

Less Stress for Glass Giants:

A stress-based topology optimization workflow for cast glass

E. Macedo Dauzacker

4688994

TU Delft

MSc Thesis P4

Report P5

15/06/2025

Mentors:

Dr. Charalampos Andriotis

Dr. Faidra Oikonomopoulou

Abstract

The muse of this project is Glass, the Brittle Beauty. Computational tools are adapted and assembled into an optimization workflow to inform the structural design of cast glass. The objective function is the weighted-sum of three components: 1) area ratio, and the p-norm of both 2) stress and 3) displacement. The Drucker-Prager stress criteria for brittle materials is used to calculate the stress component. There is a soft constraint on annealing time and even thickness embedded in the material distribution definition, which also has a considerable reduction in the number of parameters and post-processing effort needed when compared to the SIMP methodology. This material distribution definition relies on double slicing Gaussian Random Fields with a transformed sigmoid curve. The optimization algorithm is written in python. It makes use of pytorch's automatic differentiation functionality to calculate the gradients, and its Stochastic Gradient Descent (SGD) and Adam optimizers to update the material distribution parameters. It culminates in the design of an indoor pedestrian bridge with significantly lower tensile stresses than comparable projects have achieved, with appreciably lower computing power used.

Contents

| | | |
|-------|---|----|
| 1 | Introduction..... | 6 |
| 1.1 | Research..... | 12 |
| 1.1.1 | Problem statement..... | 12 |
| 1.1.2 | Research goal | 12 |
| 1.1.3 | Research question..... | 12 |
| 1.1.4 | Sub questions | 13 |
| 1.1.5 | Methodology..... | 14 |
| 2 | Glass..... | 15 |
| 2.1 | Material properties..... | 15 |
| 2.1.1 | Brittleness | 15 |
| 2.1.2 | Isotropic..... | 16 |
| 2.1.3 | Young's Modulus | 16 |
| 2.1.4 | Poisson's ratio..... | 16 |
| 2.2 | Manufacturing constraints | 16 |
| 2.2.1 | Mold..... | 16 |
| 2.2.2 | Cooling time, maximum thickness..... | 16 |
| 2.2.3 | Even thickness..... | 18 |
| 2.2.4 | Sharp edges | 18 |
| 2.3 | Structural constraints | 19 |
| 2.3.1 | Deflection | 19 |
| 2.3.2 | Safety factors | 19 |
| 2.4 | Glass composition and support condition | 19 |
| 2.5 | Quantified properties and constraints..... | 20 |
| 3 | Optimization process..... | 21 |
| 3.1 | Type: size, shape, topology..... | 22 |
| 3.2 | Objective: compliance, volume, stress | 23 |
| 3.3 | Gradient calculation..... | 24 |
| 3.4 | Optimizer step | 24 |
| 4 | Optimization methods | 27 |
| 4.1 | SIMP..... | 27 |
| 4.2 | Indirect definition of pseudo-densities | 28 |
| 4.3 | Discrete topology..... | 31 |
| 4.4 | Level-Set | 32 |
| 4.4.1 | Function parameterization | 34 |
| 4.4.2 | Geometry mapping | 36 |

| | | |
|-------|--|----|
| 4.4.3 | Update information and process | 38 |
| 4.4.4 | Applied regularization..... | 39 |
| 4.5 | (Bidirectional) Evolutionary Structural Optimization | 39 |
| 5 | Structural calculations | 40 |
| 5.1 | Finite Element Analysis..... | 40 |
| 5.2 | Fine tuning the structural model..... | 41 |
| 6 | Stress | 42 |
| 6.1 | Why is stress important in the specific case of structural glass elements? | 42 |
| 6.2 | Stress criteria | 42 |
| 6.2.1 | Rankine | 43 |
| 6.2.2 | Von Mises..... | 43 |
| 6.2.3 | (Modified) Coulomb-Mohr..... | 43 |
| 6.2.4 | Drucker-Prager criteria | 43 |
| 6.3 | Challenges and possible solutions | 44 |
| 6.3.1 | Singularity..... | 44 |
| 6.3.2 | Non-linearity | 45 |
| 6.3.3 | Local nature | 45 |
| 7 | Case Study | 46 |
| 7.1 | Context..... | 46 |
| 7.2 | Design domain | 47 |
| 7.3 | Structural problem definition | 48 |
| 7.3.1 | Loads | 48 |
| 7.3.2 | Support..... | 48 |
| 8 | Workflow & Algorithm | 49 |
| 8.1 | Inputs..... | 50 |
| 8.2 | Topology Definition..... | 50 |
| 8.2.1 | Function parameterisation: Gaussian Landscape..... | 50 |
| 8.2.2 | Geometry mapping: slicing the Landscape twice with a sigmoid | 52 |
| 8.3 | Structural Calculations | 60 |
| 8.3.1 | Stiffness matrix..... | 61 |
| 8.3.2 | Self-weight..... | 61 |
| 8.3.3 | Stress..... | 62 |
| 8.4 | Optimization | 62 |
| 8.4.1 | Objective..... | 62 |
| 8.4.2 | Constraints | 65 |
| 8.4.3 | Gradient..... | 65 |
| 8.4.4 | Optimizer | 66 |

