for features with short characteristic times will be quickly found, while features with long characteristic times will have inaccurate sensitivities and may not be found by the optimizer. This introduces a bias towards features associated with short characteristic times. The LT algorithm may converge to inferior local optima containing features with shorter characteristic times than the local optima found by the GT algorithm which contains features with longer characteristic times.

We thus identified two types of stability which can be enforced in the following ways. For local stability we require the absolute eigenvalues to be smaller than one. To promote local convergence, we either require the maximum absolute eigenvalues to be *much* smaller than one, or the interval lengths to be relatively long. A global stability criterion

is harder to define. In the authors experience, by enforcing local stability and using optimization methods which limit large changes in design such as MMA (Svanberg, 1987), global stability issues can be overcome. Since the MMA employs adaptive move limits, design changes are restrained which restricts the associated changes in adjoint $\delta\mathbf{\Lambda}_k^j$. We will illustrate this fact using the examples in Section 4.5 and particularly the example in Figure 4.10b where we will find global stability issues which resolve themselves after additional design iterations. Furthermore, in this section, we only discuss the characteristic time in broad terms, and do not give strict criteria for the adjoint equations to be stable leading to consistent sensitivities. However, in Section 4.5, we examine the stability requirements using practical examples. In Sections 4.5.1 and 4.5.2 we study the characteristic times in more detail for a thermal and a flow problem. Subsequently, in Section 4.5.1 we will find a weak global convergence criterion which is verified in Section 4.5.2.

4.3.2. THE HYBRID CHECKPOINTING/LOCAL-IN-TIME ALGORITHM

As the LT algorithm is faster than the CP algorithm but contains errors due to the adjoint approximations, we would like to combine these two algorithms to attain a more accurate and computationally efficient algorithm. We call this novel algorithm the hybrid Checkpointing/Local-in-Time (CP/LT) algorithm. In principle, the CP/LT algorithm is based on the LT algorithm. However, when errors in the approximate adjoints become too large, corrections are performed using the CP algorithm. In Figure 4.4, we illustrate that the CP/LT algorithm is based on the LT algorithms in phase A, and adaptively performs corrections in phase B.

To measure the accuracy of the adjoint approximation in the LT algorithm in phase A, an approximate adjoint error in design iteration j at checkpoint k is defined as:

$$\varepsilon_{\lambda_k}^j = \frac{\|\tilde{\mathbf{\Lambda}}_k^{j+1} - \tilde{\mathbf{\Lambda}}_k^j\|_2}{\|\tilde{\mathbf{\Lambda}}_k^{j+1}\|_2} = \frac{\|\Delta\tilde{\mathbf{\Lambda}}_k^j\|_2}{\|\tilde{\mathbf{\Lambda}}_k^{j+1}\|_2}, \tag{4.19}$$

where we denote the L^2-norm as $\|\square\|_2$. Moreover, the correction in adjoint at checkpoint k is approximated as $\Delta\tilde{\mathbf{\Lambda}}_k^j = \tilde{\mathbf{\Lambda}}_k^{j+1} - \tilde{\mathbf{\Lambda}}_k^j \approx \mathbf{\Lambda}_k^j - \tilde{\mathbf{\Lambda}}_k^j = \Delta\mathbf{\Lambda}_k^j$, where we assume $\tilde{\mathbf{\Lambda}}_k^{j+1} \approx \mathbf{\Lambda}_k^j$ due to local convergence. Because the sensitivity in Equation 4.7 is dependent on the adjoint, the error is used as a measure of accuracy of the sensitivity. If the error $\varepsilon_{\lambda_k}^j$ is higher than a user-defined threshold $\varepsilon_\lambda^{max}$, the adjoint solution and sensitivity update on domain Θ_{k-1} are deemed inaccurate and need to be corrected. It should be noted that the error $\varepsilon_{\lambda_k}^j$ is dependent on $\tilde{\mathbf{\Lambda}}_k^{j+1} = G(\tilde{\mathbf{\Lambda}}_{k+1}^j)$ which is only known after evaluating the adjoint solution on domain Θ_k in the LT algorithm. As domain Θ_k needs to be evaluated before considering correcting domain Θ_{k-1}, we do not have the state solution on domain Θ_{k-1} in memory to perform a correction. The state solution thus needs to be recomputed on domain Θ_{k-1} starting from a stored state $\boldsymbol{U}_{k-1}^j$.

Making use of the linearity of both the adjoint and sensitivity equations, the sensitivity correction may be simplified to reduce additional computational effort. In Section 4.3.1, we found that the adjoint correction can be propagated backward over an interval using $\Delta\mathbf{\Lambda}_k = G_c(\Delta\mathbf{\Lambda}_{k+1})$ as defined in Equations 4.12 and 4.13. This propagation

can also be performed for approximate $\Delta\tilde{\boldsymbol{\lambda}}_n^j$, which is propagated over one time step as:

$$\Delta\tilde{\boldsymbol{\lambda}}_n^j = -\left(\frac{\partial R_n}{\partial \boldsymbol{u}_n^j}\right)^{-\mathsf{T}} \frac{\partial R_{n+1}}{\partial \boldsymbol{u}_n^j}^{\mathsf{T}} \Delta\tilde{\boldsymbol{\lambda}}_{n+1}^j. \tag{4.20}$$

Moreover, a correction for the sensitivities is defined by substituting the improved approximation $\boldsymbol{\lambda}_n^j \approx \tilde{\boldsymbol{\lambda}}_n^j + \Delta\tilde{\boldsymbol{\lambda}}_n^j$ into the exact sensitivity formulation (Equation 4.7, for clarity repeated in Equation 4.21a) resulting in Equation 4.21b:

$$\frac{dF}{d\boldsymbol{s}} = \sum_{n=0}^{N}\left(\frac{\partial F_n}{\partial \boldsymbol{s}}\Delta t + \boldsymbol{\lambda}_n^{\mathsf{T}}\frac{\partial R_n}{\partial \boldsymbol{s}}\right), \tag{4.21a}$$

$$\frac{dF}{d\boldsymbol{s}} = \sum_{n=0}^{N}\left(\frac{\partial F_n}{\partial \boldsymbol{s}}\Delta t + \tilde{\boldsymbol{\lambda}}_n^{j^{\mathsf{T}}}\frac{\partial R_n}{\partial \boldsymbol{s}} + \Delta\tilde{\boldsymbol{\lambda}}_n^{j^{\mathsf{T}}}\frac{\partial R_n}{\partial \boldsymbol{s}}\right), \tag{4.21b}$$

for which the first two terms have already been computed in the LT algorithm and a correction for the sensitivities can be build as:

$$\Delta\frac{dF}{d\boldsymbol{s}} = \sum_{n=0}^{N}\Delta\tilde{\boldsymbol{\lambda}}_n^{j^{\mathsf{T}}}\frac{\partial R_n}{\partial \boldsymbol{s}}. \tag{4.22}$$

Consequently, when adjoint errors for interval Θ_{k-1} are too high ($\varepsilon_{\lambda_k}^j > \varepsilon_\lambda^{max}$), we propagate the approximate adjoint correction backward using Equations 4.20 while simultaneously updating the sensitivities using Equation 4.22.

Adjoint correction $\Delta\tilde{\boldsymbol{\lambda}}_n^j$ is used to correct the sensitivities as it is an easier operation than updating the sensitivities using a new and improved adjoint field. Generally, we immediately write contributions to the sensitivity vector $dF/d\boldsymbol{s}$ to memory. To update the sensitivity using an improved adjoint field, we would have to remove the contributions added using the old adjoint field first. Using the adjoint correction we can update the sensitivity vector when necessary. Moreover, we found that updates for the adjoint correction using Equation 4.20 are computationally cheaper than updates in the full adjoint using Equation 4.8. Inspecting these equations, we find that propagating the adjoint involves a matrix vector multiplication, a vector addition, and a linear system solve, while propagating the adjoint correction only requires the matrix vector multiplication and the linear system solve. Nonetheless, the computational cost of the full process is increased compared to the LT algorithm as we need to recompute the full state solution on a domain which is being corrected to compute the sensitivity corrections in Equation 4.22. Furthermore, we note that after propagating the adjoint correction backward over an interval, the propagated correction may be used to improve the approximate adjoint at the checkpoint, i.e., $\tilde{\boldsymbol{\Lambda}}_k^j = G(\tilde{\boldsymbol{\Lambda}}_{k+1}^j) + G_c(\Delta\tilde{\boldsymbol{\Lambda}}_{k+1}^j)$. Finally, whether or not to apply this correction on an interval depends on the chosen adjoint error threshold $\varepsilon_\lambda^{max}$. Based on the experience of the authors, a relatively high threshold of $\varepsilon_\lambda^{max} = 0.1$ can be chosen. Using this threshold we will find errors in sensitivity to remain below 10%.

Based on these developments, the CP/LT algorithm as shown in Figure 4.4 consists of two phases, and can be described as follows.

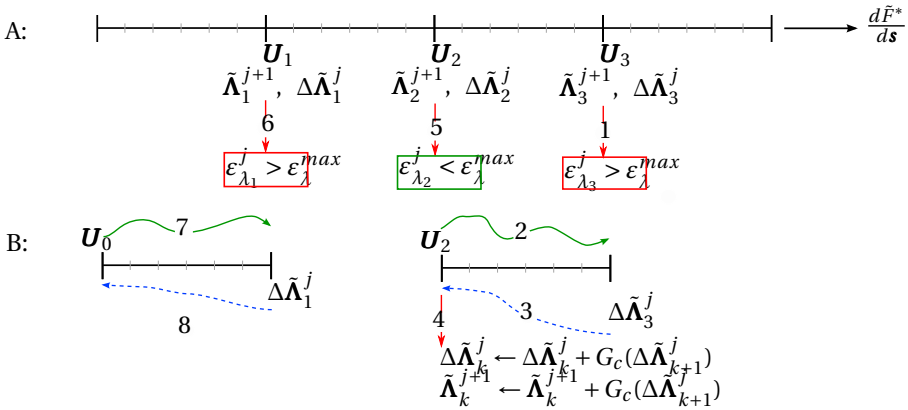


Figure 4.4: A schematic of the CP/LT algorithm. The numbers represent the order of operations. First the LT algorithm is executed in phase A resulting in approximate sensitivities $d\tilde{F}^*/d\boldsymbol{s}$. Additionally, $\boldsymbol{U}_k$ and $\Delta\tilde{\boldsymbol{\Lambda}}_k^j$ are stored on the checkpoints besides $\tilde{\boldsymbol{\Lambda}}_k^{j+1}$. In phase B, starting at the final to last interval and moving one interval backward at a time, the adjoint error is evaluated and a correction on the interval is performed if necessary. The dashed blue arrows represent the solving of the adjoint correction using Equation 4.20 and update of sensitivities using Equation 4.22. After propagating the adjoint correction backward from $k+1$ to k, the adjoint correction and adjoint approximation are both updated.

(A) **LT algorithm** The sensitivities are computed using the LT algorithm. However, besides the approximate adjoint variables $\tilde{\boldsymbol{\Lambda}}_k^{j+1}$, the adjoint correction $\Delta\tilde{\boldsymbol{\Lambda}}_k^j$, and state solution $\boldsymbol{U}_k$ are also stored at the checkpoints as shown in Figure 4.4.

(B) **Checkpointing corrections** Starting at the second to last interval $\Theta_k = \Theta_{K-2}$ using the stored $\tilde{\boldsymbol{\Lambda}}_{k+1}^{j+1}$ and $\Delta\tilde{\boldsymbol{\Lambda}}_{k+1}^j$, the error $\varepsilon_{\lambda_{k+1}}^j$ is computed. If the error is larger than the fixed threshold $\varepsilon_\lambda^{max}$, the state solution is recomputed and correction $\Delta\tilde{\boldsymbol{\Lambda}}_{k+1}^j$ is propagated backwards over the interval using Equation 4.20, while simultaneously updating the sensitivity using Equation 4.22. The backward propagation of the correction from $k+1$ to k is used to update the correction and adjoint variables at k as $\Delta\tilde{\boldsymbol{\Lambda}}_k^j \leftarrow \Delta\tilde{\boldsymbol{\Lambda}}_k^j + G_c(\Delta\tilde{\boldsymbol{\Lambda}}_{k+1}^j)$ and $\tilde{\boldsymbol{\Lambda}}_k^{j+1} \leftarrow \tilde{\boldsymbol{\Lambda}}_k^{j+1} + G_c(\Delta\tilde{\boldsymbol{\Lambda}}_{k+1}^j)$. Subsequently, a sensitivity correction is performed at interval Θ_{k-1} if necessary, and so on.

The CP/LT algorithm requires more memory than the LT and CP algorithms since the state solution $\boldsymbol{U}_k$, the adjoint field $\tilde{\boldsymbol{\Lambda}}_{k+1}^{j+1}$, and the adjoint correction $\Delta\tilde{\boldsymbol{\Lambda}}_{k+1}^{j+1}$ are stored on the checkpoints. On each interval except the first, the initial states are stored $m(K-1)$ discrete state variables, and on each interval except the last, the terminal adjoint and the terminal adjoint correction are stored $2m(K-1)$ discrete adjoint variables, resulting in the storage of $3m(K-1)$ discrete variables. Combining this with the storage of the state solution on an interval of mN/K discrete state variables, the CP/LT algorithm has a memory requirement which scales as $M \propto 3m(K-1) + mN/K$.

The computational cost of the CP/LT algorithm is not known a priori but a lower and upper bound can be derived. If the adjoint errors are negligible at all checkpoints and the sensitivities are thus reliable, no corrections are performed and the CP/LT algorithm

performs as the LT algorithm in phase A. The lower bound on the computational cost is thus the cost of the LT algorithm, i.e., $\underline{C} \propto Nc_s(1+r_c)$. However, if the adjoint errors are large at all checkpoints, a correction is performed at the first $K-1$ intervals which includes the solution of the state equations at a cost proportional to $c_s N/K$. Moreover, on each corrected domain, an adjoint correction is performed for which we estimate the computational cost as similar to the cost of the normal adjoint propagation $c_s r_c N/K$, although in practice the adjoint correction is a cheaper operation. The cost of the adjoint corrections is thus proportional to $(K-1)(c_s N/K + c_s r_c N/K) = Nc_s(1+r_c)(1-1/K)$, and an upper bound of the computational cost is derived as $\overline{C} \propto \underline{C} + Nc_s(1+r_c)(1-1/K) = Nc_s(1+r_c)(2-1/K)$.

4.3.3. PARALLEL-LOCAL-IN-TIME ALGORITHM

Parallelization is a useful method to speed up computations. To parallelize the computations, spatial domain decomposition is often used. However, this technique may suffer from communication overhead, limiting the maximum speedup. To efficiently reduce such communication, a novel Parallel-Local-in-Time algorithm is proposed in this work. The PLT algorithm is an extension of the LT algorithm and is similar to the direct multiple shooting algorithm as discussed by Carraro and Geiger (2015). For the temporal parallelization, we discretize the temporal domain into intervals Θ_k. These intervals are however decoupled and we will solve iteratively for both the approximate state and adjoint solutions. Beside the approximation of the adjoint variables as in the LT algorithm we approximate and update the initial state variables on an interval as:

$$\boldsymbol{U}_k^j \approx \tilde{\boldsymbol{U}}_k^j = U(\tilde{\boldsymbol{U}}_{k-1}^{j-1}), \tag{4.23}$$

where $U(\tilde{\boldsymbol{U}}_{k-1}^{j-1})$ is an operator which propagates initial state $\tilde{\boldsymbol{U}}_{k-1}^{j-1}$ over interval Θ_{k-1} in design iteration $j-1$ to terminal state $\tilde{\boldsymbol{U}}_k^{j-1}$. Terminal state solutions on interval $k-1$ at design iteration $j-1$ are thus used as initial states for interval Θ_k at design iteration j. Starting from the approximate initial state, the state equations can be solved on each interval in parallel. Subsequently, using approximate terminal adjoints $\tilde{\boldsymbol{\Lambda}}_{k+1}^j$, the complete sensitivity computation can be done in parallel for each interval Θ_k as illustrated in Figure 4.5. We extract the sensitivity contribution for interval Θ_k from Equation 4.7 as all contributions from time step $n = n_k+1$ at initial time $t_{n_k} = \theta_k$ until time step $n = n_{k+1}$ at terminal time $t_{n_{k+1}} = \theta_{k+1}$:

$$\frac{dF_k^*}{d\boldsymbol{s}} = \sum_{n=n_k+1}^{n_{k+1}} \left(\tilde{\boldsymbol{\lambda}}_n^{j\top} \frac{\partial R_n(\tilde{\boldsymbol{u}}_{n-1}^j, \tilde{\boldsymbol{u}}_n^j, \boldsymbol{s}, t_n)}{\partial \boldsymbol{s}} + \frac{\partial F_n(\tilde{\boldsymbol{u}}_n^j, \boldsymbol{s})}{\partial \boldsymbol{s}} \Delta t \right), \tag{4.24}$$

where we approximated all state variables as $\tilde{\boldsymbol{u}}_n^j \approx \boldsymbol{u}_n^j$ and used approximate adjoint variables $\tilde{\boldsymbol{\lambda}}_n^j$. To initialize the PLT algorithm, approximate adjoints and an acceptable guess for the state variables are required at the checkpoints. In the first design iteration, we compute the sensitivities using the LT algorithm and store the resulting state and adjoint variables $\tilde{\boldsymbol{U}}_k^{j=0}/\tilde{\boldsymbol{\Lambda}}_k^{j=0}$ at the checkpoints θ_k. Subsequently, the algorithm consists of the following three steps:

1. **Send information to computational nodes** To initialize the design evaluation, we assume there are as much computational nodes as there are intervals K, though this is not essential to the algorithm. Furthermore, we associate interval Θ_k to node k and send it an approximate initial state condition $\tilde{\boldsymbol{U}}_k^j$ (exact state solution $\boldsymbol{u}_0^j$ for the first interval), an approximate terminal adjoint condition $\tilde{\boldsymbol{\Lambda}}_{k+1}^j$ (exact adjoint solution $\boldsymbol{\lambda}_N^j$ for the final interval), and the design $\boldsymbol{s}^j$.

2. **Design evaluation** On each of the intervals Θ_k, the design is evaluated. Starting at approximate $\tilde{\boldsymbol{U}}_k^j$, the state solution is propagated forward to $\tilde{\boldsymbol{U}}_{k+1}^{j+1}$ by solving the state equations. Moreover, the state solution is stored for each time step within the interval. Subsequently, adjoint variable $\tilde{\boldsymbol{\Lambda}}_{k+1}^j$ is propagated backward to $\tilde{\boldsymbol{\Lambda}}_k^j$ using Equations 4.8 and 4.9 and used to update the interval sensitivities using Equation 4.24.

3. **Receive information from computational nodes** From each computational node k, we receive the terminal state solution $\tilde{\boldsymbol{U}}_{k+1}^j$, the initial adjoint solution $\tilde{\boldsymbol{\Lambda}}_k^j$, and domain sensitivities $dF_k^*/d\boldsymbol{s}$. Subsequently, the global sensitivity is computed by gathering the contributions of all intervals:

$$\frac{dF^*}{d\boldsymbol{s}} = \sum_{k=1}^{K} \frac{dF_k^*}{d\boldsymbol{s}}. \tag{4.25}$$

To improve approximate state and adjoint solutions, we update them using the propagated solutions on the neighboring intervals as:

$$\begin{aligned} \boldsymbol{U}_{k+1}^{j+1} &\approx U(\tilde{\boldsymbol{U}}_k^j), \\ \boldsymbol{\Lambda}_k^{j+1} &\approx G(\tilde{\boldsymbol{\Lambda}}_{k+1}^j), \end{aligned} \tag{4.26}$$

where we assume that $\boldsymbol{U}_{k+1}^{j+1} \approx U(\tilde{\boldsymbol{U}}_k^j)$ decreases the error in state, similar to the decreased error in adjoint by $\boldsymbol{\Lambda}_k^{j+1} \approx G(\tilde{\boldsymbol{\Lambda}}_{k+1}^j)$ as discussed in Section 4.3.1.

