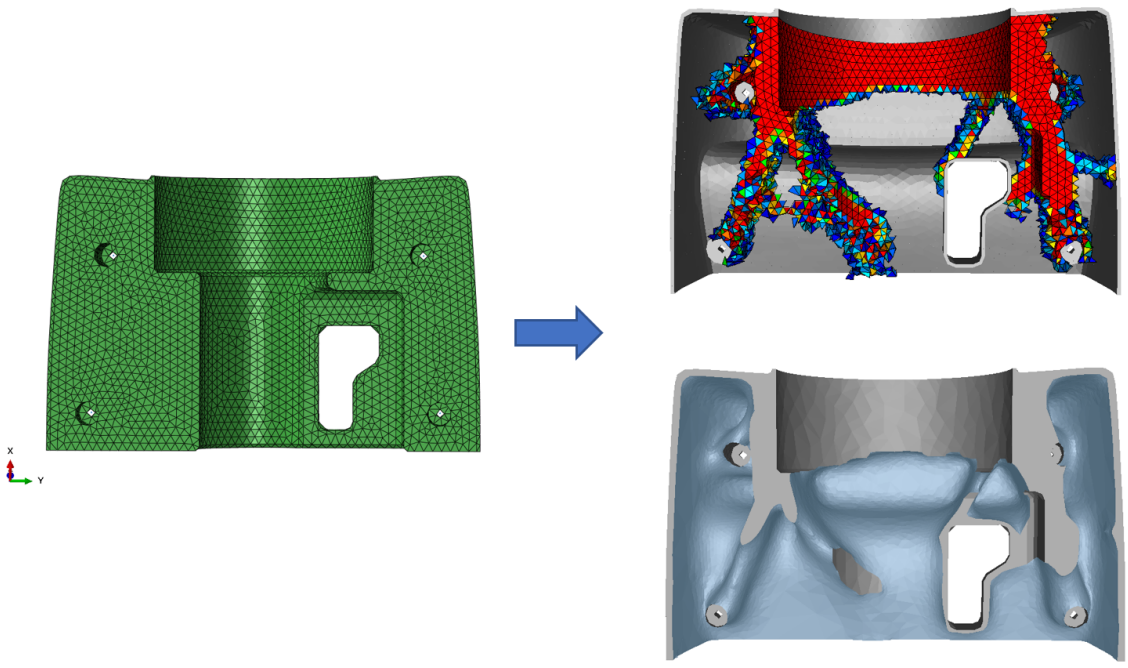
Baustatik und Baudynamik

Master Thesis

Improving demolding manufacturing constraints for gradient-based topology optimization within SIMULIA Tosca Structure

Arvinth Shankar



submitted at the

University of Stuttgart
Germany

Institute for Structural Mechanics
Prof. Dr.-Ing. habil. Manfred Bischoff

Improving demolding manufacturing constraints for gradient-based topology optimization within SIMULIA Tosca Structure

by

Arvinth Shankar

edited from

November 2023 to April 2024
(Date of Submission: 30.04.2024)

in the study program

Computational Mechanics of Materials and Structures, COMMAS (M.Sc.)

under the supervision of

Lisa-Marie Krauß, Thorsten Mertins, Peter Clausen

Declaration

- I hereby declare that I have independently written the thesis presented here.
- Only the sources and aids mentioned explicitly in the thesis have been used. I have marked as such any ideas taken over verbatim or in spirit.
- The submitted thesis was not and is not the subject of any other examination procedure, neither in its entirety nor in substantial parts.
- Likewise, I have not already published the work, either in full or in part.
- I certify that the electronic copy matches the other copies.

Stuttgart, April 30, 2024

(*Signature Student*)

Universität Stuttgart

Baustatik und Baudynamik

Master Thesis

Improving demolding manufacturing constraints for gradient-based topology optimization within SIMULIA Tosca Structure

Structural optimization accelerates the design process and saves resources compared to the traditional trial and error methods in the structural engineering field. The optimization system *SIMULIA Tosca Structure* integrates structural optimization technologies in industrial engineering environments for example *SIMULIA Abaqus*. Thereby, the structural modeling is utilized using the finite element method. In this manner, topology, sizing, shape and bead optimizations can be performed for the finite elements for fulfilling constraints and to optimize the objective function. In topology optimization the relative densities of elements are changed in order to find the optimal material distribution for the structure. The thicknesses of shell elements are considered as design variables in sizing optimization. And the optimal positions of surface FE-nodes are determined in shape and bead optimization, respectively. Design responses like mass, volume, displacements, stresses, reaction forces, energies, stiffness and eigenfrequencies can be used to define the objective function and the constraints. The corresponding mathematical optimization formulations are solved using constrained non-linear gradient-based optimization algorithms.

Additionally, restrictions on the optimization formulation can be defined that represent manufacturing processes such as casting, stamping, additive manufacturing, etc. These restrictions limit the possible design space of the mathematical problem, but provide an optimized result that fulfills the part's manufacturing requirements.

In order to obtain such results and to be beneficial for the user, the algorithms of the restrictions need to be well-defined, robust and also computational performant. In the past, one could set up models and/or optimization formulations that do not lead to the intended result, for example, the optimal structure may contain cavities, unrealistic artifacts in the solution or need too much time to find a solution.

The aim of the master thesis is to review, develop and prototype various (analytic/heuristic) techniques improving the speed, stability and robustness of Tosca's manufacturing constraints focusing on demolding restrictions.

The specific tasks are

- Literature review and selection of appropriate techniques
- Review and understanding current implementations
- Development of new techniques and/or improve existing ones
- Prototype implementation
- Verification and validation on test examples
- Testing of the implemented features using both academic benchmark and industrial scale models
- Documentation of the implementation and the numerical experiments

Recommended fields of interest
optimization methods, finite element analysis

In cooperation with:
SIMULIA DASSAULT SYSTEMES

Abstract

Stuctural optimization has gained more significance over the recent years because of its ability to produce cost-effective designs without compromising the strength of the structure. One of the categories of structural optimization is the topology optimization, which determines the distribution of material by taking into account the inputs to a finite element model. The topology optimization uses element relative densities as the design variables. It finds the optimal density values for each element by satisfying the objective functions, for example minimizing compliance, maximizing eigenfrequencies etc., as well as the constraints, for example, mass, stress, displacement etc.

A topology optimization might be accompanied by the casting restrictions to enhance manufacturability of the optimal structure by casting process. If no casting constraints are specified to the optimization, the optimal design may contain interior cavities or undercuts, that damage the material during the demolding process. Out of many available methods that enforce casting restrictions, this thesis focusses on a method that achieves castable result by parametrizing the design domain. The concept of parametrization involves replacing groups of finite elements with an equivalent set of poles, that are parallel to the casting direction. Each of these poles has three parameters, that represent the beginning and end points along with the material available in the respective poles.

The optimization software, *SIMULIA Tosca Structure* has various features implemented for manufacturing constraints, along with casting constraints based on the parametrization. The objective of this thesis is to modify the parametrization in *SIMULIA Tosca Structure*, to obtain improved results and thereby preventing some potential cavities or irregularities occurring in the existing method. Two new implementations, namely Cuboid centroids and Volume fraction form the major discussion in this work. The Cuboid centroids method follows the same approach as the existing one except for different poles. The key idea behind Volume fraction method is to identify element fractions inside each pole. Accordingly, the mapping function, which relates the parameters to element relative densities and the sensitivities, i.e. the derivatives of a design response to the element densities are reformulated.

The Volume fraction method delivers better quality results, and Cuboid centroids offer faster runtime compared to the existing method, showcasing the effectiveness and capability of the new functionalities.

The formulated methods are prototyped and tested initially with the academic models as a proof of concept and then with industrial models to assess their potential.

Kurzfassung

Die Strukturoptimierung hat in den letzten Jahren zunehmend an Bedeutung gewonnen, da sie in der Lage ist kosteneffiziente Entwürfe zu erstellen, ohne die Festigkeit der Struktur zu beeinträchtigen. Eine der Kategorien der Strukturoptimierung ist die Topologieoptimierung, die die Materialverteilung unter Berücksichtigung der Eingaben in ein Finite-Elemente-Modell bestimmt. Bei der Topologieoptimierung werden die relativen Dichten der Elemente als Entwurfsvariablen verwendet. Sie findet die optimalen Dichtewerte für jedes Element, indem sie die Zielfunktionen wie Minimierung der Nachgiebigkeit, Maximierung der Eigenfrequenzen usw. sowie die Nebenbedingungen, z.B. Masse, Spannung, Verschiebung usw., erfüllt.

Eine Topologieoptimierung kann mit Gusseinschränkungen einhergehen, um die Herstellbarkeit der optimalen Struktur im Gießverfahren zu verbessern. Werden bei der Optimierung keine Gussrestriktionen angegeben, kann das optimale Design innere Hohlräume oder Hinterschneidungen enthalten, die das Material während des Entformungsprozesses beschädigen. Von den vielen verfügbaren Methoden, die Gießrestriktionen erzwingen, konzentriert sich diese Arbeit auf eine Methode, die ein gießbares Ergebnis durch Parametrisierung des Designbereichs erzielt. Das Konzept der Parametrisierung beinhaltet das Ersetzen von Gruppen von finiten Elementen durch eine äquivalente Menge von Polen, die parallel zur Gießrichtung liegen. Jeder dieser Pole hat drei Parameter, die die Anfangs- und Endpunkte zusammen mit dem in den jeweiligen Polen verfügbaren Material darstellen.

In der Optimierungssoftware SIMULIA Tosca Structure sind verschiedene Funktionen für Fertigungs- und Gussbeschränkungen auf der Grundlage der Parametrisierung implementiert. Das Ziel dieser Arbeit ist es, die Parametrisierung in *SIMULIA Tosca Structure* zu modifizieren, um bessere Ergebnisse zu erzielen und dadurch einige potenzielle Hohlräume oder Unregelmäßigkeiten zu vermeiden, die bei der bestehenden Methode auftreten. Zwei neue Implementierungen, nämlich Cuboid centroids und Volume fraction, bilden die Hauptdiskussion in dieser Arbeit. Die Methode der Quaderschwerpunkte folgt dem denselben Ansatz wie die bestehende Methode, nur mit anderen Polen. Der Grundgedanke hinter der Methode des Volumenanteils ist die Identifizierung von Elementanteilen innerhalb jedes Pols. Dementsprechend werden die Abbildungsfunktion, die die Parameter mit den relativen Elementdichten in Beziehung setzt, und die Sensitivitäten, d. h. die Ableitungen einer Entwurfsantwort von den Elementdichten, neu formuliert.

Die Volume fraction liefert qualitativ bessere Ergebnisse, und die Cuboid centroids bieten im Vergleich zur bestehenden Methode eine kürzere Laufzeit, was die Effektivität und Leistungsfähigkeit der neuen Funktionalitäten unter Beweis stellt.

Die formulierten Methoden werden in Prototypen umgesetzt und zunächst mit akademischen Modellen als Konzeptnachweis und dann mit industriellen Modellen getestet, um ihr Potenzial zu bewerten.

Preface

I want to express my deepest appreciation to my supervisors, Thorsten Mertins and Peter Clausen from Dassault Systèmes. I am deeply inspired by them and immensely grateful for all that I have learned under their guidance. Their expertise, unwavering support, and encouragement were invaluable, and I truly believe I could not have accomplished this work without them. I believe that both academically and professionally, I have experienced significant growth and improvement. The guidance and support I received from my supervisors have played a pivotal role in this development. I feel incredibly fortunate to have had such exceptional and friendly supervisors by my side.

I also want to extend my gratitude to Prof. Dr.-Ing. Malte von Scheven, for accepting me as thesis student at the Institut für Baustatik und Baudynamik. I am deeply thankful to my University supervisor Lisa-Marie Krauß, for her continous support and valuable insights. Her kind co-operation and supervision greatly facilitated the progress of this work.

I am thankful to the entire development team at Dassault Systèmes GmbH in Karlsruhe, for their indispensable help and their assistance with any queries I had.

Finally, I am immensely grateful to my sister, parents, and friends for their unwavering encouragement and support, both during the completion of this thesis and throughout my academic journey. Their belief in me has been a constant source of strength and motivation.

Stuttgart, in April 2024 Arvinth Shankar

Contents

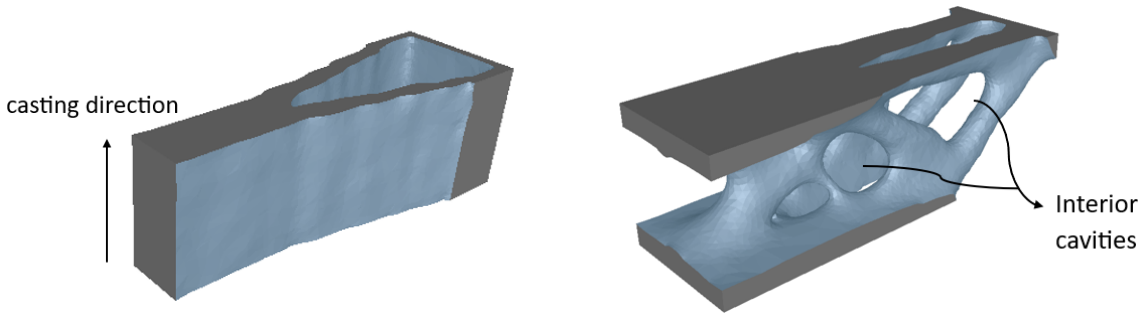
1

Introduction

The field of structural mechanics witnessed a significant technological advancement with the introduction of computers. The introduction and use of finite element analysis in the early 1940s has led to the vast development of many simulation tools that help in predicting the behavior of a structure under any physical condition. The intense competition between many industries motivates to apply certain optimization techniques along with the structural analysis. Structural optimization shows potential in generating lighter structures while still meeting specific performance requirements, leading to cost savings and sustainability benefits. In the past decade, topology optimization has emerged as a valuable tool, demonstrating its efficacy in discovering innovative designs that enhances the performance of products. Topology optimization involves an iterative approach aimed at achieving optimal structural performance by defining the material distribution within the design domain. This process takes into account various factors, including loads, displacements, boundary conditions, material properties, and more. During each iteration, finite element solvers (eg., SIMULIA Abaqus - Abaqus (2024)) are used to compute the stresses and strains within the structure based on the current material distribution. The optimization software, *SIMULIA Tosca Structure* (Tosca (2024)) utilizes gradient-based techniques to generate conceptual designs, that satisfies both load and support conditions, along with various constraints, such as mass, stiffness or eigenfrequency limits. Although topology optimization can reduce material usage and consequently lower costs, it may result in complex geometries or configurations that are impractical to manufacture. Therefore, additional manufacturing constraints are introduced to limit the range of solutions, ensuring that the structures are feasible to manufacture. Moreover, the manufacturing constraints also help in reducing the postprocessing effort of converting a design proposal to an actual design.

1.1 Motivation and objective

Topology optimization may result in an optimal structure that is difficult to manufacture by casting. This can be due to the presence of voids or interior cavities (see Fig 1.1b), or formation of undercuts in the structure. In order to obtain a castable structure (e.g, Fig 1.1a), many methods have been developed that enforce the casting restrictions to the optimization problem.



(a) *With cast constraint* **(b)** *Without cast constraint*

Figure 1.1: Topology optimization result

Harzheim und Graf (2002) introduced casting restrictions to the optimization by following a growth rule using the stress intensity in a given direction. This implementation works well, but the growth rule does not formulate a differentiable equation and is not suitable for gradient-based optimizations. The gradient-based optimizations are more robust and leads to faster convergence than non-gradient based approaches, especially for the problems with large number of design variables. Moreover, the drawback with the aforementioned implementation is that it works only with a voxel mesh (regular mesh with hexahedron finite elements). Zhou u. a. (2002) has proposed a method to obtain castable structures using the density gradient. The idea involves setting up a series of constraints that create a decreasing density gradient along the demolding direction, but this method introduces too many constraints on the design variables. This becomes a burden for the optimization algorithm, as the feasible design space becomes very limited. An alternative method has been proposed by Leiva u. a. (2004b) that enforces the casting restriction by means of parametrizing the design domain. In this method, the original design variables are converted to parameters and the gradient-based optimization algorithm works on these parameters. An advantage is that the number of parameters are less than the number of original design variables and this method does not introduce many constraints on these parameters.

SIMULIA Tosca Structure, has various features implemented for the manufacturing constraints, such as casting, symmetry conditions, etc. The casting constraint based on the parametrization in SIMULIA Tosca Structure is formulated by grouping the elements based on centroids into multiple poles. Each pole has three parameters denoting the beginning and end point along with the material available in each pole. A mapping function, that relates the parameters to design variables, is formulated based on the elements, identified by their centroids in each pole. The issue with the existing parametrization is the presence of high density elements also in the void regions, preventing clear segregation of the region with material and empty region. This subsequently leads to some inconsistent results with potential undercuts or cavities in the model. In this work, the parametrization is modified based on the volume fractions of elements, identified into multiple poles. Accordingly, the mapping function will be reformulated and therefore also the sensitivities (i.e. derivatives of a design response to

a design variable), that depends on the newly formulated mapping function. This proposed method is expected to prevent some potential undercuts, interior voids or artifacts that might occur in the existing parametrization.

The aim of this work is to develop, implement and test a prototype feature into *SIMULIA Tosca Structure* that modifies the current parametrization to obtain clear and consistent castable structures. For this objective, multiple industrial models are tested and validated using the current and new parametrization.

1.2 Tools used in this work

SIMULIA Tosca Structure is the structural nonparametric optimization software developed and licensed by Dassault Systèmes SE. The industrial finite element models, that are used for validating this work, are created with the finite element software *SIMULIA Abaqus CAE*. The optimization of the industrial models discussed in this thesis will include results obtained using Tosca Smooth. Tosca Smooth is a module available in *SIMULIA Tosca Structure*, that reconstructs the density plot to a design model, based on an isovalue or a target volume percentage. The isovalue serves as the threshold between elements showcasing zero material and those with maximum material values, guiding the construction of the isosurface between them.

1.3 Structure of the work

The subsequent chapters of this thesis comprise the following topics. In Chapter 2, the basic terminologies and some important working principles are discussed in the context of structural optimization. This chapter briefs the formulation of a typical topology optimization problem and the sensitivities. Understanding and having a solid foundation on these terms and concepts will help in comprehending the later chapters.

Chapter 3 will discuss the parametrization concept, that enforces the casting constraints to the optimization. More detailed discussions will be made on the pole parameters, grid and element parameters. The chapter concludes with the formulation of a modified topology optimization problem based on the parameters.

The Chapter 4 starts with the overall optimization process using the parametrization. The reformulated mapping functions and parameter sensitivities will constitute the important discussions in this chapter. This chapter also briefs the concepts and algorithms, that calculate the element volume fractions. The conclusion of the chapter gives an overall idea of the algorithms implemented in this work.

The final chapter will validate and compare the methods implemented in this work with the currently available method in *SIMULIA Tosca Structure*.

2

Structural optimization

This chapter starts with the introduction to structural optimization and followed by basic terminologies used in the context of structural optimization. Major categories of structural optimization are described in section 2.3. In section 2.4, a simple mathematical formulation for an optimization problem will be discussed. Additionally, optimization formulation for a simple mechanical problem will be discussed. In section 2.5, material interpolation and penalization techniques are discussed briefly. Outline of the sensitivity analysis is discussed in section 2.6 along with two approaches for computation of sensitivities. Method of Moving Asymptotes, an algorithm that helps in calculating the optimal values of design variables, will be discussed in section 2.7. In the last section, a common problem occurring in topology optimization and the solution to handle the problem will be discussed.

2.1 Introduction to structural optimization

The advancement and evolution of computational methodologies in structural design and analysis have led to the adoption of optimization techniques across various industries like automotive, aerospace, and construction. This streamlining of structural optimization expedites design processes while conserving resources in engineering applications. Structural optimization stands at the intersection of engineering, mathematics, and material science, seeking to find the optimal design solutions for a range of load-bearing structures. It involves the concept of defining objectives, whether minimizing weight or volume, maximizing stiffness, or minimizing the eigenfrequencies, while satisfying constraints such as stress thresholds, displacement limitations, material availability or any manufacturing constraints.

A commonly practiced approach involves integrating finite element analysis with optimization methods. This integration operates on individual finite elements or nodes, allowing mathematical formulations for optimization to arrive at an optimal structure. The result of optimization produced during each iteration of the analysis undergoes verification for adequate strength using finite element solvers, such as Abaqus. Hence, in this context, every iteration of optimization serves as a step closer to refining the design, enhancing performance and economizing

material usage. It involves a continuous interplay between theoretical concepts, computational models, and real-world applicability, leading to the evolution of innovative structures across diverse industries, from aerospace to civil engineering and beyond.

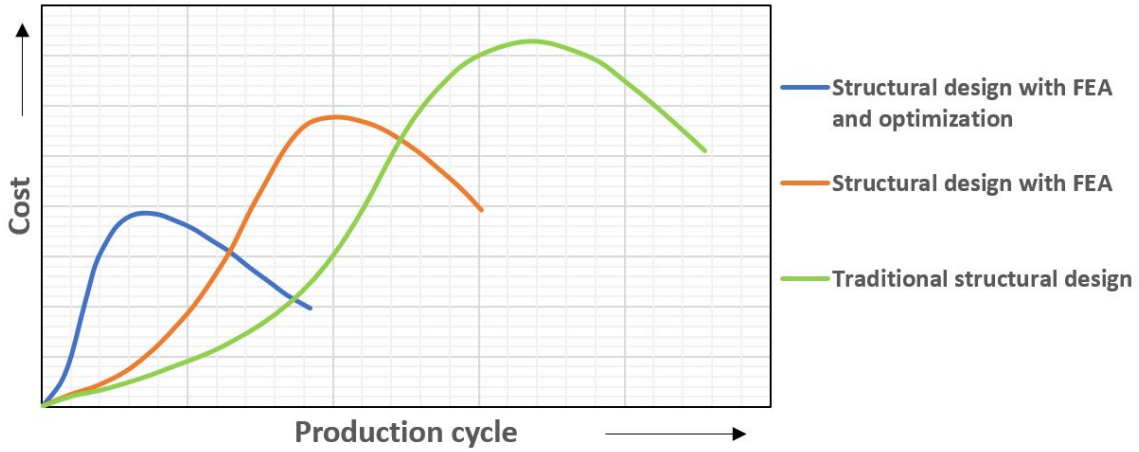


Figure 2.1: Cost comparison for structural design methods (Zhang u. a. 2020)

Fig 2.1, shows the difference in costs with and without the use of structural optimization in the design phase. This graph is only an example for understanding the cost difference and does not necessarily represent actual data from any experiment. The traditional structural design method is dependent on the design experience, often requiring multiple modifications of the design plan in later stages. Consequently, more manpower and material resources are consumed. The implementation of Finite Element Analysis (FEA) into the structural design has enhanced the design efficiency, leading to the reduction in cost in comparison to the traditional method. The cost for the design phase with FEA and optimization is initially higher due to the implementation cost of optimization tools or expertise. However, as the production cycle progress, the curve gradually slopes downward, indicating a reduction in cost over the production cycle. This downward trend symbolizes the cost-saving benefits achieved through optimized designs and efficiency gains. Unlike the optimized approach that strategically refines designs for cost reduction, the absence of such methodology in other structural design method limits the potential for cost-saving enhancements.

2.2 Basic terminologies in structural optimization

Following are the terms which are commonly used in the context of structural optimization (Christensen und Klarbring (2008)) and also used in *SIMULIA Tosca Structure*.

- **Design variables**: Variables which change throughout the optimization process to find an optimal or enhanced design result. For example: thickness, displacement of a particular structure, relative densities of each element in the finite element model, etc.

- **Design response**: Any quantity associated with the finite element model is called as the design response. In other words, design responses are any type of KPI. Key Performance Indicators (KPI) are measurable parameters that assist in tracking the performance and efficiency of a structure. For example: stress, displacement, strain energy, etc.
- **State variables**: These variables describe the current condition of the structure under certain loading conditions or constraints. They can be computed using the current design variables. For example: equivalent stresses, nodal displacements, etc.
- **Objective function**: This is the primary goal of an optimization, which is achieved by manipulating the design variables. For example: minimizing the strain energy or maximizing the stiffness, minimizing eigen frequencies, minimizing stress, etc.
- **Constraints**: Conditions or limitations that are imposed on particular design response that is necessary for a design to adhere to, example: volume constraints, displacement, manufacturing constraints, etc. Constraints limit the design space solutions when integrated into an optimization process.
- **Sensitivities** : These are the derivatives of a design response. In other words, the change of a design response with respect to a design variable. This is one of the crucial terms necessary for the optimization algorithm to achieve the optimal design satisfying the constraints.

