



A systematic approach for using density-based topology optimization to find design solutions for a heat exchanger distribution system with turbulent incompressible flow

Master's Thesis

Written by:

Jan Nowaczewski

391921

Supervised by:

Prof. Dr.-Ing. Dieter Peitsch

Dr.-Ing. Mario Eck

Dr.-Ing. Quentin Holka

Background:

Topology optimization is a powerful tool that allows engineers to accelerate the optimization process of any given part, as opposed to parametric optimization. An optimization algorithm automates the iterative cycle of simulating, evaluating, and modifying a given part until reaching certain objective values. This expedites the engineering process and makes it possible to generate many different solutions for a given problem in a small amount of time. Evaluating these solutions can indicate favorable design choices, which can be implemented into a final design by the engineer. The strength of topology optimization lies in its ability to create new forms without restriction that fulfill certain imposed constraints, such as fabrication or operating limitations. On the other hand, many optimization settings exist that can vastly alter the result. It is not a replacement for the engineer. Instead of optimizing a part directly, the engineer now searches for the starting values, constraints and objective functions that result in the best solution.

The seminal work on the method focuses on its application for structural mechanics in 1988, where its use produces the recognizable organic truss features. Since then, it rose in popularity with the advent of additive manufacturing, which facilitates the manufacturing of the complex generated structures. In recent time, it has been expanded for use in other applications, notably for fluid mechanics, the application in focus for this paper. Commercial and open-source programs are available that are capable of topology optimization, such as COMSOL, OPENFoam, and ColdStream. One of the goals of this thesis is to quantify the results from these programs generated from identical initial conditions and compare their strengths and weaknesses. Likewise, the effects of modifying initial values of the optimizer will be documented.

The work done for this thesis is in collaboration with SAFRAN Tech and therefore the exact part geometry and conditions of the optimization problem are confidential. Instead, a theoretical optimization problem will be presented in this thesis, which represents the actual one: a heat exchanger collector with turbulent incompressible flow. The examined system is intended for integration into a gas turbine engine with a recuperated cycle. This type of engine aims to increase thermal efficiency by extracting the heat from the exhaust and injecting it into the flow before the combustion chamber. All the necessary additional ducting and heat exchangers must be designed with the smallest amount of losses possible to avoid a net loss in efficiency by adding the recuperated cycle to a given engine.

Tasks:

The primary task for this thesis is the quantification of optimization results from several CFD programs and their evaluation and comparison. The model used will be that of a heat exchanger collection system for a recuperated cycle gas turbine engine. The flow speed in such applications means that a turbulent flow regime is unavoidable. Topology optimization for turbulent flow is known to be problematic; thus, methods need to be developed to overcome this. COMSOL OpenFOAM and ColdStream are the programs that will be evaluated for their effectiveness. They will be tested concerning their performance on an HPC cluster and the quality of their results. In addition, an approach will be developed to obtain useful results using topology optimization. This will comprise the choice of starting values and optimization parameters.

Supervisors:

Prof. Dr.-Ing. Dieter Peitsch (TU Berlin)

Dr.-Ing. Mario Eck (TU Berlin)

Dr.-Ing. Quentin Holka (Safran)

Eidestattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit eigenständig ohne Hilfe Dritter und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe. Alle Stellen, die den benutzten Quellen und Hilfsmitteln unverändert oder sinngemäß entnommen sind, habe ich als solche kenntlich gemacht.

Sofern generative KI-Tools verwendet wurden, habe ich Produktnamen, Hersteller, die jeweils verwendete Softwareversion und die jeweiligen Einsatzzwecke (z.B. sprachliche Überprüfung und Verbesserung der Texte, systematische Recherche) benannt. Ich verantworte die Auswahl, die Übernahme und sämtliche Ergebnisse des von mir verwendeten KI-generierten Outputs vollumfänglich selbst.

Die Satzung zur Sicherung guter wissenschaftlicher Praxis an der TU Berlin vom 15. Februar 2023. https://www.static.tu.berlin/fileadmin/www/10002457/K3-AMBI/Amtsblatt_2023/Amtliches_Mitteilungsblatt_Nr._16_vom_30.05.2023.pdf habe ich zur Kenntnis genommen.

Ich erkläre weiterhin, dass ich die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt habe.

Berlin, den 06.06.2025



.....
Unterschrift

Zusammenfassung

Die folgende Masterarbeit bietet eine umfassende Untersuchung der Topologieoptimierung als ein Ingenieurswerkzeug um komplexe, strömungsführende Bauteile auszulegen. Diese Untersuchung wurde in Zusammenarbeit mit dem Unternehmen Safran Tech durchgeführt, wo die Methode für die Auslegung eines Distributionssystems für einen Wärmetauscher eingesetzt werden sollte. Da dieser Wärmetauscher in einem Triebwerk mit Wärmerückgewinnung eingesetzt werden soll, muss gewährleistet werden, dass das Distributionssystem wenig Druckverluste verursacht und wenig Bauraum einnimmt. In dieser Arbeit wird eine generische Geometrie eines Distributors behandelt, da die Abmaße der wahren Bauteile vertraulich sind. Es wird eher ein Fokus auf die Methodik hinter der Topologieoptimierung für CFD mit turbulenten Strömungen behandelt.

Die Einleitung der Arbeit beschreibt die Grundprinzipien der Topologieoptimierung, beginnend mit ihren Anfängen in den späten 1980er Jahren. Bedeutende Arbeiten, wie die von Bendsoe und Kikuchi, legten den Grundstein für Entwicklungen in diesem Bereich. Die Fähigkeit der Topologieoptimierung, durch Materialverteilung organisch anmutende Strukturen zu erzeugen, hat in verschiedenen Branchen Aufmerksamkeit erregt. Die derzeitige Anwendung von Topologieoptimierungstechniken zeigt ihre Eignung für die Erzeugung von Geometrien, mit denen sich Strömungen effektiv steuern lassen und gleichzeitig Widerstand und Druckverluste minimiert werden.

Im methodischen Teil befasst sich die Arbeit mit dem dichte-basierten Topologieoptimierungsprozess (Density-Based Topology Optimization). Dabei kommen iterative Optimierungstechniken zum Einsatz, um mithilfe von Algorithmen, die auf die Anpassung der Materialdichten innerhalb des definierten Entwurfsraums ausgerichtet sind, eine gewünschte Leistungskennzahl zu erreichen. Sie betont die entscheidende Bedeutung der Initialisierungsprozesse und der Wahl der Parameter in den Optimierungsalgorithmen und hebt hervor, wie diese Faktoren die endgültigen Konstruktionsergebnisse maßgeblich beeinflussen. Besondere Herausforderungen, wie z. B. numerische Instabilitäten, werden zusammen mit möglichen Lösungen wie Regularisierungs- und adaptiven Filtermethoden zur Stabilisierung des Optimierungsprozesses und zur Reduzierung unerwünschter Artefakte in der Konstruktion behandelt.

Ein wichtiger Aspekt der Arbeit ist die Analyse verschiedener Softwarelösungen zur Durchführung der Topologieoptimierung, insbesondere COMSOL, OpenFOAM und ColdStream. Jedes Tool wird anhand von Kriterien wie Benutzerfreundlichkeit, Funktionalität und Recheneffizienz bewertet. Ein Befund dieser Arbeit ist, dass OpenFOAM für diese Anwendung am sinnvollsten ist, da COMSOL für die Optimierung von turbulenten Strömungen ungeeignet ist. ColdStream bietet eine modernere Lösung, aber die Rechenzeit und Resultate sind nicht akzeptabel.

Zusammenfassend kommt die Arbeit zu dem Schluss, dass die Topologieoptimierung ein leistungsfähiges Werkzeug ist, mit dem sich Konstruktionsideen schnell iterieren lassen, um eine hohe Effizienz zu erreichen. Die Beherrschung der zugrunde liegenden Prinzipien, Codierungsparameter und Strömungsdynamik ist für Ingenieure, die diese fortschrittlichen Optimierungstechniken effektiv nutzen möchten, von entscheidender Bedeutung. Das Ergebnis dieser Forschung trägt nicht nur zum akademischen Wissen bei, sondern soll auch Ingenieuren und Forschern, die die Topologieoptimierung in ihren Projekten anwenden möchten, insbesondere im Bereich der Strömungsdynamik, praktische Erkenntnisse liefern.

TABLE OF CONTENTS

Table of Contents	3
Liste des figures	6
Nomenclature	8
1. Introduction	10
1.1 Case Presentation	10
2. Methodology – Topology Optimization	16
2.1 Example Case	16
2.2 Optimization process	17
2.3 Variable Porosity Model – Design Variable ρ	18
2.4 Design variable relaxation	19
2.5 Regularization of Design Variable Field	22
2.6 Sensitivity calculation using the adjoint method	26
2.7 Optimization algorithm	28
2.8 Verification	30
2.9 Summary	31
3. Software Solutions	32
3.1 COMSOL	32
3.2 OpenFOAM	33
3.3 ColdStream	34
3.4 Summary	35
4. Case presentation	37
5. Results	39
5.1 COMSOL	39
5.2 OpenFOAM	41
5.3 ColdStream optimization	46
6. Conclusion	50
Bibliography	51

FIGURES

Figure 1.1: Heat recuperation effect shown in the Brayton cycle	11
Figure 1.2: Schematic of regenerative cycle engine	11
Figure 1.3: General optimization approach	12
Figure 1.4: Different types of automated optimization	13
Figure 1.5: Density-based topology optimization workflow	14
Figure 2.1: Geometry of example case	16
Figure 2.2: Density-based topology optimization workflow	17
Figure 2.3: Subdivision of flow domain (Alexandersen, 2023)	18
Figure 2.4: Continuous density distribution (Alexandersen, 2023)	20
Figure 2.5: SIMP Penalization Interpolation	20
Figure 2.6: RAMP Penalization interpolation	20
Figure 2.7: Effect of different α_{max} values with flow at the inlet at 40 ms	22
Figure 2.8: Checkerboard pattern in structural optimization (Sigmund & Diaz, 1995)	23
Figure 2.9: Visualization of the application of a filter with radius r_{filt}	23
Figure 2.10: Result of unregularized optimization of example collector	24
Figure 2.11: Optimization result of example case with $r_{filt} = 50\text{ mm}$	25
Figure 2.12: Hyperbolic projection function with various b values	26
Figure 2.13 : Visualization of sensitivity calculation using Finite Difference Method	27
Figure 2.14: Sensitivity field of intermediate iteration	28
Figure 2.15: Gradient-Based maximization of an arbitrary objective function J_ρ	29
Figure 2.16: Constrained optimization	29
Figure 2.17: Performance comparison between density field and extracted geometry	31
Figure 3.1: Topology optimization tutorial COMSOL – Tesla microvalve	33
Figure 3.2: Initial design search using correlation-based designs	35
Figure 3.3: Qualitative comparison of optimization programs used	36
Figure 4.1: Optimization case with boundaries	37
Figure 4.2: Two-dimensional domain	37
Figure 5.1: COMSOL test case of simple canal	40
Figure 5.2: Generated geometries of the canal test case	40
Figure 5.3: Static pressure curves over the outlets of the test cases	41
Figure 5.4: Non-converged flow in verification of $\varphi = 0,35$ case	43
Figure 5.5: Recirculation zones behind inlet canals	44

Figure 5.6: Density field generated by the refinement optimization	44
Figure 5.7: Comparison of coarse design (left) and refined design (right)	45
Figure 5.8: Close view of inlet area of refined design	45
Figure 5.9: Design generated by ColdStream	47
Figure 5.10: ColdStream's inlet structures	47
Figure 5.11: Close-up of velocity field of ColdStream verification simulation	48
Figure 5.12: Verification simulations of ColdStream (left) OpenFOAM (right)	48

NOMENCLATURE

Latin Letters

b	[-]	Projection parameter
J	[-]	Objective function
k	[m^2s^{-2}]	Turbulent kinetic energy
L	[m]	Length
n	[-]	Number of design variables
n	[-]	Vector giving the direction of the face normal
p	[Pa]	Static pressure
p	[-]	Curvature parameter for SIMP/RAMP interpolation
p_t	[Pa]	Total pressure
Re	[-]	Reynolds number
r_{filt}	[m]	Filter radius
T	[-]	Temperature
U	[ms^{-1}]	Flow velocity magnitude
u	[ms^{-1}]	Flow velocity vector

Greek Letters

α	[s^{-1}]	Inverse permeability
$\bar{\beta}$	[-]	Filtered design variable – porosity
β	[-]	Projected & filtered design variable – porosity
η	[-]	Projection Threshold
ρ	[-]	Design variable - porosity
Ω	[-]	Domain symbol

Abbreviations

CFD	<i>Computational Fluid Dynamics</i>
SIMP	<i>Solid Isotropic Material with Penalization</i>
RAMP	<i>Rational Approximation of Material Properties</i>
MMA	<i>Method of Moving Asymptotes</i>
HPC	<i>High Performance Computing</i>

1. INTRODUCTION

Topology optimization is an advanced computational design technique that computes the most efficient material distribution within a given design space to meet specific design objectives. The method has gained significant traction in the structural engineering world. It has found widespread use in industries where weight reduction is of utmost importance, such as aerospace and automotive engineering, where the organic-like structures generated through topology optimization can already be seen in use. Only in recent years has the tool been applied to be used for flow simulations, and now most commercial CFD solutions provide some sort of topology optimization feature. The goal of this paper is to use these programs to optimize the distribution system of a heat exchanger for implementation in a regenerative cycle engine. In doing so, each software will be evaluated for its efficacy, usability and performance.

The work for this thesis was done in collaboration with SAFRAN and the TU Berlin. For SAFRAN, the methods described in this paper were applied to optimize a real heat exchanger distribution system for a regenerative cycle engine concept. In an interest to protect the confidentiality of the project, a fictitious heat exchanger collector was conceived, which bears all the same features and challenges presented in the real case. All performance results are therefore not real, but the methodology used to achieve them is the same, and the performance values can be compared relatively to each other, to evaluate topology optimization practices.

1.1 Case Presentation

As demands for less emissions rise, new methods are being developed in the aviation industry to meet these ever-stricter goals. One promising technology is the recuperative cycle engine, which is a thermodynamic cycle engine that incorporates a heat recovery mechanism to improve its efficiency. In these systems, waste heat from the exhaust gasses is captured and reused to preheat the incoming air before it enters the combustion chamber. This process reduces the amount of energy needed to heat the working fluid, lowering fuel consumption and improving overall efficiency.

These engines operate on a Brayton cycle as shown in Figure 1.1, where the working fluid is compressed, heated, expanded and then cooled. In a recuperative cycle, the heat exchanger (recuperator) transfers a part of the heat from the exhaust to the incoming fluid, thus reducing the thermal energy required to bring it up to temperature.

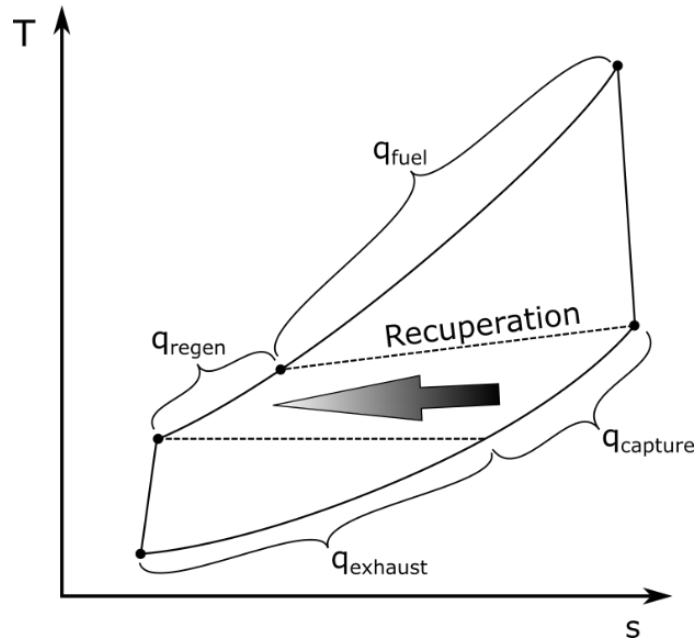


Figure 1.1: Heat recuperation effect shown in the Brayton cycle

To transfer the heat from the exhaust flow upstream, a heat exchanger is placed after the turbine and the hot exhaust flow is forced through it. Likewise, the cold airflow from the compressor is routed to the same heat exchanger before being injected into the combustion chamber for further heating. A schematic of such an engine is shown in Figure 1.2.

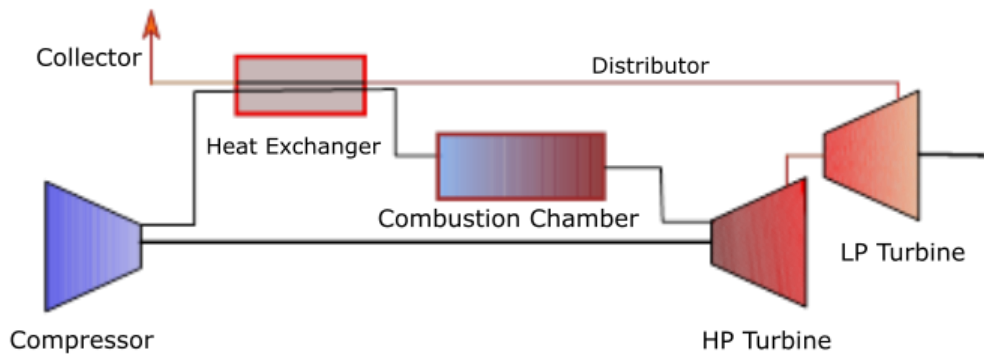


Figure 1.2: Schematic of a regenerative cycle engine

The recuperated Brayton cycle is not new technology, as it is widely implemented in applications where the weight and size of systems are of little concern. Gas turbines in power plants often come equipped with a recuperator as it greatly increases their efficiency. The technology has not yet made the transition to the aerospace industry, as space and weight come at a very high premium in aircraft. However, advances in additive manufacturing allow complex and lightweight geometries to be manufactured. This has sparked interest in implementing recuperators into aircraft engines.