As was the case in the LT algorithm, we use approximate adjoint variables and therefore expect some inaccuracies due to local and global convergence issues in the PLT algorithm. Moreover, in the PLT algorithm we have also introduced approximate state solutions which may cause additional inaccuracies. However, following a similar reasoning as in Section 4.3.1 for the stability of the approximate adjoints, the errors in approximate state solutions are expected to reduce over an interval if local convergence is satisfied. If we describe the state solution as $\boldsymbol{U}_n = \tilde{\boldsymbol{U}}_n + \Delta\boldsymbol{U}_n$, where $\Delta\boldsymbol{U}_n$ is the error in state solution, we expect the error to behave the same as a disturbance of an initial state solution. For stable transient systems, a disturbance of the initial state solution is known to dampen out over time. These disturbances can generally be said to dampen out on a similar timescale as it takes the system to reach a stable steady state solution. To enforce local stability, we thus require relatively long intervals with respect to the characteristic time of the system. Consequently, as discussed for the LT algorithm in Section 4.3.1, we expect weak global convergence issues. The PLT algorithm is expected to favor convergence

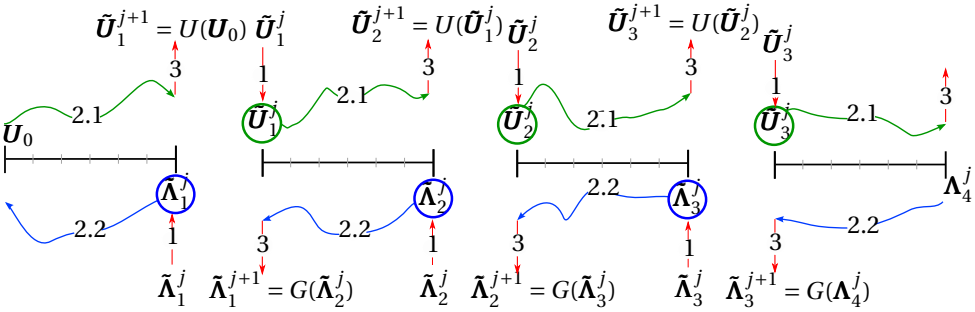


Figure 4.5: A schematic of the PLT algorithm. The numbers represent the order of operations. In the first step, we distribute all required state and adjoint fields to the intervals. In the second step, for each interval Θ_k, we evaluate and store the state solution starting from the distributed $\tilde{\boldsymbol{U}}_k^j$, and subsequently we evaluate the adjoint solution starting from distributed $\tilde{\boldsymbol{\Lambda}}_{k+1}^j$ while simultaneously computing domain sensitivity $dF_k^*/d\boldsymbol{s}$ as defined in Equation 4.24. Solving the state solutions and the adjoint solutions are labeled step 2.1 and 2.2, respectively. This does not imply that all intervals wait until all state solutions are solved. Some intervals might in fact be working on the adjoint solutions while others are working on the state solution but in general all intervals will be working on the state or adjoint solution at the same time. Finally, in the third step, all state and adjoint updates are gathered in addition to the sensitivity contributions $dF_k^*/d\boldsymbol{s}$ which are combined into the total sensitivity using Equation 4.25.

to local optima with features of shorter characteristic times as for these features accurate adjoints and states, and thus sensitivities, are found much quicker. Additionally, this bias may be enhanced by the PLT algorithm due to the approximate state solutions $\tilde{\boldsymbol{U}}_k$. Strong global convergence issues might also pose a problem for the PLT algorithm. However, in the authors experience the algorithm behaves similar to the LT algorithm. By enforcing local stability and using optimization methods which limit large changes in design, the PLT algorithm is found to avoid strong global convergence issues, and converges to stable optima with accurate sensitivities as will be shown in Section 4.5.

For the PLT algorithm to converge, we thus require errors in state solution to dampen out over successive design iterations. This may not always be the case. If the physics show chaotic behavior, small differences in initial state may cause large differences in later states. It is clear that for these types of systems, errors in state solution do not dampen out over successive design iterations. Moreover, under certain conditions the adjoint solutions of chaotic problems may grow exponentially, and the adjoint sensitivities may be inaccurate (Lea et al., 2000; Q. Wang et al., 2014). For chaotic systems even the CP and GT algorithms which compute exact sensitivities may also be inaccurate. Algorithms presented in this paper should therefore not be applied to chaotic systems without a careful consideration.

Since the PLT algorithm performs all computations simultaneously, the full state solution at all intervals (and thus the whole temporal domain) needs to be stored. Memory requirements thus scale as $M \propto Nm$. However, an increase in computational nodes often comes with an increase in memory alleviating this problem. When memory becomes a limiting factor, the parallel intervals can be subdivided once more into local subintervals where the LT or CP algorithm is employed. On these subintervals, we thus use different approximations for initial state and terminal adjoint equations. Another solution is to divide the domain into shorter intervals and evaluate some intervals sequentially in-

stead of in parallel, but still use approximate initial state and terminal adjoint solutions on each of these shorter intervals. Doing so, the full state solution does not need to be stored but only the state solutions on the intervals being evaluated.

As all state and adjoint solutions are solved once per interval, the computational cost of the PLT algorithm is the same as for GT algorithm, $C \propto Nc_s(1+r_c)$. Due to the parallelization, the computational time may be reduced. We investigate a theoretical 2D optimization problem for illustration purpose, though a similar investigation holds for 3D problems. Solving a transient 2D problem we might conceptualize as solving the discrete solution on a 3D cuboid where two of its axis represent the spatial axis and one represents the temporal axis as can be seen in Figure 4.6. Implementing a spatial domain decomposition, we cut the domain along its length and generate interface area across which the nodes have to communicate. If a time stepping algorithm is used this communication happens every time step leading to increased idle time. Moreover, the communication overhead might become even worse when nonlinear systems are solved as they may require many subsolves per time step. If a temporal parallelization can be used, the cuboid can be cut across its width generating less interface area and less communication overhead. We note that for this analysis we used a problem with more time steps than elements in the x or y direction. A problem with more elements in x or y direction than time steps may have less interface area and be more efficiently parallelized when using spatial domain decomposition techniques.

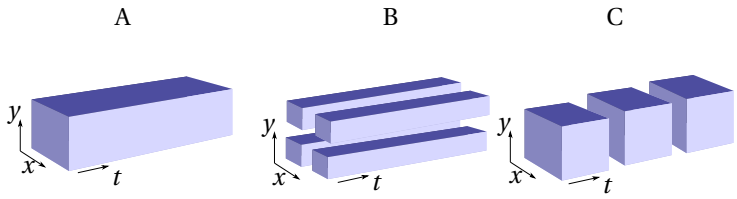


Figure 4.6: An example of the two options for paralellizing a 2D transient computation, where A is the non-parallelized domain, B is a parallelization via spatial domain decomposition, and C is a parallelization via temporal domain decomposition.

4.4. MEMORY AND COMPUTATIONAL COST

All presented sensitivity computation algorithms have their advantages and limitations and are suited for different kind of optimization problems. Based on a combination of memory requirements, computational cost, and convergence, different algorithms are recommended for different problems.

4.4.1. MEMORY REQUIREMENTS

When memory requirements are not a limiting factor, the full state solution may be stored and the GT or PLT algorithms can be used. Moreover, if the parallel algorithms are preferred for their speedup but memory is a limiting factor, a hierarchical structure of the algorithms may be used. In this paper, the CP, LT, and CP/LT algorithms were used

to reduce memory requirements as summarized in Table 4.1. However, for local stability, we require the intervals to be as long as possible and we thus use the least amount of intervals K as possible. Subsequently, it can be argued that for optimization problems where $N \gg K$ the differences in memory reduction between the algorithms are negligible, as the largest memory burden is spent on storing the complete state solution on an interval and not on storing the additional information at the checkpoints.

	GT, PLT	CP	LT	CP/LT
Memory requirements	mN	$m\left(\frac{N}{K}+K-1\right)$	$m\left(\frac{N}{K}+K-1\right)$	$m\left(\frac{N}{K}+3(K-1)\right)$
$K \ll N$	mN	$m\frac{N}{K}$	$m\frac{N}{K}$	$m\frac{N}{K}$

Table 4.1: Memory requirements M of the sequential algorithms for a problem containing K intervals, N time steps, and adjoint/state fields of size m. For problems where $K \ll N$, memory requirements are identical for all algorithms.

Under this assumption all sequential algorithms thus perform similarly with respect to memory reduction. However, if a small number of time steps is used and memory limits are reached due to m, the size of the solution at one time step, these assumptions do not hold anymore and the CP or LT algorithms should be used for the largest memory reduction.

4.4.2. COMPUTATIONAL COST

To make a comparison in terms of computational cost, an overview is given in Table 4.2. The CP/LT algorithm does not have a fixed computational cost but rather a range of possible costs with a lower bound at the cost of the GT, PLT, and LT algorithms and an upper bound above all other algorithms. However, when the state equations are highly nonlinear and $r_c \to 0$, we find the upper bound of the cost of the CP/LT algorithm in Table 4.2 to be the same as the cost of the CP algorithm. Another simplification can be made for linear implicit state equations. When the state equations are linear, adjoint equations are linear and have a similar computational cost. The resulting *implicit* adjoint equations involve a relatively expensive inverse matrix problem, while the sensitivity update only involves cheap matrix/vector multiplications. The computational cost c_a consisting of the adjoint and sensitivity update can thus be assumed to be dominated by the adjoint update which is similar to the state update: $c_a = c_s = c$ and $r_c = 1$. This particular case is shown in the bottom row of Table 4.2. However, when the state and consequently adjoint equations are linear *explicit*, both the adjoint and state equations consist only of matrix vector multiplications. In this case adjoint, state, and sensitivity updates have a similar computational cost. The cost of solving adjoint and updating sensitivity c_a will thus be larger than c_s and $r_c > 1$. This case is further investigated in Section 4.5.1.

Selecting an appropriate algorithm for a problem may depend on the computational cost and we further analyze the cost of the CP and CP/LT algorithms. Firstly, we examine the relative computational cost of the CP and GT algorithms by dividing the computa-

	GT, LT, PLT	CP/LT	CP
C	$c_s N(1+r_c)$	$[c_s N(1+r_c), c_s N(1+r_c)(2-1/K)]$	$c_s N(2+r_c-1/K)$
C, $r_c \to 0$	$c_s N$	$[c_s N, c_s N(2-1/K)]$	$c_s N(2-1/K)$
C, $r_c = 1$	$2cN$	$[2cN, cN(4-2/K)]$	$cN(3-1/K)$

Table 4.2: Computational costs C of the sequential algorithms for problems containing K intervals and N time steps. Solving one state step has a computational cost of c_s while solving one adjoint step while simultaneously updating the sensitivities has a cost of c_a, relative cost is defined as $r_c = c_a/c_s$. Contrary to the GT, PLT, LT, and CP algorithms, the CP/LT algorithm has a lower and upper bound on its computational cost. The lower bound is equal to the cost of the GT, PLT, and LT algorithms while the upper bound is higher than the cost of the CP algorithm as in the CP/LT algorithm state and adjoint solutions may be recomputed whereas in the CP algorithm only state solutions are recomputed. A simplification is shown for problems with highly nonlinear state equations where $r_c \to 0$ and for problems with linear explicit state equations where $r_c = 1$ and $c_a = c_s = c$.

tional cost of the CP algorithm C^{CP} by the cost of the GT algorithm C^{GT}:

$$\frac{C^{CP}}{C^{GT}} = \frac{c_s N(2+r_c-1/K)}{c_s N(1+r_c)} = \frac{2+r_c-1/K}{1+r_c}. \tag{4.27}$$

As expected, we find the computational cost of the GT and CP algorithms with $K = 1$ to be equal. Moreover, smaller K make the ration drop and have a diminishing effect on the increased computational cost of the CP algorithm. We thus advise for the CP algorithm to keep K as low as possible. For $K \gg 1$, the relative computational cost only depends on r_c. The biggest increase in computational cost is found for nonlinear systems where $r_c \to 0$ and $C^{CP}/C^{GT} \to 2$. Therefore, we advise to disregard the CP algorithm for nonlinear systems based on the required computational cost.

As corrections, even if only a few, are always performed for the CP/LT algorithm, not only the CP but also the CP/LT algorithm turns out to be more expensive than the GT, LT, and PLT algorithms. An informed choice between the CP/LT and CP algorithms thus depends on the number of corrections required by the CP/LT algorithm and the resulting relative computational cost. Comparing computational cost is not straightforward as the cost of the CP/LT algorithm is undefined *a priori*. We adaptively correct intervals only when errors are high and do not correct all intervals, as shown in Figure 4.4. Relative computational cost thus depends on the fraction of corrected intervals, and the relative cost of CP/LT and CP intervals. The computational cost of the terminal interval Θ_{K-1} is the same for both the CP/LT and CP algorithms as the state and adjoint equations are evaluated only once on this interval. On the other $K-1$ intervals which contain N/K time steps each, computational costs differ. In the CP algorithm, the state equations on these intervals are solved twice and the adjoint equations once, at a cost $\propto c_s(N/K)(2+r_c)$ per interval. On a corrected CP/LT interval, the state equations are solved twice and both the adjoint and adjoint correction are solved once at a cost $\propto c_s(N/K)(2+2r_c)$, and on an uncorrected interval state and adjoint are solved only once at a cost $\propto c_s(N/K)(1+r_c)$. A corrected and uncorrected interval thus have a cost relative to a CP interval, respectively as:

$$\frac{c_s(N/K)(2+2r_c)}{c_s(N/K)(2+r_c)} = 2\frac{1+r_c}{2+r_c}, \qquad \frac{c_s(N/K)(1+r_c)}{c_s(N/K)(2+r_c)} = \frac{1+r_c}{2+r_c}. \tag{4.28}$$

To compare the overall computational cost, we define the fraction of corrected intervals f_c. Since the terminal interval has the same computational cost for both CP and CP/LT, the fraction only considered the first $K-1$ intervals. Subsequently, f_c is defined as the number of corrected intervals in the whole optimization process, divided by the product of $K-1$ and the number of design iterations. It is thus the fraction of corrected intervals, averaged over all design iterations. The relative computational cost of the CP/LT algorithm to the CP algorithm can be computed as:

$$\frac{C^{CP/LT}}{C^{CP}} = \frac{K-1}{K}\left(f_c 2\frac{1+r_c}{2+r_c} + (1-f_c)\frac{1+r_c}{2+r_c}\right) + \frac{1}{K} = \frac{K-1}{K}(f_c+1)\frac{1+r_c}{2+r_c} + \frac{1}{K}, \tag{4.29}$$

where for the first $K-1$ intervals, we have differing computational cost at $(K-1)/K\%$ of the computations, and at the last interval we have the same computational cost at $1/K\%$ of the computations.

In practice we need to approximate f_c to choose whether the CP/LT or the CP algorithm is cheaper. However, in Section 4.5 we find that corrections are only performed during the first part of the optimization process where large design changes are present. Experienced designers can thus use their knowledge of the convergence behavior of the problem to estimate f_c and select the computationally most advantageous algorithm, as further discussed in the guidelines in Section 4.6.

4.5. RESULTS

In this section we investigate and compare the GT, CP, LT, CP/LT, and PLT algorithms. We investigate stability by optimizing a transient thermal problem in Section 4.5.1 and computational cost by optimizing a flow problem in Section 4.5.2. The thermal problem will minimize the temperature in heat generating components and the flow problem will optimize a piston pump. Both problems are optimized using density-based topology optimization (Bendsøe & Sigmund, 2004). In density-based topology optimization, we aim to find an optimal material distribution in a given design domain. The design domain is divided into grid cells to which design variables are attached. Subsequently, the design variables are used to interpolate continuously between phases, solid and void for the thermal problem, solid and fluid for the flow problem. As we interpolate continuously between phases, the gradient-based MMA (Svanberg, 1987) algorithm can be used. To compute the gradients in our transient thermal and flow problems, the algorithms presented in this paper will be used.

For the gradient computation, the adjoint equations need to be solved. To solve the adjoint equations, Jacobians $\partial R_n/\partial \boldsymbol{u}_n$ and $\partial R_n/\partial \boldsymbol{u}_{n-1}$ need to be constructed. We assume that $\boldsymbol{u}_n$ either contains discrete temperatures or velocities and pressures. The Jacobians will be constructed using the same approach as used by Theulings et al. (2023). For completeness, it is summarized here. The finite volume method will be used to discretize the PDE equations into a column R_n. Each i^{th} component R_n^i of the column may be associated with small subsets $\boldsymbol{u}_n^i$ and $\boldsymbol{u}_{n-1}^i$ of the complete sets of DOFs $\boldsymbol{u}_n$ and $\boldsymbol{u}_{n-1}$. To construct the Jacobians we use the (MATLAB, 2019) symbolic toolbox and construct symbolic equations for $R_n^i(\boldsymbol{u}_n^i, \boldsymbol{u}_{n-1}^i)$ in terms of the *symbolic* variables in $\boldsymbol{u}_n^i$ and $\boldsymbol{u}_{n-1}^i$. Subsequently, we use symbolic differentiation to compute $\partial R_n^i/\partial \boldsymbol{u}_n^i$, $\partial R_n^i/\partial \boldsymbol{u}_{n-1}^i$ and use MATLAB to automatically construct a function which takes DOFs $\boldsymbol{u}_n^i$, $\boldsymbol{u}_{n-1}^i$ and returns

$\partial R_n^i/\partial \boldsymbol{u}_n^i$, $\partial R_n^i/\partial \boldsymbol{u}_{n-1}^i$. This function is used to compute all derivatives after which they are assembled into $\partial R_n/\partial \boldsymbol{u}_n$ and $\partial R_n/\partial \boldsymbol{u}_{n-1}$. Moreover, derivatives $\partial R_n/\partial \boldsymbol{s}$ are computed following a similar approach.

4.5.1. STABILITY INVESTIGATION THROUGH THERMAL OPTIMIZATION

In this section we investigate the stability of the approximate LT, PLT, and CP/LT algorithms by optimizing a thermal problem. Weak global convergence is investigated by examining characteristic times and comparing it against the interval lengths. As discussed in Section 4.3.1, we expect the optimizer to be biased towards features with short characteristic times. Specifically, we expect a stronger bias for features with a long characteristic time relative to interval length $\Delta\theta$. Additionally, we examine the predicted and measured computational cost of the algorithms for problems with linear explicit state equations. The thermal problem is chosen for its simplicity as we can estimate relatively easily how fast errors in approximate adjoint decay and thus how fast we converge to accurate solutions.