2.3 Types of optimization

Following are the four major categories of structural optimization (Ide u. a. (2014)) that help in achieving a better structural design:

- **Topology optimization** : This finds the optimal material layout within the given design space. It pertains to determining characteristics like the quantity, placement and the configuration of voids, as well as establishing the connectivity of the domain. In most problems of topology optimization, relative density of each finite element serves as the design variable which is then adjusted to find the optimal material distribution of the structure.
- **Sizing optimization** : In this optimization, the thickness of shell elements are altered to attain the most efficient design while considering the objective function values and constraints.
- **Shape optimization** : This optimization is most commonly done to reduce any stress concentration, such as any sharp corners. This is achieved by moving the nodes of finite elements to optimal positions.
- **Bead optimization** : As the name says, this results in bead-like reinforcements to strengthen or stiffen the structural components, for example in sheet metals. This works similar to shape optimization by moving the finite element nodes to optimal positions.

2.4 Problem formulation

Considering α to be a design variable and a state variable defined by β, which is dependent on design variable α. The design response of the optimization problem is ψ which depends on both the variables α and β. Hence,

$$\begin{aligned} \text{minimize} \quad & \psi_{\text{obj}}(\alpha, \beta(\alpha)) \\ \text{subject to} \quad & \psi_i(\alpha, \beta) \qquad \text{where} \;\; i \in [1, m] \\ & \alpha_{\min} \leq \alpha \leq \alpha_{\max} \end{aligned} \tag{2.1}$$

The above formulation can be read as, minimizing the objective function ψ_{obj} subject to m number of constraints ψ_i and constraint for the design variable to be within $\alpha_{\min}$ and $\alpha_{\max}$ value.

Considering a structural mechanical problem and formulating it into a topology optimization problem,

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\boldsymbol{\sigma}\boldsymbol{\epsilon} \\ \text{subject to} \quad & V \leq V_{\max} \qquad \text{and} \\ & \boldsymbol{\rho}_{\min} \leq \boldsymbol{\rho} \leq \boldsymbol{\rho}_{\max} \end{aligned} \tag{2.2}$$

The problem then reads as, minimizing the strain energy density $\frac{1}{2}\boldsymbol{\sigma}\boldsymbol{\epsilon}$ with a volume constraint. $\boldsymbol{\sigma}$ is the stress calculated in the material, $\boldsymbol{\epsilon}$ is the strain, $\boldsymbol{\rho}$ is the relative density vector and acts as the design variable. Relative density of a finite element is the ratio of volume of solid material in an element to the total volume of an element. In addition to the volume constraint, the constraint on density of each finite element is that it needs to be within minimum $\boldsymbol{\rho}_{\min}$ and maximum $\boldsymbol{\rho}_{\max}$ values.

In this way, the relative densities of each finite element, here as design variables, are adjusted to determine an optimal structure, while satisfying the objective and constraint.

2.5 Interpolation and penalization for material stiffness

In the context of topology optimization, the interest lies in determining the optimal placement of material within a given design space. In other words, determining where the material should be present and at which point of space, there should be void. This is done with the help of relative density of each finite element as the design variable for the optimization. Relative density of an element is defined as follows.

$$\rho_i = \frac{V_i}{V_i^0} \tag{2.3}$$

where ρ_i is the relative density of an element, V_i is the volume of solid material in an element 'i' and V_i^0 is the total volume of an element 'i'. In discrete terms, a relative density of 0 signifies the lack of material, while a relative density of 1 signifies the presence of full material. As the final topology is desired to have either solid or void elements for better structure and manufacturability, the material stiffness of the finite elements are also defined in the following way

$$\mathbf{E}_e = \omega \mathbf{E}_e^{\mathbf{0}} \quad \text{where} \quad \omega = \begin{cases} 1, & \text{solid element} \\ 0, & \text{void element} \end{cases} \tag{2.4}$$

$\mathbf{E}_e$ is the stiffness of each element and $\mathbf{E}_e^{\mathbf{0}}$ is the assigned isotropic stiffness of each element. However, a discrete valued problem is difficult to solve as it is not differentiable and the sensitivities, which are necessary for an optimization algorithm, e.g, gradient descent algorithm, cannot be obtained. As a result of this, the discrete problem is converted to a continuous problem by means of replacing the integer variable to a continuous variable and introducing a penalization such that the desired solution contains 0-1 values.

A famous methodology that achieves this continuity through interpolation is called the Solid Isotropic Material with Penalization (SIMP) method. Using this method, the material stiffness is then formulated as,

$$\mathbf{E}_e = \boldsymbol{\rho}_e^p \mathbf{E}_e^{\mathbf{0}} \tag{2.5}$$

Here $\boldsymbol{\rho}_e$ is the relative density of each finite element which interpolates the value of stiffness between 0 and $\mathbf{E}_e^{\mathbf{0}}$. '$p$' plays an important role here and is known as the penalization factor.

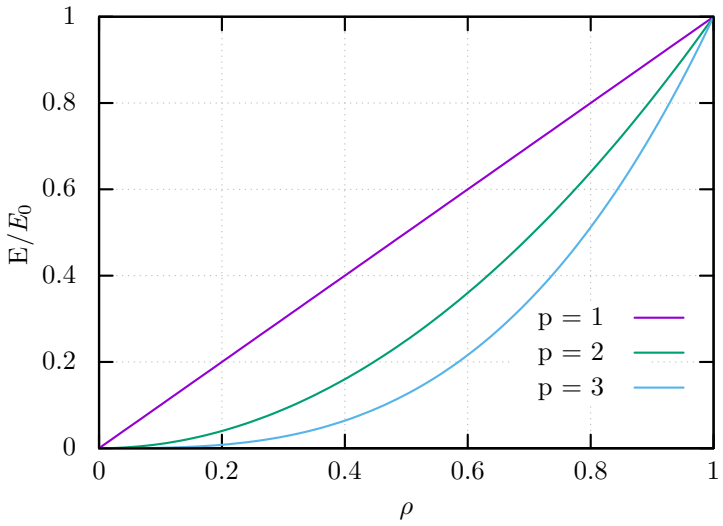


Figure 2.2: SIMP method - different penalization values

According to Bendsoe und Sigmund (1999), for the problems where there is active volume constraint, experience shows that choosing sufficiently large value for p results in desired

topology with less intermediate densities. Usually, $p \geq 3$ is required inorder to obtain a desired 0-1 result. This is because of higher penalization on the intermediate densities.

Fig 2.2, shows the interpolation curves for different values of penalization, $p = 1$, 2 and 3. For $p = 3$, the chance of relative density taking values closer to either 0 or 1 is higher than for $p = 2$ and $p = 1$. For a topology optimization, a 0-1 solution is always desired and therefore a penalization factor of 3 is normally considered.

2.6 Sensitivity analysis

Sensitivity analysis in structural optimization refers to the examination of how changes or perturbations in different design parameters or variables affect the performance or optimal solution of a structural system. The essence of sensitivity analysis lies in evaluating the change in design responses with respect to changes in specific design variables. It involves assessing how variations in parameters such as objective functions, constraints, etc., influence the optimized structural design.

To solve any optimization problem, sensitivities of objective function and constraints are crucial and they can be calculated by various methods. The computation of sensitivities for a wide range of applications to structural components and systems is given by Huang u. a. (1986). The following section will discuss the two important methods that can be used to find the sensitivities or derivatives of the design responses.

2.6.1 Direct approach

In order to explain the direct approach, consider the well known equilibrium equation based on the finite element analysis, which is,

$$\mathbf{K}(\boldsymbol{\alpha})\mathbf{x} = \mathbf{f}(\boldsymbol{\alpha}) \tag{2.6}$$

Here, $\mathbf{K}$ is the global stiffness matrix of a structure. $\mathbf{x}$ is the displacement vector and $\mathbf{f}$ is the force vector. $\boldsymbol{\alpha}$ is the design variable.

To describe the optimization problem, ψ is considered as the design response. Here, $\boldsymbol{\alpha}$ is assumed to be a vector with 'n' length. This implies that there are 'n' design variables, which have to be modified to obtain an optimal solution. The design variables can be imagined, for example, as a vector of densities of the finite elements in case of topology optimization.

Since both the global stiffness matrix and the force vector are dependent on the design variables, the solution is also design-dependent and hence, the state variable, which is the displacement vector $\mathbf{x}$, is dependent on the design variable $\boldsymbol{\alpha}$. So, $\mathbf{x} = \mathbf{x}(\boldsymbol{\alpha})$

In the majority of structural scenario, there exists a need to minimize or maximize an objective function while satisfying certain constraints such as stress, displacement and design variables.

A general function is then defined as the design response, which will explicitly depend on the design variables and also implicitly through the state variable.

$$\psi = \psi(\boldsymbol{\alpha}, \mathbf{x}(\boldsymbol{\alpha})) \tag{2.7}$$

The derivative of the design response with respect to the design variables is therefore, given by,

$$\frac{\mathrm{d}\psi}{\mathrm{d}\boldsymbol{\alpha}} = \frac{\partial\psi}{\partial\boldsymbol{\alpha}} + \frac{\partial\psi}{\partial\mathbf{x}}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\alpha}} \tag{2.8}$$

Differentiating both sides of the Eq. (2.6) with respect to $\boldsymbol{\alpha}$

$$\mathbf{K}(\boldsymbol{\alpha})\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\alpha}} = -\mathbf{x}\frac{\partial\mathbf{K}(\boldsymbol{\alpha})}{\partial\boldsymbol{\alpha}} + \frac{\partial\mathbf{f}(\boldsymbol{\alpha})}{\partial\boldsymbol{\alpha}} \tag{2.9}$$

In order to get $\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\alpha}}$,

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\alpha}} = \mathbf{K}^{-1}(\boldsymbol{\alpha})\left[\frac{\partial\mathbf{f}(\boldsymbol{\alpha})}{\partial\boldsymbol{\alpha}} - \mathbf{x}\frac{\partial\mathbf{K}(\boldsymbol{\alpha})}{\partial\boldsymbol{\alpha}}\right] \tag{2.10}$$

Eq. (2.10) is then substituted into the Eq. (2.8)

$$\frac{\mathrm{d}\psi}{\mathrm{d}\boldsymbol{\alpha}} = \frac{\partial\psi}{\partial\boldsymbol{\alpha}} + \frac{\partial\psi}{\partial\mathbf{x}}\mathbf{K}^{-1}(\boldsymbol{\alpha})\left[\frac{\partial\mathbf{f}(\boldsymbol{\alpha})}{\partial\boldsymbol{\alpha}} - \mathbf{x}\frac{\partial\mathbf{K}(\boldsymbol{\alpha})}{\partial\boldsymbol{\alpha}}\right] \tag{2.11}$$

The computational effort required to solve Eq. (2.11) is higher for large number of design variables. For example, in case of topology optimization, the density of each finite element is considered as the design variable and if the structure is discretized into a million finite elements, then Eq. (2.11) has to be solved a million times.

An alternative approach called adjoint method will address this challenge by introducing a lagrange multiplier into the formulation and thereby reducing the computational time and effort required to calculate the sensitivities. Mathematical formulations for the adjoint method will be discussed in the following section.

2.6.2 Adjoint approach

Adjoint method is widely used in gradient-based optimization due to its ability to compute the sensitivities for large scale optimization problems (Akgun u. a. (2001)). This method leads to reduced computational resources in comparison to the direct method.

The design response from Eq. (2.7) is modified such that a parameter called Lagrangian multiplier (λ) is introduced.

$$\psi = \psi(\boldsymbol{\alpha}, \mathbf{x}(\boldsymbol{\alpha})) + \boldsymbol{\lambda}^{\mathrm{T}}\underbrace{(\mathbf{K}\mathbf{x} - \mathbf{f})}_{\text{Residuum}} \tag{2.12}$$

Here, the residuum comes from Eq. (2.6), where residuum $\mathbf{R} \approx 0$.
Differentiating Eq. (2.12) with respect to design variable $\boldsymbol{\alpha}$

$$\frac{\mathrm{d}\psi}{\mathrm{d}\boldsymbol{\alpha}} = \frac{\partial\psi}{\partial\boldsymbol{\alpha}} + \frac{\partial\psi}{\partial\mathbf{x}}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\alpha}} + \boldsymbol{\lambda}^{\mathrm{T}}\left(\frac{\partial\mathbf{K}}{\partial\boldsymbol{\alpha}}\mathbf{x} + \mathbf{K}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\alpha}} - \frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\boldsymbol{\alpha}}\right) \tag{2.13}$$

In the above Eq. (2.13), if the force vector is independent of the design variables, then $\frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\boldsymbol{\alpha}}$ will not exist, $\frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\boldsymbol{\alpha}} = 0$.

The values of the lagrangian parameter $\boldsymbol{\lambda}$ are found in such a way that $\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\boldsymbol{\alpha}}$ is eliminated. Hence, Eq. (2.13) becomes

$$\frac{\mathrm{d}\psi}{\mathrm{d}\boldsymbol{\alpha}} = \frac{\partial\psi}{\partial\boldsymbol{\alpha}} + \boldsymbol{\lambda}^{\mathrm{T}}\left(\frac{\partial\mathbf{K}}{\partial\boldsymbol{\alpha}}\mathbf{x} - \frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\boldsymbol{\alpha}}\right) \tag{2.14}$$

Eq. (2.12) actually represents a Lagrangian equation

$$\mathcal{L} = \psi(\boldsymbol{\alpha}, \mathbf{x}(\boldsymbol{\alpha})) + \boldsymbol{\lambda}^{\mathrm{T}}(\mathbf{K}\mathbf{x} - \mathbf{f}) \tag{2.15}$$

In order to find the optimal solution, the differentiation of Lagrangian equation with respect to the state variable $\mathbf{x}$ should be 0. Assuming the stiffness matrix to be symmetrical, force vector to be independent of design variable and $\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathbf{x}} = 0$. From Eq. (2.15),

$$\mathbf{K}\boldsymbol{\lambda} = -\frac{\partial\psi}{\partial\mathbf{x}} \tag{2.16}$$

Eq. (2.16) is solved and the Lagrangian parameter $\boldsymbol{\lambda}$ obtained will be substituted into the equation Eq. (2.14) to calculate the final sensitivities.

It can also be understood that the computational time required to solve Eq. (2.16) is less than Eq. (2.10). This is because Eq. (2.16) is solved equal to the number of design responses available in the optimization problem, whereas Eq. (2.10) is solved for all the available design variables. Hence, adjoint method is numerically attractive for computing the sensitivities.

Moreover, if the optimization type is topology optimization and the design variables are the densities of each finite element, then Eq. (2.14) can be modified as,

$$\frac{\mathrm{d}\psi}{\mathrm{d}\boldsymbol{\rho}} = \frac{\partial\psi}{\partial\boldsymbol{\rho}} + \boldsymbol{\lambda}^{\mathrm{T}}\left(\frac{\partial\mathbf{K}}{\partial\boldsymbol{\rho}}\mathbf{x} - \frac{\mathrm{d}\mathbf{f}}{\mathrm{d}\boldsymbol{\rho}}\right) \tag{2.17}$$

Denoting the density vector $\boldsymbol{\rho}$ as ρ_j where $j \in [0,N]$ and N is the number of finite elements. Assuming the force is independent of the densities of the elements and applying the SIMP method,

$$\frac{\mathrm{d}\psi}{\mathrm{d}\rho_j} = \frac{\partial\psi}{\partial\rho_j} + p\rho_j^{p-1}\boldsymbol{\lambda}^{\mathrm{T}}\,[\mathbf{k}]_j\,\mathbf{x} \tag{2.18}$$

where p is the penalization factor as discussed already and $[\mathbf{k}]_j$ is the element stiffness matrix.

2.7 Method of Moving Asymptotes

The field of structural optimization presents a complex challenge due to its highly nonlinear and non-convex nature, involving numerous design variables along with diverse constraints. There are various techniques available for updating designs, e.g, sequential linear programming, sequential quadratic programming, method of moving asymptotes, optimality criteria etc. The Method of Moving Asymptotes introduced by Svanberg (1987) is one of the most robust and stable methods.

The algorithm to carry out structural optimization must be robust. The optimizer must handle large number of design variables and its capacity should extend to accommodate diverse constraints. The Method of Moving Asymptotes (MMA) addresses these challenges, offering solutions to handle general non-linear programming problems. This method relies on a unique form of convex approximation, which is easy to use and implement.

Considering a simple optimization problem as follows,

$$\begin{aligned} \text{minimize} \quad & \psi_{\text{obj}}(\boldsymbol{\alpha}) \\ \text{subject to} \quad & \psi_i(\boldsymbol{\alpha}) \leq \hat{\psi}_i \qquad \text{where } \; i \in [1, m] \\ & \underline{\alpha}_j \leq \alpha_j \leq \overline{\alpha}_j \qquad \text{where } \; j \in [1, n] \end{aligned} \tag{2.19}$$

Here, $\psi_{\text{obj}}(\boldsymbol{\alpha})$ is the objective function, $\psi_i(\boldsymbol{\alpha}) \leq \hat{\psi}_i$ represent 'm' number of constraints. There are 'n' number of design variables 'α' in the problem and $\underline{\alpha}_j$ and $\overline{\alpha}_j$ are their corresponding lower and upper bounds.

This method involves finding each $\psi_i^{(k)}$ by approximating ψ_i using linear expressions based on variables like $\frac{1}{(\alpha_j - L_j)}$ or $\frac{1}{(U_j - \alpha_j)}$, determined by the sign of the derivative of ψ_i at $\boldsymbol{\alpha}^k$, where k represents the current iteration. The values of L_j and U_j are adjusted across iterations, sometimes referred to as "Moving Asymptotes".

As MMA is versatile, these asymptotes can be adjusted to suit the specific convergence requirements of different problems. A general guideline, based on heuristics, regarding the modification of L_j and U_j values is as follows:

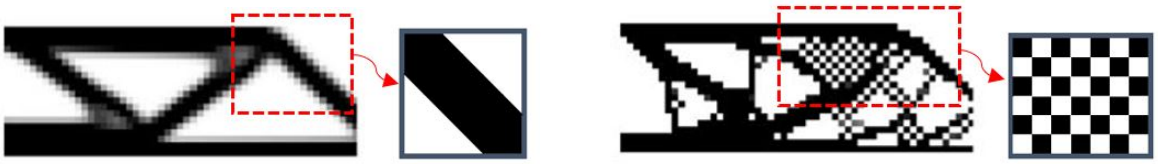
a) When there is a tendency for oscillation in the process, it requires stabilization. Achieving this involves moving the asymptotes near to the current iteration value.

b) Conversely, if the process progresses slowly in a consistent manner, it necessitates relaxation. This can be accomplished by moving the asymptotes away from the current iteration value.

For more details and formulations of this method, refer Svanberg (1987). An overview of the MMA and Globally Convergent MMA (GCMMA) hybrid algorithms can be found in Zuo u. a. (2007).

2.8 Sensitivity filter

At the core of structural optimization lies a finite element model comprising details regarding the discretization, mesh structure, load and displacement conditions. A crucial goal is to derive an optimized model that remains unaffected by changes in the mesh, ensuring mesh independence. One of the common problems encountered while carrying out topology optimization is the checkerboarding effect. This effect manifests as patterns resembling a checkerboard, where neighboring finite elements exhibit alternating solid and void characteristics. Such patterns or optimized structures typically fail to mirror physical reality and are often deemed unacceptable. Fig. (2.3) represents an example of checkerboarding effect. To prevent the



(a) No checkerboarding **(b)** With checkerboarding

Figure 2.3: Checkerboard effect (Sigmund 2001)

occurrence of the checkerboarding pattern, an image processing-based filtering technique has been incorporated (refer Sigmund (2007)), yielding results which are free from checkerboarding effect and representing the physical reality. Through this filtering process, sensitivities of the elements are adjusted using the formula below, essentially determining a weighted average of sensitivities within a specific radius.

$$\left(\frac{\partial\psi}{\partial\rho_{ei}}\right)_{\mathrm{m}} = \frac{1}{\max(\gamma,\rho_{ei})\sum_{j=1}^{n_j}\hat{D}_j}\sum_{j=1}^{n_j}\hat{D}_j\rho_{ej}\frac{\partial\psi}{\partial\rho_j}$$

$$\hat{D}_j = r_{\min} - \delta(i,j)$$

Here, $\left(\frac{\partial\psi}{\partial\rho_{ei}}\right)_{\mathrm{m}}$ are the modified sensitivities. ψ is the design response and the design variable are the densities of finite element (ρ_{ei}). γ is a small tolerance value to avoid division by zero. $\hat{D}_j$ is the convolution operator. n_j represent the set of element j for which the center to center distance $\delta(i,j)$ from element j to element i is smaller than the specified filter radius $r_{\min}$. The use of absolute distance within this convolution operator characterizes the filtering as geometric, consequently rendering the optimization problem independent of the mesh structure.

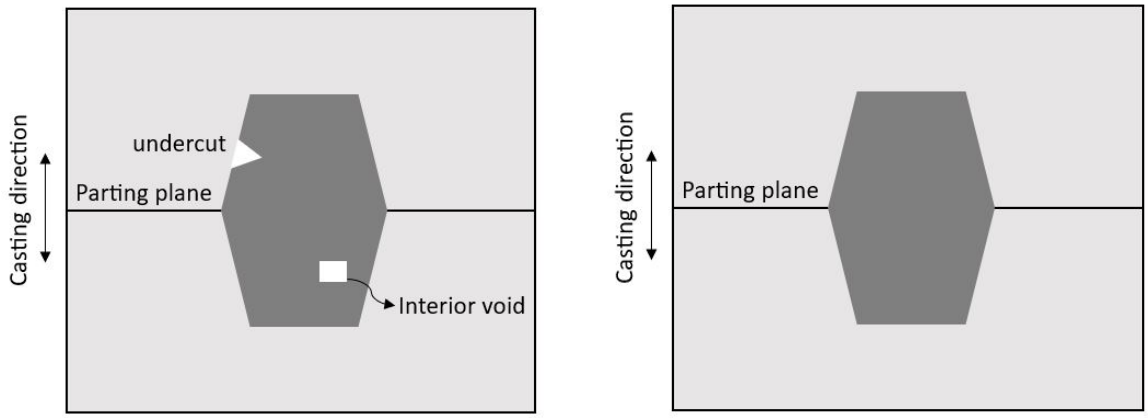
3

Casting constraint parametrization

This chapter starts with the introduction to the cast constraint topology optimization. It briefs the importance of casting restrictions to a structural design. There are various methods to enforce a casting restriction to the design and one of the methods using parametrization will be discussed in the section 3.2 . This concept of parametrization forms the basis of the current thesis work.

3.1 Introduction

The field of manufacturing engineering has evolved over many centuries. Today, there are various manufacturing methods like casting, forming, machining, etc. available to transform a material to some desired configuration. Various industries utilize one or more of these processes for manufacturing intricate structural components. Topology optimization often might result in optimal structures, that are not manufacturable using casting process.



(a) not castable **(b)** castable model

Figure 3.1: Castability of a design

They might result in the formation of undercuts or interior voids, which damage the metal part during the demolding process. Fig (3.1a) represents an example of undercuts and interior voids, whereas Fig (3.1b) represents a castable workpiece.

In the casting process, a molten metal is usually poured into the mould and after the liquid metal solidifies, the mould is removed from the parting plane along the casting direction. Fig (3.1a) and (3.1b) represent a casting process with the parting plane in the middle of the molten metal. In case of Fig (3.1a), the moulds on the top and bottom cannot be removed from the parting plane without damaging the material. This is because of the undercut and a void inside the molten material. On the other hand, Fig (3.1b) shows an example of the molten material without any voids and undercuts, and represents a model which is demoldable along the casting direction. It is often desirable to obtain a topology-optimized result, that is castable as depicted in the Fig (3.1b).

So, it becomes necessary that the manufacturability of a component is also taken into consideration during the topology optimization. There are many ways to enforce the manufacturability into the optimization. The casting constraints, for example, can be configured beforehand to the optimization in terms of demolding direction. The casting constraint prevents undercuts in the demolding direction and prohibits any interior voids, ensuring continuous material flow along the specified casting direction.