There are two primary challenges that need to be overcome for a recuperated cycle aircraft engine. The addition of ducting and a heat exchanger in the flow of the engine introduces flow losses into the system. These losses must be minimized, otherwise any gain in efficiency will be offset for no net gain or, in the worst case, net efficiency loss. Since space is limited on aircraft, the parts must also be as compact as possible. This is a classic optimization problem where volume is the constraint and pressure losses are the objective function to be minimized.

Topology optimization will be applied here to generate geometries that guide the flow to and from the heat exchanger with as little pressure loss as possible. The heat exchanger itself will not be modified, as CFD topology optimization with heat transfer is still in its infancy. However, the heat exchanger will be taken into consideration as an outlet boundary condition for the distributor and an inlet boundary condition for the collector.

In doing so, three separate CFD programs will be used, to find out which offers the best solution. The main challenge is that the modeled flow is highly turbulent, making calculation convergence difficult to achieve. Nonetheless, part of the scope of this work is to find methods with which topology optimization can be used to achieve optimal designs using these programs. While working with them, any quirks, difficulties and advantages will be noted to ultimately choose one that is best suited for this application.

1.1.1 Potential of Topology Optimization

Topology Optimization is a mathematical method which optimizes the distribution of material in a given domain under a set of loads, constraints, and boundary conditions with the goal of maximizing the performance of the system. The advantage of topology optimization is that the design can attain any shape within the design space, instead of dealing with predefined configurations. The algorithm places material where it is most effective, and leaves it out where it is not, effectively generating an optimal structure for the case that it was given.

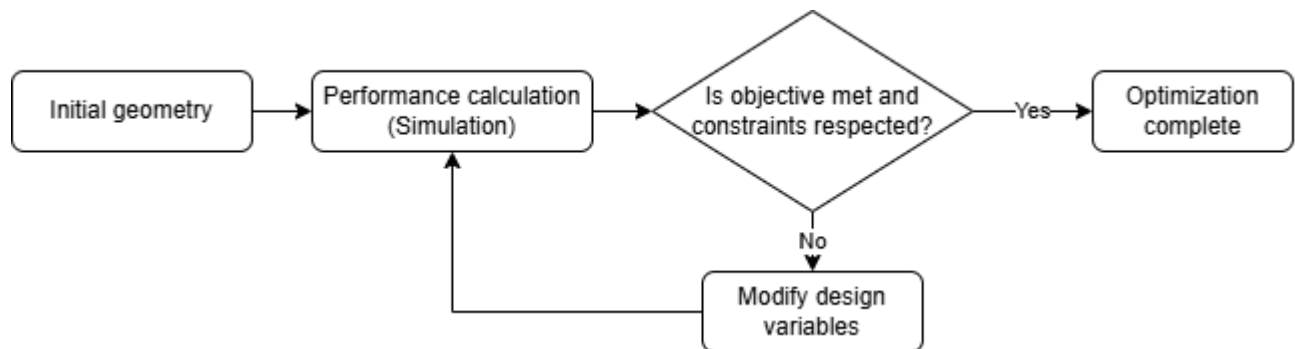


Figure 1.3: General optimization approach

Optimization is an age-old engineering practice which involves modifying and testing a design iteratively to reach a result which satisfies certain requirements. This can be done either manually or automatically with the help of a computer. The advantage of manual optimization is that the only limiting factor to how the design can be changed is the imagination of the engineer. In this case there is effectively an infinite number of design variables that can be changed, and it is up to the engineer to choose which ones need to be modified to reach the goal of finding an optimum.

On the other hand, automating the process requires discretization of the design variables, as it is not feasible to calculate the effect of changing an infinite number of variables with respect to the imposed objectives. For this reason, parameter optimization was the first form of automation. Doing so, the engineer could define a finite list of parameters which could be changed during the optimization process. This can accelerate finding an optimum, as the design update is performed automatically. However, these parameters must be chosen wisely so that they have a meaningful impact on the objectives. Likewise, it can be a very tedious process if there is a large amount of design variables to be defined.

Popular parameters chosen for optimization are size and form parameters. Using Figure 1.4 as an example, if the result from topology optimization is assumed to be the optimum, then the

initial geometry could never become the optimum by only changing those things. It shows the importance of the choices taken during initialization and parameter definition, since the result will be topologically similar. On the contrary, with topology optimization it is theoretically possible to generate the optimum geometry without having to make any design choices.

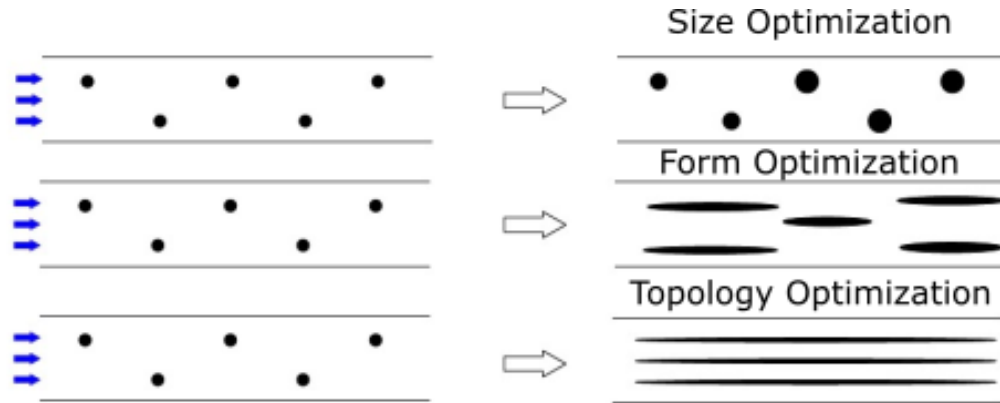


Figure 1.4: Different types of automated optimization

For designing flow components, manual optimization is a tedious process, as changes to geometry may have unexpected downstream or upstream effects. It is therefore difficult to predict how exactly a change in geometry will affect the objective value. This is especially true with turbulent flow, as the chaotic nature of it is unintuitive. With parameter optimization, this problem is overcome by calculating a sensitivity of the objective function respective to the design parameters. This removes a large amount of the guesswork. It is no longer needed to understand the whole flow field and how the geometry affects it. It suffices to know which parameters will have the greatest effect to reach the intended goal, and to set them as design parameters for the optimization algorithm to find their optimum values.

With topology optimization, the entire calculation domain is filled with design parameters. It is left up to the algorithm's discretion to figure out the exact material distribution needed to reach the objective. As such, the guesswork is reduced once again to just a few optimization parameters which need to be set to ensure a successful calculation. This is only made possible using the adjoint calculation method, which provides a computationally cheap way to determine the sensitivity of design variables. It was first introduced for turbulent flow problems by (Nielsen, Lu, Park, & Darmofal, 2004) in 2004. Irrespective of the number of design variables, the computational cost of the adjoint calculation does not change. It is what enables topology optimization to become computationally feasible and its recent addition to commercial CFD solvers is what allows topology optimization to be performed.

1.1.2 Method Maturity

Topology optimization is a method that emerged from shape and size optimization in the late 1980s for solid mechanics, with (Bendsoe & Kikuchi, 1988) being cited as the seminal paper on topology optimization. In that field, it has been used for many years and has been improved to become a reliable tool for design generation. The limiting factor, which hindered its use, was a lack of manufacturing capabilities, as the optimum structures were impossible to create using traditional manufacturing techniques. The advent of additive manufacturing broke that barrier and ever since, topology optimized parts can be found in applications where weight reduction is the main objective.

1.1.3 Density-Based Method

The most common and most mature implementation of topology optimization is the SIMP (Solid Isotropic Material with Penalization) density-based method. This method relies on a discretization of the optimization domain, where each cell is assigned a variable density, which ranges from 0 to 1. This density is multiplied by a penalization factor. In the case of structural optimization for stiffness, a penalization factor could be applied which forces the structural stiffness to tend toward zero if the density in that cell is zero, effectively modeling a void in the structure. Intermediate density values therefore represent a fictitious solid and need to be mitigated. Ideally the result is a binary field of density values. A typical workflow for density-based topology optimization is shown in Figure 1.5

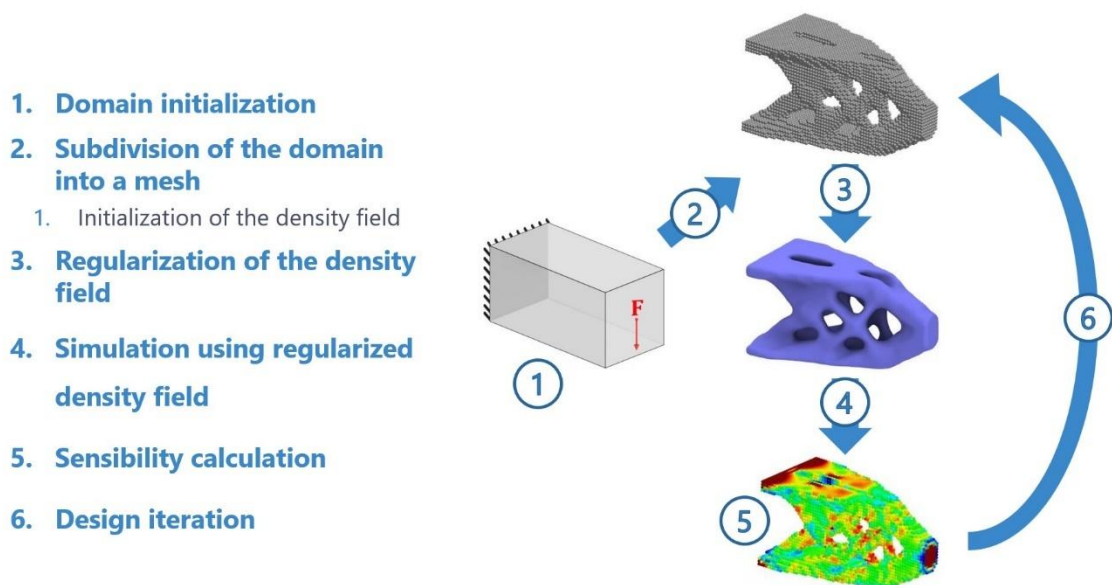


Figure 1.5: Density-based topology optimization workflow

The advantage of the density-based method is that the mesh does not change after being initialized. Only the density values of the cells are changed between iterations. This reduces the complexity of the algorithm, but it means that the mesh cannot be adapted to the geometry at every step. This can induce inaccuracies in the forward calculation since a high mesh resolution is required in areas with high stress concentrations. To remedy this, the initial mesh must be fine enough to be able to capture enough detail in every part of the domain.

1.1.4 Level-Set Method

A new method started to emerge in the early 2000s with (Allaire, 2002), who published a level-set shape optimization approach, and (Wang, Wang, & Guo, 2003) who developed a method for level-set topology optimization. This approach differs substantially from the density-based one, because instead of changing the density field at every iteration, the boundary between solid and void is morphed. It can be thought of as an advanced form of shape optimization, with the added capability of allowing topological changes, such as boundary emerging and splitting. There are two ways to model the boundary. Either a body-fitted mesh is generated to conform to the new

boundary, or to use immersed boundary models, which allows the computation of a flow on a grid which does not conform to the boundary.

The advantage of remeshing is that with proper tuning, an adapted mesh can be generated for the given solid, and zones which require fine meshing can be refined. Furthermore, the defined boundary is a more direct physical model than applying boundary models to a fixed mesh. The boundary is also a continuous line, unlike the jagged boundaries generated from density-based methods. This is an advantage for applications where surface roughness is a factor to be taken into consideration, such as for aerodynamic structures. Though the level-set solution for topology optimization has promising advantages and is used for structural optimization, it has not yet made the transition to fluid flow optimization, and there is currently no software available that allows level-set optimization for turbulent flows.

1.1.5 Application in Fluid Dynamics

When applied to fluid problems, the basic question to be answered changes from “where should the holes be?” as is the case in solid problems, to: “where should the fluid flow?” and the governing equations are much different. The inherent non-linearity of the Navier-Stokes equations makes answering this question challenging. Therefore, the first solutions proposed were made for specific flow cases where the governing PDEs could be simplified. The first application was achieved in 2002 by (Borrvall & Petersson, 2002), where it was used to optimize structures for Stokes flow, using an approach based on lubrication theory. This flow was used since it allows for the linearization of the Navier-Stokes equations as inertial forces are neglected. Later, the work was expanded for two-dimensional laminar flow by (Gersborg-Hansen, Sigmund, & Haber, 2005).

Density-based topology optimization started to be developed when variable porosity was introduced by (Evgrafov, 2005). In this paper, the governing equations were expanded to the Darcy-Stokes equations, which can consider porous materials. Over time this approach was developed for three-dimensional applications, laminar flows, unsteady flows and most recently turbulent flows. The first implementation of turbulent flow topology optimization was done by (Kontoleonos, Papoutsis-Kiachagias, Zymaris, Papadimitriou, & Giannakoglou, 2013). They used the Spalart-Allmaras turbulence model with a “frozen turbulence” assumption. This neglects variations of the turbulence field with respect to the design variables. Along with an adjoint solver for the turbulence variables, turbulent topology optimization was made possible.

Since then, fluid topology optimization has been made accessible through commercial CFD programs with adjoint solvers. It is important to note that the implementations are often rough around the edges, as the method is still being perfected. A significant challenge is the calculation stability of highly turbulent flows, as it takes just a single non-convergent simulation to interrupt the optimization process. The goal of this paper is to evaluate these programs by optimizing a flow structure with a highly turbulent flow. The advantages of each will be presented at the end.

2. METHODOLOGY – TOPOLOGY OPTIMIZATION

In this section, the process of density-based topology optimization for CFD is presented as it is implemented in the commercial programs used during the work for this thesis.

2.1 Example Case

To better illustrate every step taken in the process, a fictitious collector case will be used. On which every action taken during the optimization iteration will be visualized. It is intended to showcase the optimization parameters, and how they can affect the result.

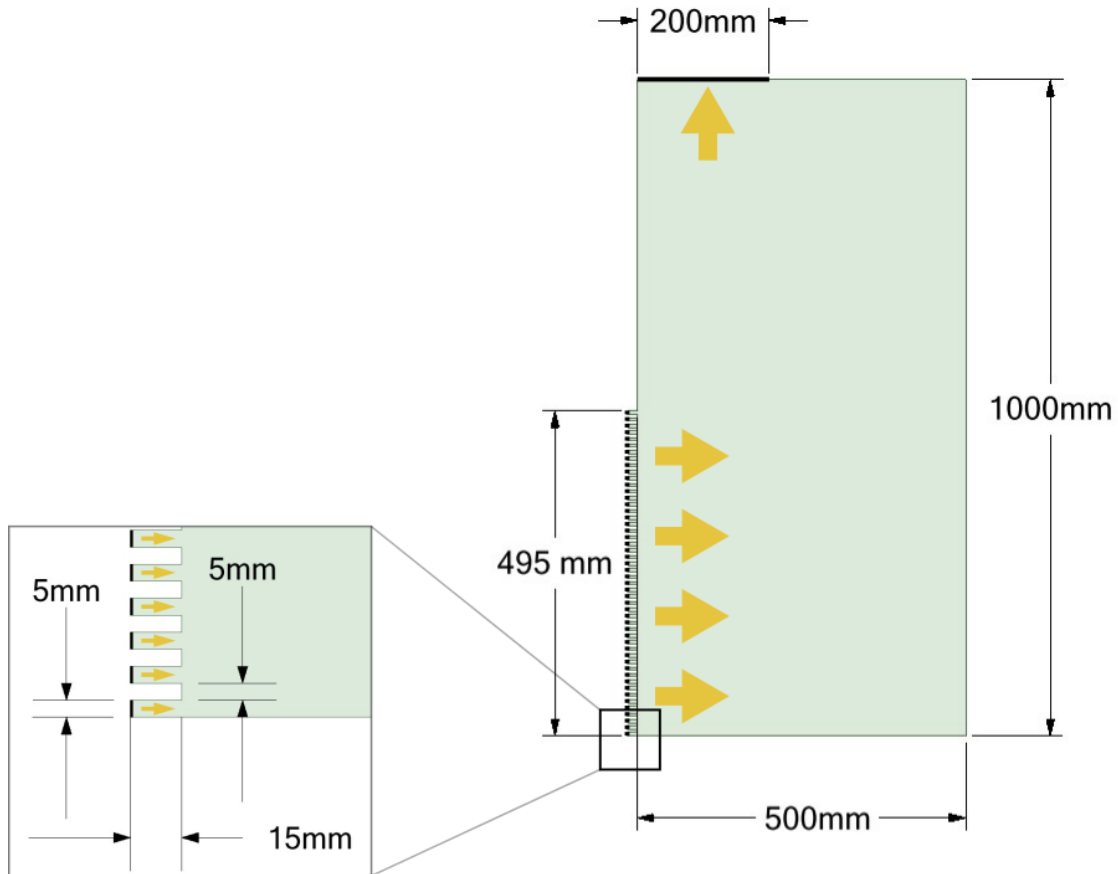


Figure 2.1: Geometry of example case

The example case has both a very large flow area, and small inlet canals which are meant to represent the miniscule channels in a real heat exchanger. A geometry like this is particularly difficult due to the differences in feature sizes needed to effectively reduce pressure losses, as the small inlet channels need a very refined mesh to be able to generate structures around and between them, whereas the flow envelope in the large surface requires macroscopic structures.

