Department of Precision and Microsystems Engineering

Overhang Control for Multi-Materials in Density-Based Topology Optimization

Niek A. van Rossem

Report no	: 2026.022
Coach	: Ir. J. van der Zwet
Professor	: Dr. ir. C. Ayas
Specialisation	: Computational Design and Mechanics (CDM)
Type of report	: MSc Thesis
Date	: March 5, 2026

TU Delft
Delft University of Technology

Precision and Microsystems Engineering

Challenge the future

Overhang control for multi-materials in Density-Based Topology Optimization

by

Niek A. van Rossem

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday March 5, 2026 at 10:00 AM.

Student number:	4901967	
Project duration:	November, 2024 – March, 2026	
Thesis committee:	Dr. ir. C. Ayas,	TU Delft
	Ir. J. van der Zwet,	TU Delft
	Dr. ir. L. Noël,	TU Delft
	Dr. J. Wu,	TU Delft
Faculty:	Faculty of Mechanical Engineering, Delft	
Submitted on:	26 February, 2026	
Document version	1.0.1	

Style: TU Delft Report Style, with modifications by Daan Zwaneveld

Preface

When I had to choose my thesis topic in the autumn of 2024, I did not know where exactly I wanted to move towards for my thesis. I had just finished a year as part of Formula Student Team Delft, where I had mainly been working in two areas, namely vehicle dynamics and mechanical design. For my thesis I wanted to do something that connected naturally to one of these. I came across a project on manufacturability of multi-material additive manufacturing, and knew that this was the project for me. What drew me to this project primarily was that it couples the theoretical with the practical, and has practical uses in the industry.

Just over a year later, and here lies my final thesis. In the past months, I have been hard at work developing two different methods to control the overhang angle in a multi-material topology optimization problem. This project has not been without its road blocks; especially the analytical derivation of the overhang filter took me much longer than expected. In the end, however, I am proud to have created a working topology optimization filter that is ready for publication.

This thesis has also been instrumental in providing me with a thorough background in topology optimization, and giving me a clear overview of all the foundational papers concerning manufacturability limitations. It taught me to write proper Pythonic code, as well as how to organize and structure large code folders; knowledge which will be incredibly useful in subsequent projects.

I would in particular like to thank Dr. ir. Can Ayas and Ir. Joran van der Zwet for their continued support and supervision of this thesis, as well as for their patience in reviewing the many revisions of the paper manuscript. In doing so, they pushed my technical writing proficiency to a new level.

I would also like to thank Ir. Manav Das and Dr. ir. Arnoud Delissen for their help with the single-material hotspot constraint and the overhang filter, respectively.

Lastly, I would like to dedicate this thesis to my grandfather, Ger Aarts, who unfortunately passed away a few months after starting this thesis.

Niek A. van Rossem
The Hague, February 2026

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

Abbreviation	Definition
AM	Additive manufacturing
BESO	Bi-directional evolutionary structural optimization
CNC	Computer numeric control
DED	Direct/directed energy deposition
DMLS	Direct metal laser sintering
DMO	Discrete material optimization
EB-PBF	Electron beam powder bed fusion
ESO	Evolutionary structural optimization
KS	Kreisselmeier-Steinhauser
L-DED	Laser direct energy deposition
L-PBF	laser powder bed fusion
LSM	Level set method
MBB	Messerschmitt-Bölkow-Blohm
MMTO	Multi-material topology optimization
MSSI	Maximum shear stress index
PBF	Powder bed fusion
RAMP	Rational approximation of material properties
RMMI	Recursive multiphase materials interpolation
SIMP	Solid isotropic material with penalization
SLM	Selective laser melting
SLS	Selective laser sintering
TO	Topology optimization
VTM	Virtual temperature method
WA-DED	wire arc direct energy deposition
WAAM	Wire arc additive manufacturing

Symbols

The symbols in the tables below apply only to Chapters 3 and 4. The equations from the literature in Chapter 1 and 2 use the notation of the source paper as much as possible.

Scalar quantities

Symbol	Definition
a	Horizontal distance between two element centroids
b	vertical distance between two element centroids
c	Compliance
c_0	Scaling factor for the objective and constraints
c_1	Compliance of the first multi-material reference MBB beam
c_2	Compliance of the second multi-material reference MBB beam
c_{sm}	Compliance of the single-material reference MBB beam
E_e	Young's modulus of an element
$E_{\max}$	Young's modulus of the stiffest material
$E_{\min}$	Young's modulus of the most compliant material

$E^{(1)}$	Young's modulus of the red alternative material
$E^{(2)}$	Young's modulus of the blue reference material
F	Applied force
f	Generic objective
g	Hotspot constraint value
h	Harmonic mean
h_1	Harmonic mean of the first multi-material reference platform design
h_2	Harmonic mean of the second multi-material reference platform design
h_3	Harmonic mean of the third multi-material reference platform design
h_4	Harmonic mean of the fourth multi-material reference platform design
h_{sm}	Harmonic mean of the single-material reference platform design
m_{dil}	Mass of the dilated design
m_{max}	Highest allowable mass
n_j	Number of elements in layer j
n_h	Number of layers in the deposition direction
n_{int}	Number of interpolation points on the edge of the cone
n_{iter}	Number of iterations for the optimization
n_{s}	Number of element centroids inside the support cone
$n_{\mathrm{s}}^{(m)}$	Number of element centroids inside the support cone for material m
P	P-norm exponent
p	Intermediate density penalization
Q	Corrected p-norm exponent
$Q^{(m)}$	Corrected p-norm exponent for material m
$\bar{q}$	Low density penalization for conductivity contribution
R	Density and Young's modulus ratio of alternative material to reference material
r	Density filter radius
$\bar{r}$	Low density relaxation for hotspot constraint
T_{cr}	Critical temperature (SMTO)
$T_{\mathrm{cr}}^{(1)}$	Critical temperature of the red alternative material
$T_{\mathrm{cr}}^{(2)}$	Critical temperature of the blue reference material
T_{max}	Maximum temperature in the design domain
s	Ratio of dilated mass to nominal mass
V	Volume
v	Volume fraction
v_{f}	Volume fraction of the frame
$v_{\mathrm{f,max}}$	Upper limit for the volume fraction of the frame
v_{p}	Volume fraction of the platform
$v_{\mathrm{p,min}}$	Lower limit for the volume fraction of the platform
$w_{(k,j)}$	Penalization for conductivity contribution
$w_{(k,j)}^{(\mathrm{ang})}$	Angular penalization for conductivity contribution
$w_{(k,j)}^{(\mathrm{rad})}$	Radial penalization for conductivity contribution
α	Critical overhang angle
$\alpha^{(1)}$	Critical overhang angle of the red alternative material
$\alpha^{(2)}$	Critical overhang angle of the blue reference material
$\bar{\alpha}$	Scaling factor for hotspot constraint
β	Heaviside projection sharpness
β_{max}	Maximum Heaviside projection sharpness
ϵ	Small shift for mixture calculation from material densities
ε	Small shift for smooth minimum calculation
η	Heaviside threshold
η_{ero}	Eroded Heaviside threshold
η_{nom}	Nominal Heaviside threshold
η_{dil}	Dilated Heaviside threshold
κ	Thermal interaction length

ν	Poisson's ratio
ξ_0	Overestimation suppression parameter
π	Pi
ρ	Physical density
$\rho^{(1)}$	Physical density of the red alternative material
$\rho^{(2)}$	Physical density of the blue reference material
τ_f	Stopping criterion for objective change
τ_x	Stopping criterion for design change
$\bar{\chi}_\mathrm{min}$	Minimum normalized density to avoid singularities
ω_k	Natural frequency of mode k

Vector quantities

Symbol	**Definition**
$\mathbf{a}^{(m)}$	Density summation of elements in $S_e^{(m)}$, raised to the power P
$\mathbf{f}$	Force vector
$\mathbf{T}$	Temperature field
$\mathbf{T}_\mathrm{mmto}$	Temperature field, corrected for MMTO
$\mathbf{u}$	Displacement vector
$\mathbf{x}$	Input density vector (SMTO)
$\mathbf{x}^{(\mathrm{s})}$	Input density vector (RMMI)
$\mathbf{x}^{(\mathrm{m})}$	Input mixture vector (RMMI)
$\bar{\mathbf{x}}^{(\mathrm{s})}$	Filtered input density vector (RMMI)
$\bar{\mathbf{x}}^{(\mathrm{m})}$	Filtered input mixture vector (RMMI)
$\hat{\mathbf{x}}^{(\mathrm{s})}_\mathrm{ero}$	Filtered and projected input density vector (eroded, RMMI)
$\hat{\mathbf{x}}^{(\mathrm{s})}_\mathrm{nom}$	Filtered and projected input density vector (nominal, RMMI)
$\hat{\mathbf{x}}^{(\mathrm{s})}_\mathrm{dil}$	Filtered and projected input density vector (dilated, RMMI)
$\hat{\mathbf{x}}^{(\mathrm{m})}_\mathrm{nom}$	Filtered and projected input mixture vector (nominal, RMMI)
$\boldsymbol{\zeta}$	Support field density vector
$\boldsymbol{\lambda}^{(m)}$	Vector of multipliers for the adjoint formulation for material m
$\boldsymbol{\mu}^{(m)}_\mathrm{ero}$	Filtered and projected density vector for material m (eroded)
$\boldsymbol{\mu}^{(m)}_\mathrm{nom}$	Filtered and projected density vector for material m (nominal)
$\boldsymbol{\mu}^{(m)}_\mathrm{dil}$	Filtered and projected density vector for material m (dilated)
$\boldsymbol{\Xi}$	Maximum support density vector
$\boldsymbol{\Xi}^{(m)}$	Maximum support density vector for material m (layer-wise notation)
$\hat{\boldsymbol{\Xi}}^{(m)}$	Scaled maximum support density vector for material m
$\boldsymbol{\xi}$	Printable density vector (SMTO)
$\boldsymbol{\xi}^{(m)}_\mathrm{ero}$	Printable density vector of material m (eroded)
$\boldsymbol{\xi}^{(m)}_\mathrm{ero}$	Printable density vector of material m (nominal)
$\boldsymbol{\xi}^{(m)}_\mathrm{ero}$	Printable density vector of material m (dilated)
$\boldsymbol{\xi}$	Singularity-protected printable density field
$\boldsymbol{\varphi}_k$	Eigenvector of mode shape k
$\boldsymbol{\chi}^{(\mathrm{s})}_\mathrm{ero}$	Printable density vector (eroded, RMMI)
$\boldsymbol{\chi}^{(\mathrm{s})}_\mathrm{nom}$	Printable density vector (nominal, RMMI)
$\boldsymbol{\chi}^{(\mathrm{s})}_\mathrm{dil}$	Printable density vector (dilated, RMMI)
$\boldsymbol{\chi}^{(\mathrm{m})}_\mathrm{ero}$	Printable mixture vector (eroded, RMMI)
$\boldsymbol{\chi}^{(\mathrm{m})}_\mathrm{nom}$	Printable mixture vector (nominal, RMMI)
$\boldsymbol{\chi}^{(\mathrm{m})}_\mathrm{dil}$	Printable mixture vector (dilated, RMMI)

Matrix quantities

Symbol	Definition
$\mathbf{D}$	Support element selection matrix for $\tilde{S}_{j\|i}$
$\mathbf{D}^{(m)}$	Support element selection matrix for $\tilde{S}_{j\|i}$ for material m
$\mathbf{H}^{(l)}$	Weights matrix for element interpolation of point l
$\mathbf{H}^{(l\|m)}$	Weights matrix for element interpolation of point l for material m
$\mathbf{K}$	Stiffness matrix
$\mathbf{k}_e^0$	Element stiffness matrix
$\mathbf{M}$	Mass matrix
$\mathbf{m}_e^0$	Element mass matrix

Sets

Symbol	Definition
S_e	Set of element centroids that fall within the support cone of element e
T_e	Set of interpolation points on the edge of the support cone of element e

Operators

Symbol	Definition
d	Derivative
∂	Partial derivative
diag	Places an $n \times 1$ vector out over the diagonal of an $n \times n$ matrix
max	Take the maximum of the inputs
min	Take the minimum of the inputs
Π	Product over the inputs
Σ	Summation over the inputs

Indices

Symbol	Definition
e	Element index
i	Element index in layer-wise notation (e.g. element i in layer j)
j	Layer index
k	Generic index; defined on a per-case basis
l	Interpolation point index
m	Material index
p	Generic index; defined on a per-case basis
q	Generic index; defined on a per-case basis
s	Row index for $\mathbf{D}$
t	Column index for $\mathbf{D}$

1

Introduction

Engineers are constantly looking to push the limits of design. In industries such as aerospace or motorsports it is of great importance to make parts lighter, stiffer, and more durable. To this end, new manufacturing methods are developed that can produce previously infeasible geometries, and new design methodologies are devised to find performance in the geometries themselves. Two such developments that work well together are additive manufacturing (AM) and topology optimization (TO). AM techniques decouple production time and difficulty from geometric complexity, and TO provides a relatively straightforward method of generating intricate optimized shapes that take advantage of this.

Multi-material AM has also become feasible over the last five years, further increasing the design freedom. It allows engineers to create 3D printed parts that join multiple materials together in a single production process, eliminating the need for post-production assembly. Differences in material properties can be utilized to improve performance or to combine functionality. Examples of these are structures that strategically utilize a heavier material to modify the shape of its natural modes, structures that use the difference in thermal expansion coefficient to change shape with temperature, and many others. In multiphysics problems each material can be optimized for its own function. A heat exchanger can become a structural component of the larger system, with a core made of a high-conductivity material like copper to maximize heat transfer, and an outer shell of steel or titanium to provide strength.

Even though AM has much greater design freedom than conventional manufacturing methods, TO results are not directly manufacturable. The exact limitations differ per type of AM method used, but common restrictions include the inability to print features smaller than a given minimum size, the inability to print floating geometry or steep overhanging features, and the inability to print designs with enclosed voids, as the unmelted material cannot be removed from them.

TO designs can be post-processed to make them manufacturable, by adding support structures, by either filling fully enclosed cavities or drilling holes, and by manually increasing the thickness of thin geometry. This is labour-intensive engineering work that slows down the design-to-production process, and introduces the risk of creating unexpected problems or unintended behaviour arising from geometric modifications. Implementing schemes directly into TO that account for these limitations ensures that the design the optimizer produces is printable with minimal modifications, retaining its optimality.

1.1. Motivation and scope

For single-material topology optimization (SMTO), many schemes currently exist that enable the creation of print-ready designs, but not all of these can be implemented in multi-material topology optimization (MMTO) without major changes. MMTO also comes with its own limitations, such as the reduced strength of the material interface, or only being able to print a single material per layer. This thesis aims to add a valuable contribution to the field of topology optimization by developing a method to effectively restrict overhangs of multi-material designs, bringing the ability to create print-ready multi-material designs one step closer to reality.

1.2. Outline

This thesis is structured as follows: the rest of this chapter provides an introduction to both AM and TO, including the relevant AM processes, common material interpolation schemes, and the most commonly used filters and projection schemes for TO. Chapter 2 provides an overview of the current state-of-the-art in manufacturability constraints and filters, including enclosed void detection, overhang control methods, and interface constraints for MMTO. Chapter 3 provides a brief derivation and proof-of-concept examples for a multi-material extension to the geometric hotspot detection constraint. Chapter 4 constitutes the main body of this thesis, here a novel multi-material overhang filter is proposed that can restrict the critical overhang angle of each material phase independently to a predetermined custom angle. This chapter also serves as the manuscript for a paper, which will be submitted for publication in a scientific journal. To conform to the structure required for a paper, this chapter contains a brief repeat of the introduction and literature survey. Finally, Chapter 5 provides a performance comparison between the methods, and Chapter 6 provides a final conclusion to the thesis.

1.3. Background on additive manufacturing

AM is a set of production processes that gradually build up a part's geometry by solidifying metal powder or wire in designated areas. Unlike traditional subtractive production methods such as milling or lathing, the cost and duration of an AM process are largely independent of the number and complexity of the features of the part itself. Instead, it is proportional to the amount of material to be printed. Due to the high dimensional freedom, it is a process that is ideally suited to work in conjunction with TO. Recent developments have also made it possible to recycle unused powder, which significantly reduces material wastage compared to traditional machining processes. Much like CNC machining, the flexibility of the machines and the lack of required tooling make AM primarily suited to one-off or small batch production and prototyping.

An overview of common metal AM processes is provided in Figure 1.1. This chapter will primarily focus on three specific processes, namely laser powder bed fusion (L-PBF), laser direct energy deposition (L-DED), and wire arc direct energy deposition (WA-DED), as these are the primary methods with which metal parts containing multiple materials can be produced.

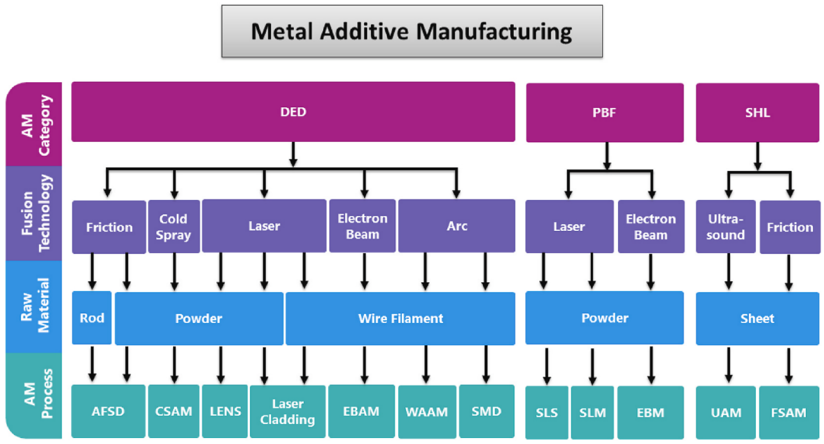


Figure 1.1: An overview of metal AM processes (Zainelabdeen et al., 2024).

1.3.1. Laser powder bed fusion

PBF is the process of building up a part layer-wise by melting or sintering metal powder in selected areas (Irrinki et al., 2020; Singh et al., 2021). A thin layer of powder is spread evenly across a build plate by a recoater, and this powder is subsequently fused into solid material in specific areas by a scanning laser (L-PBF) or an electron beam (EB-PBF). When this process is finished the recoater adds another layer of powder on top of the current one, and the process is repeated. Recently developed machines provide the capability to combine different metals into a single part (Aconity3D GmbH). Due to the powder-based nature of the production method, it is not possible to manufacture parts with true phase gradients (Aerosint), instead all parts are required to have explicitly defined interfaces. Another restriction that a designer needs to be cautious of, is that not all metals are compatible with one another, as some combinations form brittle intermetallic phases on the interface, or develop cracks due to a difference in thermal expansion coefficients (Bandyopadhyay et al., 2022). PBF can produce parts without requiring support structures, provided the overhang angle is not too great. If the overhang angle does exceed the critical value, care must be taken to ensure that support structures are accessible for removal, or otherwise do not impede part functionality.

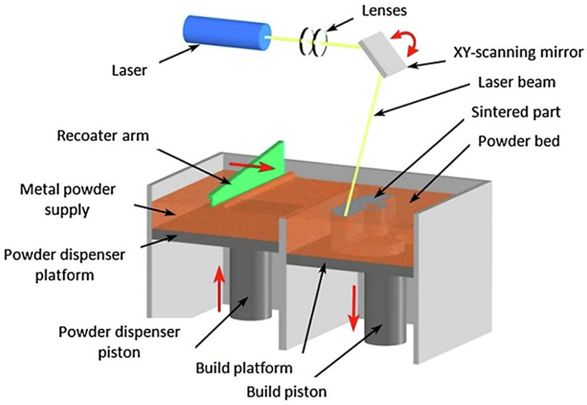


Figure 1.2: Schematic illustration of the L-PBF process (Singh et al., 2021).

1.3.2. Direct energy deposition

L-DED is the second major form of powder-based additive manufacturing, and is currently the most common form of multi-material AM. A robotic arm containing both the laser and powder feed nozzle moves through the build volume and simultaneously deposits and melts powder in the desired location (Martukanitz, 2020). A common application of DED is hybrid manufacturing: here additional features are printed onto an existing machined or wrought component. This approach combines the advantages of both classical and additive manufacturing. In comparison to PBF, the deposition rate of DED is significantly higher. A necessary consequence of this greater deposition rate, however, is that the minimum feature size is larger than what is possible with PBF. As the powder composition can be continuously varied during the melt process, DED is capable of producing designs with true material gradients. In some machines, the clamp that holds the workpiece can also be rotated (DMG Mori), which eliminates the overhang restriction.

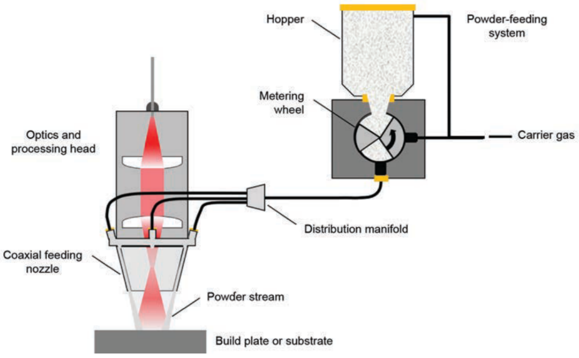


Figure 1.3: Schematic illustration of a laser DED system (Martukanitz, 2020).

1.3.3. Wire arc additive manufacturing

The last of the three primary multi-material AM methods is wire arc DED, often called wire arc additive manufacturing (WAAM). It is considered to be part of the DED family of production processes, but for the purposes of this chapter it will be treated as a separate category. WAAM uses wire feedstock as the material source instead of powder, which significantly decreases the cost, but also increases the minimum feature size compared to powder-based methods (Li et al., 2022). Much like with L-DED, material is simultaneously deposited and melted by a robotic arm which moves through the design space. WAAM has the highest material deposition rate of all the methods described here which makes it highly suitable for the production of large components. Post-processing in the form of heat treatments and machining operations is often necessary to reduce the residual stresses and improve the surface quality. Like L-DED, it is also possible to produce graded materials with WAAM (Wei et al., 2019).

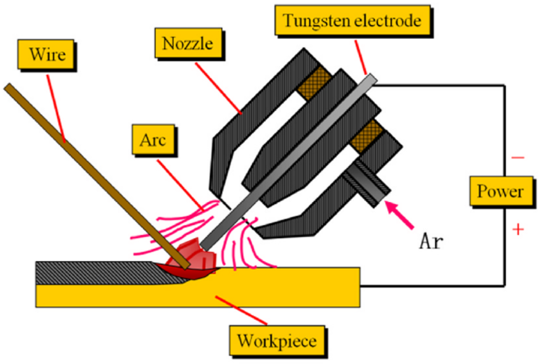


Figure 1.4: Schematic illustration of the Wire Arc Additive Manufacturing process (Li et al., 2022).

1.4. Background on topology optimization

The goal of TO is to find the optimum material distribution in a given domain that minimizes a set objective while not violating any prescribed constraints. The optimization problem is commonly stated in the negative-null form as follows:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, n_1 \\ & h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, n_2 \\ & x_{\min} \leq \mathbf{x} \leq x_{\max}, \end{aligned}$$

where the goal is to minimize the objective f by varying the values in the design vector $\mathbf{x}$. A commonly used objective is minimum compliance (Eq. 1.1), but alternative physics, such as maximizing the lowest n natural frequencies (Eq. 1.2), or minimizing the peak stress (Eq. 1.3) are also possible.

$$c = \mathbf{f}^T \mathbf{u} \tag{1.1}$$

$$h = \sum_{i=1}^{n_{\text{modes}}} \frac{1}{\omega_i} \tag{1.2}$$

$$\sigma_{\max} = \left(\sum_{e=1}^{n_{\text{el}}} \bar{\sigma}_e^P \right)^{1/P} \tag{1.3}$$

As most commonly used objectives are monotonic, adding more material always makes it better, running an unconstrained optimization problem is almost never feasible in practice, as this will always result in either a completely filled or completely void design domain. Mathematical constraints, indicated by g_i and h_j, are used to impose limitations on the final shape of the part, which ensures it meets further requirements not captured by the objective.

The most commonly used constraint is a limitation on the total volume fraction, meaning the percentage of the design space that is occupied. The compliance minimization problem with volume constraint can also be reversed, at which point it becomes a mass minimization problem with a maximum compliance constraint. Which form is more appropriate depends on the task at hand. Constraints have far more uses than these, and can be used to impose manufacturability limitations, or to ensure a part meets a requirement not captured by the objective.

Part geometry can be represented by either a boundary surface that is explicitly defined as a mathematical function, or by a normalized density field in a discrete mesh. The engineer then applies loads and boundary conditions to specified areas, and the program solves the initial value problem via finite element analysis (FEA) for a given initial material distribution or shape.

When using a boundary surface, this boundary is allowed to shift during each iteration of the optimization process, until the part design reaches an optimum shape. The level set method (LSM) proposed by Allaire et al. (2002) for example intersects a higher dimension surface with the design space, as shown in Figure 1.5. This results in a contour or surface that defines the part boundary.

In density-based TO, a normalized density field is defined throughout the design domain as in Figure 1.6; A density of zero indicates void and a density of one indicates the presence of material. During each iteration the density value of each element is modified, based on either the analytical or numerical sensitivity. The design space is generally discretized with a regular mesh of square or cubic elements, though irregular meshes are occasionally used.

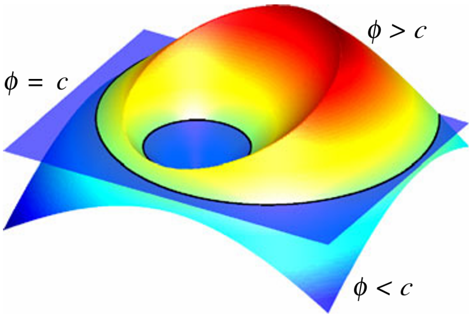


Figure 1.5: Example of the level set method (van Dijk et al., 2013).

Figure 1.6: Example of density based TO (Sigmund, 2007).

LSM provides an explicit and crisp part boundary without any post-processing, which can be beneficial for constraints that rely on the existence of an explicit edge. However, successful implementation of manufacturing constraints in LSM is significantly more complicated (van Dijk et al., 2013) than for density-based TO, therefore the rest of this thesis will focus on density-based TO exclusively.

The optimizer adds or removes material in order to reduce the value of the objective by determining its sensitivity over the change in material density for each element, and then moving towards an optimum with a gradient-based optimization approach. The ESO and BESO approaches (Querin et al., 1998) calculate the numerical sensitivity for each prospective change, and then execute only a specified fraction of the most sensitive ones. Other methods, such as the Method of Moving Asymptotes (MMA) proposed by Svanberg (1987), or a method of Optimality Criteria (OC), require analytical sensitivities. For this, the derivative of both the objective and the constraints with respect to the design field must be found. This includes all modifications to the design field, such as filters, projections, and interpolation schemes. Fortunately, this process can be simplified by using the chain rule: the partial derivative from the outputs of each operation to its inputs can be solved separately, after which they are chained together:

$$\frac{\mathrm{d}f}{\mathrm{d}x_e} = \frac{\partial f}{\partial z_e}\frac{\partial z_e}{\partial y_e}\frac{\partial y_e}{\partial x_e} \tag{1.4}$$

Therefore, the sensitivity of a new filter or constraint can be derived as if it were a stand-alone operation, and then be inserted into the existing process.

Another consequence of the requirement for analytical sensitivities is that element densities cannot simply be represented with a binary choice of 0 or 1. Instead, a continuous field on the interval $[0, 1]$ is used to allow the sensitivities to be found. Material properties are interpolated as a function of the element densities.

1.4.1. Material interpolation for SMTO

In order for the final design to be physically realizable, all elements must have a density of either zero or one. Thus the optimizer must be able to push intermediate densities to either void or solid. This can be achieved by using a curved interpolation scheme for one of the material properties, where intermediate densities are penalized. Commonly used interpolation models are the Solid Isotropic Material with Penalization (SIMP) model by Bendsøe (1989), shown in Equation 1.5, and the Rational Approximation of Material Properties (RAMP) model by Stolpe and Svanberg (2001), shown in Equation 1.6. A visual representation of both is provided in Figure 1.7.

$$E = x^p E_0 \tag{1.5}$$

$$E = \frac{x}{1 + q\,(1 - x)} \tag{1.6}$$

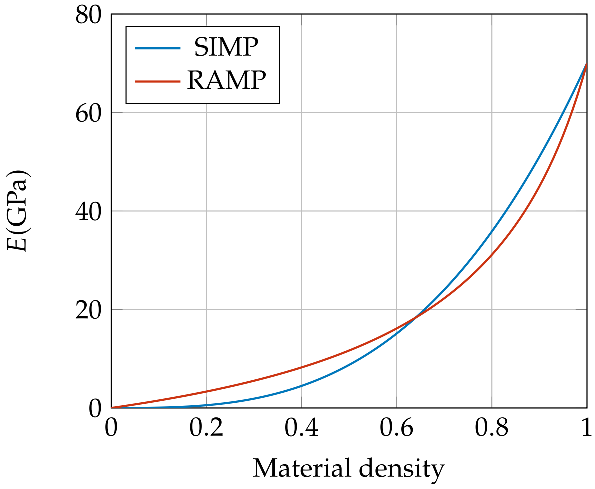


Figure 1.7: SIMP and RAMP interpolation schemes for AlSi10Mg. Penalization p is set to 3 for SIMP, and q is set to 4 for RAMP.

Both of these methods ensure that the specific stiffness of intermediate density material is artificially lowered, making it inefficient to use if the mass of the design is interpolated linearly, and pushing the optimizer to a 0-1 design.

1.4.2. Material interpolation for MMTO

Density-based methods for TO can be expanded to multi-materials without significant effort. To achieve this multiple density fields can be defined, each one corresponding to a different material. To be physically realizable, each point in the domain must at most be occupied by only a single material in full, or by a mixture of materials whose combined normalized density does not exceed one.

Sigmund and Torquato (1997) proposed the Recursive Multiphase Material Interpolation (RMMI) scheme, which couples the density fields of the different materials together. They achieve this by extending the existing SIMP model in a recursive manner. The first design variable x_1 acts as a total material density parameter, where a value of zero signifies void, and a value of one signifies that the element is filled with a material. The other design variables $x_2, \ldots, x_m$ act as mixture parameters. Changing their value switches between different materials. This coupling ensures that the sum of material densities remains below one. Penalization can be applied to each design variable independently, to ensure a 0-1 design with clear material interfaces. The expression for the RMMI model for a three-phase problem (two materials plus void) is shown in Equation 1.7. An example is plotted in Figure 1.8.

$$E_e = x_{1e}^{p_1}\left(x_{2e}^{p_2}E_{\max} + \left(1 - x_{2e}^{p_2}\right)E_{\min}\right) \tag{1.7}$$

Hashin and Shtrikman (1963) proposed a set of expressions that capture the lower and upper bound of expected material mixture properties, based on physical tests. These HS bounds can be used in MMTO in a similar manner to RMMI, with the advantage of being able to capture a difference in Poisson's ratio between the materials. Using this model also guarantees that material mixtures are physically accurate, which is essential for AM techniques like DED that can print such mixtures.

The Discrete Material Optimization (DMO) method by Stegmann and Lund (2005) takes an alternative approach to RMMI. Instead of defining total density and mixture coefficients, each field $x^{(i)}$ refers to a specific material. Instead of using the material densities directly, they are used to calculate a material weight coefficient w_i. This weight is not just based on the density of that particular element, but also on the density of every other element. Increasing the density of one element would decrease the density of every other element, which ensures that the total material density does not exceed one. Penalization on intermediate densities similar to the SIMP method is used here to drive the densities to 0 or 1, and convergence is achieved if a specified percentage of elements is made up of a single material.

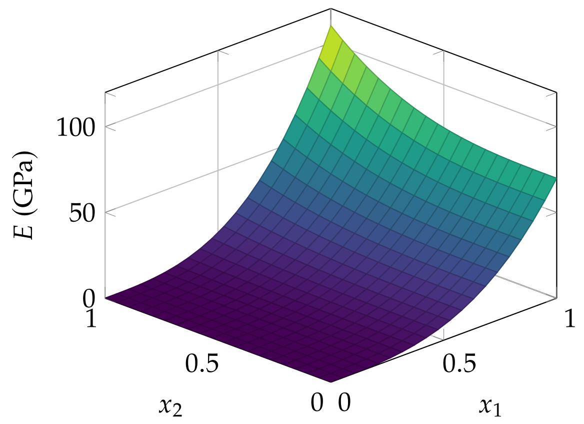


Figure 1.8: RMMI scheme for AlSi10Mg ($x_2 = 0$) and Ti6Al4V ($x_2 = 1$). $p_1 = p_2 = 3$

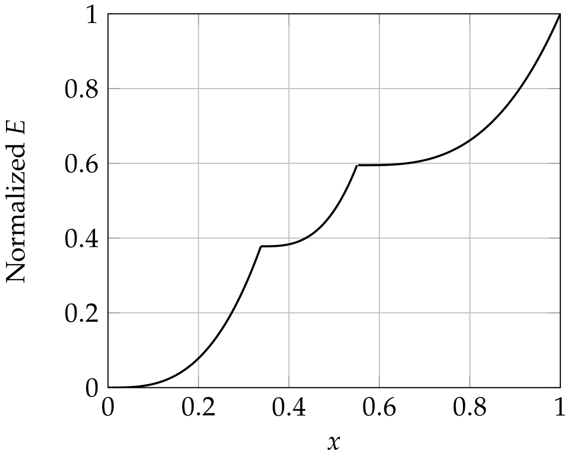


Figure 1.9: Ordered SIMP material interpolation for AlSi10Mg, Ti6Al4V, and stainless steel 316L. $p = 3$

In RMMI and DMO the number of design variables grows linearly with the number of candidate materials. In an effort to combat this Zuo and Saitou (2016) proposed the Ordered SIMP model: here the entire domain is defined by only a single density field, and candidate materials are ordered based on their normalized physical density. Intermediate densities are no longer penalized outright, but rather the densities in between the candidate materials are penalized instead. An example of this can be seen in Figure 1.9. While this does marginally reduce computational effort, it can cause convergence issues if the Young's modulus of the candidate materials does not increase together with their physical density.