TRANSIENT THERMAL MODEL

Before describing the optimization problem, we introduce the thermal physic and numerical model. The characteristic timescales related to the physics and discretization are investigated as they play an important role in the stability of the algorithms and set up of the optimization problem. First, we discretize the transient equations in space and time. A two-dimensional transient thermal problem is considered and is defined on temporal domain $t \in [0, t_t]$ and spatial domain $\vec{x} \in \Omega$ with boundary $\Gamma = \Gamma_d \cup \Gamma_n$. The thermal problem is governed by:

$$\begin{aligned} \alpha_s \rho_s c_{ps} \dot{T} - \nabla \cdot (\alpha_s \kappa_s \nabla T) - \alpha_s Q = 0, &\quad \text{on } \Omega, \\ T = T_\Gamma, &\quad \text{on } \Gamma_d, \\ \alpha_s \kappa_s \nabla T \cdot \boldsymbol{n} = \boldsymbol{q}_T \cdot \boldsymbol{n}, &\quad \text{on } \Gamma_n, \\ T = \hat{T}, &\quad \text{on } \Omega \text{ at } t = 0, \end{aligned} \tag{4.30}$$

where the subscript $\square_s$ denotes solid material properties, $T(\vec{x}, t)$ is the temperature field with time derivative $\dot{T} = \partial T/\partial t$, the solid density, specific heat capacity, and thermal conductivity are ρ_s, c_{ps}, and κ_s, respectively, $Q(\vec{x}, t)$ is an externally applied heat load, $T_\Gamma(\vec{x}, t)$ is the fixed temperature on boundary Γ_d, $\boldsymbol{q}_T(\vec{x}, t)$ is an applied heat flux on boundary Γ_n with outward normal $\boldsymbol{n}$, and $\hat{T}(\vec{x})$ is the initial temperature distribution. The solid volume fraction $\alpha_s(\vec{x})$ is used as design variable. The solid domain is defined by $\alpha_s(\vec{x}) = 1$ and the void domain by $\alpha_s(\vec{x}) = \underline{\alpha}_s$ which is a lower bound on the volume fraction for which $0 < \underline{\alpha}_s \ll 1$. The transient thermal equations are discretized on a Cartesian mesh illustrated in Figure 4.7 using the finite volume techniques as described by Versteeg and Malalasekera (2007). Specifically, we use the same techniques as used by Gersborg-Hansen et al. (2006), where the arithmetic average is used to interpolate discrete design variables at the interface between mesh elements. Moreover, we discretize in time using

a forward Euler scheme such that we find discretized equations of the form:

$$R_n(\boldsymbol{T}_{n-1}, \boldsymbol{T}_n, \boldsymbol{s}, t_n) = \boldsymbol{M}(\boldsymbol{s})\frac{\boldsymbol{T}_n - \boldsymbol{T}_{n-1}}{\Delta t} - \boldsymbol{K}(\boldsymbol{s})\boldsymbol{T}_{n-1} - \boldsymbol{Q}_n(\boldsymbol{s}),$$
$$R_0(\boldsymbol{T}_0) = \boldsymbol{T}_0 - \hat{\boldsymbol{T}}, \tag{4.31}$$

where $\boldsymbol{T}_n$ and $\boldsymbol{Q}_n$ are the vectors of discrete nodal temperatures and heat loads at time t_n, $\hat{\boldsymbol{T}}$ is the vector of discrete initial temperatures, and $\boldsymbol{s}$ is the vector of discrete design variables. The advantage of using the explicit forward Euler updating scheme is the relatively small computational cost attributed to performing one time step.

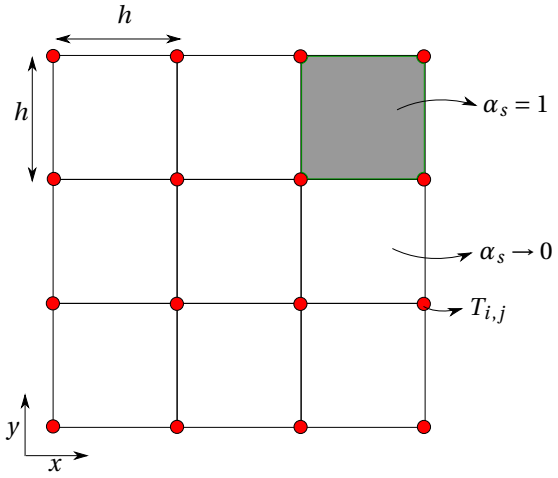


Figure 4.7: The equidistant uniform mesh used to discretize the thermal problem. Red dots are the temperature nodes with DOFs $T_{i,j}$ and densities are defined per element where $\alpha_s = 1$ is a solid element and $\alpha_s \to 0$ a void element.

To inspect stability, we approximate the characteristic times of the physics and discretization scheme. Although performing a forward Euler update is inexpensive, the scheme for transient thermal equations has a relatively strict stability constraint on the time step defined as:

$$\Delta t < \frac{h^2}{2D}, \tag{4.32}$$

where h is the mesh size and $D = \kappa_s/(\rho_s c_{ps})$ is the thermal diffusivity. Thermal diffusivity in both the solid ($\alpha_s = 1$) and void domains ($\alpha_s \to 0$) will stay the same as in Equation 4.30 both the conductivity κ_s and the density ρ_s are multiplied with α_s and $D = \alpha_s\kappa_s/(\alpha_s\rho_s c_{ps}) = \kappa_s/(\rho_s c_{ps})$. The strict time step constraint leads to the need for many updates to be performed which increases computational time. However, we find that this small time step may be required to accurately represent the physics. The characteristic timescale for a thermal diffusion problem (Picioreanu et al., 2000) can be defined as:

$$\tau = \frac{L^2}{D}, \tag{4.33}$$

where L is the characteristic lengthscale of the problem. The characteristic timescale indicates a time over which a transient response settles. If we are interested in a design with a feature size of a few elements ($L \approx h$) we find that these features have a characteristic timescale of $\tau_h \approx h^2/D$, and thus need the relatively small time step defined in Equation 4.32 to accurately model such features. A disadvantage of using a small time step is that it will cause large memory requirements. However, the methods presented in this paper can be used to keep these memory requirements to a minimum. Since this is a linear system, the local stability analysis in Section 4.3.1 holds. The adjoint equations are stable if the state equations are stable when we satisfy Equation 4.32. Moreover, as discussed in Section 4.3.1, the adjoint equations react and stabilize over similar timescales as the state equations. We can thus use the estimate for characteristic time in Equation 4.33 to estimate over which timescales adjoint errors converge to zero.

TRANSIENT THERMAL OPTIMIZATION PROBLEM

The problem in Figure 4.8 is designed to investigate the limitations of the approximate algorithms. The problem consists of a plate on which three heat generating components are attached. Material and problem specific parameters are given in Table 4.3, and result in a diffusivity of $D = 10^{-3}$. The characteristic timescale of the plate is estimated as

L	h	t_t	Δt	ρ_s	c_{ps}	κ_s
0.1	$\frac{10^{-2}}{12}$	10	$\frac{1}{5760}$	100	10	1

Table 4.3: The parameters used for the thermal optimization scenario as shown in Figure 4.8 with results shown in Section 4.5.1.

$\tau = 10$, which is used as final time $t_t = \tau = 10$. The bottom boundary of the plate is cooled and has fixed temperature $T = 0$ while all other boundaries are isolated. Three time dependent heat sources Q_1, Q_2, and Q_3 are added to the plate:

$$Q_1 = 10^3, \quad Q_2 = \begin{cases} \frac{3}{4} \cdot 10^3, & \text{if } t > \frac{t_t}{3} \\ 0, & \text{otherwise} \end{cases}, \quad Q_3 = \begin{cases} 6 \cdot 10^3, & \text{if } t > \frac{2t_t}{3} \\ 0, & \text{otherwise} \end{cases} \tag{4.34}$$

The objective of the optimization procedure is to minimize the average temperature at the heat sources:

$$F = \int_0^{t_t} \int_{\Omega_Q} T \, d\Omega \, dt, \tag{4.35}$$

where Ω_Q is the gray domain in Figure 4.8 where heat loads are applied.

Furthermore, to prevent checkerboarding and regularize the optimization procedure, a smoothing and continuous Heaviside projection filter are needed. First all i^{th} design variables s_i of $\boldsymbol{s}$ are smoothed using the filter as presented by Bruns and Tortorelli (2001) using the design variables within a distance r to the center of s_i, resulting in smoothed variables $\bar{s}_i$. Subsequently, the continuous Heaviside function as presented by F. Wang et al. (2011) is applied to the smoothed design variables $\bar{s}_i$ which projects them to a 0/1 solution as:

$$\tilde{s}_i = \frac{\tanh(\beta\eta) + \tanh(\beta(\bar{s}_i - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))}, \tag{4.36}$$

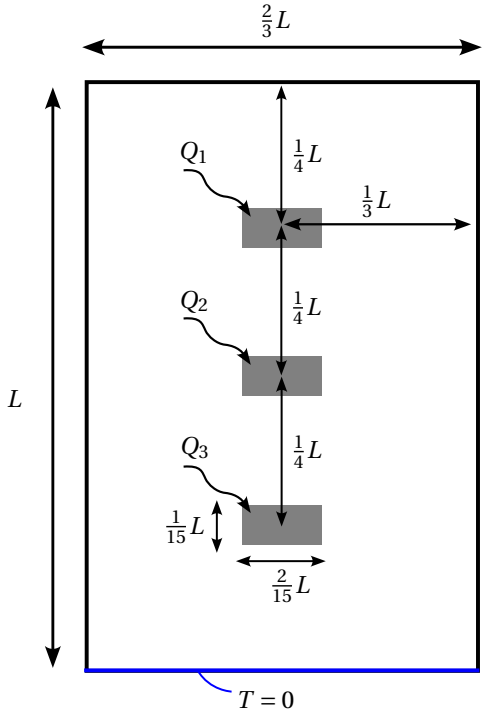


Figure 4.8: The thermal optimization problem with parameters in Tables 4.3, 4.4. On the gray areas, time dependent heat loads are applied as defined in Equation 4.34. The heat loads are defined such that Q_2 delivers half as much and Q_3 twice as much energy to the system as Q_1. On the bottom boundary, a heat sink is applied which fixes the temperature to $T = 0$ and all other boundaries are isolated.

where $\tilde{s}_i$ is the design variable after the projection filter is applied, η is the threshold value, and β indicates the slope and sharpness of the projection filter. Generally, β is increased during an optimization procedure through a continuation scheme such that in the earlier design iteration the continuous Heaviside projection does not restrict the convergence and in the later iterations the design is pushed to a 0/1 solution. In this work we set the slope to $\underline{\beta}$ in the first ten design iterations after which we increase it with $\Delta\beta$ each iteration until it reaches its upper bound $\beta = \overline{\beta}$. Subsequently, a volume constraint is applied on the smoothed and projected design variables $\tilde{\boldsymbol{s}}$. The constraint imposes an upper limit V_f on the solid volume fraction:

$$g_v(\tilde{\boldsymbol{s}}) = \frac{\sum_{i=1}^{N_d} \tilde{s}_i}{N_d} - V_f \leq 0, \tag{4.37}$$

where N_d is the number of discrete design variables $\tilde{s}_i$ in $\tilde{\boldsymbol{s}}$. Moreover, the Solid Isotropic Material with Penalization (SIMP) material interpolation (Bendsøe & Sigmund, 1999) is used which defines a relation between design variables and material properties. We use SIMP to interpolate the solid volume fraction as:

$$\alpha_s(\tilde{s}_i) = \underline{\alpha}_s + (1 - \underline{\alpha}_s)\tilde{s}_i^p, \tag{4.38}$$

where $\underline{\alpha}_s$ is the lower bound on the solid volume fraction and p regulates the convexity of the interpolation function. All optimization parameters can be found in Table 4.4.

Finally, using the discrete thermal model in Equation 4.31, the discrete optimization

r	$\underline{\beta}$	$\Delta\beta$	$\overline{\beta}$	p	V_f	$\underline{\alpha}_s$	$\varepsilon_\lambda^{max}$
$3h$	2	0.1	8	3	0.3	10^{-3}	0.1

Table 4.4: The optimization parameters for the thermal problem in Figure 4.8. We smooth the design over a radius of three elements ($r = 3h$), increase the Heaviside projection slope from β to $\overline{\beta}$ by $\Delta\beta$ over iterations 10 to 70 and apply the SIMP interpolation to the volume fraction α_s using factor $\overline{p}$. Furthermore, a maximum allowable adjoint error of $\varepsilon_\lambda^{max}$ is set for the CP/LT algorithm.

problem is defined as:

$$\begin{aligned} \underset{\boldsymbol{s}}{\text{minimize}} \quad & F \approx \sum_{n=0}^{N-1} F_n(\boldsymbol{T}_n, \boldsymbol{s})\Delta t, \\ \text{subject to} \quad & R_n(\boldsymbol{T}_{n-1}, \boldsymbol{T}_n, \boldsymbol{s}, t_n) = 0 \quad n \in \{1, 2, \dots N\}, \\ & R_0(\boldsymbol{T}_0) = 0, \\ & g_v(\tilde{\boldsymbol{s}}) \leq 0, \end{aligned} \tag{4.39}$$

We examine the characteristic times of the heat generating components to compare them to the interval lengths and make predictions about the designs that will result from the optimization procedure. We expect the optimal design to connect all heated domains to the heat sink at the bottom boundary. The heat sources are defined such that Q_2 delivers half as much and Q_3 twice as much heat to the system as Q_1. Connecting the top domain directly to the heat sink might be disadvantageous as this allows for a direct flow of thermal energy from the top domain through the center and bottom domains. As the top source is located further away from the heat sink, we expect it to have a larger characteristic time, leading to a slower decay of errors in adjoint. We estimate the characteristic timescales of the heat sources using their respective distances to the heat sink: $L_1 = \frac{3}{4}L$, $L_2 = \frac{1}{2}L$, and $L_3 = \frac{1}{4}L$. Subsequently, the characteristic timescales are estimated following Equation 4.33:

$$\tau_1 = \frac{L_1^2}{D} = \frac{\left(\frac{3}{4}L\right)^2}{D} = \frac{9}{16}\frac{L^2}{D} = \frac{9}{16}\tau, \qquad \tau_2 = \frac{1}{4}\tau, \qquad \tau_3 = \frac{1}{16}\tau. \tag{4.40}$$

The characteristic timescale of the bottom source is thus almost an order of magnitude smaller than the characteristic timescale of the top source. Moreover, the distance between the top two heat sources is $L_{1,2} = \frac{1}{4}L = L_3$, and the characteristic timescale of interaction between these two sources is thus also $\tau_{1,2} = \tau_3 = \frac{1}{16}\tau$. Erroneous adjoints carrying information about the interaction between the two top domains may thus converge to the correct adjoint much quicker than erroneous adjoints carrying information about the interaction between the top influx and heat sink. We thus expect the approximate algorithms to experience problems around the top influx domain.

To estimate the memory requirements of this optimization problem we compute the number of DOFs in x and y-direction respectively as $N_x = 81$ and $N_y = 121$. Subsequently, by using the fact that each DOF is stored as a double containing 8 bytes the size of one state solution is computed as $m = 8N_xN_y \approx 78$ kB. Since the number of time steps is $N = t_t/\Delta t = 57.6 \cdot 10^3$, we estimate the memory requirements assuming that $K \ll N$. Following Table 4.1, the GT and PLT algorithms require a memory of $M = mN \approx 4.5$ GB and the CP, LT, and CP/LT algorithms require a memory of $M \approx mN/K \approx 4.5/K$ GB. With these reasonable memory requirements, no large K is needed. However, if a similar problem is solved in 3D with a resolution of $N_z = N_x = 81$ in z-direction, this would increase the size of the state solution to $m = 8N_xN_yN_z \approx 6.4$ MB. The memory requirements would thus increase as $M \approx mN \approx 368$ GB for the GT and PLT algorithms. We thus argue that the relatively high number of intervals $K = 20$ should be used to reduce the theoretical memory in 3D to $M \approx mN/K \approx 17.8$ GB for the CP, LT, and CP/LT algorithms.

REFERENCE RESULTS

Using the thermal optimization problem, we analyze the stability of the approximate LT, PLT, and CP/LT algorithms. We expect the adjoint errors in the approximate algorithms to dampen out over multiple design iterations. However, due to the large errors in the first few design iterations, the optimization procedure may experience weak global convergence issues discussed in Section 4.3.1. The approximate LT, CP/LT, and PLT algorithms might take a different convergence path than the exact GT and CP algorithms and end up in different local optima with a bias towards design features with short characteristic times.

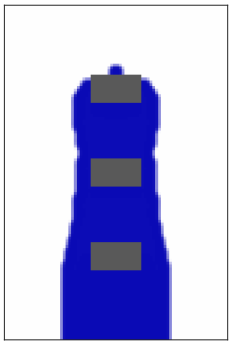


Figure 4.9: The bowling pin like design found by the GT and CP ($K = 20$) algorithms characterized by an objective value $f^*_{GT} = 1.0263$. The design connects all influx domains to the lower heat sink. The hat on top of the pin is an artifact of the optimization procedure.

As a reference we optimize the problem using the GT and CP ($K = 20$) algorithms which both lead to the same optimum shown in Figure 4.9. The optimal *bowling pin* like design connects all source domains to the bottom heat sink. We expect to find weak global convergence issues in the designs around the top source using the approximate

algorithms. To analyze these errors we measure the characteristic time of the top influx domain in the GT and CP design in Figure 4.9. To measure the characteristic time, a constant heat flux is applied only to the top interval ($Q_1(t) = 10^3$, $Q_2(t) = Q_3(t) = 0$). Under these conditions, the characteristic time is defined as the time it takes for the transient response to reach a solution with 5% error compared to the steady state solution. The characteristic time of the upper domain in the GT and CP design is measured as $\tau_1 = 6.12$.

Moreover, to measure the accuracy of the approximate sensitivities computed using the LT, CP/LT, and PLT algorithms, we compare them with sensitivities obtained with the GT algorithm. For each design and sensitivity $d\tilde{F}/d\boldsymbol{s}$ found by the approximate algorithms, an error is defined by comparing with exact sensitivities $dF/d\boldsymbol{s}$ computed by the GT algorithm on the same design:

$$\varepsilon' = \frac{\| \frac{d\tilde{F}}{d\boldsymbol{s}} - \frac{dF}{d\boldsymbol{s}} \|_2}{\| \frac{dF}{d\boldsymbol{s}} \|_2}. \tag{4.41}$$

Furthermore, a deviation in objective value Δf^* from the GT optimum characterized by f^*_{GT} is computed as:

$$\Delta f^* = \frac{f^* - f^*_{GT}}{f^*_{GT}}, \tag{4.42}$$

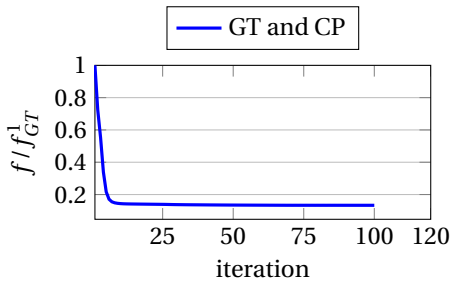
where a positive Δf^* is a deteriorated objective and a negative Δf^* is an improved objective.

APPROXIMATE ALGORITHM DESIGNS AND STABILITY

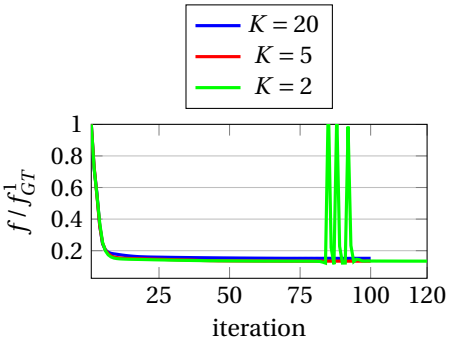
We compare the baseline design to the designs found by the approximate algorithms. To investigate the influence of the number of intervals K, we optimize for $K = 2$, 5, and 20. Optimal designs for the LT, CP/LT, and PLT algorithms can be found in Table 4.5, objective histories in Figure 4.10, and sensitivity error histories in Figure 4.11. The most striking difference in design is found for $K = 20$ using the LT and PLT algorithms in Tables 4.5a and 4.5b. These designs contain a large island around the upper influx domain which is disconnected from the heat sink. As can be seen by the increased objective values, this is an inferior local optimum with $\Delta f^* = 14\%$ for both the LT and PLT algorithms. The optimizer finds this inferior optimum due to the weak global convergence issues as discussed in Section 4.3.1. Whereas the intervals with $K = 20$ have a length of $\Delta\theta = t_t/K = 0.5$, the characteristic time of the upper domain in the GT design was found as $\tau_1 = 6.12$. The interval length is thus an order of magnitude smaller than the characteristic time. In fact, if we assume a static unchanging design, the adjoint error should propagate over $\tau_1/\Delta\theta \approx 12$ intervals. As adjoint errors can only reduce over one interval each design iteration, it would take 12 design iterations for adjoint errors around the top influx to reduce sufficiently. Moreover, as can be seen in the convergence plots of the objective in Figure 4.10, it takes around ten iterations for the design to stabilize. The LT and PLT designs thus stabilize into an inferior local optimum before adjoint errors sufficiently reduce. These results suggest that to prevent weak global convergence, the length of an interval should not be an order smaller than the characteristic time:

$$\Delta\theta = \frac{t_t}{K} > \frac{\tau}{10}, \tag{4.43}$$

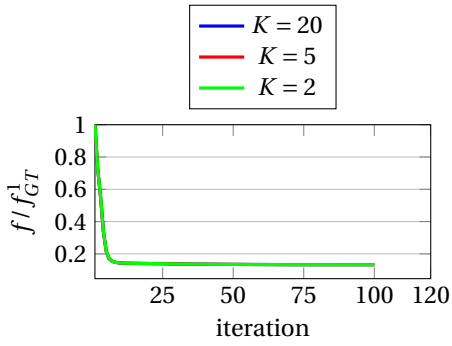
which we defined as the weak global convergence requirement.