2.2 Optimization process

The workflow of the density-based method is depicted in Figure 2.2 and will be followed in the following sections.

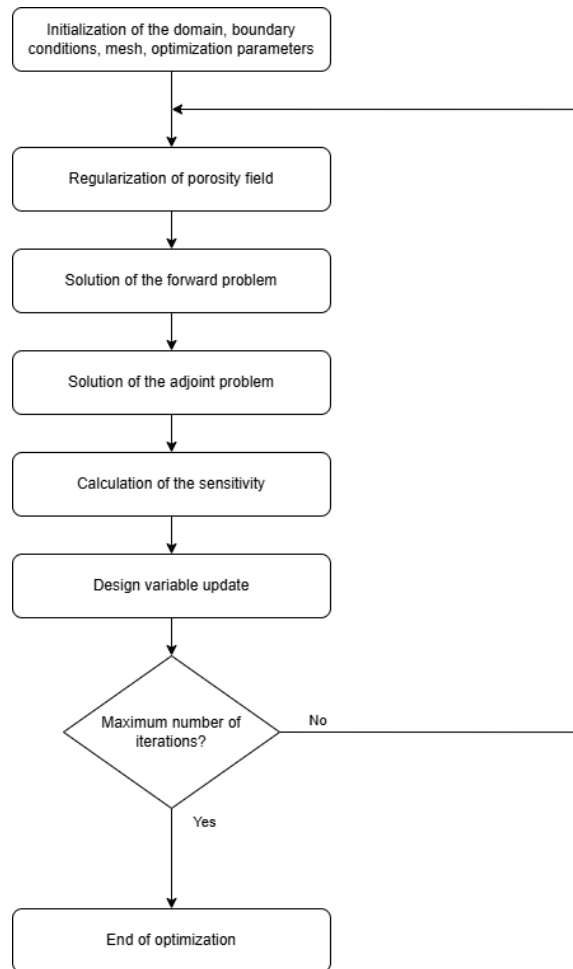


Figure 2.2: Density-based topology optimization workflow

Being an iterative process, it is necessary to define a maximum number of iterations which should be calculated, after which the solution is saved. Alternatively, an end condition may be the change in the value of the objective function compared to the value from the previous iteration. If the change is below a certain threshold and the constraints are all met, it can be assumed that an optimum has been reached, and the optimization can be stopped.

2.2.1 Initialization

The initialization of an optimization case is similar to that of a normal simulation, where the geometry, the boundaries, the boundary conditions and the mesh must be defined. When meshing for topology optimization, boundary layer elements on the walls are not necessary as the walls are yet to be generated. Instead, the mesh should be sufficiently fine to be able to capture viscous effects to a reasonable degree, and to generate smooth structures, since the same mesh is used for the forward solution and structure generation. The geometry and boundary conditions for the example case can be seen in Figure 2.1.

Furthermore, there are multiple optimization parameters which need to be set, such as the maximum penalization factor α_{max} , the filter radius r_{filt} , and the projection parameter b . These are the principal values which need to be chosen well, as they will influence the solution. The effects of these parameters will be explained in the following sections.

Lastly, to guide the optimization in the desired direction, at least one objective function must be given, which the algorithm will attempt to either minimize or maximize. In this case the objective function will be the total pressure drop between the inlet and the outlet of the domain. This value should be minimized. If multiple objective functions are needed, then they also should be weighted accordingly to their importance. In addition, an optimization constraint should be set to limit the algorithm. This constraint is defined as the fraction of volume taken up by the fluid within the domain. The amount of the solid region is calculated by integrating the density design variable ρ over the volume of the domain as shown in equation (2.1).

$$\varphi = 1 - \frac{\int_V \rho \, dV}{\int_V dV} \quad (2.1)$$

In this chapter all optimizations done with the example case for parameter visualization were done with a volume constraint of 0,5.

2.3 Variable Porosity Model – Design Variable ρ

The base hypothesis which enables the topology optimization in CFD is that of the variable porosity model. With it a solid flow obstruction can be modeled in the flow field. To do this, the domain is split into two subdomains. One contains the fluid and the other contains the solid as seen in Figure 2.3, where Ω_s represents the solid domain and Ω_f the fluid domain.

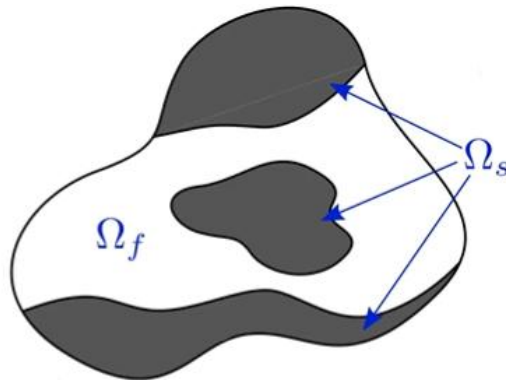


Figure 2.3: Subdivision of flow domain (Alexandersen, 2023)

These domains are represented by the design variable ρ , which is defined following equation (2.2):

$$\rho = \begin{cases} 1 & \text{if } \in \Omega_s \\ 0 & \text{if } \in \Omega_f \end{cases} \quad (2.2)$$

ρ is the design variable used for optimization, and it represents the material distribution. When $\rho = 0$ in a cell, it belongs to the fluid domain where flow is not penalized and when $\rho = 1$ it represents the solid domain. To model the obstruction within the fluid, an impermeability factor is

introduced as a function of ρ . This factor acts as a Brinkmann penalty term and is added to the NS-equations (2.3) as a body force f_i (2.4).

$$\begin{aligned} \rho u_j \frac{\partial u_i}{\partial x_j} - \mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \frac{\partial p}{\partial x_i} - f_i &= 0 \\ \frac{\partial u_i}{\partial x_i} &= 0 \end{aligned} \quad (2.3)$$

$$f_i = -\alpha u_i \quad (2.4)$$

The function of f_i is to penalize the flow velocity with the use of a Brinkmann penalty factor α . This factor itself is a function of the local density ρ as described in equation (2.5).

$$\alpha(\rho) = \begin{cases} \infty & \text{if } \rho \in \Omega_s \\ 0 & \text{if } \rho \in \Omega_f \end{cases} \quad (2.5)$$

The Brinkmann penalty factor is commonly used to model porous media within fluid flow. By setting the penalty factor to a large value in the solid domain, it will drive the velocity in these cells toward zero, as in an infinitely impermeable porous medium. By setting the penalty factor to zero in the fluid domain, it removes the porous effect in cells which are supposed to represent the fluid.

2.4 Design variable relaxation

While a binary definition of the design variable ρ would be the most representative for differentiating between solid and fluid, the definition must be relaxed to allow intermediate values. Fundamentally, the optimum design is found by finding the global maximum or minimum of the objective function with respect to the design variables. An efficient and numerically stable method of finding them is by using gradient-based optimization algorithms, which calculate the derivative of the objective function to find how the design variables should be updated. Popular algorithms are the Method of Moving Asymptotes (MMA), as presented by (Svanberg, 1987) and the Globally Convergent Method of Moving Asymptotes (GCMMA), as presented by (Zillober, 1993). As these algorithms rely on gradient calculations, the objective function must be differentiable. Therefore, a continuous density definition is needed to facilitate optimization. The effect of this is illustrated in Figure 2.4, where the material distribution from Figure 2.3 has been relaxed to allow for intermediate values.

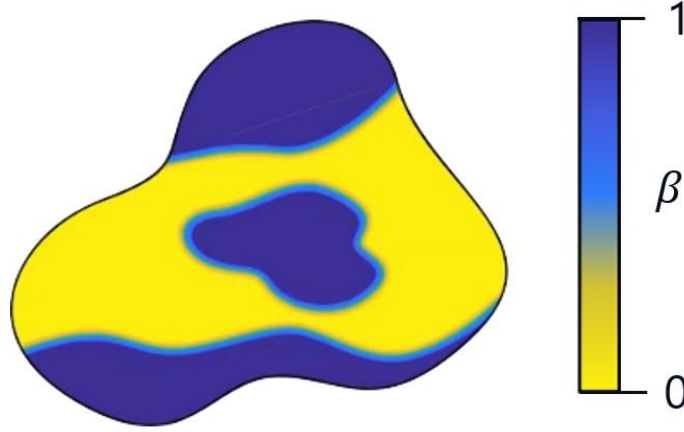


Figure 2.4: Continuous density distribution (Alexandersen, 2023)

As the design variable ρ has now been relaxed, the previous assignment of the Brinkmann penalty factor from equation (2.5) must be expanded. To do this there are interpolation schemes which are used in the realm of topology optimization to assign penalization values for a continuous density distribution. The most common are SIMP (Solid Isotropic Material with Penalization) and RAMP (Rational Approximation of Material Properties)

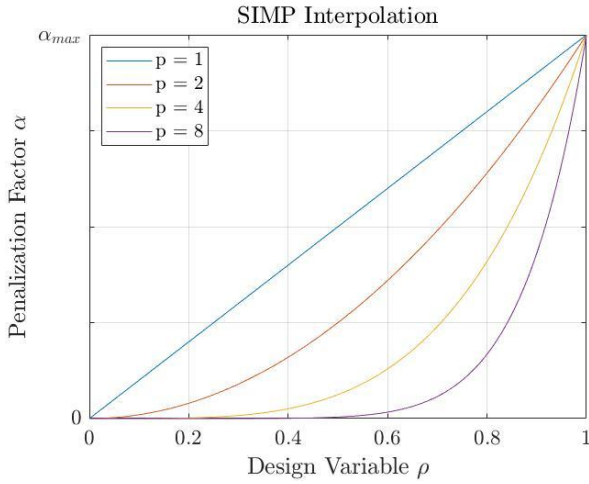


Figure 2.5: SIMP Penalization Interpolation

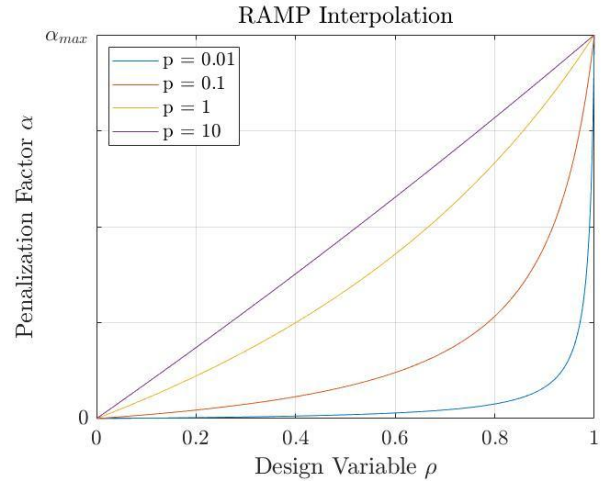


Figure 2.6: RAMP Penalization interpolation

The aim of interpolating the penalization factor is to reduce the amount of intermediate density values within the design variable field, because they are not useful for the optimization. The goal is to find the best distribution of solid material to guide the flow, and the addition of the porosity model is just used to facilitate the optimization. Thus, a mostly binary distribution of ρ is desired as a result. Ideally, the interpolation parameter p should be chosen such that the function becomes as steep as possible to suppress these intermediate values. Through numerical experiments it has been shown that aggressive interpolation can destabilize the calculation, since small changes to the design variable can have drastic effects on the penalization factor. With the addition of an interpolation function $f_{interp}(\rho)$, equation (2.3) can be expanded to:

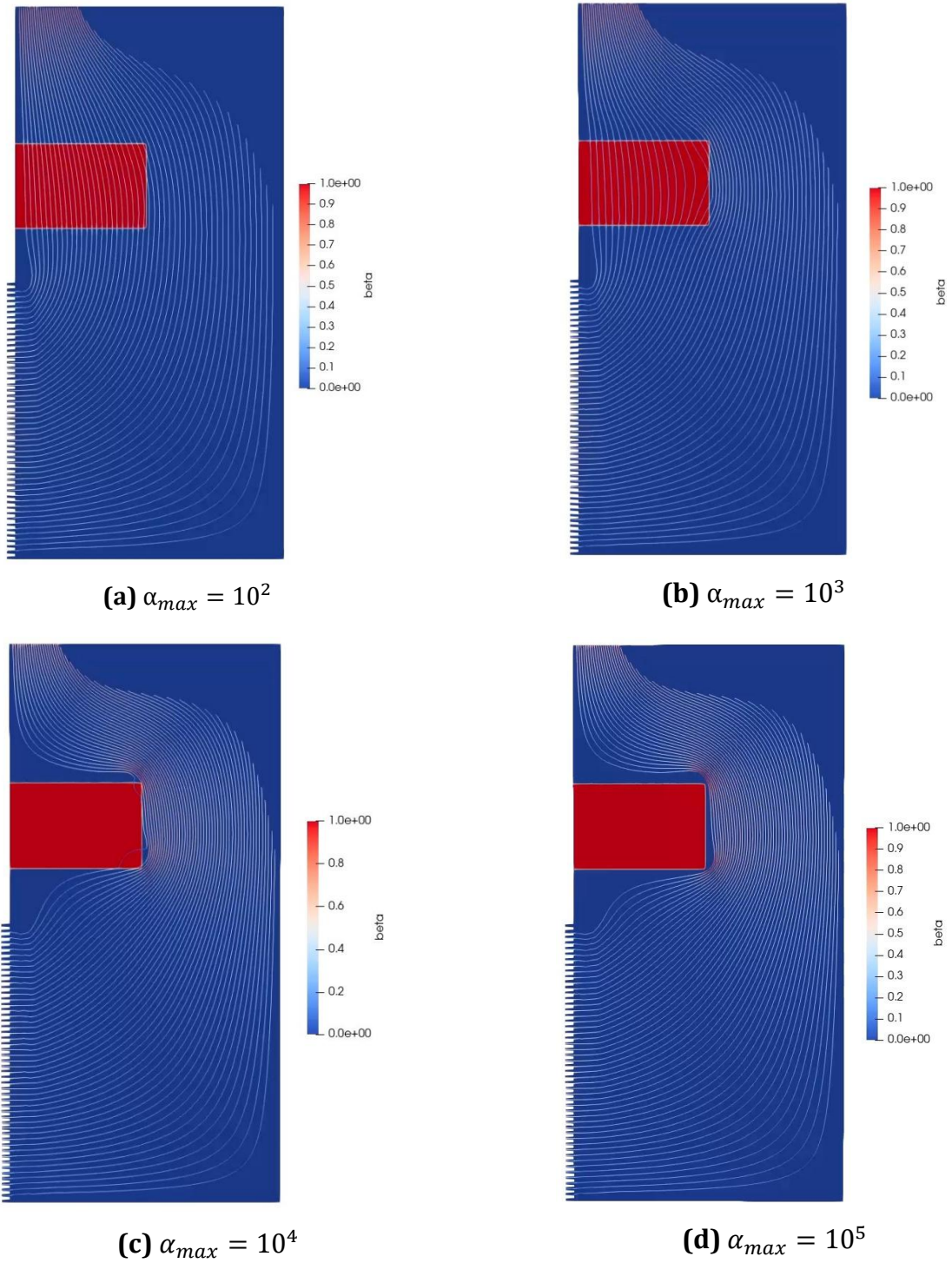
$$\rho u_j \frac{\partial u_i}{\partial x_j} - \mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) + \frac{\partial p}{\partial x_i} - f_{interp}(\rho) u_i = 0 \quad (2.6)$$

$$f_{interp}(\rho) = \begin{cases} \alpha_{max} & \text{if } \rho = 1 \\ 0 & \text{if } \rho = 0 \end{cases} \quad (2.7)$$

Regardless of the interpolation function chosen, the penalization factor α reaches a maximum value when $\rho = 1$. This maximum value is called α_{max} and is an important parameter that needs to be set before an optimization run. According to equation (2.5), α_{max} should be chosen as large as possible to effectively penalize the flow in the solid domain. However, as with very steep interpolation parameters, setting α_{max} too high can destabilize the optimization, as even cells with very small density values would block flow. An optimum value must be chosen for α_{max} , since a low value would result in flow being insufficiently penalized in the solid domain, and a high value would make the optimization numerically unstable. Furthermore, the penalization factor is multiplied with the flow velocity, making the choice of α_{max} highly dependent on the expected flow velocities.