Zhou u. a. (2002) has proposed a method to obtain castable structures. The idea involves setting up a series of constraints that make the lower rows of elements to have higher densities than the higher rows of elements, essentially creating a decreasing density gradient along the demolding direction. This implementation would result in castable component along the given casting direction, but this approach is expensive as it imposes numerous additional constraints on the design variables. This is illustrated with the finite element model in Fig. (3.2). $i, j, ..., n$ denote the element indices for the first row of elements. The interior cavities in this model can

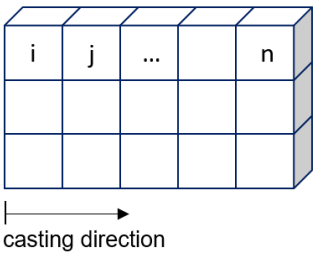


Figure 3.2: FE model with cast constraint (Zhou u. a. 2002)

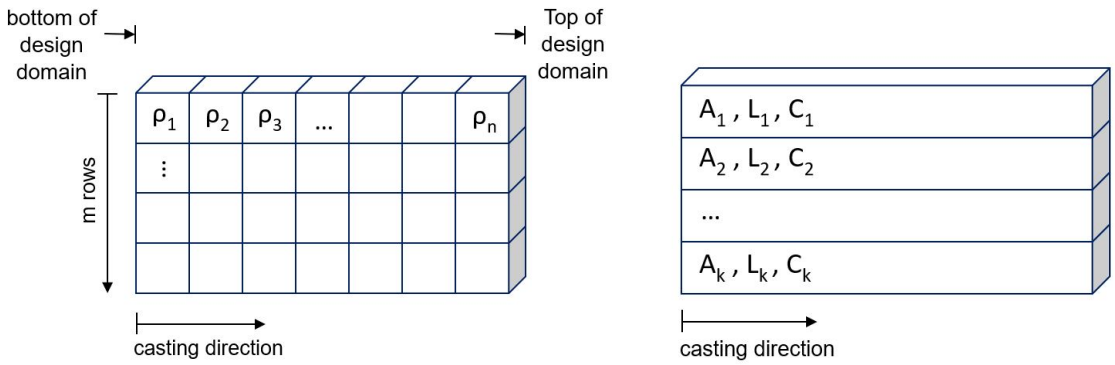
be prevented with the constraint, $\rho_i \geq \rho_j \geq ... \geq \rho_n$, where $\rho_i, \rho_j, ..., \rho_n$ represent the densities of elements in the same line along the casting direction. Similar constraints can be given to other elements that fall in the same line. An alternative method was suggested by Leiva u. a. (2004b), which produces castable design through parametrization. An advantage of this implementation is that it decreases the design variables without adding more constraints on the design variables. This idea of parametrization will be discussed in detail in the following section, which also forms as a basis for the major part of this thesis work.

3.2 Parametrization

In order to comprehend the concept of parametrization, it is essential to first delve into specific terminologies such as poles, pole parameters, etc., which will be discussed in detail in this section.

3.2.1 Formation of poles

The idea of enforcing casting constraint to the optimization is through parametrization of the element relative densities. The relative densities are converted to parameters which explicitly consider the casting direction. Fig (3.3a) shows an example of a finite element model. The first row of elements have relative densities as ρ_1, ρ_2, ρ_3 until ρ_n (see Fig (3.3a)), where n is the number of finite elements in each row that is pointing along the casting direction. The first row of elements can equivalently be represented by a single pole with three parameters A_1, L_1 and C_1. Similarly, all other rows of elements can be represented by poles and their parameters as shown in Fig (3.3b). The poles start from the bottom of the design domain and grows till the top of the design domain along the casting direction. The ends of the design domain are as marked in Fig (3.3a). As a whole, 'n x m' finite elements can be represented as 'k' number of poles, where 'n' is the number of finite elements equally in each row and 'm' is the number of rows of finite elements. The poles are constructed based on a projection algorithm as shown



(a) Finite elements **(b)** Poles

Figure 3.3: Parametrization of relative densities

in Fig (3.4). The centroids of the elements are projected onto a 2D plane and this plane is perpendicular to the casting direction. The algorithm starts with an element, which has minimum centroidal coordinate value in the casting direction. The neighbours of this element are identified within a specified radius on the 2D plane. These elements constitute a pole. The algorithm continues with the remaining elements forming second pole. This is continued till each projected element is contained uniquely inside a pole. Note that the field of parallel poles are oriented along the casting direction passing through the entire design domain. In this way, a discrete element mesh is replaced by set of poles with suitable parameters.

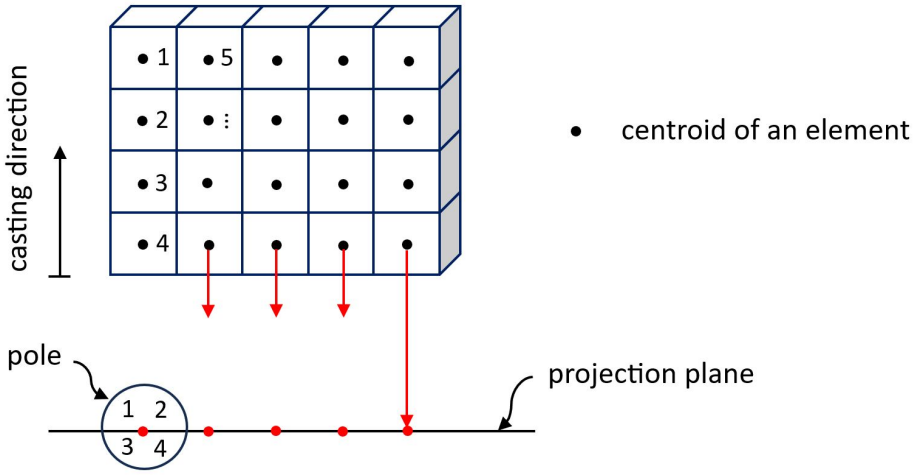


Figure 3.4: Projection of element centroids

3.2.2 Definitions for pole parameters

In previous subsection, it was discussed that each pole have three parameters and these parameters inherently enforce the casting constraint on the material. If there are 'k' number of poles, then the definitions of these parameters are given as follows

$A_k^p \rightarrow$ Density fraction of the pole; It is similar to the relative density of elements

$L_k^p \rightarrow$ Pole growth fraction; pole fraction, that is filled with material

$C_k^p \rightarrow$ Pole carve fraction; pole fraction, that has no material

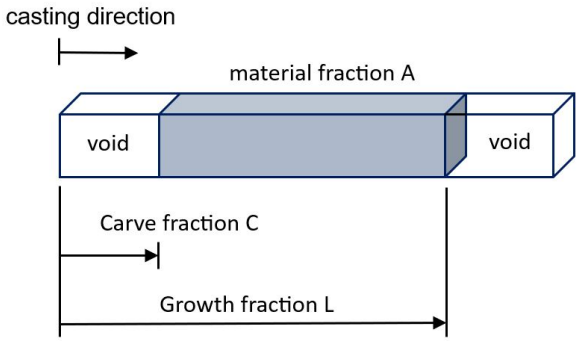


Figure 3.5: Pole parameters

Fig (3.5) shows the characterization of each pole 'k' by three parameters. It is important to note that both L_k^p and C_k^p are evaluated along the casting direction. The density of each pole ρ_k^p, is then given in terms of their parameters as,

$$\rho_k^p = A_k^p \cdot L_k^p \cdot (1 - C_k^p) \tag{3.1}$$

3.2.3 Grid parameters

In order to prevent excessive variations of the parameter values among the adjacent poles, the parameter values of each pole, are determined at each grid as weighted average of the respective pole parameters. This is explained in the following way,

$$A_j^g = \sum_{k=1}^{NP} w_{jk}(x_j, y_j) A_k^p \quad , \quad L_j^g = \sum_{k=1}^{NP} w_{jk}(x_j, y_j) L_k^p \quad , \quad C_j^g = \sum_{k=1}^{NP} w_{jk}(x_j, y_j) C_k^p \tag{3.2}$$

$$\text{where} \quad w_{jk} = \frac{e^{-\alpha(\frac{D_{jk}}{s})^2}}{\sum_{k=1}^{NP} e^{-\alpha(\frac{D_{jk}}{s})^2}} \quad \text{and} \quad D_{jk}(x_j, y_j) = \sqrt{(x_j - x_k)^2 + (y_j - y_k)^2}$$

Here, A_j^g, L_j^g, C_j^g are the grid parameters. The difference of grid parameters from the pole parameters A_k^p, L_k^p, C_k^p is that the grid parameters are the parameter values of each pole after filtering and the pole parameter values are unfiltered values.

w_{jk} is the Gaussian weighting factor. This weighting factor is normalized, so that when the pole parameters are 1.0, the respective grid parameters also take the value of 1.0. D_{jk} is the radial distance from a grid 'j' to any pole 'k'. It represents the distance between the filtered pole and its filtering neighbors. Here, z coordinate axis is assumed to be the casting direction and, the element centroids are projected onto a 2d plane along the z direction to form a pole. This is the reason for the distance D_{jk} and weighting factor w_{jk} to be functions of x_j and y_j, but independent of any z values. The parameters s and α in the exponential expression are predefined values and affects the effective radius and strength of the filter. Alternatively, the weighting factors can be expressed in linear forms in the following way,

$$w_{jk} = \frac{1 - \frac{D_{jk}}{s}}{\sum_{k=1}^{NP} 1 - \frac{D_{jk}}{s}} \quad \text{and} \quad D_{jk}(x_j, y_j) = \sqrt{(x_j - x_k)^2 + (y_j - y_k)^2} \tag{3.3}$$

Here, $D_{jk} < s$ and s denotes the filtering radius. Linear filter considers poles located at a larger distance from the center pole stronger into account compared to the exponential filter. Additionally, different filtering radii can be used for the pole density fraction A_k and pole carve fraction C_k. This is because the carve fraction C has to take more neighboring poles into account for filtering. The carve fraction C marks the beginning of the material, and therefore necessitates a smooth transition towards neighboring poles to prevent cracking. Conversely, if the filtering radius for fraction A is too large, the optimized model may include an excessive number of intermediate density elements. However, the desired result must be binary '0-1' without intermediate densities. In order to implement this difference in filtering radius, a filtering fraction $f = \frac{r_A}{r_C}$ is introduced. It is defined as the ratio of the filtering radius of A to the filtering radius for C.

3.2.4 Element parameters

Parametrizing the design domain ensures that the optimized structure is castable. However, it is important to map the pole parameters to the original finite elements and their respective relative densities as the result for a topology optimization. In order to calculate the parameters at the element level, the grid parameters that contain the respective elements are averaged in the following way

$$A_i^e = \sum_{j=1}^{NG_i} \frac{A_j^g}{NG_i} \qquad L_i^e = \sum_{j=1}^{NG_i} \frac{L_j^g}{NG_i} \qquad C_i^e = \sum_{j=1}^{NG_i} \frac{C_j^g}{NG_i} \tag{3.4}$$

NG_i is the number of grids associated with the element i. This averaging helps in achieving a better approximation for each element on coarse meshes. The element parameters are defined as a function of pole parameters through a interpolation function ψ_i in the following way

$$A_i^e = \psi_i(\mathbf{A}) \qquad L_i^e = \psi_i(\mathbf{L}) \qquad C_i^e = \psi_i(\mathbf{C})$$

$$\begin{aligned} \psi_i(\mathbf{A}) &= \sum_{j=1}^{NG_i} \frac{1}{NG_i} \Big(\sum_{k=1}^{NP} w_{jk}(x_j, y_j) A_k^p\Big) \qquad \psi_i(\mathbf{L}) = \sum_{j=1}^{NG_i} \frac{1}{NG_i} \Big(\sum_{k=1}^{NP} w_{jk}(x_j, y_j) L_k^p\Big) \\ \psi_i(\mathbf{C}) &= \sum_{j=1}^{NG_i} \frac{1}{NG_i} \Big(\sum_{k=1}^{NP} w_{jk}(x_j, y_j) C_k^p\Big) \end{aligned} \tag{3.5}$$

$\psi_i(\mathbf{A})$, $\psi_i(\mathbf{L})$, $\psi_i(\mathbf{C})$ are the interpolating functions for parameters $\mathbf{A}$, $\mathbf{L}$ and $\mathbf{C}$. Here, $\mathbf{A}$, $\mathbf{L}$, $\mathbf{C}$ represent the vector of pole parameters with its length equal to the number of poles.

3.2.5 Analytical growth rule

With the help of the growth fraction L_i^e and carve fraction C_i^e in the vicinity of the element i, the actual material growth and actual material carved can be calculated as,

$$\begin{aligned} H_i^e(L_i^e) &= (H_{i,\max}^e - H_{i,\min}^e) L_i^e + H_{i,\min}^e \\ B_i^e(C_i^e) &= (H_{i,\max}^e - H_{i,\min}^e) C_i^e + H_{i,\min}^e \end{aligned} \tag{3.6}$$

Here, $H_i^e(L_i^e)$ and $B_i^e(C_i^e)$ are the actual growth and carved material on the vicinity of the element i. $H_{i,\max}^e$ and $H_{i,\min}^e$ are the maximum and minimum heights of the pole that contains the element i. Fig. (3.6) shows a detailed description of all these dimensions. In this figure, z coordinate axis of the finite element model is chosen as the casting direction.

From the actual material growth and the carved material, the element level parameters, i.e., the element growth and carve fraction can be obtained as given in the Eq. (3.7).

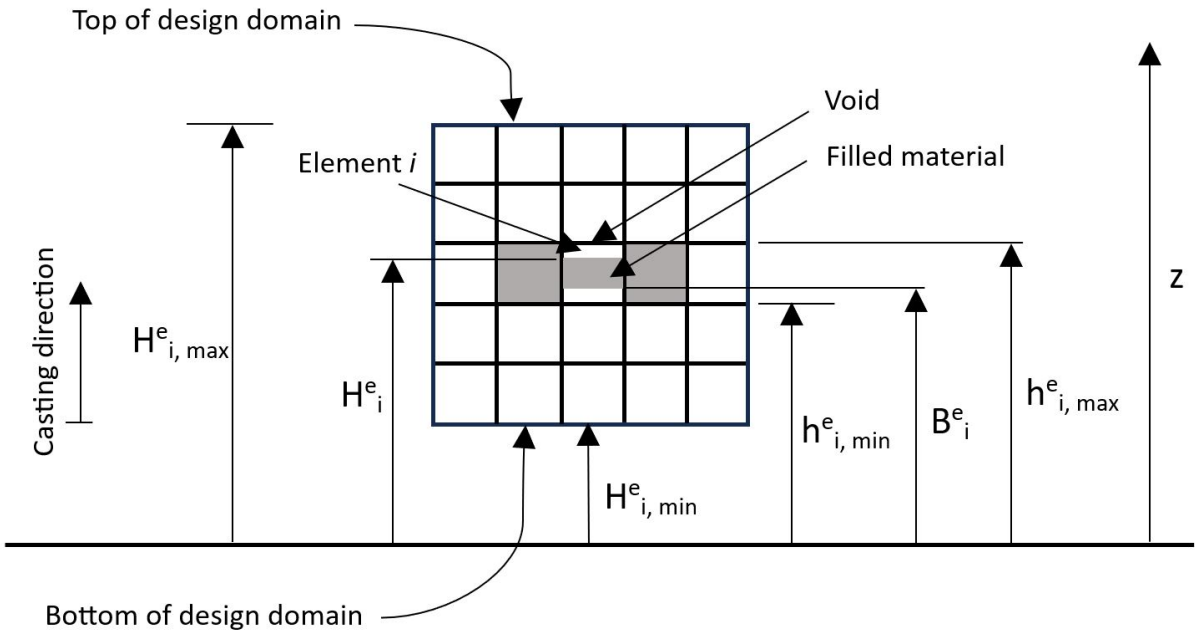


Figure 3.6: Growth and carve parameters

$$l_i^e = \phi(H_i^e) = \begin{cases} 1, & \text{if} \quad H_i^e \geq h_{i,\max}^e \\ \phi(H_i^e), & \text{if} \quad h_{i,\min}^e \leq H_i^e \leq h_{i,\max}^e \\ 0, & \text{if} \quad H_i^e \leq h_{i,\min}^e \end{cases}$$

$$c_i^e = \phi(B_i^e) = \begin{cases} 0, & \text{if} \quad B_i^e \geq h_{i,\max}^e \\ 1 - \phi(B_i^e), & \text{if} \quad h_{i,\min}^e \leq B_i^e \leq h_{i,\max}^e \\ 1, & \text{if} \quad B_i^e \leq h_{i,\min}^e \end{cases} \tag{3.7}$$

$$\text{with} \quad \phi(H_i^e) = \frac{H_i^e - h_{i,\min}^e}{h_{i,\max}^e - h_{i,\min}^e} \quad \text{and} \quad \phi(B_i^e) = \frac{B_i^e - h_{i,\min}^e}{h_{i,\max}^e - h_{i,\min}^e}$$

Here, l_i^e and c_i^e are the element growth and carve fraction respectively. $h_{i,\min}^e$ and $h_{i,\max}^e$ are the minimum and maximum height of the element i. Therefore, these fractions can be expressed as a function of the pole parameters as

$$\begin{aligned} l_i^e &= \phi(H_i^e(\psi_i(\mathbf{L}))) \\ c_i^e &= 1 - \phi(B_i^e(\psi_i(\mathbf{C}))) \end{aligned} \tag{3.8}$$

From Eq. (3.8),

$$b_i^e = l_i^e \cdot c_i^e = \phi(H_i^e(\psi_i(\mathbf{L}))) \cdot (1 - \phi(B_i^e(\psi_i(\mathbf{C})))) \tag{3.9}$$

Here, b_i^e represents a rule for filling the element completely or emptying the element. With the help of this growth rule, a 0-1 solution can be obtained, which is the desired result for a topology optimization. In addition to analytical growth rule formulations, there are other functions, for example, smooth Heaviside function, which also approximate the solution to 0-1. The formulation using smooth Heaviside function will be discussed in next section.

3.2.6 Heaviside function

Liu u. a. (2015) have presented a method of topology optimization for the design of vertically-walled stiffeners manufactured using casting process. In this method, two separate design variables are used, namely a density field representing the topology of the ribs and a height field that represents the profile of the ribs. This density field and height field is similar to the pole area density A and pole growth fraction L respectively, discussed in the previous sections. Gersborg und Andreasen (2010) have proposed an explicit Heaviside parametrization in gradient driven topology optimization that provides a way to obtain a castable design. Considering the element growth fraction l_i^e and defining it using a smooth Heaviside function,

$$l_i^e = H(L_k^p, \beta, s) = \frac{e^{\beta(L_k^p - s)}}{1 + e^{\beta(L_k^p - s)}} \quad , \quad s = \frac{h_0 + H_{i,\min}^e}{H_{i,\max}^e - H_{i,\min}^e} \tag{3.10}$$

where $H(L_k^p, \beta, s)$ is the Heaviside function. L_k^p is the growth fraction of the pole in which element i belongs. β is a constant and s is the normalized centroid value. $H_{i,\min}^e$ and $H_{i,\max}^e$ are minimum and maximum heights of the design domain. h_0 is the height of the element's centroid parallel to the casting direction.

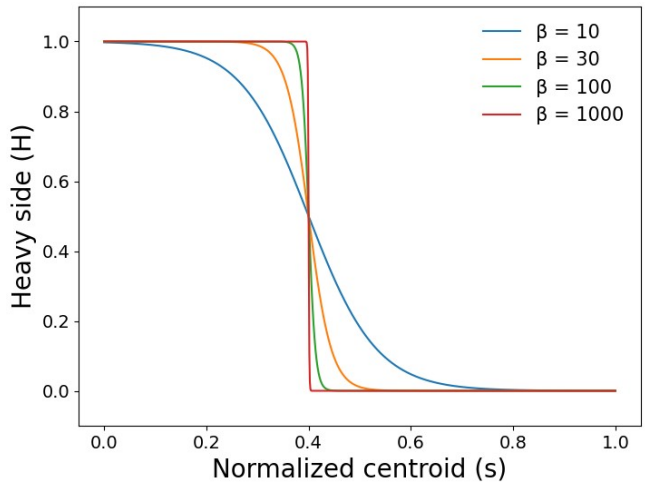


Figure 3.7: Heaviside function (Liu u. a. (2015))

Here, β is an important factor, which affects the result of the optimization. With low β value, the optimization can result in intermediate density elements. If the β value is high, the approximation becomes steeper and discontinuous. As a result, it cannot be differentiated to

obtain sensitivities. Fig (3.7) shows, for different values of β, the change in the steepness of the Heaviside curve, assuming a value of 0.4 for L_k^p. An equation similar to Eq. (3.10) can be made for the element carve fraction c_i^e with the help of pole carve fraction C_k^p and the normalized centroid value s. The difference in comparison to Eq. (3.10) is that the carve fraction for the poles is used instead of growth fractions. Moreover, the subtraction is reversed in the power of the exponential terms.

$$c_i^e = H(C_k^p, \beta, s) = \frac{e^{\beta(s-C_k^p)}}{1 + e^{\beta(s-C_k^p)}} \quad , \quad s = \frac{h_0 + H_{i,\min}^e}{H_{i,\max}^e - H_{i,\min}^e} \tag{3.11}$$

3.2.7 Element relative density

The correlation between the element relative density and the element parameters is

$$\rho_i^e = A_i^e \cdot l_i^e \cdot c_i^e \tag{3.12}$$

where, ρ_i^e is the relative density of each finite element, A_i^e is the element area density, l_i^e is the element growth fraction and c_i^e is the element carve fraction. A_i^e is the density of the element when it is fully filled. The multiplication of l_i^e and c_i^e, i.e., $l_i^e \cdot c_i^e$ determines whether an element has full density ($l_i^e \cdot c_i^e = 1.0$), takes an intermediate value ($0.0 \leq l_i^e \cdot c_i^e \leq 1.0$) or whether an element is completely empty ($l_i^e \cdot c_i^e = 0.0$). Note that the element growth and carve fractions are parallel to the given casting direction.

Combining Eq. (3.5), (3.9) and (3.12) yields the following equation :

$$\rho_i^e = \psi_i(\mathbf{A}) \cdot \phi(H_i^e(\psi_i(\mathbf{L}))) \cdot (1 - \phi(B_i^e(\psi_i(\mathbf{C})))) \tag{3.13}$$

Eq. (3.13) gives an explicit relation between the element relative density and the pole parameters. This equation can also be differentiated with respect to the pole parameters, which helps in computing the necessary sensitivities needed for the optimizer.

3.3 Formulation for a cast constraint optimization

In the cast constraint parametrization, the pole parameters are considered as the design variables for the optimizer. A topology optimization problem with the cast constraints based on parametrization is given as,

$$\begin{aligned} \text{Minimize} \quad & \psi_{obj}(\rho_1, \rho_2, \rho_3, ..., \rho_n) \\ \text{subject to} \quad & \psi_{const,i} \ \text{, where } i \in [0, m] \\ & \rho_j = \rho_j(\mathbf{A}, \mathbf{L}, \mathbf{C}) \ , j \in [0, n] \\ & 0.0 \leq A_k^p, L_k^p, C_k^p \leq 1.0; \ \ k = 1, \text{Number of poles} \end{aligned} \tag{3.14}$$

Here, the objective function is represented as ψ_{obj} and with 'm' number of constraints ψ_{const}. Note that the relative densities for each element ρ_j is represented based on the parameters, through a mapping function as defined in Eq. (3.13). The design variable constraint in the above formulation is the restriction of the parameter values between 0.0 and 1.0. This is similar to having a relative density value from 0.0 to 1.0 in any topology optimization problem.

It is important to highlight that this parametric topology optimization utilizes less number of design variables than the original. For example, considering a cube that is discretized into 10×10×10 finite elements. The number of pole design variables will be 3×10×10, specifically the three parameters A, L and C for each of the 10×10 poles. In this way, there is 70% reduction in number of design variables.

In the current work, the parametrization will be improved based on the volume fractions of finite elements in each pole. Consequently, the mapping function from Eq. (3.13) and the sensitivities, which are the derivatives of a design response with respect to the parameters, will be modified to improve the result of cast constraint parametrization. The reformulation of mapping function, sensitivities and the need for modification will be discussed in detail in the following chapter.