1.4.3. Filtering and projection

General density-based TO suffers from two problems: checkerboard patterns with an artificially high stiffness are generated, shown in Figure 1.10, and there is no mesh independence in the solution. Thus the finer the mesh becomes, the more intricate the geometry of the optimized result will be.

The checkerboard effect is an artefact of the discretization of the design domain. Due to the nodal connectivity, these regions have the same stiffness as completely solid material, but only half its mass. This is not physically realizable, and thus action must be taken to remove or penalize this. A density filter is commonly used to average the element densities within a specified radius, which creates a "blurring" effect. This sets a minimum length over which the density can vary between zero and one, thus averaging out any checkerboard region in the design. Commonly used filters are the image-based filter by Bruns and Tortorelli (2001) and the PDE-based filter by Lazarov and Sigmund (2011). Several more filters, as well as some discrete methods, are discussed in the book by Bendsøe and Sigmund (2002).

Figure 1.10: Checkerboard effect.

The mesh dependence problem is partially solved by adding the filter, as density gradients that fall within the filter radius become averaged out. A consequence of this filtering however is that there will always be a small coating of intermediate density material between the part and void regions. To convert the filtered design back to a 0-1 design, a projection scheme can be used. Here the intermediate densities above a specified threshold are projected to one, and densities below the threshold are projected to zero. Projection is often added as an extra step in the optimization process to ensure optimality of the final design. For gradient-based TO it is thus required to be a continuous function. Guest et al. (2004) and Sigmund (2007) provided a normalized version of the projection method proposed by Kreisselmeier and Steinhauser (1983), but a later tanh-based Heaviside projection by Wang et al. (2011), given in Equation 1.8 and visualized in Figure 1.11, is the most commonly used method today.

$$\hat{x}_e = \frac{\tanh\left(\beta\eta\right) + \tanh\left(\beta\left(x_e - \eta\right)\right)}{\tanh\left(\beta\eta\right) + \tanh\left(\beta\left(1 - \eta\right)\right)} \qquad (1.8)$$

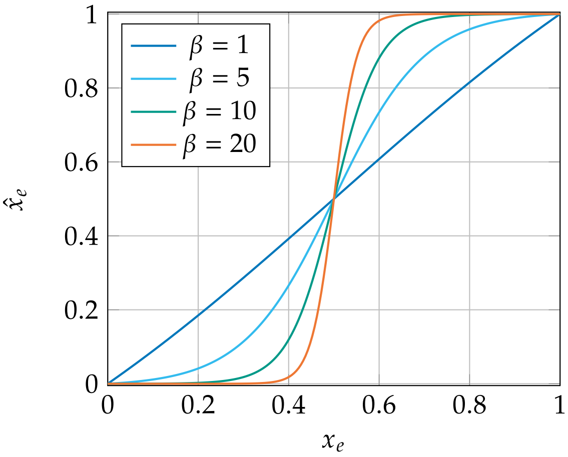


Figure 1.11: tanh-based Heaviside projection for various sharpness levels.

Since all production methods have a minimum feature resolution, it makes sense to ensure that the topology optimized designs explicitly adhere to this. Guest et al. (2004) proposed a method where a projection scheme with a shifted threshold is applied. This threshold dilates the design to enlarge thin features, which allows this dilated field to be producible. Sigmund (2007) proposed a method to ensure minimum-size for both the material and void regions by sequentially applying density filtering and projection with various thresholds. In a later paper, Sigmund (2009) also proposed a robust optimization method by running three projection schemes in parallel: one eroded, one nominal, and one dilated. The objective would then be based on the worst of the three, ensuring any thin and fragile geometry would be eliminated.

2

Literature Study

2.1. Introduction to manufacturability constraints

Design of multi-material parts via topology optimization had already been discussed by Bendsøe and Sigmund (2002) in their book on general topology optimization, but until recently the practical applications of MMTO were limited. At their time of publication there were simply no feasible manufacturing methods that could produce these multi-material designs, and most research focused either on SMTO, or on the production of discrete single-material parts that would then be joined together after production. Problems of interest during this time were tooling accessibility and the ability to assemble the parts.

In recent years additive manufacturing of multi-metal parts is becoming feasible for mainstream manufacturing, with multiple companies now offering machines capable of producing parts that blend or connect different metals during the build process (Aconity3D GmbH; Aerosint; DMG Mori). Consequently, there has also been a renewed interest in research regarding MMTO, especially as a design tool to create print-ready parts. Essential aspects include limiting the size and shape of the interface to what is acceptable, dealing with the lower interface strength, and adapting general AM limitations, such as enclosed void detection and overhang control, to work with multi-material design spaces. This chapter will provide an overview of the current state-of-the-art of manufacturability constraints. Existing constraints will be discussed and compared, including their adaptation into MMTO if applicable.

2.2. Enclosed void detection

Designs with enclosed cavities are generally not producible via PBF, as the unmelted powder has no means of being extracted after production. Additionally, these regions are inaccessible for post processing, which is often required for parts produced via WAAM. Liu et al. (2015) investigated a method to detect and remove these voids by running a conductivity problem in parallel with the standard FEA problem, which they called the virtual temperature method (VTM). In this conductivity problem, voids are given both a high thermal conductivity and a large heat input. Solid regions are given a very low conductivity, and the walls of the design domain are held at a fixed temperature. A virtual temperature field can then be calculated via a parallel thermal compliance FEA, which is solved via DMO. By limiting the maximum temperature of the domain, any enclosed voids can be removed, as these will have no means of quickly dissipating their heat. Voids that have a connection to the outer walls on the other hand will maintain a low temperature. By making the temperature limit more or less strict, the size of the interface between the void and the boundary can be controlled to some extent.

Yaghoobi et al. (2024) applied the VTM to MMTO. Their variant follows a very similar approach, but uses RMMI instead of DMO to perform the thermal analysis. No distinction between the material phases needs to be made, only between material and void. The conductivity of an element is found via Equation 2.1. This ensures that the conductivity is only non-zero for very low material densities. The heat flux follows a linear interpolation based on the element density, as seen in Equation 2.2.

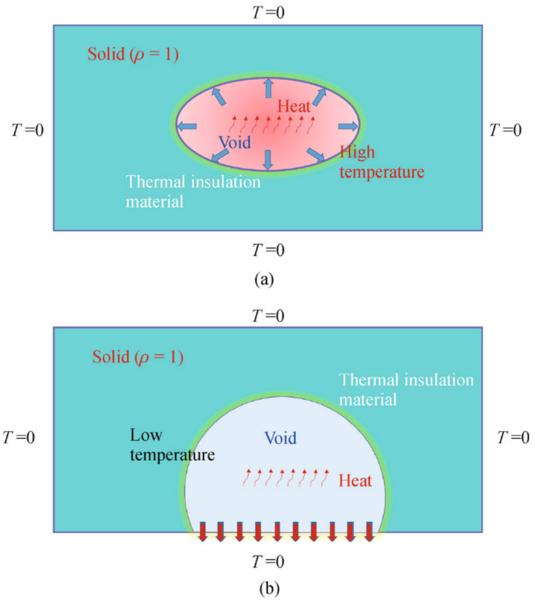


Figure 2.1: Virtual temperature method (Liu et al., 2015).

$$k^e = \left(k_{\min} + (1 - k_{\min})\left(1 - \sum_{i=1}^{m} \hat{x}_i^e\right)^q\right) k_0 \tag{2.1}$$

$$f^e = \left(1 - \sum_{i=1}^{m} \hat{x}_i^e\right) f_0 \tag{2.2}$$

VTM does not allow for much control over the shape of the void, and often leads to parts with a large interior cavity that only has a small connection to the boundary. Sabiston and Kim (2020) proposed an alternative diffusion-based approach to provide greater control over this. Rather than creating an opening, it penalizes interior cavities altogether. After each topology optimization iteration a particle is initialized in each element. These particles then travel in a specified direction with a velocity dependent on the material density: the denser the region it travels through, the slower it will move. This process is repeated for multiple directions, and the time to the boundary is stored for each, as shown in Figure 2.2 (C-F). The shortest diffusion time for each element, shown in Figure 2.2 (B), provides a measure of the surrounding material density. Long diffusion times indicate the presence of material in each direction, and thus a high likelihood of the element being part of an enclosed void. Since large areas of intermediate density will also generate a long diffusion time, density penalization is applied as shown in Equation 2.3. The corrected minimum diffusion time $\lambda_{c,e}$ is found by multiplying the original time with the penalized reverse density $(1 - x_e)^w$. By setting an upper limit on the diffusion time the design can be pushed to an outcome free of inaccessible interior geometry.

$$\lambda_{c,e} = \lambda_e (1 - x_e)^w \tag{2.3}$$

Xiong et al. (2020) proposed a method that aims to minimize the amount of material that has to be removed in order to get rid of enclosed voids. They made use of the discrete BESO approach, which does not require analytical sensitivities, and thus allows them to explicitly sort the elements into different sets, labelled as either material, boundary, or void. Then a connectivity graph is created for the nodes that indicate the connections to its neighbours. Each connection is given a weight, as shown in Equation 2.4. To improve computational efficiency this graph is created using a larger mesh as shown in Figure 2.3 (a). The cyan cells in Figure 2.3 (b) represent the nodes that lie within the part boundary, the blue cells represent the nodes on the part boundary, and the purple cells represent external voids. The yellow, red, and green cells represent the three internal voids. The connections between the internal voids and

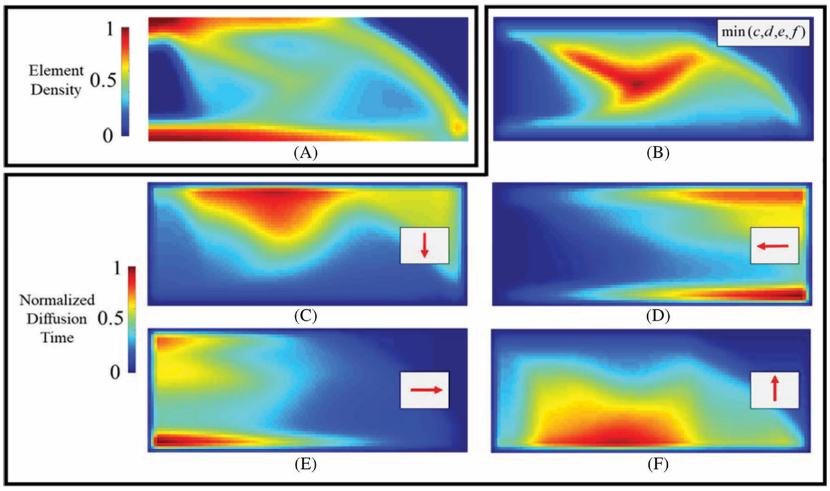


Figure 2.2: Material density field of an MBB beam (A), minimum diffusion time (B), and diffusion time fields for four principal directions (C-F) (Sabiston and Kim, 2020).

the boundary are then identified, and expanded into the original mesh as shown in Figure 2.4. The shortest paths between the voids and the boundary can then be found via the weights assigned. Voids can also be eliminated by connecting them to other voids, and thus these paths are identified as well. Multiple combinations of tunnels are possible, however to minimize the effect of the material removal on the structural performance of the part, the algorithm will only select the combination of tunnels with the lowest amount of removed material, while ensuring that all internal voids are eliminated.

$$w_{i,j} = \phi_i^{(2)} w_i^{(2)} + \phi_j^{(2)} w_j^{(2)}, \quad \text{with} \quad \phi_i^{(2)} = \begin{cases} 1, & \text{if } v_i^{(2)} \in V_\Gamma^{(2)} \\ 0.5, & \text{if } v_i^{(2)} \in V_\Omega^{(2)} \\ 0, & \text{otherwise} \end{cases} \tag{2.4}$$

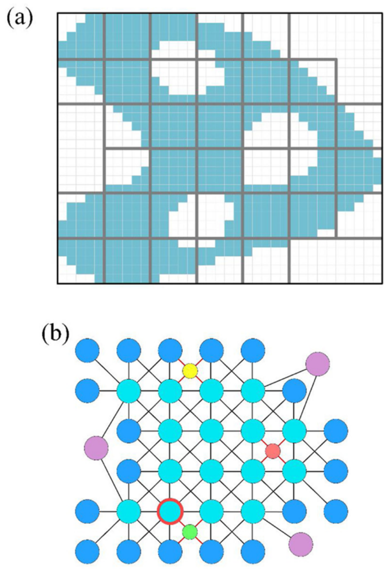


Figure 2.3: Part with larger mesh (a) and connectivity graph (b) (Xiong et al., 2020).

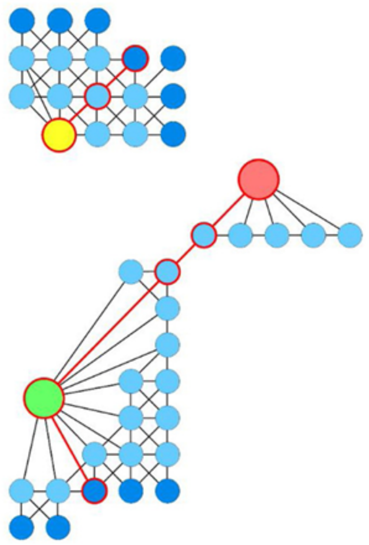


Figure 2.4: Shortest path from internal void to boundary (Xiong et al., 2020).

van der Zwet et al. (2023) proposed a cumulative flood-fill method that fills any enclosed void with material. The algorithm loops over the low-density elements in the design domain, starting with the boundary elements. For the element i in the queue that has the lowest density, the adjacent element j with the lowest density is added to the queue as well, and the flooded density ξ_i of the already processed element i is added to the unflooded density ρ_j of element j to find the flooded density ξ_j:

$$\xi_j = \rho_j + \xi_i \tag{2.5}$$

After which the process is repeated with the element in the queue that at this point has the lowest density. The flooded density ξ of the boundary is initialized as the unflooded density ρ, as they are by definition in contact with the edge of the design domain. The process ends when the queue is empty, indicating that there are no low-density elements in the domain that are fully enclosed by higher density elements. Since the flooded density field will often have densities greater than one, a smooth minimum projection is applied to bring the densities down below one:

$$\psi_i = \left(\xi_i^{-q} + 1\right)^{1/q} \tag{2.6}$$

The net effect of this filter is that it forces all cavities that do not have a connection to the outside void to be filled with material, since the flooded density of the most near-void element in its path gets added to the elements in the cavity. The optimizer is then forced to either fill the enclosed void, or to provide a path towards the outside boundary.

2.3. Overhang filters and constraints

A significant limitation that almost all additive manufacturing processes have is that they cannot print steep overhang angles without supports. Initial investigations into constraining the overhang in topology optimization were done by Brackett et al. (2011) and Leary et al. (2014). However, Gaynor et al. (2014) were the first to propose a functional strategy for avoiding overhangs. Their method projects a wedge downwards with a specified radius and aperture, which have been specified beforehand. The average density of the elements within the wedge is calculated, and if this exceeds a certain threshold the element being considered is said to be properly supported. The algorithm was found to make heavy use of intermediate densities however to generate support material, which is not physically realizable. They were able to combat this to an extent with a threshold function, which required that multiple elements within the support region are non-zero.

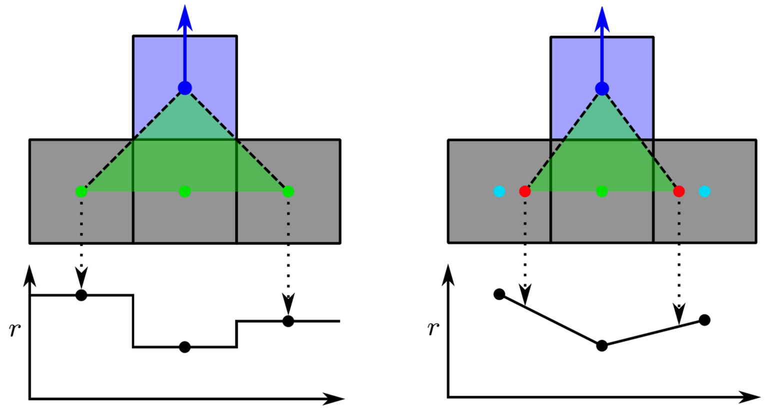


Figure 2.5: Original overhang filter by Langelaar (2016a) (left) and modified density interpolation scheme by Delissen et al. (2022) (right). Figure from Delissen et al. (2022).

Langelaar (2016a,b) proposed a filter that could limit the overhang angle of a design to 45°. This is achieved by mapping the material density x to a new printed density ξ. First a support region $S_{(i,j)}$ is defined for each element (i, j) in the mesh, consisting of both the element directly below as well as those adjacent to it. Secondly the highest material density value of these elements is stored in the variable $\Xi_{(i,j)}$. Finally the material density of element (i, j) is filtered as seen in Equation 2.8, after which the printed element density $\xi_{(i,j)}$ is obtained. If any of the elements in the support region is anything other than void, element (i, j) can become filled in with that same density without the overhang exceeding 45°. Changing the maximum overhang angle is theoretically possible by changing the aspect ratio of the mesh, but this quite impractical in a lot of cases as it also makes the minimum member size filter anisotropic. The p-norm function in Equation 2.8 will slightly overestimate the maximum, and thus a slow build-up of densities occurs across the layers of the part. This can be avoided by either increasing the value of the ξ_0 parameter through the iterations, or by projecting the density field with the Heaviside equation.

$$\xi_{(i,j)} = \frac{1}{2}\left(x_{(i,j)} + \Xi_{(i,j)} - \left(\left(x_{(i,j)} - \Xi_{(i,j)}\right)^2 + \epsilon\right)^{1/2} + \sqrt{\epsilon}\right) \tag{2.7}$$

$$\Xi_{(i,j)} = \left(\sum_{k=1}^{n_S} \xi_k^P\right)^{1/Q}, \qquad Q = \frac{\log(n_S)}{\log(\xi_0)} \tag{2.8}$$

Hoffarth et al. (2017) investigated the implementation of this filter in the commercial Tosca structure. This included support for irregular meshes, material and geometric non-linearities, and critical angles other than 45° via a cone-shaped control volume. No explanation of the implementation of these methods was provided in this paper. Langelaar (2018) proposed an update to the filter that also supported irregular mesh as well as sacrificial support structure optimization. This is achieved by mapping the density field from the irregular FEA mesh to a separate background mesh. The overhang angle filter can then be applied to this background mesh, after which it is mapped back to the original FEA mesh. As the build plate mesh is now fully separate from the actual part, this also allows the part to be rotated in the mapping stage. Part orientation on the build plate can now be included in the optimization process. Some care is needed with this method however as the mapping process introduces a slight smoothing of the density field.

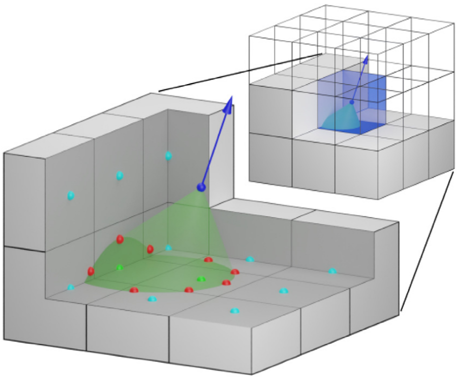
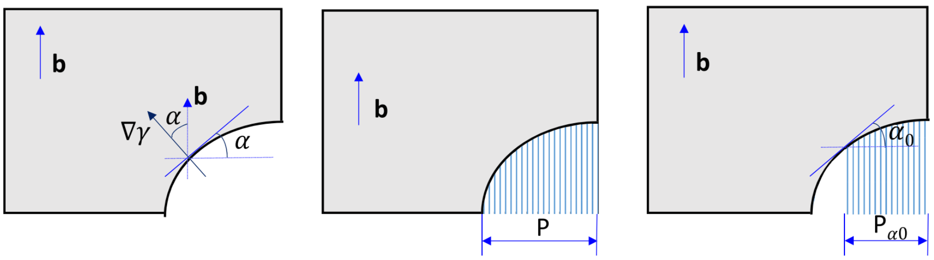


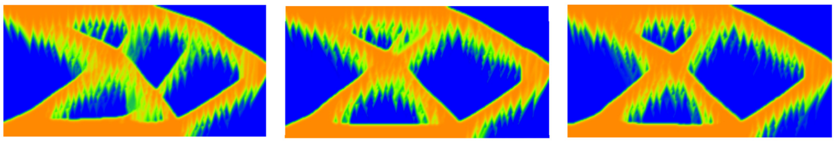
Figure 2.6: Cone projection method (Delissen et al. (2022)).

Delissen et al. (2022) added functionality for different overhang angles without modifying the aspect ratio of the mesh. This was achieved by interpolating the densities of the support region at a location in the element, rather than at its centre, as seen in Figure 2.5. This enhancement also supports build directions that are not aligned with the mesh, which is achieved with by projecting a cone with the specified overhang angle as its aperture in the negative build direction, similar to the method proposed by Gaynor et al. (2014) and Hoffarth et al. (2017). A visualization of this is provided in Figure 2.6. All elements whose centre is within this cone are included in the support region, and a specified number of points on the edge of the cone are used via interpolation. An additional improvement is that the overhang angle can now also be accurately limited in every direction, whereas before this was only possible in the directions that aligned with the mesh. With the prior approach the shape projected was not a cone but rather a pyramid. In the diagonal axes the angle would be either overestimated, if a 5-element support region is used, or underestimated, if a 9-element support region is used. Limiting to angles less than 45° is still troublesome with this filter, as all the elements in the support region have to already be processed by it.

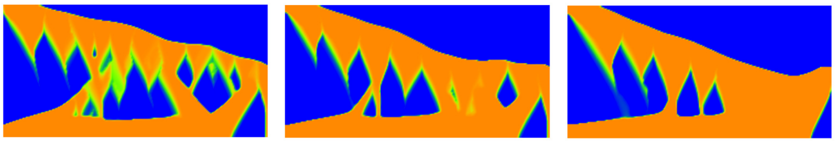


(a) Overhang angle α (b) Projected undercut perimeter P (c) Projected perimeter P_{α_0}

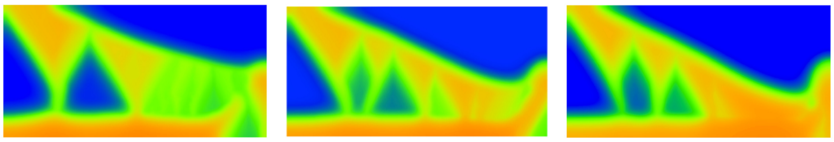
Figure 2.7: Undercut detection method (Qian, 2017).



(a) $q = 3$, c=74, $\varepsilon = 0.34$ (b) $q = 5$, c=85, $\varepsilon = 0.29$ (c) $q = 7$, c=91, $\varepsilon = 0.28$



(d) $q = 3$, c=79, $\varepsilon = 0.16$ (e) $q = 5$, c=87, $\varepsilon = 0.09$ (f) $q = 7$, c=91, $\varepsilon = 0.08$



(g) $q = 3$, c=137, $\varepsilon = 0.6$ (h) $q = 5$, c=227, $\varepsilon = 0.5$ (i) $q = 7$, c=311, $\varepsilon = 0.46$

Figure 2.8: Final designs using the undercut constraint, various filter radii (Qian, 2017).

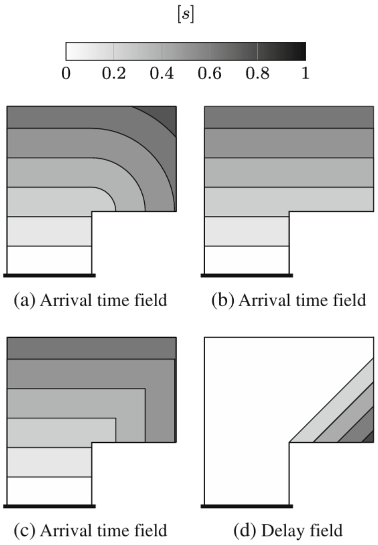
Qian (2017) proposed an alternative formulation for the overhang constraint. Instead of a filter he limits the area of the undercut region of a design. This undercut region is defined using the overhang angle, which can be calculated by taking the dot product of the build direction $\boldsymbol{b}$ and the gradient of the density field $\nabla\tilde{\gamma}$, as seen in Figure 2.7. To find the undercut angle the part boundary needs to be clearly defined. Therefore the dot product is multiplied with a Heaviside projection of itself, as seen in Equation 2.9. This ensures that the material gradient is only non-zero at the part boundary. Furthermore, it guarantees that only undercuts have a non-zero value: when the gradient is negative, such as on the top surface of a part, its Heaviside projection equals zero, and the region is not included. The undercut regions are projected onto the build plate, and this area can then be limited. Since the above implementation constrains the density gradient it could lead to designs with regions of constant intermediate densities. To avoid this an extra constraint is added which limits the amount of grey material.

$$\int_{\Omega} H\left(\boldsymbol{b} \cdot \nabla\tilde{\gamma}\right) \boldsymbol{b} \cdot \nabla\tilde{\gamma} \, \mathrm{d}\Omega \leq \bar{P} \tag{2.9}$$

The equation above can be expanded to only limit the projected undercut area beyond the threshold angle α_0 by projecting with a shifted Heaviside function $H_{\bar{\alpha}}$. $\bar{\tilde{\gamma}}$ represents the density field after the standard Heaviside projection, which is still present. The gradient is also normalized through dividing it by $||\bar{\tilde{\gamma}}||$. The shifted Heaviside projection ensures that the transition from zero to one occurs at $\cos\left(\bar{\alpha}\right)$. The new constraint formulation is given in Equation 2.10.

$$\int_{\Omega} H_{\bar{\alpha}} \left(\boldsymbol{b} \cdot \frac{\nabla\bar{\tilde{\gamma}}}{||\nabla\bar{\tilde{\gamma}}||_2}\right) \boldsymbol{b} \cdot \nabla\bar{\tilde{\gamma}} \, \mathrm{d}\Omega \leq \bar{P}_{\bar{\alpha}}, \qquad \bar{\tilde{\gamma}} = H\left(\tilde{\gamma}\right) \tag{2.10}$$

Similar to the filter proposed by Langelaar this method supports build directions that are not aligned with the mesh, though no numerical implementation of this was provided. It is also capable of limiting



(a) Arrival time field (b) Arrival time field

(c) Arrival time field (d) Delay field

Figure 2.9: Time arrival field from velocity function f_0 (a), f_1 (b), f_2 (c), and delay field between f_1 and f_2 (d) (van de Ven et al., 2018).

to angles less than 45°, which is something the filter proposed by Langelaar struggles with. The main drawback of this constraint is that it will allow boundary oscillations as shown in Figure 2.8. These designs are not printable due to the presence of unsupported geometry. This is a result of the local nature of the filter, and as of yet no filtering or projection method has been able to consistently generate designs without this boundary oscillation.

van de Ven et al. (2018, 2020) proposed an overhang filter based on front propagation. This approach aims to restrict the overhang angle of a part by defining a time delay field between two front propagation methods. By defining a direction-dependent velocity functions for each front they were able to create a field where the only non-zero values occur in the regions that are not manufacturable. If a curve or surface is initialized at the build plate and propagates through the part with velocity f_0, it will propagate as in Figure 2.9 (a), where the arrival time is plotted over the part domain. By modifying the velocity based on the propagation direction the minimum time arrival field in Figure 2.9 (b) can be generated. Here the velocity is set to f_0 when parallel to the build direction, and goes to infinity perpendicular to the build direction. The relation for this is given in Equation 2.11.

$$f_1\left[\boldsymbol{a}\right] = \frac{f_0}{\boldsymbol{a}\cdot\boldsymbol{b}} \tag{2.11}$$

A third time arrival field, plotted in Figure 2.9 (c) is generated where the speed is lowered in the direction which is less than the critical overhang angle α_{oh}. The relation for this is given in Equation 2.12. By subtracting the first field from the second field one obtains a time delay field. The time delay will remain zero in regions where the overhang angle is less than the critical value, and the delay will only be greater than zero in the non-manufacturable regions. Downwards propagation is included in the non-manufacturable region by default, as there is no direct path from the build plate to said region. The delay field can finally be used as a filter to convert the density field into a printed density field, similar to the Langelaar filter.

$$f_2\left(\boldsymbol{a}\right) = \frac{f_0}{\max\left(\tan\left(\alpha_{\text{oh}}\right)||\mathbf{P}\boldsymbol{a}||,\ |\boldsymbol{a}\cdot\boldsymbol{b}|\right)} \tag{2.12}$$

2.4. Hotspot detection

Besides the material not being supported, another reason for not being able to print steep overhanging surfaces is to prevent overheating. The heat generated by the laser can only flow away from the input through the part itself, since both the surrounding air as well as the unmelted powder have a much lower conductivity. In regions where there is a steep overhang there is no direct connection to the build plate, and thus a hotspot occurs. This leads to poor surface finish and warping (Ranjan et al., 2017). The exact value of the overhang angle generally lies between 20° and 65°; however, the exact value depends on both the material and the machine used (Hussain and Kim, 2025). Besides the angle with respect to the build direction, the angle with respect to the recoater direction is of importance as well when PBF is used to produce the part. Meyer et al. (2023) investigated the production quality of various multi-material designs, and developed a set of guidelines based on the results they obtained. They discovered that the material interfaces should be aligned with the build direction as much as possible. This promotes interfacial diffusion of the different materials, which increases the strength of the interface and reduces porosity. They also found that the angle of the longest axis of the part with respect to the recoater should ideally be kept at 30° or lower to minimize the forces exerted on it. Liu et al. (2018) found results that conflict with this. Their experiments showed that parts with the long axis parallel the recoater had a larger chance of failure, as thermally-induced distortions caused the roller to collide with the part. They believed the thermal distortion of each part to be similar in magnitude, as process parameters such as laser power and scanning strategy were kept the same for all parts. The difference in outcome could potentially be explained by the part geometry or materials used. Meyer et al. (2023) produced multi-material cubes from 316L stainless steel and CuCrZr, while Liu et al. (2018) produced a version of the MBB-beam out of 17-4PH stainless steel. Further research would thus be required to establish the cause of this discrepancy.

In the previous section the overhanging features were always constrained to a fixed angle, which at times might be too strict, depending on the surrounding geometry. A more physics-based approach to the overhang constraint was first proposed by Ranjan et al. (2017). They argued that local overheating is the primary reason for not being able to print overhangs. Since loose powder has a significantly lower thermal conductivity, steep overhangs and floating features cannot dissipate their heat sufficiently quickly to the build plate, and a hotspot emerges. According to their view, simply constraining the overhang angle was not sufficient to avoid overheating, and these filters are at the same time too restrictive, leading to suboptimal designs. A physics-based hotspot detection method was described in detail by Ranjan et al. (2022, 2023) in later papers. In this approach the design is subdivided into a series of overlapping slabs, each one representing a build layer during the production process. A steady-state thermal analysis is then performed to find the temperature field. Heat flux and boundary conditions are applied as shown in Figure 2.10 (b), mimicking the AM production process. Hotspots can be found and constrained by taking a smooth maximum of the temperatures in each slab. Validation has shown that this filter is effective in reducing hotspots during the production process. Ranjan et al. (2022) also found that the thickness of the slab significantly affected the outcome, and thus it should be chosen carefully. In his experiments he chose the thickness of the slab to be equal to the estimated thermal interaction length; however, he notes that this length is likely not constant throughout the part, and more research is needed to make a better estimate.

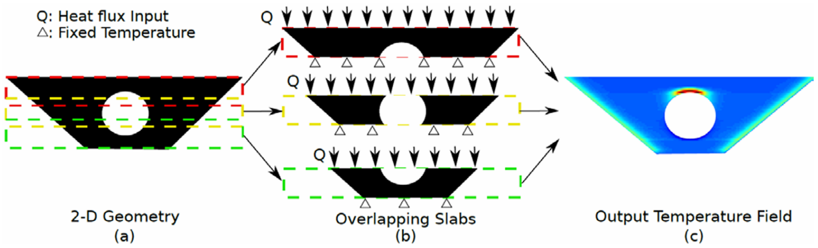


Figure 2.10: Hotspot detection method by Ranjan et al: (a) Overlapping slabs (b) thermal analysis (c) resulting temperature field (Ranjan et al., 2017).