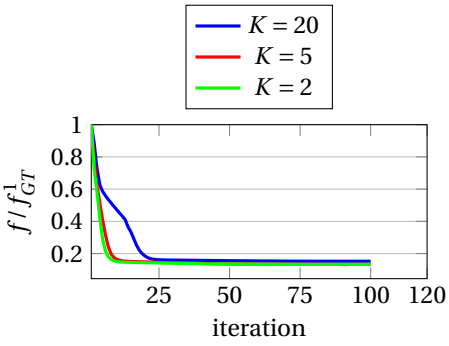
(a) GT and CP objective convergence.



(b) LT objective convergence.



(c) CP/LT objective convergence.



(d) PLT objective convergence.

Figure 4.10: The objective convergence for designs in Figure 4.9 and Table 4.5. Objective values are normalized using the objective in the first design iteration computed using the GT algorithm f^1_{GT}. The GT and CP algorithms both compute exact sensitivities and show the same convergence behavior. Additionally, as the CP/LT algorithm corrects erroneous sensitivities and computes fairly accurate sensitivities as can be seen in Figure 4.11, it also shows similar convergence behavior to the GT and CP algorithms. The objective convergence of the LT algorithm using $K = 2$ shows large fluctuations towards the end. These fluctuations are caused by global instabilities in the design. However, the global instabilities dampen out and the design again converges to a stable optimum. The PLT convergence plot shows that when more intervals are used, the objective converges slower over more design iterations. This is caused by the approximations in state variables which leads to larger changes in objective.

Other significant deviations from the reference are found in the $K = 5$ and $K = 2$ LT designs in Tables 4.5d and 4.5g and convergence in Figure 4.10b. Firstly, the LT $K = 2$ convergence plot shows some large instabilities during iterations 80 to 100. These errors are caused by global instabilities as discussed in Section 4.3.1. Small changes in design lead to large changes in adjoints and consequently in sensitivities. Subsequently, these sensitivity changes lead to larger changes in design and so forth. In Figure 4.11a, we compare the LT sensitivities to the ones computed using the GT algorithm on the same design. We verify the increase in sensitivity error over design iterations 80 to 100. Moreover, we find that sensitivity errors do not reduce monotonically, and increase in parts

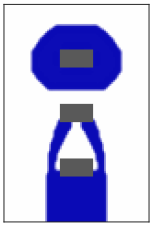
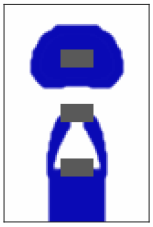
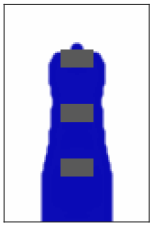
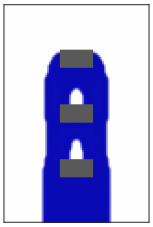
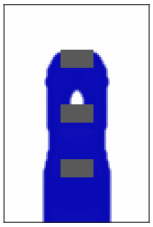
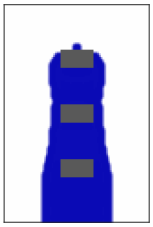
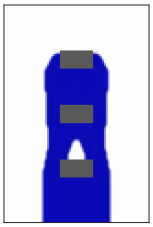
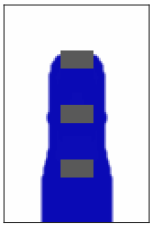
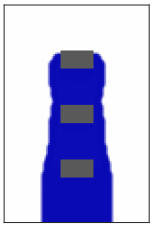
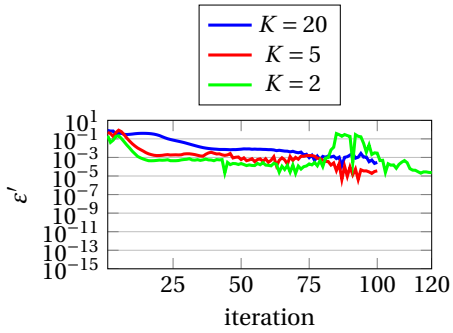
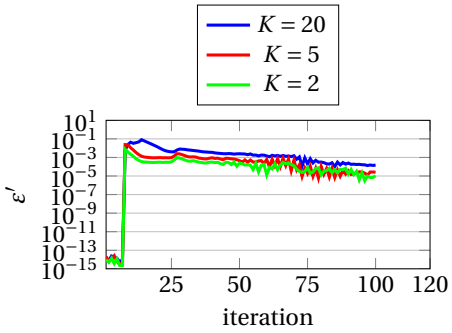
	LT	PLT	CP/LT
$K = 20$			
	(a) $f^* = 1.1671$, $\Delta f^* = 14\%$.	**(b)** $f^* = 1.1727$, $\Delta f^* = 14\%$.	**(c)** $f^* = 1.0266$, $\Delta f^* = -0.03\%$.
$K = 5$			
	(d) $f^* = 1.0290$, $\Delta f^* = 0.26\%$.	**(e)** $f^* = 1.0303$, $\Delta f^* = 0.40\%$.	**(f)** $f^* = 1.0266$, $\Delta f^* = -0.03\%$.
$K = 2$			
	(g) $f^* = 1.0299$, $\Delta f^* = 0.35\%$.	**(h)** $f^* = 1.0247$, $\Delta f^* = -0.16\%$.	**(i)** $f^* = 1.0250$, $\Delta f^* = -0.13\%$.

Table 4.5: The designs, objective values f^*, and relative changes in objective Δf^* found using the approximate sensitivity computation algorithms. The change in objective Δf^* is relative to the objective found using the GT and CP algorithms, where a positive Δf^* increases and a negative Δf^* deteriorates the objective. Both the LT and PLT algorithms disconnect the upper heat source from the heat sink for $K = 20$ leading to an inferior optimized design. These errors emerge as the adjoint errors do not converge quickly enough and the design gets stuck in an inferior local optimum. The CP/LT designs do not suffer from this problem as they correct the largest errors in adjoint.

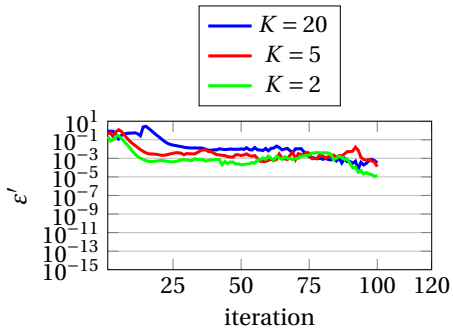
of the convergence history. This is also the case for the sensitivity error in the PLT and CP/LT algorithms in Figure 4.11. Due to the stability of MMA, the design is able to revert back to a stable design and converges after 20 additional iterations. We can conclude that an increase in sensitivity errors leading to global instabilities is difficult to predict but may happen and is a risk inherent to the LT and PLT algorithms.



(a) LT sensitivity error convergence.



(b) CP/LT sensitivity error convergence.



(c) PLT sensitivity error convergence.

Figure 4.11: The convergence of the sensitivity error ε' for all design iterations. In Figure 4.11a a clear correlation can be found between increasing sensitivity errors for LT, $K = 2$ and the global instabilities found in Figure 4.10b. Moreover, none of the figures show a monotonously decreasing sensitivity error as they all show some error fluctuations. These fluctuations are caused by global instabilities, but when they remain small have no noticeable effect on the objective convergence and design updates. The sensitivity errors in the CP/LT algorithms are of numerical precision in the first few design iterations where all intervals are corrected. However, as the algrotihm stops correcting all intervals, errors shoot up to about $\varepsilon' \approx 10^{-2}$.

The aim of the CP/LT algorithm is to correct the errors in the LT algorithm, and we expect it to be less prone to global stability issues. Table 4.5c shows that the CP/LT algorithm is able to find the bowling pin design for $K = 20$ and did not suffer from weak global stability issues. Moreover, no global convergence issues are found for the CP/LT algorithms in Figure 4.10c. In Figure 4.11b, the sensitivity error ε' for the CP/LT algorithm is shown. During the first seven iterations, all domains are corrected and the error in sensitivity is close to numerical accuracy. When corrections stop in the eighth design iteration, an error in sensitivities is introduced and reduces over the subsequent iterations. Furthermore, the CP/LT results illustrate that the approximate algorithms find optima which are slightly different but not necessarily inferior to the ones found using exact algorithms. The designs found by the CP/LT algorithm for $K = 5$ and $K = 2$ perform slightly better than the GT and CP design, as can be seen by the negative Δf^*. To summarize, global convergence problems are alleviated by the CP/LT algorithm, and the

algorithm generally shows a more stable convergence behavior than the LT algorithm.

Finally, we compare the designs and convergence behavior of the PLT algorithm to the other algorithms. As previously discussed, the PLT algorithm suffers from weak global convergence issues for $K = 20$. Moreover, the convergence of the objective for the PLT algorithm is much slower than for the other algorithms, see Figure 4.10d, even when few intervals are used. This issue is caused by the errors in state variable which require more iterations to converge together with the adjoint variables. It should be noted that the PLT algorithm only requires more time to reach a stable topology and objective value during the start of the optimization procedure. Subsequently, the PLT algorithm only slightly adapts the topology and improves the objective value. Whereas other algorithms take about ten iterations, the PLT algorithm with $K = 20$ takes 20 to find an adequate topology. As the optimizer runs for at least 100 design iterations, this does not have a large influence on the overall design convergence.

COMPUTATIONAL COST OF THE THERMAL RESULTS

Although the thermal problem is specifically designed to investigate stability, we also analyze the computational wall times found in Table 4.6. We show the wall time to compute sensitivities summed over all design iterations $t_{solv} = t_{state} + t_{adj}$, which consists of the time required to solve the state solution t_{state} and the time to update the adjoint and sensitivity t_{adj}. We note that for the measured wall time we summed the times for all design iterations in the optimization process. The serial algorithms were all run on one CPU, while the PLT algorithm was run on the same number of CPUs as intervals K.

	GT	CP $K = 20$	LT $K = 20$	LT $K = 5$	LT $K = 2$
t_{solv} (min)	117.0	150.3	118.9	114.6	138.0

	CP/LT $K = 20$	CP/LT $K = 5$	CP/LT $K = 2$	PLT $K = 20$	PLT $K = 5$	PLT $K = 2$
t_{solv} (min)	132.8	127.3	127.0	7.698	25.66	64.99

Table 4.6: The computational time $t_{solv} = t_{state} + t_{adj}$ is the sum over all design iteration and consists of the time to solve the state equations t_{state} and the time to solve adjoint equations while simultaneously updating sensitivities t_{adj}.

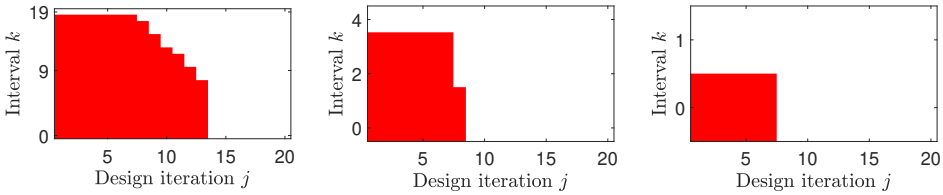
Among the serial algorithms, the GT and LT $K = 20$ and $K = 5$ are the fastest. However, as the LT algorithm using $K = 2$ required 20 additional design iterations due to the strong global stability issues, it required more computational time. Compared to the increased computational time in the CP/LT algorithms, the LT algorithm with $K = 2$ performed worse due to the additional iterations, even though computational time for the CP/LT algorithm is increased by performing the adjoint corrections. The worst performing algorithm was the CP algorithm due to the recomputation of the state solutions. The parallel PLT algorithm was run using the same number of CPUs as there were intervals K. Each interval was thus attributed to one CPU. We observed a large decrease in computational time using more intervals as for $K = 20$ the PLT algorithm took about one

fifteenth of the time of the GT algorithm. A perfect scaling of one twentieth of the time is not achieved here and will be further investigated in Section 4.5.2.

We verify the relative computational costs predicted in Section 4.4.2. By assuming that the measured wall times are proportional to the computational cost ($t_{state} \propto c_s$, $t_{adj} \propto c_a$), relative computational cost $r_c = c_a/c_s = t_{state}/t_{adj}$ is approximated using the computational times of GT: $r_c = t^{GT}_{adj}/t^{GT}_{state} = 84.19/32.83 \approx 2.56$. Since both state and adjoint updates are simple matrix vector multiplications, they have the same cost. Because the sensitivity computation is also a matrix vector multiplication, it has a similar cost. This leads to a larger c_a which contains both adjoint solve and sensitivity computation, compared to c_s which contains only the state solve. Subsequently, we compare the analytical relative computational cost of the CP algorithm in Equation 4.27, to the measured computational cost, respectively as:

$$\frac{C^{CP}}{C^{GT}} \approx \frac{2 + r_c - 1/K}{1 + r_c} = 1.27, \qquad \frac{C^{CP}}{C^{GT}} = \frac{t^{CP}_{solv}}{t^{GT}_{solv}} = \frac{150.3}{117.0} = 1.28, \tag{4.44}$$

which is in agreement with our predictions in Equation 4.27.



(a) $K = 20$ **(b)** $K = 5$ **(c)** $K = 2$

Figure 4.12: The intervals Θ_k, that require corrections for the CP/LT algorithm are flagged in red. The sensitivities are corrected only during the first part of the procedure. when more intervals are used, we require corrections in more design iterations, leading to increased computational times as shown in Table 4.6. Moreover, we find that starting from the terminal interval less and less intervals are corrected. This is caused by the fact that the terminal adjoint value is exact and this correction slowly propagates throughout the system.

A disadvantage of the corrections performed by the CP/LT algorithm is that they increase computational cost. In Table 4.6, we find that the GT algorithm takes 117.0 minutes to compute sensitivities, while the CP/LT algorithm takes at least 10 minutes more. The more intervals are used, the more computational time is increased. The increase in computational time for more intervals is caused by the additional time spent in adjoint corrections. This remark is supported by the number of intervals in each design iteration for which a correction is carried out as illustrated in Figure 4.12. For $K = 20$, 5, and 2, corrections are only performed in the first 12, 9, and 8 design iterations, respectively. All intervals except the last are corrected the first seven iterations for each K. At the eight iteration, the design stabilizes and the number of corrected intervals reduces. As the correct terminal adjoint propagates backward-in-time, fewer intervals need to be

corrected. Intervals which are not corrected start at the terminal interval and propagate backward-in-time. Since the correct terminal adjoint can only propagate one interval at a time, using more intervals leads to a slower decrease in adjoint error, a larger number of performed corrections, and an increased computational time.

The observation that corrections are performed only in the first iterations can be used to analyze the relative computational cost of CP/LT *a priori*. A designer with some knowledge of their optimization problem can estimate how many iterations are required for the design to stabilize. Subsequently, the percentage of corrected intervals f_c, as defined in Section 4.4.2, can be estimated as the percentage of unstable to stable design iterations. Equation 4.29 is then used to compare the CP/LT and CP performance. We verify Equation 4.29 for CP/LT $K = 20$. First, we compute the average percentage of corrected intervals over all design iterations, as shown in Figure 4.12a, as $f_c = 0.112$. Using Equation 4.29 and comparing it to the measured relative computational cost, we find respectively:

$$\frac{C^{CP/LT}}{C^{CP}} \approx \frac{K-1}{K}(f_c+1)\frac{1+r_c}{2+r_c}+\frac{1}{K} = 0.875, \qquad \frac{C^{CP/LT}_{comp}}{C^{CP}_{comp}} = \frac{t^{CP/LT}_{solv}}{t^{CP}_{solv}} = \frac{132.8}{150.3} = 0.884, \quad (4.45)$$

which is in agreement with our predictions in Equation 4.29.

4.5.2. COMPUTATIONAL COST INVESTIGATION THROUGH FLOW OPTIMIZATION

In this section, we compare the computational cost and the computational wall time of the presented algorithms. A transient flow problem is optimized as the nonlinear flow physics increase the complexity and consequently the computational effort. Furthermore, the problem is used to emphasize the difference between adjoint updates which are linear by nature and state updates which may be linear or nonlinear. Subsequently, the speedup of the PLT algorithm using multiple computational nodes is investigated. Additionally, we examine local/global stability and design convergence. In the PLT speedup investigation, we verify the weak global convergence requirement found in Equation 4.43.

TRANSIENT FLOW MODEL

To ensure stability of the approximate algorithms, we investigate the characteristic timescale of the flow model, its discretization, and solution scheme. We discretize the transient equations in space and time. A two-dimensional transient flow problem is examined and is defined on temporal domain $t \in [0, t_t]$ and spatial domain $\vec{x} \in \Omega$ with

boundary Γ. The flow problem is governed by the Navier-Stokes equations:

$$\begin{aligned}
\rho_f \dot{\mathbf{v}} + \rho_f \nabla \cdot \left(\mathbf{v}\mathbf{v}^\intercal\right) &= -\nabla p + \mu_f \nabla^2 \mathbf{v} - \frac{k_f(\alpha_f)}{\alpha_f}\mathbf{v}, && \text{on } \Omega, \\
\nabla \cdot \mathbf{v} &= 0, && \text{on } \Omega, \\
\mathbf{v} &= \mathbf{v}_\Gamma, && \text{on } \Gamma_d^{\mathrm{v}}, \\
p &= p_\Gamma, && \text{on } \Gamma_d^{p}, \\
\mathbf{v} &= 0, && \text{on } \Gamma_w, \\
\mathbf{v} = \hat{\mathbf{v}},\ p, &= \hat{p}, && \text{at } t = 0,
\end{aligned} \tag{4.46}$$

where the subscript $\square_f$ denotes fluid material properties, $\mathbf{v}(\vec{x}, t) = [u, v]^\intercal$ is the velocity field with time derivative $\dot{\mathbf{v}} = \partial \mathbf{v}/\partial t$ and component u in x-direction and v in y-direction, $p(\vec{x}, t)$ is the pressure field, $\alpha_f(\vec{x})$ is the design field which represents the fluid volume fraction and is thus set to $\alpha_f = 1$ in the fluid domain. Furthermore, ρ_f and μ_f are the fluid density and dynamic viscosity, respectively, and $k_f(\alpha_f)$ is the Darcy impermeability used to inhibit flow in the solid domain where the fluid volume fraction tends to zero $\alpha_f \to 0$. Furthermore, $\mathbf{v}_\Gamma(\vec{x})$ and $p_\Gamma(\vec{x})$ are the fixed flow and pressure on boundary Γ_d^{v} and Γ_d^{p}, respectively, the flow at the solid/fluid interface Γ_w is prescribed as $\mathbf{v} = \mathbf{0}$, and $\hat{\mathbf{v}}(\vec{x})$ and $\hat{p}(\vec{x})$ are the initial flow and pressure distributions. The Navier-Stokes equations are discretized using the finite volume method as described by Theulings et al. (2023) on a staggered grid as shown in Figure 4.13. Moreover, they are solved using the SIMPLE

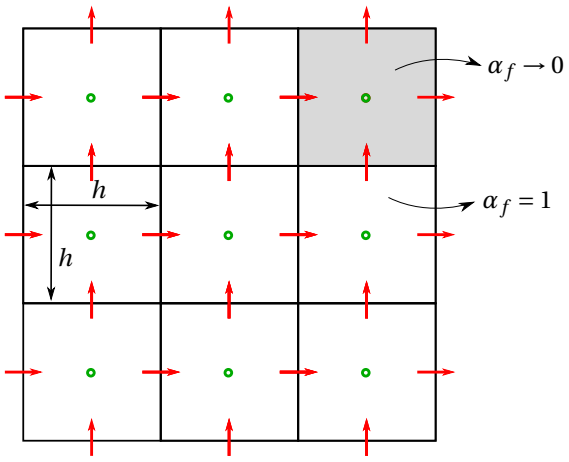


Figure 4.13: The equidistant uniform mesh used to discretize the flow problem. Green circles are the pressure nodes and densities are defined per element where $\alpha_f = 1$ is a fluid element and $\alpha_f \to 0$ is a solid element. Horizontal red arrows are the u-velocity DOFs while vertical red arrows represent the v-velocity DOFs.

algorithm as described by Versteeg and Malalasekera (2007). As the SIMPLE algorithm is

an implicit algorithm, the equations take the form:

$$\begin{aligned} R_n^m(\mathbf{v}_{n-1},\mathbf{v}_n,\boldsymbol{p}_n,\boldsymbol{s},t_n) &= \boldsymbol{M}^t\frac{\mathbf{v}_n-\mathbf{v}_{n-1}}{\Delta t}+\boldsymbol{M}^c(\mathbf{v}_n)\mathbf{v}_n+\boldsymbol{C}^p\boldsymbol{p}_n-\boldsymbol{D}\mathbf{v}_n+\boldsymbol{K}(\boldsymbol{s})\mathbf{v}_n,\\ R_n^c(\mathbf{v}_n) &= \boldsymbol{C}^c\mathbf{v}_n,\\ R_0^v(\mathbf{v}_0) &= \mathbf{v}_0-\hat{\mathbf{v}},\\ R_0^p(\boldsymbol{p}_0) &= \boldsymbol{p}_0-\hat{\boldsymbol{p}}, \end{aligned} \tag{4.47}$$

where R_n^m and R_n^c are the discretized momentum and continuity equations, respectively, and $\mathbf{v}_n$, $\boldsymbol{p}_n$ are vectors of discrete nodal velocities and pressures at time step t_n with initial condition $\hat{\mathbf{v}}$ and $\hat{\boldsymbol{p}}$, respectively. As can be seen by the dependence of the convective matrix $\boldsymbol{M}^c$ on $\mathbf{v}_n$, the momentum equations are nonlinear and several subsolves are required to advance one time step in the SIMPLE algorithm. Each subsolve includes the assembly of the *pressure correction matrix* and subsequent solution of the pressure correction which involves a linear matrix solve. However, a backward adjoint step only involves one matrix assembly and one solve as the adjoint equations are linear by nature. Solving a backward adjoint step is thus cheaper than solving a forward state step.