Figure 2.7 depicts the effect of different values of α_{max} in the example collector case, where a solid obstacle ($\rho = 1$) was placed in the way of the flow between inlet and outlet. In Figure 2.7(a), the solid is ignored by the flow; the flow lines are not even distorted. The solid zone in Figure 2.7(b) exhibits similar behavior, although there is more flow being forced around the obstacle. With $\alpha_{max} = 10^4$, flow is being penalized significantly, but a small amount can be seen passing through the corners of the rectangle; α_{max} needs to be increased further. Figure 2.7(d) shows an acceptable amount of penalization, as there are no streamlines in the solid region. $\alpha_{max} = 10^5$ is the minimum acceptable value for this case, but higher values could be needed if thin structures were to be generated. For following calculations, α_{max} is set to 10^6 , because the calculation remains stable, and it ensures that flow is blocked even in smaller structures, such as those around the small inlet canals.

Figure 2.7: Effect of different α_{max} values with flow at the inlet at 40 m/s



2.5 Regularization of Design Variable Field

As mentioned in the previous chapter, topology optimization is prone to numerical instabilities. These are described in (Sigmund & Petersson, Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima, 1998). The most common issue encountered, which is specific to topology optimization, is that of checkerboard patterns:

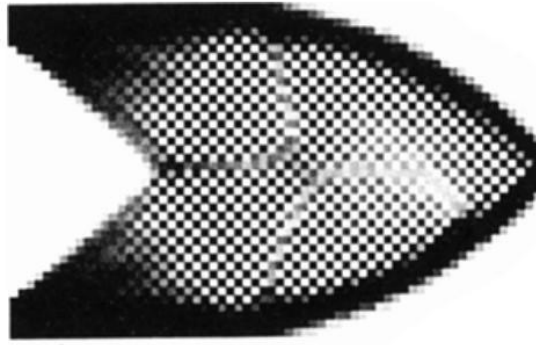


Figure 2.8: Checkerboard pattern in structural optimization (Sigmund & Diaz, 1995)

The checkerboard phenomenon occurs because it allows for a “light” structure with good mechanical properties. Since weight reduction is often the objective in structural topology optimization, the checkerboard pattern is a favorable configuration. To prevent the generation of this unrealistic geometry, the design variable field is regularized by filtering and projecting. The procedure is presented in (Bendoe & Sigmund, 2004).

2.5.1 Filtration

The filtering process of β is comparable to the use of a convolution filter in image processing to blur an image. In topology optimization, it can be thought of as “smoothing” the density field ρ (Figure 2.9). To do this, a filter radius must be chosen, with which a circle is defined. Generated structures that are smaller than the size of this circle will be eliminated. The filtered density variable is referred to as: $\bar{\beta}$.

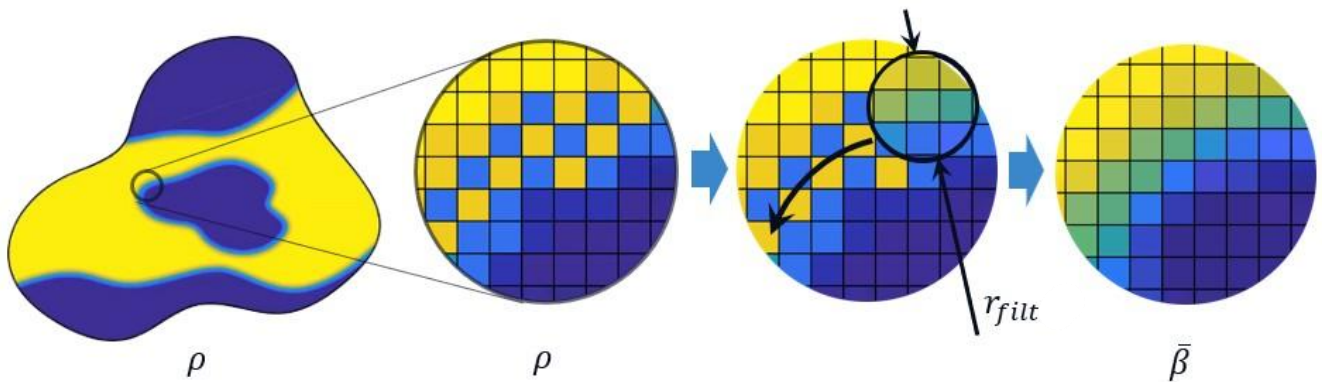


Figure 2.9: Visualization of the application of a filter with radius r_{filt}

The filter radius r_{filt} becomes yet another important optimization parameter which can dictate the resulting structure. If the filter radius is larger than the mesh element size, then checkerboard patterns will be suppressed. Likewise, it renders the solution independent to further mesh refinement, as structures at the scale of a single element will be smoothed out.

In the realm of topology optimization for fluid flow, where pressure losses are to be minimized, the material constraint is often inversed. Filling the optimization domain with solid material means that the flow field itself is smaller and therefore the flow component that holds it lighter. During the simulations done for this paper, the checkerboard pattern was never observed. Nevertheless, the filter radius can serve to enforce manufacturing constraints, such as a minimal feature size of a 3D-printer. It can also prevent small channels from being formed, as in Figure 2.10,

which might marginally reduce pressure loss, but the added weight and complexity of the added ducting would negate any gain.

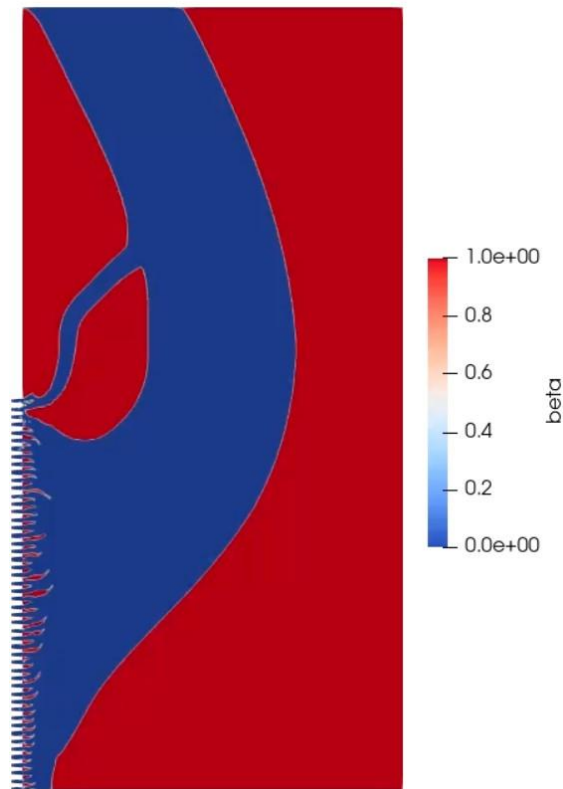


Figure 2.10: Result of unregularized optimization of example collector

The process of filtering the design variable helps guide the optimization in the right direction, however an unwanted side effect is that the border between fluid and solid is blurred, leaving many elements with intermediate densities. Figure 2.11 shows the result of optimizing the example collector case with a large filter size and the intermediate densities are visible.

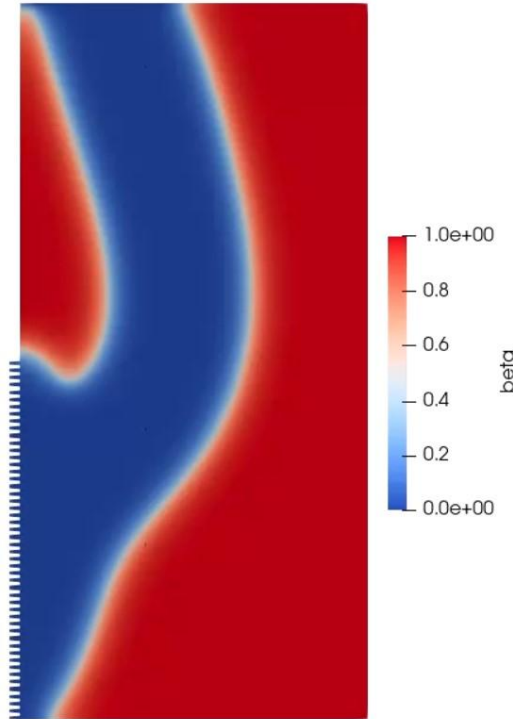


Figure 2.11: Optimization result of example case with $r_{filt} = 50 \text{ mm}$

2.5.2 Projection

To eliminate the unwanted design variable values from the filtered field $\bar{\beta}$, a second processing step is used which is named projection. The addition of a projection step was first proposed by (Guest, Prévost, & Belytschko, 2004), where a Heaviside projection is implemented. The Heaviside function is a non-differentiable step function, which projects the design variables into binary values. Any value below a threshold η is set to 0, and any value above η to 1. If β is the projected design variable field, then the Heaviside projection could be expressed as follows:

$$\beta(\bar{\beta}) = \begin{cases} 1 & \text{if } \bar{\beta} > \eta \\ 0 & \text{if } \bar{\beta} < \eta \end{cases} \quad (2.8)$$

Although the Heaviside projection is effective at removing intermediate densities, its application in areas where the density is very close to the threshold η can cause irregularities to appear such as jagged edges. To remedy this, (Xu, Cai, & Cheng, 2010) presented a differentiable Heaviside projection function. It is the most widely adapted projection method, and its function is defined by equation (2.9).

$$\beta = \frac{\tanh(\eta b) + \tanh(b(\bar{\beta} - \eta))}{\tanh(\eta b) + \tanh(b(1 - \eta))} \quad (2.9)$$

This hyperbolic projection function introduces yet another important optimization parameter, which is b . It controls the steepness of the function at the threshold η , with higher values making the projection more aggressive, thus imitating the Heaviside projection. The effect of varying b is displayed in Figure 2.12.

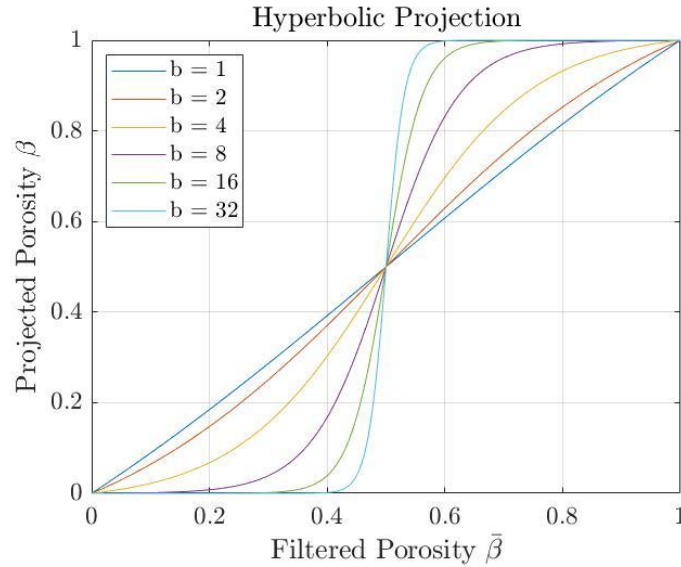


Figure 2.12: Hyperbolic projection function with various b values

2.5.3 Summary

Regularization was implemented into topology optimization as a method to mitigate numerical instabilities and to render the solutions more feasible with respect to manufacturing constraints. In the specific case of topology optimization of turbulent flow components to reduce pressure loss, it was noticed that the best solution for any given domain was to leave it empty and not add any solid material. Since this is deemed an uninteresting solution, a volume constraint is implemented, to force the optimizer to fill the domain with a certain amount of solid. Although the constraint is based on an inequality, meaning that more solid material than the constraint defines is permissible, the algorithm was never observed to add more solid than allowed. Therefore, the checkerboard instability is not a concern for this application, since it is a numerical exploit that the optimizer uses to decrease the overall amount of solid.

Both r_{filt} and b are governing parameters which have a strong influence on the result of optimization and should be chosen according to the application. Since checkerboard patterns are not a concern, r_{filt} should be chosen to respect manufacturing limitations or to suppress the generation of small flow channels which would unnecessarily complexify the structure for minimal pressure loss minimization. However, with modern additive manufacturing which allows small structures to be fabricated, and a goal to generate such detailed topologies, regularization can be omitted.

2.6 Sensitivity calculation using the adjoint method

To find an optimum design, the design variables must be updated between each iteration in the optimization loop. It would be infeasibly costly to find the optimum through “brute force”, especially with the amount of design variables needed in topology optimization. Therefore, a sensitivity is calculated every iteration to guide the optimization towards an optimum. It is a quantification of the effect a change of a design variable would have with respect to the objective function. In essence, it answers the question of “where should I put material?” or “where should I take away material?”.

2.6.1 Adjoint method

Topology optimization is enabled by the so-called adjoint method, which is a computationally efficient manner of calculating the sensitivity of the objective function with respect to the design variables. Most notably requiring just one evaluation calculation per objective function, regardless of the number of design variables.

2.6.1.1. Finite Difference Method

Traditionally, to evaluate the gradient of an objective function, the effect of a small change of every design variable would be evaluated. For every change, the whole design field needs to be calculated to evaluate its effect on the objective function J . This is called the method of finite differences, and it works well when there are few design variables to be evaluated. However, it scales poorly with an increasing number of design variables, as showcased in Figure 2.13.

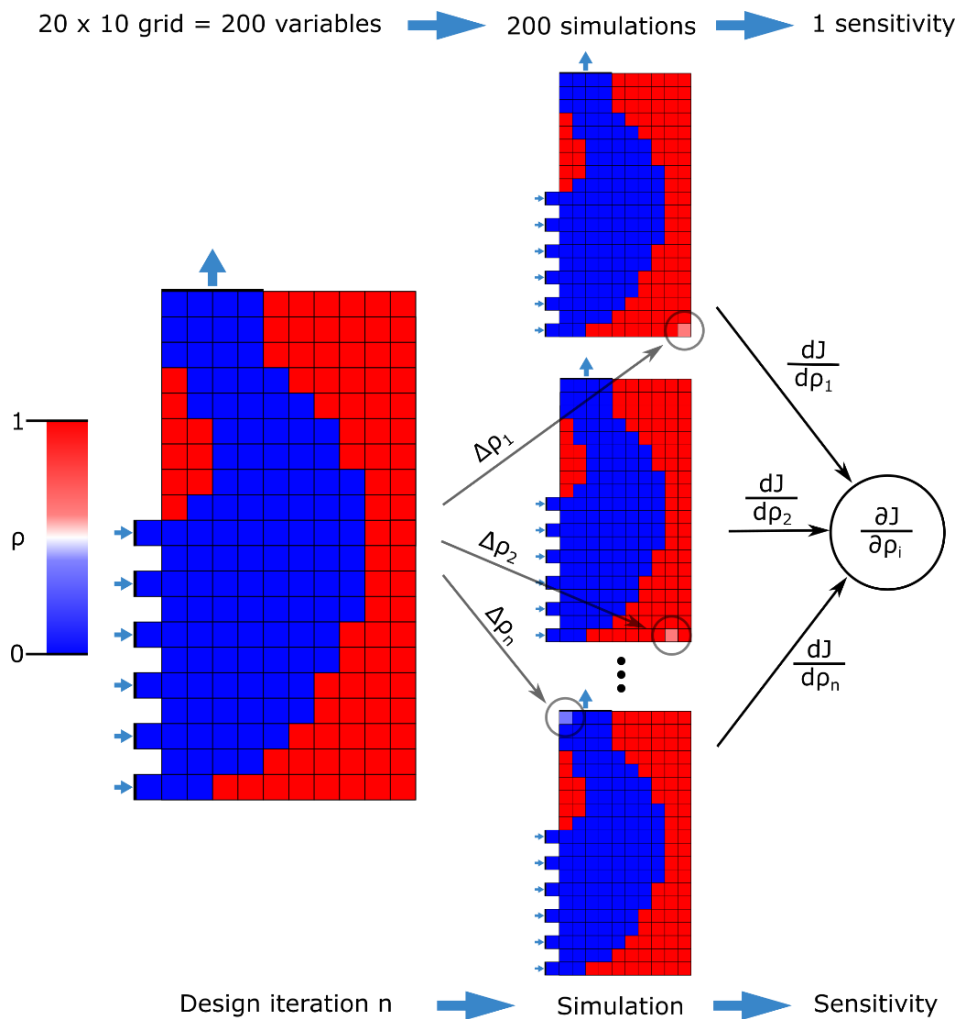


Figure 2.13 : Visualization of sensitivity calculation using Finite Difference Method

Topology optimization inherently has many design variables to evaluate, so the finite difference method quickly reaches its limits. Even the very coarse 20x10 mesh from Figure 2.13 requires 200 simulations for the sensitivity calculation plus one for the solution of the forward problem. If n is the amount of design variables that are to be optimized, then each optimization iteration needs $n + 1$ simulations. Since n is equal to the amount of mesh elements for density-based topology optimization, any mesh refinement would exponentially increase the calculation cost, since the amount of simulations to be performed would increase, as well as the computation cost of each one.

2.6.1.2. Adjoint Calculation

To calculate the gradient more efficiently, (Jameson, Schmidt, & Turkel, 1988) derived the adjoint of the NS-Equations. The adjoint problem is comprised of a set of field equations which relate the sensitivity of the objective function to the flow field. These adjoint field equations are linear differential equations, which are solved in a similar fashion to the forward problem, where all flow variables are replaced by their adjoint version. The adjoint variables themselves are a mathematical construct which enable the calculation of the sensitivity.