Das (2023) in his thesis, and later in a paper (Das et al., 2025), proposed an alternative geometry-based approach to hotspot detection, with the aim to improve computational efficiency. In this method the conductivity is not estimated via static FEA but rather through an estimation based on the local geometry. A hemispherical region with a radius equal to the mean conduction length acts as the region that forms the basis for a conductivity estimation. The constraint proceeds through the design domain layer by layer in the build direction to create a virtual conductivity field. With both this and the density field, an approximate temperature field can be calculated, the maximum of which forms the input for a constraint. The contribution of elements within the conduction radius is weighted by both a radial and an axial weight function, given in Equation 2.13. This ensures that elements closest to the one being considered contribute the most, and also that elements that are nearby but not thermally connected do not significantly contribute. Experiments by Das (2023) showed that both the slab analysis by Ranjan et al. (2022) and his geometric estimation provide significantly different temperature fields than a fully transient analysis. However, as Das (2023) notes, most of the difference occurs in the lower temperature region. both methods are still able to predict critical hotspot zones accurately. His experiments also showed that the geometric approach reduced average computation time from 1.34 s to 1.10 s per iteration, an improvement of approximately 18%. He also notes that improvements in the code can be made to increase this even further. The overhang filter by Langelaar (2016a) is still significantly faster than both, requiring only 0.35 s per iteration.

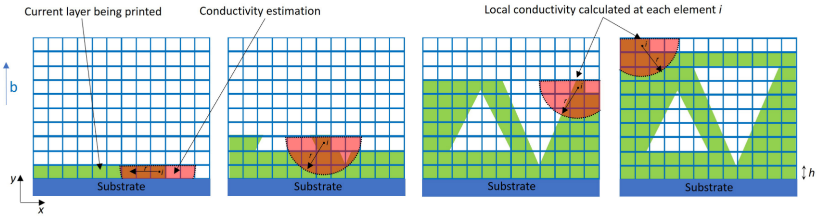


Figure 2.11: Geometric conductivity estimate (Das, 2023).

$$wr_j = r - \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2}, \qquad wa_j = \frac{1}{\psi} \arctan\left(\frac{y_i - y_j}{x_i - x_j}\right) \tag{2.13}$$

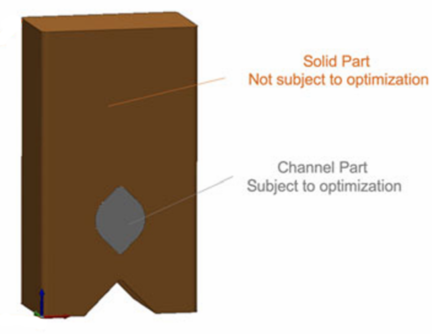


Figure 2.12: Cooling channel optimization (Iyer et al., 2024).

Iyer et al. (2024) proposed a framework to predict crack initiation during the build and release process. Cracks form as the result of large plastic strains forming from steep thermal gradients. Avoiding these is critical to designing reliably producible parts. They found that the maximum shear strain index (MSSI) gave a reliable indication of regions that were at risk of crack initiation, and by minimising the MSSI they were able to produce geometries with a significantly lower risk of developing cracks during production. The MSSI however requires knowledge of the residual stress and strain fields after the production process, and these can normally only be found either via experimental calibration or via transient multiphysics simulations. Experimental calibration is only valid for a specific combination of material, geometry, and process parameters, and thus not suitable for use in a design process. Transient simulations are very time and energy intensive, and extracting design sensitivities from them is often not possible, thus they also cannot be feasibly implemented in topology optimization. Iyer et al. (2024) chose to use a physics-informed neural network instead, which was trained on the results of multiphysics simulations for various geometries similar to that of a typical output of the example problem. They were able to significantly reduce the computation time of the optimization process, however as the neural network was trained exclusively on various outcomes of the specific design problem, it would require retraining for every significantly different problem encountered. Furthermore, it was only tested on the relatively straightforward cooling channel optimization problem shown in Figure 2.12. It is unclear how it performs on more complex problems with a larger number of design variables.

2.5. Other manufacturability considerations for MMTO

2.5.1. Accessibility and assembly

Feng and Yamada (2024) investigated constraints regarding the assembly and size of a design made up of multiple individually printed parts. The proposed applications for their constraints were both large-scale parts that exceed the build volume of a typical 3D printer, and multi-material assemblies where each material phase has to be printed separately. The first constraint they described was an upper limit on the principal dimensions of the bounding box of each phase. Via the oriented bounding box method (Feng et al., 2023) they first found the eigenvectors of this bounding box, which provided its principal directions. Then via the weighted centroid and p-norm the dimensions of the box itself can be found. Finally these dimensions form the input for an inequality constraint. The constraint itself does not enforce the connectivity of the material phase, and thus there can be multiple individual parts that belong to the same phase. These are still considered to belong to the same bounding box, which significantly limits the design freedom.

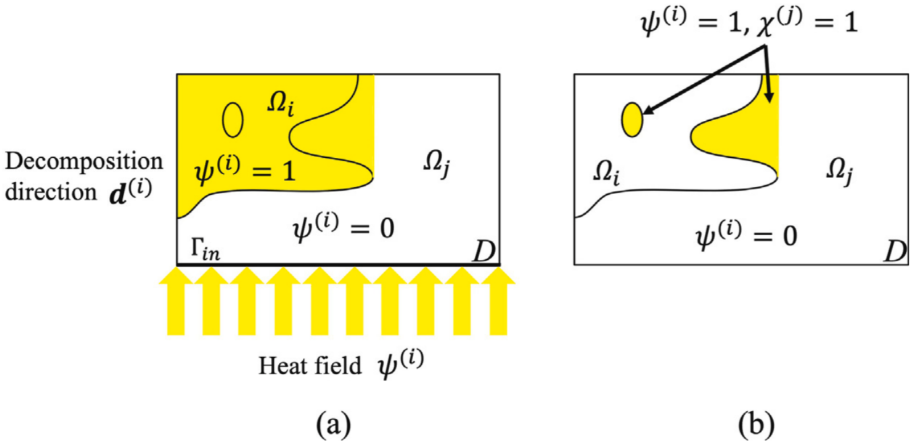


Figure 2.13: The fictitious heat field (a) and the extracted unprintable geometry (b) (Feng and Yamada, 2024).

The second constraint they proposed was a filter that ensured that the different pieces of the assembly can still be joined together after production. The approach to this was similar to the tooling accessibility as proposed by Langelaar (2019). It acts as a filter that detects undercuts and interior discontinuities and then removes them. However, as Feng and Yamada (2024) use the level set method instead of an density-based method such as SIMP, the implementation has to be modified. Both undercuts and interior discontinuities are detected by a heat field $\psi^{(i)}$ that travels in a specified direction. The heat field only passes through material i and any material downstream of it, as seen in Figure 2.13. The heat field domain $\psi^{(i)} = 1$ then becomes the new domain for material phase i. In their implementation the assembly direction and order has to be defined before the optimization itself starts, and as a result the performance of the final design heavily depends on how this is set by the user. It will not always be clear beforehand what orientation will yield the best result, which means multiple attempts are often required. More complicated assembly steps such as rotations are also not included, and could be an area of future research. However, as Zheng et al. (2021) noted, the type of MMAM where different material phases have to be assembled after production is quickly becoming irrelevant as the industry improves.

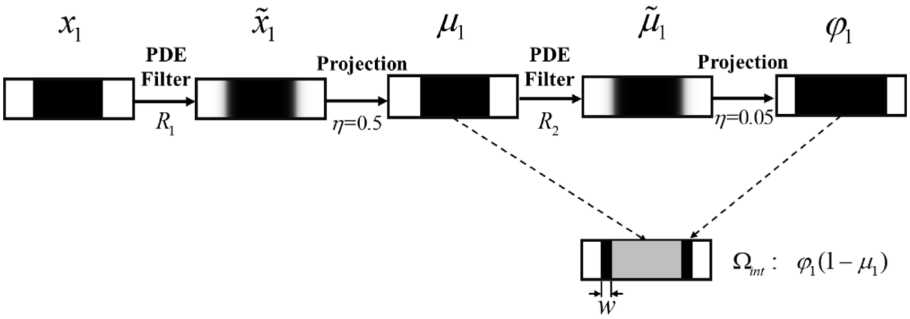


Figure 2.14: A schematic diagram of the interface detection algorithm (Hu et al., 2022).

2.5.2. Interface strength

One of the main limitations of multi-metal AM is that the interface strength is generally significantly lower than that of the pure material phases, either due to poor interfacial bonds or due to residual stresses occurring through a difference in thermal expansion (Zainelabdeen et al., 2024). Thus when designing a functional multi-material part it is essential that the stresses on the interface are sufficiently limited. Hu et al. (2022), proposed a method to extract the subset of elements that belong to a material interface by making use of PDE filters and Heaviside projection. This process is shown in Figure 2.14. First the density field x_1 is filtered and projected to make the design mesh-independent. This new field $\tilde{x}_1$ is then filtered once more and projected with a diluted version of the Heaviside equation to generate the field φ_1. By multiplying φ_1 and the reverse of μ_1 a new density field can be created where only the interface elements equal one. The stress tensors in these elements were then converted to effective normal and shear stress measures, which in turn formed the basis for the failure criteria constraint.

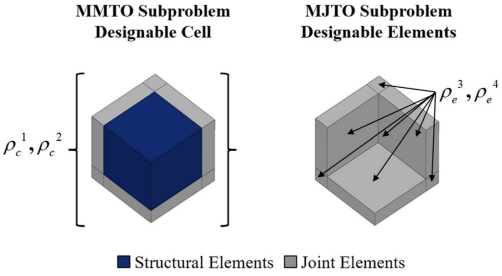


Figure 2.15: Mesh with interface sub-elements (Florea et al., 2019).

Florea et al. (2019) proposed a method to apply specific material properties to only the interface region. Their interest was not in MMAM in the strictest sense but rather in assemblies of topology optimized parts that have to be joined after production. The purpose of modifying the material properties at the interface was to represent different kinds of joints such as welds or adhesive bonds, and to apply constraints to the accessibility of these interfaces. Their optimization algorithm was split into two sub problems. First a general topology optimization was performed, and then the interface joints were added and optimized. They used a specialized mesh originally proposed by Zhou and Saitou (2018), which is irregular but periodic. Each element is split into eight sub-elements: one main structural cell and seven joint elements, as shown in Figure 2.15. The main elements were described by the RMMI model, using design variables ρ_1 and ρ_2, and the joint elements would be described by a new set of design variables ρ_3 and ρ_4. ρ_3 determines the existence of a material interface joint, and equals one only in the presence of a material interface. ρ_4 is equivalent to the mixture parameter as used in the RMMI model, but in this case it defines the type of joint present. Through multiple iterations of the optimization process a design results which takes into account the distinct stiffness of the joint. If the Young's modulus of the interface is set to be significantly lower than that of the pure material phases it will lead to a design with a minimal interface.

2.5.3. Interface size and shape

López et al. (2020) conducted a very comprehensive investigation into multi-material constraints. They proposed six constraints in total, many of which had not yet been implemented before. This review will discuss only those that are relevant for additive manufacturing. Since they used the ordered SIMP scheme, they must first map the single design variable ρ_e to a new variable that equals one if and only if the element belongs to material M_1. A threshold function f based on the error function is used to achieve this. The continuous nature of this function allows sensitivities to be calculated.

Their first design rule is a limit on the number of individual material islands. This is especially relevant for multi-material PBF designs, as a large number of small islands significantly increases the risk of powder contamination between the different materials. Their algorithm limits the islands by going over all the elements in the design space, and then counting its neighbours of the same material phase. If one or more neighbours with the same material phase is found, the current element is given the same label as its neighbours. If no matching neighbour is found the element is given a new label. This index system then forms the basis for an inequality constraint. The sensitivity for this constraint is calculated using the derivative of f, as shown in Equation 2.14, and depends on the number of neighbours with the same material, n_n.

$$\frac{\partial n_I}{\partial \rho_e} = \begin{cases} 2f\frac{\partial f}{\partial \rho_e}, & \text{if } n_n = 0 \\ \left(1-2f\right)\frac{\partial f}{\partial \rho_e}, & \text{if } n_n = 1 \\ -2\left(n_n - 1\right) f \frac{\partial f}{\partial \rho_e}, & \text{if } n_n > 1 \end{cases} \tag{2.14}$$

Their second design rule, shown in Equation 2.15, can limit the size of the material interface s_e to a specific value. This constraint algorithm loops over the six neighbouring elements of element e, and filtering out the elements that belong to material M_1, again with f. The selector function $c\left(e, e_n\right)$ then filters out only the interfaces between materials M_1 and M_2. This function is left discrete, which means that the sensitivity of s_e is a local sensitivity that assumes $c\left(e,\ e_n\right)$ is constant.

$$s_e\left(\rho\right) = S_e \sum_e f\left(\rho_e\right) \sum_{e_n} \left(f\left(\rho_{e_n}\right) - f\left(\rho_{e_n}\right)\right) c\left(e, e_n\right) \tag{2.15}$$

Liu et al. (2024) provided a minimum-size constraint for MMTO which also penalized the interface surface area. They did not use the common density filter as described by Bendsøe and Sigmund (2002) but rather created an adaptation for a discrete RMMI material model. Their constraint consists of a weighted sum of all elements within a specified radius, provided in Equation 2.16. $\phi_i^{E_m}$ is only non-zero if element E belongs to material phase m. The value of the local average $\phi_i^{\sim E_m}$ will equal one if and only if all elements within radius R belong to the same material phase, otherwise it will be less than one. By constraining the sum of the local averages the minimum member size can be enforced. Elements at the edge of the material-void boundary per definition cannot have this sum be equal to one, thus the constraint in these elements is relaxed. Elements on the interface between different materials however remain in the constraint in full, which means that the larger the surface area of a material interface is, the more it contributes to the constraint penalization. This significantly reduces the size of the interface, which is claimed to improve manufacturability. Setting a hard upper limit on the size of the interface, or removing it altogether, is however not possible. It is also not possible to distinguish between different interfaces when dealing with a design problem consisting of three materials or more.

$$\phi_i^{\sim E_m} = \frac{\sum_{f\in NR_i} v_f \phi_i^{E_m}}{\sum_{f\in NR_i} v_f}, \quad NR_i = \left\{f | \operatorname{dist}\left(i, f\right) \le R\right\} \tag{2.16}$$

Since they make use of the RMMI model, constraint sensitivities are required. The gradient of Equation 2.16 is provided in Equation 2.17. H_f is a selector function that equals one if the neighbouring element in question lies within the specified radius. $\vec{d}_{i,f}$ is the vector from element E to the neighbouring element, and $\vec{e}_x$ and $\vec{e}_y$ are the horizontal and vertical unit vectors respectively.

$$\frac{\partial \phi_i^{\sim E_m}}{\partial x} = \frac{\sum H_f v_f \phi_f^{E_m}\left(\vec{d}_{i,f}\cdot\vec{e}_x\right)}{\sum H_f v_f}, \quad \frac{\partial \phi_i^{\sim E_m}}{\partial y} = \frac{\sum H_f v_f \phi_f^{E_m}\left(\vec{d}_{i,f}\cdot\vec{e}_y\right)}{\sum H_f v_f} \tag{2.17}$$

2.5.4. Material preselection

In their multi-material approach, Bendsøe and Sigmund (2002) applied an equality constraint to the volume fraction of each material. Limiting the phase of each material to a specific amount might lead to suboptimal results if it is not necessary to be so strict with the material fractions. Since then many researchers have opted to use a total mass constraint instead, but López et al. (2020) proposed an additional constraint in the form of an upper limit on the volume fraction of each individual material phase. This volume fraction is calculated by summing up the volumes of all the elements belonging to a certain phase. Which phase an element belongs to is again determined by the approximating function.

López et al. (2020) also proposed a material preselection algorithm. Materials are ranked based on their density and their stiffness, and can then be categorized as either dominant or non-dominant. Non-dominant materials can be discarded as they have no additional value for the problem at hand. This works for relatively simple optimization problems. When there are multiple response functions, however, such as in multiphysics problems, determining which materials are non-dominant becomes very challenging. Liu et al. (2024) also investigated the question of how to determine if a material would be dominant in a design problem. They specifically focused on what they called the coexistence and degradation of materials. In a general compliance minimization problem with only a total mass constraint, often used in industry, tends to favour solutions only using the material with the highest specific stiffness. This led to claims of the specific stiffness to be the correct criterion for determining whether a material should be preselected or not (Huang and Li, 2021). Liu et al. (2024) however claimed that this was not correct, and that the necessary conditions for material coexistence were more specific than that. Through experimentation they were able to show that designs which incorporated a second material with a lower specific stiffness could still achieve a lower compliance. They then aimed to find an empirical relation between the stiffness and density ratios of a candidate material that could determine whether it was dominant or not. Their results, shown in Figure 2.16, proved that there was a region in the material property domain where a multi-material design still has added value over a single material design, which was not covered by the initial criterion.

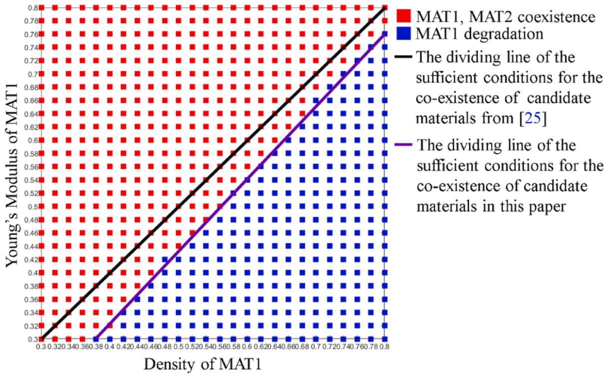


Figure 2.16: Material degradation diagram (Liu et al., 2024).

From these results they proposed an empirical relation. However further experiments by the author of this thesis have shown that this heavily depends on the solver settings and specific design problem. Their relation was derived purely from the results of the MBB-beam analysis, and thus cannot be used as a general criterion for material preselection. Furthermore, all numerical examples in the paper by Liu et al. (2024) have as their initial condition a design space filled completely with the heaviest material, instead of an equal distribution of all material phases. This introduces a bias towards the heaviest material.

2.6. Summary of existing methods

This chapter aims to provide an overview of the current state of the art in manufacturing constraints for multi-material AM, and to detect areas of further possible research. The existing MMTO constraints can be divided into three different classes. The first class contains SMTO manufacturability schemes that have been adapted to MMTO. The second class contains methods that focus on the interface behaviour specifically, such as its size, shape, or stress limit. The third class contains the methods that are aimed at conventional manufacturing techniques, such as tooling accessibility or assembly. The first class includes the enclosed void detection constraints as well as the overhang and hotspot methods. Considering the scope of this thesis only the first and second classes are of interest.

Interface-specific properties such as a modified Young's modulus has already been investigated by Florea et al. (2019). Improvements in this area can be made by means of enriched FEM, however this would require rigorous experimental testing to provide the inputs to a more detailed physical model. Additionally enriched FEM is not compatible with density-based topology optimization but instead requires the level set method. Therefore this is considered to be beyond the scope of this project. Most essential work regarding the size of the interface has already been carried out by López et al. (2020), and finally the material preselection is likely to remain in control of the designer. The optimum selection of materials is too reliant on the problem at hand, and thus universal selection criteria are hard if not impossible to formulate.

Enclosed void detection has already been successfully applied to MMTO by Yaghoobi et al. (2024), and did not need significant changes to be functional. Overhang constraints on the other hand require extra effort to successfully accommodate multi-material problems, especially if the critical overhang angle is different for each candidate material. While Langelaar (2018) did mention his filter is compatible with MMTO, no detailed implementation of this was provided, nor did it include support for custom overhang angles at this time. The latest iteration of the overhang filter by Delissen et al. (2022) is in the opinion of the author the most suitable variant to be adapted to MMTO, as the front propagation filter will struggle when the front travels through multiple material phases. The hotspot detection filters by Ranjan et al. (2023) and Das et al. (2025) provide novel physics-based approaches to solving the overhang problem, and as of yet they have not been applied to MMTO either. These can make use of the additional design freedom provided by multi-material, for example through the strategic placement of high-conductivity material to make certain previously infeasible features or overhang angles manufacturable. Therefore both Langelaar's overhang filter, and the hotspot constraint by Das et al. have been chosen to be adapted to MMTO. In the tables below an overview of all the constraints discussed is provided.

Author	**Type**	**usable in MMTO?**	**TO Method**
Liu et al. (2015)	Virtual temperature method	yes	SIMP
Sabiston and Kim (2020)	Diffusion	yes	SIMP
Xiong et al. (2020)	Tunnelling	yes	BESO
Yaghoobi et al. (2024)	Virtual temperature method (MM)	yes	RMMI
van der Zwet et al. (2023)	Cumulative flood fill	yes	SIMP

Table 2.1: Summary of existing methods for removing enclosed voids.

Author	**Type**	**usable in MMTO?**	**TO Method**
Das et al. (2025)	Hotspot detection (geometric)	no	SIMP
Delissen et al. (2022)	Overhang angle (custom build dir.)	no	SIMP
Feng and Yamada (2024)	Undercut detection	yes	Level set
Gaynor et al. (2014)	Overhang angle (wedge)	no	SIMP
Hoffarth et al. (2017)	Overhang angle (custom angle)	no	SIMP
Langelaar (2016a,b, 2018)	Overhang angle (45° only)	no	SIMP
Qian (2017)	Undercut detection	no	SIMP
Ranjan et al. (2022)	Hotspot detection (slab)	no	SIMP
van de Ven et al. (2018, 2020)	Overhang angle (front prop.)	no	SIMP

Table 2.2: Summary of existing methods for limiting the overhang angle.

Author	Type	usable in MMTO?	TO Method
Iyer et al. (2024)	Crack detection	no	SIMP
Liu et al. (2024)	Minimum size (MM)	yes	Discr. RMMI
López et al. (2020)	Material preselection	yes	Ordered SIMP
López et al. (2020)	Material islands limitation	yes	Ordered SIMP
López et al. (2020)	Maximum interface size	yes	Ordered SIMP
Guest et al. (2004)	Dilated projection minimum size	no	SIMP
Sigmund (2007)	Morphology-based minimum size	no	SIMP
Sigmund (2009)	Robust minimum size	no	SIMP

Table 2.3: Summary of existing minimum size methods or similar.

Author	Type	usable in MMTO?	TO Method
Florea et al. (2019)	Joint accessibility	yes	RMMI
Iyer et al. (2024)	Crack detection	no	SIMP
Kopatsch et al. (2024)	Tooling accessibility	yes	Ordered SIMP
Langelaar (2019)	Tooling accessibility	no	SIMP
Liu et al. (2024)	Minimum feature size	yes	DMO
Liu et al. (2024)	Material preselection	yes	DMO
López et al. (2020)	Material preselection	yes	Ordered SIMP
López et al. (2020)	Material islands limitation	yes	Ordered SIMP
López et al. (2020)	Maximum interface size	yes	Ordered SIMP

Table 2.4: Summary of existing assembly constraints and methods.

Author	Type	usable in MMTO?	TO Method
Cheng et al. (2024)	Maximum stress (p-norm)	yes	Ordered SIMP
Hu et al. (2022)	Maximum interface stress (p-norm)	yes	RMMI
Pereira et al. (2004)	Maximum stress (p-norm)	no	SIMP
Ramani (2010)	Stress (cost function)	yes	BESO

Table 2.5: Summary of existing stress constraints.

Author	Type	usable in MMTO?	TO Method
Iyer et al. (2024)	Crack detection	no	SIMP
Li et al. (2023)	Temperature upper limit	yes	RMMI
Liu et al. (2024)	Material preselection	yes	DMO
López et al. (2020)	Material preselection	yes	Ordered SIMP
López et al. (2020)	Total mass upper limit	yes	Ordered SIMP
Shah et al. (2022)	Natural frequency	yes	SIMP
Zhou and Saitou (2018)	Interface stiffness penalty	yes	Modified SIMP

Table 2.6: Summary of other existing methods.

3

Multi-material hotspot detection

3.1. Introduction

The first overhang control method for MMTO is an extension to the geometric hotspot detection method by Das et al. (2025). The working principle of the single-material version of this constraint is as follows: An estimate of the local conductivity of an element is made based on the surrounding geometry by overlaying a hemispheric conductivity domain on the mesh. The densities of the elements within this domain are added via a weighted sum to determine the conductivity. The weights are set to apply both radial and angular penalization. The conductivity field is reversed to create a pseudo-temperature field. This field is then subjected to a density correction in order to relax the constraint for low-density elements. The maximum temperature is determined via the p-norm equation, and forms the input to an inequality constraint.

3.2. Derivation of the multi-material extension

In order to extend this to MMTO, the critical temperature should be independently controllable for each material phase. This is achieved by shifting the temperature field based on the material mixture field. First, the temperature field is generated for the general design with the equation given by Das et al. (2025) as

$$T_e = \left(1 - \frac{\sum_{k \in N_e} x_e^{\bar{q}} w_k}{\sum_{k \in N_e} w_k}\right) x_e^{\bar{r}}, \tag{3.1}$$

where x_e denotes the density of an element. Parameter $\bar{q}$ is used to penalize the conductivity contribution of low density elements, and $\bar{r}$ is used to relax the temperature constraint for low densities. The weights $w_{(p,q)}$ are selected based on the surrounding geometry that falls within a specified radius, with a strong bias towards the build plate direction:

$$w_{(p,q)} = w_{(p,q)}^{(\mathrm{rad})} w_{(p,q)}^{(\mathrm{ang})}, \tag{3.2}$$

where $w_{(p,q)}^{(\mathrm{rad})}$ is the radial penalization, and $w_{(p,q)}^{(\mathrm{ang})}$ the angular penalization:

$$w_{(p,q)}^{(\mathrm{rad})} = \frac{1}{\kappa}\left(\kappa - \sqrt{(a_i - a_j)^2 + (b_i - b_j)^2}\right) \tag{3.3a}$$

$$w_{(p,q)}^{(\mathrm{ang})} = \frac{1}{\pi/2} \arctan\left(\frac{b_i - b_j}{a_i - a_j}\right), \tag{3.3b}$$

with κ as the thermal interaction length, and a and b as respectively the horizontal and vertical distance between the current element p and the surrounding element q. We then introduce a critical temperature $T_{\mathrm{cr}}^{(m)}$ for each material m. The temperature field obtained in Eq. (3.1) can be shifted based on the mixture field $\mathbf{x}_2$:

$$x_{2e} = \frac{\mu_e^{(1)}}{\mu_e^{(1)} + \mu_e^{(2)} + \epsilon}, \tag{3.4}$$

where $\mu_e^{(1)}$ and $\mu_e^{(2)}$ are the density fields of material 1 and 2, respectively. ϵ is a small shift to avoid singularities. For a three-phase optimization problem (two materials plus void), the equation for the corrected temperature field reads:

$$\mathbf{T}_{\text{mmto}} = \mathbf{T} + \left(T_{\text{cr}}^{(1)} - T_{\text{cr}}^{(2)}\right)(1 - \mathbf{x}_2), \tag{3.5}$$

The corrected field is then constrained to the critical temperature of the first material. The p-norm maximum of the temperature field is then taken, and implemented in a constraint:

$$T_{\text{max}} = \left(\frac{1}{n_{\text{el}}} \sum_{i=1}^{n_{\text{el}}} T_e^P\right)^{1/P} \tag{3.6}$$

$$g = c_0 \left(\frac{\bar{\alpha} T_{\text{max}}}{T_{\text{cr}}} - 1\right). \tag{3.7}$$

To counter the overestimation of the p-norm operation, Das et al. (2025) introduced a scaling factor $\bar{\alpha}$. This factor is equal to the ratio of the p-norm output and the true maximum, and is updated every $n_{\bar{\alpha}}$ iterations. The steepness of the allowable overhang can be controlled with the critical temperature T_{cr}, and the strength of the constraint itself is scaled with c_0.

3.3. Sensitivity analysis

In order for the multi-material hotspot constraint to be implemented in gradient-based TO, the analytical sensitivities are required. The chain rule is applied to split the total sensitivity of the constraint g to the density and mixture fields $\mathbf{x}_1$ and $\mathbf{x}_2$ into the individual contributions. Thus only the partial derivatives of the new operations need to be found, which can then be inserted into the existing chain.

$$\frac{\partial g}{\partial \mathbf{x}_2} = \frac{\partial g}{\partial T_{\text{max}}} \frac{\partial T_{\text{max}}}{\partial \mathbf{T}_{\text{mmto}}} \left(\frac{\partial \mathbf{T}_{\text{mmto}}}{\partial \mathbf{T}} \frac{\partial \mathbf{T}}{\partial \mathbf{x}_k} + \frac{\partial \mathbf{T}_{\text{mmto}}}{\partial \mathbf{x}_k}\right) \tag{3.8}$$

for $k = 1, 2$. The final term for $k = 1$ equals zero:

$$\frac{\partial T_{\text{mmto},e}}{\partial x_{1e}} = 0 \tag{3.9}$$

and for $k = 2$, it equals:

$$\frac{\partial T_{\text{mmto},i}}{\partial x_{2j}} = \begin{cases} T_{\text{cr}}^{(2)} - T_{\text{cr}}^{(1)}, & i = j \\ 0, & \text{else.} \end{cases} \tag{3.10}$$

The sensitivity of the MMTO temperature field to the original temperature field equals one:

$$\frac{\partial T_{\text{mmto},e}}{\partial T_e} = 1 \tag{3.11}$$

The sensitivity from the constraint value g to the maximum temperature T_{max} is given by the sensitivity of the standard inequality constraint. All other sensitivities match the single-material hotspot detection constraint.

3.4. Numerical tests

3.4.1. Problem definition

The multi-material hotspot filter is implemented in TO, after which some numerical tests are performed. The study is performed on the 2D MBB beam problem (Olhoff et al., 1991), with a domain of 180×60 bilinear quadrilateral elements. The material is interpolated using the RMMI model (Sigmund and Torquato, 1997), with $\mathbf{x}_1$ as the density field, and $\mathbf{x}_2$ as the mixture field. A density filter (Bruns and Tortorelli, 2001) is first applied, followed by Heaviside projection (Wang et al., 2011) with three thresholds: eroded, nominal, and dilated. The filtered and projected fields are then used to find both the temperature and Young's modulus fields. The stiffness of the blue material is twice that of the red material, however the blue material also has a stricter critical temperature $T_{\mathrm{cr}}^{(2)}$, which is kept at 0.75 for the sweep over $T_{\mathrm{cr}}^{(1)}$, and at 0.80 for the sweep over $n_{\bar{\alpha}}$. The red material has a lenient critical temperature of $T_{\mathrm{cr}}^{(1)} = 0.95$, unless otherwise specified.

A simplified version of the Robust formulation is used, where the stiffness matrix is determined by the eroded design, and the volume fraction is based on the dilated design. To ensure that the nominal design adheres to the correct volume fraction, the volume fraction of the dilated design is scaled down to that of the nominal design every 25 iterations before it is implemented in the constraint. The temperature field for the hotspot constraint is based on the eroded design. An overview of the parameters used is given in Table 3.1.

Symbol	Value	Description
c_0	2500	Scaling factor for objective and constraints
τ_f	10^{-5}	Relative objective change stopping criterion
τ_x	10^{-5}	Relative design change stopping criterion
$v_{\max}$	0.5	Maximum volume fraction
n_{iter}	600	Maximum number of iterations
$E^{(1)}$	1 GPa	Young's modulus of the red material
$E^{(2)}$	2 GPa	Young's modulus of the blue material
ν	0.33	Poisson ratio
F	1 N	Applied force
p	3	Intermediate density penalization
r	4 elements	Filter radius
κ	12 elements	Thermal interaction length
$\bar{r}$	0.1	Constraint relaxation for low densities
$\bar{q}$	5	Conductivity penalization for low densities
$n_{\bar{\alpha}}$	25 iterations	P-norm correction update rate
$\beta_{\max}$	20	Maximum projection sharpness
ξ_0	0.5	Overestimation suppression threshold
η_{ero}	0.6	Eroded threshold
η_{nom}	0.5	Nominal threshold
η_{dil}	0.4	Dilated threshold

Table 3.1: Parameter values used for the multi-material hotspot example.

3.4.2. Results

Figures 3.3 and 3.4 show two parameter sweeps of the hotspot constraint; the former shows the designs for a range of values of the critical temperature of the red material $T_{\mathrm{cr}}^{(1)}$, and the latter shows the designs for a range of the scaling factor update rate $n_{\bar{\alpha}}$. Relative compliance is displayed under each design, and has been normalized against the reference design in Figure 3.1. In all designs, the red material, which has a more lenient critical temperature, is used as a coating on the overhanging surfaces. The main structure is made up of the blue material, as this has the highest stiffness. In Figure 3.2 we see the outcome of a design made up of two materials whose critical temperature is lenient for both.