To ensure stability of the approximate algorithms and the time stepping scheme, we estimate the characteristic timescale of the physics and discretization scheme. For the SIMPLE algorithm to converge, the time step is constrained by the Courant-Friedrichs-Lewy (CFL) condition as:

$$\Delta t < \frac{h}{\overline{u}}, \tag{4.48}$$

where $\overline{u}$ is the maximum flow magnitude. Furthermore, the bound on the time step is related to a characteristic timescale. If we imagine a parcel of fluid flowing over a distance L with velocity $\overline{u}$, it travels this distance in:

$$\tau^\rho = \frac{L}{\overline{u}}. \tag{4.49}$$

The characteristic timescale for flow can be estimated by τ^ρ, which is in fact the characteristic timescale of inertia (Picioreanu et al., 2000). However, in the Navier-Stokes equations, another characteristic time is defined by the viscous dissipation in Equation 4.46:

$$\tau^\mu = \frac{L^2}{\nu_f}, \tag{4.50}$$

where $\nu_f = \mu_f/\rho_f$ is the diffusivity of momentum, more commonly called the kinematic viscosity. Whereas τ^ρ is related to the characteristics of the flow, τ^μ is related to the characteristics of the viscous energy dissipation. Although no local stability criterion for nonlinear state equations was derived in Section 4.3.1, we generally found the adjoint equations to be stable when the constraint in Equation 4.48 was applied. Furthermore, for strong/weak global stability, the characteristic timescale in Equations 4.49 and 4.50 will be considered in the problem setup.

For the thermal problem in Section 4.5.1, the time step constraint and characteristic times at the element scale were similar. However, they are not necessarily directly related for the flow problem. We thus investigate the possible length scales of features

in the design, and use it to find the characteristic times of these features. In topology optimization, we may design features of a few elements $L \approx h$ and are interested in characteristic times of:

$$\tau_h^\rho \approx \frac{h}{u}, \qquad \tau_h^\mu \approx \frac{h^2}{\nu_f}, \tag{4.51}$$

where τ_h^ρ is already satisfied by the stability condition in Equation 4.48. However, τ_h^μ may be smaller than τ_h^ρ, and in design components of a few elements long viscous effects may dominate. A smaller time step may thus be needed to accurately represent the transient effects in these design features. One could think of a design with many narrow channels with low flow speeds and low Reynolds numbers. It is left to the user to make an *a priori* estimation of the type of design resulting from the optimization procedure and select the appropriate time steps.

TRANSIENT FLOW OPTIMIZATION PROBLEM

The problem in Figure 4.14 with parameters in Table 4.7 is designed such that nonlinear flow effects are of importance for the optimized designs. We design a piston pump without moving parts. A piston pumps fluid up and down at the top boundary Γ_{pump}, where an oscillating flow is applied as:

$$\begin{aligned} v_p(t) &= \overline{v}\sin(4\pi t), \\ u_p(t) &= 0, \end{aligned} \tag{4.52}$$

with $\overline{v}$ the maximum inlet velocity in y-direction over time as found in Table 4.7. We optimize the mass flow to the right from inlet Γ_{in} to outlet Γ_{out}, and apply a constant pressure on both the inlet and outlet $p_\Gamma = 0$. During the downstroke of the piston, we

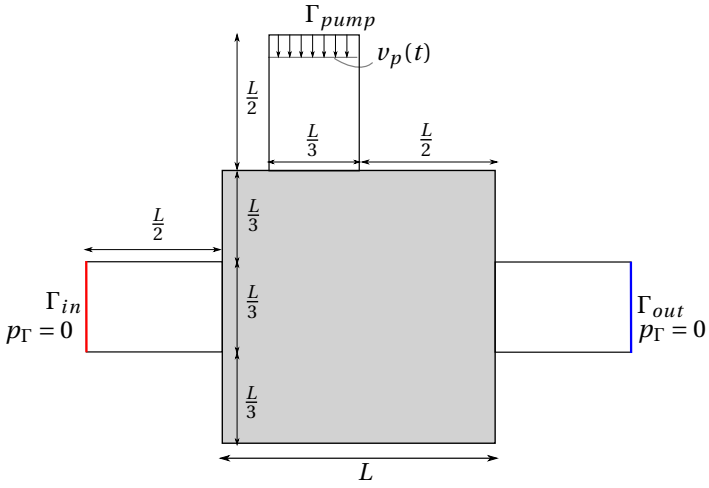


Figure 4.14: The piston pump optimization problem. At the top inlet/outlet, a fluctuating velocity is prescribed and at the red inlet and blue outlet only reference pressures are prescribed. The white domains are non-design areas, and in the gray domain, the material layout is optimized.

L	h	t_t	Δt	ρ_f	μ_f	$\overline{v}$
1	$\frac{1}{42}$	5	$\frac{1}{42}$	1	$\frac{1}{60}$	1

Table 4.7: The discretization and material parameters used for the flow optimization problem in Figure 4.14.

expect fluid to be pushed out of the system through Γ_{out}; while during the upstroke, we expect fluid to be pulled into the systems through Γ_{in}. Subsequently, we maximize the volumetric flow through the system from inlet Γ_{in} to outlet Γ_{out}:

$$F = \int_0^{t_t} \left(-\int_{\Gamma_{in}} \mathbf{v}\cdot\boldsymbol{n} d\Gamma + \int_{\Gamma_{out}} \mathbf{v}\cdot\boldsymbol{n} d\Gamma \right) dt, \tag{4.53}$$

where $\boldsymbol{n}$ is the outward normal on the boundaries. To normalize the problem, we define a theoretical maximum of a piston pump with moving parts which closes off the inlet during the downstroke and closes off the outlet during the upstroke. In such a pump, the volumetric flow to the right through the in- and outlet is the same as the absolute volumetric flow through Γ_{pump}:

$$\overline{F} = \int_0^{t_t} \int_{\Gamma_{pump}} |\overline{v}\sin(4\pi t)| d\Gamma dt = \overline{v}\frac{L}{3}\frac{10}{\pi}. \tag{4.54}$$

Subsequently, we use the theoretical optimum and minimize for the normalized objective $F_p = (\overline{F} - F)/\overline{F} = 1 - F/\overline{F}$. The normalized objective can be interpreted as the decrease in efficiency of the pump without moving parts compared to a pump with moving parts. Furthermore, using the material parameters in Table 4.7, the Reynolds number is estimated as:

$$Re = \frac{\rho_f \overline{v} L}{\mu_f} = 60. \tag{4.55}$$

The optimal design is expected to use the nonlinearity of moderate Reynolds flow.

r	$\underline{\beta}$	$\Delta\beta$	$\overline{\beta}$	V_f	q	$\underline{\alpha}_f$	$\varepsilon_\lambda^{max}$
$3h$	1	0.14	8	0.6	2	10^{-1}	0.1

Table 4.8: The optimization parameters used for the flow problem in Figure 4.14. We smooth the design over a radius of three elements ($r = 3h$) and increase the Heaviside projection slope from $\underline{\beta}$ to $\overline{\beta}$ by $\Delta\beta$ over iterations 10 to 60. Furthermore, a maximum allowable adjoint error of $\varepsilon_\lambda^{max}$ is set for the CP/LT algorithm.

To regularize the optimization procedure, filters and interpolation functions are used. Firstly, design variables s_i are smoothed (Bruns & Tortorelli, 2001) and projected (F. Wang et al., 2011) following the approach described in Section 4.5.1 resulting in $\tilde{s}_i$. The volume constraint $g_v(\tilde{\boldsymbol{s}})$ is applied using Equation 4.37. The interpolation of the fluid volume fraction α_f, the Darcy impermeability $k_f(\alpha_f)$ and the maximum impermeability $\overline{k}_f$ are defined following Theulings et al. (2023). For both α_f and k_f, a linear

interpolation is used:

$$\begin{aligned} \alpha_f(\tilde{s}_i) &= \underline{\alpha}_f + (1-\underline{\alpha}_f)\tilde{s}_i, \\ k_f(\alpha_f) &= \overline{k}_f(1-\alpha_f), \end{aligned} \tag{4.56}$$

such that in the fluid domain ($\alpha_f = 1$), no penalization ($k_f = 0$) is imposed and in the solid domain ($\alpha_f = \underline{\alpha}_f \to 0$), the maximum penalization ($\overline{k}_f(1-\underline{\alpha}_f) \approx \overline{k}_f$) is prescribed. Moreover, the maximum penalization $\overline{k}_f$ is set using parameter $H^e = h^{-2}$ and the elemental Reynolds number $Re^e = \rho_f \overline{v} h/\mu$:

$$\overline{k}_f = 10^q \begin{cases} \mu H^e, & Re^e \leq 1, \\ \mu H^e Re^e, & Re^e > 1. \end{cases} \tag{4.57}$$

Parameter q is used to set the magnitude of the flow penalization. All optimization parameters can be found in Table 4.8. Furthermore, using the material parameters in Table 4.7, the elemental Reynolds number is computed as:

$$Re^e = \frac{60}{42}, \tag{4.58}$$

resulting in a maximum penalization of:

$$\overline{k}_f = 10^q \mu_f H^e Re^e = 4200. \tag{4.59}$$

Finally, using the discrete model in Equation 4.47, the discrete optimization problem is stated as:

$$\boxed{\begin{array}{lll} \underset{\boldsymbol{s}}{\text{minimize}} & F_p \approx \sum_{n=0}^{N-1} F_n(\mathbf{v}_n, \boldsymbol{s})\Delta t & \\ \text{subject to} & R_n^m(\mathbf{v}_{n-1}, \mathbf{v}_n, \boldsymbol{p}_n, \boldsymbol{s}) = 0 & n \in \{1,2,\dots N\}, \\ & R_n^c(\mathbf{v}_n) = 0 & n \in \{1,2,\dots N\}, \\ & R_0^v(\mathbf{v}_0), & \\ & R_0^p(\boldsymbol{p}_0), & \\ & g_v(\boldsymbol{s}) \leq 0, & \end{array}} \tag{4.60}$$

Characteristic times are inspected to investigate the weak global convergence requirement in Equation 4.43. First, we compare the characteristic time of the flow and of the viscous diffusion:

$$\begin{aligned} \tau^\rho &= \frac{L}{\overline{v}} = 1, \\ \tau^\mu &= \frac{L^2}{\nu_f} = 60, \end{aligned} \tag{4.61}$$

where we estimated the maximum flow speed as the maximum flow at the inlet $\overline{v}$, and substituted $\nu_f = \mu_f/\rho_f$ using the values in Table 4.7. Since the characteristic time of the

viscous diffusion is an order of magnitude higher than that of the inertia, and we examine a problem with terminal time $t_t = 5\tau^\rho$, we expect transient effects of inertia to be dominant. Furthermore, weak global convergence issues emerged in Section 4.5.1 when Equation 4.43 was not satisfied. We thus expect that they may occur in this problem when the interval length $\Delta\theta = t_t/K$ is an order of magnitude smaller than τ^ρ and thus:

$$K > \frac{10t_t}{\tau^\rho} = 50. \tag{4.62}$$

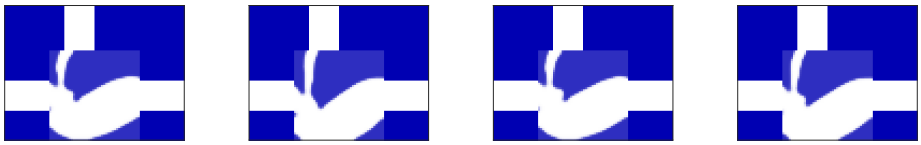
In this section, we will use PLT with $K = 48$ and may thus find weak global convergence issues. Strong global convergence issues are harder to predict and may occur. In the authors experience, convergence is generally achieved when sufficient design iterations are used.

Similar to the analysis in Section 4.5.1, we estimate the memory requirements of the problem and select the appropriate number of intervals K. As the CFL condition in Equation 4.48 defines an upper bound on the time step dependent on h, the number of time steps and memory requirements are dependent on the spatial resolution. The number of flow and pressure DOFs in this problem is approximately $N_d \approx 7500$. As each DOF is stored as a double, the size of the state solution at a time step is $m = 8N_d \approx 60$ kB. Furthermore, using the parameters in Table 4.7, we store the state solution at $N = t_t/\Delta t = 210$ time steps. Assuming that $K \ll N$, memory requirements following Table 4.1 scale as $M = mN \approx 0.13$ GB for the GT and PLT algorithms and as $M = mN/K \approx 0.13/K$ GB for the CP, LT, and CP/LT algorithms. Although for this relatively small two-dimensional problem, there is no need for memory reducing algorithms, memory requirements scale up quickly when three-dimensional problems are investigated and the mesh is refined. For the analysis of the computational cost, we will use $K = 8$ intervals.

COMPUTATIONAL COST COMPARISON

In Section 4.4.2 Table 4.2, we made a theoretical approximation of the computational cost of the algorithms. In this section, we compare these theoretical approximations with measured computational costs. All computations of the serial algorithms are carried out on the same type of CPU. Moreover, to quantify computational cost, we measure wall time. Since the same type of CPUs with similar computational power are used, we expect wall time to be a sufficiently accurate measure of computational cost. However, for the PLT algorithm the same number of CPUs as there are intervals is used, contrary to the serial algorithms which use only one CPU. As the PLT algorithm uses more computational power, we expect a significant speedup.

Different designs lead to different flow solutions which require a different computational effort to solve. To compare speedup, we would like the resulting designs to be similar. As $K = 8$ intervals are used and since we expect global convergence issues to start at $K = 50$ intervals as shown in Equation 4.62, we expect similar designs to be found by all algorithms. This is confirmed by the designs in Figure 4.15. However, as shown in Figure 4.16a, the different algorithms take a different convergence path to their local optima. During the design convergence, the computational cost of solving the nonlinear state equations may differ at any given design iteration. This effect is responsible for some small deviations between expected and measured wall times.



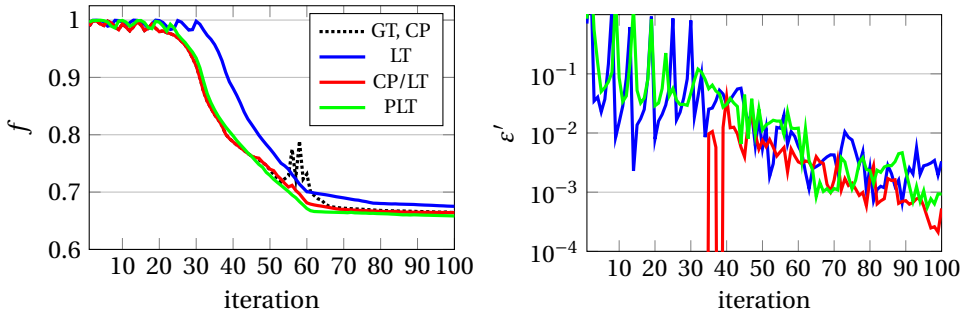
(a) GT and CP, $f^* = 0.6645$, $\Delta f^* = 0\%$.

(b) LT, $f^* = 0.6752$, $\Delta f^* = 1.61\%$.

(c) CP/LT, $f^* = 0.6643$, $\Delta f^* = -0.03\%$.

(d) PLT, $f^* = 0.6584$, $\Delta f^* = -0.92\%$.

Figure 4.15: The optimal designs, objectives f^*, and relative changes in objective Δf^* for the flow problem found using the GT, CP, LT, CP/LT, and PLT algorithms. For $K = 8$ intervals, no weak global stability problems are encountered and all designs and objective values are similar.



(a) The objective convergence.

(b) The sensitivity error convergence.

Figure 4.16: The objective and sensitivity error convergence of the flow designs in Figure 4.15. The GT and CP algorithms both compute exact sensitivities and show the same convergence behavior. After the initial fluctuations in design and objective during the first 30 iterations, the design stabilizes and the sensitivity errors reduce.

To verify stability and convergence, we further analyze the objective and sensitivity error convergence in Figure 4.16. We note that the objective is increased over iterations 55 to 60 for the GT and CP algorithms. This is caused by the nonlinearity of the flow solution leading to an increase in the objective. Moreover, as the CP/LT algorithm corrects erroneous sensitivities and computes exact sensitivities in the first 34 iterations as can be seen in Figure 4.16b, it follows the same convergence path as the GT and CP algorithms during these iterations. Furthermore, during the first 20 to 30 iterations, the design is subject to significant change and sensitivity errors remain high as shown in Figure 4.16b. This coincides with the relatively high and fluctuating objective values found in Figure 4.16a. Once the topology stabilizes and only the shape of the design changes, the objective and sensitivity errors start to decrease. Due to the nonlinearity of the equations, the objective and sensitivity errors do not decrease monotonically and may fluctuate significantly for both exact and approximate algorithms. However, given sufficient design iterations, the objective generally stabilizes and sensitivity errors reduce.

Subsequently, we examine and compare the computational cost of the algorithms. Firstly, the relative computational cost of the CP algorithm in Equation 4.27 is investigated. In Figure 4.17, we show three wall times: the time to solve the state equations t_{state}, the time to update the adjoint while simultaneously updating sensitivities t_{adj}, and $t_{solv} = t_{state} + t_{adj}$. We note that for t_{state} and t_{adj} we summed the wall times for each of the 100 design iterations in the optimization process. Wall times $t_{state} \propto c_s$ or $t_{adj} \propto c_a$ are used as measures of the computational cost of solving state or solving adjoint and computing sensitivities, respectively. Using the measured wall time of the GT algorithm in Figure 4.17b, we compute the relative computational cost as: $r_c = t^{GT}_{adj}/t^{GT}_{state} = 14.81/149.0 \approx 0.10$. Solving the state equations is more expensive than solving the adjoint equations as shown in Figure 4.17a. We compare the derived relative cost of the CP algorithm in Equation 4.27 to the measured relative cost, respectively as:

$$\frac{C^{CP}}{C^{GT}} \approx \frac{2 + r_c - 1/K}{1 + r_c} = 1.80, \qquad \frac{C^{CP}_{comp}}{C^{GT}_{comp}} = \frac{t^{CP}_{solv}}{t^{GT}_{solv}} = \frac{308.4}{163.8} = 1.88, \tag{4.63}$$

which shows a relatively low error of 8% between our approximation and the measure relative cost. Furthermore, t_{solv} is mainly increased by a growing t_{state} as shown in Figure 4.17a. Reducing the memory by using more intervals significantly increases the computational time required for the CP algorithm.