The determination of the adjoint solution is often referred to as solving the forward problem “backwards”, since perturbations of the adjoint variables tend to propagate downstream instead of upstream. This makes the adjoint solution sensitive to diverging, despite the adjoint equations being linear. In effect, care must be taken during the setup of the case, as unstable flow conditions such as flow separation can destabilize the adjoint calculation, thus destabilizing the optimization. During the numerical experiments conducted for this work, it was noticed that mesh quality, boundary condition definitions and flow conditions all had a profound impact on the stability of the adjoint calculation.

Once calculated, the adjoint solution can be combined with the forward solution to determine the sensitivity field. Using the sensitivity, the optimization algorithm decides where material should be placed or removed to affect the objective function in the desired manner.

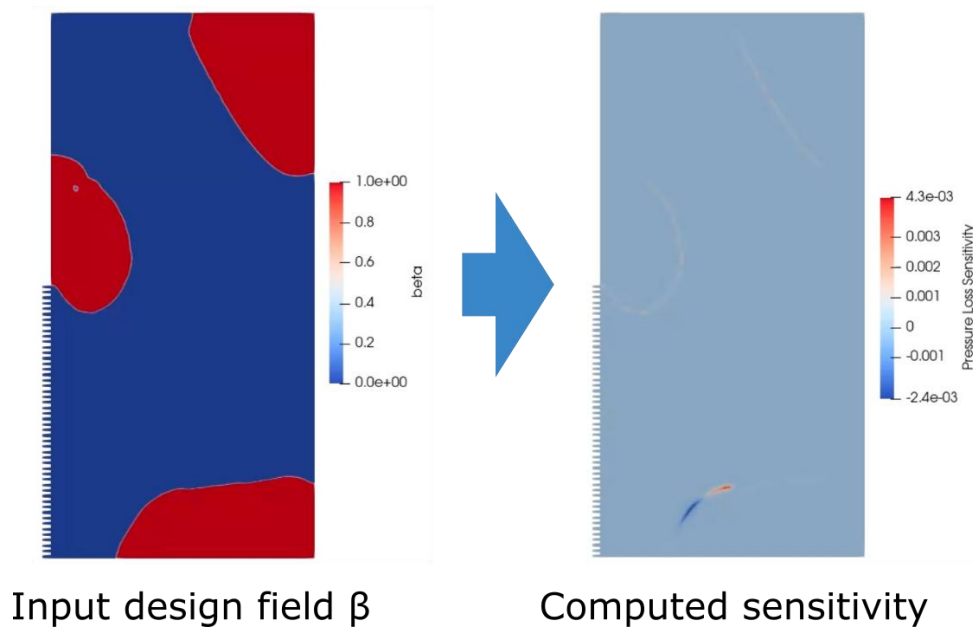


Figure 2.14: Sensitivity field of intermediate iteration

For example, from the sensitivity field shown in Figure 2.14, the design variable field for the next iteration can be deduced. To minimize the objective function, which in this case is pressure loss, material should be added to the regions in red and removed from the regions in blue. The amount of change in the design variables is then computed by the optimization algorithm.

2.7 Optimization algorithm

The objective function in an optimization problem can be expressed as J and is a function of the design variables of the system. In density-based topology optimization the design variables are the material densities of each cell in the mesh ρ_i . Therefore, the objective function can be written as $J(\rho_i)$. The goal is to find the design variables which maximize or minimize the function $J(\rho_i)$.

2.7.1 Gradient-Based optimization

To visualize how gradient based methods work, an example is proposed with a single design variable and objective function in Figure 2.15. These methods are highly dependent on the initialization of the design variable, as they iterate the design variable by following a positive gradient for maximization, or a negative one for minimization. Though this makes them fast to converge and numerically stable, they can get stuck on local minima. Figure 2.15 displays an example with just one design variable, but in the case of topology optimization, there can be millions.

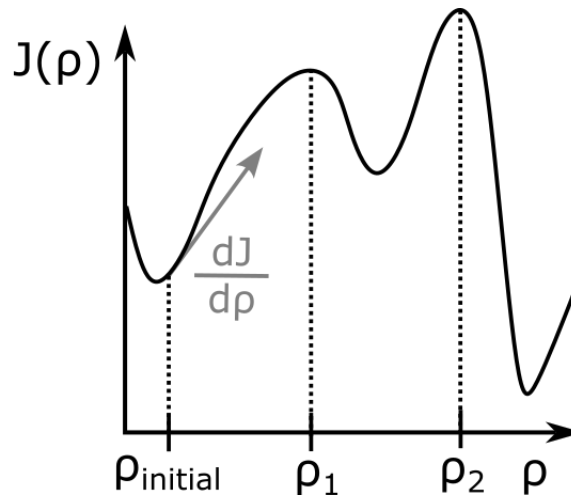


Figure 2.15: Gradient-Based maximization of an arbitrary objective function $J(\rho)$

Optimization constraints are often employed to guide the optimization towards more physical solutions and to enforce certain conditions, such as a volume constraint for the example collector. This can further impede the optimizer from finding an actual optimum. As seen in Figure 2.16, following the positive gradient to maximize the objective function would lead towards ρ_{min} , which is an imposed constraint. Since the solution where ρ is below ρ_{min} would be considered infeasible, the optimizer would converge towards the solution $J(\rho_{min})$, which is far from ideal. The problems linked to the use of gradient-based solvers can be mitigated by varying the initialization, but this requires multiple optimizations and can be costly to compute.

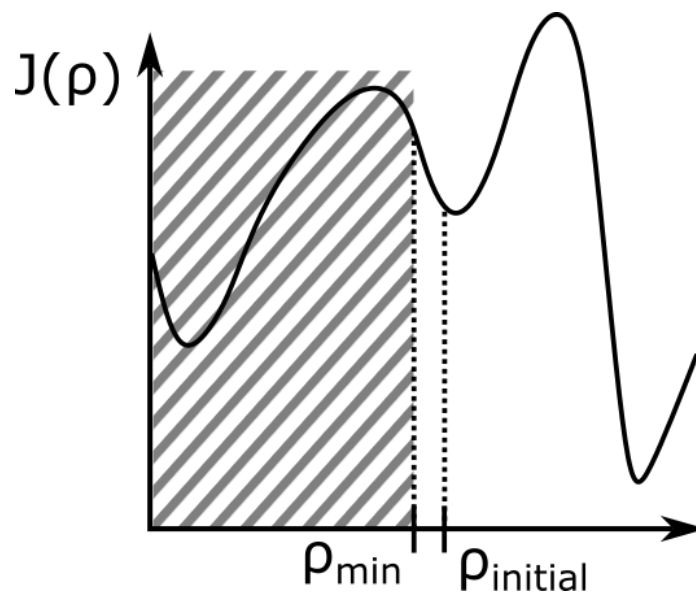


Figure 2.16: Constrained optimization

2.7.2 The MMA algorithm

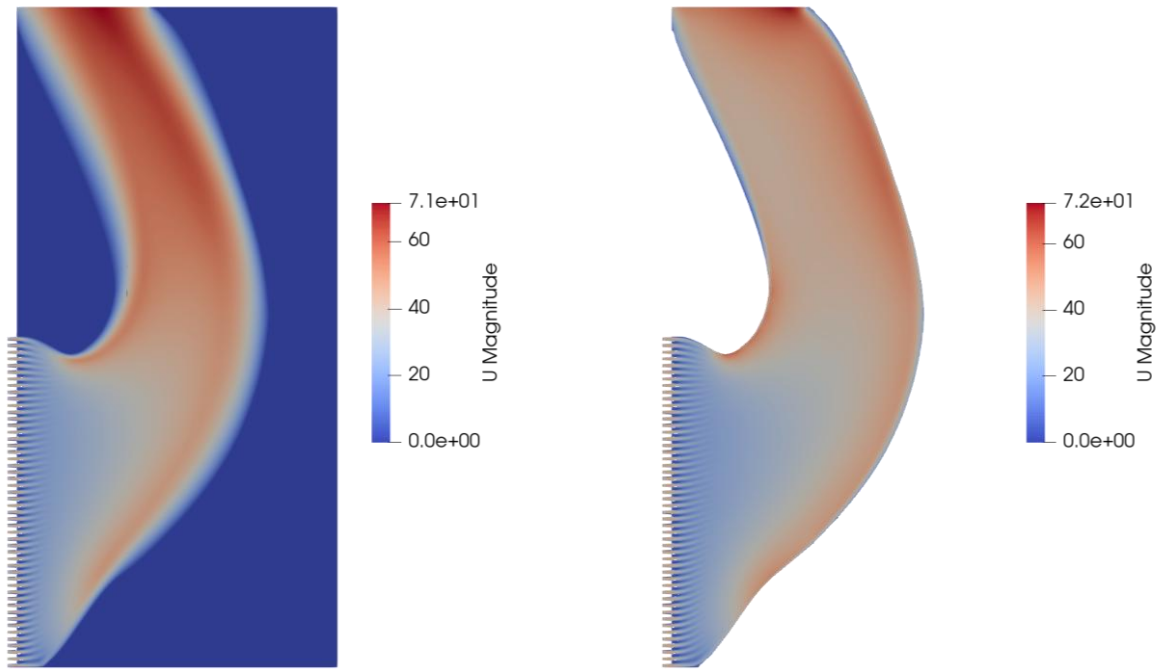
The fastest and most robust methods of finding function maxima are those based on the function's gradient. It is for this reason that the design variable relaxation was implemented in chapter 2.4. The most popular algorithm used for topology optimization is known as MMA (Method of Moving Asymptotes) and it was first proposed by (Svanberg, 1987). MMA gained popularity for topology optimization applications because of its convergence speed and stability, even for complex problems with many design variables. Furthermore, it can accept multiple objective functions with associated weights. These functions can be expressed as constraints to further guide the optimization in the desired direction.

2.8 Verification

The density-based method relies on multiple physical models, such as the Brinkmann penalized solid and design variable relaxation to determine an optimal geometry. With these applied, real physical phenomena that are inherent to turbulent flow are neglected for the sake of computation. These models have a physical basis, so they can direct the solution towards a realistic optimum, however the results of the forward calculations can often deviate from realistic values. Furthermore, the mesh used is not adapted to the geometry at every iteration. Refining the mesh to a degree where the elements are small enough within the whole domain would cause an infeasibly expensive calculation, especially considering that many optimization iterations are needed to find an optimum. Theoretically, it would be necessary to fill the entire mesh with elements that are small enough to capture the viscous boundary layer, since it cannot be predicted where the wall will be at any given iteration.

During testing, the value of the objective function (dP_t) was always large for intermediate results. However, between iterations it was observed to decrease, as minimization was the goal. To be able to determine the actual performance of an optimized geometry, a verification simulation is necessary. Verification describes the process of extracting the geometry, applying a body-fitted mesh, and running a simulation until convergence. This way, all the models applied in chapters 2.3 and 2.4 are removed. There is no more porous density field to obstruct flow, and the mesh can be adapted to the flow conditions. Notably, boundary-layer capturing mesh elements can be placed along the generated walls. At this stage, the geometry will not change anymore, so areas can be identified where mesh refinement is needed, such as areas where flow separation, or high degrees of turbulence are expected.

Extracting the geometry is done by cutting the optimization domain into two parts. By defining a threshold density value (usually $\rho = 0,5$), any elements exceeding it are considered part of the solid domain, and thus removed, keeping only the fluid domain. This boundary can then be defined as a wall, to be used for the verification simulation.



Direct solution of final iteration of optimization:
 $J_{P_t} = 1229 \text{ Pa}$

(b) Verification simulation using extracted
 geometry with body-fitted mesh $J_{P_t} =$
 422 Pa

Figure 2.17: Performance comparison between density field and extracted geometry

The difference in performance can be dramatic, depending on the geometry generated. Figure 2.17 shows almost a threefold difference between the two cases. If observing the objective function during optimization, it should have an upwards trend if maximization is the goal, and a downward trend if minimization is the goal, but no conclusion can be made about the exact performance until verification is performed. For the rest of this paper, if a performance metric for a design is given, it is to be assumed that it was from a verification simulation.

2.9 Summary

The power of topology optimization is its ability to quickly iterate through designs and reach optima. Iterating designs is an engineering method to improve performance, which has been done for decades by engineers. Topology optimization is a tool which can accelerate the design process if understood and applied correctly. Parameters such as r_{filt} , b , and α_{max} and their involvement in every step need to be well understood to obtain good results. This takes practice, just like with any other tool. However, a good understanding of the governing physics, and an intuition on how to steer the optimization towards desired results, leads to a dramatic increase in productivity. However, the learning curve is steep and can make the tool inaccessible to engineers not willing to invest time to learn.

3. SOFTWARE SOLUTIONS

In recent years, fluid topology optimization has emerged from its infancy and been implemented into commercial applications. These programs tend to make it more accessible by handling the choice of certain parameters in the background and only showing what is deemed as relevant information to the user. If done correctly, an engineer with basic CFD knowledge would be able to create topologically optimized designs quickly, without needing a long period of acclimatization with the process. If done poorly, it can lower the customizability of optimization parameters and hide calculations, so that if problems do arise, they are tedious to fix. Part of the work of this paper is to use three currently available programs that offer CFD topology optimization as a feature, and to evaluate them for their viability as an engineering tool, using the same case of the heat exchanger distribution and collection system as a benchmark.

3.1 COMSOL

The most mature program of the three is the multiphysics simulation software called COMSOL. It was released in 1998 as a FEA software for the analysis of coupled physics phenomena. Notably, it was only capable of solid structure analysis. In 2010, version 4.0 was released which implemented a CFD module for the platform. Later in 2017, topology optimization was added, including for the CFD module, making it one of the first applications of the method.

3.1.1 Topology optimization implementation

COMSOL has been a popular software solution for topology optimization for solid structures for years, as it offers an all-in-one package for coupled physics simulations. The advantage is that a topology optimized design can easily be imported into another module for analysis and the modules can even be coupled to allow for things like thermal and structural optimization at the same time. This flexibility is the main strength of this application, as transitions between physics simulations are seamless.

For fluid topology optimization, the method used is the density-based model. The implementation strikes a good balance between accessibility and customization depth. The documentation is very detailed and intuitive, and there are example cases of every module, which have step-by-step instructions on how to construct them from scratch. Following these examples, an optimization case can be set up and run without any prior knowledge. The tutorial for topology optimization guides the user through the case set up for a tesla microvalve optimization. It is a simple case, but it showcases every step of the set up well. Furthermore, the user has access to most optimization parameters and can change them if needed to adapt them for different cases. Being licensed software, the source code is not accessible, which leaves some things impossible to change, but in general COMSOL offers a user-friendly interface without compromising on customizability.

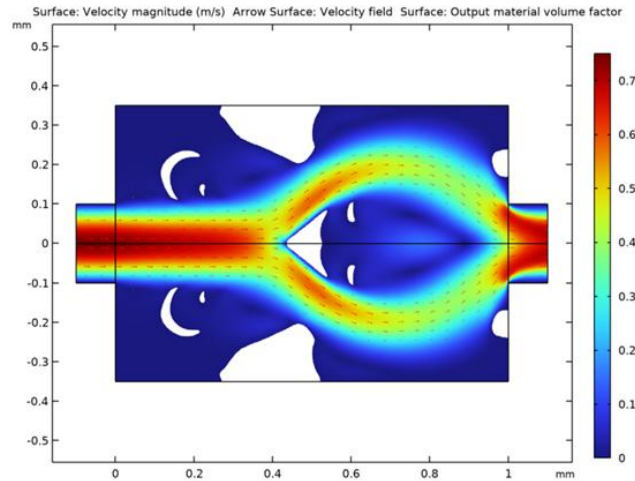


Figure 3.1: Topology optimization tutorial COMSOL – Tesla microvalve

3.1.2 Calculation time

Topology optimization is a resource intensive calculation, despite the efficiency of the adjoint method. Hundreds of simulations need to be calculated to converge on an optimum. Although the optimization itself is calculated sequentially, the forward and adjoint problems can be parallelized for significant reductions in computation time. With the computing power of a high-performance computing (HPC) cluster, designs could be generated quickly. Unfortunately, COMSOL does not support parallel computing combined with topology optimization as of version 6.1. Though low detail 2D optimization cases can be calculated in tens of minutes, the time increases with increasing mesh refinement. Furthermore, since the simulations can only be run locally, it is effectively restricted to one operation at a time.

On the other hand, COMSOL offers a streamlined workflow. Geometry creation, meshing, optimizing and verifying are all done within the same software and allow for recursive changes. Adjusting the mesh of a case after it has been run can be done in a few clicks.

3.2 OpenFOAM

The second CFD program tested is OpenFOAM. It is open-source software based on C++ and was released in 2004. Being open source, it is possible to customize every part of the software, making it a popular choice for developing new methods for CFD. For the inexperienced user, this can be disadvantageous at first, since there are no restrictions, thus being more susceptible to user error. Furthermore, the software has no graphical interface, and cases are manipulated solely through file movements and commands which are entered in a terminal. This makes the learning curve for just being able to use the program steep, let alone to perform topology optimization.

OpenFOAM was updated with an adjoint solver in 2023, which was developed by a team from the National Technical University of Athens. This solver, named *adjointOptimisationFoam*, enabled topology optimization to be performed. The solver was published with documentation found in: (Giannakoglou, Papoutsis-Kiachagias, & Galanos, 2023). Apart from this documentation, not many resources are available online, as the tool is still very new and has not seen extensive use. Some tutorials are available as part of OpenFOAM, but they lack explanation and serve more as working templates to be modified for use with different cases.