Figure 3.1: Reference multi-material design without a hotspot constraint. Note: Due to the nature of the compliance objective, the optimizer will always converge to a design made up of exclusively the material with the highest stiffness. $c/c_{\mathrm{ref}} = 254.9\%$

Figure 3.2: Example of a design with lenient critical temperatures for both materials. $T_{\mathrm{cr}}^{(1)} = 0.95$, and $T_{\mathrm{cr}}^{(2)} = 0.90$.

The design in Figure 3.3d is not printable. Much like the design in Figure 3.2, it contains stalactite-like structures, that are the result of the optimizer converging to a local minimum too fast. In the first iteration of the optimization, the overestimation of the p-norm is high, as the entire design domain has the same density. Thus, $\bar{\alpha}$ is low, and the hotspot constraint is relaxed. This allows the optimizer to converge towards the unconstrained design, until iteration $n_{\bar{\alpha}}$, at which point the scaling factor $\bar{\alpha}$ is updated, and the constraint becomes more heavily enforced. If there is too much unsupported material in the design before the constraint is tightened, stalactites form in an attempt to provide support. Designs with these structures then form a local plateau in function evaluation value, as moving the tip of this stalactite in any direction does not change the constraint value. The sensitivities are zero, and the structures remain in the final design.

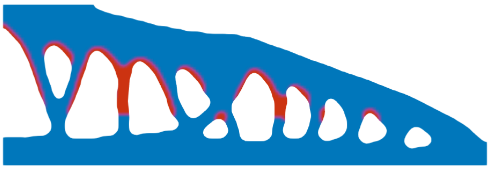
In the design in Figure 3.2, both critical temperatures are high, allowing the optimizer to converge to a design that is very close to the reference design in Figure 3.1. As there is a lot of unsupported material in the design when $n_{\bar{\alpha}}$ is updated, it becomes unstable, and eventually converges to a poor local minimum. This is a major limitation of the multi-material hotspot constraint, and so far no parameter changes have been able to alleviate this.

From the designs in Figure 3.4, we see that the slower the update rate becomes, the more the final design starts to resemble the reference design in Figure 3.1. If the design is more heavily converged before the hotspot constraint becomes strictly enforced, it tends to end up closer to the reference design. The relative compliance is the lowest in Figure 3.4c, though its design closely matches the result in Figure 3.4d.

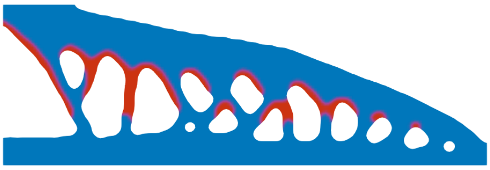
3.5. Discussion

The hotspot constraint in its current form is functional, and capable of restricting the overhang angle for each material separately, but it has poor convergence behaviour. The hotspot constraint restricts changes in the design too much, preventing the optimizer from converging to a well-performing local minimum. Relaxing the constraint more for intermediate densities alleviates this to an extent, but this promotes intermediate densities in the final design too much.

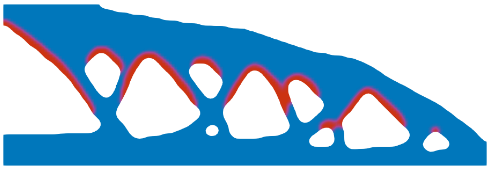
Furthermore, there is only a small window of constraint settings that allow the optimizer to converge to a printable design, and this window shifts each time a change is made to the optimization problem. In particular, the constraint relies heavily on the scaling factor $\bar{\alpha}$ that corrects the p-norm to the true maximum. In the first iteration, the overestimation of the p-norm will be great, as the entire design domain has the same temperature. This creates a low value for the correction $\bar{\alpha}$, which corresponds to lenient constraint enforcement. On the next update at iteration $n_{\bar{\alpha}}$, the correction factor $\bar{\alpha}$ is increased, and the design is modified to adhere to the stricter constraint. If $\bar{\alpha}$ is updated too soon, the solution is significantly restricted from moving through the design space, and a poor local minimum is obtained. If $\bar{\alpha}$ is updated too infrequently, the design has already sufficiently moved into the direction of a local minimum, and the hotspot constraint cannot be effectively enforced any more. These problems are not unique to the multi-material variant, as the regular version suffers from the same issue.



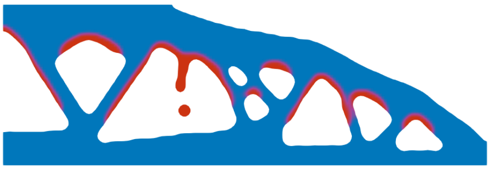
(a) $T_{\mathrm{cr}}^{(1)} = 0.80$, $c/c_{\mathrm{sm}} = 129.6\%$



(b) $T_{\mathrm{cr}}^{(1)} = 0.85$, $c/c_{\mathrm{sm}} = 130.8\%$

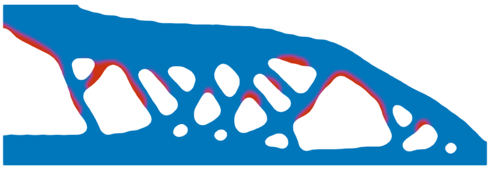


(c) $T_{\mathrm{cr}}^{(1)} = 0.90$, $c/c_{\mathrm{sm}} = 119.9\%$

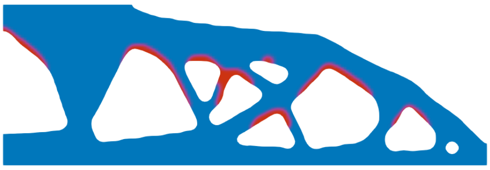


(d) $T_{\mathrm{cr}}^{(1)} = 0.95$, $c/c_{\mathrm{sm}} = 115.3\%$

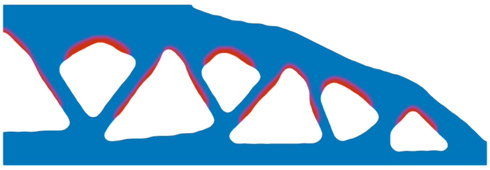
Figure 3.3: MBB beam with hotspot constraint, for various critical temperatures of the red material. The critical temperature of the blue material is kept constant at 0.75



(a) $n_\alpha = 5$, $c/c_{\mathrm{sm}} = 116.9\%$



(b) $n_\alpha = 10$, $c/c_{\mathrm{sm}} = 116.0\%$



(c) $n_\alpha = 25$, $c/c_{\mathrm{sm}} = 110.8\%$

(d) $n_\alpha = 50$, $c/c_{\mathrm{sm}} = 115.1\%$

Figure 3.4: MBB beam with hotspot constraint, for various update rates n_α. This sweep is performed with $T_{\mathrm{cr}}^{(1)} = 0.95$ and $T_{\mathrm{cr}}^{(2)} = 0.80$.

A common observation during the optimization process is that the stalactite-like structures prevent the constraint from being satisfied, and that this causes a continuous increase of the scaling factor $\bar{\alpha}$. The constraint kept getting stricter, which negatively affected the entire design, but did not remove the stalactite. This can be seen in Figure 3.2: even though it has the most lenient critical temperature for both materials, the overhanging surfaces are also by far the most inclined.

The derivation and some numerical tests of this constraint have been included in this thesis as a proof-of-concept. Due to the behaviour described above, a further parameter study for this constraint has been omitted, and focus has been redirected to the overhang filter described in the next chapter.

4

Multi-material overhang filter

This chapter is intended as a manuscript for journal submission. Therefore the formatting will differ slightly from the rest of the document.

4.1. Abstract

Multi-material additive manufacturing (AM) is a rapidly developing field, that enables the production of parts designed with increased design freedom, to achieve superior performance compared to regular AM. Topology optimization (TO) is ideally suited to leverage the increased design freedom AM provides over conventional manufacturing methods, but resulting TO designs do not automatically conform to manufacturability limitations of AM, such as minimum feature size or maximum overhang angle. Existing single-material overhang filters and constraints cannot account for differences in critical overhang angle when applied to multi-material TO. This paper proposes a novel multi-material overhang filter, that allows for independent control of the critical overhang angle of each material, and permits inter-material support. We demonstrate the filter's functionality for a compliance case, as well as for eigenfrequency optimization problems in both 2D and 3D. The filter accurately enforces the overhang angle for a wide range of critical angles, and can create printable designs without major performance loss.

Keywords

Topology optimization · Overhang filter · Multi-material · Additive manufacturing · Manufacturability

4.2 Introduction

Topology optimization (TO) is a design methodology that aims to find the optimum geometric layout of a structure based on a specified objective, such as minimum compliance (Bendsøe, 1989), maximum eigenfrequency (Ma et al., 1995; Pedersen, 2000), or minimum stress (Allaire et al., 2004). Due to the geometrical complexity of topology optimized designs, conventional manufacturing techniques are often inefficient or even infeasible. A set of production processes that can accommodate this degree of geometrical complexity is additive manufacturing (AM). AM is unique among manufacturing methods because the cost and production duration is largely independent of the geometrical complexity. However, AM is still subject to manufacturing limitations, and to ensure the manufacturability of the topology optimized designs, these limitations should be taken into account during the design process.

Recent advances in metal AM made it possible to print parts made up of multiple materials (Schneck et al., 2021; Zainelabdeen et al., 2024; Clare et al., 2025). This allows engineers to tailor the selection of materials to the design requirements, increasing the design freedom. For density-based TO, various multi-material interpolation schemes have been proposed, such as Recursive Multiphase Material Interpolation (RMMI), proposed by Sigmund and Torquato (1997), Discrete Material Optimization (Stegmann and Lund, 2005), and Ordered Simplified Isotropic Material Interpolation (Zuo and Saitou, 2016). Some manufacturability constraints and filters, such as those for removing enclosed voids (Liu et al., 2015; Sabiston and Kim, 2020; Yaghoobi et al., 2024; van der Zwet et al., 2023; Xiong et al., 2020) can be directly implemented in these multi-material schemes, as they are not material-specific. Other authors have proposed manufacturability constraints that are specific to multi-material topology optimization (MMTO), such as restrictions on the material interface size and shape (López et al., 2020; Liu et al., 2024).

A limitation that has not been addressed in MMTO is control of the critical overhang angle: a major limitations of AM is that it cannot print parts with steep overhangs without support structures. These supports increase production time, and will have to be removed after production. Restricting the design space to only include designs that are printable without supports can significantly reduce post-processing time and material waste. Initial attempts at creating an overhang filter were proposed by Brackett et al. (2011), Leary et al. (2014), and Gaynor et al. (2014). The latter proposed a method in which a cone is projected downwards from each element. The densities within this cone are then summed, and a correction factor is determined between 0 and 1 with a Heaviside function, once a threshold is exceeded. The densities of the solid elements are multiplied with their correction factor, and thus is only allowed to exist if it is sufficiently supported.

In the algorithm proposed by Langelaar (2016a,b), a support region in the previous layer is defined for each element, made up of 3 elements in 2D, and either 5 or 9 elements in 3D. The maximum filtered density in this support region then provides an upper limit for the density in the current element. This filter requires a structured mesh, and the critical overhang angle can only be changed by modifying the element aspect ratio. Subsequent studies by Langelaar (2018) removed the mesh dependency, and made the filter suitable for an unstructured meshes by means of projecting the design onto a regular background mesh. This also allows for simultaneous optimization on the build orientation. Hoffarth et al. (2017) and Delissen et al. (2022) added support for custom critical overhang angle through element interpolation via cone projection, similar to the method employed by Gaynor et al. (2014).

While Langelaar's overhang filter remains a widely used method, some authors suggested alternative approaches for density-based TO. Qian (2017) proposed a method where the unsupported area is projected onto the build plate. A constraint is then applied on this area, which restricts the total size of the overhanging area. Finally, a wave front propagation strategy was proposed by van de Ven et al. (2018, 2020). In this approach, two wave fronts propagate through the design. The first one travelling at a constant speed, and a second whose speed is reduced if its velocity vector exceeds the critical overhang angle. An arrival time difference between the two fields is present exclusively in the unsupported material, which can then be effectively removed. This method can be applied to unstructured meshes and supports custom critical overhang angles by default, and is effectively as a continuous version of Langelaar's filter (van de Ven et al., 2021).

Besides insufficient support, steep overhang angles also leads to overheating, since metal powder has a much lower conductivity than solid metal (King et al., 2015), and the heat generated by the laser must thus primarily be transferred to the build plate via the already printed solid. An overhang filter with a fixed angle might sometimes be too strict, and at other times not strict enough. Ranjan et al. (2017) and Das et al. (2025) proposed physics-based methods to restrict overheating, which can account for this by making an estimate of the local conductivity of each element. The critical overhang angle can therefore vary depending on the thermal and mechanical properties of the material. Consequently, while considering overhang limitations for a multi material design, varying critical overhang angles for each material phase is of paramount importance. For this reason existing overhang control methods cannot directly be implemented in MMTO.

For single-material topology optimization (SMTO) ample amount of studies addressed the incorporation of manufacturability limitations to a topology optimization framework (Bayat et al., 2023), but for MMTO this is still an active area of research. Most notably, there is currently no published research proposing an overhang filter for MMTO. Since different materials generally have different critical overhang angles due to their different mechanical and thermal properties, the existing overhang constraints and filters cannot directly be used in MMTO.

This paper proposes a novel multi-material overhang filter, which is capable of accounting for the different critical overhang angles of each material phase in both 2D and 3D problems. Following, Delissen et al. (2022), the cone projection concept is extended to MMTO in order to add custom angle support for each material individually. The filter passes through the design domain sequentially layer by layer; In each layer, the following steps are taken. First a support density field is determined as the sum of the filtered densities of all material phases in the previous layer. This support field is then scaled down for each material to create an accurate upper limit. Finally, the unfiltered densities in the current layer are restricted to the maximum support value.

The remainder of the paper is organized as follows: Section 4.3 describes the MMTO overhang filter, including interpolation for custom angles and sensitivity analysis. Section 4.4 contains various numerical tests to validate the functionality of the filter. Section 4.5 describes the problem definition for the compliance and eigenfrequency example cases, the results of which are presented in Section 4.6. Finally, Section 4.7 provides a discussion on the results, and conclusions are provided in Section 4.8.

4.3 Multi-material overhang filter formulation

The goal of the proposed filter is to restrict the design space of an MMTO problem to exclude overhanging design features that hinder manufacturability in multi-material AM. The filter removes unsupported material at every optimization iteration, ensuring that the final design is printable.

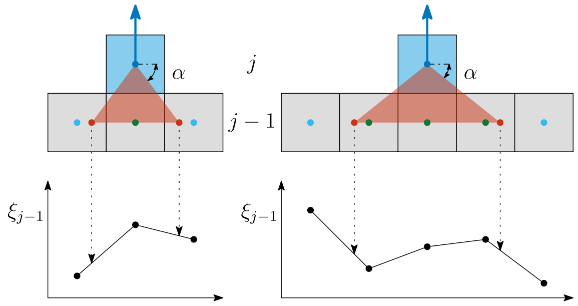
4.3.1 *Notation*

The filter derivation requires frequent use of equations in layer-dependent notation. To differentiate between general and layer-wise quantities, a tilde symbol is used to indicate fields referring to a single layer, as well as a subscript that denotes the layer number of interest. The layers numbering follows the order of deposition in the build direction. Thus, while $\boldsymbol{\mu}$ and μ_e denote the general density field and density of element e respectively, $\tilde{\boldsymbol{\mu}}_j$ and $\tilde{\mu}_{j|i}$ denote the density field of layer j and that of element $i = 1, \dots, n_j$ in layer j, where n_j is the number of elements in layer j.

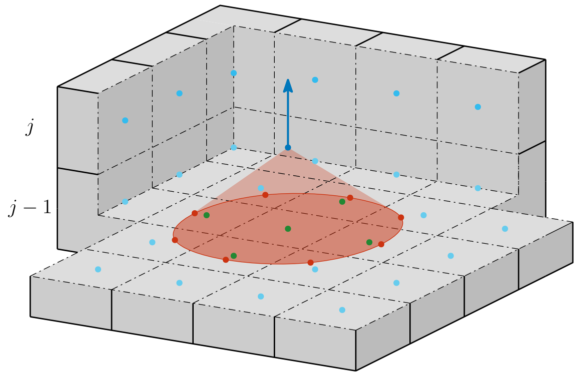
4.3.2 *Custom angle interpolation*

The work by Langelaar (2016a) restricts a single-material design to a fixed critical overhang angle, defined by the element aspect ratio. Furthermore, for the 3D case either a 5 or 9-element support region was used (Langelaar, 2016b), which creates a pyramid-shaped support region. This implies the filtered overhang angle was either underestimated (in case of the 9 element region) or overestimated (in case of the 5 element region). To remove this mesh dependency while retaining the option of filtering to a specific angle, Delissen et al. (2022) modified the implementation for custom critical overhang angles. A mathematical derivation for the custom angle critical overhang filter was not provided, and thus it will be briefly discussed in this section for the sake of completeness.

The filter converts the field of blueprint densities $\tilde{\boldsymbol{\mu}}_j$ to a printable density field $\tilde{\boldsymbol{\xi}}_j$ in a layer-wise manner. For each element in layer j, a cone with the critical overhang angle is projected onto the layer $j - 1$, as shown in Figure 4.1. For each element a set of support element centroids $\tilde{S}_{j|i}$ is then determined, which contains n_s element centroids



(a) 2D element interpolation for a critical overhang angle α greater than 45°, $n_\mathrm{s} = 1$ (left) and a critical overhang angle less than 45°, $n_\mathrm{s} = 3$ (right). For both cases $n_\mathrm{int} = 2$.



(b) 3D element interpolation with $n_\mathrm{s} = 5$ and $n_\mathrm{int} = 8$. Elements have been cut on the dashed lines to make the interpolation visible.

Figure 4.1: Schematic of element interpolation in 2D (a) and 3D (b). Element centroids that make up the support set $\tilde{S}_{j|i}$ shown in green. Interpolation points that make up $\tilde{T}_{j|i}$ shown in red. All other centroids are shown in blue.

in layer $j-1$ that fall within the cone projection, shown in green in Figure 4.1. Additionally, n_int interpolation points are defined on the edge of the cone projection on layer $j-1$, as depicted in Figure 4.1 by the red dots. These points make up the set $\tilde{T}_{j|i}$. The printable densities $\tilde{\boldsymbol{\xi}}_{j-1}$ of the surrounding elements is linearly interpolated at these points, as shown in Figure 4.1a, and their inclusion ensures accurate prescription of the critical overhang angle. The maximum of the printable densities $\tilde{\boldsymbol{\xi}}_j$ in $\tilde{S}_{j|i}$ and $\tilde{T}_{j|i}$ becomes the support density underneath element i in layer j as

$$\tilde{\Xi}_{j|i} = \max\left(\xi_k\right), \quad k \in \tilde{S}_{j|i} \cup \tilde{T}_{j|i}, \qquad (4.1)$$

where $\tilde{\Xi}_{j|i}$ is the maximum support density for element i in layer j. To ensure the filter is differentiable and can be implemented in gradient-based TO, the maximum in Eq. (4.1) is calculated with a modified version of the p-norm equation. The maximum support density for all elements in layer j is:

$$\tilde{\boldsymbol{\Xi}}_j = \left(\mathbf{D}\tilde{\boldsymbol{\xi}}^P_{j-1} + \sum_{l=1}^{n_\mathrm{int}} \left(\mathbf{H}^{(l)}\tilde{\boldsymbol{\xi}}_{j-1}\right)^P\right)^{1/Q}, \qquad (4.2)$$

where P controls the smoothness of the operation. Matrix $\mathbf{D}$ is used to select the printable densities of the elements in $\tilde{S}_{j|i}$ from the array $\tilde{\boldsymbol{\xi}}_{j-1}$ containing the printable densities of the entire layer $j-1$. Each entry $D_{(s,t)}$ is set to one if the centroid of element t in layer $j-1$ exists in the set of support centroids $\tilde{S}_{j|i}$ of element s in layer j, and set to zero otherwise.

The interpolated densities $\mathbf{H}^{(l)}\tilde{\boldsymbol{\xi}}_{j-1}$ on the edge of the cone are found by linear interpolation of the printable densities at the surrounding element centroids. The interpolation weights are stored in matrix $\mathbf{H}^{(l)}$. Each matrix stores the weights of a single interpolation point $l = 1, \ldots, n_\mathrm{int}$, for all elements in a layer.

Since the p-norm operation will overestimate the true maximum for intermediate densities of equal value, the printable densities $\tilde{\boldsymbol{\xi}}_j$ in layer j are slightly higher than the true support maximum. This increase in printable densities leads to a slow build-up of unsupported material across the layers. In order to minimize this effect Langelaar (2016a) penalized the output of the p-norm by setting Q:

$$Q = P + \frac{\log n_\mathrm{s}}{\log \xi_0}. \qquad (4.3)$$

The parameter ξ_0 is the intermediate density for which no overestimation of the smooth maximum is allowed, and can be tuned to make the penalization stricter. Densities below ξ_0 will be underestimated, while densities above ξ_0 will be overestimated.

The maximum support density $\tilde{\Xi}_{j|i}$ is then used in the smooth minimum together with the blueprint density field $\tilde{\mu}_{j|i}$ to find the printable densities $\tilde{\xi}_{j|i}$ via a smooth minimum:

$$\tilde{\xi}_{j|i} = \frac{1}{2}\left(\tilde{\mu}_{j|i} + \tilde{\Xi}_{j|i} - \sqrt{\varepsilon + \left(\tilde{\mu}_{j|i} - \tilde{\Xi}_{j|i}\right)^2}\right) + \frac{1}{2}\sqrt{\varepsilon}, \qquad (4.4)$$

for layer j, where ε controls the smoothness of the function. The above process is repeated for all layers $j = 1, \ldots, n_h$ in the build direction.

4.3.3 *Multi-material overhang filter*

Figure 4.2 provides an overview of the multi-material filter algorithm. In the following section, $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(2)}$ denote the blueprint densities of material 1 and 2 respectively, and $\boldsymbol{\xi}^{(1)}$ and $\boldsymbol{\xi}^{(2)}$ denote the printable densities of material 1 and 2. The multi-material filter works with a layer-wise approach, where $\tilde{\boldsymbol{\xi}}^{(1)}_j$ and $\tilde{\boldsymbol{\xi}}^{(2)}_j$ determine the support for $\tilde{\boldsymbol{\mu}}^{(1)}_{j+1}$ and $\tilde{\boldsymbol{\mu}}^{(2)}_{j+1}$.

First, a support field $\tilde{\zeta}_j$ is calculated, which will take the place of the printable densities $\tilde{\boldsymbol{\xi}}_j$ in the smooth maximum. The field $\tilde{\zeta}_j$ is equal to the sum of the printable densities $\tilde{\boldsymbol{\xi}}_j$ of both materials in layer j. This allows each phase to be built on top of the other,

$$\tilde{\zeta}_{j|i} = \tilde{\xi}^{(1)}_{j|i} + \tilde{\xi}^{(2)}_{j|i}. \tag{4.5}$$

Next, the support field $\tilde{\zeta}_j$ of layer j is used to find the maximum support density $\tilde{\Xi}_{j+1}$ for all materials in layer $j+1$. The approach here is similar to Eq. (4.2). The interpolated densities are defined separately for each material, taking into account the material-specific critical overhang angles $\alpha^{(m)}$ and subsequent radius of the cone projection. Exponent $Q^{(m)}$ and matrix $\mathbf{H}^{(l|m)}$ are material-dependent as well. The former is a function of the number of centroids $n^{(m)}_{\mathrm{s}}$ in $\tilde{S}^{(m)}_{j|i}$, and the latter is a function of the critical overhang angle $\alpha^{(m)}$,

$$\tilde{\Xi}^{(m)}_j = \left(\mathbf{D}^{(m)}\tilde{\zeta}^P_{j-1} + \sum_{l=1}^{n_{\mathrm{int}}} \left(\mathbf{H}^{(l|m)}\tilde{\zeta}_{j-1}\right)^P\right)^{1/Q^{(m)}}. \tag{4.6}$$

The filter should ensure that the sum of the printable densities $\tilde{\boldsymbol{\xi}}^{(m)}_j$ of all materials does not exceed the maximum support density. As $\tilde{\Xi}^{(m)}_j$ in Eq. (4.6) for each material m is determined as the maximum of $\tilde{\zeta}_{j-1}$, which is not material specific, using $\tilde{\Xi}^{(m)}_j$ in the smooth minimum directly would separately filter each material according to the support density. Thus, if for example the maximum support density of an element is 0.4, each material phase would be separately restricted to this density, leading to a total element density of 0.8. This would allow the printable density $\tilde{\boldsymbol{\xi}}^{(1)}_j$ to be double the support density $\tilde{\zeta}_j$ in each layer, leading to a rapid build-up of unsupported material. To prevent this, a scaling factor is determined as the material mixture of the blueprint densities of layer j:

$$\tilde{x}^{(\mathrm{m})}_{j|i} = \frac{\tilde{\mu}^{(1)}_{j|i}}{\tilde{\mu}^{(1)}_{j|i} + \tilde{\mu}^{(2)}_{j|i} + \epsilon}, \tag{4.7}$$

where ϵ is a small correction factor to prevent division by zero for elements in which the density of both materials equals zero. $\tilde{\Xi}^{(m)}_{j|i}$ is then corrected as follows:

$$\hat{\tilde{\Xi}}^{(1)}_{j|i} = \tilde{\mu}^{(\mathrm{m})}_{j|i}\tilde{\Xi}^{(1)}_{j|i} \tag{4.8a}$$

$$\hat{\tilde{\Xi}}^{(2)}_{j|i} = \left(1 - \tilde{\mu}^{(\mathrm{m})}_{j|i}\right)\tilde{\Xi}^{(2)}_{j|i}. \tag{4.8b}$$

This provides a suitable upper limit for each material phase. It retains the ratio of materials present in the blueprint field, while ensuring the total density not to exceed the support density $\tilde{\zeta}_j$. The smooth minimum operation finds the printable densities for both materials in the current layer,

$$\tilde{\xi}^{(m)}_{j|i} = \frac{1}{2}\left(\tilde{\mu}^{(m)}_{j|i} + \hat{\tilde{\Xi}}^{(m)}_{j|i} - \sqrt{\varepsilon + \left(\tilde{\mu}^{(m)}_{j|i} - \hat{\tilde{\Xi}}^{(m)}_{j|i}\right)^2}\right) + \frac{1}{2}\sqrt{\varepsilon}. \tag{4.9}$$

4.3.4 *Sensitivity analysis*

In order for the filter to be implemented in gradient-based topology optimization, the sensitivity of the objective f to the design variables of both materials, $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(2)}$, must be found. The total sensitivity can be expanded by making use of the chain rule, which simplifies the problem into a set of smaller sub-problems:

$$\frac{\partial f}{\partial \boldsymbol{\mu}^{(1)}} = \frac{\partial f}{\partial \boldsymbol{\xi}^{(1)}}\frac{\partial \boldsymbol{\xi}^{(1)}}{\partial \boldsymbol{\mu}^{(1)}} + \frac{\partial f}{\partial \boldsymbol{\xi}^{(1)}}\frac{\partial \boldsymbol{\xi}^{(1)}}{\partial \boldsymbol{\mu}^{(2)}} \tag{4.10a}$$

$$\frac{\partial f}{\partial \boldsymbol{\mu}^{(2)}} = \frac{\partial f}{\partial \boldsymbol{\xi}^{(1)}}\frac{\partial \boldsymbol{\xi}^{(1)}}{\partial \boldsymbol{\mu}^{(2)}} + \frac{\partial f}{\partial \boldsymbol{\xi}^{(2)}}\frac{\partial \boldsymbol{\xi}^{(2)}}{\partial \boldsymbol{\mu}^{(2)}}. \tag{4.10b}$$

The four partial derivatives of $\boldsymbol{\xi}^{(m)}$ to $\boldsymbol{\mu}^{(n)}$ need to be found. In the schematic overview depicted in Figure 4.2, the outputs $\boldsymbol{\xi}^{(1)}$ and $\boldsymbol{\xi}^{(2)}$ are shown in orange, while the inputs $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(2)}$ are shown in light blue. We can see that each partial derivative $\partial\boldsymbol{\xi}^{(m)}/\partial\boldsymbol{\mu}^{(n)}$ consists of a direct, intra-layer component, as well as an inter-layer component via the support field $\boldsymbol{\zeta}$. The printable densities $\tilde{\boldsymbol{\xi}}_j$ of layer j depend on the blueprint densities $\tilde{\boldsymbol{\mu}}_j$ of layer j itself, but also on the printable densities $\tilde{\boldsymbol{\xi}}_{j-1}$, which in turn depend on $\tilde{\boldsymbol{\xi}}_{j-2}$, and so forth up until $j = 1$. The blueprint densities of layer j affect the printable densities $\tilde{\boldsymbol{\xi}}^{(m)}_j$ of layer

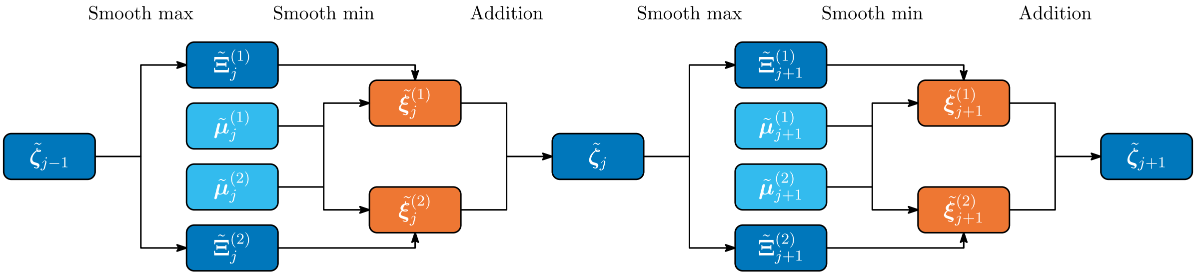


Figure 4.2: Flow chart of the MMTO overhang filter, shown for two layers. The blueprint densities $\tilde{\boldsymbol{\mu}}_j^{(1)}$ and $\tilde{\boldsymbol{\mu}}_j^{(2)}$ are shown in light blue, and the printable densities $\tilde{\xi}_j^{(1)}$ and $\tilde{\xi}_j^{(2)}$ are shown in orange. The support field $\tilde{\zeta}_j$ and the smooth maxima $\tilde{\Xi}_j^{(1)}$ and $\tilde{\Xi}_j^{(2)}$ are shown in dark blue. The text above the figure indicates the operation performed between the columns of blocks.

j, which affect the printable densities $\tilde{\boldsymbol{\xi}}_k$ of all layers $k = j+1, \ldots, n_h$ above it. The first term in Eq. (4.10a) can be written explicitly in terms of individual layers as follows,

$$\frac{\partial f}{\partial \boldsymbol{\xi}^{(1)}}\frac{\partial \boldsymbol{\xi}^{(1)}}{\partial \boldsymbol{\mu}^{(1)}} = \frac{\partial f}{\partial \boldsymbol{\xi}^{(1)}}\frac{\partial \boldsymbol{\xi}^{(1)}}{\partial \tilde{\boldsymbol{\mu}}_1^{(1)}} + \ldots + \frac{\partial f}{\partial \boldsymbol{\xi}^{(1)}}\frac{\partial \boldsymbol{\xi}^{(1)}}{\partial \tilde{\boldsymbol{\mu}}_{n_h}^{(1)}}. \tag{4.11}$$

Each of the partial derivatives in Eq. (4.11) can then be expanded to capture the interaction between the layers:

$$\begin{aligned}\frac{\partial f}{\partial \boldsymbol{\xi}^{(1)}}\frac{\partial \boldsymbol{\xi}^{(1)}}{\partial \tilde{\boldsymbol{\mu}}_1^{(1)}} = {}& \frac{\partial f}{\partial \tilde{\boldsymbol{\xi}}_1^{(1)}}\frac{\partial \tilde{\boldsymbol{\xi}}_1^{(1)}}{\partial \tilde{\boldsymbol{\mu}}_1^{(1)}} + \frac{\partial f}{\partial \tilde{\boldsymbol{\xi}}_2^{(1)}}\frac{\partial \tilde{\boldsymbol{\xi}}_2^{(1)}}{\partial \tilde{\boldsymbol{\mu}}_1^{(1)}} \\ & + \ldots + \frac{\partial f}{\partial \tilde{\boldsymbol{\xi}}_{n_h}^{(1)}}\frac{\partial \tilde{\boldsymbol{\xi}}_{n_h}^{(1)}}{\partial \tilde{\boldsymbol{\mu}}_1^{(1)}}. \end{aligned} \tag{4.12}$$

And finally, each $\partial\tilde{\boldsymbol{\xi}}_p/\partial\tilde{\boldsymbol{\mu}}_q$ can also be split up into the specific layers; the final term in Eq. (4.12) for example, can be written as

$$\frac{\partial \tilde{\boldsymbol{\xi}}_{n_h}}{\partial \tilde{\boldsymbol{\mu}}_j} = \left(\prod_{k=j+1}^{n_h-1} \frac{\partial \tilde{\boldsymbol{\xi}}_{k+1}}{\partial \tilde{\boldsymbol{\xi}}_k}\right)\frac{\partial \tilde{\boldsymbol{\xi}}_{j+1}}{\partial \tilde{\boldsymbol{\mu}}_j}. \tag{4.13}$$

From the equations above, we see that the total sensitivity would have the form of a very large triangular matrix, which is very computationally demanding to include in TO. To improve the efficiency, we can calculate the sensitivities in a layer-wise fashion using an adjoint formulation, like done by Langelaar (2016a). The majority of the terms in the equation above are reused for every layer, and thus have to be computed only once per iteration.