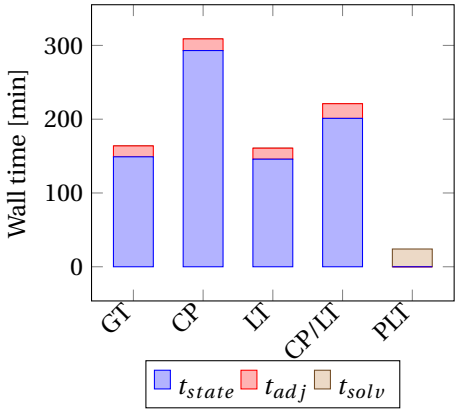
Secondly, we compare the computational cost of the LT algorithm to the GT and CP algorithms. Notably, the LT algorithm performed similarly to the GT algorithm. The LT algorithm is even slightly faster than the GT algorithm. This may be attributed to the fact that a different convergence path is followed by the LT algorithm as shown in Figure 4.16a with the objective convergence graph and in Figure 4.15b with the slightly different design layout. Different designs have different flow solutions which may take more or less time to solve.

Thirdly, the computational cost of the CP/LT algorithm is analyzed. We expected the CP/LT algorithm to have a cost between the GT and CP algorithms. Figure 4.18 depicts the corrected intervals during the CP/LT run. During the first 34 design iterations, all intervals except the last are corrected. Hereafter, the number of corrected intervals reduced and after 39 design iterations, no intervals were corrected. Even though for 39 of the 100 design iterations corrections are performed and $f_c = 0.38$, the computational cost remains lower than the cost of the CP algorithm, as the cost of computing the state solution is dominant and the cost of performing the adjoint corrections is relatively low ($r_c \approx 0.10$). We verify the relative computational cost of CP/LT to CP in Equation 4.29 to the measured cost, respectively as:

$$\frac{C^{CP/LT}}{C^{CP}} \approx \frac{K-1}{K}(f_c + 1)\frac{1 + r_c}{2 + r_c} + \frac{1}{K} = 0.78, \qquad \frac{C^{CP/LT}_{comp}}{C^{CP}_{comp}} = \frac{t^{CP/LT}_{solv}}{t^{CP}_{solv}} = \frac{220.9}{308.4} = 0.72, \tag{4.64}$$

which show that our measurements are in agreement with our approximations. As expected, although the CP/LT algorithm is more expensive than the LT algorithm, it outperforms the CP algorithm.

Finally, we compare the PLT algorithm to the other algorithms. While all other algorithms are run on one CPU, the PLT algorithm is run on eight CPUs. Furthermore, for



(a) A visualization of the wall times.

		GT	CP	LT	CP/LT	PLT
t_{solv}	[min]	163.8	308.4	160.3	220.9	24.07
t_{state}	[min]	149.0	293.0	145.8	201.1	-
t_{adj}	[min]	14.81	15.37	14.54	19.77	-

(b) The measured wall times.

Figure 4.17: The wall times t_{solv}, t_{state} and t_{adj} for the designs in Figure 4.15. The PLT algorithm was ran on 8 CPUs instead of 1 and consequently has a much lower t_{solv}. For the PLT algorithm some workers might still be working on the state solution while others are already working on the adjoint and we thus measure t_{solv} as the time from the start until the end of the sensitivity computation.

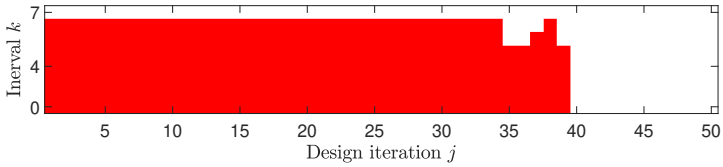


Figure 4.18: The intervals Θ_k, that require corrections for the CP/LT algorithm are flagged in red. Compared to the corrected intervals for the thermal problem in Figure 4.12, more corrections are performed. This is caused by a slower stabilization of the design and more significant changes in design over the first 30 iterations. After a stable topology is found, errors in adjoint reduce over design iterations 30 to 39. Since more corrections are performed in the 37^{th} and 38^{th} designs than in the 35^{th} and 36^{th} designs, adaptively correcting intervals helps in maintaining low sensitivity errors when design changes are large and disturb sensitivities.

the PLT algorithm some workers may still be working on the state solution while others are already working on the adjoint solution. Consequently, we do not measure t_{state} or t_{adj} separately. For the PLT algorithm we measure wall time t_{solv} starting when sending initial states and terminal adjoint to all intervals, and ending after retrieving terminal states, initial adjoints, and partial sensitivities $dF_k^*/d\boldsymbol{s}$, which are combined into total sensitivities $dF^*/d\boldsymbol{s}$. Each of the $K = 8$ intervals is attributed to one CPU. Since the PLT

algorithm was allowed to use more computational power, the measured wall times are not a fair comparison of computational cost. Moreover, comparing the speedup with respect to the GT algorithm, we find it to be $t_{solv}^{GT}/t_{solv}^{PLT} \approx 6.8$. The speedup does not scale linearly with the computational power, which will be further examined. However, as illustrated in Figure 4.17a, the increased computational power resulted in a large decrease in computational time.

PLT SPEEDUP

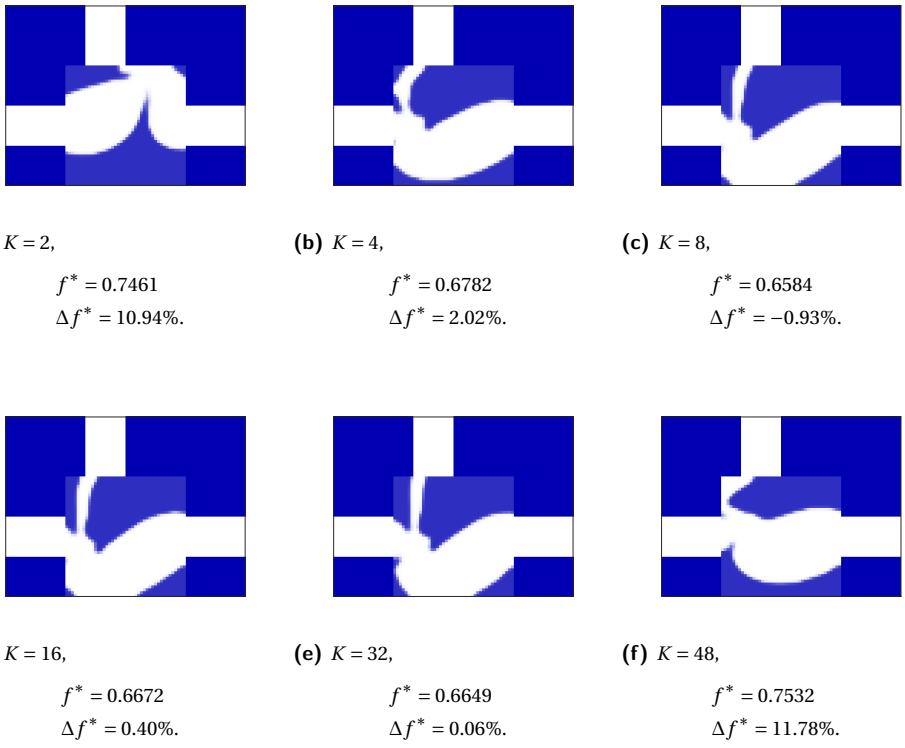
In the previous section, we found that the speedup of the PLT algorithm does not scale linearly with the number of intervals and computational workers K. In this section, we will further examine the computational speedup and causes of slowdown for the PLT algorithm. There are two main causes for computational slowdown: communication overhead and load balancing. Communication overhead is defined as the idle time of computational workers waiting for data transfer. Load balancing issues are caused by an unequal distribution of tasks over computational workers, resulting in idle times on the workers with cheap computational tasks.

The speedup is examined for $K = 2, 4, 8, 16, 32, 48$ intervals. We investigate the designs and objectives to verify if the results can be used for a fair comparison of speedup. Two designs significantly differ in shape and objective value from the GT design. The $K = 2$ and $K = 48$ designs, as shown in Figure 4.19, display a more than 10% lower performance compared to the GT design. Moreover, the different designs have different flow solutions which lead to different computational costs. We measure the average computational time over five state solves in the optimal GT and PLT $K = 2$, $K = 48$ designs as $\bar{t}_{state} = 86.0$, $\bar{t}_{state} = 84.6$, and $\bar{t}_{state} = 92.5$ seconds, respectively. The $K = 48$ design requires significantly more time than the GT design. Moreover, the $K = 48$ design needs 20 additional iterations to converge due to global stability issues as we approach the predicted stability limit in Equation 4.62 of $K = 50$ intervals. Comparing the speedup between two different designs is inherently unfair and is a limitation of our methods to measure speedup. However, since the $K = 4, 8, 16, 32$ designs are similar in shape and objective to the GT design, a comparison is carried out based on these results.

K	1 (GT)	2	4	8	16	32	48
t_{solv} (min)	163.8	90.60	43.87	24.00	13.07	7.68	6.90
t_{ov}/t_{solv}	-	0.162%	0.269%	0.574%	1.65%	3.48%	4.52%
LBS	-	0.560%	1.95%	8.23%	11.5%	23.7%	29.2%

Table 4.9: The wall time t_{solv}, and percentage of slowdown caused by overhead (t_{ov}/t_{solv}) and load balancing (LBS) for using an increasing number of CPUs. All CPUs are of the same type and are attributed one of the K intervals used in the PLT algorithm. Moreover, after measuring the communication overhead as in Equation 4.65, we find it to have a relatively small contribution to slowdown compared to the LBS.

First, we examine the slowdown due to communication overhead. We measure t_{solv} starting when sending initial states and terminal adjoints to the intervals and ending when the final sensitivities $dF^*/d\boldsymbol{s}$ are found, and $\hat{t}_{solv}$ the wall time on the slowest worker to solve the state and adjoint equations and combine them into partial sensitivities $dF_k^*/d\boldsymbol{s}$. On the slowest worker, we start the time measurement after initial states



(a) $K = 2$,
$f^* = 0.7461$
$\Delta f^* = 10.94\%$.

(b) $K = 4$,
$f^* = 0.6782$
$\Delta f^* = 2.02\%$.

(c) $K = 8$,
$f^* = 0.6584$
$\Delta f^* = -0.93\%$.

(d) $K = 16$,
$f^* = 0.6672$
$\Delta f^* = 0.40\%$.

(e) $K = 32$,
$f^* = 0.6649$
$\Delta f^* = 0.06\%$.

(f) $K = 48$,
$f^* = 0.7532$
$\Delta f^* = 11.78\%$.

Figure 4.19: The designs, optimized objectives f^*, and relative changes in objective Δf^* computed using the PLT algorithm with varying number of intervals K. Two different designs are found for $K = 2$ and $K = 48$ compared to the designs for $K = 4, 8, 16, 32$ which are similar to the GT design in Figure 4.15a. Since different designs have different nonlinear flow solutions, computational times differ.

and terminal adjoint are received and end before sending terminal states, initial adjoints, and partial sensitivities. We note that t_{solv} and $\hat{t}_{solv}$ are summations of the wall times over all design iterations j. Subsequently, the overhead time is defined as the difference in wall times between the complete computation and the computation on the slowest worker:

$$t_{ov} = t_{solv} - \hat{t}_{solv}. \tag{4.65}$$

Table 4.9 provides the percentage of slowdown t_{ov}/t_{solv} caused by communication overhead and shows that communication overhead is negligible for low number of intervals and increases with the number of intervals. For $K = 48$, 4.52% of the time is spent on communication. The increased communication overhead is caused by the relatively short interval lengths of only four or five time steps for $K = 48$. The PLT algorithm suffers from similar problems as common parallel domain decomposition with small computational domains leading to large communication overhead.

Second, we investigate the slowdown due to load balancing. In Figure 4.20, we plot the relative speedup $t^{GT}_{solv}/t^{PLT}_{solv}$. Additionally, we compute a theoretical best speedup

which does not scale linearly due to the first load balancing issue. Dividing the $N = t_t/\Delta t = 210$ time steps computed using Table 4.7 by the $K = 48$ intervals, we find that the intervals either contain four or five time steps. The theoretical best speedup we can achieve is thus $210/5 = 42$ and not 48. Similar limits on the speedup are computed for $K = 4, 8, 16, 32$. The first load balancing issue is caused by discrete interval lengths.

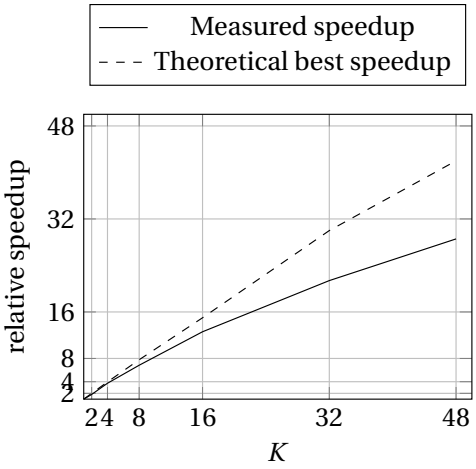


Figure 4.20: The speedup $t^{GT}_{solv}/t^{PLT}_{solv}$ of the PLT algorithm. The theoretical best speedup does not scale linearly with the number of intervals K due to the discrete interval lengths. Moreover, solving nonlinear equations finding the solution on some intervals requires more computational work as shown in Figure 4.21, which causes the slowdown found in Table 4.9. We note that the additional 20 design iterations were taken into account to compute the speedup for $K = 48$ by averaging the time for sensitivity computation and computing the speedup as $(120 \cdot t^{GT}_{solv})/(100 \cdot t^{PLT}_{solv})$.

In Figure 4.20 we find a lower performance than the theoretical best speedup, which cannot be fully explained by communication overhead. The additional decrease in speedup is caused by the nonlinear flow solution being more expensive on some intervals than on others. To measure the cost of solving the state equations on interval Θ_k in design iteration j, we measure the wall time for the state computation $t^{k,j}_{state}$ locally on each CPU. Subsequently, we measure the average time for solving the state equations on an interval at a given design as:

$$\bar{t}^{j}_{state} = \frac{\sum_{k=1}^{K} t^{k,j}_{state}}{K}. \tag{4.66}$$

In the optimal scenario, each interval would have the same computational work and would spend the same time ($\bar{t}^{j}_{state}$) on the state solution. Figure 4.21 shows a map of the relative computational cost of the intervals, compared to the average computational cost of the intervals $(t^{k,j}_{state} - \bar{t}^{j}_{state})/\bar{t}^{j}_{state}$ for $K = 16$. The more expensive intervals take 20% longer than the cheaper ones, leading to a large decrease in speedup. We note that this measured increase may also result from the load balancing issues due to discrete intervals beside the issues due to expensive time steps. Subsequently, we measure load

balancing slowdown (LBS) by comparing the optimal wall time $\bar{t}^j_{state}$ with the wall time spend on solving the state solution on the slowest worker $\hat{t}^j_{state}$, and average over design iterations j as:

$$\text{LBS} = \frac{\sum_j (\hat{t}^j_{state} - \bar{t}^j_{state})}{\sum_j \bar{t}^j_{state}}. \tag{4.67}$$

In Table 4.9, the LBS can be found to be relatively large compared to the communication overhead. Load balancing issues are thus the main obstacle preventing speedup in the PLT algorithm for this example. When dealing with simulations that show less variation between intervals, this problem is expected to vanish.

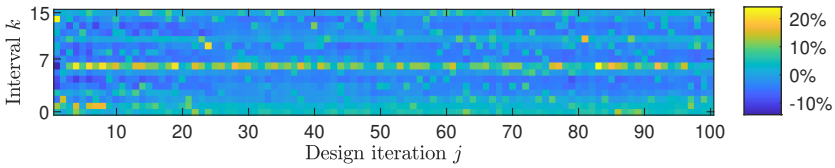


Figure 4.21: The relative cost of solving the state solution on an interval for $K = 16$. For each design iteration j we measure the time to solve the state equations on Θ_k as $t^{k,j}_{state}$ and compute the average $\bar{t}^k_{state}$ following Equation 4.66. Subsequently, we normalize the time required to solve the state equations on an interval $(t^{k,j}_{state} - \bar{t}^j_{state})/\bar{t}^j_{state}$. We find that certain intervals are more expensive to solve than others throughout the optimization procedure. Similar observations were made for different values of K.

4.6. GUIDELINES FOR ALGORITHM SELECTION

Five algorithms for adjoint sensitivity computation are examined in this paper and their advantages and limitations are discussed. Using our results and experience, we formulate guidelines for the selection of an appropriate algorithm based on the characteristics of the transient optimization problem to be solved. A topic not discussed in the main body of this paper was implementation. All algorithms consist of sensitivity computations on intervals. The GT algorithm computes the complete sensitivity on the complete time interval at once. All other algorithms compute partial sensitivities on subintervals separately. To compute the sensitivity on any interval, an initial state, terminal adjoint, design, and (time dependent) boundary conditions are required. A piece of code that can compute partial sensitivities on an interval can thus be used for all algorithms. Even the adjoint correction in the CP/LT algorithm is a simplified version of the sensitivity computation on an interval. Following this approach implementing all the CP, LT, CP/LT and PLT algorithms should not take more time than implementing the GT algorithm. Implementation time is thus not included in the guidelines for algorithm selection.

As discussed in Section 4.1, only when boundary condition are complex and change drastically over time, or when nonlinear physics are considered should the algorithms in this work be used. Subsequently, algorithms should be compared on three properties discussed in this paper: 1) memory requirements, 2) computational cost, and 3) convergence behavior. A decision tree taking into account these properties is given in

Figure 4.22 and is discussed in the remainder of this section.

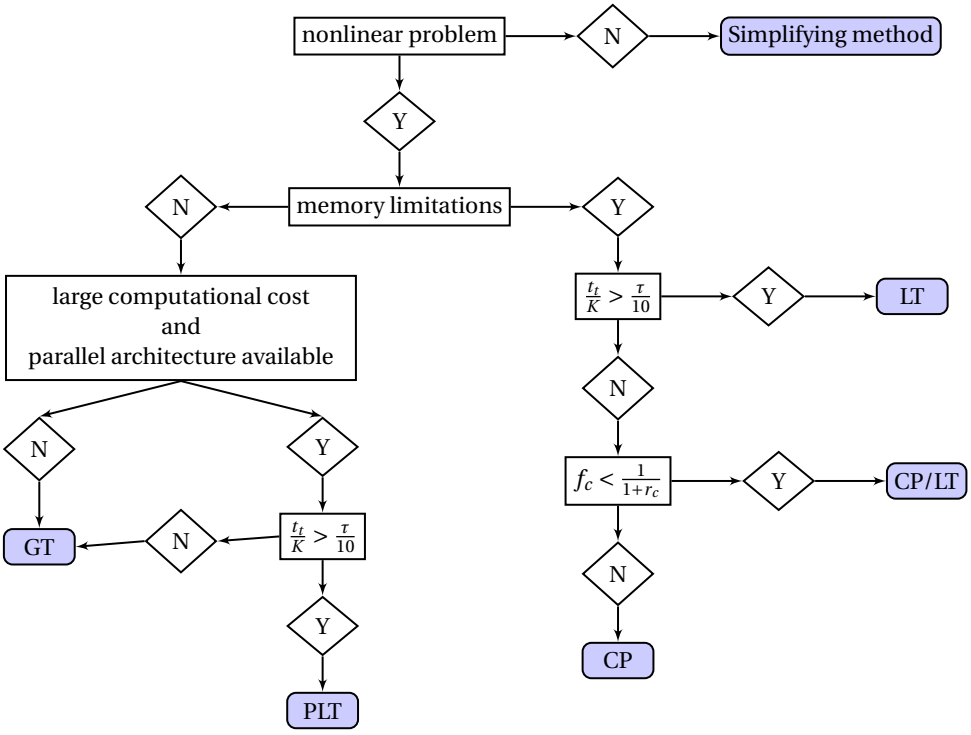


Figure 4.22: A decision tree for selecting the correct transient sensitivity computation algorithm. Decisions are based on memory requirements, computational cost and algorithmic stability. To analyze stability, characteristic time τ needs to be estimated. To compare computational cost of the CP/LT and CP algorithms, an estimation of the number of corrected intervals f_c, and relative computational cost r_c needs to be made.