4

Mapping with volume fraction

This chapter starts with the overview of the optimization process using the cast constraint parametrization. The main discussions of this chapter will be on the reformulated mapping functions and sensitivities based on the volume fraction of elements in each pole. This chapter also briefs the element splitting and cutting concepts that help in calculating the element volume fractions along with pseudo algorithms to calculate and store the volume fraction values in sparse format. Near the end of the chapter, the overall code that is implemented for this work will be discussed.

4.1 Overall optimization process

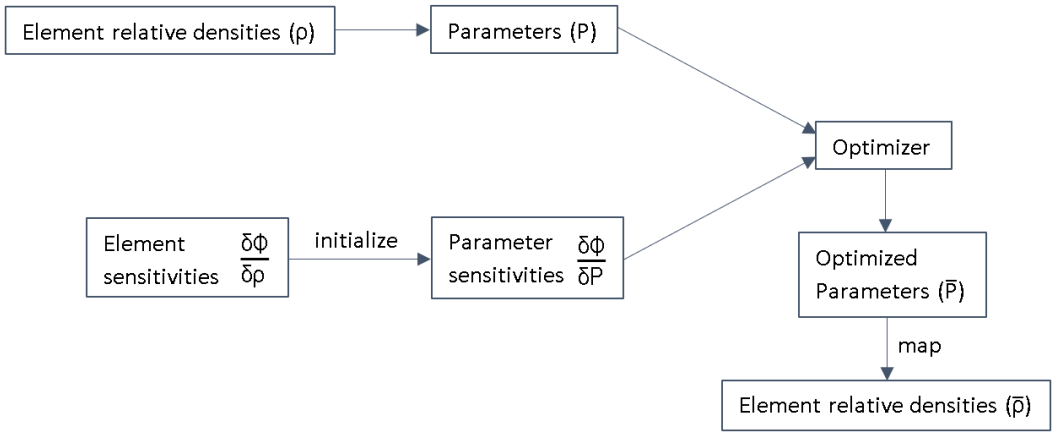


Figure 4.1: Overview of a topology optimization with cast constraint parametrization

The overview of optimization process considering the parametrization is shown in Fig (4.1). The parameters 'P' here refer to the pole parameters **A**, **L** and **C** discussed in the previous chapter. ϕ refers to any design response. Note that the pole parameters **A**, **L** and **C** are vectors of length equal to the number of poles. This process of optimization using the parameters is explained in the following way. Parameters **A**, **L** and **C** are initialized from the original relative

densities of each finite element. In addition, the parameter sensitivities $\frac{\partial \phi}{\partial \mathbf{A}}$, $\frac{\partial \phi}{\partial \mathbf{L}}$ and $\frac{\partial \phi}{\partial \mathbf{C}}$ are obtained from the known element sensitivities through a mapping. Both the parameters and their corresponding sensitivities are given to the optimizer. Here, the parameters become the design variables and the optimizer gives back the optimized values of the parameters. It is denoted as '$\overline{P}$' and in terms of pole parameters as $\overline{\mathbf{A}}$, $\overline{\mathbf{L}}$ and $\overline{\mathbf{C}}$. These parameters have to be transformed back to the relative densities of each finite element. This is done with the help of a mapping function, that will be discussed in the following sections. This process is repeated until the convergence.

4.2 General mapping formulation

The initial optimization problem reads as,

$$\begin{aligned} &\text{minimize} && \psi_{\text{obj}}(\mathbf{x}) \\ &\text{subject to} && \psi_{const,i}(\mathbf{x}) \qquad \text{where } \; i \in [1, m] \end{aligned} \tag{4.1}$$

Here, ψ_{obj} is the objective function depending on the design variables $\mathbf{x}$. There are 'm' number of constraints $\psi_{const,i}$, that depends on the design variables. In a cast constraint parametrization, the design variables are transformed to some parameters '$\mathbf{P}$' and the optimizer works on these parameters. This transformation is described through a mapping function as $\mathbf{x} = F(\mathbf{P})$. Hence, the new optimization problem is also transformed based on the parameters as,

$$\begin{aligned} &\text{minimize} && \widetilde{\psi}_{\text{obj}}(\mathbf{P}) \\ &\text{subject to} && \widetilde{\psi}_{const,i}(\mathbf{P}) \qquad \text{where } \; i \in [1, m] \end{aligned} \tag{4.2}$$

Consequently, the element sensitivities are mapped to the parameter sensitivities with the help of the mapping function F mentioned before. The objective and the constraint sensitivities are given as follows.

$$\frac{\mathrm{d}\widetilde{\psi}_{\text{obj}}}{\mathrm{d}\mathbf{P}} = \frac{\partial \psi_{\text{obj}}}{\partial \mathbf{x}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{P}} \qquad \text{and} \qquad \frac{\mathrm{d}\widetilde{\psi}_{const,i}}{\mathrm{d}\mathbf{P}} = \frac{\partial \psi_{const,i}}{\partial \mathbf{x}} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{P}} \tag{4.3}$$

Here $\frac{\partial \psi_{\text{obj}}}{\partial \mathbf{x}}$ and $\frac{\partial \psi_{const,i}}{\partial \mathbf{x}}$ represent the known element objective and constraint sensitivities respectively. The term, $\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\mathbf{P}}$ is calculated based on the defined mapping function F. The mapping function F defined using the pole parameters and the relative densities of the finite elements will be discussed in the next section.

4.3 Mapping of pole parameter to element relative density

The design variables for a topology optimization are the relative densities of each finite element ρ_i^e and the parameters are pole parameters $\mathbf{A}$, $\mathbf{L}$ and $\mathbf{C}$. The mapping function between them

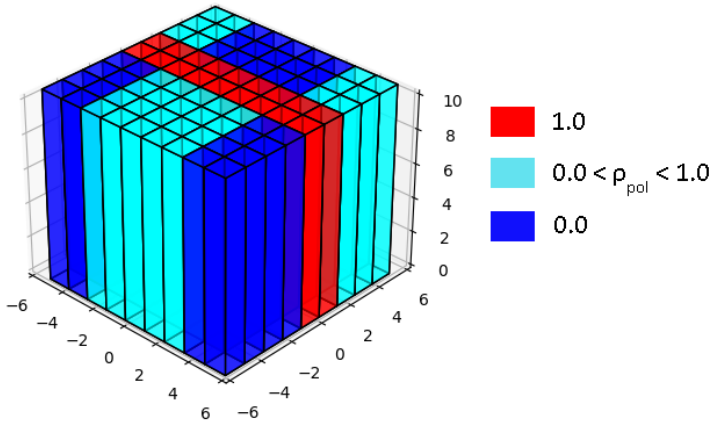
is given as,

$$\boldsymbol{\rho} = F(\mathbf{A}, \mathbf{L}, \mathbf{C}) \tag{4.4}$$

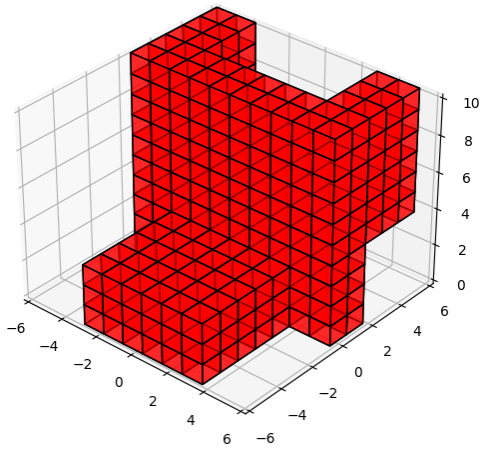
This mapping function F is in turn formulated as multiplicative functions of each parameter separately,

$$F = f_a(\mathbf{A}).f_l(\mathbf{L}).f_c(\mathbf{C}) \tag{4.5}$$

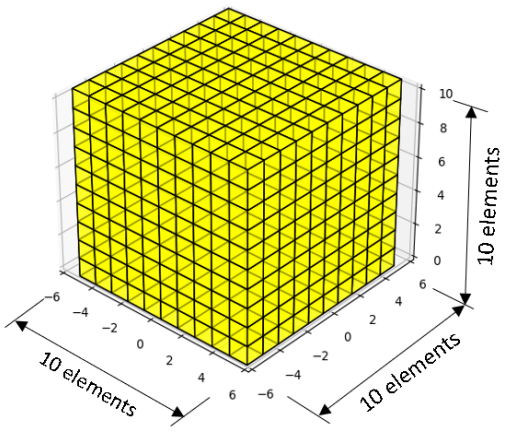
The functions $f_l(\mathbf{L})$ and $f_c(\mathbf{C})$ are defined as given in the equation Eq. (3.8) using the analytical growth rule discussed in the section 3.2.5. l_i^e and c_i^e in the Eq. (3.8) basically represents the functions $f_l(\mathbf{L})$ and $f_c(\mathbf{C})$ respectively. These functions can also be formulated using the smooth Heaviside function described in the section 3.2.6. The function $f_a(\mathbf{A})$ is defined in a simplified manner as $f_a(\mathbf{A}) = \psi_i(\mathbf{A})$, where $\psi_i(\mathbf{A})$ is a function that is defined in the Eq. (3.5).



(a) *Pole densities*



(b) *Element relative densities*



(c) *Initial finite element model*

Figure 4.2: Visualization of density fields

Fig (4.2) shows a simple visualization of the pole densities and the final element density field, that is obtained from the pole densities with the help of mapping function described before. Note that the parameters A, L and C are combined to represent as pole densities in the Fig (4.2a); refer Eq. (3.1). ρ_{pol} in the figure represents the pole densities. In this case, the pole size is considered to be equal to the edge length of an element. It can be observed that a pole has full density, if all the elements inside that pole are completely filled, i.e., all elements inside a particular pole has relative density value of 1.0 (represented in red marking). On the other hand, the pole takes a value of 0.0, if none of the elements inside that pole are filled.

4.4 Sensitivities

Based on the function 'F' defined in Eq. (4.4), the sensitivities of parameters are also calculated from the element sensitivities through a chain rule of differentiation as,

$$\begin{aligned}
\frac{\mathrm{d}\phi}{\mathrm{d}\mathbf{A}} &= \frac{\partial\phi}{\partial\boldsymbol{\rho}}\frac{\mathrm{d}\boldsymbol{\rho}}{\mathrm{d}\mathbf{A}} = \frac{\partial\phi}{\partial\boldsymbol{\rho}}\frac{\mathrm{d}F(\mathbf{A},\mathbf{L},\mathbf{C})}{\mathrm{d}\mathbf{A}} \\
\frac{\mathrm{d}\phi}{\mathrm{d}\mathbf{L}} &= \frac{\partial\phi}{\partial\boldsymbol{\rho}}\frac{\mathrm{d}\boldsymbol{\rho}}{\mathrm{d}\mathbf{L}} = \frac{\partial\phi}{\partial\boldsymbol{\rho}}\frac{\mathrm{d}F(\mathbf{A},\mathbf{L},\mathbf{C})}{\mathrm{d}\mathbf{L}} \\
\frac{\mathrm{d}\phi}{\mathrm{d}\mathbf{C}} &= \frac{\partial\phi}{\partial\boldsymbol{\rho}}\frac{\mathrm{d}\boldsymbol{\rho}}{\mathrm{d}\mathbf{C}} = \frac{\partial\phi}{\partial\boldsymbol{\rho}}\frac{\mathrm{d}F(\mathbf{A},\mathbf{L},\mathbf{C})}{\mathrm{d}\mathbf{C}}
\end{aligned} \tag{4.6}$$

Here, ϕ refers to any design response. $\frac{\partial\phi}{\partial\boldsymbol{\rho}}$ are the already known element sensitivities. In most cases, these are obtained from finite element solver (e.g. *SIMULIA Abaqus*). The term that involves the differentiation of function F with respect to $\mathbf{A}$, $\mathbf{L}$ and $\mathbf{C}$ are calculated based on the functions $f_a(\mathbf{A})$, $f_l(\mathbf{L})$ and $f_c(\mathbf{C})$ respectively.

$$\begin{aligned}
\frac{\mathrm{d}F(\mathbf{A},\mathbf{L},\mathbf{C})}{\mathrm{d}\mathbf{A}} &= \frac{\mathrm{d}f_a(\mathbf{A})}{\mathrm{d}\mathbf{A}}.f_l(\mathbf{L}).f_c(\mathbf{C}) \\
\frac{\mathrm{d}F(\mathbf{A},\mathbf{L},\mathbf{C})}{\mathrm{d}\mathbf{L}} &= \frac{\mathrm{d}f_l(\mathbf{L})}{\mathrm{d}\mathbf{L}}.f_a(\mathbf{A}).f_c(\mathbf{C}) \\
\frac{\mathrm{d}F(\mathbf{A},\mathbf{L},\mathbf{C})}{\mathrm{d}\mathbf{C}} &= \frac{\mathrm{d}f_c(\mathbf{C})}{\mathrm{d}\mathbf{C}}.f_a(\mathbf{A}).f_l(\mathbf{L})
\end{aligned} \tag{4.7}$$

With the help of Eq. (3.5), (3.6), (3.7) and (3.8), the derivatives are calculated as,

$$\begin{aligned}
\frac{\mathrm{d}f_a(\mathbf{A})}{\mathrm{d}A_k^p} &= w \\
\frac{\mathrm{d}f_l(\mathbf{L})}{\mathrm{d}L_k^p} &= \frac{H^e_{\max} - H^e_{\min}}{h^e_{\max} - h^e_{\min}} \cdot w \qquad \text{Here,} \quad w = \sum_{j=1}^{NG_i} \frac{1}{NG_i} \sum_{k=1}^{NP} w_{jk}(x_j, y_j) \\
\frac{\mathrm{d}f_c(\mathbf{C})}{\mathrm{d}C_k^p} &= -\frac{H^e_{\max} - H^e_{\min}}{h^e_{\max} - h^e_{\min}} \cdot w
\end{aligned} \tag{4.8}$$

Here A_k^p, L_k^p and C_k^p are the parameters of the pole 'k' in which the element 'i' belongs. The variable 'j' in the term 'w' comes into account if the filtering of pole is performed. $H^e_{\min}$ and

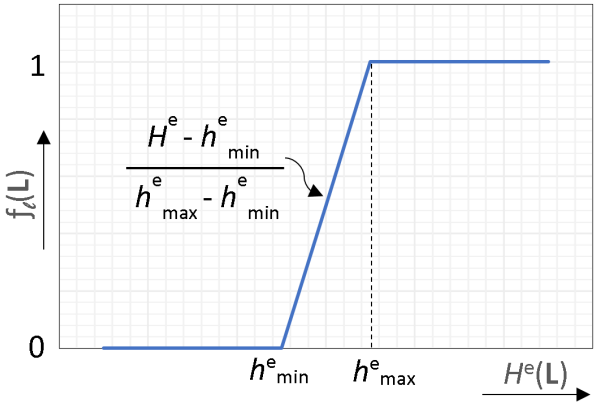
$H^e_{\max}$ are the minimum and maximum heights of the design domain. $h^e_{\min}$ and $h^e_{\max}$ are the minimum and maximum heights of each element, as described in Fig (3.6). The differentials $\frac{\mathrm{d}f_l(\mathbf{L})}{\mathrm{d}\mathbf{L}}$ and $\frac{\mathrm{d}f_c(\mathbf{C})}{\mathrm{d}\mathbf{C}}$ were calculated, based on the chain rule of differentiation respectively as,

$$\begin{aligned}\frac{\mathrm{d}f_l(\mathbf{L})}{\mathrm{d}\mathbf{L}} &= \frac{\mathrm{d}f_l(\mathbf{L})}{\mathrm{d}L^p_k} = \frac{\partial\phi(H^e_i)}{\partial H^e_i}\frac{\partial H^e_i(L^e_i)}{\partial L^e_i}\frac{\mathrm{d}L^e_i}{\mathrm{d}L^p_k} \qquad \text{(refer Eq. (3.8))} \\ \frac{\mathrm{d}f_c(\mathbf{C})}{\mathrm{d}\mathbf{C}} &= \frac{\mathrm{d}f_c(\mathbf{C})}{\mathrm{d}C^p_k} = -\frac{\partial\phi(B^e_i)}{\partial B^e_i}\frac{\partial B^e_i(C^e_i)}{\partial C^e_i}\frac{\mathrm{d}C^e_i}{\mathrm{d}C^p_k}\end{aligned} \tag{4.9}$$

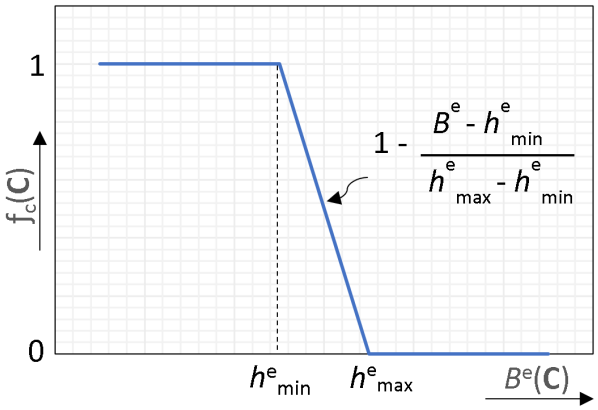
Each of these terms were further simplified as,

$$\begin{aligned}\frac{\partial\phi(H^e_i)}{\partial H^e_i} &= \frac{1}{h^e_{\max}-h^e_{\min}} \;, \quad \frac{\partial H^e_i(L^e_i)}{\partial L^e_i} = H^e_{\max}-H^e_{\min} \;, \quad \frac{\mathrm{d}L^e_i}{\mathrm{d}L^p_k} = w \\ \frac{\partial\phi(B^e_i)}{\partial B^e_i} &= \frac{1}{h^e_{\max}-h^e_{\min}} \;, \quad \frac{\partial B^e_i(C^e_i)}{\partial C^e_i} = H^e_{\max}-H^e_{\min} \;, \quad \frac{\mathrm{d}C^e_i}{\mathrm{d}C^p_k} = w\end{aligned} \tag{4.10}$$

Here, 'w' takes the same expression as Eq. (4.8).

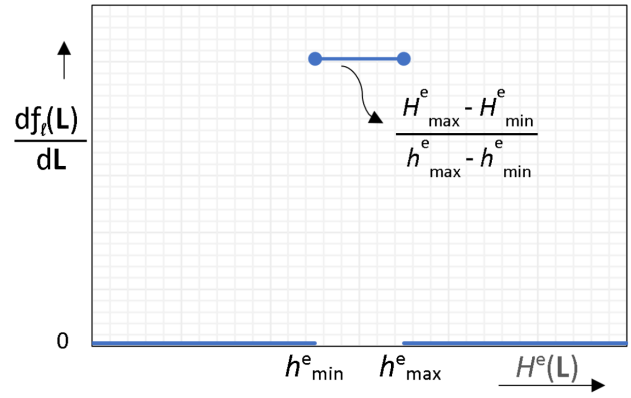


(a) $f_l(\mathbf{L})$ *vs* $H^e(\mathbf{L})$

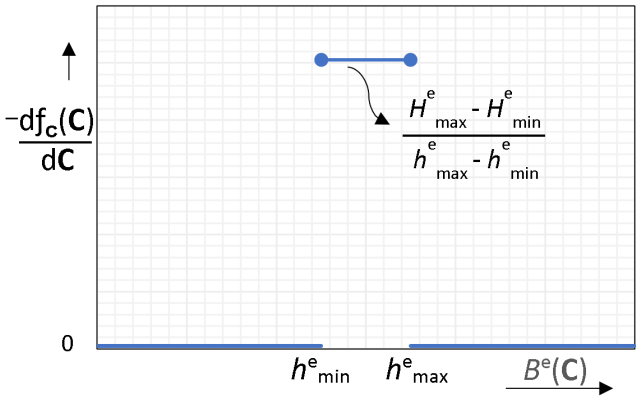


(b) $f_c(\mathbf{C})$ *vs* $B^e(\mathbf{C})$

Figure 4.3: Function plot



(a) $\frac{df_l(\mathbf{L})}{d\mathbf{L}}$ *vs* $H^e(\mathbf{L})$



(b) $\frac{df_c(\mathbf{C})}{d\mathbf{C}}$ *vs* $B^e(\mathbf{C})$

Figure 4.4: Differentiation plot

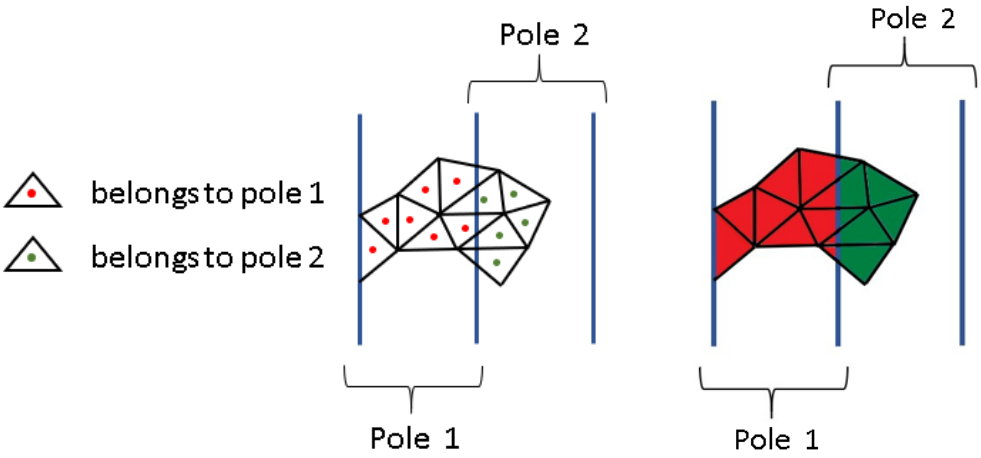
Assuming the filtering of poles is not used, the Gaussian weight matrix w_{jk} in the Eq. (4.8) becomes a unit matrix. Then the plot for the functions $f_l(\mathbf{L})$ and $f_c(\mathbf{C})$ can then be represented as shown in Fig (4.3) and their corresponding differentiation plots in the Fig (4.4).

Here, $H^e(\mathbf{L})$ and $B^e(\mathbf{C})$ represent the actual material growth and the actual material carved (refer Fig (3.6)). Note that the differentiation plot for function $f_c(\mathbf{C})$ already has negative term into the y coordinate. This is particularly done to make the differentiation plots for both $f_l(\mathbf{L})$ and $f_c(\mathbf{C})$ consistent.

4.5 Poles based on volume fraction of elements

The elements that form as one single pole are identified based on the location of centroids falling into the respective poles. This is clearly depicted in the Fig (4.5a). By this method, the elements that have their centroids in one pole but partially cutting through the boundary between two poles obtain a density value based on the parameter value of the pole in which the element centroid is identified.

An alternative method is proposed, wherein the density of each element is calculated based on the volume fractions of that element in each pole (refer Fig (4.5b)). By doing so, the result of the optimization is expected to have a clear segregation of the solid and void regions. In other words, this volume fraction method will prevent some potential undercuts, interior voids or artifacts that might appear in the centroid method. Both the mapping functions and the sensitivities described in section 4.3 and 4.4 respectively, will be modified based on the element volume fractions. This will be discussed in the following section.



(a) *Poles based on element centroids*

(b) *Poles based on element volume fractions*

Figure 4.5

4.6 Mapping using volume fractions

By introducing volume fractions of elements into each pole, the mapping and sensitivities are altered based on the calculated volume fractions. The function $f_a(\mathbf{A})$ in the mapping function 'F' (see Eq. (4.5)) is defined using the volume fraction as,

$$f_a(\mathbf{A}) = V_{i,k}^{frac} \sum_{j=1}^{NG_i} \frac{1}{NG_i} (\sum_{k=1}^{NP} w_{jk}(x_j, y_j) A_k^p) \tag{4.11}$$

Here, $\mathbf{A}$ is the pole parameter and w_{jk} is the weight matrix for filtering. The difference of the function $f_a(\mathbf{A})$ in comparison to the function $f_a(\mathbf{A})$ defined using centroids in section 4.3, is the introduction of the volume fractions. $V_{i,k}^{frac}$ represents the volume fraction of element i in the kth pole. The volume fractions of the finite elements are stored as a matrix format in *SIMULIA Tosca Structure*. The rows of this matrix are the element indices of those elements with cast restriction and the column consists of pole indices. The functions $f_l(\mathbf{L})$ and $f_c(\mathbf{C})$

are defined in the same way as discussed in section 4.3. Moreover, the sensitivities from the Eq. (4.7) is modified as,

$$\frac{\mathrm{d}f_a(\mathbf{A})}{\mathrm{d}\mathbf{A}} = \frac{\mathrm{d}f_a(\mathbf{A})}{\mathrm{d}A_k^p} = V_{i,k}^{frac} \cdot w \quad \text{where}, \, w = \sum_{j=1}^{NG_i} \frac{1}{NG_i} (\sum_{k=1}^{NP} w_{jk}(x_j, y_j)) \tag{4.12}$$

If there is no filtering of poles, the above differentiation will just be equal to the volume fraction matrix. The results of the differentiation were validated with the results from Finite Difference Method (refer Appendix 7.1). The sensitivities for the parameters **L** and **C** also change through the function $f_a(\mathbf{A})$, that uses volume fraction (refer Eq. (4.7)). In order to calculate the volume fraction of elements, element splitting and plane-edge cutting algorithm is used. The subsequent sections will cover the discussion of these concepts. However, it is necessary to initially discuss the formation of a pole to understand how the planes that make up a pole can be employed in the cutting algorithm.