In the multi-material formulation, the sensitivities of the two materials are coupled together at every layer through the support field $\boldsymbol{\zeta}$, as shown in Figure 4.2. For each layer j, the total derivative from the objective f to $\tilde{\boldsymbol{\zeta}}_j$ can be captured as

$$\begin{aligned}\frac{\partial f}{\partial \tilde{\boldsymbol{\zeta}}_j} = {}& \left(\tilde{\boldsymbol{\lambda}}_{j+1}^{(1)}\right)^T \frac{\partial \tilde{\boldsymbol{\xi}}_{j+1}^{(1)}}{\partial \tilde{\Xi}_{j+1}^{(1)}}\frac{\partial \tilde{\Xi}_{j+1}^{(1)}}{\partial \tilde{\boldsymbol{\zeta}}_j} \\ & + \left(\tilde{\boldsymbol{\lambda}}_{j+1}^{(2)}\right)^T \frac{\partial \tilde{\boldsymbol{\xi}}_{j+1}^{(2)}}{\partial \tilde{\Xi}_{j+1}^{(2)}}\frac{\partial \tilde{\Xi}_{j+1}^{(2)}}{\partial \tilde{\boldsymbol{\zeta}}_j}. \end{aligned} \tag{4.14}$$

The multipliers $\tilde{\boldsymbol{\lambda}}_j^{(m)}$ are chosen to capture the sensitivity of layers $j+1$ to n_h:

$$\left(\tilde{\boldsymbol{\lambda}}_j^{(m)}\right)^T = \frac{\partial f}{\partial \tilde{\boldsymbol{\xi}}_j^{(m)}} + \left(\tilde{\boldsymbol{\lambda}}_{j+1}^{(m)}\right)^T \frac{\partial \tilde{\boldsymbol{\xi}}_{j+1}^{(m)}}{\partial \tilde{\boldsymbol{\xi}}_j^{(m)}} \tag{4.15}$$

Each multiplier contains the multiplier of the previous layer, recursively adding the new inter-layer sensitivity. The derivative between the printable densities $\tilde{\boldsymbol{\xi}}^{(m)}$ of two layers can be written as:

$$\frac{\partial \tilde{\boldsymbol{\xi}}_{j+1}^{(m)}}{\partial \tilde{\boldsymbol{\xi}}_j^{(m)}} = \frac{\partial \tilde{\boldsymbol{\xi}}_{j+1}^{(m)}}{\partial \tilde{\Xi}_{j+1}^{(m)}}\frac{\partial \tilde{\Xi}_{j+1}^{(m)}}{\partial \tilde{\boldsymbol{\zeta}}_j}\frac{\partial \tilde{\boldsymbol{\zeta}}_j}{\partial \tilde{\boldsymbol{\xi}}_j^{(m)}}. \tag{4.16}$$

The total sensitivities from f to $\tilde{\boldsymbol{\mu}}_j^{(1)}$ and $\tilde{\boldsymbol{\mu}}_j^{(2)}$ are found by taking $\partial f/\partial\tilde{\boldsymbol{\zeta}}_j$ obtained in Eq. (4.15), multiplying it with the sensitivity from the support field $\tilde{\boldsymbol{\zeta}}_j$ to the blueprint densities $\tilde{\boldsymbol{\mu}}_j^{(m)}$, and finally adding the intra-layer component:

$$\begin{aligned}\frac{\partial f}{\partial \tilde{\boldsymbol{\mu}}_j^{(m)}} = {}& \frac{\partial f}{\partial \tilde{\boldsymbol{\zeta}}_j}\left(\frac{\partial \tilde{\boldsymbol{\zeta}}_j}{\partial \tilde{\boldsymbol{\xi}}_j^{(1)}}\frac{\partial \tilde{\boldsymbol{\xi}}_j^{(1)}}{\partial \tilde{\boldsymbol{\mu}}_j^{(m)}} + \frac{\partial \tilde{\boldsymbol{\zeta}}_j}{\partial \tilde{\boldsymbol{\xi}}_j^{(2)}}\frac{\partial \tilde{\boldsymbol{\xi}}_j^{(2)}}{\partial \tilde{\boldsymbol{\mu}}_j^{(m)}}\right) \\ & + \frac{\partial f}{\partial \tilde{\boldsymbol{\xi}}_j^{(1)}}\frac{\partial \tilde{\boldsymbol{\xi}}_j^{(1)}}{\partial \tilde{\boldsymbol{\mu}}_j^{(m)}} + \frac{\partial f}{\partial \tilde{\boldsymbol{\xi}}_j^{(2)}}\frac{\partial \tilde{\boldsymbol{\xi}}_j^{(2)}}{\partial \tilde{\boldsymbol{\mu}}_j^{(m)}}. \end{aligned} \tag{4.17}$$

Most of the sensitivity terms in the equations above are simply the derivative of the smooth minimum, sometimes in a slightly modified formulation, given in Appendix A. The sensitivity of the cone interpolation concerns the smooth maximum. The sensitivity $\partial\tilde{\Xi}_j^{(m)}/\partial\tilde{\zeta}_{j-1}$ must first be split up via the chain rule to solve the $1/Q$ exponent:

$$\begin{aligned} \frac{\partial \tilde{\Xi}_j^{(m)}}{\partial \tilde{\zeta}_{j-1}} &= \frac{\partial \tilde{\Xi}_j^{(m)}}{\partial \tilde{\mathbf{a}}_j^{(m)}} \frac{\partial \tilde{\mathbf{a}}_j^{(m)}}{\partial \tilde{\zeta}_{j-1}}, \quad \text{with} \\ \tilde{\mathbf{a}}_j^{(m)} &= \mathbf{D}^{(m)} \tilde{\zeta}_{j-1}^{P} + \sum_{l=1}^{n_{\text{int}}} \left(\mathbf{H}^{(l|m)} \tilde{\zeta}_{j-1} \right)^{P}. \end{aligned} \tag{4.18}$$

The first half is solved as:

$$\frac{\partial \tilde{\Xi}_j^{(m)}}{\partial \mathbf{a}_j^{(m)}} = \frac{P}{Q} \operatorname{diag} \left(\tilde{\mathbf{a}}_j^{(m)} \right)^{1/Q-1}. \tag{4.19}$$

The second term is:

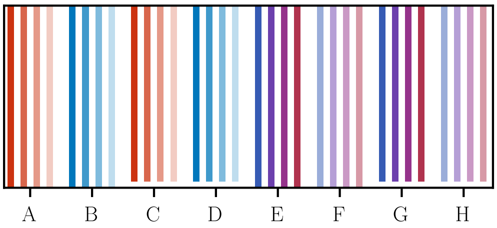
$$\begin{aligned} \frac{\partial \tilde{\mathbf{a}}_j^{(m)}}{\partial \tilde{\zeta}_{j-1}} &= \operatorname{diag} \left(\mathbf{D} \tilde{\zeta}_{j-1}^{P-1} \right) \\ &+ \sum_{l=1}^{n_{\text{int}}} \left(\mathbf{H}^{(l|m)} \right)^{T} \operatorname{diag} \left(\mathbf{H}^{(l|m)} \tilde{\zeta}_{j-1} \right)^{P-1}. \end{aligned} \tag{4.20}$$

4.4 Numerical verification of filter behaviour

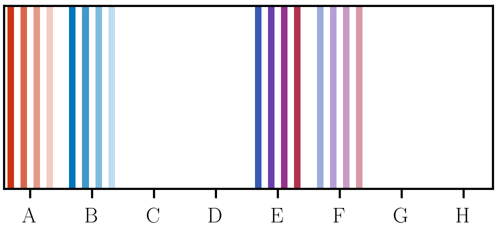
We apply the MMTO overhang filter to several numerical test cases to verify its behaviour. First, the effectiveness of the overestimation suppression parameter ξ_0 is examined. Next, a check is performed to confirm that it filters to the correct critical overhang angle for all materials. For all examples in this section, material 1, displayed in red, will be filtered to a critical overhang angle of $\alpha^{(1)} = 20°$, while material 2, displayed in blue, will be filtered to a critical overhang angle of $\alpha^{(2)} = 60°$.

4.4.1 Effect of overestimation suppression parameter

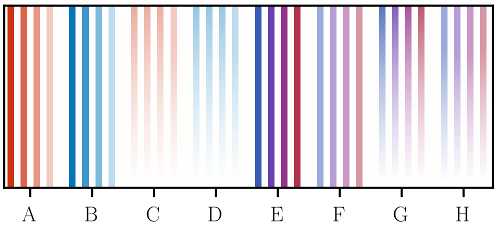
The vertical bars in Figure 4.3 are used to characterize the density change over many layers as a result of the filter. Figure 4.3a shows a set of vertical bars as design features of various densities and material mixtures. Blocks A, B, C, and D are bars of a single material, with a density ranging between 1 an 0.25. For blocks C and D the first layer is removed, making them



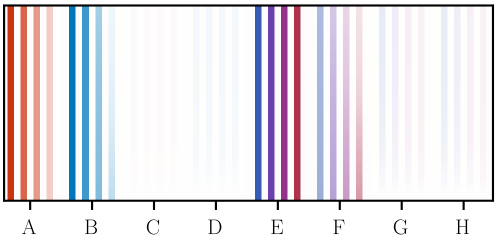
(a) Reference pattern.



(b) Ideal output.



(c) Filter output for $\xi_0 = 0$.



(d) Filter output for $\xi_0 = 0.5$.

Figure 4.3: Vertical bars pattern. Blocks A, B, E, and F contain supported material of various densities and mixtures. Bars C, D, G, and H contain unsupported material. The color bar shows the material mixture for full density material.

unsupported. Blocks E, F, G, and H comprise multi-material bars with a density of 1.00 (E and G), and a density of 0.5 (F and H), with mixtures 0.8, 0.6, 0.4, and 0.2 from left to right. Blocks G and H also have their first layer removed. An ideal filter output, shown in Figure 4.3b, would pass blocks A, B, E, and F intact, while blocks C, D, G, and H are removed.

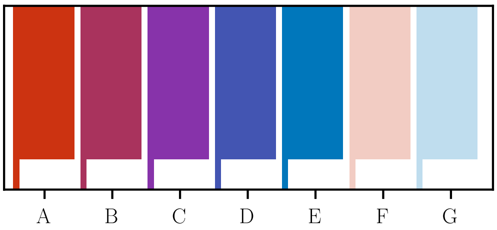
Figs. 4.3c and 4.3d show the output of the MMTO overhang filter for an overestimation threshold of 0 and 0.5. Blocks A and B show that the high density bars are not significantly modified for both thresholds. The two low density bars on the left of block B remain intact for $\xi_0 = 0$, but show a reduction in density for $\xi_0 = 0.5$ in Figure 4.3d. This suggests that materials with a strict critical overhang angle are more sensitive to density suppression for intermediate densities less than ξ_0, as this effect is not present with the red material in block A. The unsupported bars C and D still appear for $\xi_0 = 0$, but almost disappear for $\xi_0 = 0.5$. The value of ξ_0 thus represents a trade-off between suppressing unsupported material and not suppressing low-density supported material.

When $\xi_0 = 0$, the density build-up for mixtures (G and H) is noticeably stronger than for single materials (C and D). When ξ_0 is increased from 0 to 0.5, the reduction for low density material is less noticeable for mixtures than for pure materials, but still present.

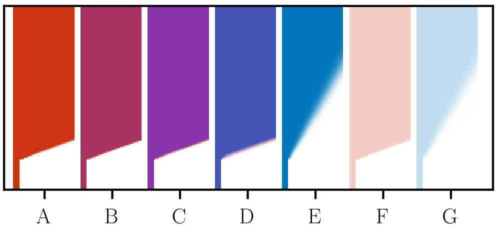
4.4.2 *Effect of overhang angle*

The second pattern test is used to verify that the materials are filtered to the correct angle. Figure 4.4 contains overhanging structures. Layouts A to E contain full density structures, ranging from pure material 1 for block A to pure material 2 for block E. The blocks in-between contain mixtures with a step size of 0.25. Blocks F and G contain structures of pure material with a density of only 0.25.

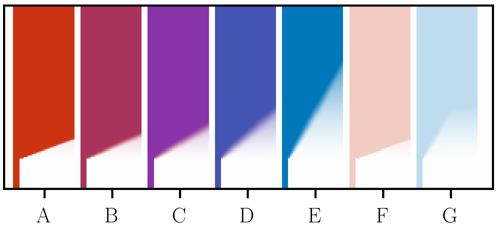
Figure 4.4b shows the ideal output. Notable here is that the overhanging surface contains a density gradient, which is especially noticeable for the stricter overhang angle. This is an artefact of the element interpolation, which outputs intermediate densities for all overhang angles where the interpolation points do not coincide with an element centroid. Projecting the filtered densities to zero or one removes this gradient, but more detailed analysis showed that such projection alters the effective overhang angle. Most projection



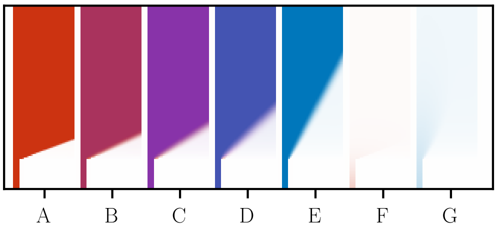
(a) Reference pattern.



(b) Ideal output.



(c) Filter output for $\xi_0 = 0$.



(d) Filter output for $\xi_0 = 0.5$.

Figure 4.4: Overhang pattern. Layouts A and E contain single material structures with a density of 1. Blocks B, C, and D contain multi-material structures of mixture 0.25, 0.5, and 0.75 respectively, and a density of 1. Blocks F and G contain single-material structures with a density of 0.25. Color bar shows the full density mixture.

schemes use a density threshold of 0.5, which does not coincide with the true transition of zero density that defines the critical overhang angle. Setting the density threshold of the projection scheme to zero would create a design with the correct critical overhang angle, but in combination with some form of density filtering this would be an extremely dilated design, which is undesirable.

The material mixtures in layouts B, C, and D of Fig. (4.4b) show that the final overhang angle will follow that of the least strict material. A coating of this material is present on the surface, which provides support. This coating becomes more prominent as the mixture shifts toward the material with the stricter overhang angle, and its thickness depends on the mesh size.

Figures 4.4c and 4.4d again show the true filter output for $\xi_0 = 0$ and $\xi_0 = 0.5$. The pure materials in layouts A and E are filtered to the correct angles, but a density build-up is again visible for the lower value of ξ_0. The gradient on the overhanging surface already becomes noticeable in mixtures, as the coating becomes less clearly defined when the mixture shifts to the material with the stricter critical overhang angle. There is again a build-up of densities for $\xi_0 = 0$, and a removal of supported low-density material for $\xi_0 = 0.5$, as shown by layouts F and G in Figure 4.4d.

From these tests we can conclude that the density build-up can effectively be suppressed in a similar manner as the single material overhang filter. The MMTO overhang filter also enforces the correct critical overhang angle for each material, though some inaccuracy persists for higher values of α.

4.5 Problem formulation

Next, the filter is integrated in three topology optimization case studies, where the results are compared to both a reference single-material design with an overhang filter, and to multi-material designs without the overhang filter. The design domains of the following problems are discretized with a structured finite element mesh consisting of square and cubic elements for 2D and 3D respectively, with 1 mm side length. The Method of Moving Asymptotes (Svanberg, 1987) is used, with a move limit of 0.2. The material properties are interpolated with the RMMI scheme (Sigmund and Torquato, 1997), where $\mathbf{x}^{(s)}$ defines the density field, and $\mathbf{x}^{(m)}$ defines the mixture field. These fields are first filtered through a density filter (Bruns and Tortorelli, 2001) with radius r to create $\bar{\mathbf{x}}^{(s)}$ and $\bar{\mathbf{x}}^{(m)}$. Next, Heaviside projection (Wang et al., 2011)

$$\hat{x}_e = \frac{\tanh\left(\beta\eta\right) + \tanh\left(\beta\left(\bar{x}_e - \eta\right)\right)}{\tanh\left(\beta\eta\right) + \tanh\left(\beta\left(1-\eta\right)\right)} \tag{4.21}$$

is applied to $\bar{\mathbf{x}}^{(s)}$ with three different thresholds, denoted by η, to create eroded, nominal, and dilated fields $\hat{\mathbf{x}}^{(s)}_{\text{ero}}$, $\hat{\mathbf{x}}^{(s)}_{\text{nom}}$, and $\hat{\mathbf{x}}^{(s)}_{\text{dil}}$. The projection is applied to $\bar{\mathbf{x}}^{(m)}$ with only the nominal threshold. The projection sharpness β is increased from 1 to 20 between iterations 10 and 210 in an exponential manner. Finally, the density and mixture fields are converted to the material density fields:

$$\boldsymbol{\mu}^{(1)} = \hat{\mathbf{x}}^{(s)}\hat{\mathbf{x}}^{(m)} \tag{4.22a}$$

$$\boldsymbol{\mu}^{(2)} = \hat{\mathbf{x}}^{(s)}\left(1 - \hat{\mathbf{x}}^{(m)}\right), \tag{4.22b}$$

after which the overhang filter is applied. The order of the materials is of importance in the stiffness interpolation, and thus the expressions in Eq. (4.22) are flipped if the Young's modulus of material 2 is higher than that of material 1.

The printable density fields $\boldsymbol{\xi}^{(1)}$ and $\boldsymbol{\xi}^{(2)}$ are then converted back to the RMMI fields $\boldsymbol{\chi}^{(s)}$ and $\boldsymbol{\chi}^{(m)}$ for the stiffness interpolation, the equation for which will be presented in the next sections, as it is specific to each problem. An overview of the optimization settings used by all three problems can be found in Table 4.1. For all cases in this paper, material 1 refers to the red material, whose properties are varied. Material 2 refers to the blue material, whose properties remain constant.

4.5.1 2D compliance minimization

The first study considers a 2D MBB beam (Olhoff et al., 1991). A visual representation of the load case is shown in Figure 4.5. The design is discretized with 180 × 60 bilinear quadrilateral elements.

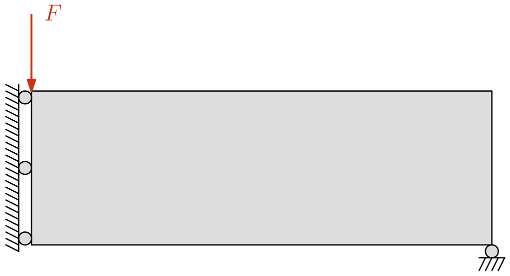


Figure 4.5: Illustration of the MBB problem definition.

Symbol	Value	Description
c_0	10	Scaling factor for objective and constraints
τ_f	10^{-5}	Relative objective change stopping criterium
τ_x	10^{-5}	Relative design change stopping criterium
n_{iter}	400	Maximum number of iterations
$E^{(2)}$	100 GPa	Young's modulus of the blue reference material
$\rho^{(2)}$	$1000\,\text{kg}\,\text{m}^{-3}$	Density of the blue reference material
$\alpha^{(2)}$	45°	Critical overhang angle of the blue reference material
ν	0.33	Poisson ratio
p	3	Intermediate density penalization
χ_{min}	10^{-9}	Minimum density
β_{max}	20	Maximum projection sharpness
ξ_0	0.5	Overestimation suppression threshold

Table 4.1: Parameter values used by all problems.

The reference material, with constant properties, is displayed in blue. Its material properties are specified in Table 4.1. The alternative material is displayed in red. The density of both materials is kept constant with a value of $1000\,\text{kg}\,\text{m}^{-3}$, while the Young's modulus and critical overhang angle of the red material vary. The Young's modulus for an element is interpolated as:

$$E_e = \check{\chi}^{(s)}_{\text{ero},e}\Big(E_{\text{max}}\big(\chi^{(m)}_{2\text{ero},e}\big)^p + E_{\text{min}}\Big(1-\big(\chi^{(m)}_{2\text{ero},e}\big)^p\Big)\Big), \tag{4.23}$$

where E_{max} is always the Young's modulus of the stiffer material, and E_{min} is always the Young's modulus of the more compliant material. Both density and mixtures are penalized with parameter p. Printable density $\chi^{s}_{1\text{ero},e}$ is modified as:

$$\check{\chi}^{(s)}_{1\text{ero},e} = \chi_{\text{min}} + (1-\chi_{\text{min}})\,\chi^{(s)}_{1\text{ero},e}, \tag{4.24}$$

to avoid singularities (Sigmund, 2007). The optimization problem is defined as:

$$\begin{aligned}
\min_{\mathbf{x}^{(s)},\mathbf{x}^{(m)}} \quad & c = \mathbf{f}^T\mathbf{u} \\
\text{s.t.} \quad & \mathbf{K}\mathbf{u} = \mathbf{f} \\
& \frac{s\,m_{\text{dil}}}{m_{\text{max}}} - 1 \leq 0 \\
& 0 \leq \mathbf{x}^{(s)} \leq 1 \\
& 0 \leq \mathbf{x}^{(m)} \leq 1,
\end{aligned}$$

where c is the compliance of the design, and $\mathbf{f}$ is the applied force vector. The scaling factor s is the ratio between the dilated and nominal mass, and is used to scale down the mass of the dilated design to the mass of the nominal design every 25 iterations. The stiffness matrix $\mathbf{K}$ is based on the stiffness of the eroded fields, and is assembled from the element stiffness matrices $\mathbf{k}^0_{\text{e}}$ as follows

$$\mathbf{K} = \sum_e E_e\left(\chi^{(s)}_{\text{ero},e}, \chi^{(m)}_{\text{ero},e}\right)\mathbf{k}^0_e, \tag{4.25}$$

with E_e as in Eq. (4.23). The displacement vector $\mathbf{u}$ is obtained with linear Finite Element Analysis. The mass used in the constraint is based on the dilated design:

$$m_{\text{dil}} = \sum_e \rho^{(1)}\xi^{(1)}_{\text{dil},e} + \rho^{(2)}\xi^{(2)}_{\text{dil},e}. \tag{4.26}$$

Final compliance values reported are based on the nominal design. The settings that are specific to this problem are shown in Table 4.2.

Symbol	Value	Description
F	10 N	Applied force
m_{max}	5.8 g	Maximum mass
r	4 el.	Density filter radius
η_{ero}	0.6	Eroded threshold
η_{nom}	0.5	Nominal threshold
η_{dil}	0.4	Dilated threshold

Table 4.2: Parameter values used for the 2D MBB problem.

4.5.2 2D eigenfrequency maximization

The second numerical example is an eigenfrequency maximization problem of a 2D elevated platform. The settings that are specific to this problem are shown in Table 4.3, and an illustration of the boundary conditions are provided in

Figure 4.6. The darker shade of gray illustrates the platform, and has a minimum volume fraction constraint of 100% for the reference material. The volume of this region is defined via a constraint to ensure the platform itself is also required to be properly supported. The lighter shade of gray illustrates the frame domain which provides support to the platform, and has a maximum volume fraction constraint of 40%, irrespective of the material choice. The frame domain has a size of 120 × 120 elements, and the platform domain is 120 × 20 elements.

Symbol	Value	Description
r	4 el.	Density filter radius
$v_{\mathrm{f,max}}$	0.4	Frame vol. frac.
$v_{\mathrm{p,min}}$	1.0	Platform vol. frac.
η_{ero}	0.6	Eroded threshold
η_{nom}	0.5	Nominal threshold
η_{dil}	0.4	Dilated threshold

Table 4.3: Parameter values used by the 2D eigenfrequency problem.

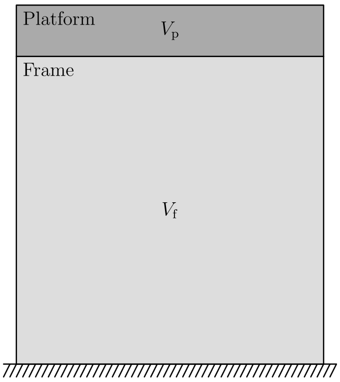


Figure 4.6: Schematic illustration of the 2D elevated platform. The dark region denotes the platform, and the lighter region denotes the design domain for the frame. V_{p} and V_{f} represent the volumes of each domain.

We again employ the Heaviside projection in Eq. (4.21) with three different thresholds, with the eigenfrequencies calculated using the stiffness matrix **K** of the eroded design, and the mass matrix **M** of the dilated design. The blue reference material is again kept constant with a density of 1000 kg m^{-3} and a Young's modulus of 100 GPa. Both the density and the Young's modulus of the red alternative material are varied. For this, a scaling factor R is introduced, which scales the material properties of the alternative material with respect to the reference material, e.g. $R = 0.4$ means the alternative material has a Young's modulus of 40 GPa, and a density of 400 kg m^{-3}.

The optimization problem reads:

$$\begin{aligned}
\min_{\mathbf{x}^{(\mathrm{s})},\mathbf{x}^{(\mathrm{m})}} \quad & h = \frac{1}{\omega_1^2} + \frac{1}{\omega_2^2} + \frac{1}{\omega_3^2} \\
\text{s.t.} \quad & \left(\mathbf{K} - \omega_k^2 \mathbf{M}\right) \varphi_k = \mathbf{0}, \quad \text{for } k = 1,2,3 \\
& 1 - \frac{v_{\mathrm{p}}\left(\mathbf{x}^{(\mathrm{s})}, \mathbf{x}^{(\mathrm{m})}\right)}{v_{\mathrm{p,min}}} \leq 0 \\
& \frac{v_{\mathrm{f}}\left(\mathbf{x}^{(\mathrm{s})}, \mathbf{x}^{(\mathrm{m})}\right)}{v_{\mathrm{f,max}}} - 1 \leq 0 \\
& 0 \leq \mathbf{x}^{(\mathrm{s})} \leq 1 \\
& 0 \leq \mathbf{x}^{(\mathrm{m})} \leq 1
\end{aligned}$$

where the objective h is the harmonic mean of the lowest three eigenfrequencies ω_1, ω_2, and ω_3 (Ma et al., 1995). The mass matrix **M** is based on the dilated density fields:

$$\mathbf{M} = \sum_e \left(\xi^{(1)}_{\mathrm{dil},e} + \xi^{(2)}_{\mathrm{dil},e}\right) \mathbf{m}_e^0, \tag{4.27}$$

where $\mathbf{m}_{\mathrm{e}}^0$ denotes the element mass matrix. The density interpolation for the mass matrix uses a piecewise formulation from Du and Olhoff (2007):

$$\rho_e^{(m)} = \begin{cases} \rho^{(m)} \left(\xi^{(m)}_{\mathrm{dil},e}\right)^6 / 0.1^5, & \xi^{(m)}_{\mathrm{dil},e} < 0.1 \\ \rho^{(m)} \xi^{(m)}_{\mathrm{dil},e}, & \text{else}, \end{cases} \tag{4.28}$$

for materials $m = 1, 2$. This formulation significantly reduces the mass of the structure for densities below 0.1 in order to avoid localized eigenmodes.

The Young's modulus is calculated using Eq. (4.23), but for this problem we use a corrected density by Zhu et al. (2009):

$$\begin{aligned}
\check{\chi}^{(\mathrm{s})}_{\mathrm{ero}} = {} & \left(\chi_{\min} + (1 - \chi_{\min})\right) \\
& \left(0.01 \chi^{(\mathrm{s})}_{\mathrm{ero}} + 0.99 \left(\chi^{(\mathrm{s})}_{\mathrm{ero}}\right)^p\right),
\end{aligned} \tag{4.29}$$

which includes a small linear part for densities below 0.01 to avoid excessive displacements in (near) void regions.

4.5.3 3D eigenfrequency maximization

The elevated platform study is also performed in 3D, with a frame domain of 40 × 40 × 28 elements, and a platform thickness of 2 elements. The volume fraction constraint in the frame domain is changed to 25%. The rest of the problem formulation follows the 2D problem, though only a single Heaviside projection threshold is used.

The settings that are specific to this problem are provided in Table 4.4, and the boundary conditions are shown in Figure 4.7.

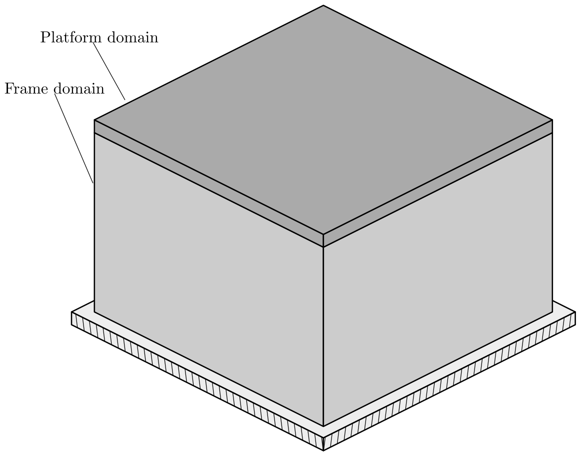


Figure 4.7: Schematic illustration of the 3D elevated platform problem definition. The darker shade of gray denotes the platform domain, with a volume constraint of 100%. The lighter shade of gray denotes the frame domain, with a volume constraint of 25%.

Symbol	Value	Description
r	2 el.	Density filter radius
$v_{\mathrm{f,max}}$	0.25	Frame vol. frac.
$v_{\mathrm{p,min}}$	1.0	Platform vol. frac.
η	0.5	Projection threshold

Table 4.4: Parameter values used by the 3D eigenfrequency problem.

4.6 Results

4.6.1 Compliance minimization of a 2D MBB beam

Figure 4.8 shows a single-material design with a critical overhang angle of 45°, a density of 1000 kg m^{-3}, and a Young's modulus of 100 GPa, as per Table 4.1. Figure 4.9 shows two multi-material designs without overhang filter. In Figure 4.9a the red alternative material is made more compliant than the blue reference material, and has a Young's modulus of 50 GPa. In Figure 4.9b the red material is made stiffer, with a Young's modulus of 150 GPa. Note that for a compliance problem, the optimizer will always tend towards the material with the highest specific stiffness if there is no other advantage to using the weaker material, such as a difference in critical overhang angle.

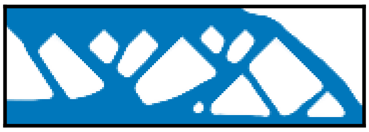


Figure 4.8: Reference single-material design for the MBB beam with $\alpha^{(2)} = 45°$. Material properties according to Table 4.1.

(a) $E^{(1)} = 50\,\mathrm{GPa}$.

(b) $E^{(1)} = 150\,\mathrm{GPa}$.

Figure 4.9: Multi-material MBB designs without overhang constraint. Material properties according to Tables 4.1 and 4.2.

The designs in Figs. 4.8 and 4.9 are used as a reference for the parameter sweep in Table 4.5. The top row in Table 4.5 contains designs with $E^{(1)} = 50\,\mathrm{GPa}$, matching the material properties of the result in Figure 4.9a, but with an overhang filter applied. The bottom row contains designs with $E^{(1)} = 150\,\mathrm{GPa}$, matching the material properties of the design in Figure 4.9b. The critical overhang angle of the red material varies from 15° in the leftmost column, to 60° in the rightmost column. The compliance of the designs is normalized against the compliance of both the single-material design c_{sm}, and the compliance of the multi-material designs without overhang filter c_1 and c_2, shown in Figure 4.9.