| | | |
|---------|--|-----|
| 8.4.5 | Optimization inputs..... | 66 |
| 8.5 | Summary of workflow | 66 |
| 9 | Results | 69 |
| 9.1 | SGD optimizer..... | 69 |
| 9.2 | Adam optimizer | 70 |
| 9.3 | Proposed topologies | 72 |
| 10 | Post-processing | 73 |
| 10.1 | From superimposed stress fields to discrete topology | 73 |
| 10.2 | Post-processed design performance | 77 |
| 11 | Conclusion | 78 |
| 11.1 | Design in context..... | 78 |
| 11.2 | Research goals..... | 79 |
| 11.3 | Research question | 80 |
| 12 | Discussion..... | 81 |
| 12.1 | Evaluation of design..... | 81 |
| 12.2 | Comparison of results..... | 82 |
| 12.3 | Algorithmic performance | 86 |
| 12.3.1 | Computational complexity | 86 |
| 12.3.2 | Is the algorithm a good representation of the problem? | 87 |
| 12.4 | Wider application possibilities..... | 90 |
| 12.5 | Further research..... | 92 |
| 12.5.1 | Optimizer..... | 92 |
| 12.5.2 | Learning rate..... | 92 |
| 12.5.3 | Parallel processing..... | 92 |
| 12.5.4 | Hard Manufacturing criteria | 93 |
| 12.5.5 | Initialization of shift and spread parameters..... | 93 |
| 12.5.6 | Number of basis functions..... | 93 |
| 12.5.7 | Sigmoid slopes | 95 |
| 12.5.8 | Amplitude..... | 96 |
| 12.5.9 | Structured investigation..... | 96 |
| 12.5.10 | Further automation of post-processing..... | 97 |
| 12.5.11 | Expansion to the third dimension..... | 97 |
| 12.5.12 | Structural problem definition..... | 97 |
| 13 | Bibliography | 98 |
| 14 | Appendix..... | 102 |
| 14.1 | Full domain MATLAB code validation..... | 102 |
| 14.2 | Design validation with ANSYS | 103 |

| | | |
|--------|---|-----|
| 14.3 | Preliminary Machine Learning Research..... | 106 |
| 14.3.1 | Layers | 106 |
| 14.3.2 | Deep Generative Models | 107 |
| 14.3.3 | Variational Autoencoders (VAE) | 107 |
| 14.3.4 | Conditional Generative Adversarial Network (cGAN) | 108 |
| 14.4 | Brain-storming workflows/ The paths not taken | 110 |
| 14.5 | Loss graphs of the material distributions chosen for post processing..... | 112 |
| 14.6 | Reflection..... | 114 |
| 14.7 | Stress paths..... | 116 |

1 Introduction

Structural design plays with the combination of form and material properties that consolidate in an aesthetic expression. For example, the shape of an inverted moment diagram, together with brittle baked clay elements, come together in a romantic brick arch.

Glass is one of those indisputably beautiful materials – together with ceramics, wood and rocks – which have endured the test of time, and once present in our buildings, have never left. It has too much to offer. It transcends the boundary of practicality and enters into the realm of art. How can we continue using this material, given the new possibilities for design and manufacturing? What new roles can this ancient personality play in the buildings of today?

Does the value of this brittle beauty go beyond being a window? Additionally to its translucent visual property, glass has surprisingly high compressive strength, and is also endlessly recyclable (with reservations made to possible additives used). This leads to the question of whether it could also be a valuable structural material.