4.7 Formation of cuboid poles

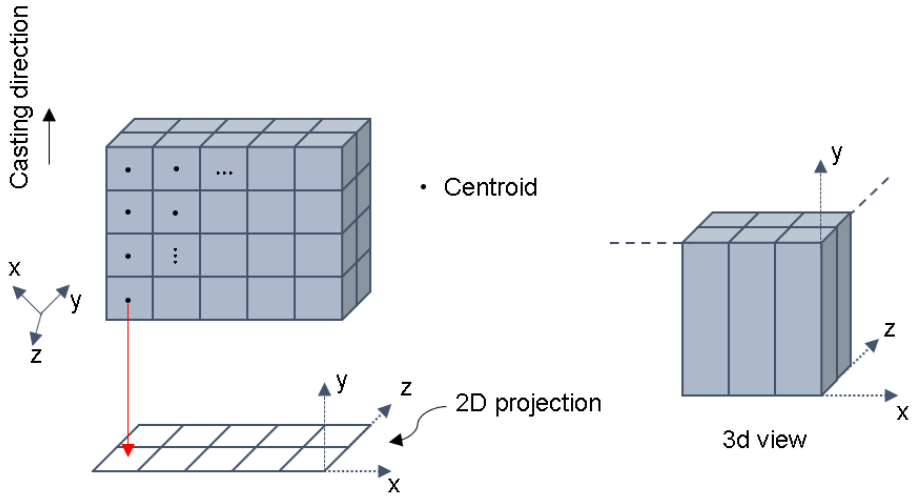


Figure 4.6: Formation of poles

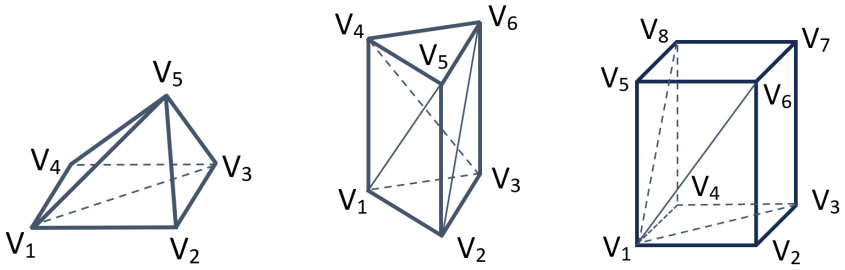
Fig (4.6) represents the formation of cuboid poles (or 2d grids) from a given finite element model. Inorder to form the grids, one of the global coordinate axis will be rotated such that it points along the given casting direction. In the figure, y-axis of the grids, points along the casting direction. After the coordinate rotation, the boundary for the grids will be formed based on the minimum and maximum nodal coordinate of the model. The element centroids will be projected to the projection plane. Based on the location of the centroids that fall into each grid, the elements will be identified into the respective grid. Moreover, the element centroids of each grid are sorted based on the coordinate axis, that points along the casting

direction (Here, y-axis). The grids can be imagined to be a cuboid in a 3d view (refer Fig (4.6)), where each cuboid has 6 planes, referring to a single pole. The planes of these poles will be used in the cutting algorithm, that calculates the volume fraction of finite elements in each pole.

4.8 Calculation of volume fractions

This section will discuss the element splitting and cutting algorithms to calculate the volume fractions of elements in each pole.

4.8.1 Element splitting



(a) *Pyramid splitting* **(b)** *Prism splitting* **(c)** *Hexahedron splitting*

Figure 4.7: Solid finite elements splitting into tetrahedrons (Dompierre u. a. 1999)

Other available solid finite elements in addition to tetrahedron elements are hexahedron, prism or wedge elements, pyramid elements and some degenerated hexahedron, prism and pyramid elements. Degenerated elements are those elements, that are formed by collapsing an element by one or more edges. For example: any 4 nodes of a hexahedron element can be merged to a single node, such that it results in a pyramid element. The application of degenerated elements is beyond the scope of this work, and hence, are not discussed in detail. The basic

Table 4.1: Sequence of vertices forming tetrahedron elements

Element type	Sequence of vertices
Pyramid element	[V1,V2,V3,V5], [V1,V3,V4,V5]
Prism element	[V1,V2,V3,V6], [V1,V2,V6,V5], [V1,V5,V6,V4]
Hexahedron element	[V1,V2,V3,V6], [V1,V3,V8,V6], [V1,V3,V4,V8], [V1,V6,V8,V5], [V3,V8,V6,V7]

concept of element splitting is that all these solid element types can be split fundamentally into multiple number of tetrahedrons. For example, a hexahedron element is split to 5 tetrahedrons, a prism element to 3 tetrahedrons, etc. A given element is split into tetrahedrons

based on their outermost nodes. A pyramid, prism and hexahedron element splitting is shown in Fig. (4.7). The sequence of vertices, that should be connected to split them to 2, 3, 5 number of tetrahedrons respectively, is shown in the Table 4.1. Here, $[V1, V2, ...]$ represent the outermost node numberings of the elements.

Fig (4.8) shows a 3d view of a single hexahedron element splitting into 5 tetrahedrons. Each split tetrahedron is illustrated with different colors to distinguish them from the original hexahedron element. For the degenerated elements, a similar concept of element splitting works except for different vertice numberings, in comparison to other solid element types. After element splitting, each split tetrahedrons will be further divided to sub-tetrahedrons, based on the cutting planes. The cutting planes are those planes that form a pole, as discussed earlier. The following section will discuss the calculation of volume fractions using each split tetrahedron and the cutting planes.

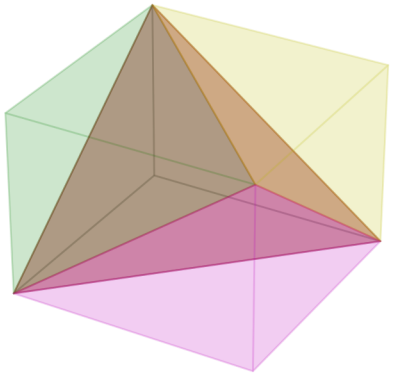


Figure 4.8: Hexahedron splitting into tetrahedrons

4.8.2 Tetrahedral cutting method

Each split tetrahedron from the element splitting or the tetrahedron elements from the original finite element model undergoes a recursive split into sub-tetrahedrons, based on the cutting planes. The cutting process for the tetrahedrons can be explained in the following way.

- At first, the planes that intersect the element bounding box are identified.
- Cut a plane with the element edges to obtain the intersecting points on the plane.
- Based on the number of element vertices on each side of the plane, a lookup table (refer Schweiger und Arridge (2016)) is used to obtain new tetrahedrons based on the intersection points and the edge points.
- These new tetrahedrons are iteratively checked if they are cut by any planes and further divided into new tetrahedrons.
- This process is repeated until all the divided tetrahedrons are not cut by any planes.

- The volume of all the sub-tetrahedrons identified in respective poles, is cumulatively added and divided with the total volume of an element to obtain the volume fractions of an element in corresponding poles.

Below is one example of a tetrahedron cut by a plane. Based on the number of edge points on the left and right side of the plane and the intersection points, it is split into sub-tetrahedrons with the help of look up table mentioned before. In this current example, one sub-tetrahedron is identified on the right side of the plane and a prism on the left side of the plane. The prism is further split into multiple tetrahedrons, for the purpose of calculating the volume fractions inside the poles.

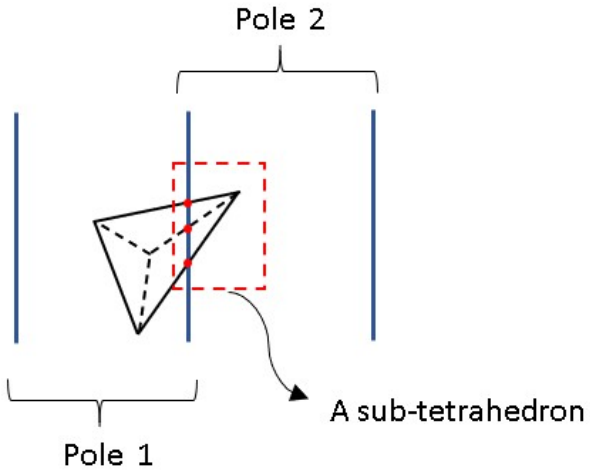


Figure 4.9: Tetrahedral cutting

The calculated volume fractions are stored as a matrix with rows defining the element number and the column with pole numbers. However, there is a drawback of storing the volume fraction values using such a matrix. If the number of finite elements in a model is sufficiently large, say 1 million elements, and the number of poles formed is 100,000 (e.g, 1/10th of total elements), the storage for the matrix becomes a problem. In addition, the computation of sensitivities is expected to be higher due to the large number of entries in the volume fraction matrix. Nevertheless, this problem can be prevented by storing the matrix as a sparse format. The formation of sparse matrix and the performance gain due to the sparse implementation will be dealt in the next section. Fig (4.10) represents a pseudocode for calculating the element volume fractions. As a brief description, the algorithm begins by looping through all elements, that have casting constraints. Then, it identifies the element type. Based on the finite element type, the corresponding splitting algorithm is used to divide them into multiple tetrahedrons. It is worth noting that if the model consists of tetrahedron finite elements, then splitting does not happen. After the element splitting, each tetrahedron undergoes the cutting procedure to further divide them, and finally, the element fractions are calculated cumulatively in each pole. A special care is also taken in the code, to neglect the poles that do not have any element

```
for i in elements                                  \\ loop through each casting element

   if ( element->type is hex/prism/pyramid )       \\ identify the element type

       call splitting_algorithm                    \\ call the corresponding splitting
                                                   \\ function based on the element type.
                                                   \\ Returns the vertices of all split
                                                   \\ tetrahedrons

   for j in split_tetrahedron                      \\ loop through each split tetrahedron

       {

         \\ perform the cutting algorithm

         \\ store the volume fractions
         \\ of elements in each pole

       }
```

Figure 4.10: Pseudocode for calculating volume fractions

fractions inside it. This helps to avoid initializing unnecessary parameter values for the poles with no element fractions.

4.8.3 Performance and memory improvement

The pole size is usually preferred to be less than the average element edge length. Increasing the pole size to a larger value, may lead to inaccuracy and results in more intermediate density elements. On the other hand, decreasing the pole size to a very minimum value, will result in bad performance. A study on the pole size using some results, will be explained in detail in the next chapter.

For a pole size less than average edge length, the fractions of one element are expected to be present only in very few poles. As a result, the volume fraction contribution of that element in the remaining poles is null. If the volume fractions are stored in a matrix with element numbers in rows and pole numbers in columns, there will be a lot of null entries in the matrix. Instead of storing the whole dense matrix, a sparse format is implemented to store only the non-zero entries of the matrix.

A simple and clear definition for the sparsity of a matrix is given mathematically by Scott und Tůma (2023) as,

$$S\left\{\boldsymbol{V}^{frac}\right\} = \left\{(i,j) \mid v_{ij}^{frac} \neq 0, 1 \leq i \leq n_e \text{ and } 1 \leq j \leq n_p\right\} \tag{4.13}$$

Here, $\boldsymbol{V}^{frac}$ is the volume fraction matrix. v_{ij}^{frac} represents each entry in the matrix. n_e and n_p are the number of elements and the number of poles respectively. $S\left\{\boldsymbol{V}^{frac}\right\}$ means the sparsity pattern of the volume fraction matrix. There are various methods for storing a sparse data such as Compressed row and column storage, Diagonal storage, Skyline storage etc. (re-

fer Saad (1990)). In this work, a Compressed Storage Format (CSR) is implemented. An example of the CSR implementation is given in Barrett u. a. (1996).

CSR format stores consecutive nonzeros of matrix rows in contiguous memory locations. For the nonsymmetric matrix $\boldsymbol{V}^{frac}$, following 3 vectors will be required: one for storing the volume fraction values as floating-point numbers (`frac_vals`) and other two for integers (`col_id`, `row_ptr`). The `frac_vals` vector stores the values of nonzero entries from the matrix $\boldsymbol{V}^{frac}$, by traversing in a row-wise fashion. The `col_id` vector stores the pole indices of the entries in the `frac_vals` vector, i.e., if `frac_vals`$(k) = v_{ij}^{frac}$, then `col_id` $= j$. The `row_ptr` vector stores the locations in the `frac_vals`(k) vector that starts a row, i.e., if `frac_vals`(k) $= v_{ij}^{frac}$, then `row_ptr`$(i) \leq k <$ `row_ptr`$(i+1)$. Below Fig (4.11) shows a pseudocode for storing the volume fraction values in the sparse format.

```
for i in elements                          \\ loop through casting elements
    call split_elements                    \\ split the elements to
                                              tetrahedrons
    num_nz = 0, row_ptr(1)= 1              \\ initialize number of non-zero entries
                                              and row pointer for the first element
    for j in split_tetrahedrons
     {
       call calc_vol_frac()                \\ calculate the volume fractions of the
                                              element in each pole
       num_nz = num_nz + 1                 \\ increase the num_nz if a non-zero entry
                                              is identified
       frac_vals(num_nz) = vol_frac        \\ store the non-zero entry
       col_id(num_nz) = pole_index         \\ store the pole index in the column array
     }
     row_ptr(i+1) = num_nz+1               \\ store the row pointer for next element
```

Figure 4.11: Pseudocode for CSR storage

The matrix multiplication that is necessary during the sensitivity computation, is carried out with the help of the three variables, `frac_vals`, `col_id` and `row_ptr`. The number of loops for the multiplication depends on the row pointer, which gives the information on the number of non-zeros per row. Fig (4.12) shows, how a matrix multiplication with a vector is performed (refer Liu und Vinter (2015) for an example). The advantage of using sparse storage, especially during matrix-vector multiplication, is that it considers only the non-zero volume fraction entries. This stands in contrast to the dense matrix, which considers all entries in the matrix for the multiplication. This is a major limitation of using a dense format. Therefore, the matrix-vector operation, that is used both in the mapping function and computation of sensitivities are optimized with sparse storage. It is also important to note that the volume fractions values are stored directly in a sparse format, while it is being calculated. This helps to gain better performance than storing as dense matrix and then converting to sparse format.

The advantages in terms of performance and efficient memory utilization became evident upon implementing the sparse storage format. The upcoming chapter will quantitatively compare the performance and memory usage between the dense matrix implementation and the sparse implementation using a specific example.

```
for i in elements                          \\ loop through casting elements
   for j = row_ptr(i) to row_ptr(i+1) \\ loop through the non-zero entries per row
      \\ do the multiplication of
         sparse matrix with a vector
      result(i) = frac_vals(j) * vector(col_id(j))
```

Figure 4.12: CSR multiplication with vector

4.9 Overall code implementation

Having discussed the reformulated mapping function and the sensitivities based on volume fractions, this section gives an overview of the code implemented in this thesis for the casting constraint parametrization.

The casting constraint based on parametrization in the current *SIMULIA Tosca Structure* uses a different grid (see Fig. (3.4)) in comparison to the square grids (see Fig. (4.6)) in this work. Using square grids offers an advantage because the planes within each grid can be used as cutting planes, simplifying the calculation of element volume fractions within each grid.

Fig. (4.13) provides a overall idea of the code implemented in this work. Many additional details have been included between each major step depicted in the figure. However, understanding the major steps outlined in the code suffices. The overview of the code can be explained in the following way.

Starting with a given finite element model and with the specified casting direction, cuboid poles are formed that points along the casting direction. To achieve this, one of the global coordinate axis is rotated with the help of a rotation matrix, to point it along the casting direction. Then, the cast parametrization can be done in any of the following two methods.

Poles based on centroids: This method follows the same approach as the existing code in *SIMULIA Tosca Structure*, where the elements are identified into the poles based on the location of their centroids. However, the difference is the configuration of the poles used. After identifying the elements to corresponding poles, the pole parameters and sensitivities are initialized. These are given as input to the optimizer. After the optimizer provides the optimized parameter values based on a gradient descent algorithm, the parameters are mapped to corresponding element relative densities.

Poles based on volume fraction: In this method, parametrization is based on the volume fractions of elements in each pole. This method, as discussed before, is expected to provide

results that contain clear segregation of the solid and void regions. First, the volume fractions are calculated using the element split and cutting algorithm. Similar to previous method, parameters and sensitivities are initialized. However, the difference is that the sensitivities are calculated based on the mapping function, that uses the volume fractions (see Eq. (4.11) and Eq. (4.12)). Finally the optimized parameter values obtained from the optimizer are mapped to element densities based on the fractions of elements present in each pole.

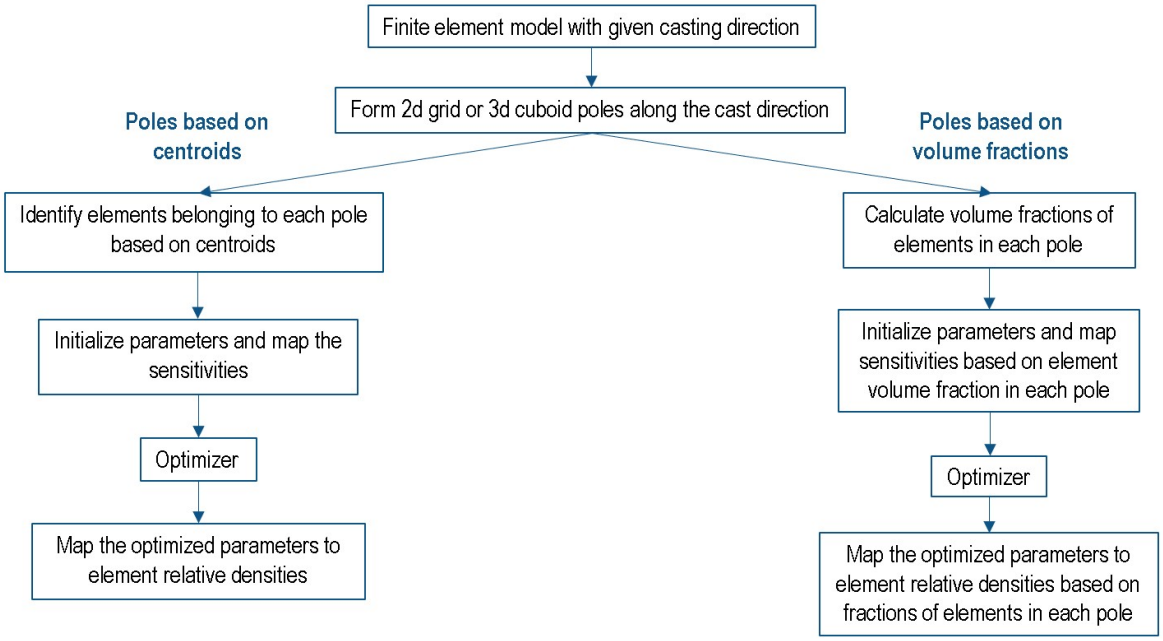


Figure 4.13: Overall code implementation

4.10 Summary

In this chapter, the reformulated mapping functions and sensitivities based on the volume fractions of elements in each pole, were discussed. The use of sparse format to store the volume fraction values were discussed. The performance gain and the memory efficiency for this sparse storage will be dealt with quantitative results in the next chapter. The Fig. (4.14) and (4.15) shows all three methods for cast parametrization. The first is currently available method in *SIMULIA Tosca Structure*, where the centroids are projected to a circular grid, same as in Fig. (3.4) and the second method (Cuboid centroids in Fig. 4.14) is similar to first method that it works on the centroids, but the poles are formed based on the grids shown in Fig. (4.6). Using the same cuboid poles, the mapping function and parameter sensitivities are reformulated for the third method, based on the volume fractions of elements (see Fig. (4.15)). Here, $\mathbf{P}$, $\boldsymbol{\rho}$, V_f and ϕ are the parameters, element densities, volume fractions and design response respectively. The following chapter will discuss some results comparing these three methods.

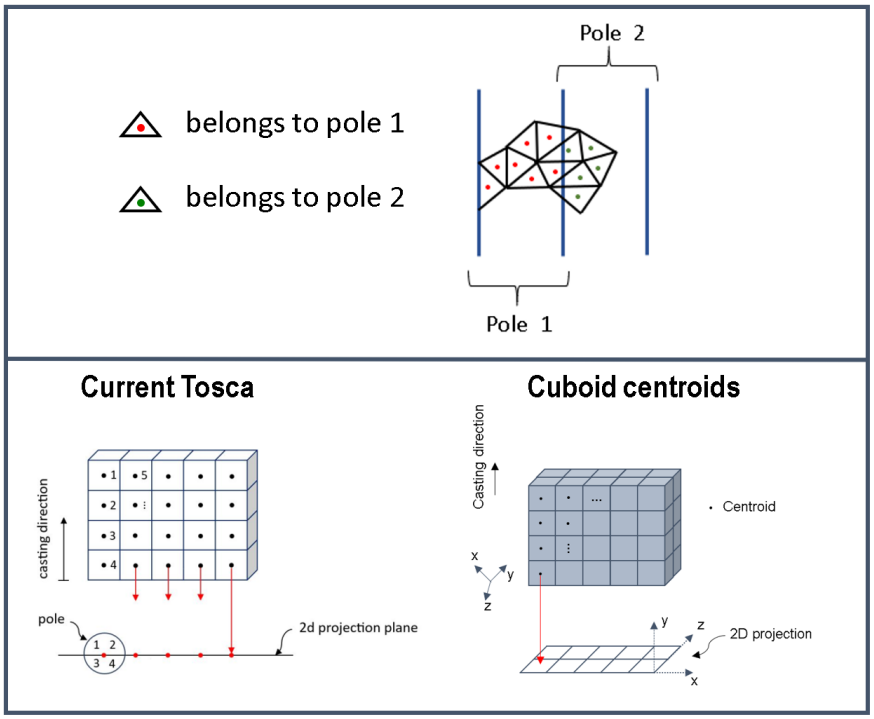


Figure 4.14: Current *Tosca* and Cuboid centroid method

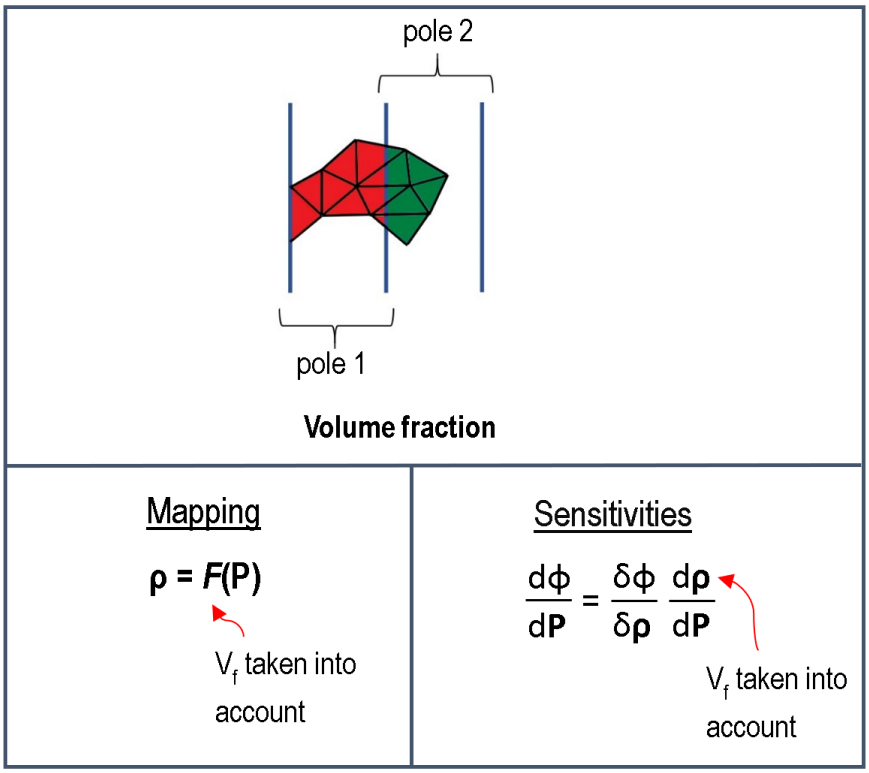


Figure 4.15: Volume fraction method

5

Industrial models

This chapter will discuss few industrial models and compare the results of the optimization from the current *SIMULIA Tosca Structure* with the methods implemented in this work.