The multi-material designs in the top row of Table 4.5 are mainly made up of the blue reference material, as this has the highest specific stiffness. Their performance closely matches the single-material design. The design with $\alpha^{(1)} = 15°$ makes use of a small amount of the alternative material to allow steeper overhang angles to be used, and can thus slightly outperform the single-material design with $\alpha^{(2)} = 45°$ (Figure 4.8). The design with $\alpha^{(1)} = 30°$ uses the red material for the same purpose, but to a lesser extent, since the critical overhang angle is stricter. Its performance matches that of the single-material design. The designs with $\alpha^{(1)} = 45°$ or 60° can no longer take

	$\alpha^{(1)} = 15°$	$\alpha^{(1)} = 30°$	$\alpha^{(1)} = 45°$	$\alpha^{(1)} = 60°$
$E^{(1)} =$ 50 GPa	 $c/c_{\mathrm{sm}} = 98.3\%$ $c/c_1 = 101.8\%$	 $c/c_{\mathrm{sm}} = 100.2\%$ $c/c_1 = 103.8\%$	 $c/c_{\mathrm{sm}} = 100.4\%$ $c/c_1 = 104.0\%$	 $c/c_{\mathrm{sm}} = 100.4\%$ $c/c_1 = 103.9\%$
$E^{(1)} =$ 150 GPa	 $c/c_{\mathrm{sm}} = 63.7\%$ $c/c_2 = 98.7\%$	 $c/c_{\mathrm{sm}} = 64.3\%$ $c/c_2 = 99.7\%$	 $c/c_{\mathrm{sm}} = 66.4\%$ $c/c_2 = 103.0\%$	 $c/c_{\mathrm{sm}} = 69.1\%$ $c/c_2 = 107.2\%$

Table 4.5: Minimum compliance MBB beam results, for various critical overhang angles and Young's moduli of the red material. All other material properties according to Tables 4.1 and 4.2. The relative compliances are shown below the designs. c_{sm} refers to the compliance of the design shown in Figure 4.8. c_1 and c_2 refer to the compliance of the designs in Figure 4.9a and Figure 4.9b respectively.

advantage of the alternative material to allow for steeper overhangs, and thus closely match the single-material design in both performance and geometry. All designs in the top row perform worse than the reference multi-material design in Figure 4.9a. The compliance is in the range of 2 to 5 percent higher due to the varying levels of overhang constraint.

The designs on the bottom row of Table 4.5 are primarily made up of the red material, which for this row has a higher specific stiffness. The compliance relative to the single-material design is in the order of 64% to 70%, which matches the expected decrease, as the Young's modulus of the blue material of Figure 4.8 is two thirds that of the red material. The blue material has no added value when $\alpha^{(1)} = 15°, 30°$, or $45°$, and thus is not used there. In the design where $\alpha^{(1)} = 60°$, the weaker blue material is used to provide support, similar to what the red material is used for in the designs in the left side of the top row.

For the compliance problem, MMTO uses an alternative material primarily to allow for steeper overhang angles. In the first two designs on the top row, as well as in the rightmost design in the bottom row, the more compliant material is primarily used as a coating when it has a lower critical overhang angle. This is undesirable unless minimum-size filtering is added to ensure that this layer is sufficiently thick.

4.6.2 *Eigenfrequency maximization of a 2D elevated platform*

A parameter sweep similar to that of the compliance problem is performed for the 2D eigenfrequency optimization, the results of which are shown in Table 4.6. The relative objective is again compared to both a single material design of the blue reference material, shown in Figure 4.10, and multi-material designs without an overhang filter, which are shown in the leftmost column of Table 4.6.

Figure 4.10: Reference single-material design for the MBB beam. Material properties according to Table 4.1.

For the eigenfrequency problem, both the density and the Young's modulus of the alternative material are varied by multiplying them with a factor R, ranging from 0.4 for the results on the top row to 1.6 for the bottom row. Each variation is again

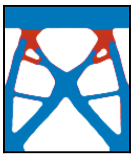
	No overhang filter	$\alpha^{(1)} = 15^\circ$	$\alpha^{(1)} = 30^\circ$	$\alpha^{(1)} = 45^\circ$	$\alpha^{(1)} = 60^\circ$
$R = 0.4$	 $h/h_{\mathrm{sm}} = 90.9\%$ $h/h_1 = 100.0\%$	 $h/h_{\mathrm{sm}} = 92.4\%$ $h/h_1 = 101.7\%$	 $h/h_{\mathrm{sm}} = 94.7\%$ $h/h_1 = 104.2\%$	 $h/h_{\mathrm{sm}} = 99.5\%$ $h/h_1 = 109.5\%$	 $h/h_{\mathrm{sm}} = 100.0\%$ $h/h_1 = 110.0\%$
$R = 0.8$	 $h/h_{\mathrm{sm}} = 92.5\%$ $h/h_2 = 100.0\%$	 $h/h_{\mathrm{sm}} = 97.6\%$ $h/h_2 = 105.4\%$	 $h/h_{\mathrm{sm}} = 96.8\%$ $h/h_2 = 104.6\%$	 $h/h_{\mathrm{sm}} = 96.4\%$ $h/h_2 = 104.2\%$	 $h/h_{\mathrm{sm}} = 96.7\%$ $h/h_2 = 104.4\%$
$R = 1.2$	 $h/h_{\mathrm{sm}} = 82.4\%$ $h/h_3 = 100.0\%$	 $h/h_{\mathrm{sm}} = 84.5\%$ $h/h_3 = 102.6\%$	 $h/h_{\mathrm{sm}} = 87.6\%$ $h/h_3 = 106.3\%$	 $h/h_{\mathrm{sm}} = 86.8\%$ $h/h_3 = 105.4\%$	 $h/h_{\mathrm{sm}} = 88.9\%$ $h/h_3 = 108.0\%$
$R = 1.6$	 $h/h_{\mathrm{sm}} = 70.6\%$ $h/h_4 = 100.0\%$	 $h/h_{\mathrm{sm}} = 75.6\%$ $h/h_4 = 107.1\%$	 $h/h_{\mathrm{sm}} = 76.9\%$ $h/h_4 = 109.0\%$	 $h/h_{\mathrm{sm}} = 77.5\%$ $h/h_4 = 109.8\%$	 $h/h_{\mathrm{sm}} = 77.5\%$ $h/h_4 = 109.8\%$

Table 4.6: Resulting designs for a parameter sweep for the 2D eigenfrequency maximization problem. The leftmost column shows the multi-material reference designs without an overhang filter. The relative objective is shown below each design, with h_{sm} referring to the single-material design in Figure 4.10, and h_1 to h_4 referring to the multi-material design without overhang filter in the respective row.

tested for overhang angles ranging from 15° to 60°. The blue reference material's properties are defined in Table 4.1. All variations resulted in a multi-material design, unlike the minimum compliance MBB beam. The multi-material designs also all outperform the single-material design, proving that there is performance to be gained by expanding the design domain with multi-materials.

A clear pattern is present in Table 4.6: the bottom of the structure is always made of the stiffer and denser material. For a single material structure the natural frequencies do not change if both the Young's modulus and the density are scaled by an equal amount, since the eigenfrequency depends only on the ratio of the mass and stiffness. For a multi-material structure however, this no longer holds. Close to the constrained boundary elements, increasing the stiffness is more beneficial than reducing the mass, as the stiffness also affects the displacement amplitude of the structure farther away from the boundary. Near the top of the structure the opposite holds; Here the displacements are large, and the contribution of the element stiffness to the stiffness of the overall structure is low. Minimizing the mass becomes critical in this region, and this is reflected in the results.

In the cases where the alternative (red) material is both lighter and more compliant, it is primarily used to provide support for the platform. Even when $\alpha^{(1)} = 60°$, the lower mass of the alternative material outweighs the stricter overhang limitation, and it is used to create support structures, though in a more selective manner. In the cases where the alternative material is both denser and stiffer, it is used to create the primary structure, and the reference material is used to support the platform.

When the critical overhang angle of the alternative material becomes more strict, improvement in performance with respect to the single-material design diminishes. However, since the design problem is non-convex, different local minima result with some outliers. In the bottom two rows of Table 4.6, this trend is not quite as strong as in the top rows, since the overhang angles at lower parts of the structure are not as critical. The performance improvement compared to the single-material design is also far greater if the alternative material is stiffer and denser: the bottom rows in Table 4.6 consistently show a lower objective than the top rows. The higher density of the alternative material, which makes up the majority of the frame in bottom rows, means that the frame now takes on a larger share of the total mass of the structure, compared to the platform. This moves the mass distribution closer to the fixed boundary elements, improving the results.

There is a prevalence of intermediate densities in the designs of the rightmost row for $R = 0.4$ and $R = 1.6$. This occurs because these intermediate density structures exist primarily to support the platform. Further research showed that neither Heaviside projection nor increasing the value of ξ_0 removes these intermediate densities. The optimizer balances the densities of these elements to the minimum needed to provide support, and there is no incentive for it to increase the densities beyond this. A different penalization on intermediate densities could thus be required.

4.6.3 *Eigenfrequency maximization of a 3D elevated platform*

The parameter sweep performed for the 2D platform is repeated for the 3D platform to demonstrate that the filter functions in 3D as well. The normalized objectives are shown in Figure 4.14 and Figure 4.15. The harmonic mean is again normalized against the harmonic mean of a single material reference design with overhang filter, shown in Figure 4.11, and four multi-material reference designs without overhang filter, two of which are shown in Figure 4.12. The left picture in each figure shows the full structure from the outside, while the right picture shows a cut view to expose the internal geometry.

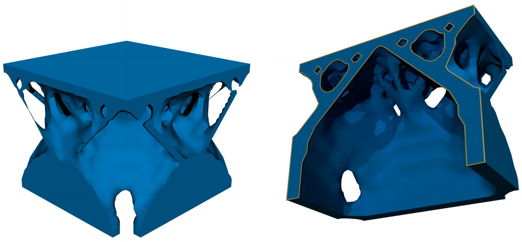
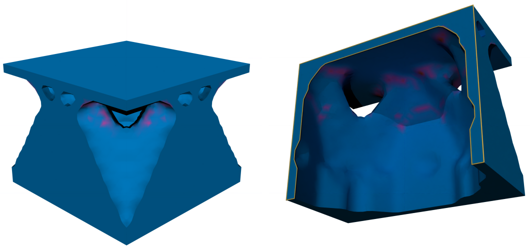
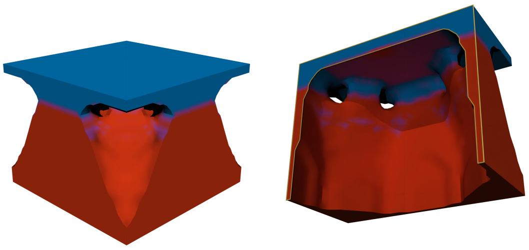


Figure 4.11: Reference single-material design for the 3D elevated platform. Critical overhang angle of 45°.

Figure 4.12a shows a design where the alternative material is lighter and more compliant than the reference material, and Figure 4.12b shows a design where it is denser and stiffer. The other two reference designs closely match these in geometry, and thus have been omitted from the figure. Similar to the 2D case, the lower part of the structure is made of the stiffer and denser material. However, unlike the 2D case, the lighter alternative material is not used to support the platform in Figure 4.12a. The coarser mesh used in the 3D case will more heavily penalize changes in mixture, as intermediate density material exists on the interface due to the density filter. This lowers the stiffness of the structure and is the likely reason why the red material remains unused.



(a) $R = 0.8$.

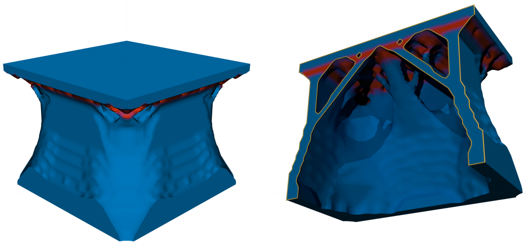


(b) $R = 1.2$.

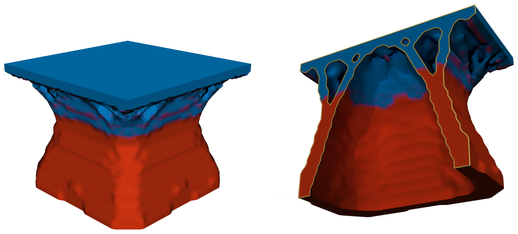
Figure 4.12: Reference multi-material designs for the 3D elevated platform. No overhang filter.

A selection of designs from the parameter sweep are shown in Figure 4.13 that together cover the four types of designs: Two designs with an alternative material that has a more lenient critical overhang angle; one with a lighter and more complaint alternative material (Fig. 4.13a), and one with a stiffer and denser alternative material (Fig. 4.13b), and two designs with an alternative material that has a stricter critical overhang angle; again one with a lighter and more compliant alternative material (Fig. 4.13c), and one with a stiffer and denser alternative material (Fig. 4.13d).

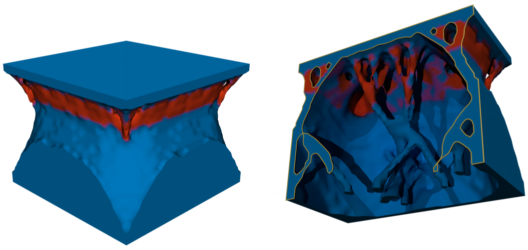
Figure 4.13a shows a design where the density and Young's modulus of the alternative material are multiplied with $R = 0.4$, and which has a critical overhang angle of 30°. In this design, the alternative material is only used locally to provide support for the platform. Unlike the single-material design in Fig. 4.11, the red material is used here to provide support. The more lenient critical overhang angle gives it a distinct advantage over the reference material that it did not have without the filter.



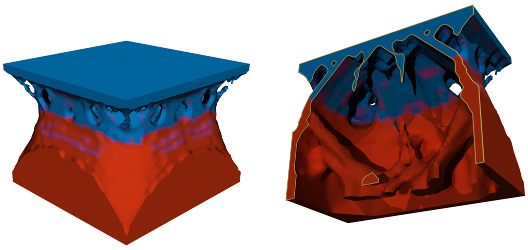
(a) $R = 0.4$, $\alpha^{(1)} = 30°$.



(b) $R = 1.6$, $\alpha^{(1)} = 15°$.



(c) $R = 0.4$, $\alpha^{(1)} = 60°$.



(d) $R = 1.6$, $\alpha^{(1)} = 45°$.

Figure 4.13: Multi-material design with overhang constraint for the 3D elevated platform.

The design in Figure 4.13b has an alternative material with with $R = 1.6$, and a critical overhang angle of 15°. In this design, the main structure is made up of the alternative material, but the support for the platform is not, even though the alternative material has a lower critical overhang angle than the reference material. This suggests that for regions far away from the constrained boundary elements, the negative effects of the higher density outweigh the positive effects of a more lenient critical overhang constraint.

In the design in Fig. 4.13c the alternative material is again lighter and more compliant than the reference material, with $R = 0.4$, but it has a stricter overhang angle of 60°. This design makes use of an internal structure to support the platform, and like the design in Fig. 4.13a the alternative material is only used to provide support for the platform.

Finally, in Figure 4.13d we see a design with an alternative material that has $R = 1.6$, but also has a strict critical overhang angle of 45°. The primary structure is again made up of the alternative material, much like the design in Fig. 4.13b, and an internal structure similar to that in Fig. 4.13c is created to provide support for the platform.

Figure 4.14 shows the relative objectives with respect to the single-material design with overhang filter. The normalized harmonic mean is plotted against the critical overhang angle for all ratios R of the alternative material. Just like in the 2D case, all multi-material designs perform better than the single-material design.

For all four lines, there is a jump in relative harmonic mean when $\alpha^{(1)}$ increases from 30° to 45°, which is a phenomenon that is not present in the 2D sweep. In the 3D problem, the rigidity of the frame is maximized by creating a vertical hollow column with a wide cross-section. If the critical overhang angle of both materials is above 45°, an internal structure is used to provide support for the platform, which does not contribute to increasing the natural frequency. In the 2D case however, rigidity is maximized by adding cross bars between the two main legs. Even when the critical overhang angle of the alternative material is low, the material that makes up the bracing maintains an overhang angle of approximately 45°. Thus the effect is not present here.

The margins in 3D are greater however, with the best multi-material design having a relative harmonic mean of only 63.9%, versus 75.6% for the best 2D design. The designs where the alternative material is stiffer and denser than the reference material also tend to score better than the ones where it is lighter and more compliant. This is also a trend that was already present in the 2D case. An interesting observation is that for three out of the four cases, the design with a critical overhang angle of 30° outperforms the design with a more lenient angle of 15°. This could be explained by there being more intermediate density material when material is interpolated at 15°, as the interpolation points are farther away from the element centroids.

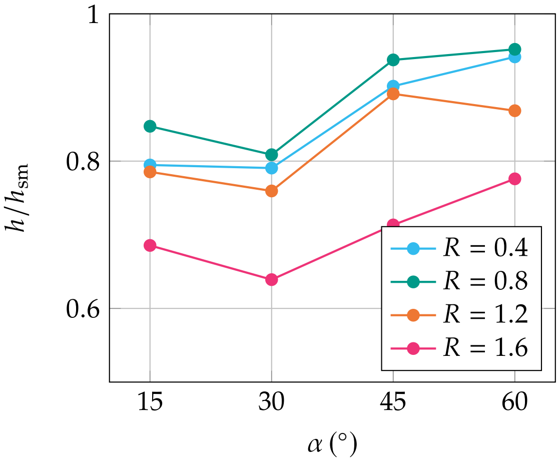


Figure 4.14: Relative objective for the 3D eigenfrequency problem, plotted against critical overhang angles for four ratios of the alternative material. The harmonic mean h_{sm} of the single-material design in Figure 4.11 is used to normalize the results.

Figure 4.15 shows the harmonic mean of the designs normalized against the four multi-material reference designs without overhang filter. The designs with a shallow overhang constraint (15° and 30°) consistently outperform the design without the overhang filter, arising from the non-convexity of the problem. Only the designs with a critical overhang angle of 45° or higher perform worse than the design without an overhang constraint, with a few exceptions.

As shown in Figs. 4.13c and 4.13d, designs where the critical overhang angle of the alternative material is high tend towards a solution with an internal support structure. The material that is used for this does not contribute significantly to increasing the eigenfrequency, leaving less material for use in the main frame. This is likely the cause of the increased harmonic mean.

4.7 Discussion

While the entire algorithm is continuous and differentiable, the layer-wise coupling between the materials makes derivation complex. This is exacerbated when more materials are added, and makes extending this filter for 3 or more materials cumbersome.

The compliance case shows that the optimization will often lead to designs with a coating on the overhanging surfaces, made up of the

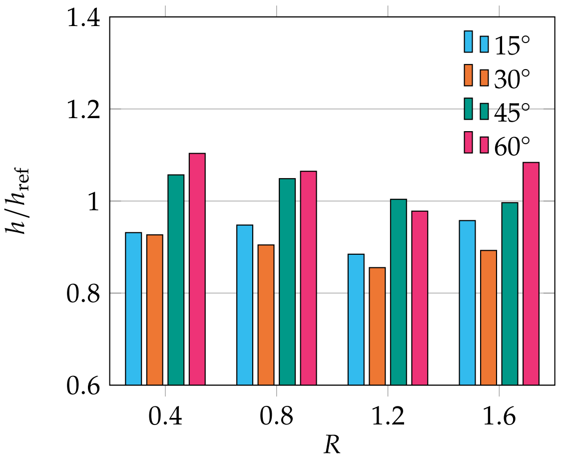


Figure 4.15: Relative harmonic mean for the 3D eigenfrequency problem, for four critical overhang angles. h_{ref} refers to the harmonic mean of the four multi-material designs in Figure 4.11. Each cluster of columns is normalized against the reference design with the same value for R.

material with the lowest critical overhang angle. The thickness of this coating is currently not controllable, and is determined exclusively by the mesh size. Thin coatings will in all likelihood not be able to provide sufficient support for the structure to make these kinds of designs manufacturable. Minimum feature size filters exist for SMTO (see Sigmund, 2007), but for MMTO there is currently no reliable method to control the size of material islands, that can be implemented in RMMI.

As can be seen in Figure 4.4, the overhang filter produces a gradient on the overhanging surface whenever the interpolation points do not coincide with the element centroids. This effect is already present in the single-material variant of the filter, and is primarily noticeable for critical overhang angles greater than 45°. As a result the optimizer has more difficulties converging to a 0-1 design for cases with strict overhang angles. Heaviside projection can be used to remove the intermediate density material, but this would change the effective overhang angle of the filter. For shallow overhang angles of 45° or less this effect is negligible.

The overhang angle at the apex of arch-like features can be inaccurate for coarse meshes, since two adjacent elements are allowed to exist with no material directly underneath them. When post-processing these designs, the local overhang angle may exceed the critical overhang angle, as can be seen in Figure 4.13b.

4.8 Conclusion

This paper proposes an overhang filter for topology optimization that removes unsupported material in multi-material structures. It restricts the density of each material such that the sum of densities does not exceed the total available support density. The support is based on the density sum of all materials in a circular support region directly below the element.

Numerical pattern tests confirmed that the filter limits the overhang angle accurately for a wide range of critical angles, and that the density overestimation due to the smooth maximum can be effectively suppressed using the same approach used for the single-material filter. The overestimation suppression parameter ξ_0 represents a trade-off between sufficiently suppressing overestimation, and not removing supported material of intermediate density. Analysis shows that the behaviour and sensitivity of this parameter closely matches that of the single-material filter, and that a similar range of reasonable values can be used. The filter can be directly implemented in any existing density-based topology optimization methods, and works in both 2D and 3D.

For the compliance problem the optimizer will always converge to a design made of the material with the highest specific stiffness. An alternative material is used if it has a more lenient critical overhang angle, but the thin coatings shown in the results are not manufacturable. In the eigenfrequency problem, the optimizer makes proper use of the different materials available, and each multi-material structure outperforms the single-material reference design. Compared to the multi-material designs without overhang filter, the filtered designs have minimal performance loss while ensuring printability for critical overhang angles up to 60°.

This filter allows engineers to create multi-material designs in topology optimization that are support-free. When combined with enclosed void detection and constraints on the size and number of material islands, it enables the design of manufacturable multi-material parts that are printable with minimal post-processing. The custom angle support for each material phase allows engineers to tailor the filter to a wide range of materials. The development of minimum feature size controls for MMTO remains an open challenge that warrants further investigation.

5

Performance comparison

5.1. Problem formulation

This chapter provides a performance comparison between the hotspot constraint and the overhang filter; both in their multi-material forms. An equivalency between the overhang angle and pseudo-temperature is provided by Das (2023) for various member thicknesses, and this data has been used to convert the critical overhang angles used by the overhang filter to a suitable critical temperature for the hotspot constraint. The data for this is shown in Table 5.1. The assumed member thickness is 2 elements.

α	T_{cr}
30°	0.94
45°	0.85
60°	0.76

Table 5.1: Equivalent critical temperature for a given overhang angle. Data from Das (2023).

Tests are performed on an MBB beam with the same settings as in Chapter 3. The settings for the problem are given in Table 3.1, and the settings that are specific to the overhang filter are given in Table 5.2. The blue material again forms the reference material, with a Young's modulus twice that of the red alternative material. The critical overhang angle of the blue material is kept constant at 60°, or the equivalent critical temperature 0.76. The red material has a variable critical overhang angle and temperature.

Symbol	Value	Description
P	40	Smooth maximum exponent
n_{int}	2	Number of interpolation points
ξ_0	0.5	Overestimation suppression

Table 5.2: Overhang filter settings for performance comparison.

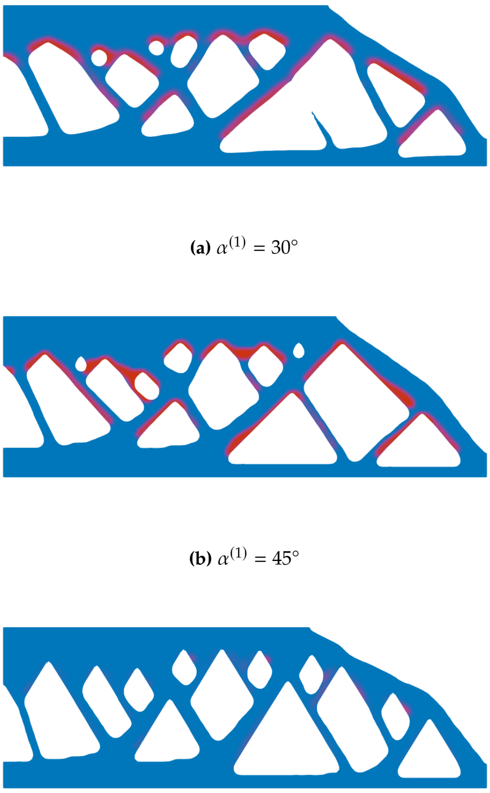
Both methods are implemented in Python with the PyMOTO library (Delissen et al.) to obtain the results for this thesis. The algorithms have been optimized for performance through layer-wise vectorization of the operations, and large matrices are stored in sparse format wherever possible. Wall clock times for both methods are measured on an Intel Core i7-8750H and 16 GB of RAM, with all optimizations performed on a single core.

5.2. Results

Both the overhang filter and the hotspot constraint are tested for three critical overhang thresholds of the red material. The designs are shown in Figure 5.1 and 5.2. The compliance of all the designs is plotted in Figure 5.3, and is again normalized against the reference design without overhang control in Figure 3.1.

In the designs where the red material has a lower critical overhang angle than the blue material (Figs. 5.1a, 5.1b, 5.2a, and 5.2b), the red material is again used as a coating on the overhanging surfaces. In Figs. 5.1c and 5.2c the red material has no added value, and is thus not used. This is consistent for both the overhang filter and the hotspot constraint, though the hotspot constraint exhibits a higher increase in compliance when the overhang threshold of the red material is decreased from 0.85 to 0.76, as can be seen in Figure 5.3. The design in Figure 5.1a uses an intermediate density member to provide support for one of the surfaces, which has been partially clipped away by the post-processing algorithm.

The overhang filter and the hotspot constraint each drive the design to a different local minimum, which is clearly visible by the overall shape of the part. The local minimum that the overhang filter consistently pushes the optimizer towards is a better performing one, and its objective, shown in Figure 5.3, is lower than that of the hotspot constraint for every case.

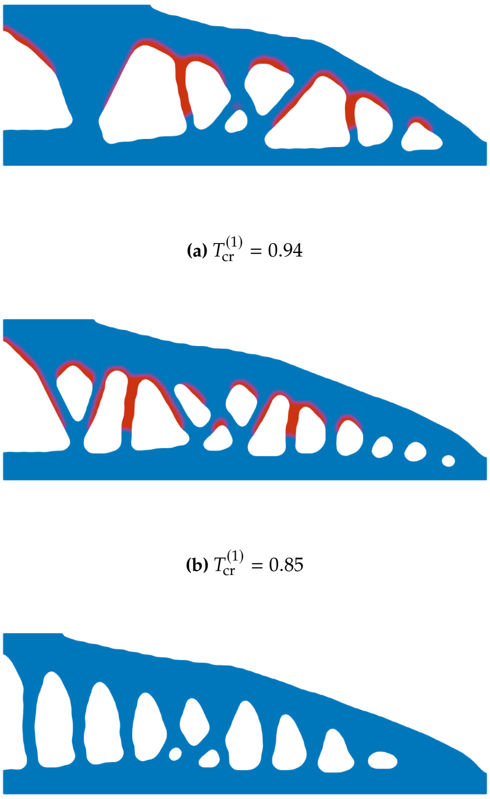


(a) $\alpha^{(1)} = 30°$

(b) $\alpha^{(1)} = 45°$

(c) $\alpha^{(1)} = 60°$

Figure 5.1: MBB beam with overhang filter, for various critical overhang angles of the red material. The critical overhang angle of the blue material is kept constant at 60°. Material properties according to Table 3.1 and 5.2.



(a) $T_{\mathrm{cr}}^{(1)} = 0.94$

(b) $T_{\mathrm{cr}}^{(1)} = 0.85$

(c) $T_{\mathrm{cr}}^{(1)} = 0.76$

Figure 5.2: MBB beam with hotspot constraint, for various critical temperatures of the red material. The critical temperature of the blue material is kept constant at 0.76. Material properties according to Table 3.1.

The time per iteration is where the biggest discrepancy between the two methods occurs. The performance impact of the hotspot constraint is far greater than that of the overhang filter. It increases the time per iteration by more than a factor of 5, compared to only a factor of 1.2 for the overhang filter.

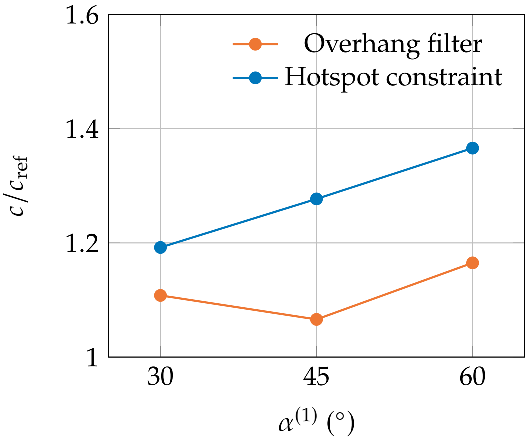


Figure 5.3: Relative compliance. Lower is better.

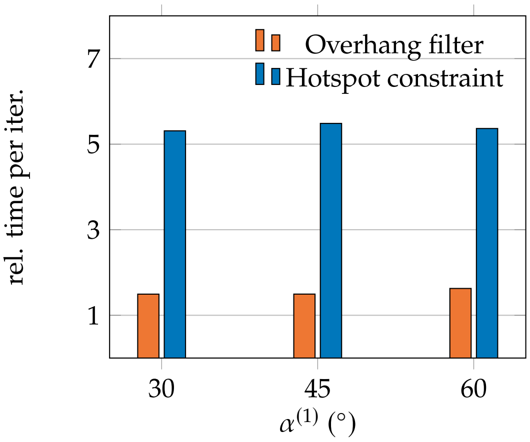


Figure 5.4: Relative time per iteration. Lower is better.

Figure 5.5 shows the convergence behaviour of the optimization that lead to the designs in Figs. 5.1b and 5.2b. The objective change over the iterations is shown as solid lines, and the hotspot constraint value is shown as a dashed line. The objective over time results show that while the overhang filter has a minimal impact on the convergence behaviour, the hotspot constraint disrupts it significantly. Every time the scaling factor $\bar{\alpha}$ is updated, the design temporarily destabilizes.

From these results it can be concluded that the multi-material overhang filter outperforms the multi-material hotspot constraint.

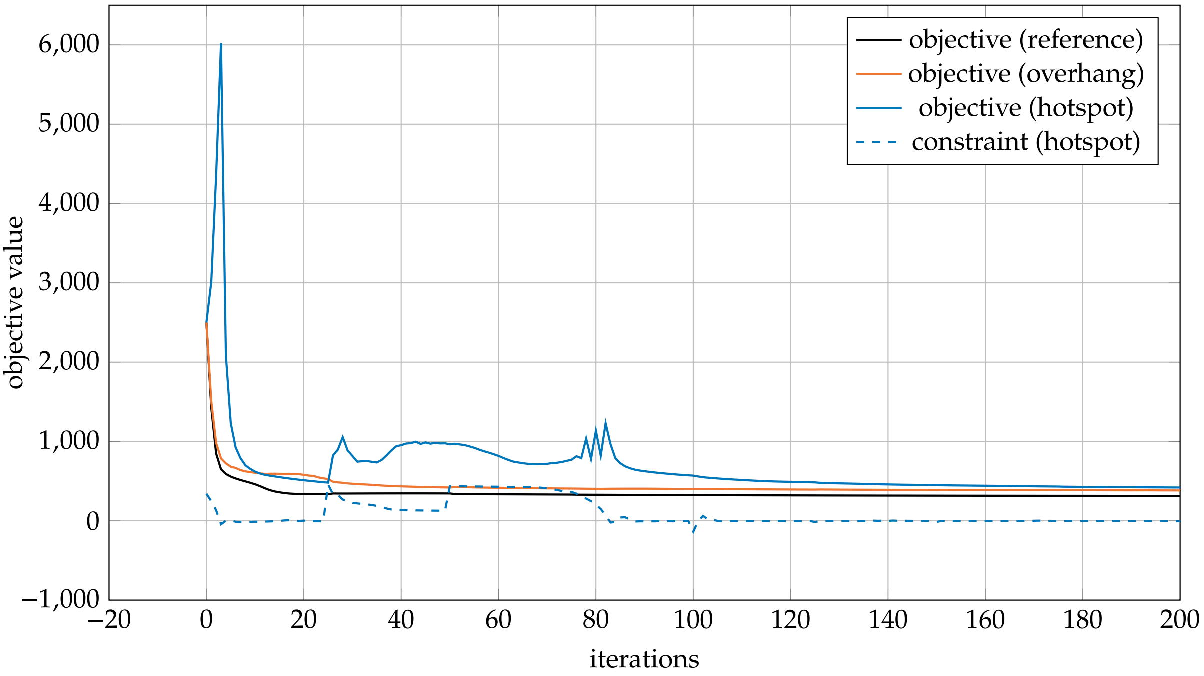


Figure 5.5: Convergence behaviour for the overhang filter and hotspot constraint. Reference problem without overhang control shown in black.

6

Conclusion

This thesis proposes two methods for controlling the overhang angle in MMTO. The first method is a multi-material extension to the geometric hotspot constraint, and the second is a novel multi-material overhang filter.

The overhang filter allows for explicit control of the overhang angle of each material separately, and can be added to a topology optimization scheme without major impact to the performance. It has a minimal impact on the convergence behaviour, and can generate printable designs for a wide range of critical overhang angles. Its functionality has been verified with numerical tests, and has been implemented in a variety of 2D and 3D problems.