When memory requirements are not an issue and the complete state equations may be stored, we recommend to choose between the GT and PLT algorithms and move to the left part of the decision tree in Figure 4.22. To select an algorithm, the computational cost and availability of a parallel architecture are considered. When the overall computational cost is low, resulting in short computational times, the GT algorithm is the safest choice as it does not suffer from global stability problems. If computational cost is high and a parallel computation architecture is available, we can consider the PLT algorithm. However, the PLT algorithm should only be used if weak global convergence, as discussed in Section 4.3.1, is satisfied. For this we found a weak global convergence requirement in Equation 4.43, the length of an interval $\Delta\theta = t_t/K$ is required to be longer than one tenth of the characteristic time $t_t/K > \tau/10$. If weak global convergence is not satisfied, we take the risk of ending up in an inferior local optimum with features of relatively short characteristic times, as illustrated in Section 4.5. It is up to the user to approximate characteristic times and to weigh the speedup of the PLT algorithm against the possibility of inferior local optima. Moreover, if the user upgrades their computational system to a parallel system for speedup when setting up the optimization prob-

lem, available memory is also increased. Subsequently, the user should reevaluate the question of memory limitations, and the possibility of using the PLT algorithm, which has the authors' preference for its low computational time.

If memory limitations are constraining, the user should turn to the right part of the decision tree in Figure 4.22 and choose between the CP, LT, and CP/LT algorithms. We assume the smallest number of intervals K to satisfy the memory constraints are used. For the LT and CP/LT algorithms, we use the lowest K possible as this improves convergence behavior, for the CP algorithm the lowest K reduces computational cost. Selecting the appropriate algorithm depends on both computational cost and convergence behavior. As predicted in Section 4.4.2 and shown in Section 4.5, the LT algorithm is generally the fastest followed by the CP/LT or CP algorithm. The exception is the convergence of the thermal problem using LT with $K = 2$ in Section 4.5.1. However, we included exactly these results as they illustrate this exception, but did not generally find this type of behavior for the LT algorithm. If weak global convergence is satisfied ($t_t/K > \tau/10$) we use the LT algorithm.

If memory requirements impose limitations and weak global convergence is not guaranteed, we need to choose between the CP/LT and CP algorithms. Selecting the appropriate algorithm depends on computational cost. In Equation 4.29 we show the relative computational cost of the CP/LT algorithm to the CP algorithm. The relative cost is dependent on the number of intervals K, the relative cost of the adjoint and sensitivity computation c_a to the state computation c_s: $r_c = c_a/c_s$, and an estimate for the percentage of corrected intervals f_c. To compare the computational cost of the CP/LT and CP algorithms, we thus need to compute at what percentage of corrected intervals the CP/LT algorithm becomes more expensive than the CP algorithm. Subsequently, the choice of algorithm is based on a prediction of f_c. We use the CP/LT algorithm when the ratio $C^{CP/LT}/C^{CP}$ from Equation 4.29 is lower than one, resulting in:

$$f_c < \frac{1}{1+r_c}, \tag{4.68}$$

which allows us to decide between the CP and CP/LT algorithm in Figure 4.22. Relative cost r_c can be easily approximated by running the optimization procedure for only one iteration and measuring the time spent on adjoint and state equations. However, for estimating f_c some insight into the problem is required. As shown in Figures 4.12 and 4.18, only during the first design iterations when design changes are large are corrections necessary. An experienced designer may thus estimate the number of design iterations associated with large design changes and the number of iterations required to converge. Subsequently, an estimation of f_c can be made *a priori*, and an informed choice between the CP/LT and CP can be made. Furthermore, we note that for highly nonlinear systems where $r_c \to 0$, we find $f_c < 1$ and may thus always use the CP/LT algorithm.

4.7. DISCUSSION AND CONCLUSION

In this work we thoroughly examined five algorithms for transient sensitivity computation in topology optimization. Two of these algorithms are new, and proposed for the first time in this paper. This has been motivated by the fact that in topology optimization

of transient problems, memory- and time-efficient adjoint sensitivity analysis is crucial. The algorithms have been compared in terms of memory requirements, computational cost, and stability. Firstly, we examined the state-of-the-art GT, CP, and LT algorithms. The GT algorithm serves as the reference and suffers from severe memory limitations which motivates the development of new algorithms. To overcome these memory limitations, the most common approach is the CP algorithm. Although the CP algorithm reduces memory usage, it significantly increases the required computational time due to recomputation of the state solutions, as shown in Section 4.5.2. To avoid recomputation of the state solutions, the LT algorithm may be used. However, due to the approximations of adjoints and consequent stability and convergence issues described in Section 4.3.1 and illustrated in Section 4.5.1, using the LT algorithm may compromise the optimization outcome. To solve these stability issues, we introduced the hybrid CP/LT algorithm which does not show the convergence issues of the LT algorithm at the cost of an increased computational time. However, the hybrid algorithm showed a clear reduction in computational time compared to the CP algorithm.

We point out the importance of understanding the physics and characteristic times of the optimization problem for selecting an appropriate optimization approach. First of all, only when the characteristic timescale is long enough compared to the optimization time horizon or when sufficiently complex physics/load cases are examined do we benefit from the algorithms examined in this work. If the characteristic timescale is short compared to the time horizon of the optimization problem, equivalent static load algorithms can be used, and when the physics/load cases are simple enough, model order reduction techniques can be considered. Secondly, comparing the characteristic time to the interval lengths associated to the LT algorithm, it can be determined whether stability and convergence issues arise and the hybrid CP/LT algorithm is required. When interval lengths are an order of magnitude smaller than the estimated characteristic time of the optimization problem, convergence issues can arise in the (P)LT algorithms.

To address the challenge of computational time in transient optimization, the novel PLT algorithm was proposed. In essence, the PLT algorithm is an extension of the LT algorithm where not only adjoints, but also states are approximated during optimization. Consequently, the PLT algorithm also suffers from similar stability and convergence problems as the LT algorithm. However, by inspecting the characteristic time of the optimization problem, stability and convergence issues may be identified *a priori*. This results in an upper limit on the number of intervals K. Although the PLT algorithm reduces computational time, it does not reduce memory requirements. However, when computational power is scaled up, the total available memory is also often increased.

For future research, we note the possibility to apply spatial domain decomposition techniques for further acceleration when speedup using the PLT algorithm is limited by stability and convergence constraints. A comparison between domain decomposition techniques and the PLT algorithm was not included in this work and is recommended. Similar to domain decomposition techniques, the PLT algorithm suffers from communication overhead when the decomposition becomes too fine. In addition, for problems with a varying cost per time step (e.g., nonlinear problems), the efficiency of the PLT algorithm is affected by load balancing issues. This presents an opportunity for the development of algorithms which address these load balancing issues by adaptively in-

creasing the interval length of cheap intervals and decreasing the length of expensive intervals. Furthermore, another subject for further research is the implementation of a hierarchical approach to reduce the memory requirements of the PLT algorithm. In such an approach, the intervals used by the PLT algorithm can be further divided into subintervals which are used to locally apply the LT, CP, or CP/LT algorithm.

5

Conclusions and Recommendations

The main goal of this thesis is to improve the ease of use of Topology Optimization (TO) for transient flow and thermal problems, such that the benefits of TO can be leveraged in the aerospace industry. Using simple but goal-oriented examples, the characteristics of state-of-the-art and novel methods for TO of flow problems and for transient sensitivity analysis have been examined in detail. Consequently, recommendations and guidelines for these methods have been given, such that engineers can make informed choices when using them for TO of transient flow and thermal problems.

5.1. Conclusions

To improve the ease of use of TO for aerospace problems, three challenges were defined in Section 1.3 and are addressed throughout this thesis. To conclude, we achieved:

- **A better understanding of flow models for density-based TO**

  By inspecting the VANS equations for porous flow, this thesis provides new insights into the adaptation of flow models for TO in Chapter 2. Exploring the literature on VANS models, three novel options to distinguish solid and fluid areas in TO are identified: the pressure penalization, the updated inertia term, and the second Brinkman correction. We conclude that the pressure penalization is able to improve the accuracy of the flow solution, without a large tendency to converge to inferior local optima, often associated with highly accurate designs and large flow penalization. Moreover, current penalization approaches are unable to appropriately penalize the flow for both dominant inertia and viscous forces, and require tuning to balance accuracy of the flow solution and convergence behavior of the optimization process.

- **A reliable approach for density-based TO of flow problems**

  In Chapter 3, a continuation approach for TO of flow problems is constructed, which decreases the tendency to converge to inferior local optima, while maintaining the accuracy of the flow solution at the optimum. The continuation relies on a predictable flow reduction in the solid domain and its relation to the convexity of the pressure drop objective. The pressure drop objective is convex when the flow reduction in gray areas found in intermediate designs is low, and concave when the flow reduction is high. In the earlier stages of optimization, we allow for low flow reduction, resulting in a convex objective response and large design updates, in the later stages, we use a large flow reduction, resulting in accurate flow solutions but a concave objective response and consequent small design updates.

  Flow reduction is only predictable when a Darcy penalization is present to inhibit flow with dominant viscous terms, and a Forchheimer penalization to inhibit flow with dominant inertia terms. We conclusively show that when using only the Darcy penalization, flow cannot be appropriately inhibited in both scenarios. Moreover, in the Darcy with Forchheimer (DF) approach, unstable flow solutions are found when the flow field in an updated design is initialized using the flow field from the previous design to speed up computations. Flow solutions are stable in the novel Darcy with filtered Forchheimer (DFF) approach, which adds a penalization based on a filtered velocity field and reliably reduces the flow in the solid domain to a predefined magnitude. To conclude, using the predictable flow reduction in the DFF approach and its relation to convexity of the pressure drop objective, an informed parameter selection and continuation strategy for TO of flow problems is achieved.

- **A reduction of time and/or memory requirements in TO of transient thermal and flow problems**

  By investigating algorithms for memory reduction in transient sensitivity analysis, time and/or memory requirements are reduced in Chapter 4. State-of-the-art algorithms to reduce computational cost are (i) the Checkpointing (CP) algorithm, which increases the computational time, and (ii) the Local-in-Time (LT) algorithm, which introduces approximations of the adjoint variables. All methods, which reduce time or memory requirements, are based on splitting the temporal domain into intervals. When approximations of the adjoint are used, the interval size may define a limit on the maximum characteristic time in the optimized designs. To remove this limit while maintaining a computationally efficient algorithm, the novel hybrid Checkpointing/Local-in-Time (CP/LT) algorithm is introduced. Additionally, the novel Parallel-Local-in-Time (PLT) algorithm computes the sensitivity contributions on the intervals in parallel. Numerical examples on a piston pump design showed significant speedups up to a factor 28. However, as this approach uses approximations of the initial conditions on the intervals, it may limit the maximum characteristic time.

  To choose an appropriate algorithm, two characteristics of the optimization problems need to be examined. Firstly, the maximum characteristic time may be limited by the interval length. Secondly, to select a computationally efficient algo-

rithm, the relative computational cost of the state and adjoint solve needs to be determined. Clear guidelines to select an appropriate algorithm are provided in this thesis. In conclusion, both through these guidelines and the two novel algorithms, the practical applicability of TO for transient thermal and flow problems is improved.

From a global perspective, using the specific contributions on TO for flow problems from this thesis, a more general approach to reliably adapt physics for density-based TO is provided. The approach consists of two main steps:

1. To explore the possibilities for adapting a physics model to density-based TO, the physics are investigated in a homogenized sense. Similar to the original work by Bendsøe (1989), where homogenized stress-strain relations are adapted for solid/void TO, we adapt the porous flow VANS equations for solid/fluid TO. By inspecting the homogenized equations, insight into the behavior of the physics, objectives, and constraints in the different phases is gained, facilitated by the vast body of literature available on homogenized or volume averaged equations. These insights can be used to define an appropriate approach to distinguish between the different phases, but not to define the exact interpolation functions and parameters.

2. An order analysis is performed to define the exact interpolation functions and parameters which ensure that the physics are appropriately interpolated between the different phases. In this work, the discretized forces in fluid and solid areas are balanced when the flow in the solid domain is much lower than the fluid domain flow.

To conclude, using this general approach, appropriate models with a systematic choice of parameters for density-based TO can be derived, which improve the ease of use for engineers.

5.2. RECOMMENDATIONS

Firstly, it is strongly recommended to apply the interpolation, penalization, and parameter selection proposed in this thesis to flow TO problems in the future. As shown, this alleviates the need for manual parameter tuning and provides a methodical approach to perform flow TO. Furthermore, for the sake of reproducibility and systematic advancement in research, it is recommended that flow TO publications provide full details on interpolation and parameter choices.

Beyond the content of this thesis, several directions for future research can be recommended. Generalizing the approach to adapt physics to TO, as presented in this conclusion, we recommend the investigation of two problems in particular: TO of thermo-fluid problems, and TO of turbulent flow problems.

To design high-performance heat exchangers, tackling TO of thermo-fluid problems starts by using the DFF approach for flow TO presented in this thesis. As parameter tuning is often required, additional research is needed to adapt the thermal convection-diffusion equations to density-based TO. Moreover, by applying the volume average

and by inspecting the literature on thermal convection-diffusion through porous media, a physically consistent interpolation can be derived. A straightforward approach to thermo-fluid TO interpolates the thermal capacity of the solid and fluid phases in the convective term. However, the solid does not convect thermal energy. Interpolating solid and fluid properties in the convection term may result in nonphysical behavior. Secondly, an order analysis for the convection-diffusion equation is advised. The order analysis should inform a parameter interpolation and continuation strategy, such that solid conduction is dominant in the solid domain and fluid convection or conduction in the fluid domain. As flow velocities are not zero in the solid domain, some convection will always be present. However, using the prediction of flow reduction presented in this thesis, it is possible to make an estimation of the convective heat flux and use it to appropriately reduce the convection in the solid domain.

To investigate TO of high Reynolds flow problems, we propose to extend the methods in this work to turbulent flow TO. Turbulence can be modeled using a Large Eddy Simulation or Direct Numerical Simulation, transient by nature, which thus require the techniques for TO of transient problems presented in this thesis. However, sensitivities of such chaotic phenomena may be unstable which needs additional investigation. Additionally, these simulations often have excessive computational cost inhibiting their use for TO altogether. A more common approach is to use the Reynolds-Averaged Navier-Stokes (RANS) equations to model turbulent flow in TO. As the filtered Forchheimer penalization is already required for TO of moderate Reynolds problems, we expect it to be crucial for TO of high Reynolds flow problems. Additionally, the RANS equations include closure relations to model turbulent field properties. To apply the solid/fluid boundary conditions on these turbulent fields continuously, a penalty approach similar to the one in the momentum equation is often used, see Dilgen et al. (2018). To select an appropriate penalty magnitude, an order analysis on the closure equations should be performed. This approach is recommended to achieve a reliable method for TO of turbulent flow problems.

Furthermore, TO of nonlinear path-dependent problems may suffer from the same memory limitations as TO of transient problems. The algorithms to balance time and memory requirements for TO of transient problems can likely be adapted to find a compromise between these requirements in TO of nonlinear path-dependent problems. However, further analysis of the characteristics of these nonlinear problems is required to appropriately adapt the schemes for transient sensitivity analysis. In transient sensitivity analysis, the interval length imposes a limitation on the maximum characteristic time found in the optimized design. For optimization of nonlinear path-dependent problems, it is interesting to examine which design properties are influenced by splitting the nonlinear iteration procedure into multiple parts. Consequently, an adaptation of the PLT algorithm may be used to speed up TO of nonlinear path-dependent problems.

Finally, the inhibition of flow in the solid domain through the Darcy penalization is generalized as a penalty approach by Bruns (2007). In this type of approach, constraints in the design domain are enforced through penalties. The penalization approach presented in this thesis shows similarities to, for example, methods for changing the sup-

port location in TO of structural compliance problems (Buhl, 2002). Instead of velocities in the solid areas inhibited by the Darcy penalization, displacements in support areas can be inhibited using a penalty approach. A similar method to the one presented in this thesis can be used to select the appropriate penalty magnitude and continuation approach for such problems.

BIBLIOGRAPHY

Aage, N., Poulsen, T., Gersborg-Hansen, A., & Sigmund, O. (2008). Topology optimization of large scale Stokes flow problems. *Struct Multidisc Optim*, *35*, 175–180.

Abdelhamid, M., & Czekanski, A. (2023). On the Calculation of the Brinkman Penalization Term in Density-Based Topology Optimization of Fluid-Dependent Problems. *arXiv preprint arXiv:2302.14156*.

Alexandersen, J. (2022). Topography optimisation of fluid flow between parallel plates of spatially-varying spacing: Revisiting the origin of fluid flow topology optimisation. *Structural and Multidisciplinary Optimization*, *65*(5), 152.

Alexandersen, J. (2023). A detailed introduction to density-based topology optimisation of fluid flow problems with implementation in MATLAB. *Structural and Multidisciplinary Optimization*, *66*(1), 12.

Alexandersen, J., Aage, N., Andreasen, C. S., & Sigmund, O. (2014). Topology optimisation for natural convection problems. *International Journal for Numerical Methods in Fluids*, *76*(10), 699–721.

Alexandersen, J., & Andreasen, C. S. (2020). A review of topology optimisation for fluid-based problems. *Fluids*, *5*(1), 29.

Alexandersen, J., Andreasen, C. S., Aage, N., Lazarov, B. S., & Sigmund, O. (2013). Topology optimisation for coupled convection problems. *10th world Congress on structural and multidisciplinary optimization, Orlando*, 19–24.

Alfonsi, G. (2009). Reynolds-Averaged Navier-Stokes Equations for Turbulence Modeling. *Appl. mech. Rev.*, *62(4)*, 040802 (20 pages).

Allaire, G., Jouve, F., & Toader, A.-M. (2004). Structural optimization using sensitivity analysis and a level-set method. *Journal of Computational Physics*, *194*(1), 363–393.

Alonso, D. H., & Silva, E. C. N. (2022). Topology optimization applied to the design of Tesla-type turbine devices. *Applied Mathematical Modelling*, *103*, 764–791.

Andreasen, C. S., Gersborg, A. R., & Sigmund, O. (2009). Topology optimization of microfluidic mixers. *International Journal for Numerical Methods in Fluids*, *61*(5), 498–513.

Angot, P., Goyeau, B., & Ochoa-Tapia, J. A. (2017). Asymptotic modeling of transport phenomena at the interface between a fluid and a porous layer: Jump conditions. *Physical Review E*, *95*(6), 063302.

Åström, K. J., & Murray, R. M. (2020). *Feedback systems: An introduction for scientists and engineers* (2nd ed.). Princeton university press.

Beavers, G. S., & Joseph, D. D. (1967). Boundary conditions at a naturally permeable wall. *Journal of fluid mechanics*, *30*(1), 197–207.

Behrou, R., Ranjan, R., & Guest, J. K. (2019). Adaptive topology optimization for incompressible laminar flow problems with mass flow constraints. *Computer Methods in Applied Mechanics and Engineering*, *346*, 612–641.

Bendsøe, M. P. (1989). Optimal shape design as a material distribution problem. *Structural optimization, 1*, 193–202.