3.2.1 Topology optimization implementation

As most programs, the topology optimization method implemented is the density-based model. Every parameter can be tuned or untuned, as there are no restrictions. The only default settings available are the ones for the tutorial cases but depending on the application which needs to be optimized, these presets may be poorly suited. Therefore, it is required to have a certain knowledge of the effects of each setting to be able to start using the software effectively. However, once this knowledge is attained, OpenFOAM provides limitless customization.

A major downside of the software is its lack of good meshing utility and geometry creation software. This makes it mandatory to use other programs to create the CAD geometries of the cases, and to mesh them, for any domain that is more complicated than a box. Many file formats are accepted to be able to import geometry and mesh files into OpenFOAM, but the conditions for being able to import things like boundary names, conditions and cell zones are strict and often arbitrary. This junction between OpenFOAM and other software is yet another hurdle to overcome before OpenFOAM can be utilized to its full potential.

Despite its issues which can be overcome with enough practice, OpenFOAM is still a powerful tool for fluid topology optimization, especially because calculations can easily be parallelized. This allows users to take advantage of HPC clusters and accelerate the calculation process. This also allows calculations to be run remotely from the local machine, meaning that multiple optimizations can be run simultaneously. For this reason, most calculation results were obtained using OpenFOAM, as time was limited.

3.3 ColdStream

ColdStream is the newest software platform used in this work. It was released in 2016 by the Belgian company Diabatix. It is mainly marketed as a cloud-based topology optimization solution for the optimization of cooling components, such as heatsinks. It specializes in coupled thermal-fluid cases but can handle pure fluid cases as well. Furthermore, ColdStream was conceived with topology optimization in mind and is a specialty software for doing just that. It is marketed as a solution to save engineering time and workload by automating the entire optimization process, requiring minimal input and tuning from the engineer. Its design philosophy is opposite to that of OpenFOAM, where it is up to the user to choose every parameter to obtain results. With ColdStream, the amount of input variables is very limited. The domain geometry is uploaded to their cloud service, then objective functions can be defined by which the domain should be optimized. Additionally, a manufacturing method must be defined for the resulting design, which controls the size of the smallest structures and in which direction they will be generated, to adapt the design for the manufacturing process, such as 3D printing. Apart from these settings, nothing else can be changed.

To guarantee an optimum result, the proposed workflow according to the documentation is to start with fast concept generation, to explore possible solutions. These are correlation-based optimizations, generated using artificial intelligence, that are fast to calculate but aren't optimal. Then, the best correlation-based design should be extracted, verified, and submitted for a detailed concept generation. The detailed concept generation is where the actual topology optimization is performed, based off the initial design generated in the first step. The idea is to be able to find the correct initialization which will result in a global optimum of the objective functions when topology optimized.

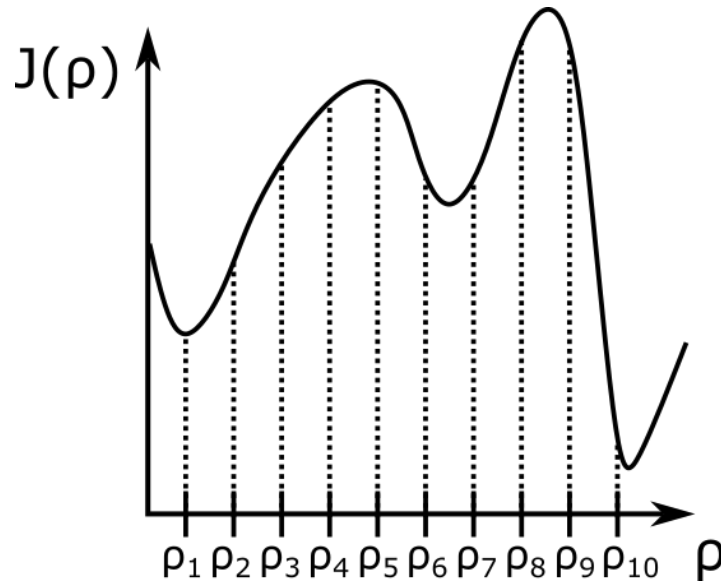


Figure 3.2: Initial design search using correlation-based designs

ColdStream’s correlation-based design approach generates many initial, rough solutions. Using the fictitious objective function graph from chapter 2.7, the interest of doing so is visualized. The initial designs are denoted by ρ_1 through ρ_{10} in Figure 3.2. From these, one design is chosen by verifying the best performing designs through a verification simulation. In this case, ρ_8 or ρ_9 . Then, this initial design is transferred to the topology optimizer, which is gradient based. If enough data points were generated, then the best performing one should be near global optimum of the objective function. Thus, optimization using that as an initialization point should lead the optimization algorithm to converge at the maximum.

3.3.1 Topology optimization implementation

Though the concept of finding a global maximum using fast, correlation-based designs is promising, its execution in ColdStream is highly automated and allows little human intervention while calculating. During testing, the example collector case as described in chapter 2.1 was optimized. For topology optimization, there are three degrees of detail that can be chosen, which define the mesh refinement. These are: “draft” (10^5 mesh elements), “conceptual” (10^6 mesh elements), and “detailed” (10^7 mesh elements). Then, a manufacturing method must be chosen, which will constrain the geometry to make it manufacturable using the defined method. Density-based topology optimization is performed, but the details as to optimization algorithm, regularization method, or penalization interpolation, etc. are unknown and chosen automatically in the backend.

The ColdStream license provided has an allowance of 200 credits, which are advertised to be equivalent to processor cores for computation. These 200 credits can be distributed among calculations as seen fit. If all 200 credits are allocated to a single optimization using the “detailed” preset, the calculation finishes in 2 weeks, provided nothing goes wrong during calculation. Using the “draft” preset cuts this time in half but only allows for very macroscopic geometries to be generated with little detail.

3.4 Summary

Between the three programs tested, each one can be defined by a strive to be intuitive and easy to use, and the amount of customization of the optimization allowed. Generally, these two aspects share a linear relationship. OpenFOAM offers full control, but it has a steep learning curve which must first be overcome. Diabatix’s ColdStream is the opposite, allowing very little control,

thus avoiding confusion and making it a very approachable program. COMSOL lies in between. It is an advanced software package with extensive customizability, while also providing good documentation and a well-designed graphical interface.

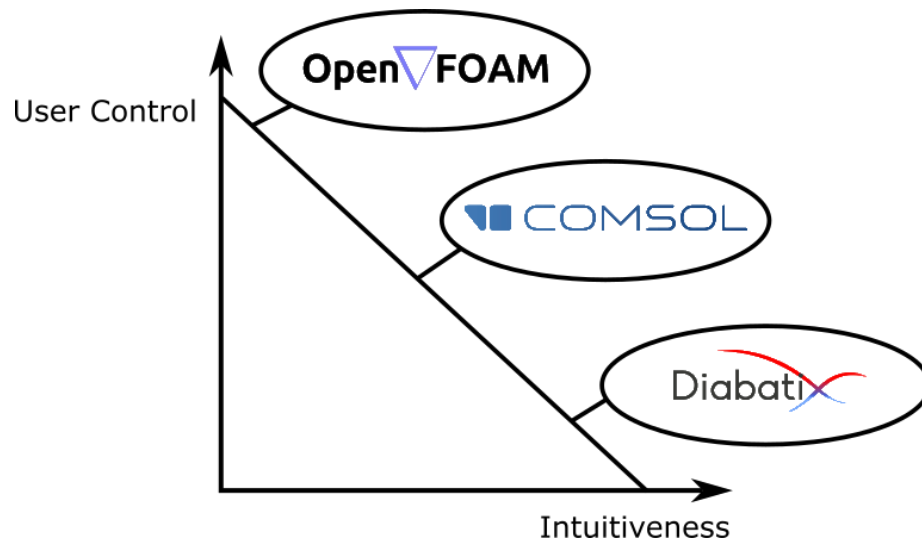


Figure 3.3: Qualitative comparison of optimization programs used

4. CASE PRESENTATION

Using the three programs presented, one optimization domain will be used between them, so that the quality of each optimization can be compared. This domain is the same as presented in Figure 2.1 and represents a fictitious collector which is used to collect the flow coming out of the heat exchanger and to guide it towards an exit. The performance of the collector is measured by the total pressure drop between the inlet and the outlet and should be minimized. The optimization bounds and boundary conditions are presented in Figure 4.1.

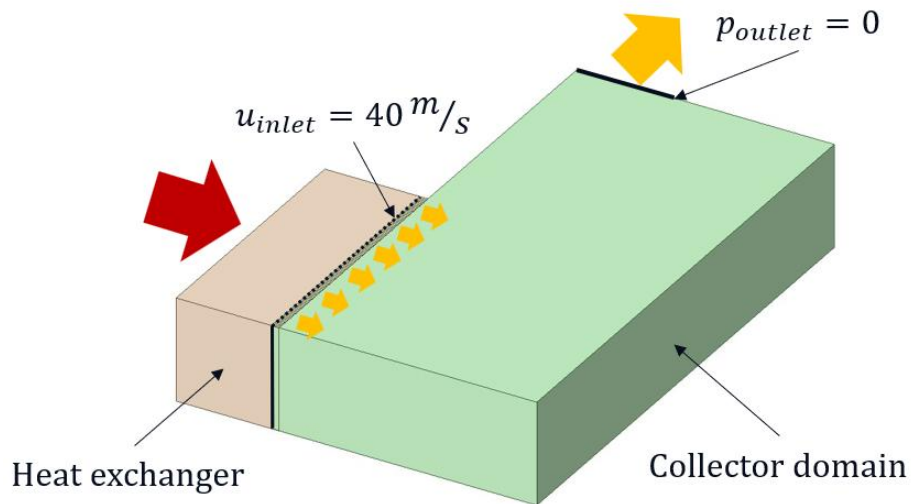


Figure 4.1: Optimization case with boundaries

Topology optimization is a computationally intensive process, since hundreds of iterations can be necessary to obtain a satisfactory result, and each iteration consists of multiple simulations. Therefore, this case was reduced to two dimensions, assuming that the geometry can later be extruded along the surface normal to obtain a valid three-dimensional geometry.

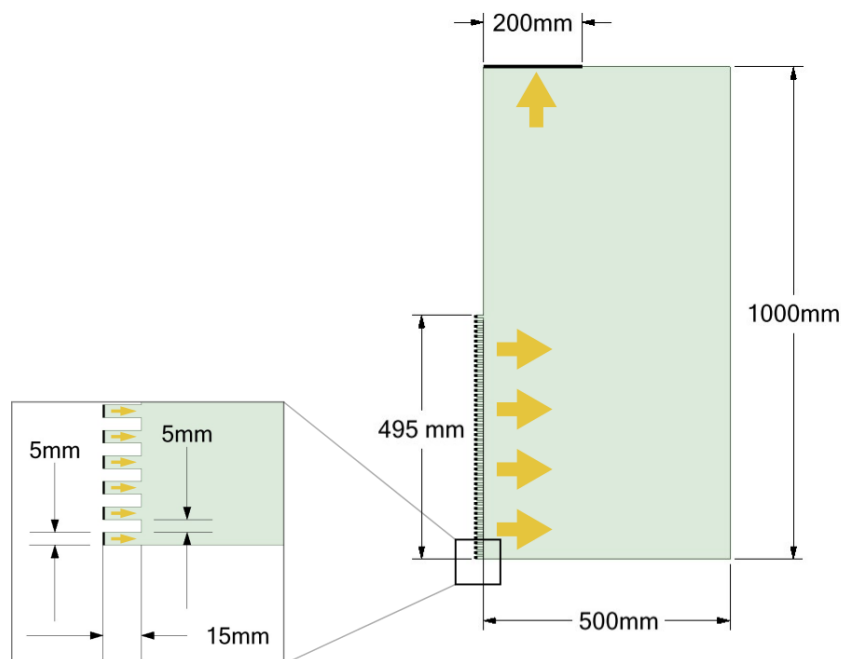


Figure 4.2: Two-dimensional domain

The heat exchanger consists of many small channels through which the exhaust passes and transfers its heat. To model this, the inlet of the domain is discretized and set back. There are 50 channels in total, with a width of 5mm each. The domain itself is large and represents absolute design limits which should not be exceeded. However, this is meant for an aerospace application, so generated designs that are smaller and bend the flow at a sharper angle will be preferred. This is a secondary consideration though; the primary objective is to reduce pressure losses.

4.1.1 Flow Properties

This section is used to showcase the material properties that were defined for the flow. These values are fictitious and do not reflect the flow properties that are present in the real collector which was optimized for SAFRAN.

Name	Symbol	Value
Temperature	T	200°C
Density	ρ	$0,749\text{ kg}/\text{m}^3$
Kinematic Viscosity	ν	$3,455 \cdot 10^{-5}\text{ m}^2/\text{s}$
Reynolds Number	Re	1 157 742

Table 1: Properties of air used for optimization

Given these values it can be determined that the flow is turbulent, so a turbulence model must be defined. The Spalart-Allmaras turbulence model is chosen, because its adjoint equation is implemented in all three programs used and it is the most mature turbulence model for topology optimization. The model consists of a single transport equation, making it favorable for optimization as only one adjoint equation needs to be derived to implement it.

5. RESULTS

Before beginning the process of topology optimization, there are parameters whose values must be set to ensure that the optimization produces good results. Notably, α_{max} must be set high enough to ensure that the penalized material placed during the optimization sufficiently blocks flow. This factor is highly dependent on the flow velocity present in the field, as a slow laminar flow with speeds below $U = 0,01 \text{ m/s}$ will not need as high of a penalization as turbulent flows at speeds of $U = 100 \text{ m/s}$. Likewise, the penalty cannot be set arbitrarily high, as this can destabilize the optimization. For this case, the effect of α_{max} is already investigated in Figure 2.7, where it is shown that a value of $\alpha_{max} = 10^5$ is sufficient to block flow. However, tests show that setting $\alpha_{max} = 10^6$ does not destabilize the algorithm and helps block flow through even the smallest obstacles, so this is the value used to obtain the results presented. The regularization parameters r_{filt} and b vary, according to the goal of the optimization; whether small structures should be generated or not.

5.1 COMSOL

The results obtained from using COMSOL with the collector case shown in Figure 4.2 prove that it is not a suitable program. Several issues were encountered with the software, and it was not possible to obtain a coherent result. As mentioned in chapter 3.1, parallelization of optimization is not possible in COMSOL. This makes it tedious to use refined meshes, as the calculation can take a long time. Furthermore, it was observed that the optimization becomes unstable with an increasing amount of mesh elements, especially with a turbulence model applied. It is suspected that the implementation of the adjoint solver in COMSOL, known as the discrete adjoint method, causes these issues. This method needs a large amount of RAM to calculate the adjoint. The machines used to run COMSOL did not have enough memory to facilitate COMSOL's solver. This could have been remedied using the HPC cluster, but the batch calculation is a part of the parallelization module, which is blocked for topology optimization. Due to this, only cases with a moderate number of mesh elements could be computed.

During testing, another quirk of COMSOL was discovered which further hindered its use. As seen in Figure 4.1, the outlet of the domain is defined with the boundary condition: $p = 0 \text{ Pa}$. When applying a static pressure boundary condition such as this one in COMSOL, it is defined by equation (5.1). Instead of a simple pressure constraint, the condition is expanded to include the turbulent kinetic energy k multiplied by a unity vector, which is normal to the boundary.

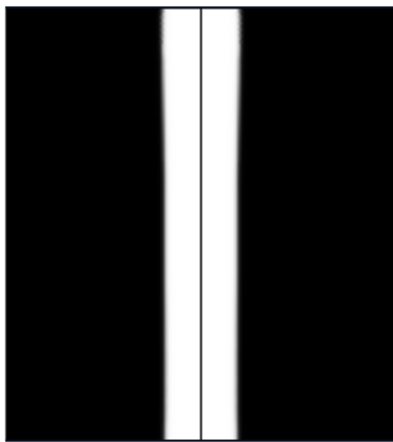
$$k \cdot n - p = 0 \quad (5.1)$$

This boundary condition is valid for normal simulations; however, the optimization algorithm can exploit this equation when it needs to optimize the pressure drop between inlet and outlet. By generating a geometry, which can induce turbulence at the outlet, optimization can effectively drive the static pressure up, due to the definition of the boundary condition. To verify that this is the case, a test is set up in COMSOL of a simple symmetric canal, the setup is shown in Figure 5.1.

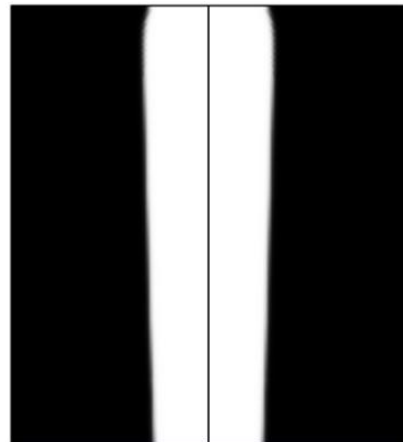


Figure 5.1: COMSOL test case of simple canal

As with the collector, the optimization was set to reduce the pressure loss. This case was optimized once with a laminar flow at $Re = 10^3$ and once with a turbulent flow at $Re = 10^6$ with a turbulence model applied. The outlet and inlet had the same boundary conditions applied as used for the collector. The optimized geometries were then extracted, and a verification simulation was done with turbulent flow in both geometries.