The hotspot constraint has as its primary advantage that it aims more specifically at preventing overheating. Because this effect is not exclusively driven by overhang angle, the hotspot constraint selectively allows for steeper overhangs if there is enough surrounding geometry to dissipate the heat away from the melt pool. However, it disturbs the optimizer far more than is the case with the overhang filter, and in its current form it restricts the movement in the design space too much. Furthermore, the hotspot constraint needs a narrow window of filter settings to converge to a printable design. Unsupported stalactite-like structures often appear in the solution, and the optimizer cannot remove these in many cases, as their sensitivity information is zero. Moving the tip of a stalactite in any direction does not immediately change the maximum temperature, but rather only when it moves far enough to merge with another surface, and thus, the optimizer does not see the benefit in removing these.

From the performance comparison in Chapter 5, it is clear that the overhang filter outperforms the hotspot constraint in every metric. It has better convergence behaviour, lower computational effort, and consistently lower objective values.

6.1. Recommendations for future research

The long-term goal that this thesis contributes to, is to be able to create multi-material TO designs that are print-ready directly. To achieve this, the three major limitations that need to be accounted for in TO are removing enclosed voids, restricting the overhang angle, and controlling the minimum feature size.

Existing enclosed void detection methods are implementable in MMTO without major issues. They rely purely on being able to recognize the void phase, which is possible with all commonly used multi-material schemes. Limiting overhang angles is now also possible in MMTO, which means that the last major manufacturability limitation that remains for MMTO is minimum feature size. This will be needed to avoid the thin coatings seen on the results in this thesis. Currently the only minimum-size constraint for MMTO is designed for discrete TO, and there is none for the common RMMI or DMO interpolations. A standard robust formulation is not sufficient for creating designs where each material island has a minimum size, as the input fields of RMMI and DMO are independent. This provides an opportunity for future research.

A less critical manufacturing limitation, is that multi-material parts produced via PBF should not have a large number of small material islands, as this would lead to powder contamination. The results in this thesis do not show that designs with small material islands are a common output, but some designs do exhibit them. There presently exists a TO constraint that limits the number of islands, but this works exclusively with the Ordered SIMP scheme, and here too there is none that can be used in RMMI.

A possible future development of this filter is to adapt it to work with unstructured meshes. This is theoretically possible, as the cone interpolation is partially mesh-independent; it only depends on the mesh having layers with a consistent thickness in the deposition direction.

As can be seen in the overhang test in Fig. 4.4, there is a gradient on the overhanging surfaces when the critical overhang angle is high. This gradient causes an inaccuracy for high values of the critical overhang angle. A projection scheme that specifically targets this is another future research opportunity.

Bibliography

Aconity3D GmbH. Additive Manufacturing — aconity3d.com. `https://aconity3d.com/products/aconity-x/`, 2025. [Accessed 04-02-2025].

Aerosint. Aerosint - Multi-Material 3D Printing — aerosint.com. `https://aerosint.com/`, 2025. [Accessed 04-02-2025].

G. Allaire, F. Jouve, and A. Toader. A level-set method for shape optimization. *Comptes Rendus. Mathématique*, 334:1125–1130, 1 2002. ISSN 1778-3569. doi:10.1016/S1631-073X(02)02412-3.

G. Allaire, F. Jouve, and H. Maillot. Topology optimization for minimum stress design with the homogenization method. *Structural and Multidisciplinary Optimization*, 28:87–98, 9 2004. ISSN 1615-147X. doi:10.1007/s00158-004-0442-8.

A. Bandyopadhyay, Y. Zhang, and B. Onuike. Additive manufacturing of bimetallic structures. *Virtual and Physical Prototyping*, 17:256–294, 4 2022. ISSN 1745-2759. doi:10.1080/17452759.2022.2040738.

M. Bayat, O. Zinovieva, F. Ferrari, C. Ayas, M. Langelaar, J. Spangenberg, R. Salajeghe, K. Poulios, S. Mohanty, O. Sigmund, and J. Hattel. Holistic computational design within additive manufacturing through topology optimization combined with multiphysics multi-scale materials and process modelling. *Progress in Materials Science*, 138:101129, 2023. ISSN 0079-6425. doi:10.1016/j.pmatsci.2023.101129.

M.P. Bendsøe. Optimal shape design as a material distribution problem. *Structural Optimization*, 1: 193–202, 12 1989. ISSN 0934-4373. doi:10.1007/BF01650949.

M.P. Bendsøe and O. Sigmund. *Topology Optimization: Theory, Methods and Applications*. Springer Berlin, Heidelberg, 2 edition, 10 2002. ISBN 978-3-540-42992-0. doi:10.1007/978-3-662-05086-6.

D. Brackett, I. Ashcroft, and R. Hague. *Topology Optimization for Additive Manufacturing*. PhD thesis, Loughborough University, 8 2011.

T.E. Bruns and D.A. Tortorelli. Topology optimization of non-linear elastic structures and compliant mechanisms. *Computer Methods in Applied Mechanics and Engineering*, 190:3443–3459, 3 2001. ISSN 00457825. doi:10.1016/S0045-7825(00)00278-4.

F. Cheng, Q. Zhao, and L. Zhang. Non-probabilistic reliability-based multi-material topology optimization with stress constraint. *International Journal of Mechanics and Materials in Design*, 20:171–193, 2 2024. ISSN 1569-1713. doi:10.1007/s10999-023-09669-2.

A.T. Clare, P. Woizeschke, B. Rankouhi, F.E. Pfefferkorn, D. Bartels, M. Schmidt, and W.W. Wits. Metal multi-material additive manufacturing: Overcoming barriers to implementation. *CIRP Annals*, 74(2): 869–893, 2025. ISSN 0007-8506. doi:10.1016/j.cirp.2025.05.004.

M.N.S. Das. A geometric approach for controlling local overheating in topology optimization for additive manufacturing. M.sc. thesis, Delft University of Technology, Mekelweg 2, 2628 CD Delft, 8 2023.

M.N.S. Das, R. Ranjan, K. Wu, J. Wu, and C. Ayas. A physics-motivated geometric method for overheating prevention in topology optimization for additive manufacturing. *Computer Methods in Applied Mechanics and Engineering*, 447:118363, 12 2025. ISSN 00457825. doi:10.1016/j.cma.2025.118363.

A. Delissen, E. Boots, D. Laro, H. Kleijnen, F. van Keulen, and M. Langelaar. Realization and assessment of metal additive manufacturing and topology optimization for high-precision motion systems. *Additive Manufacturing*, 58:103012, 10 2022. ISSN 22148604. doi:10.1016/j.addma.2022.103012.

A. Delissen, S. Koppen, J. van der Zwet, and B. de Almeida. Pymoto: Modular framework for topology optimization with semi-automatic derivatives (v1.5.1), 2025.

DMG Mori. LASERTEC 125 DED hybrid - DMG MORI USA — us.dmgmori.com. `https://us.dmgmori.com/products/machines/additive-manufacturing/powder-nozzle/lasertec-125-ded-hybrid`, 2025. [Accessed 04-02-2025].

J. Du and N. Olhoff. Topological design of freely vibrating continuum structures for maximum values of simple and multiple eigenfrequencies and frequency gaps. *Structural and Multidisciplinary Optimization*, 34:91–110, 6 2007. ISSN 1615-147X. doi:10.1007/s00158-007-0101-y.

Y. Feng and T. Yamada. Multi-material topology optimization for additive manufacturing considering maximum build volume and assembly process. *Engineering Analysis with Boundary Elements*, 163: 616–640, 6 2024. ISSN 09557997. doi:10.1016/j.enganabound.2024.04.007.

Y. Feng, M. Noda, Y. Noguchi, K. Matsushima, and T. Yamada. Multi-material topology optimization for additive manufacturing considering dimensional constraints. *Computer Methods in Applied Mechanics and Engineering*, 410:116027, 2023. ISSN 0045-7825. doi:https://doi.org/10.1016/j.cma.2023.116027.

V. Florea, M. Pamwar, B. Sangha, and I.Y. Kim. 3d multi-material and multi-joint topology optimization with tooling accessibility constraints. *Structural and Multidisciplinary Optimization*, 60:2531–2558, 12 2019. ISSN 1615-147X. doi:10.1007/s00158-019-02344-1.

A.T. Gaynor, N.A. Meisel, C.B. Williams, and J.K. Guest. Topology optimization for additive manufacturing: Considering maximum overhang constraint. In *15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. American Institute of Aeronautics and Astronautics, 6 2014. ISBN 978-1-62410-283-7. doi:10.2514/6.2014-2036.

J. K. Guest, J. H. Prévost, and T. Belytschko. Achieving minimum length scale in topology optimization using nodal design variables and projection functions. *International Journal for Numerical Methods in Engineering*, 61(2):238–254, 2004. doi:https://doi.org/10.1002/nme.1064.

Z. Hashin and S. Shtrikman. A variational approach to the theory of the elastic behaviour of multiphase materials. *Journal of the Mechanics and Physics of Solids*, 11:127–140, 9 1963. doi:https://doi.org/10.1016/0022-5096(63)90060-7.

M. Hoffarth, N. Gerzen, and C. Pedersen. Alm overhang constraint in topology optimization for industrial applications. In *12th World Congress on Structural and Multidisciplinary Optimisation*, pages 1–11. Springer Braunschweig, Germany, 7 2017.

J. Hu, Y. Liu, Y. Luo, H. Huang, and S. Liu. Topology optimization of multi-material structures considering a piecewise interface stress constraint. *Computer Methods in Applied Mechanics and Engineering*, 398:115274, 8 2022. ISSN 00457825. doi:10.1016/j.cma.2022.115274.

X. Huang and W. Li. A new multi-material topology optimization algorithm and selection of candidate materials. *Computer Methods in Applied Mechanics and Engineering*, 386:114114, 12 2021. ISSN 00457825. doi:10.1016/j.cma.2021.114114.

A. Hussain and D. Kim. Fabrication of metal alloy structures with overhang features in laser-based powder bed fusion: A critical review of challenges and latest developments. *Journal of Manufacturing Processes*, 135:112–130, 2 2025. ISSN 15266125. doi:10.1016/j.jmapro.2025.01.031.

H. Irrinki, S.D. Nath, A.A. Akilan, and S.V. Atre. Laser powder bed fusion. In *Additive Manufacturing Processes*. ASM International, 06 2020. ISBN 978-1-62708-290-7. doi:10.31399/asm.hb.v24.a0006621.

N. Iyer, A.M. Mirzendehdel, S. Raghavan, Y. Jiao, E. Ulu, M. Behandish, S. Nelaturi, and D. Robinson. Pato: Producibility-aware topology optimization using deep learning for metal additive manufacturing. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 18:7459–7476, 12 2024. ISSN 1955-2513. doi:10.1007/s12008-024-01905-z.

W.E. King, A.T. Anderson, R.M. Ferencz, N.E. Hodge, C. Kamath, S.A. Khairallah, and A.M. Rubenchik. Laser powder bed fusion additive manufacturing of metals; physics, computational, and materials challenges. *Applied Physics Reviews*, 2:041304, 12 2015. doi:10.1063/1.4937809.

J. Kopatsch, M. Polzer, M. Bartz, and B. Wartzack. Integration of manufacturing constraints into multi-material topology optimization. In Dieter Krause, Kristin Paetzold-Byhain, and Sandro Wartzack, editors, *DS 133: Proceedings of the 35th Symposium Design for X (DFX2024)*, pages 231–240. The Design Society, 2024. doi:10.35199/dfx2024.24.

G. Kreisselmeier and R. Steinhauser. Application of vector performance optimization to a robust control loop design for a fighter aircraft. *International Journal of Control*, 37(2):251–284, 1983. doi:10.1080/00207179.1983.9753066.

M. Langelaar. An additive manufacturing filter for topology optimization of print-ready designs. *Structural and Multidisciplinary Optimization*, 55:871–883, 7 2016a. ISSN 16151488. doi:10.1007/s00158-016-1522-2.

M. Langelaar. Topology optimization of 3d self-supporting structures for additive manufacturing. *Additive Manufacturing*, 12:60–70, 10 2016b. ISSN 22148604. doi:10.1016/j.addma.2016.06.010. Expands the AM filter to a 3D case.

M. Langelaar. Combined optimization of part topology, support structure layout and build orientation for additive manufacturing. *Structural and Multidisciplinary Optimization*, 57:1985–2004, 5 2018. ISSN 1615-147X. doi:10.1007/s00158-017-1877-z.

M. Langelaar. Topology optimization for multi-axis machining. *Computer Methods in Applied Mechanics and Engineering*, 351:226–252, 7 2019. ISSN 00457825. doi:https://doi.org/10.1016/j.cma.2019.03.037.

B.S. Lazarov and O. Sigmund. Filters in topology optimization based on helmholtz-type differential equations. *International Journal for Numerical Methods in Engineering*, 86(6):765–781, 2011. doi:https://doi.org/10.1002/nme.3072.

M. Leary, L. Merli, F. Torti, M. Mazur, and M. Brandt. Optimal topology for additive manufacture: A method for enabling additive manufacture of support-free optimal structures. *Materials & Design*, 63: 678–690, 11 2014. ISSN 02613069. doi:10.1016/j.matdes.2014.06.015.

S. Li, Y. Zhang, and S. Liu. Multi-material topology optimization method for transient thermal structure with constraint on permissible temperatures. *International Journal for Numerical Methods in Engineering*, 124:4022–4057, 9 2023. ISSN 0029-5981. doi:10.1002/nme.7299.

Y. Li, C. Su, and J. Zhu. Comprehensive review of wire arc additive manufacturing: Hardware system, physical process, monitoring, property characterization, application and future prospects. *Results in Engineering*, 13:100330, 3 2022. ISSN 25901230. doi:10.1016/j.rineng.2021.100330.

H. Liu, C. Wang, Y. Zhang, and Y. Liang. Multi-material structural discrete variable topology optimization with minimum length scale control under mass constraint. *Computer Methods in Applied Mechanics and Engineering*, 420:116701, 2 2024. ISSN 00457825. doi:10.1016/j.cma.2023.116701.

J. Liu, A.T. Gaynor, S. Chen, Z. Kang, K. Suresh, A. Takezawa, L. Li, J. Kato, J. Tang, C.C.L. Wang, L. Cheng, X. Liang, and A.C. To. Current and future trends in topology optimization for additive manufacturing. *Structural and Multidisciplinary Optimization*, 57:2457–2483, 6 2018. ISSN 1615-147X. doi:10.1007/s00158-018-1994-3.

S. Liu, Q. Li, W. Chen, L. Tong, and G. Cheng. An identification method for enclosed voids restriction in manufacturability design for additive manufacturing structures. *Frontiers of Mechanical Engineering*, 10:126–137, 6 2015. ISSN 20950241. doi:10.1007/s11465-015-0340-3.

C. López, S. Burggraeve, P. Lietaert, J. Stroobants, X. Xie, S. Jonckheere, B. Pluymers, and W. Desmet. Model-based, multi-material topology optimization taking into account cost and manufacturability. *Structural and Multidisciplinary Optimization*, 62:2951–2973, 12 2020. ISSN 1615-147X. doi:10.1007/s00158-020-02641-0.

Z. Ma, N. Kikuchi, and H. Cheng. Topological design for vibrating structures. *Computer Methods in Applied Mechanics and Engineering*, 121(1):259–280, 1995. ISSN 0045-7825. doi:10.1016/0045-7825(94)00714-X.

R.P. Martukanitz. Directed-energy deposition processes. In *Additive Manufacturing Processes*. ASM International, 06 2020. ISBN 978-1-62708-290-7. doi:10.31399/asm.hb.v24.a0006549.

I. Meyer, M. Oel, T. Ehlers, and R. Lachmayer. Additive manufacturing of multi-material parts – design guidelines for manufacturing of 316l/cucrzr in laser powder bed fusion. *Heliyon*, 9:e18301, 8 2023. ISSN 24058440. doi:10.1016/j.heliyon.2023.e18301.

N. Olhoff, M.P. Bendsøe, and J. Rasmussen. On cad-integrated structural topology and design optimization. *Computer Methods in Applied Mechanics and Engineering*, 89(1):259–279, 1991. ISSN 0045-7825. doi:10.1016/0045-7825(91)90044-7. Second World Congress on Computational Mechanics.

N. Pedersen. Maximization of eigenvalues using topology optimization. *Structural and Multidisciplinary Optimization*, 20:2–11, 8 2000. doi:10.1007/s001580050130.

J.T. Pereira, E.A. Fancello, and C.S. Barcellos. Topology optimization of continuum structures with material failure constraints. *Structural and Multidisciplinary Optimization*, 26:50–66, 1 2004. ISSN 1615-147X. doi:10.1007/s00158-003-0301-z.

X. Qian. Undercut and overhang angle control in topology optimization: A density gradient based integral approach. *International Journal for Numerical Methods in Engineering*, 111:247–272, 7 2017. ISSN 0029-5981. doi:10.1002/nme.5461.

O.M. Querin, G.P. Steven, and Y.M. Xie. Evolutionary structural optimisation (eso) using a bidirectional algorithm. *Engineering Computations*, 15:1031–1048, 12 1998. ISSN 0264-4401. doi:10.1108/02644409810244129.

A. Ramani. A pseudo-sensitivity based discrete-variable approach to structural topology optimization with multiple materials. *Structural and Multidisciplinary Optimization*, 41:913–934, 6 2010. ISSN 1615-147X. doi:10.1007/s00158-009-0455-4.

R. Ranjan, Y. Yang, C. Ayas, M. Langelaar, and F. van Keulen. Controlling local overheating in topology optimization for additive manufacturing. In W. Dewulf, editor, *Proceedings of the Special Interest Group Meeting: Dimensional Accuracy and Surface Finish in Additive Manufacturing*, pages 1–3. EUSPEN, 10 2017. ISBN 978-0995775114.

R. Ranjan, C. Ayas, M. Langelaar, and F. van Keulen. Controlling local overheating in topology optimization for additive manufacturing. *Structural and Multidisciplinary Optimization*, 65:162, 6 2022. ISSN 1615-147X. doi:10.1007/s00158-022-03258-1.

R. Ranjan, Z. Chen, C. Ayas, M. Langelaar, and F. Van Keulen. Overheating control in additive manufacturing using a 3d topology optimization method and experimental validation. *Additive Manufacturing*, 61:103339, 1 2023. ISSN 22148604. doi:10.1016/j.addma.2022.103339.

G. Sabiston and I.Y. Kim. Void region restriction for additive manufacturing via a diffusion physics approach. *International Journal for Numerical Methods in Engineering*, 121:4347–4373, 10 2020. ISSN 0029-5981. doi:10.1002/nme.6434.

M. Schneck, M. Horn, M. Schmitt, C. Seidel, G. Schlick, and G. Reinhart. Review on additive hybrid- and multi-material-manufacturing of metals by powder bed fusion: state of technology and development potential. *Progress in Additive Manufacturing*, 6:881–894, 12 2021. ISSN 2363-9520. doi:0.1007/s40964-021-00205-2.

V. Shah, M. Pamwar, B. Sangha, and I.Y. Kim. Multi-material topology optimization considering natural frequency constraint. *Engineering Computations*, 39:2604–2629, 7 2022. ISSN 0264-4401. doi:10.1108/EC-07-2021-0421.

O. Sigmund. Morphology-based black and white filters for topology optimization. *Structural and Multidisciplinary Optimization*, 33:401–424, 1 2007. doi:10.1007/s00158-006-0087-x.

O. Sigmund. Manufacturing tolerant topology optimization. *Acta Mech Sin*, 25:227–239, 3 2009. doi:10.1007/s10409-009-0240-z.

O. Sigmund and S. Torquato. Design of materials with extreme thermal expansion using a three-phase topology optimization method. *Journal of the Mechanics and Physics of Solids*, 45(6):1037–1067, 1997. ISSN 0022-5096. doi:10.1016/S0022-5096(96)00114-7.

D.D. Singh, T. Mahender, and A.R. Reddy. Powder bed fusion process: A brief review. *Materials Today: Proceedings*, 46:350–355, 2021. ISSN 22147853. doi:10.1016/j.matpr.2020.08.415.

J. Stegmann and E. Lund. Discrete material optimization of general composite shell structures. *International Journal for Numerical Methods in Engineering*, 62(14):2009–2027, 2005. doi:10.1002/nme.1259.

M. Stolpe and K. Svanberg. An alternative interpolation scheme for minimum compliance topology optimization. *Structural and Multidisciplinary Optimization*, 22:116–124, 9 2001. ISSN 1615-147X. doi:10.1007/s001580100129.

K. Svanberg. The method of moving asymptotes—a new method for structural optimization. *International Journal for Numerical Methods in Engineering*, 24:359–373, 2 1987. doi:10.1002/nme.1620240207.

E. van de Ven, R. Maas, C. Ayas, M. Langelaar, and F. van Keulen. Continuous front propagation-based overhang control for topology optimization with additive manufacturing. *Structural and Multidisciplinary Optimization*, 57:2075–2091, 5 2018. ISSN 1615-147X. doi:10.1007/s00158-017-1880-4.

E. van de Ven, R. Maas, C. Ayas, M. Langelaar, and F. van Keulen. Overhang control based on front propagation in 3d topology optimization for additive manufacturing. *Computer Methods in Applied Mechanics and Engineering*, 369:113169, 9 2020. ISSN 00457825. doi:10.1016/j.cma.2020.113169.

E. van de Ven, R. Maas, C. Ayas, M. Langelaar, and F. van Keulen. Overhang control in topology optimization: a comparison of continuous front propagation-based and discrete layer-by-layer overhang control. *Structural and Multidisciplinary Optimization*, 64(2):761–778, June 2021. doi:10.1007/s00158-021-02887-2.

J. van der Zwet, A. Delissen, and M. Langelaar. Prevention of enclosed voids in topology optimization using a cumulative sum flood fill algorithm. *Advances in Engineering Software*, 186:103530, 12 2023. ISSN 09659978. doi:10.1016/j.advengsoft.2023.103530.

N.P. van Dijk, K. Maute, M. Langelaar, and F. van Keulen. Level-set methods for structural topology optimization: a review. *Structural and Multidisciplinary Optimization*, 48:437–472, 9 2013. ISSN 1615-147X. doi:10.1007/s00158-013-0912-y.

F. Wang, B.S. Lazarov, and O. Sigmund. On projection methods, convergence and robust formulations in topology optimization. *Structural and Multidisciplinary Optimization*, 43:767–784, 6 2011. ISSN 1615-147X. doi:10.1007/s00158-010-0602-y.

H.L. Wei, H.K.D.H. Bhadeshia, S.A. David, and T. DebRoy. Harnessing the scientific synergy of welding and additive manufacturing. *Science and Technology of Welding and Joining*, 24(5):361–366, 2019. doi:10.1080/13621718.2019.1615189.

Y. Xiong, S. Yao, Z. Zhao, and Y.M. Xie. A new approach to eliminating enclosed voids in topology optimization for additive manufacturing. *Additive Manufacturing*, 32:101006, 3 2020. ISSN 22148604. doi:10.1016/j.addma.2019.101006.

N. Yaghoobi, M.H. Abolbashari, and B. Hassani. Eliminating enclosed voids in simultaneous optimization of topology and bi-material distribution of three-dimensional structures for additive manufacturing. *Structural and Multidisciplinary Optimization*, 67:137, 8 2024. ISSN 1615-147X. doi:10.1007/s00158-024-03852-5.

I.H. Zainelabdeen, L. Ismail, O.F. Mohamed, K.A. Khan, and A. Schiffer. Recent advancements in hybrid additive manufacturing of similar and dissimilar metals via laser powder bed fusion. *Materials Science and Engineering: A*, 909:146833, 9 2024. ISSN 09215093. doi:10.1016/j.msea.2024.146833.

X. Zheng, C. Williams, C.M. Spadaccini, and K. Shea. Perspectives on multi-material additive manufacturing. *Journal of Materials Research*, 36:3549–3557, 9 2021. ISSN 0884-2914. doi:10.1557/s43578-021-00388-y.

Y. Zhou and K. Saitou. Gradient-based multi-component topology optimization for stamped sheet metal assemblies (mto-s). *Structural and Multidisciplinary Optimization*, 58:83–94, 2018. doi:10.1007/s00158-017-1878-y.

J. Zhu, W. Zhang, and P. Beckers. Integrated layout design of multi-component system. *International Journal for Numerical Methods in Engineering*, 78:631–651, 5 2009. ISSN 0029-5981. doi:10.1002/nme.2499.

W. Zuo and K. Saitou. Multi-material topology optimization using ordered simp interpolation. *Structural and Multidisciplinary Optimization*, 55(2):477–491, 2016. doi:10.1007/s00158-016-1513-3.

A

Sensitivity Equations

First, the total blueprint density of an element is defined as the summation of the individual material densities:

$$\tilde{x}^{(\mathrm{s})}_{j|i} = \tilde{\mu}^{(1)}_{j|i} + \tilde{\mu}^{(2)}_{j|i} \tag{A.1}$$

The blueprint mixture field is given in Eq. (4.7). Next, the derivatives of the printable density fields $\tilde{\xi}^{(m)}_{j|i}$ with respect to the blueprint densities $\tilde{\mu}^{(m)}_{j|i}$ of the same material, and in the same layer, are:

$$\frac{\partial \tilde{\xi}^{(1)}_{j|i}}{\partial \tilde{\mu}^{(1)}_{j|i}} = \frac{1}{2}\left[1 - \frac{\tilde{\Xi}^{(1)}_{j|i}\tilde{\mu}^{(1)}_{j|i}}{\left(\tilde{x}^{(\mathrm{s})}_{j|i}\right)^2} + \frac{\tilde{\Xi}^{(1)}_{j|i}}{\tilde{x}^{(\mathrm{s})}_{j|i}} - \frac{\tilde{\mu}^{(1)}_{j|i} - \tilde{\Xi}^{(1)}_{j|i}\tilde{x}^{(\mathrm{m})}_{j|i}}{\sqrt{\epsilon + \left(\tilde{\mu}^{(1)}_{j|i} - \tilde{\Xi}^{(1)}_{j|i}\tilde{x}^{(\mathrm{m})}_{j|i}\right)^2}}\left(1 + \frac{\tilde{\Xi}^{(1)}_{j|i}\tilde{\mu}^{(1)}_{j|i}}{\left(\tilde{x}^{(\mathrm{s})}_{j|i}\right)^2} - \frac{\tilde{\Xi}^{(1)}_{j|i}}{\tilde{x}^{(\mathrm{s})}_{j|i}}\right)\right] \tag{A.2}$$

$$\frac{\partial \tilde{\xi}^{(2)}_{j|i}}{\partial \tilde{\mu}^{(2)}_{j|i}} = \frac{1}{2}\left[1 + \frac{\tilde{\Xi}^{(2)}_{j|i}\tilde{\mu}^{(1)}_{j|i}}{\left(\tilde{x}^{(\mathrm{s})}_{j|i}\right)^2} - \left(1 - \frac{\tilde{\Xi}^{(2)}_{j|i}\tilde{\mu}^{(1)}_{j|i}}{\left(\tilde{x}^{(\mathrm{s})}_{j|i}\right)^2}\right)\frac{\tilde{\mu}^{(2)}_{j|i} - \tilde{\Xi}^{(2)}_{j|i}\left(1 - \tilde{x}^{(\mathrm{m})}_{j|i}\right)}{\sqrt{\epsilon + \left(\tilde{\mu}^{(2)}_{j|i} - \tilde{\Xi}^{(2)}_{j|i}\left(1 - \tilde{x}^{(\mathrm{m})}_{j|i}\right)\right)^2}}\right] \tag{A.3}$$

The derivatives between the printable densities $\tilde{\xi}^{(m)}_{j|i}$ of one material to the other in the same layer are:

$$\frac{\partial \tilde{\xi}^{(1)}_{j|i}}{\partial \tilde{\mu}^{(2)}_{j|i}} = -\frac{1}{2}\left[\frac{\tilde{\Xi}^{(1)}_{j|i}\tilde{\mu}^{(1)}_{j|i}}{\left(\tilde{x}^{(\mathrm{s})}_{j|i}\right)^2} + \frac{\tilde{\Xi}^{(1)}_{j|i}\tilde{\mu}^{(1)}_{j|i}\left(\tilde{\mu}^{(1)}_{j|i} - \tilde{\Xi}^{(1)}_{j|i}\tilde{x}^{(\mathrm{m})}_{j|i}\right)}{\left(\tilde{x}^{(\mathrm{s})}_{j|i}\right)^2\sqrt{\epsilon + \left(\tilde{\mu}^{(1)}_{j|i} - \tilde{\Xi}^{(1)}_{j|i}\tilde{x}^{(\mathrm{m})}_{j|i}\right)^2}}\right] \tag{A.4}$$

$$\frac{\partial \tilde{\xi}^{(2)}_{j|i}}{\partial \tilde{\mu}^{(1)}_{j|i}} = \frac{1}{2}\left(\frac{\tilde{\mu}^{(1)}_{j|i}}{\left(\tilde{x}^{(\mathrm{s})}_{j|i}\right)^2} - \frac{1}{\tilde{x}^{(\mathrm{s})}_{j|i}}\right)\left(\tilde{\Xi}^{(2)}_{j|i} + \frac{\tilde{\Xi}^{(2)}_{j|i}\left(\tilde{\mu}^{(2)}_{j|i} - \tilde{\Xi}^{(2)}_{j|i}\left(1 - \tilde{x}^{(\mathrm{m})}_{j|i}\right)\right)}{\sqrt{\epsilon + \left(\tilde{\mu}^{(2)}_{j|i} - \tilde{\Xi}^{(2)}_{j|i}\left(1 - \tilde{x}^{(\mathrm{m})}_{j|i}\right)\right)^2}}\right) \tag{A.5}$$

The derivatives of the printable densities $\tilde{\xi}^{(m)}_{j|i}$ to the smooth maximum $\tilde{\Xi}^{(m)}_{j|i}$ are:

$$\frac{\partial \tilde{\xi}^{(1)}_{j|i}}{\partial \tilde{\Xi}^{(1)}_{j|i}} = \frac{1}{2}\left(\tilde{x}^{(\mathrm{m})}_{j|i} + \frac{\tilde{\mu}^{(1)}_{j|i}\left(\tilde{\mu}^{(1)}_{j|i} - \tilde{\Xi}^{(1)}_{j|i}\tilde{x}^{(\mathrm{m})}_{j|i}\right)}{\tilde{x}^{(\mathrm{s})}_{j|i}\sqrt{\epsilon + \left(\tilde{\mu}^{(1)}_{j|i} - \tilde{\Xi}^{(1)}_{j|i}\tilde{x}^{(\mathrm{m})}_{j|i}\right)^2}}\right) \tag{A.6}$$

$$\frac{\partial \tilde{\xi}^{(2)}_{j|i}}{\partial \tilde{\Xi}^{(2)}_{j|i}} = \frac{1}{2}\left[1 - \tilde{x}^{(\mathrm{m})}_{j|i} - \frac{\left(\tilde{\mu}^{(2)}_{j|i} - \tilde{\Xi}^{(2)}_{j|i}\left(1 - \tilde{x}^{(\mathrm{m})}_{j|i}\right)\right)\left(\tilde{x}^{(\mathrm{m})}_{j|i} - 1\right)}{\sqrt{\epsilon + \left(\tilde{\mu}^{(2)}_{j|i} - \tilde{\Xi}^{(2)}_{j|i}\left(1 - \tilde{x}^{(\mathrm{m})}_{j|i}\right)\right)^2}}\right] \tag{A.7}$$

Finally, the derivative of the support field $\tilde{\zeta}_{j|i}$ to the printable density fields $\tilde{\xi}^{(m)}_{j|i}$ always equals one:

$$\frac{\partial \tilde{\zeta}_{j|i}}{\partial \tilde{\xi}^{(m)}_{j|i}} = 1 \tag{A.8}$$