Bendsøe, M. P., & Sigmund, O. (1999). Material interpolation schemes in topology optimization. *Archive of applied mechanics, 69*, 635–654.

Bendsøe, M. P., & Sigmund, O. (2004). *Topology optimization: Theory, methods, and applications*. Springer-Verlag.

Borrvall, T., & Petersson, J. (2002). Large-scale topology optimizaiton in 3D using parallel computing. *Comput. Methods Appl. Mech. Engrg., 190*, 6201–6229.

Borrvall, T., & Petersson, J. (2003). Topology optimization of fluids in Stokes flow. *International journal for numerical methods in fluids, 41*(1), 77–107.

Breugem, W.-P., & Boersma, B.-J. (2005). Direct numerical simulations of turbulent flow over a permeable wall using a direct and a continuum approach. *Physics of fluids, 17*(2), 025103.

Brinkman, H. C. (1949). A calculation of the viscous force exerted by a flowing fluid on a dense swarm of particles. *Flow, Turbulence and Combustion, 1*(1), 27–34.

Bruns, T. E., & Tortorelli, D. A. (2001). Topology optimization of non-linear elastic structures and compliant mechanisms. *Computer methods in applied mechanics and engineering, 190*(26-27), 3443–3459.

Bruns, T. (2007). Topology optimization by penalty (top) method. *Computer Methods in Applied Mechanics and Engineering, 196*(45-48), 4430–4443.

Buhl, T. (2002). Simultaneous topology optimization of structure and supports. *Structural and multidisciplinary optimization, 23*, 336–346.

Carraro, T., & Geiger, M. (2015). Direct and indirect multiple shooting for parabolic optimal control problems. In *Multiple shooting and time domain decomposition methods* (pp. 35–67). Springer.

CFD Module User's Guide v. 6.1. (2022). COMSOL Multiphysics® COMSOL AB. Stockholm, Sweden.

Challis, V. J., & Guest, J. K. (2009). Level set topology optimization of fluids in Stokes flow. *International journal for numerical methods in engineering, 79*(10), 1284–1308.

Chen, C., Yaji, K., Yamada, T., Izui, K., & Nishiwaki, S. (2017). Local-in-time adjoint-based topology optimization of unsteady fluid flows using the lattice Boltzmann method. *Mechanical Engineering Journal, 4*(3), 17–00120.

Choi, W., & Park, G. (2002). Structural optimization using equivalent static loads at all time intervals. *Comput. Methods Appl. Mesh. Engrg, 191*, 2077–2094.

COMSOL Multiphysics® v.6.1. (n.d.). www.comsol.com

Dbouk, T. (2017). A review about the engineering design of optimal heat transfer systems using topology optimization. *Applied Thermal Engineering, 112*, 841–854.

Dilgen, C. B., Dilgen, S. B., Fuhrman, D. R., Sigmund, O., & Lazarov, B. S. (2018). Topology optimization of turbulent flows. *Computer Methods in Applied Mechanics and Engineering, 331*, 363–393.

Evgrafov, A. (2005). The limits of porous materials in the topology optimization of stokes flows. *Applied Mathematics and Optimization, 52*(3), 263–277.

Fang, L., Vandewalle, S., & Meyers, J. (2022). A parallel-in-time multiple shooting algorithm for large-scale PDE-constrained optimal control problems. *Journal of Computational Physics, 452*, 110926.

Gander, M. J. (2015). 50 years of time parallel time integration. In *Multiple shooting and time domain decomposition methods* (pp. 69–113). Springer.

Gersborg-Hansen, A., Bendsøe, M. P., & Sigmund, O. (2006). Topology optimization of heat conduction problems using the finite volume method. *Structural and multidisciplinary optimization*, *31*(4), 251–259.

Gersborg-Hansen, A., Sigmund, O., & Haber, R. B. (2005). Topology optimization of channel flow problems. *Structural and multidisciplinary optimization*, *30*, 181–192.

Goyeau, B., Lhuillier, D., Gobin, D., & Velarde, M. (2003). Momentum transport at a fluid–porous interface. *International Journal of Heat and Mass Transfer*, *46*(21), 4071–4081.

Griewank, A. (1992). Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation. *Optimization Methods and software*, *1*(1), 35–54.

Griewank, A., & Walther, A. (2000). Algorithm 799: Revolve: An implementation of checkpointing for the reverse or adjoint mode of computational differentiation. *ACM Transactions on Mathematical Software (TOMS)*, *26*(1), 19–45.

Grimm, J., Pottier, L., & Rostaing-Schmidt, N. (1996). Optimal time and minimum space-time for reversing a certain class of prorgams. *Computational Differentiation: Techniques, Applications, and Tools*, 161–172.

Guest, J. K., & Prévost, J. H. (2006). Topology optimization of creeping fluid flows using a Darcy–Stokes finite element. *International Journal for Numerical Methods in Engineering*, *66*(3), 461–484.

Guest, J. K., & Prévost, J. H. (2007). Design of maximum permeability material structures. *Computer Methods in Applied Mechanics and Engineering*, *196*(4-6), 1006–1017.

Guillaume, P., & Idris, K. S. (2004). Topological sensitivity and shape optimization for the Stokes equations. *SIAM Journal on Control and Optimization*, *43*(1), 1–31.

Haftka, R. T. (1981). Techniques for thermal sensitivity analysis. *International Journal for Numerical Methods in Engineering*, *17*(1), 71–80.

Hassani, B., & Hinton, E. (1998). A review of homogenization and topology optimization I-homogenization theory for media with periodic structure. *Computers and Structures*, *69*, 707–717.

Hauke, G. (2001). Simple stabilizing matrices for the computation of compressible flows in primitive variables. *Computer methods in applied mechanics and engineering*, *190*(51-52), 6881–6893.

Hauke, G., & Hughes, T. (1994). A unified approach to compressible and incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, *113*(3-4), 389–395.

Hernandez-Rodriguez, R., Angot, P., Goyeau, B., & Ochoa-Tapia, J. A. (2022). Momentum transport in the free fluid-porous medium transition layer: One-domain approach. *Chemical Engineering Science*, *248*, 117111.

Heuveline, V., & Walther, A. (2006). Online checkpointing for parallel adjoint computation in PDEs: Application to goal-oriented adaptivity and flow control. *European Conference on Parallel Processing*, 689–699.

Hooijkamp, E. C., & Keulen, F. v. (2018). Topology optimization for linear thermo-mechanical transient problems: Modal reduction and adjoint sensitivities. *International Journal for Numerical Methods in Engineering*, *113*(8), 1230–1257.

Howes, F., & Whitaker, S. (1985). The spatial averaging theorem revisited. *Chemical Engineering Science*, *40*, 1387–1392.

Hughes, T. J., & Mallet, M. (1986). A new finite element formulation for computational fluid dynamics: III. the generalized streamline operator for multidimensional advective-diffusive systems. *Computer methods in applied mechanics and engineering*, *58*(3), 305–328.

Jensen, K. E. (2018). Topology optimization of Stokes flow on dynamic meshes using simple optimizers. *Computers & Fluids*, *174*, 66–77.

Kang, B., Choi, W., & Park, G. (2001). Structural optimization under equivalent static loads transformed from dynamic loads based on displacement. *Computers and Structures*, *79*, 145–154.

Kondoh, T., Matsumori, T., & Kawamoto, A. (2012). Drag minimization and lift maximization in laminar flows via topology optimization employing simple objective function expressions based on body force integration. *Structural and Multidisciplinary Optimization*, *45*(5), 693–701.

Kreissl, S., & Maute, K. (2012). Levelset based fluid topology optimization using the extended finite element method. *Structural and Multidisciplinary Optimization*, *46*, 311–326.

Kreissl, S., Pingen, G., & Maute, K. (2011). Topology optimization for unsteady flow. *International Journal for Numerical Methods in Engineering*, *87*(13), 1229–1253.

Kristiansen, H., & Aage, N. (2022). An open-source framework for large-scale transient topology optimization using petsc. *Structural and Multidisciplinary Optimization*, *65*(10), 295.

Lazarov, B. S., & Sigmund, O. (2011). Filters in topology optimization based on helmholtz-type differential equations. *International Journal for Numerical Methods in Engineering*, *86*(6), 765–781.

Lea, D. J., Allen, M. R., & Haine, T. W. (2000). Sensitivity analysis of the climate of a chaotic system. *Tellus A: Dynamic Meteorology and Oceanography*, *52*(5), 523–532.

Li, H., Kondoh, T., Jolivet, P., Furuta, K., Yamada, T., Zhu, B., Zhang, H., Izui, K., & Nishiwaki, S. (2022). Optimum design and thermal modeling for 2d and 3d natural convection problems incorporating level set-based topology optimization with body-fitted mesh. *International Journal for Numerical Methods in Engineering*, *123*(9), 1954–1990.

Li, Q., Sigmund, O., Jensen, J. S., & Aage, N. (2021). Reduced-order methods for dynamic problems in topology optimization: A comparative study. *Computer Methods in Applied Mechanics and Engineering*, *387*, 114149.

Li, X., Wu, K., Zhao, L., & Fan, X. (2024). Topology optimization of regenerative cooling structures under high reynolds number flow with variable thermo-physical properties. *Applied Thermal Engineering*, 124602.

Li, Y., Saitou, K., & Kikuchi, N. (2004). Topology optimization of thermally actuated compliant mechanisms considering time-transient effect. *Finite elements in analysis and design*, *40*(11), 1317–1331.

Lundgaard, C., Alexandersen, J., Zhou, M., Andreasen, C. S., & Sigmund, O. (2018). Revisiting density-based topology optimization for fluid-structure-interaction problems. *Structural and Multidisciplinary Optimization*, *58*, 969–995.

Mahdavi, A., Balaji, R., Frecker, M., & Mockensturm, E. (2006). Topology optimiziton of 2D continua for minimum compliance using parallel computing. *Struct Multidisc Optim*, *32*, 121–132.

Martins, J. R., Sturdza, P., & Alonso, J. J. (2003). The complex-step derivative approximation. *ACM Transactions on Mathematical Software (TOMS)*, *29*(3), 245–262.

MATLAB. (2019). *Version 9.6.0 (r2019a)*. The MathWorks Inc.

Michaël, R., Delphine, R., Pierre-Henri, C., & Alain, B. (2020). Modelling of flow through spatially varying porous media with application to topology optimization. *arXiv preprint arXiv:2004.10712*.

Nield, D. (1991). The limitations of the Brinkman-Forchheimer equation in modeling flow in a saturated porous medium and at an interface. *International Journal of Heat and Fluid Flow*, *12*(3), 269–272.

Nørgaard, S., Sigmund, O., & Lazarov, B. (2016). Topology optimization of unsteady flow problems using the lattice Boltzmann method. *Journal of Computational Physics*, *307*, 291–307.

Ochoa-Tapia, J. A., & Whitaker, S. (1995). Momentum transfer at the boundary between a porous medium and a homogeneous fluid—I. Theoretical development. *International Journal of Heat and Mass Transfer*, *38*(14), 2635–2646.

Olesen, L. H., Okkels, F., & Bruus, H. (2006). A high-level programming-language implementation of topology optimization applied to steady-state Navier–Stokes flow. *International Journal for Numerical Methods in Engineering*, *65*(7), 975–1001.

Philippi, B., & Jin, Y. (2015). Topology optimization of turbulent fluid flow with a sensitive porosity adjoint method (spam). *arXiv preprint arXiv:1512.08445*.

Picelli, R., Moscatelli, E., Yamabe, P. V. M., Alonso, D. H., Ranjbarzadeh, S., dos Santos Gioria, R., Meneghini, J. R., & Silva, E. C. N. (2022). Topology optimization of turbulent fluid flow via the TOBS method and a geometry trimming procedure. *Structural and Multidisciplinary Optimization*, *65*(1), 1–25.

Picioreanu, C., Van Loosdrecht, M. C., & Heijnen, J. J. (2000). Effect of diffusive and convective substrate transport on biofilm structure formation: A two-dimensional modeling study. *Biotechnology and bioengineering*, *69*(5), 504–515.

Qian, X. (2022). On-the-fly dual reduction for time-dependent topology optimization. *Journal of Computational Physics*, *452*, 110917.

Svanberg, K. (1987). The method of moving asymptotes—a new method for structural optimization. *International journal for numerical methods in engineering*, *24*(2), 359–373.

Svanberg, K. (2004). Some modelling aspects for the MATLAB implementation of MMA. *KTH Royal Institute of Technology, Stockholm*.

Takezawa, A., Zhang, X., Tanaka, T., & Kitamura, M. (2020). Topology optimisation of a porous unit cell in a fluid flow considering Forchheimer drag. *International Journal of Computational Fluid Dynamics*, *34*(1), 50–60.

Theulings, M. J. B., Langelaar, M., van Keulen, F., & Maas, R. (2023). Towards improved porous models for solid/fluid topology optimization. *Structural and Multidisciplinary Optimization*, *66*(6), 133.

Theulings, M., Maas, R., Noël, L., van Keulen, F., & Langelaar, M. (2024). Reducing time and memory requirements in topology optimization of transient problems. *International Journal for Numerical Methods in Engineering*, *125*(14), e7461.

Theulings, M., Noël, L., Langelaar, M., & Maas, R. (2025). Reducing parameter tuning in topology optimization of flow problems using a darcy and forchheimer penalization. *Computer Methods in Applied Mechanics and Engineering*, *443*, 118027.

Tian, Y., Gao, R., Wang, Y., Jing, R., Li, A., Dong, X., & Hao, X. (2024). A turbulent flow topology optimization method for diverging tee resistance reduction. *Journal of Building Engineering*, *97*, 110609.

Vafai, K., & Kim, S. (1995). On the limitations of the Brinkman-Forchheimer-extended Darcy equation. *International Journal of Heat and Fluid Flow*, *16*(1), 11–15.

Valdés-Parada, F. J., Ochoa-Tapia, J. A., & Alvarez-Ramirez, J. (2007). Diffusive mass transport in the fluid–porous medium inter-region: Closure problem solution for the one-domain approach. *Chemical Engineering Science*, *62*(21), 6054–6068.

Versteeg, H. K., & Malalasekera, W. (2007). *An introduction to computational fluid dynamics: The finite volume method*. Pearson education.

Wang, F., Lazarov, B. S., & Sigmund, O. (2011). On projection methods, convergence and robust formulations in topology optimization. *Structural and multidisciplinary optimization*, *43*, 767–784.

Wang, Q., Hu, R., & Blonigan, P. (2014). Least squares shadowing sensitivity analysis of chaotic limit cycle oscillations. *Journal of Computational Physics*, *267*, 210–224.

Wang, Q., Moin, P., & Iaccarino, G. (2009). Minimal repetition dynamic checkpointing algorithm for unsteady adjoint calculation. *SIAM Journal on Scientific Computing*, *31*(4), 2549–2567.

Whitaker, S. (1969). Advances in theory of fluid motion in porous media. *Industrial & engineering chemistry*, *61*(12), 14–28.

Whitaker, S. (1986). Flow in porous media I: A theoretical derivation of Darcy's law. *Transport in porous media*, *1*(1), 3–25.

Whitaker, S. (1996). The Forchheimer equation: A theoretical development. *Transport in Porous media*, *25*(1), 27–61.

Wu, C., & Zhang, Y. (2024). Flow topology optimization at high reynolds numbers based on modified turbulence models. *Aerospace*, *11*(7), 525.

Wu, S., Zhang, Y., & Liu, S. (2019). Topology optimization for minimizing the maximum temperature of transient heat conduction structure. *Structural and Multidisciplinary Optimization*, *60*(1), 69–82.

Yaji, K., Ogino, M., Chen, C., & Fujita, K. (2018). Large-scale topology optimization incorporating local-in-time adjoint-based method for unsteady thermal-fluid problem. *Structural and Multidisciplinary Optimization*, *58*(2), 817–822.

Yamaleev, N. K., Diskin, B., & Nielsen, E. J. (2010). Local-in-time adjoint-based method for design optimization of unsteady flows. *Journal of computational physics*, *229*(14), 5394–5407.

Zhao, J., & Wang, C. (2016). Dynamic response topology optimization in the time domain using model reduction method. *Structural and Multidisciplinary Optimization, 53*(1), 101–114.

CURRICULUM VITÆ

Maarten Jan Bernard THEULINGS

14-05-1995 Born in Gouda, The Netherlands.

EDUCATION

2019–2024 PhD in Mechanical Engineering
Delft University of Technology, Delft
Thesis: Topology optimization of Transient Fluidic and Thermal Devices
Promotor: Prof. dr. ir. M. Langelaar
Copromotor: Dr. ir. L. Noël

2017–2019 Master in Mechanical Engineering
Delft University of Technology, Delft
Thesis: Foldable FEM: Using enriched and mixed/hybrid methods for the mesh-independent modeling of folds

2013–2017 Bachelor in Mechanical Engineering
Delft University of Technology, Delft
Minor: Philosophy
Leiden University, Leiden

2007–2013 VWO / Gymnasium
Antoniuscollege, Gouda

LIST OF PUBLICATIONS

3. **Theulings, M. J. B.**, Noël. L., Langelaar, M. & Maas, R. (2025). Reducing parameter tuning in topology optimization of flow problems using a Darcy and Forchheimer penalization. *Computer Methods in Applied Mechanics and Engineering*, 443, 118027, https://doi.org/10.1016/j.cma.2025.118027

2. **Theulings, M. J. B.**, Maas, R., Noël. L., van Keulen, F. & Langelaar, M. (2024). Reducing time and memory requirements in topology optimization of transient problems. *International Journal for Numerical Methods in Engineering*, 125(14), e7461, https://doi.org/10.1002/nme.7461

1. **Theulings, M. J. B.**, Langelaar, M., van Keulen, F. & Maas, R. (2023). Towards improved porous models for solid/fluid topology optimization. *Structural and Multidisciplinary Optimization*, 66(6), 133, https://doi.org/10.1007/s00158-023-03570-4

ACKNOWLEDGEMENTS

As the saying goes, it takes a village to write a PhD thesis. Without the personal support of many friends and family, I would not have been able to take the professional guidance of my colleagues and supervisors.

First, I would like to thank my supervisors. My promotor *Matthijs Langelaar* and co-promotor *Lise Noël*, who helped guide my sometimes chaotic research in the right direction, without inhibiting my creativity. I would also like to thank my former supervisor *Fred van Keulen*. After supervising the master thesis of my sister, Fred decided that he wanted to supervise another Theulings sibling and offered me a PhD project in collaboration with the *Netherlands Aerospace Centre* (NLR). At the NLR, I was hired by *Tonny ten Dam*, on who I could count to put personal development before scientific development. The final supervisor was *Robert Maas* from NLR, whose curious mind and rigorous mathematical checks helped me to explain my ideas and accurately write them down.

Two families deserve my recognition. My parents *Henk* and *Anita* who raised me to be a curious person, and helped me choose the right path as a mechanical engineer, my sister *Annemarie* for helping me with her experiences of doing a PhD herself and my sister *Erica* for exploring the research group first and listening to my struggles. During the difficult Corona lockdown, I could stay at my partners parents. In the first week, after telling *Frans* in the evening that my back hurt from working while sitting on the bed, he built me a desk the next morning while I was still asleep. Here, I could work efficiently during lockdown. Staying in the house with *Frans, Marjon, Emma, Otto* and *Nikola*, gave my life some excitement and made me stay motivated.

The discussion of ideas, whether scientific, political or non-sensical is important to develop as a researcher and person. Over the past decades, I had many discussions, with many people who I will not name so I can't forget anyone. Thanks go out to my colleagues who distracted me from too many days of writing by showing me their new and interesting problems. My friend from highschool and before I would like to thank for the special bond we build over the past decade or longer, and the different perspectives on life they show me. It is important to sometimes wind down, have a drink at a bar, and have passionate discussions about things that don't matter. For this, I thank the friends I made at my student association the Bolk, where I feel welcome, even as I am growing older.

My final thanks go out to my partner *Emma*. When my head gets stuck thinking about my research, spending time with you gets it free to think about other things. Without you, I would have spend more time on my research, but made much slower progress.