(a) Case optimized with laminar flow



(b) Case optimized with turbulent flow

Figure 5.2: Generated geometries of the canal test case

The optimized geometry of the laminar case provides the expected result of a straight canal. However, in Figure 5.2 (b), the optimization algorithm increased the cross section of the canal along its length, only to quickly decrease it at the outlet. This is done to increase turbulence. After extracting both geometries and simulating both with turbulent flow, the static pressure along the outlet boundary was measured for both geometries.

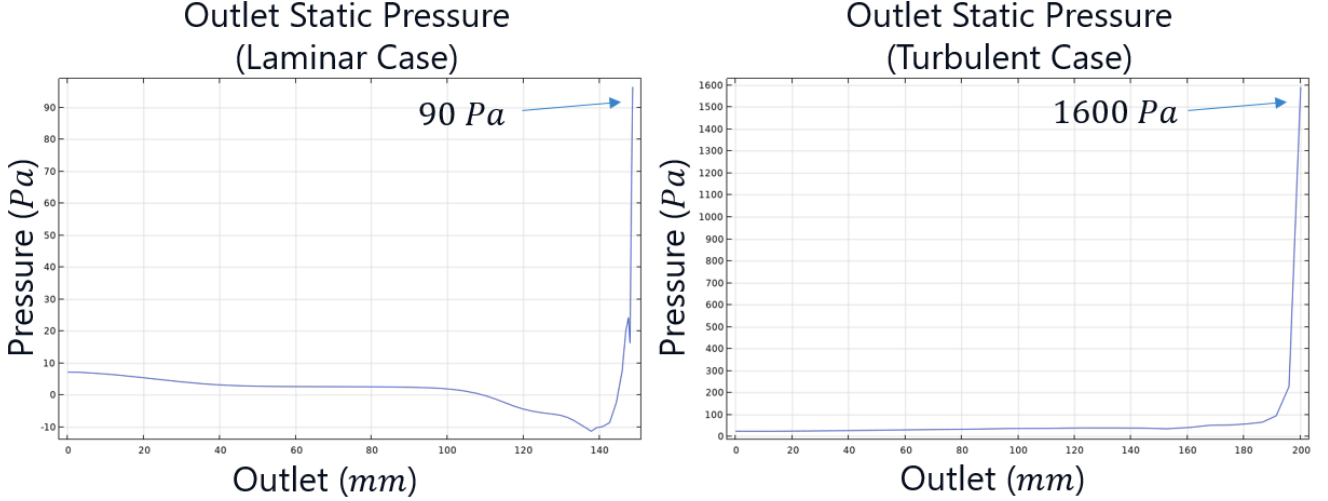


Figure 5.3: Static pressure curves over the outlets of the test cases

Being a turbulent simulation, and given the definition of the outlet boundary condition, both cases have a non-zero static pressure across the outlet. However, the effect is dramatic with the turbulent case, which exhibits a peak of 1600 Pa. This effect is drastic enough to cause the turbulent case to exhibit a negative pressure loss between inlet and outlet. Because of this boundary definition, which cannot be changed within COMSOL, it was not used for any optimizations of the collector.

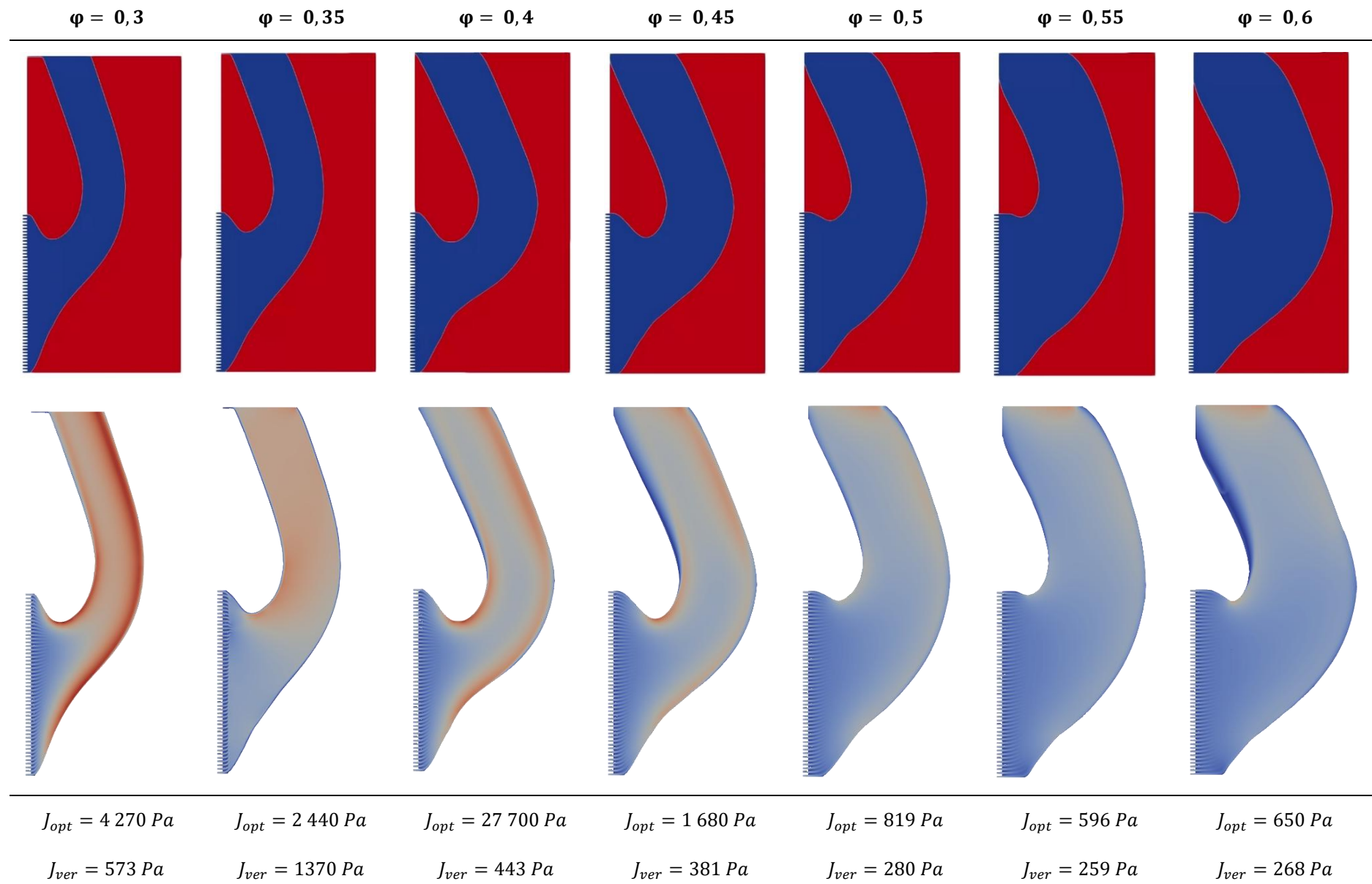
5.2 OpenFOAM

The boundary conditions in OpenFOAM are implemented in a more straightforward manner, and the outlet pressure is kept at the value defined. It also supports cluster computing, which greatly accelerates calculations.

The optimization criteria for the collector are not well defined, apart from the goal of minimizing pressure drop while avoiding a large geometry. These criteria are expressed by the objective function $J = dP_t$ and the volume constraint φ , however no hard limits are given for either one. Therefore, a systematic optimization approach is taken, whereby multiple optimizations are run, while varying the volume constraint. This way, an optimum design can be chosen which fulfills both criteria the best.

5.2.1 Coarse Optimization

To find the best flow envelope, which will determine the size of the collector, a series of coarse optimizations are performed. These are meant to give insight into how the volume constraint affects the outer form and the pressure loss. Seven optimizations are presented, with volume constraints ranging from $\varphi = 0,3$ to $\varphi = 0,6$. These are done with a uniform mesh that is fine enough to resolve macroscopic turbulent effects. The filter radius is set to $r_{filt} = 1cm$ to suppress any fine structures, as these optimizations are only meant to give the outer contour. The projection parameter is set to $b = 20$, which was found to be sufficient to eliminate most zones with intermediate densities. These designs were then extracted and verified with CFD simulations to extract their performance metrics. The results are compiled in Table 2. The top row shows the density field from the last iteration of the optimization, where red is the solid domain and blue the fluid domain. Below are the velocity fields, which were calculated by simulating the extracted geometry from each optimization. The pressure loss for both cases is given at the bottom.

**Table 2:** Results of coarse geometry optimization

The results from the series of coarse optimizations show a steady decline in pressure loss with a growing volume constraint. There are three anomalies within the results which do not support this hypothesis. The first is the pressure loss of the verified geometry with the volume constraint $\varphi = 0,35$. This is due to an instability in the verification simulation, which made the simulation difficult to converge to a solution. Every other verification simulation successfully converged.

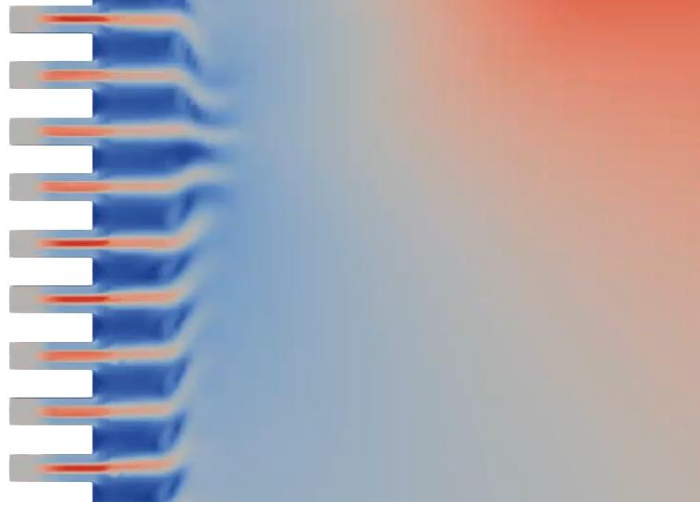


Figure 5.4: Non-converged flow in verification of $\varphi = 0,35$ case

Another anomaly is the objective function value J_{opt} for the case with $\varphi = 0,4$. The value is abnormally high. This is explained by an anomaly in the way OpenFOAM weighted the objective function which caused it to be elevated. Though it is high, the objective function was minimized by the optimization algorithm relative to the objective function values of other iterations. Nonetheless, the generated geometry is coherent with expectations, and the verified objective function follows the downward trend.

Finally, the optimization using a volume constraint of $\varphi = 0,6$ creates more pressure loss than its preceding design. This is a problem with the gradient-based optimization algorithm. The optimization converged on a local minimum of the objective function which did not have a good performance value. The volume constraint is only an upper limit, meaning that a volume constraint of $\varphi = 0,6$ should allow a design with a volumetric fraction of $\varphi = 0,55$. Therefore, if the optimization algorithm was globally convergent, it would find a solution that has at most the same pressure drop as the preceding optimization. This is not the case; therefore, it must have found a false optimum.

Table 2 showcases the necessity of verification simulations when using density-based topology optimization. The value of the objective function is greater than the value of the verified one by a factor of ~ 60 in the case of the optimization with $\varphi = 0,4$. The objective function between iterations only has relative meaning and is minimized as such.

The best result from the first series of optimizations is the one with the volumetric fraction $\varphi = 0,5$, because the designs after it have diminishing gains in pressure loss reduction. It strikes a good balance between pressure loss and encumbrance, so it is chosen to be optimized further, with a more refined optimization. Inspecting the velocity field near the inlet canals, recirculation zones can be identified in the wakes between canals. These are identified as a source of pressure loss and should be accounted for in the refinement optimization run. The distance between the canals is $5mm$, and the filter radius r_{filt} of the coarse optimizations was set to $1cm$. Therefore, any structures

in this area that were generated during the first round of optimizations were suppressed by the regularization filter.

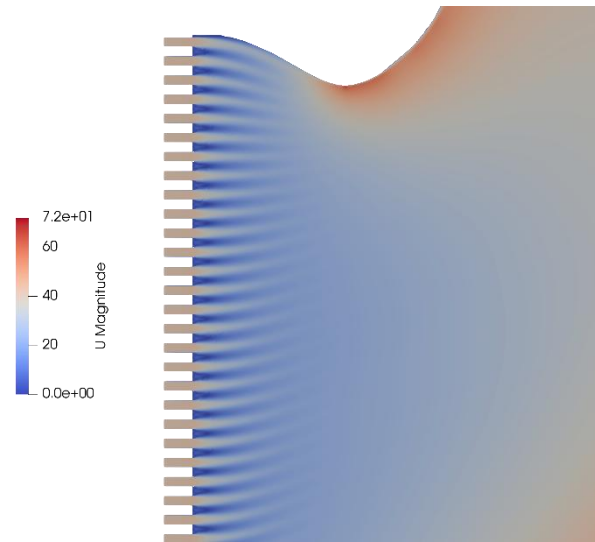


Figure 5.5: Recirculation zones behind inlet canals

The refinement optimization is adapted to target the inlet area. The mesh is refined to an element size of $0,05mm$ to generate smooth structures at the necessary scale. Furthermore, there is no volume constraint set, and regularization is turned off. This allows the optimizer to place material freely, without being forced to place it by the constraint.

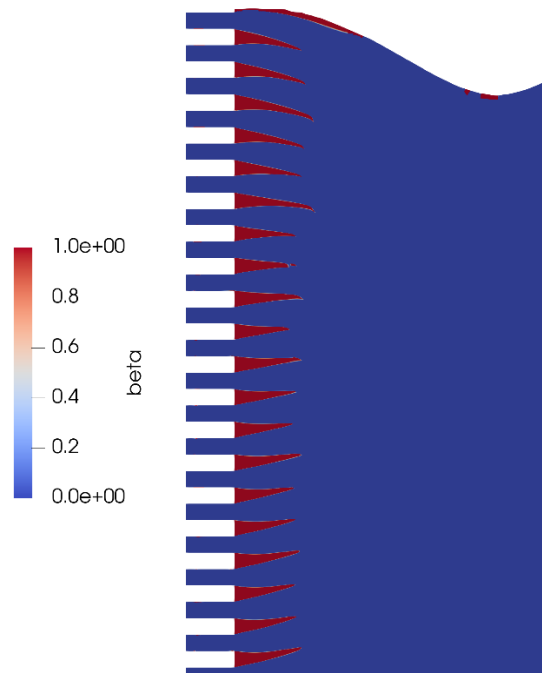


Figure 5.6: Density field generated by the refinement optimization

Comparing the shapes of the recirculation zones seen in Figure 5.5 and the structures in Figure 5.6, they seem to match. The optimization algorithm correctly identifies the zones that create losses and fills them, as expected. The verification of the refined design confirms that it performs better than the design with just the flow envelope.