B
Code

B.1. Multi-material overhang filter

Below is a print-out of the Python implementation for the multi-material overhang filter. It is made to work with PyMOTO v1.5.1. Newer versions of PyMOTO have some architectural differences, and will therefore not work.

```python
from pymoto import Module, DomainDefinition
import numpy as np
from scipy.sparse import lil_matrix, csr_matrix

def get_elemnumber(self, layer_index, location, mask, flatten: bool = True):
    """Small function to find the element numbers for an array. This array can be masked."""

    el = [None, None, None]
    el[self.dir_layer] = layer_index  # ind_layer - self.dx_layer
    if mask is not None:
        el[self.dir_orth1] = location[0][mask]
        el[self.dir_orth2] = location[1][mask]
    else:
        el[self.dir_orth1] = location[0]
        el[self.dir_orth2] = location[1]
    els = self.domain.get_elemnumber(*el)
    if flatten:
        return els.flatten()
    else:
        return els

class OverhangFilterMMTO(Module):
    r""" Implementation of overhang filter by Langelaar (2016, 2017). Original implementation by Arnoud Delissen, modifications by Niek van Rossem.

    It proceeds layer by layer through the entire domain. For each element in the current layer, the maximum printable density for both materials is determined by a smooth maximum of the supporting elements :math:‘s_i = \text{smax}(\mathbf{y}_\text{supp})‘. Then, the final printed density values are obtained by a smooth minimum operation of the desired density :math:‘x_i‘ and the maximum printable densities :math:‘x_i‘ as :math:‘y_i = \text{smin}(x_i, s_i)‘.

    Modifications from original filter by Delissen:
    - Added continuation option for ‘‘xi_0‘‘. This is now an input signal rather than a keyword arg.
    - Added custom angle interpolation
```

```
    - Added support for MMTO

    Input Signal:
    - ``x_1``: The unfiltered field of material 1 :math:`\mathbf{x}_1`
    - ``x_2``: The unfiltered field of material 2 :math:`\mathbf{x}_2`
    - ``xi_0``: The density value for which zero overshoot is required. ( ``0 <=
        xi_0 <= 1`` ).

    Output Signal:
    - ``xprint_1``: Filtered field of material 1 :math:`\mathbf{x}_\mathrm{print
        ,1}`, without overhangs
    - ``xprint_2``: Filtered field of material 2 :math:`\mathbf{x}_\mathrm{print
        ,2}`, without overhangs

    Args:
        domain: The domain layout

    Keyword Args:
        direction: Print direction as array or string, e.g. ``[0, -1]`` (in 2D) or
             ``"y-"`` for negative y direction. Currently, only directions aligned
            with one of the Cartesian axes are supported. Default is ``[0, 1, 0]``
        alpha1: Maximum overhang angle of material 1 in degrees. A value of ``0``
            supports floating features, while an angle of ``90`` allows only
            vertical columns. Default is ``45.0``
        alpha2: Maximum overhang angle of material 2 in degrees. A value of ``0``
            supports floating features, while an angle of ``90``  allows only
            vertical columns. Default is ``30.0``
        p: Exponent of the smooth maximum function ( ``p > 0`` ). Higher p
            increases accuracy, but reduces smoothness. Default is ``40.0``
        eps: Smooth minimum regularization parameter ( ``eps >= 0`` ). Lower eps
            increases accuracy, but reduces smoothness. Default is ``1e-4``
    """

    def _prepare(
        self,
        domain: DomainDefinition,
        direction=(0.0, 1.0, 0.0),
        alpha1: float = 45,
        alpha2: float = 30,
        p: float = 40,
        eps: float = 1e-4,
        n_int: int = 12
    ):

        assert 0.0 <= alpha1 <= 90.0, "Choose an alpha1 between 0 and 90"
        assert 0.0 <= alpha2 <= 90.0, "Choose an alpha2 between 0 and 90"

        # Parse print direction
        if isinstance(direction, str):
            # Print axis
            axes = np.argwhere([a in direction.lower() for a in ['x', 'y', 'z']]).
                flatten()
            if axes.size != 1:
                raise ValueError(f"Wrongly specified print direction {direction},
                    should be e.g. \"+x\", \"-y\"")

            # Print direction
            direction = [0.0, 0.0, 0.0]
            direction[axes[0]] = -1.0 if '-' in direction else +1.0
        direction = np.asarray(direction, dtype=np.float64).flatten()
        if direction.size < 3:
```

```
            direction = np.pad(direction, (0, 3 - direction.size), 'constant', constant_values=0.0)
        elif direction.size > 3:
            direction = direction[:3]

        self.direction = direction / np.linalg.norm(direction)
        self.domain = domain
        if self.domain.dim == 2:
            assert self.direction[2] == 0.0, "Z-direction must be zero for 2-dimensional domain"
        assert abs(self.direction).sum() >= 1.0 - 1e-10, "The print direction must be aligned with either x, y or z directions"

        # Size of the domain and its elements
        self.size = [domain.nelx, domain.nely, max(domain.nelz, 1)]
        elsize = [domain.unitx, domain.unity, domain.unitz]
        domain_size = [self.size[i] * elsize[i] for i, _ in enumerate(np.arange(3))]

        dir_layer = int(np.argmax(abs(direction)))  # The axis of the print direction
        self.dx_layer = int(np.sign(direction[dir_layer]))  # Iteration direction

        dir_orth1 = (dir_layer + 1) % 3
        dir_orth2 = (dir_layer + 2) % 3
        if dir_orth1 == 2 and domain.dim == 2:
            dir_orth1, dir_orth2 = dir_orth2, dir_orth1  # Make sure the z-direction is last for 2D

        # cone radius
        radius = 2 * [None]
        radius[0] = elsize[dir_layer] * np.tan(np.pi/2 - np.deg2rad(alpha1))
        radius[1] = elsize[dir_layer] * np.tan(np.pi/2 - np.deg2rad(alpha2))

        # only 2 sampling points should be used for 2D cases
        if self.domain.dim == 2 and n_int != 2:
            print(f"For 2D problems, only 2 interpolation points should be used. ‘n_int‘ changed from {n_int} to 2")
            n_int = 2

        # find interpolation point offsets from centroid
        theta = np.linspace(0, 2 * np.pi * (1 - 1 / n_int), n_int)
        inter_offset1 = n_int * [None]
        inter_offset2 = n_int * [None]
        for i, angle in enumerate(theta):
            xOffset1 = radius[0] * np.cos(angle)
            yOffset1 = radius[0] * np.sin(angle)
            xOffset2 = radius[1] * np.cos(angle)
            yOffset2 = radius[1] * np.sin(angle)
            inter_offset1[i] = [xOffset1, yOffset1]
            inter_offset2[i] = [xOffset2, yOffset2]

        # dummy layer with element locations
        X, Y = np.meshgrid(range(self.size[dir_orth1]), range(self.size[dir_orth2]), indexing='ij')
        entire_layer = [X.flatten(), Y.flatten()]
        nel_layer = int(self.size[dir_orth1] * self.size[dir_orth2])

        # list of all centroids in a layer
        centroid = np.meshgrid(elsize[dir_orth1] * (1 / 2 + np.arange(self.size[dir_orth1])), elsize[dir_orth2] * (1 / 2 + np.arange(self.size[
```

```
            dir_orth2])))

        # matrices with interpolation weights
        H1 = n_int * [None]
        H2 = n_int * [None]
        for i, point in enumerate(np.arange(n_int)):

            # find global location of interpolation point
            inter_global1 = 2 * [None]
            inter_global2 = 2 * [None]
            inter_global1[0] = centroid[0] + inter_offset1[i][0]
            inter_global1[1] = centroid[1] + inter_offset1[i][1]
            inter_global2[0] = centroid[0] + inter_offset2[i][0]
            inter_global2[1] = centroid[1] + inter_offset2[i][1]
            inter_global1 = np.asarray(inter_global1)
            inter_global2 = np.asarray(inter_global2)

            # initialize weights matrix
            ihj1 = lil_matrix((nel_layer, nel_layer), dtype='float')
            ihj2 = lil_matrix((nel_layer, nel_layer), dtype='float')

            # loop over the elements (material 1)
            for j, val in enumerate(zip(inter_global1[0].reshape((-1, 1)),
                inter_global1[1].reshape((-1, 1)))):

                # skip all sampling points not inside the domain
                if 0 < val[0] > domain_size[dir_orth1] or 0 < val[1] > domain_size
                    [dir_orth2]:
                    continue
                else:
                    # find distances of each element w.r.t. the current
                        interpolation point
                    dist = 2 * [None]
                    dist[0] = np.abs(val[0] - centroid[0].reshape((-1, 1)))
                    dist[1] = np.abs(val[1] - centroid[1].reshape((-1, 1)))

                    # filter out the irrelevant elements
                    idx = (np.where(dist[0] < elsize[dir_orth1], True, False)
                    * np.where(dist[1] < elsize[dir_orth2], True, False)).flatten
                        ()
                    dist[0], dist[1] = dist[0][idx], dist[1][idx]

                    # find associated weights and add to matrices
                    wgt = (1.0 - dist[0] / elsize[dir_orth1]) * (1.0 - dist[1] /
                        elsize[dir_orth2])
                    ihj1[j, idx] = wgt.flatten()

            # loop over the elements (material 2) # TODO: clean this up
            for j, val in enumerate(zip(inter_global2[0].reshape((-1, 1)),
                inter_global2[1].reshape((-1, 1)))):

                # skip all sampling points not inside the domain
                if 0 < val[0] > domain_size[dir_orth1] or 0 < val[1] > domain_size
                    [dir_orth2]:
                    continue
                else:
                    # find distances of each element w.r.t. the current
                        interpolation point
                    dist = 2 * [None]
                    dist[0] = np.abs(val[0] - centroid[0].reshape((-1, 1)))
                    dist[1] = np.abs(val[1] - centroid[1].reshape((-1, 1)))
```

```python
                    # filter out the irrelevant elements
                    idx = (np.where(dist[0] < elsize[dir_orth1], True, False)
                    * np.where(dist[1] < elsize[dir_orth2], True, False)).flatten()
                    dist[0], dist[1] = dist[0][idx], dist[1][idx]

                    # find associated weights and add to matrices
                    wgt = (1.0 - dist[0] / elsize[dir_orth1]) * (1.0 - dist[1] / elsize[dir_orth2])
                    ihj2[j, idx] = wgt.flatten()

            # store the matrix in the list
            H1[i] = ihj1.tocsr()  # TODO: check which method is fastest
            H2[i] = ihj2.tocsr()

        # number of centroids that can fit inside the circle
        nel_offset1 = np.floor(radius[0] / [elsize[dir_orth1], elsize[dir_orth2]]).astype(int)
        nel_offset1 = np.array([np.arange(-nel_offset1[0], nel_offset1[0] + 1),
                                np.arange(-nel_offset1[1], nel_offset1[1] + 1)])
        nel_offset2 = np.floor(radius[1] / [elsize[dir_orth1], elsize[dir_orth2]]).astype(int)
        nel_offset2 = np.array([np.arange(-nel_offset2[0], nel_offset2[0] + 1),
                                np.arange(-nel_offset2[1], nel_offset2[1] + 1)])

        # rectangular block of centroids
        block_offsets1 = np.array([np.repeat(nel_offset1[0], nel_offset1[1].size),
                                   np.tile(nel_offset1[1], nel_offset1[0].size)]).T.tolist()
        block_offsets2 = np.array([np.repeat(nel_offset2[0], nel_offset2[1].size),
                                   np.tile(nel_offset2[1], nel_offset2[0].size)]).T.tolist()

        # create mask for all elements outside the cone
        nel_block1 = np.array(block_offsets1)[:, 0].size
        mask1 = nel_block1 * [None]
        for i, offset in enumerate(block_offsets1):
            dist = np.linalg.norm(np.asarray([elsize[dir_orth1], elsize[dir_orth2]]) * offset)
            if self.domain.dim == 2:
                mask1[i] = True if dist <= radius[0] and offset[1] == 0 else False
            else:
                mask1[i] = True if dist <= radius[0] else False
        nel_block2 = np.array(block_offsets2)[:, 0].size
        mask2 = nel_block2 * [None]
        for i, offset in enumerate(block_offsets2):
            dist = np.linalg.norm(np.asarray([elsize[dir_orth1], elsize[dir_orth2]]) * offset)
            if self.domain.dim == 2:
                mask2[i] = True if dist <= radius[1] and offset[1] == 0 else False
            else:
                mask2[i] = True if dist <= radius[1] else False

        # apply the mask to the offset list to remove the centroids outside the cone
        if alpha1 < alpha2:
            self.layer_offsets = np.array(block_offsets1)[mask1].tolist()
        elif alpha1 >= alpha2:
            self.layer_offsets = np.array(block_offsets2)[mask2].tolist()
        self.nsampling = np.array([np.array(block_offsets1)[mask1][:, 0].size,
```

```
        np.array(block_offsets2)[mask2][:, 0].size], dtype='int')

        # create masks since not all elements may be inside the domain
        support_idx = [None for _ in range(np.max(self.nsampling))]
        offset_masks = [None for _ in range(np.max(self.nsampling))]
        for i, offset in enumerate(self.layer_offsets):
            support_idx[i] = (entire_layer[0] + offset[0],
                              entire_layer[1] + offset[1])
            offset_masks[i] = np.logical_and((support_idx[i][0] >= 0) * (
                support_idx[i][0] < self.size[dir_orth1]),
                                             (support_idx[i][1] >= 0) * (
                                                 support_idx[i][1] < self.size[
                                                 dir_orth2]))

        # create mask for support region
        mask = 2 * [None]
        rem = [np.max([0, 0.5 * (self.nsampling[1] - self.nsampling[0])]),
               np.max([0, 0.5 * (self.nsampling[0] - self.nsampling[1])])]
        mask[0] = int(rem[0]) * [False] + self.nsampling[0] * [True] + int(rem[0])
            * [False]
        mask[1] = int(rem[1]) * [False] + self.nsampling[1] * [True] + int(rem[1])
            * [False]

        # store parameters
        self.p = p
        self.H1 = H1
        self.H2 = H2
        self.eps = eps
        self.mask = mask
        self.n_int = n_int
        self.q, self.shift, self.backshift, self.smax = None, None, None, None
        self.dir_layer = dir_layer
        self.dir_orth1 = dir_orth1
        self.dir_orth2 = dir_orth2
        self.entire_layer = entire_layer
        self.support_idx = support_idx
        self.offset_masks = offset_masks

    def _response(self, x_mat1, x_mat2, xi_0):

        assert x_mat1.dtype == x_mat2.dtype, "both input fields must have same
            dtype!"

        # shift and backshift values
        dbl_min = np.finfo(x_mat1.dtype).tiny
        self.shift = 100.0 * pow(dbl_min, 1.0 / self.p)  # Small shift to prevent
            division by 0
        self.q = self.p + np.log(1.0 * self.nsampling) / np.log(xi_0)
        self.backshift = pow(self.nsampling, 1 / self.q) * pow(self.shift, self.p
            / self.q) * 0.95

        # convert material fields to RMMI fields
        self.x_den = x_mat1 + x_mat2
        self.x_mix = np.divide(x_mat1, self.x_den + dbl_min)

        xprint1, xprint2 = x_mat1.copy(), x_mat2.copy()
        self.xcorr = np.zeros_like(x_mat1, dtype=x_mat1.dtype)
        self.smax1 = np.zeros_like(x_mat1, dtype=x_mat1.dtype)
        self.smax2 = np.zeros_like(x_mat2, dtype=x_mat2.dtype)
        self.sum1 = np.zeros_like(x_mat1, dtype=x_mat1.dtype)
        self.sum2 = np.zeros_like(x_mat2, dtype=x_mat2.dtype)
```

```

        # layer index
        ind_layer = 1 if self.dx_layer >= 0 else self.size[self.dir_layer] - 2  # Starting index

        # Loop over all the layers
        while 0 <= ind_layer < self.size[self.dir_layer]:

            # allocate space for smooth max
            keep1 = np.zeros_like(self.entire_layer[0], dtype=x_mat1.dtype)
            keep2 = np.zeros_like(self.entire_layer[0], dtype=x_mat2.dtype)

            # create support field
            els = get_elemnumber(self, ind_layer, self.entire_layer, None, True)

            # loop over the interior elements
            for i, (supp_idx, supp_mask) in enumerate(zip(self.support_idx, self.offset_masks)):
                els = get_elemnumber(self, ind_layer - self.dx_layer, supp_idx, supp_mask, True)
                self.xcorr[els] = xprint1[els] + xprint2[els]
                if self.mask[0][i]:
                    keep1[supp_mask] += np.power(self.xcorr[els] + self.shift, self.p)
                if self.mask[1][i]:
                    keep2[supp_mask] += np.power(self.xcorr[els] + self.shift, self.p)

            # loop over the interpolation points
            els = get_elemnumber(self, ind_layer - self.dx_layer, self.entire_layer, None, True)
            for i, val in enumerate(np.arange(self.n_int)):
                xcorr_interp1 = self.H1[i] * self.xcorr[els]
                xcorr_interp2 = self.H2[i] * self.xcorr[els]
                keep1 += np.power(xcorr_interp1 + self.shift, self.p)
                keep2 += np.power(xcorr_interp2 + self.shift, self.p)

            # smooth maximum (Xi)
            max_supp1 = np.power(keep1, 1.0 / self.q[0]) - self.backshift[0]
            max_supp2 = np.power(keep2, 1.0 / self.q[1]) - self.backshift[1]

            # elements in current layer
            els = get_elemnumber(self, ind_layer, self.entire_layer, None, False)

            # store these for the sensitivity
            self.sum1[els] = keep1  # sum(xcorr^P)
            self.sum2[els] = keep2
            self.smax1[els] = max_supp1  # sum(xcorr^P)^(1/Q) - s2
            self.smax2[els] = max_supp2

            # smooth minimum
            r1 = x_mat1[els] - np.multiply(self.x_mix[els], max_supp1)
            r2 = x_mat2[els] - np.multiply(1.0 - self.x_mix[els], max_supp2)
            xprint1[els] = (x_mat1[els] + np.multiply(self.x_mix[els], max_supp1) - np.sqrt(np.power(r1, 2) + self.eps) + np.sqrt(self.eps)) / 2
            xprint2[els] = (x_mat2[els] + np.multiply(1.0 - self.x_mix[els], max_supp2) - np.sqrt(np.power(r2, 2) + self.eps) + np.sqrt(self.eps)) / 2

            # update layer
            ind_layer += self.dx_layer
```

```python

        return xprint1, xprint2

    def _sensitivity(self, df_dxprint1, df_dxprint2):

        # input and output signals
        x_mat1 = self.sig_in[0].state
        x_mat2 = self.sig_in[1].state
        df_dxmat1 = np.zeros_like(df_dxprint1, dtype=df_dxprint1.dtype)
        df_dxmat2 = np.zeros_like(df_dxprint2, dtype=df_dxprint2.dtype)

        # copy of input for modification (first few terms will be added to this)
        df_dxcorr = np.zeros_like(df_dxprint1, dtype=df_dxprint1.dtype)

        # current layer index
        ind_layer = self.size[self.dir_layer] - 1 if self.dx_layer >= 0 else 0  # Starting index (="ending" in response)

        # Loop over all the layers, starting at the top
        while True:

            # get element numbers of elements in current layer
            els = get_elemnumber(self, ind_layer, self.entire_layer, None, False)

            # dxprint1_dxmat1
            term1 = - 0.5 * self.smax1[els] * x_mat1[els] / self.x_den[els]**2
            term2 = 0.5 * self.smax1[els] / (self.x_den[els])
            term3 = - 0.25 * (x_mat1[els] - self.smax1[els] * self.x_mix[els]) * (2 * self.smax1[els] * x_mat1[els] / (self.x_den[els]**2) - 2 * self.smax1[els] / self.x_den[els] + 2)
            term4 = np.sqrt(self.eps + (x_mat1[els] - self.smax1[els] * self.x_mix[els])**2)
            dxprint1_dxmat1 = term1 + term2 + term3 / term4 + 0.5

            # dxprint1_dxmat2
            term1 = - 0.5 * self.smax1[els] * x_mat1[els] / self.x_den[els]**2
            term2 = - 0.5 * self.smax1[els] * x_mat1[els] * (x_mat1[els] - self.smax1[els] * self.x_mix[els])
            term3 = np.sqrt(self.eps + (x_mat1[els] - self.smax1[els] * self.x_mix[els])**2) * (self.x_den[els]**2)
            dxprint1_dxmat2 = term1 + term2 / term3

            # dxprint2_dxmat1
            term1 = 0.5 * self.smax2[els] * (x_mat1[els] / (self.x_den[els]**2) - 1 / self.x_den[els])
            term2 = 0.5 * self.smax2[els] * (x_mat2[els] - self.smax2[els] * (1 - self.x_mix[els])) * (x_mat1[els] / (self.x_den[els]**2) - 1 / self.x_den[els])
            term3 = np.sqrt(self.eps + (x_mat2[els] - self.smax2[els] * (1 - self.x_mix[els]))**2)
            dxprint2_dxmat1 = term1 + term2 / term3

            # dxprint2_dxmat2
            term1 = 0.5 * self.smax2[els] * x_mat1[els] / (self.x_den[els]**2)
            term2 = - 0.25 * (x_mat2[els] - self.smax2[els] * (1 - self.x_mix[els])) * (2 - 2 * self.smax2[els] * x_mat1[els] / (self.x_den[els]**2))
            term3 = (self.eps + (x_mat2[els] - self.smax2[els] * (1 - self.x_mix[els]))**2)**0.5
            dxprint2_dxmat2 = term1 + term2 / term3 + 0.5

```

```
        # output arrays
        df_dxmat1[els] = df_dxprint1[els] * dxprint1_dxmat1 + df_dxprint2[els] * dxprint2_dxmat1 + df_dxcorr[els] * (dxprint1_dxmat1 + dxprint2_dxmat1)
        df_dxmat2[els] = df_dxprint1[els] * dxprint1_dxmat2 + df_dxprint2[els] * dxprint2_dxmat2 + df_dxcorr[els] * (dxprint1_dxmat2 + dxprint2_dxmat2)

        ## START OF PREPARATIONS FOR LAYER BELOW

        ## first term

        # dxprint1_dsmax1
        term1 = 0.5 * self.x_mix[els]
        term2 = 0.5 * x_mat1[els] * (x_mat1[els] - self.smax1[els] * self.x_mix[els])
        term3 = np.sqrt(self.eps + (x_mat1[els] - self.smax1[els] * self.x_mix[els])**2) * (self.x_den[els])
        dxprint1_dsmax1 = term1 + term2 / term3

        # dxprint2_dsmax2
        term1 = - 0.5 * self.x_mix[els]
        term2 = - 0.25 * (x_mat2[els] - self.smax2[els] * (1 - self.x_mix[els])) * (2 * self.x_mix[els] - 2)
        term3 = (self.eps + (x_mat2[els] - self.smax2[els] * (1 - self.x_mix[els]))**2)**0.5
        dxprint2_dsmax2 = term1 + term2 / term3 + 0.5

        # (input + recursive part) * first term
        df_dsmax1 = (df_dxprint1[els] + df_dxcorr[els]) * dxprint1_dsmax1
        df_dsmax2 = (df_dxprint2[els] + df_dxcorr[els]) * dxprint2_dsmax2

        # first half of second term
        dsmax1_dkeep1 = self.p / self.q[0] * np.power(self.sum1[els], 1 / self.q[0] - 1)
        dsmax2_dkeep2 = self.p / self.q[1] * np.power(self.sum2[els], 1 / self.q[1] - 1)

        # input * first term * first half of second term
        df_dkeep1 = df_dsmax1 * dsmax1_dkeep1
        df_dkeep2 = df_dsmax2 * dsmax2_dkeep2

        # loop over the interior points
        for i, (supp_idx, supp_mask) in enumerate(zip(self.support_idx, self.offset_masks)):
            els = get_elemnumber(self, ind_layer - self.dx_layer, supp_idx, supp_mask, False)
            if self.mask[0][i]:
                df_dxcorr[els] += df_dkeep1[supp_mask] * np.power(self.xcorr[els] + self.shift, self.p - 1)
            if self.mask[1][i]:
                df_dxcorr[els] += df_dkeep2[supp_mask] * np.power(self.xcorr[els] + self.shift, self.p - 1)

        # loop over the interpolation points
        els = get_elemnumber(self, ind_layer - self.dx_layer, self.entire_layer, None, False)
        for i, val in enumerate(np.arange(self.n_int)):
            xcorr_interp1 = self.H1[i] * self.xcorr[els]
            xcorr_interp2 = self.H2[i] * self.xcorr[els]
            xcorr_raised1 = np.diag(np.power(xcorr_interp1 + self.shift, self.
```

```
                    p - 1))
                xcorr_raised2 = np.diag(np.power(xcorr_interp2 + self.shift, self.
                    p - 1))
                df_dxcorr[els] += (np.matvec(self.H1[i].T * xcorr_raised1,
                    df_dkeep1)
                                + np.matvec(self.H2[i].T * xcorr_raised2,
                                    df_dkeep2))

            # update layer
            ind_layer -= self.dx_layer

            # end if we’ve reached the first layer
            if not 1 <= ind_layer < self.size[self.dir_layer] - 1:
                break

        # base layer is directly transferred
        els = get_elemnumber(self, ind_layer, self.entire_layer, None, False)
        df_dxmat1[els] = df_dxprint1[els] + df_dxcorr[els]
        df_dxmat2[els] = df_dxprint2[els] + df_dxcorr[els]

        return df_dxmat1, df_dxmat2, 0
```

B.2. Hotspot detection method

Below you can find a print-out of the hotspot detection block. It is made to work with PyMOTO v1.5.1. Newer versions of PyMOTO have some architectural differences, and will therefore not work. The block takes the density field as an input, and outputs the pseudo-temperature field. The p-norm operation, multi-material correction, and constraint equation have to be implemented manually, and this can be done with the `MathGeneral` blocks already present in PyMOTO.

```
from pymoto import Module, DomainDefinition
import numpy as np
from typing import Union
from scipy.sparse import lil_matrix, csr_matrix
from scipy.signal import convolve, correlate
from numbers import Number

# regular hotspot detection
class HotspotDetection(Module):
    r""" Implementation of geometric hotspot detection method by Das (2025). This
        is intended to be a more physics-based alternative to the overhang
        constraint.

    For every element in the design domain it estimates the conductivity based on
        the surrounding geometry and material and process parameters. From this
        conductivity a normalised temperature field is then calculated via ‘
        temperature = 1 - conductivity‘.

    To use this in a constraint, calculate the maximum temperature of the field
        via the p-norm method, and add this as a constraint.

    Code architecture based on the OverhangFilter module by Arnoud Delissen.

    Input Signal:
    - ‘‘x‘‘: The input material field

    Output Signal:
    - ‘‘tau‘‘: The normalised temperature field

    Args:
        domain: The domain layout
```

```

    Keyword Args:
        direction: Print direction as array or string, e.g. ‘‘[0, -1]‘‘ (in 2D) or ‘‘"y-"‘‘ for negative y direction. Currently, only directions aligned with one of the Cartesian axes are supported. Default is ‘‘[0, 1, 0]‘‘
        Cp: The specific heat capacity of the material. No default is specified, thus this must be implemented manually
        h: The hatch spacing of the build process. Increasing the hatch spacing will lower the thermal interaction length. Default is ‘‘0.01‘‘
        k: Thermal conductivity of the material
        q: Exponent of the conductivity penalization for intermediate densities. Default is ‘‘3‘‘
        v: Scanning speed of the build process. Increasing this value will decrease the thermal interaction length. Default is ‘‘0.05‘‘
        rho: Material density. No default is specified.
        R: [OPTIONAL] Use this to manually set the conductivity filter radius. Normally the radius is calculated via the material properties and process parameters. Using R will overwrite the radius calculated. Try to keep this value between 1 and 20 if you choose to use this

    References:
        - Das, M.N.S. et al. (2025) *A physics-motivated geometric method for overheating prevention in topology optimization for additive manufacturing*. Computer Methods in Applied Mechanics and Engineering, Volume 447. ‘doi: 10.1016/j.cma.2025.118363 <https://doi.org/10.1016/j.cma.2025.118363>‘
    """

    def _prepare(
        self,
        domain: DomainDefinition,
        direction=(0.0, 1.0, 0.0),
        Cp: float = None,
        h: float = 0.1,
        k: float = None,
        q: float = 3.0,
        v: float = 300,
        rho: float = None,
        R: float = None
    ):

        # copy element number function
        def get_elemnumber2(nelx, nely, eli: Union[int, np.ndarray], elj: Union[int, np.ndarray], elk: Union[int, np.ndarray]):

            temp = (elk * nely + elj) * nelx + eli

            if np.size(temp) == 1:
                result = int(temp)
            elif np.size(temp) > 1:
                result = [int(item) for item in temp]
            else:
                raise Exception("length of element location arrays must be greater than 0!")

            return result

        # creates arrays with element locations
        def get_elemarrays(nelx, nely, nelz):
            elx = np.repeat(np.arange(nelx), nely * max(nelz, 1))
            ely = np.tile(np.repeat(np.arange(nely), max(nelz, 1)), nelx)
```

```
            elz = np.tile(np.arange(max(nelz, 1)), (nelx) * nely)
            return elx, ely, elz

        # Parse print direction
        if isinstance(direction, str):

            # Print axis
            axes = np.argwhere([a in direction.lower() for a in ['x', 'y', 'z']]).flatten()
            if axes.size != 1:
                raise ValueError(f"Wrongly specified print direction {direction}, should be e.g. \"+x\", \"-y\"")

            # Print direction
            direction = [0.0, 0.0, 0.0]
            direction[axes[0]] = -1.0 if '-' in direction else +1.0

        direction = np.asarray(direction, dtype=np.float64).flatten()
        if direction.size < 3:
            direction = np.pad(direction, (0, 3-direction.size), 'constant', constant_values=0.0)
        elif direction.size > 3:
            direction = direction[:3]

        direction = direction / np.linalg.norm(direction)
        if domain.dim == 2:
            assert direction[2] == 0.0, "Z-direction must be zero for 2-dimensional domain"
        assert abs(direction).sum() >= 1.0 - 1e-10, "The print direction must be aligned with either x, y or z directions"

        # Size of the domain
        size = [domain.nelx, domain.nely, max(domain.nelz, 1)]

        # find indices of build direction and orthogonal directions
        dir_layer = int(np.argmax(abs(direction)))  # The axis of the print direction
        dir_orth1 = (dir_layer + 1) % 3
        dir_orth2 = (dir_layer + 2) % 3
        if dir_orth1 == 2 and domain.dim == 2:
            dir_orth1, dir_orth2 = dir_orth2, dir_orth1  # Make sure the z-direction is last for 2D

        # iteration direction
        it_layer = int(np.sign(direction[dir_layer]))

        # find thermal interaction length
        if R == None:
            assert rho != None, "Specify a material density or conductivity filter radius"
            assert Cp != None, "Specify a material heat capacity or conductivity filter radius"
            assert k != None, "Specify a material conductivity or conductivity filter radius"

            # calculate layer area
            if domain.dim == 2:
                A = domain.element_size[dir_orth1] * size[dir_orth1]
            else:
                A = domain.element_size[dir_orth1] * size[dir_orth1] * domain.element_size[dir_orth2] * size[dir_orth2]
```

```

            alpha = k/(rho*Cp)
            th = A/(h*v)
            r = np.sqrt(alpha*th)

            # number of layers to be added as build plate
            R = int(r / domain.element_size[dir_layer])
            assert R > 1, "The thermal interaction length is only 1 element or
                less!"
            assert R < 20, "The thermal interaction length is longer than 20
                elements!"
        else:
            r = R*domain.element_size[dir_layer]

        elx, ely, elz = get_elemarrays(domain.nelx, domain.nely, domain.nelz)

        # reorder locations to be consistent with element numbering
        order = get_elemnumber2(domain.nelx, domain.nely, elx, ely, elz)
        elx[order] = elx
        ely[order] = ely
        elz[order] = elz

        # initialise weights matrix
        iwj = lil_matrix((domain.nel, domain.nel), dtype='float')

        # loop over all the elements
        for i, temp1 in enumerate(np.arange(domain.nel)):

            # find distances
            dist = [None, None, None]
            dist[0] = domain.unitx * np.array(elx[i] - elx)
            dist[1] = domain.unity * np.array(ely[i] - ely)
            dist[2] = domain.unitz * np.array(elz[i] - elz)
            D = np.sqrt(np.power(dist[0], 2) + np.power(dist[1], 2) + np.power(
                dist[2], 2))

            # indices of elements within conductivity domain
            idx = np.where(D <= r, True, False)

            # remove all data of irrelevant elements
            dist[0] = dist[0][idx]
            dist[1] = dist[1][idx]
            dist[2] = dist[2][idx]
            D = D[idx]

            # create mask
            mask = np.where(it_layer * dist[dir_layer] >= 0.0, 1.0, 0.0)

            lat = np.sqrt(np.power(dist[dir_orth1], 2) + np.power(dist[dir_orth2],
                 2))
            wa = np.multiply(mask, abs(np.arctan2(dist[dir_layer], lat)))
            iwj[i, idx] = np.multiply(wa, (r - D))

        # final weights matrix (CSR is best for the operations performed in
            _response and _sensitivity)
        self.iwj = iwj.tocsr() + csr_matrix(np.eye(domain.nel))
        self.jwi = self.iwj.transpose()

        # Parameters used later
        self.q = q
        self.elx = elx
```

```
        self.domain = domain

    def _response(self, x):

        # create copy of input
        xcopy = x.copy().reshape((-1, 1))

        # calculate conductivity domain
        x_q = np.power(xcopy, self.q)
        keep1 = self.iwj*x_q
        keep2 = self.iwj*np.ones_like(xcopy)
        mu    = np.divide(keep1, keep2)
        tau   = np.ones_like(mu) - mu

        return tau.flatten()

    def _sensitivity(self, dx):

        # create copy of input and expanded field
        x = self.sig_in[0].state.reshape((-1,1))

        # copy input signal
        dxcopy = np.reshape(dx, (-1, 1))

        # calculate sensitivity
        keep1 = -self.q*np.multiply(self.jwi.toarray(), np.power(x, self.q-1)) #
            find a multiply function that works with sparse arrays
        keep2 = np.multiply(np.ones((self.domain.nel, self.domain.nel)), np.array(
            self.iwj.sum(axis=1))).transpose()
        dtau_dx = np.divide(keep1, keep2)
        dtau = np.matvec(dtau_dx, dxcopy.transpose()).transpose()

        return dtau.flatten()
```