5.1 Oil pan model

Fig. (5.1) illustrates the finite element model of oil pan. The oil pan model consists of approximately 100,000 linear tetrahedron elements and is fixed on the hexagonal indents (see Fig. (5.2a)), that are present on the four corners. Both the displacements and rotations are fixed in this region. Surface loads are applied to each element on the outer surfaces (see Fig. (5.2b)), excluding the front face. The optimization inputs for this model are minimizing the compliance (i.e. summation of strain energy of the elements) as objective and local volume constraint less than 50 percent. The local volume constraint means that the optimal structure will produce thin ribs for low constraint value and thick ribs for high constraint values. The surface of the oil pan, where forces are exerted, is excluded from the design space (called frozen region). This implies that its topology remains unchanged throughout the optimization. Finally, the casting constraint is applied for full design height in the direction perpendicular to the z-axis of the model. The full design height means that the bounds of the parameter C will be zero ($C_{min} = C_{max} = 0.0$) and the bounds for parameter L is 1.0 ($L_{min} = L_{max} = 1.0$).

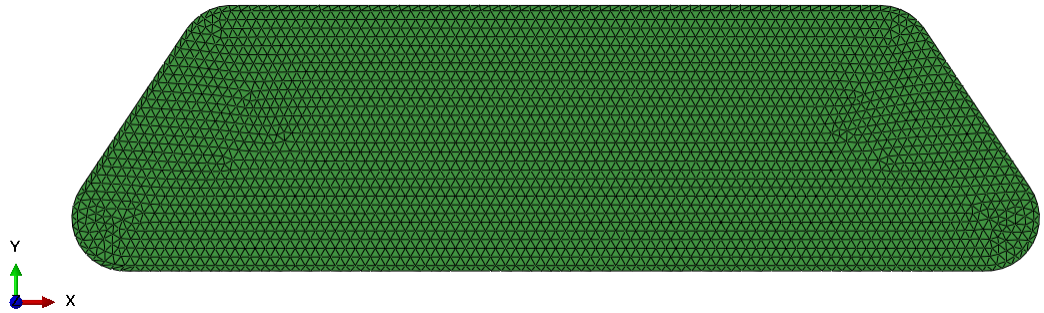
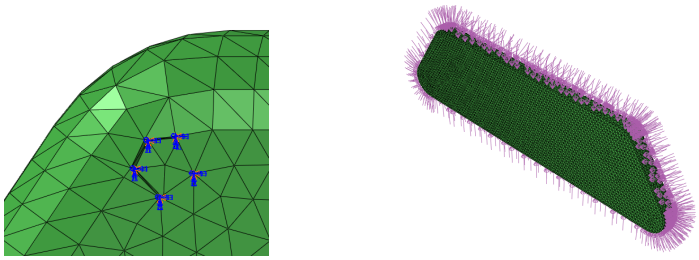


Figure 5.1: FE model of oil pan

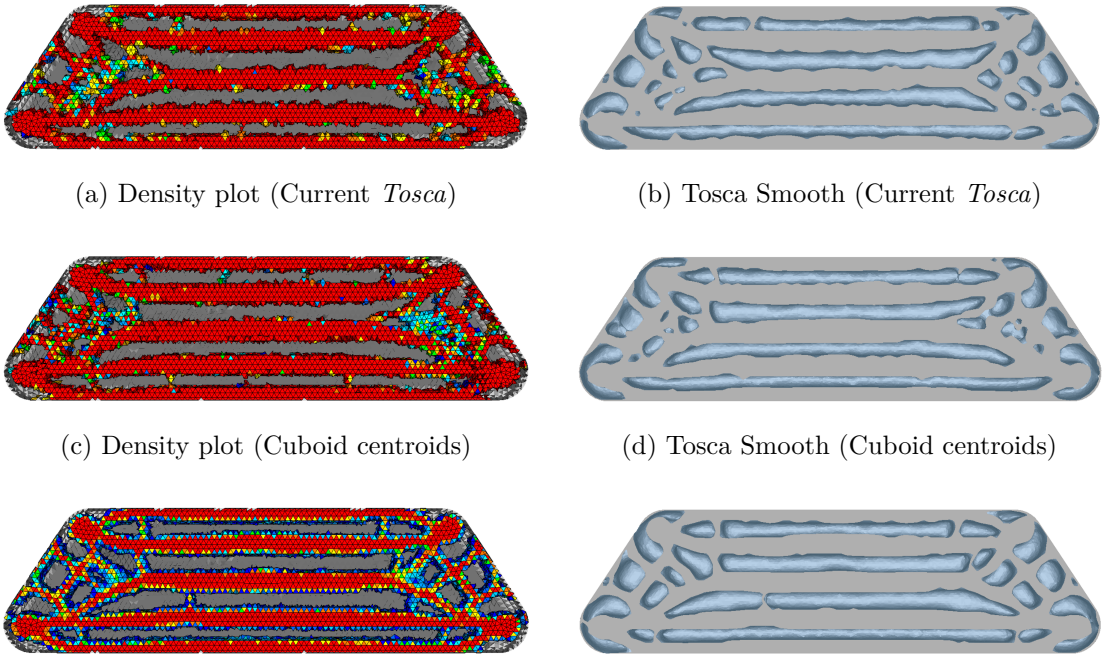


(a) *Boundary condition*

(b) *Loading condition*

Figure 5.2: Boundary and loading condition

The results of the optimization are compared for three methods, namely the Current *Tosca*, Cuboid centroids and Volume fraction. Fig. (5.3) shows the density plot distribution and the Tosca Smooth results. Note that the results from Tosca Smooth are obtained for an isovalue of 0.3. From the Fig. (5.3e), the density distribution for Volume fraction method is observed to



(a) Density plot (Current *Tosca*)

(b) Tosca Smooth (Current *Tosca*)

(c) Density plot (Cuboid centroids)

(d) Tosca Smooth (Cuboid centroids)

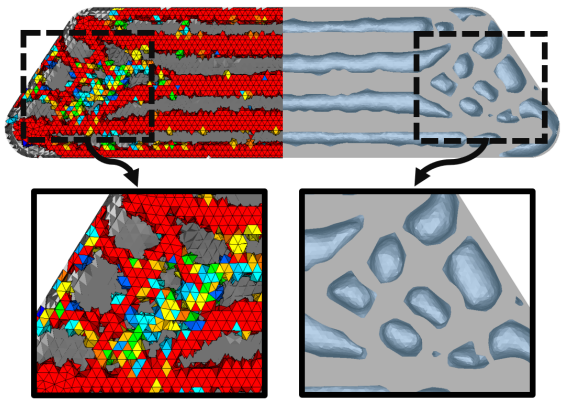
(e) Density plot (Volume fraction)

(f) Tosca Smooth (Volume fraction)

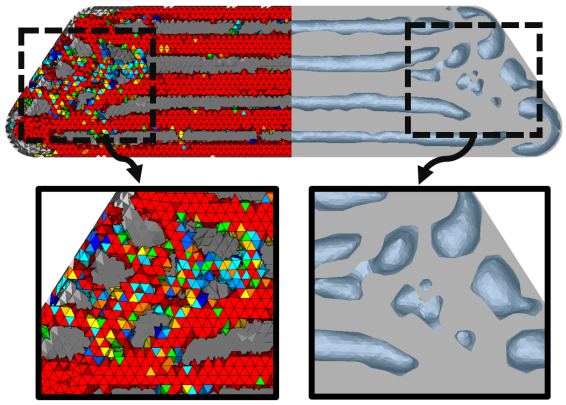
Figure 5.3: Optimization results - Density distribution and Tosca Smooth

have clearer results for the ribs and the voids in comparison to other two methods (Fig. 5.3a and Fig. 5.3c). This is especially visible near the left and right ends of the oil pan. Moreover, the Tosca smooth results also show better surfaces for the Volume fraction method near the ends of the model. This will be easy to comprehend after taking a closer look at the two ends of the oil pan (see Fig. (5.4)). The first half in the figure represents the density plot and the

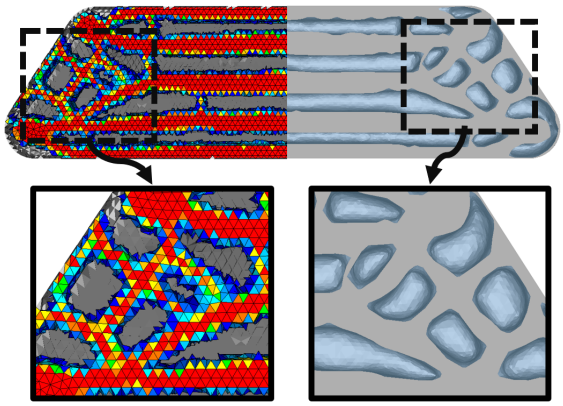
Tosca Smooth result in the second half. The results from Current *Tosca* and Cuboid centroids have more intermediate densities on the ribs and does not show clear distinction of the solid and void regions. This can lead to potential undercuts or some cavities inside the model. However, the results from Volume fraction method is comparatively better, showing a clear segregation of the solid and void regions, avoiding potential undercuts in these regions. The



(a) Current *Tosca*



(b) Cuboid centroids



(c) Volume fraction

Figure 5.4: Ends of the oilpan model

reason for the Volume fraction method to show better results can be explained in the following way : considering two poles on the boundary of the solid and void regions, pole 1 located in the solid region, receiving a pole density value close to 1.0 and the other pole receives a density value of 0.0, as it lies in the void region. In Current *Tosca* and Cuboid centroids, the elements are identified into respective poles, based on the location of their centroids. While mapping the pole densities to the element densities in the respective poles, the centroid based methods (both Current *Tosca* and Cuboid centroids) yield results that have elements with higher densities falling into the void regions. (Fig. (5.5)). In this figure, the elements depicted in red indicate those with full relative density value. The example illustrated in Fig. (5.5)

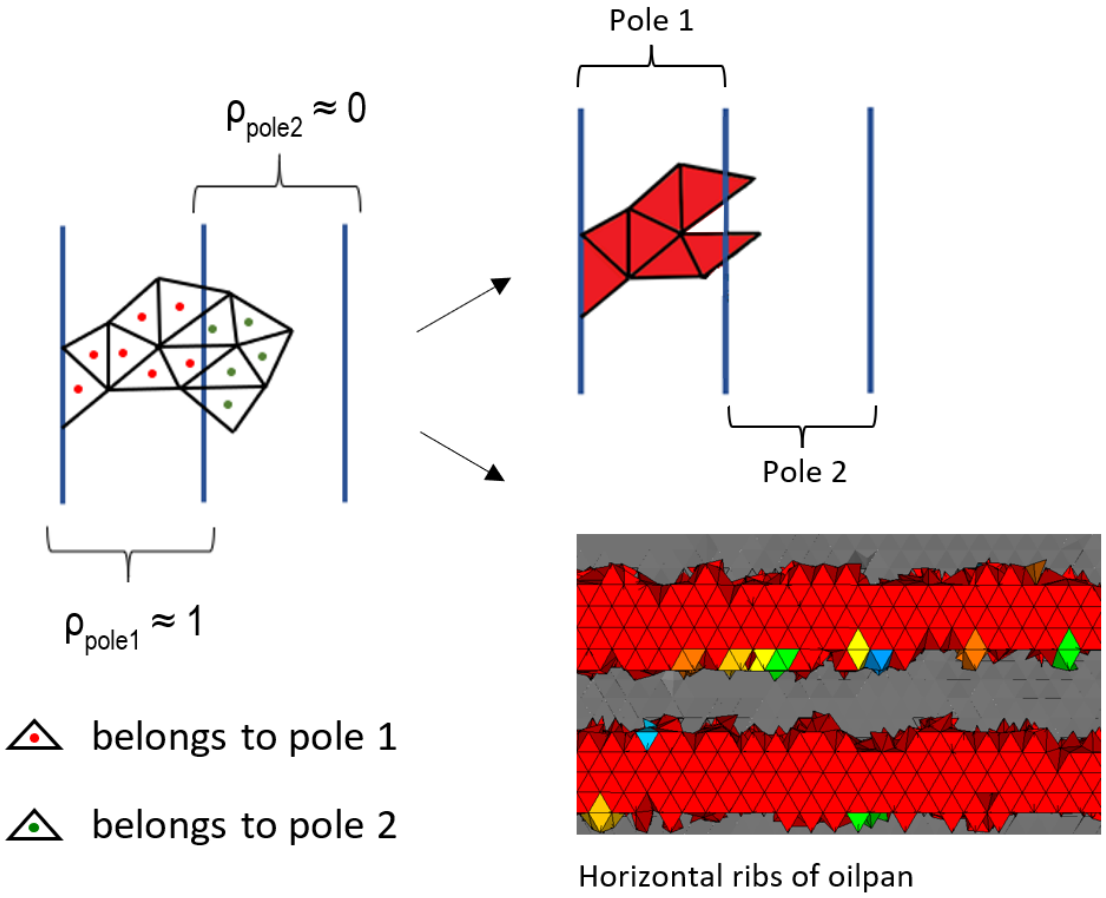


Figure 5.5: Centroid based mapping (both Current *Tosca* and Cuboid centroids)

examines only two poles in the solid and void boundary. This concept can be extended to all poles situated at the boundaries. As a result, the boundaries of the ribs may appear inconsistent due to the irregular pattern of high density elements in the void areas. This is observable in the density plot of the horizontal ribs in the oil pan model. The inconsistency in higher density values along the boundaries stands as a primary factor contributing to the formation of undercuts or cavities within the model. This consequently manifests as irregular surfaces in the results obtained from Tosca Smooth. If the mesh is coarse or consists of unstructured finite elements, there is a greater chance that the density plot of the model will display uneven high-density elements along the boundaries. This problem is relatively avoided in the Volume fraction method, because the elements that cut through the boundary of the poles receive an intermediate or a low density value (elements in blue, see Fig. (5.6)), showing a separation of the high density and empty elements. This gives a clear segregation of the region with and without material, especially along the boundaries of the ribs. As a result, it prevents potential undercuts or cavities within the model. The presence of these low or intermediate density elements in the boundary of the poles also help in better interpolation of the density field to form smoother surfaces. This is the reason for better surfaces from Tosca Smooth. It is also

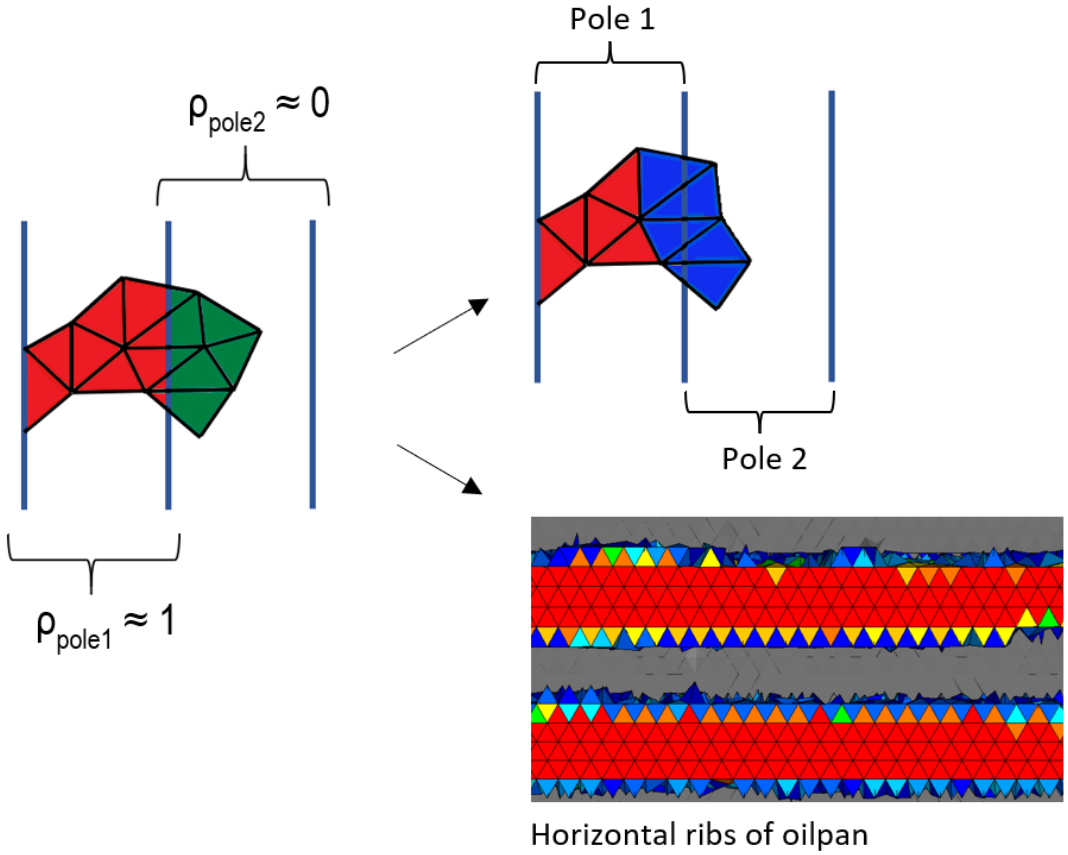
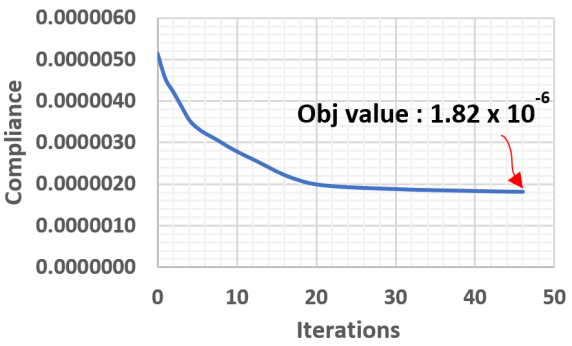
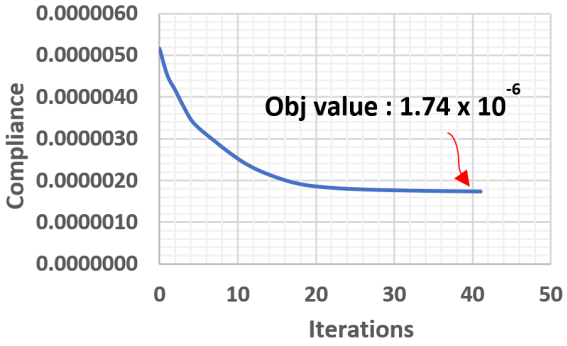


Figure 5.6: Volume fraction mapping

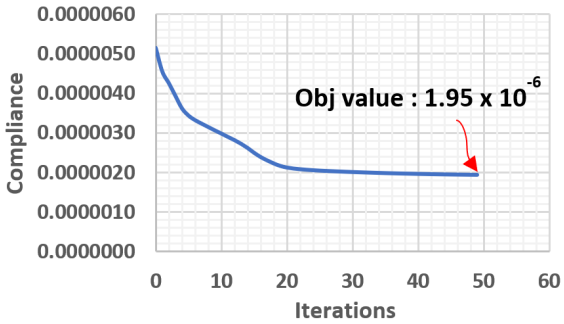
important to ascertain whether the optimization results satisfy the objective of minimizing compliance. The objective function values are plotted against the optimization iterations as shown in Fig. (5.7) for all three methods. In all the plots, the objective function values appear to decrease consistently with each iteration, showing no fluctuations in the graph. At the end of the optimization, the Cuboid centroids method is observed to have 4 percent less compliance value than Current *Tosca*, and the Volume fraction method has 7 percent more than Current *Tosca*. In all tested models with minimizing compliance as the objective, a consistent trend in compliance values emerges. At the end of optimization, the compliance value for the model using the Cuboid centroids method is nearly equal to or lower, in some instances, compared to the Current *Tosca*. This outcome is expected due to the identical formulation of mapping function and sensitivities in both methods. Conversely, compliance values obtained through the Volume fraction method consistently surpass those of the other two methods. However, the disparity in compliance values between the Volume fraction method and the centroids-based methods is not substantial, affirming the continued efficacy of the Volume fraction method. The reason for higher compliance value is due to the presence of intermediate or low density elements. This makes the entire structure less stiff. Since the compliance is inversely related to the stiffness of the structure, it results in higher compliance value at the end of optimization. In Current *Tosca* and Cuboid centroids, the number of high density elements is



(a) *Current Tosca*



(b) *Cuboid centroids*



(c) *Volume fraction*

Figure 5.7: Plot of compliance values and optimization iterations

more. Consequently, the whole structure becomes stiffer and less compliant at the end of the optimization.

5.2 Performance comparison

This section discusses the performance of the Volume fraction method, examining its implementation using both dense matrix and a sparse storage format. Furthermore, it evaluates the runtime performance by comparing all three methods.

5.2.1 Performance gain from sparse storage

The sparse storage format as discussed in section 4.8.3 is implemented and the gain in performance and memory usage is studied. The y-axis of the graph in Fig. (5.8) represent the optimization time in seconds per cycle. The optimization for oil pan model with Volume fraction method is run with three different number of finite elements in combination with three different CPUs, namely 1, 4 and 16. The time taken for the Volume fraction method using the

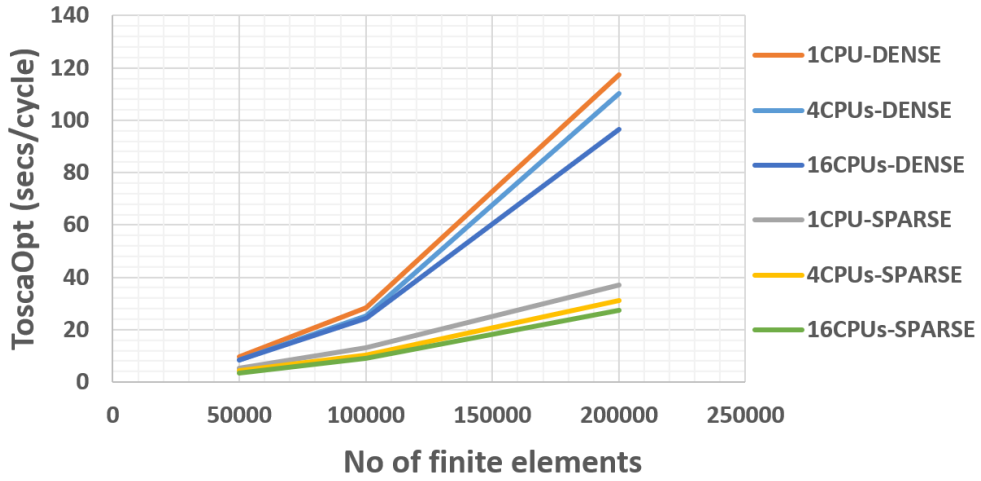


Figure 5.8: Performance plot for dense vs sparse storage

dense matrix storage is clearly more than three times the time taken for the Volume fraction method implemented with sparse storage. Further observation revealed a significant consumption of time in computing the sensitivities using the dense matrix. Table 5.1 shows the usage

Table 5.1: Memory usage for dense and sparse format

No of elements	Dense	Sparse
50,000	2GB	29MB
100,000	7GB	57MB
200,000	22GB	118MB

of memory for dense and sparse storage. Here, memory in GB and MB refers to gigabytes and megabytes respectively. The memory occupied by 200,000 finite elements is around 22GB for dense storage, compared to 118MB for sparse storage. This difference becomes more significant as the number of finite elements in the model increases. Henceforth, given its memory efficiency and superior performance, the Volume fraction method is implemented with sparse storage.