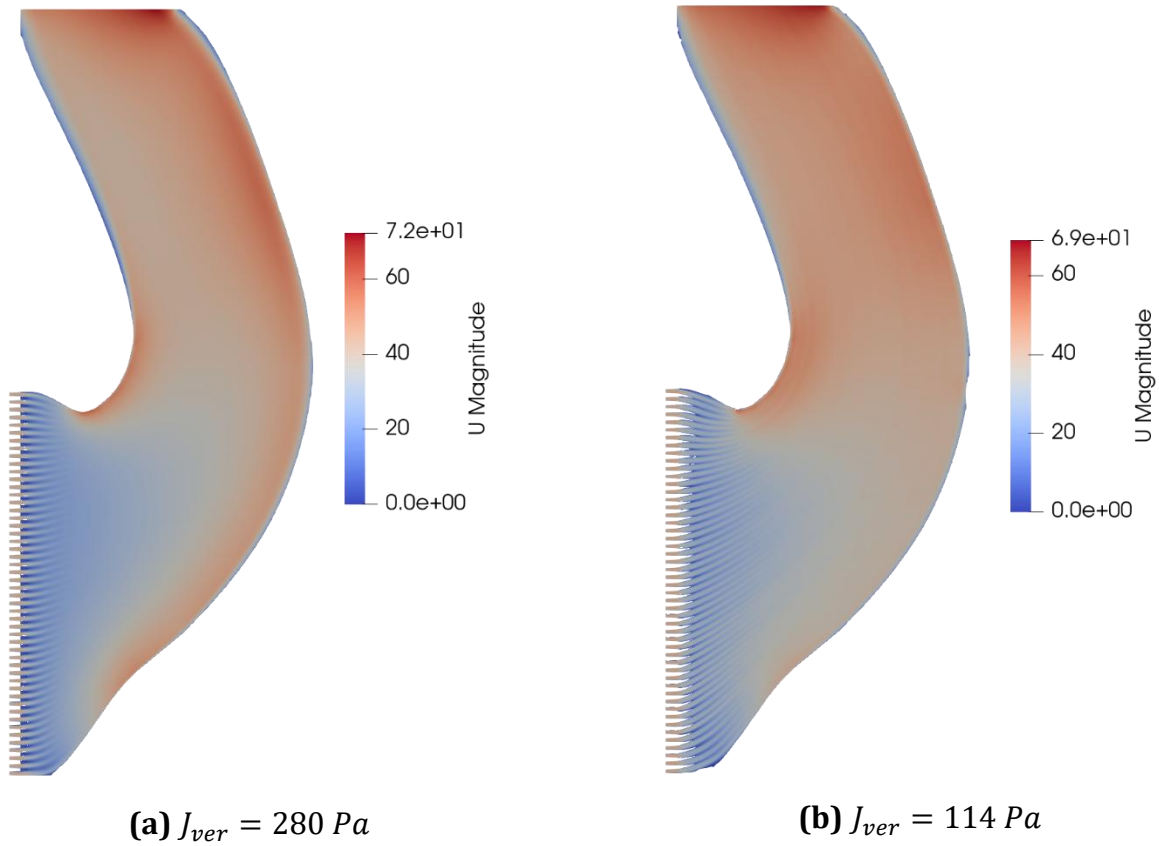


Figure 5.7: Comparison of coarse design (left) and refined design (right)

The pressure loss reduction gained by performing a second optimization is significant and shows the value of doing so, since the pressure drop is $\sim 60\%$ lower. Looking closer at the velocity field near the inlet, it becomes apparent that the spike-like structures effectively fill and mitigate the amount of recirculation at the ends of the canals from the heat exchanger and therefore reduce pressure losses

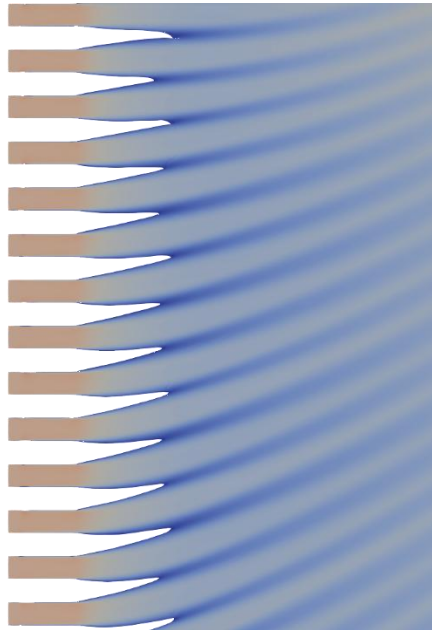


Figure 5.8: Close view of inlet area of refined design

5.3 ColdStream optimization

The license for ColdStream was acquired during the final two months of work for this thesis, which meant that there was little time for testing. This is exacerbated by ColdStream's calculation times, which can be very long, even when dedicating all 200 credits towards a single job. For this reason, the suggested ColdStream workflow is not followed. Only the topology optimization tool is used, without using the correlation-based designs as an initialization. This also allows a more direct comparison between just the topology optimization implementations between OpenFOAM and ColdStream.

ColdStream's levels of detail for a given optimization case vary by the amount of mesh cells, thus the calculation time is irrespective of the geometry. The interest of applying calculation saving methods such as using a two-dimensional domain or using a plane of symmetry to cut the domain in half instead improve the mesh refinement. Using all 200 credits, the optimization times for all three levels of detail are listed in **Table 3**. These times are estimates and can differ from the actual computation time. If the optimizer converges on a solution early, it will stop. However, if a problem arises that the automated program cannot handle, manual intervention is necessary by the team at ColdStream, which can take multiple days to resolve. The time needed to optimize the two-dimensional collector case was three weeks, as some meshing issues were encountered, and 180 credits were dedicated, allowing for other small calculations in parallel.

Detailed Case	15 days
Conceptual Case	10 days
Draft Case	8 days

Table 3: Computation times for different levels of detail with ColdStream

Since the computation time with ColdStream is so long, a series of optimizations to find the best volume constraint for the collector is infeasible. Even by submitting them as draft cases, the seven optimizations would take almost two months to complete, assuming no interruptions in the calculations. This is a big disadvantage for ColdStream, since the volume constraint is only a vague measure of the encumbrance and mass of the design. It is more of an optimization parameter that needs to be determined to yield the desired results. Using the data gained from OpenFOAM, the volume constraint is set to $\varphi = 0,5$ for the ColdStream optimization.

The design generated by ColdStream is generated from a detailed case of the two-dimensional collector domain with the objective function of pressure loss reduction between the inlet and the outlet. ColdStream natively does not support two-dimensional simulations, so the domain is submitted as a very thin plane, to force the mesh to have a thickness of one cell. The only other setting that can be adjusted is that of the manufacturing method, which is an indirect way of adjusting the filter radius r_{filt} . This was chosen to be as fine as permissible by the program, which allows for structures that are as thin as $1mm$. The space between the inlet canals of the domain is $5mm$ wide, so this method should be able to generate structures there.



Figure 5.9: Design generated by ColdStream

The result of the detailed optimization from ColdStream vaguely resembles that of the one from OpenFOAM, at least in the general form of the flow envelope. There are two anomalous regions above and below the inlet area, which have questionable efficacy for pressure loss minimization. Nevertheless, there are small structures at the inlet canals, which take the shape of small plates extending from the wall. Clearly, the mesh and filter radius set by ColdStream are fine enough to permit structures at that scale, but their form is not satisfactory when compared to the result from OpenFOAM.



Figure 5.10: ColdStream's inlet structures

The verification calculation is also done with ColdStream, as it provides an integrated module to perform this. The verification is just one simulation, with a body-fitted mesh of the design generated. Doing this manually should only take some hours, but using ColdStream, it takes three days. Nothing needed to be done besides simply giving the command and waiting.

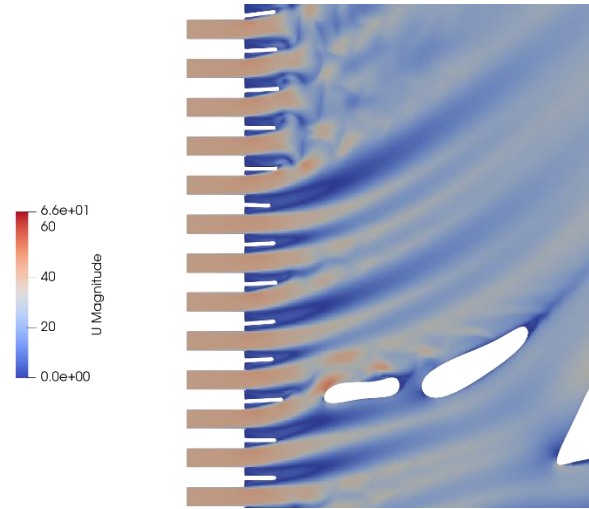


Figure 5.11: Close-up of velocity field of ColdStream verification simulation

The result of the verification is disappointing, as the calculation is not converged. This is evident due to the instabilities seen in the flow field near the inlet as shown in Figure 5.11. Since the simulation is not converged the result for the performance of the design may be inaccurate. Nevertheless, the ColdStream design is not optimal when evaluated by engineering intuition, especially compared to the design generated by OpenFOAM. Even if the pressure loss value calculated is not accurate, it is difficult to imagine that the real performance is better than the that of the design from OpenFOAM.

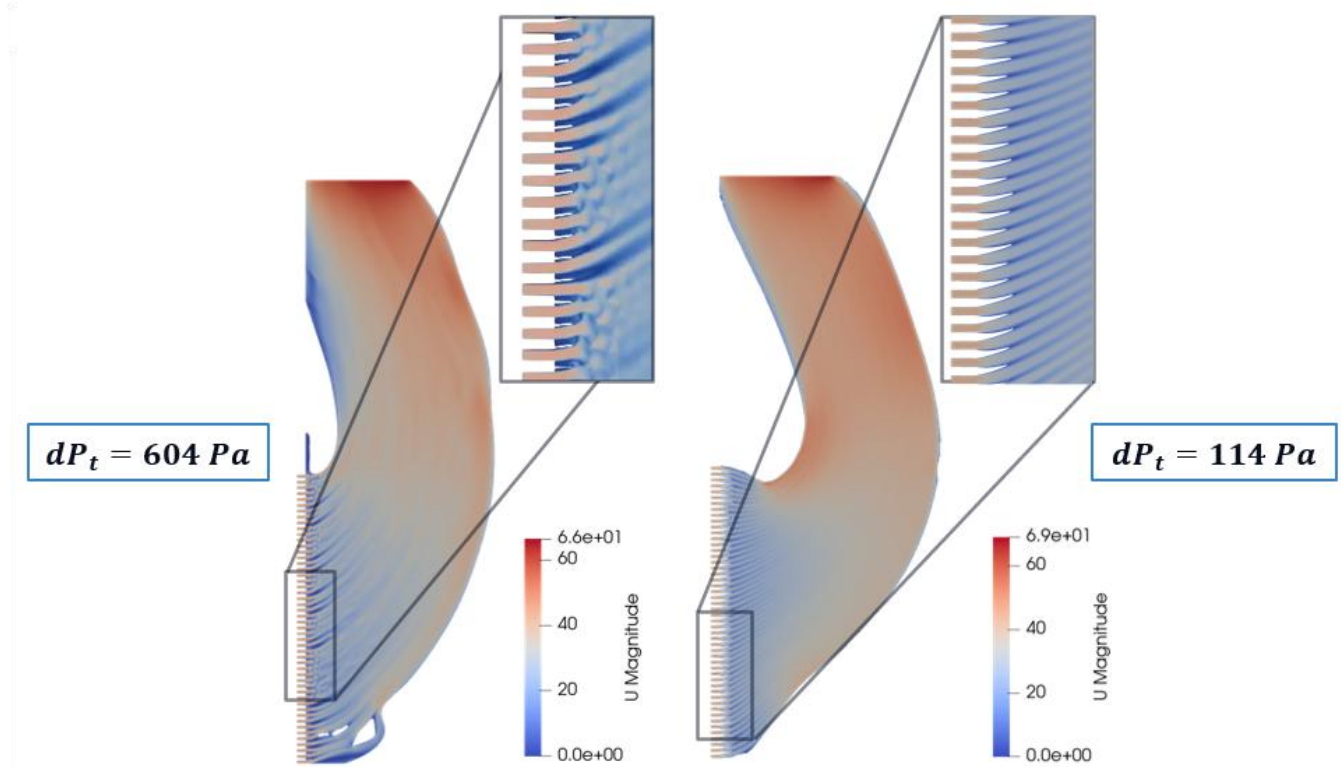


Figure 5.12: Verification simulations of ColdStream (left) OpenFOAM (right)

Due to the design philosophy of ColdStream, there are no settings that could be adjusted as a user to steer the optimization in a more favorable decision, apart from using the correlation-based design feature to initialize the geometry. However, the calculation time required to complete the entire workflow is long, even though the amount of work needed to be done is minimal. Furthermore, it is not certain that initializing the design would result in a far better optimization result, since it seems that the problem lies primarily with the non-convergence of the forward problem. The verification simulation is often done with more iterations and to ensure a converged and accurate result, whereas the forward problem during optimization is solved with more speed and stability in mind. Therefore, if the verification simulation is unstable, then the forward simulation likely had the same issues. This means that the sensitivity is calculated based off imprecise data, which causes the optimizing algorithm to place material in the wrong places.

Using OpenFOAM, a stability problem like the one observed with ColdStream could be alleviated by tuning the under-relaxation factors or the discretization schemes of the solver, but this is impossible in ColdStream. Furthermore, debugging a problem such as this requires many simulations, to see how changes affect the results. This way the problem can be iteratively stabilized. Even if it were possible to tune the simulation parameters of ColdStream, the debugging would take months of waiting.

6. CONCLUSION

Topology optimization is a powerful tool which has already seen widespread adoption and success for linear problems, such as for structural engineering. The application of the method for fluid problems comes with the challenge of the inherent non-linearity of the governing equations and the instability of the solution. Despite this, the density-based method presented in this thesis has been applied successfully and is beginning to be implemented into commercial CFD programs. The testing of these programs reveals the low maturity level of the method and the difficulties in its implementation and use.

COMSOL has the potential to be a good framework for topology optimization, but the implementation needs work to be able to be applied for turbulent flows. As shown in the results, linear flow optimization is possible with COMSOL, and a simple change to the pressure outlet boundary condition would make it applicable for turbulent flows as well. Furthermore, the implementation of parallelization and a more stable adjoint solver would fix all the issues that COMSOL currently faces. As a dedicated topology optimization software, ColdStream is disappointing. It is meant to be easy to use, but this makes the program over-simplified for such a complicated method as topology optimization. In theory, if the design generated by ColdStream were as good as the design from OpenFOAM, it would be an impressive result given that very little effort was needed. If work is done to stabilize the calculation it could be a viable tool. Work also needs to be done to lower the calculation time, because even though much manual work is not needed, waiting multiple weeks for a result which may not even be interesting is sub-optimal. Finally, OpenFOAM proves to be deliver good results, though it takes time to learn. Once the software architecture and topology optimization as a method is understood, OpenFOAM is a powerful tool which leaves unlimited control to the user.

With time, fluid topology optimization could reach the same maturity level as it already has for structures. Currently, it is not mature enough to be reliably used for the generation of designs, especially given the limitations of gradient-based optimization. Furthermore, the fact that the performance of the design during density-based optimization is not realistic and only used as a comparison to other iterations, makes it difficult to define real objectives that should be reached. Methods like level-set topology optimization aim to overcome this issue but are at an even lower maturity level. However, the utility of topology optimization to generate design ideas cannot be overstated. Even if a design has some anomalies, the algorithm that generated them is based on physical models. With a good understanding of the physics and the methodology, design trends can be extracted from the algorithm and refined using other tools.

BIBLIOGRAPHY

- Alexandersen, J. (2023). A detailed introduction to density-based topology optimisation of fluid flow problems with. *Structural and Multidisciplinary Optimization*.
- Allaire, G. J. (2002). A level-set method for shape optimization. *Comptes rendus. Mathématique*, 1124-1130.
- Bendoe, M. P., & Sigmund, O. (2004). *Topology Optimization*. Springer.
- Bendsoe, M. P., & Kikuchi, N. (1988). Generating optimal topologies in structural design using a homogenization method. *Computer Methods in Applied Mechanics and Engineering*, 197-224.
- Borrvall, T., & Petersson, J. (2002). Topology optimization of fluids in Stokes flow. *Numerical Methods in Fluids*, 77-107.
- Challis, V. J. (2009). A discrete level-set topology optimization code written in MATLAB. *Structural and Multidisciplinary Optimization*, 453-464.
- COMSOL. (2025, 02 24). Retrieved from Release History: <https://www.comsol.com/release-history>
- Deaton, J. D., & Grandhi, V. R. (2013). A survey of structural and multidisciplinary continuum topology optimization: post 2000. *Structural and Multidisciplinary Optimization*, 1-38.
- Evgrafov, A. (2005). The Limits of Porous Materials in the Topology Optimization of Stokes Flows. *Applied Mathematics and Optimization*, 263-277.
- Gersborg-Hansen, A., Sigmund, O., & Haber, R. B. (2005). Topology optimization of channel flow problems. *Structural and Multidisciplinary Optimization*, 181-192.
- Giannakoglou, K., Papoutsis-Kiachagias, E. M., & Galanos, N. (2023). *adjointOptimisationFoam, an OpenFOAM-based optimisation tool*. Athens: National Technical University of Athens.
- Giles, M. B., & Pierce, N. A. (2000). An Introduction to the Adjoint Approach to Design. *Flow, Turbulence, and Combustion*, 393-415.
- Guest, J. K., Prévost, J. H., & Belytschko, T. (2004). Achieving minimum length scale in topology optimization using nodal design variables and projection functions. *International Journal for Numerical Methods in Engineering*, 238-254.
- Holka, Q. (2022). *Optimisation topologique des échangeurs de chaleur en*. Paris: Mines Paris-PSL.
- Jameson, A., Schmidt, W., & Turkel, E. (1988). Numerical Solutions of Euler Equations by the Adjoint Method. *AIAA Journal*, 431-439.
- Kontoleonos, E., Papoutsis-Kiachagias, E., Zymaris, A., Papadimitriou, D., & Giannakoglou, K. (2013). Adjoint-based constrained topology optimization for viscous flows, including heat transfer. *Engineering Optimization*, 941-961.
- Mandler, H., & Weigand, B. (2022). On frozen-RANS approaches in data-driven turbulence modeling: Practical relevance of turbulent scale consistency during closure inference and application. *International journal of Heat and Fluid Flow*.
- Nielsen, E. J., Lu, J., Park, M. A., & Darmofal, D. L. (2004). An implicit, exact dual adjoint solution method for turbulent flows on unstructured grids. *Computers & Fluids* 33, 1131-1155.
- Sigmund, O., & Diaz, A. (1995). Checkerboard patterns in layout optimization. *Structural Optimization*, 40-45.

- Sigmund, O., & Petersson, J. (1998). Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima. *Structural optimization*, 68-75.
- Svanberg, K. (1987). The method of moving asymptotes - a new method for structural optimization. *Internation Journal for Numerical Methods in Engineering*, 359-373.
- Verdijck, N. (2024). ColdStream workflows. Leuven, Belgium.
- Wang, M. Y., Wang, X., & Guo, D. (2003). A level set method for structural topology optimization. *Computer Methods in Applied Mechanics and Engineering*, 227-246.
- Xu, S., Cai, Y., & Cheng, G. (2010). Volume preserving nonlinear density filter based on heaviside functions. *Structural and Multidisciplinary Optimization*, 495-505.
- Zillober, C. (1993). A globally convergent version of the method of moving asymptotes. *Structural and Multidisciplinary Optimization*, 166-174.