5.2.2 Performance comparison for three methods

Fig. (5.9) shows the performance comparison of the oil pan model for all three methods run with 16 CPUs. The optimization time in seconds per cycle is taken in y-axis and different number of finite elements in the x-axis. Note that the Volume fraction method is implemented with sparse storage, due to its efficient performance as discussed in the previous section. Comparing all three methods, Cuboid centroids performs much better than other two methods in all cases. This holds true for all tested models. Efficient formation of poles is the main reason for the less time, whereas the Current *Tosca* consumes a lot of time in forming poles and identifying elements to poles. Volume fraction method gains some performance, because it

uses the same grid as the Cuboid centroids, but the overhead on calculation of volume fraction values leads to significant difference with the Cuboid centroids. This is especially observed for the fine meshes. However, the Volume fraction method is comparable with the Current *Tosca* and even better in case with large number of finite elements. Hence, it can be inferred that

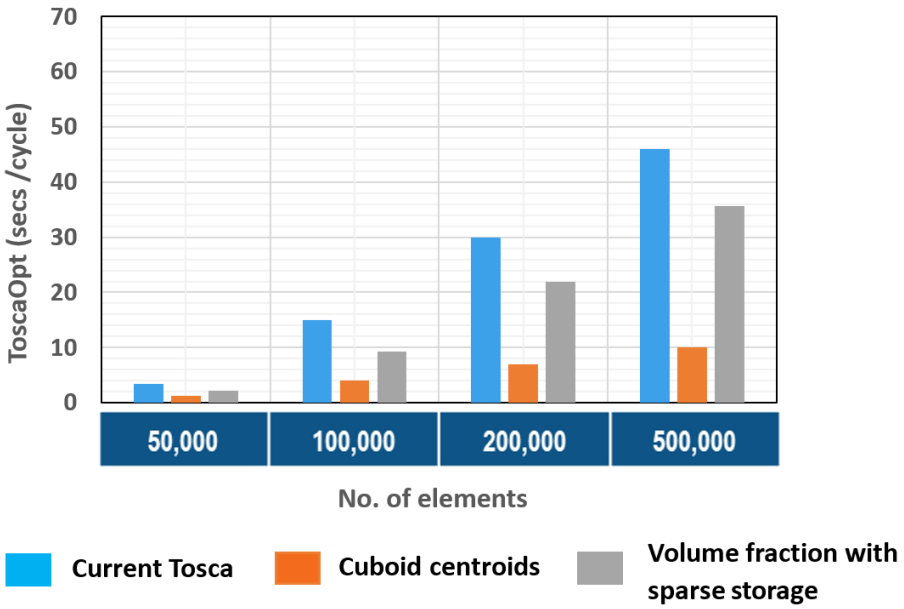


Figure 5.9: Performance plot for all three methods

the qualitative results from the Volume fraction method is not completely compensated by the runtime performance.

5.3 Steering box model

Fig. (5.10) shows the finite element model of the steering box. This model consists of approximately 130,000 linear tetrahedron finite elements. There are connector elements present on the four holes in the model for the bushing.

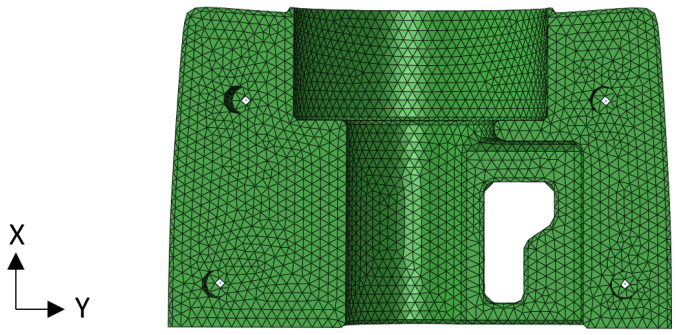


Figure 5.10: Steering box FE model

Fig. (5.11) provides a clearer visualization of the steering box model, presenting various perspectives. The model is fixed on the lower region (Fig. (5.12a)) and loaded with concentrated

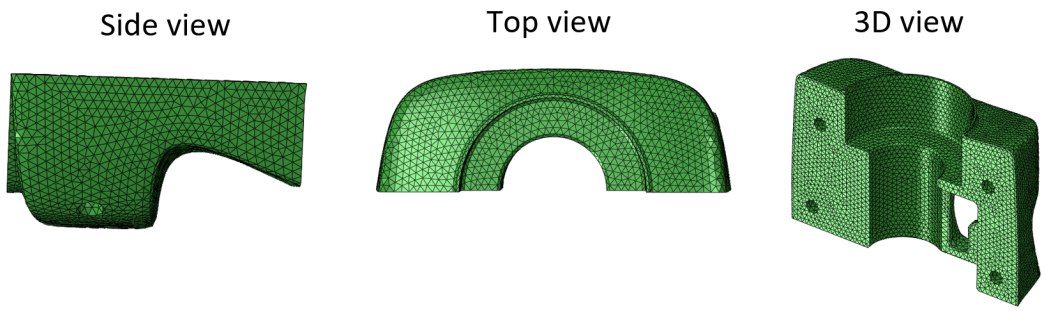
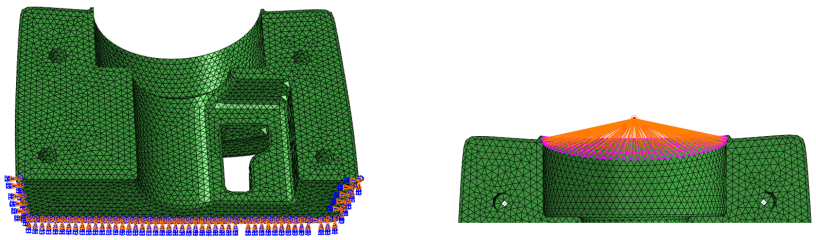


Figure 5.11: Steering box - different views

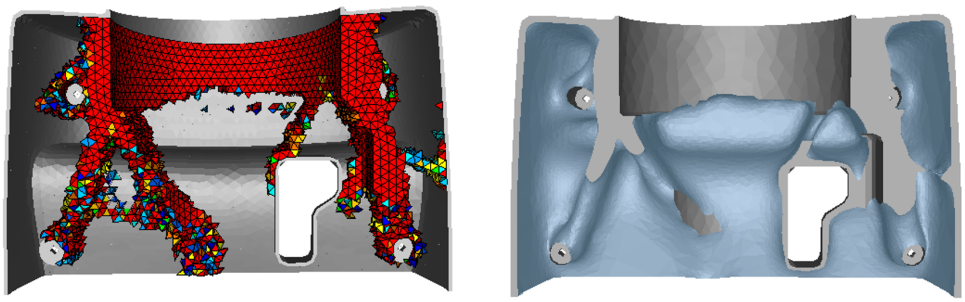
forces in x, y and z directions on the node, that has coupling constraint to the top circular face in the model (Fig. (5.12b)). Minimizing compliance and local volume of 70 percent



(a) *Boundary condition* **(b)** *Coupling constraint*

Figure 5.12: Inputs to FE model

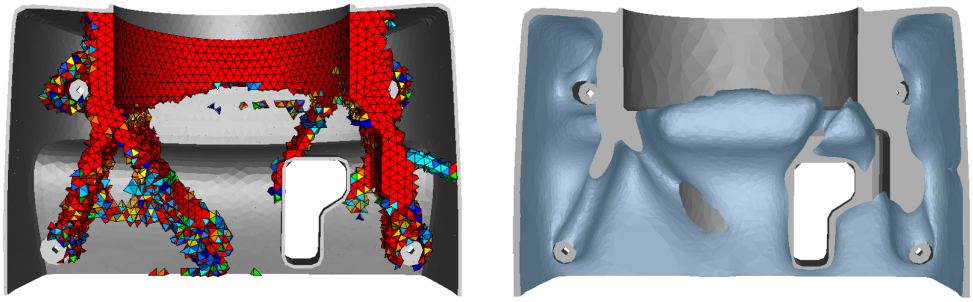
forms the objective function and constraint respectively. The casting direction is perpendicular to the front face and the casting constraint is applied with variable rib height. This means that the parameter bounds for C and the minimum bound for parameter L will be zero ($C_{min} = C_{max} = L_{min} = 0.0$), but the maximum value of the parameter L is set to be a variable value, decided by the optimizer. Fig. (5.13) and Fig. (5.14) shows the density plot and Tosca



(a) *Density plot* **(b)** *Tosca Smooth*

Figure 5.13: Current *Tosca*

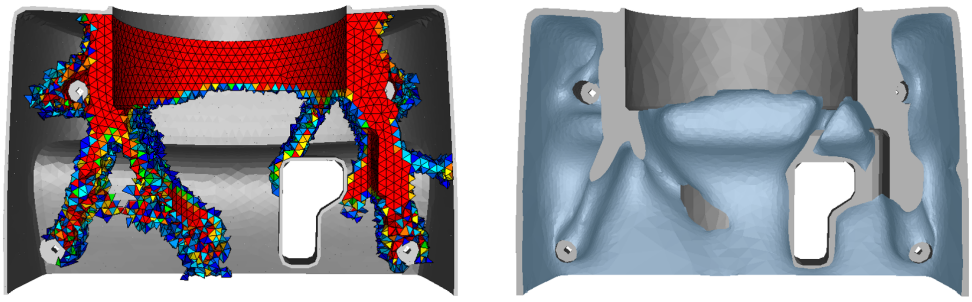
Smooth results for the Current *Tosca* and Cuboid centroids method respectively. Few elements are observed to be sticking to the frozen areas near the circular face of the model. This occurs in Current *Tosca* and Cuboid centroids, while these artifacts are removed in Volume fraction method (Fig. (5.15)). One possible explanation may stem from the manner in which



(a) *Density plot* **(b)** *Tosca Smooth*

Figure 5.14: Cuboid centroids

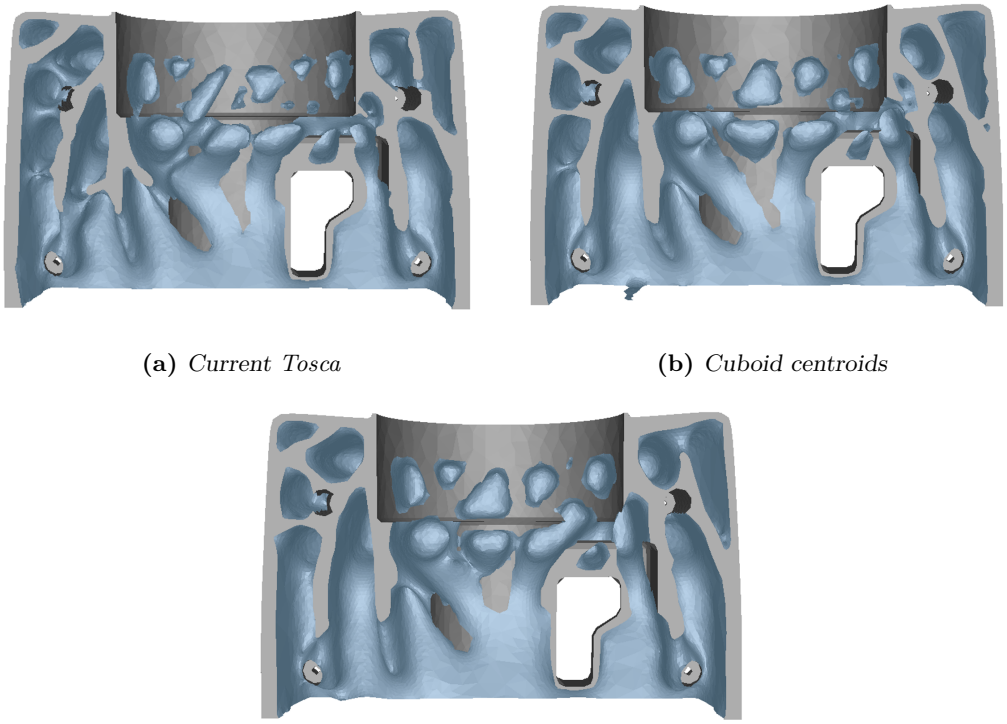
the optimizer evaluates pole parameter values in the Current *Tosca* and Cuboid centroids. It is plausible that, in these evaluations, the optimizer assigns non-zero density values to few poles located near the circular region within the model, in contrast to the zero density values assigned to poles in the Volume fraction method. Such optimized zero density values in the Volume fraction method could arise due to alterations in sensitivity formulation, which depend on the element volume fractions within a pole. However, there is no more improvements or



(a) Density plot (b) Tosca Smooth

Figure 5.15: Volume fraction

significant differences observed in Volume fraction over other methods. Volume fraction may show some significant differences, if the model is complex or has large number of solid and void regions. Hence, the local volume constraint is changed from 70% to 33%, such that the optimal structure has more ribs and voids (Fig. (5.16)). The results are then studied again with the reduced local volume. From the results obtained, the Volume fraction method showed better surfaces with clear holes in the model, compared to Current *Tosca* and Cuboid centroids method. In many locations within the optimized model, notable enhancements were observed

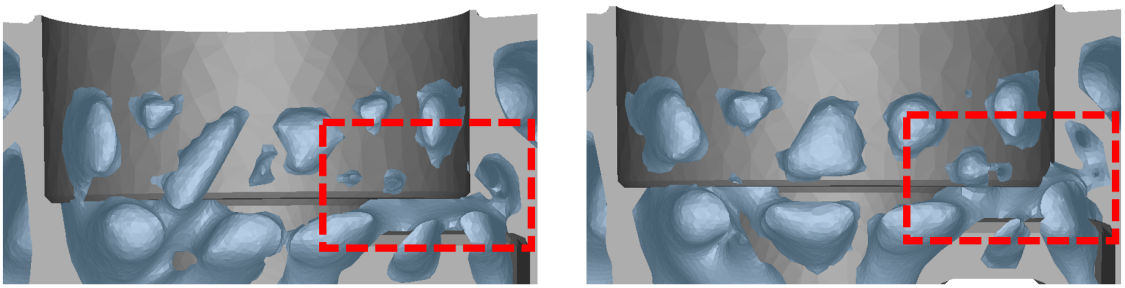


(a) *Current Tosca* **(b)** *Cuboid centroids*

(c) *Volume fraction*

Figure 5.16: Tosca Smooth results (33% Local volume)

when using the Volume fraction method. However, only one exemplary instance of such improvement is discussed herein. Taking a closer look at the top circular section of the model (see red boxes in Fig. (5.17) and Fig. (5.18)) shows significant differences. Interior cavities are formed in both Current *Tosca* and Cuboid centroids. This phenomenon arises from the irregular pattern of high-density elements along the boundaries of the ribs, reflecting the same observation made in the oil pan model. This is comparatively prevented in Volume fraction method, producing clear ribs and voids.



(a) *Current Tosca* **(b)** *Cuboid centroids*

Figure 5.17: Tosca Smooth results

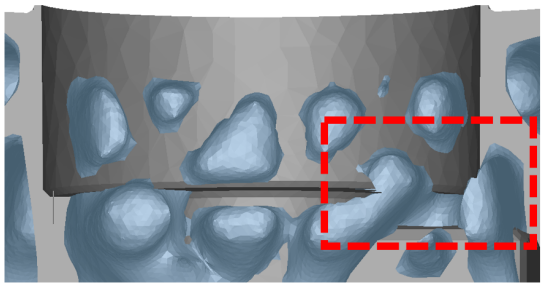


Figure 5.18: Tosca Smooth results for Volume fraction method

5.4 Wing model

Fig. (5.19) illustrates the finite element model of the wing. This model consists of approximately 2.5 million tetrahedron finite elements. It is pin supported at its root along with

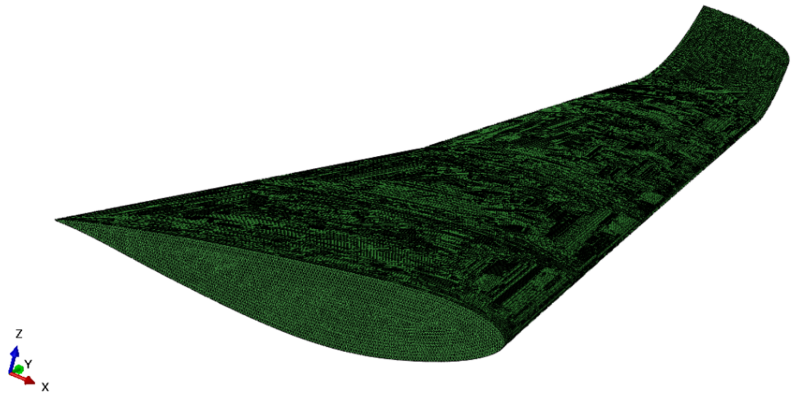
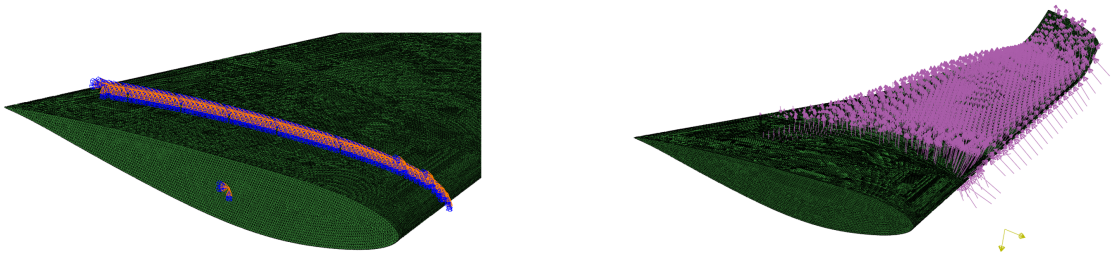


Figure 5.19: Wing FE model

clamping condition for some elements near its root (Fig (5.20)). Loads are applied on the top and bottom surfaces of the wing model. The inputs to the optimization are minimizing

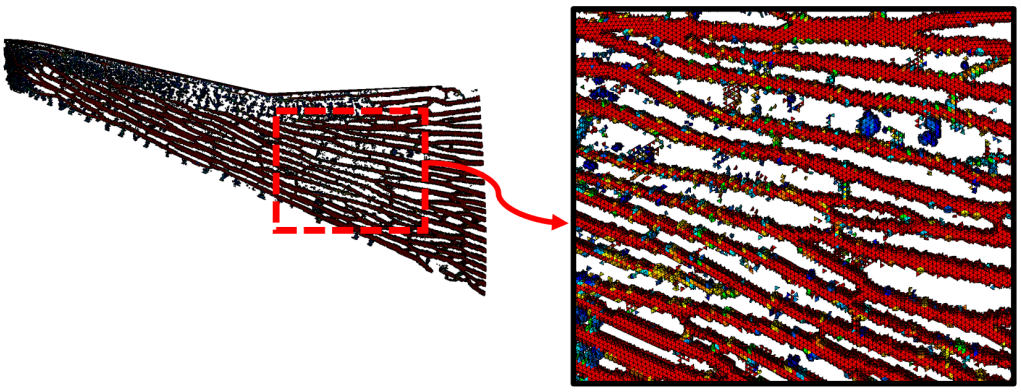


(a) Boundary condition

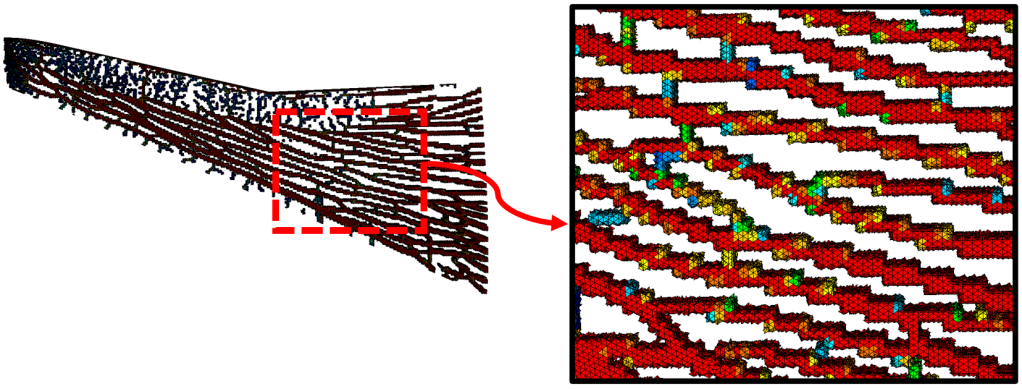
(b) Loading condition

Figure 5.20: Inputs to FE model

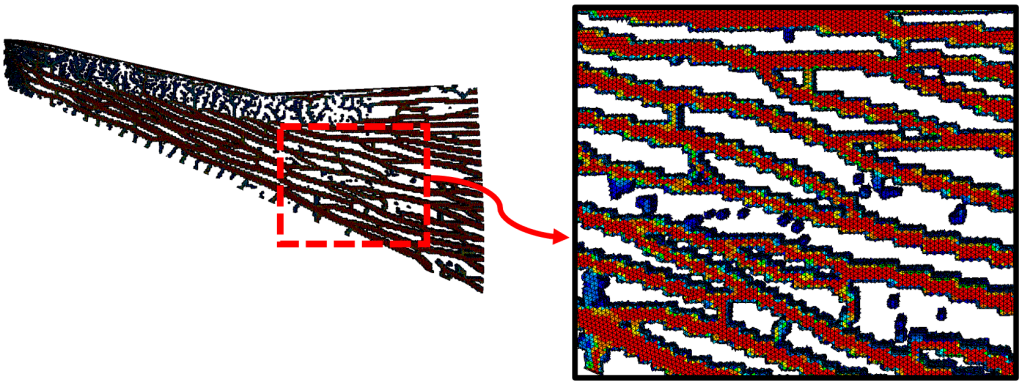
the compliance as objective function and global mass constraint of 40%. Additionally, a max member constraint is given to the optimization, that restricts the thickness of the ribs within a specified value. The surface of the wing model made of shell elements are chosen as the frozen elements, which means that they are non-optimizing elements. Only the interior elements are



(a) Current *Tosca*



(b) Cuboid centroids

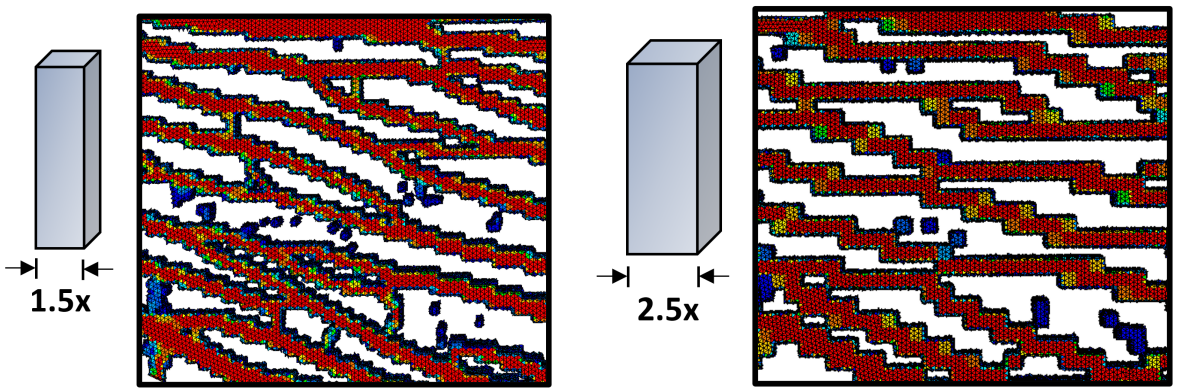


(c) Volume fraction

Figure 5.21: Density distribution

designed using the topology optimization. Fig. (5.21) shows the density plot distribution for the wing model using all three methods. The result from Current *Tosca* contains a lot of intermediate and high density elements spread into the void regions. This makes it difficult to distinguish the ribs from the empty regions. The presence of high density elements in the void regions is comparatively less in the Volume fraction method. This is also better in the case of Cuboid centroids. However, Cuboid centroids result contains ribs with more intermediate density elements. In case of Volume fraction, the elements with low or intermediate densities fall mostly on the boundary of the ribs, that helps in distinguishing the ribs and the voids.

The concerning factor in the Cuboid centroids and Volume fraction method is the presence of step-like patterns. A more detailed analysis explained the reason for step patterns. This is due to the larger pole size in comparison to the average element edge length. Based on the minimum and maximum nodal coordinate of the model, the boundary for the grids will be determined. The volume of the model is then calculated based on the dimensions of the grid boundary. Eventually, the pole size is determined using the volume and the number of finite elements in the model. The pole size for this wing model is by default considered as 1.5 times the average element edge length. However, a pole size less than average edge length is usually preferred for better results. This is because the pole size influences the elements that are grouped to a pole and in turn affects the mapping of pole to element density values. Fig. (5.22) shows the results of the Volume fraction method for pole sizes greater than average



(a) *1.5 times average edge length* **(b)** *2.5 times average edge length*

Figure 5.22: Results for pole size greater than average element edge length

element edge length of the model, whereas Fig. (5.23) shows the result for pole size less than the average edge length. The relative values of the pole size to the average element edge length is also mentioned for each model. It can be observed that the step patterns reduces, if the pole size is less than the average edge length (e.g, Fig. (5.23)). On the other hand, the step patterns become much obvious as the pole size increases. The increase in pole size reduces the number of poles needed to approximate the total number of finite elements, leading to decreased accuracy. Conversely, a smaller pole size relative to the average edge length requires more poles, resulting in accurate and better results.

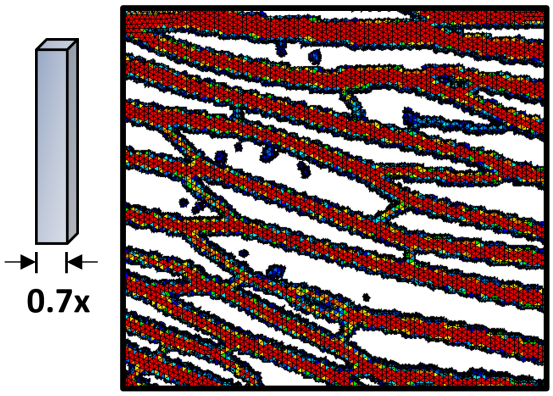


Figure 5.23: Pole size - 0.7 times average edge length

5.5 Car door model

The car door model consists of approximately 500,000 tetrahedron finite elements. The boundary condition is given as pin support to the door hinges and a vertically downward force is applied on the door handle (Fig. (5.24)). Minimizing compliance and local volume of 40%

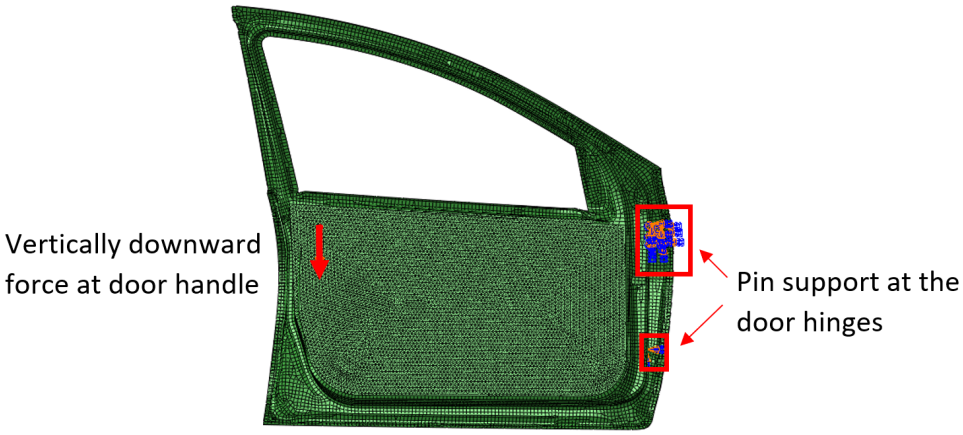
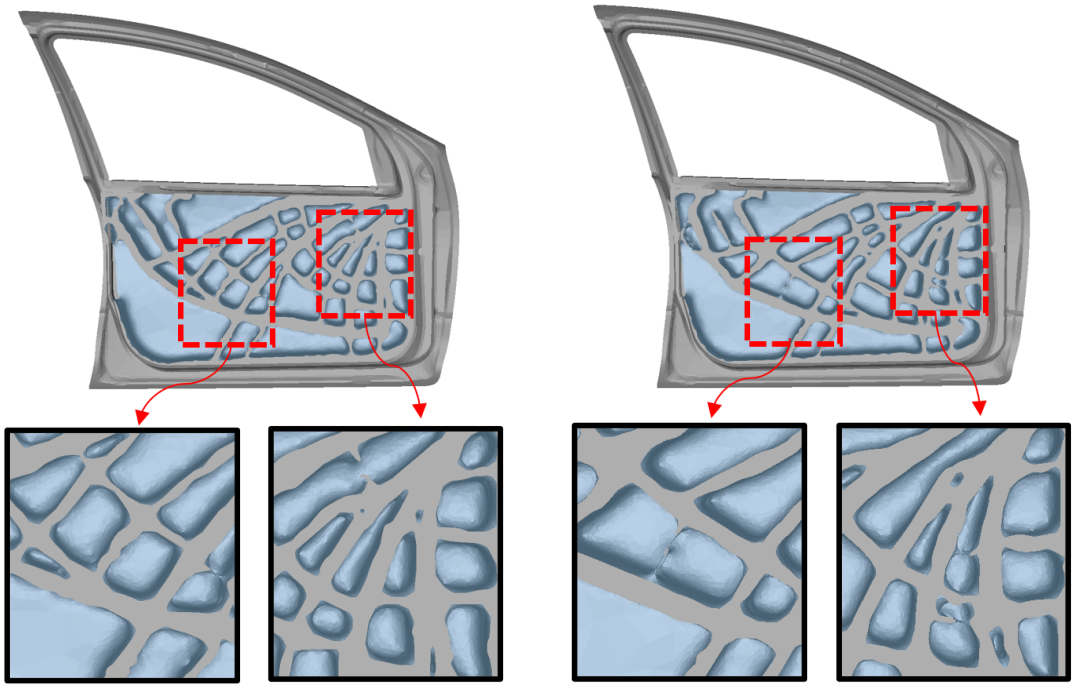


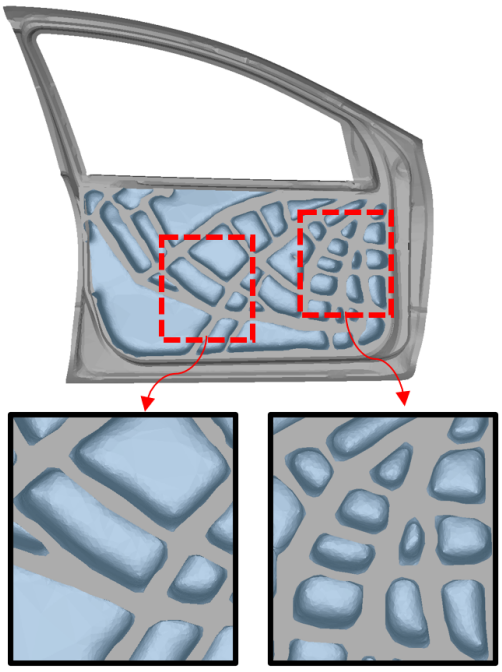
Figure 5.24: Car door FE model

forms the objective and constraint. The interior body of the model below window will constitute the design space and rest of the model is set frozen. The ribs are constrained to grow for the full design height. Therefore, the bounds for the parameter C and L will be zero and one respectively. Tosca Smooth results for all three methods are compared. A clear pattern of holes with less irregular surfaces are observed for the Volume fraction method. Certain irregular surfaces in the Current *Tosca* and Cuboid centroids method lead to potential undercuts in the model. This is shown in Fig (5.25a) and Fig (5.25b). The relative volume percentages are also noted as 53.58%, 53.26% and 52.88% for Current *Tosca*, Cuboid centroids and Volume fraction method respectively. This confirms that there is no significant differences in the amount of material present in the model. The volume percentage values are greater than the local volume of 40%, since the relative volume is calculated along with the frozen areas.



(a) *Current Tosca (relative volume - 53.58%)*

(b) *Cuboid centroids (relative volume - 53.26%)*



(c) *Volume fraction (relative volume - 52.88%)*

Figure 5.25: Tosca Smooth for car door model

6

Summary and future work

In this chapter, a final summary of the thesis is provided. Important aspects and a short recap of the results will be discussed along with potential future steps.

6.1 Summary

In this work, a modified parametrization, that achieves a better and consistent results for the casting constraints in a topology optimization was presented. Fundamental concepts on the parameters, that inherently enforce the casting restriction and the mapping formulation, that relates parameters to element relative densities were discussed. Understanding the beforementioned concepts were necessary for formulating the mapping and sensitivities based on the newly proposed concept. Two methods were introduced in this work, namely Cuboid centroids and Volume fraction. Cuboid centroids method uses square grids, compared to the circular grids in *SIMULIA Tosca Structure*. However, the mapping function and sensitivity formulations remained the same for both methods. The advantage of using square grids are realized in the Volume fraction method, that uses the planes of the pole to identify fractions of elements in a pole. The mapping function and sensitivities were reformulated based on the element volume fractions and the results of the sensitivities were also validated with the finite difference method. The optimization with casting constraints for the industrial models, showed improved results for the new implementation. Table (6.1) shows a comparison of the results from the new implementations with the currently available method. Regarding the compliance value, the Current *Tosca* and Cuboid centroids method resulted in low compliance

Table 6.1: Comparison of results

Comparing quantities	**Current** *Tosca*	**Cuboid centroids**	**Volume fraction**
Low compliance values	++	++	+
Density plot distribution	−	−	+
Tosca Smooth	−	−	++
Runtime performance	O	++	O

values. Volume fraction results contained many intermediate or low density elements along the boundaries of the solid and void regions. This reduced the stiffness of the whole structure, leading to a higher compliance value. But the difference was not significant in comparison to other two methods. On the other hand, the presence of these low or intermediate density elements in Volume fraction results, showed a clear segregation of the regions with full material and empty regions. Current *Tosca* and Cuboid centroids method had higher density elements in the void regions, that resulted in a negative impact with respect to the density plots. Concerning the Tosca Smooth results, both Current *Tosca* and Cuboid centroids had interior cavities and some irregularities. The surfaces were better and smooth for the Volume fraction method, thereby preventing potential undercuts or cavities in the model. Regarding the runtime performance, the Cuboid centroids was optimal, because of its efficient grid formation. The Volume fraction method gains some performance as it uses the same grid as Cuboid centroids. However, the overhead on calculations of element volume fractions brings down the performance. Nevertheless, it is comparable or even better than Current *Tosca* in case of fine meshes.

6.2 Conclusion

The starting point of the thesis was understanding and modularizing the existing code for the casting constraint parametrization in *SIMULIA Tosca Structure*. This made it efficient to bring the new prototypes proposed in this thesis. The main aim of the thesis is to improve the existing formulation in order to obtain consistent castable results, thereby preventing potential cavities, irregularities or some artifacts produced in the existing method. The optimization results for the casting constraint were improved by implementing two methods through this work. One of the methods, namely Cuboid centroids, provides optimal runtime performance, mainly due to faster grid formation compared to the existing method. Another method, known as Volume fraction, focusses on the mapping function and sensitivities. They were reformulated using the element volume fractions, resulting in good qualitative results, especially for Tosca Smooth. The presence of low or intermediate density elements along the boundaries of solid and void provides a clear distinction between regions with material and no material. This significantly aids in better interpolation of the densities along the boundaries, yielding smoother surfaces with fewer irregularities for Tosca Smooth. If the model is sufficiently complex or contains many ribs and voids, then the Volume fraction method might yield promising results. On the other hand, the Volume fraction method is less likely to show major improvements over the existing methods for simpler models. The runtime performance for the Volume fraction method is relatively higher, especially in cases of fine meshes. This was also noted as a limitation for this implementation. Each of the implemented functionalities were verified with appropriate unit tests. This made sure that the desired and expected results are obtained for each function. Finally, the new implementations were validated with industrial models and the results of the optimization were compared with the results from the existing method. The results confirmed the efficacy of the new implementations.

6.3 Future work

The summary discussed that the runtime performance for Volume fraction method is more, compared to the Cuboid centroids. One of the ideas is to parallelize the volume fraction calculations, so that the time gap between Volume fraction and Cuboid centroids method is reduced.

The algorithms implemented in this work can also be extended to work for different grid, e.g, a 2d tria grid, that is capable of capturing much more intricacies in the model. Another idea

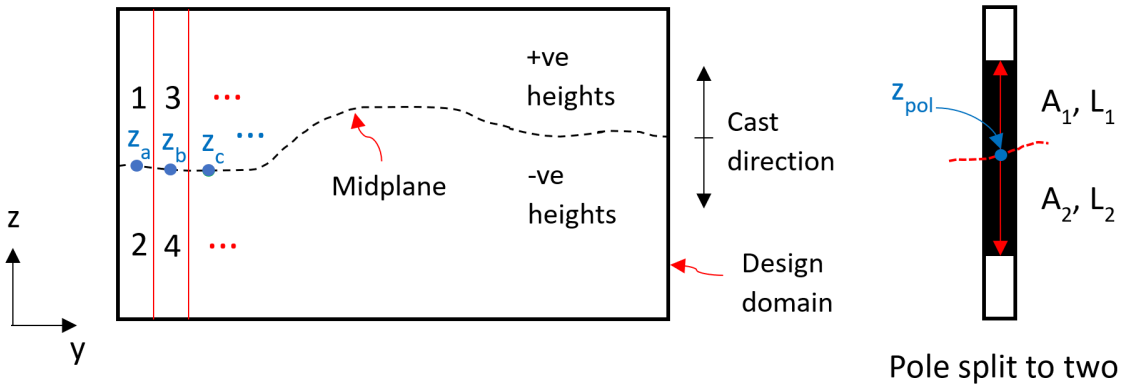


Figure 6.1: Parting plane inside the model

is to utilize the algorithm that forms cuboid poles to create a parting plane inside the model. Fig. (6.1) shows a design domain with a parting plane. The casting direction will be upwards and downwards away from the parting plane. The idea is to identify elements above and below midplane as positive and negative heights of their centroids. This is followed by creating a cuboid pole for the whole domain and then splitting into two subdomains (refer Leiva u. a. (2004a) for a simple example) based on the positive and negative heights. Here, the bounds of the parameter C will be zero, so that no interior cavities are formed near the midplane. Subsequent idea is to include the heights of the midplane points (denoted as $z_a, z_b, ...$) along with parameters A and L as design variables in the optimization. This allows to generate by itself a varying midplane for the model.

7

Appendix

7.1 Finite Difference Method

The sensitivity calculations performed in this work are validated with the Finite Difference Method (FDM). FDM provides a simple way to acquire the sensitivity information regarding a design response, see Huang u. a. (1986) or Choi und Kim (2004). FDM approximates the derivative of a function or design response with respect to design variables by evaluating the difference in function values before and after perturbing the design variables. This is given mathematically as follows.

$$\frac{\mathrm{d}\psi(\mathbf{x})}{\mathrm{d}\mathbf{x}} = \frac{\psi(\mathbf{x}+\delta\boldsymbol{x})-\psi(\mathbf{x})}{\delta\boldsymbol{x}} \tag{7.1}$$

Here, $\psi(\mathbf{x})$ is some function or design response dependent on the design variables $\mathbf{x}$. $\delta\boldsymbol{x}$ are the perturbations of design variables $\mathbf{x}$. The above expression is a forward difference method. If the perturbation were $-\delta\boldsymbol{x}$ instead of $+\delta\boldsymbol{x}$ in Eq. (7.1), then the method will be a backward difference method. The central difference method will have the perturbations in both the positive and negative directions.

Consider the sensitivity relations given in Eq. (4.6) and Eq. (4.7). The differential functions $\frac{\mathrm{d}f_a(\mathbf{A})}{\mathrm{d}\mathbf{A}}$, $\frac{\mathrm{d}f_l(\mathbf{L})}{\mathrm{d}\mathbf{L}}$ and $\frac{\mathrm{d}f_c(\mathbf{C})}{\mathrm{d}\mathbf{C}}$ can be validated with the help of finite difference method. Considering the function $f_a(\mathbf{A})$ given in terms of the parameter $\mathbf{A}$ in Eq. (4.11). Assuming a simple case, where no filtering of poles is performed, the weight matrix is unity and the function $f_a(\mathbf{A})$ reduces to a simple expression as,

$$f_a(\mathbf{A}) = V_{i,k}^{frac} \cdot A_k^p \tag{7.2}$$

To determine the $\frac{\mathrm{d}f_a(\mathbf{A})}{\mathrm{d}\mathbf{A}}$ using FDM, the perturbations are introduced for the parameter A_k^p.

$$\frac{\mathrm{d}f_a(\mathbf{A})}{\mathrm{d}\mathbf{A}} = \frac{f_a(\mathbf{A}+\delta\boldsymbol{A})-f_a(\mathbf{A})}{\delta\boldsymbol{A}} = \frac{V_{i,k}^{frac}\cdot(A_k^p+\delta A_k^p) - V_{i,k}^{frac}\cdot A_k^p}{\delta A_k^p} \tag{7.3}$$

Here, $f_a(\mathbf{A})$ is a linear equation depending on the parameter $\mathbf{A}$, so the result of the differentiation does not vary for any value of perturbation $\delta\boldsymbol{A}$. However, the perturbation value is more important, if the function is nonlinearly dependent on the parameter $\mathbf{A}$. In nonlinear problems, it is preferable to use small perturbation values. Larger perturbation values can lead to significant variations in results. Similarly, for the functions $f_l(\mathbf{L})$ and $f_c(\mathbf{C})$ defined in the plots (see Fig. 4.3), the finite difference method can be used to validate the differentials $\frac{\mathrm{d}f_l(\mathbf{L})}{\mathrm{d}\mathbf{L}}$ and $\frac{\mathrm{d}f_c(\mathbf{C})}{\mathrm{d}\mathbf{C}}$. Note that the differentiation of both these functions are not valid at the minimum $h^e_{\min}$ and maximum $h^e_{\max}$ heights of an element. FDM can be applied to the linear function between $h^e_{\min}$ and $h^e_{\max}$.

The partial differentiation of the design response with respect to the element relative densities, $\frac{\partial\phi}{\partial\boldsymbol{\rho}}$ (see Eq. 4.6) is obtained from solver in most case. Nevertheless, this differentiation can also be validated with the forward difference as,

$$\frac{\partial\phi(\boldsymbol{\rho})}{\partial\boldsymbol{\rho}} = \frac{\phi(\boldsymbol{\rho}+\delta\boldsymbol{\rho})-\phi(\boldsymbol{\rho})}{\delta\boldsymbol{\rho}} \qquad \text{where, } \delta\boldsymbol{\rho} \text{ - perturbation in element densities} \tag{7.4}$$

Bibliography

Abaqus 2024
: Abaqus: SIMULIA Abaqus Documentation - Version 2024. Dassault Systèmes. (2024)

Akgun u. a. 2001
: Akgun, Mehmet; Garcelon, John; Haftka, Raphael; Walsh, Joanne; Wu, K.: Efficient Structural Optimization for Multiple Load Cases Using Adjoint Sensitivities. In: *Aiaa Journal - AIAA J* 39 (2001), 03, S. 511–516

Barrett u. a. 1996
: Barrett, Richard; Berry, Michael; Chan, Tony; Demmel, James; Donato, June; Dongarra, Jack; Eijkhout, Victor; Pozo, Roldan; Romine, Chris; Vorst, Henk Van der: Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods. In: *Mathematics of Computation* 64 (1996), 09

Bendsoe und Sigmund 1999
: Bendsoe, Martin; Sigmund, Ole: Material interpolation schemes in topology optimization. In: *Archive of Applied Mechanics* 69 (1999), 11, S. 635–654

Choi und Kim 2004
: Choi, Kyung; Kim, Nam: *Structural Sensitivity Analysis and Optimization I—Linear Systems*. Bd. 1. 11 2004. – ISBN 978-0387232324

Christensen und Klarbring 2008
: Christensen, Peter; Klarbring, Anders: *An Introduction to Structural Optimization*. Bd. 153. 01 2008. – ISBN 978-1-4020-8665-6

Dompierre u. a. 1999
: Dompierre, Julien; Labbé, Paul; Vallet, Marie-Gabrielle; Camarero, Ricardo: How to Subdivide Pyramids, Prisms, and Hexahedra into Tetrahedra. (1999), 01, S. 195–204

Gersborg und Andreasen 2010
: Gersborg, Allan; Andreasen, Casper: An explicit parameterization for casting constraints in gradient driven topology optimization. In: *Structural and Multidisciplinary Optimization* 44 (2010), 01, S. 875–881

Harzheim und Graf 2002
Harzheim, Lothar; Graf, Gerhard: TopShape: An attempt to create design proposals including manufacturing constraints. In: *International Journal of Vehicle Design - INT J VEH DES* 28 (2002), 01

Huang u. a. 1986
Huang, E.; Choi, K.; Komkov, V.: *Design Sensitivity Analysis of Structural Systems.* Academic Press Inc., Harcourt, 1986

Ide u. a. 2014
Ide, Takanori; Otomori, Masaki; Leiva, Juan; Watson, Brian: Structural optimization methods and techniques to design light and efficient automatic transmission of vehicles with low radiated noise. In: *Structural and Multidisciplinary Optimization* 50 (2014), 12

Leiva u. a. 2004a
Leiva, Juan; Watson, Brian; Kosaka, Iku: An Analytical Directional Growth Topology Parameterization to Enforce Manufacturing Requirements. In: *Collection of Technical Papers - AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference* 2 (2004), 04. ISBN 978-1-62410-079-6

Leiva u. a. 2004b
Leiva, Juan; Watson, Brian; Kosaka, Iku: An Analyticall Bi-Directional Growth Parameterization to Obtain Optimal Castable Topology Designs. In: *Collection of Technical Papers - 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference* 5 (2004), 08. ISBN 978-1-62410-019-2

Liu u. a. 2015
Liu, Shutian; Li, Quhao; Chen, Wenjiong; Rui, Hu; Tong, Liyong: H-DGTP—a Heaviside-function based directional growth topology parameterization for design optimization of stiffener layout and height of thin-walled structures. In: *Structural and Multidisciplinary Optimization* 52 (2015), 06, S. 1–11

Liu und Vinter 2015
Liu, Weifeng; Vinter, Brian: CSR5: An Efficient Storage Format for Cross-Platform Sparse Matrix-Vector Multiplication, 03 2015

Saad 1990
Saad, Yousef: SPARSKIT:A basic toolkit for sparse matrix computations. (1990), 01

Schweiger und Arridge 2016
Schweiger, Martin; Arridge, Simon: Basis mapping methods for forward and inverse problems: BASIS MAPPING METHODS. In: *International Journal for Numerical Methods in Engineering* 109 (2016), 05

Scott und Tůma 2023
Scott, Jennifer; Tůma, Miroslav: *An Introduction to Sparse Matrices.* S. 1–18, 01 2023. – ISBN 978-3-031-25819-0

Sigmund 2001
Sigmund, Ole: Sigmund, O.: A 99 Line Topology Optimization Code Written in MATLAB. Structural and Multidisciplinary Optimization 21, 120-127. In: *Structural and Multidisciplinary Optimization* 21 (2001), 04, S. 120–127

Sigmund 2007
Sigmund, Ole: Morphology-based black and white filters for topology optimization. In: *Structural and Multidisciplinary Optimization* 33 (2007)

Svanberg 1987
Svanberg, Krister: The method of moving asymptotes - A new method for structural optimization. In: *International Journal for Numerical Methods in Engineering* 24 (1987), 02, S. 359 – 373

Tosca 2024
Tosca: SIMULIA Tosca Documentation - Version 2024. Dassault Systèmes. (2024)

Zhang u. a. 2020
Zhang, Yisheng; Wu, Xiangdong; Guan, Bobin; Zhao, Zhe; Wan, Min: Application and practical validation of topology optimization technology for the frame of biaxial tensile testing machine. In: *Structural and Multidisciplinary Optimization* 62 (2020), 09

Zhou u. a. 2002
Zhou, Ming; Fleury, Raphael; Shyy, Yaw-Kang; Thomas, Harold; Brennan, Jeffrey: Progress in Topology Optimization with Manufacturing Constraints. (2002), 09. ISBN 978-1-62410-120-5

Zuo u. a. 2007
Zuo, Kong-Tian; Chen, Li-Ping; Zhang, Yunqing; Yang, Jingzhou: Study of key algorithms in topology optimization. In: *International Journal of Advanced Manufacturing Technology* 32 (2007), 04, S. 787–796