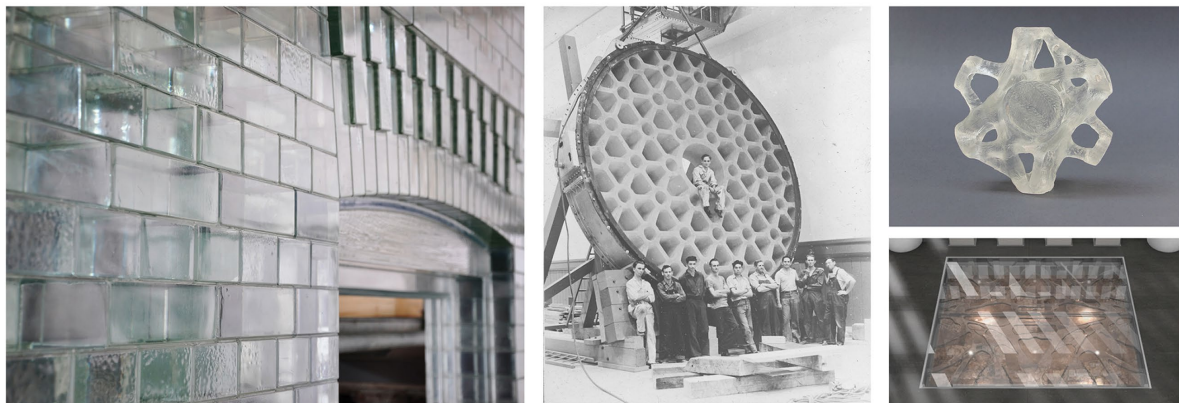


Figure 1.1: (from left to right) The Crystal Houses façade made of cast glass bricks/ Mirror of Mt. Palomar telescope. Image source: Collection of the Rakow Research Library, The Corning Museum of Glass/ Glass node (top) and floor (bottom) designed with TO. (Oikonomopoulou et al., 2022)

The use of glass as a structural material poses many challenges (due to its brittle nature and weapon-like shards) and the annealing time needed (Jewett et al., 2024), but also opens doors for curiosity: given the material properties of glass, what structural shape would allow it to perform best? The performance of structural glass is measured by its structural performance, as well as annealing time (dictated by the thickness of the cross section, mass distribution, and glass composition) (ibid).

Topology optimization (TO) has been developed to answer the question: how should material be distributed within a design domain given its material properties, boundary conditions, loads, constraints and objective?

This indicates that topology optimization could be a valuable tool to aid in creating a material distribution of glass that will cool/anneal within a reasonable amount of time, by imposing manufacturability constraints on the thickness of the cross section. Additionally, it could also help reduce the amount of material needed, and consequently the energy needed in the heating of the material, by including the minimization of volume in the objective function. Exploration of stress-reduction would also be valuable given the brittle nature of glass.

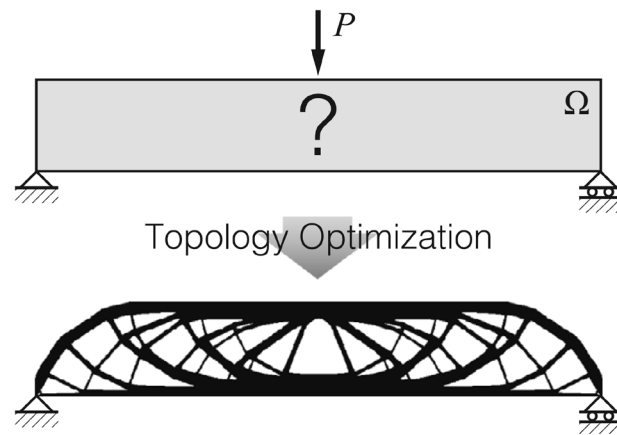


Figure 1.2: Example of how topology optimization can be used to identify an efficient structure within a design domain with specified loads and boundary conditions. In this example, the design domain is discretised with solid elements. The design is generated using a density-based topology optimization framework where the material volume is restricted to be 50% of the design domain. (Carstensen et al., 2023)

TO has been popularized by the work of Sigmund (Sigmund, 2001), who wrote an accessible paper coupled with a 99 line MATLAB code using the Solid-Isotropic Material with Penalization (SIMP) methodology and a compliance objective. Ten years later, this led to publication of a more efficient 88 lines code created in collaboration with Andreassen, Clausen, Schevenels and Lazarov (Andreassen et al., 2011).

The SIMP methodology excels in being easy to understand and having freedom of design. But it also has drawbacks. Sometimes, even this freedom itself is undesirable. Langelaar (2016) took the manufacturability constraints as the driving force for developing a methodology that is compatible with the pseudo-density approach, but is specifically tailored to Additive Manufacturing. It evaluates the performance of the geometry in its *as-printed* form including any supports needed during the printing process. This ensures that the optimal topology is the one with the lowest cost, including the extra material and printing effort needed to construct it.

Carstensen et al. (2023) also focused on tailoring TO to AM, but developed a different framework than their peers. They focused on creating a ground structure of user-defined, discrete elements that could be used in the composition of the optimized topology, connected nodes in pre-specified positions. In this way, they are able to guarantee manufacturability and circumvent the drawbacks of the SIMP methodology that derive from its use of intermediate densities.

Guest et al. (2004) were particularly concerned with avoiding the drawbacks of the SIMP methodology that are caused by the element densities being directly used as the design variables. These drawbacks include very small features that are not physically reproducible, or whose structural performance in reality does not correspond to the abstraction made in the Finite Element Analysis. They developed a method where the minimum thickness of the cross section anywhere on the structural element is user-defined. This workflow still uses element pseudo-densities, as SIMP does, but these densities are calculated as a function of the nodal densities, which are the design variables.

Parallel to the development of the SIMP methodology and its adaptations, researches worked on the Level-Set Method (LSM). This approach relies on implicitly defining the topology through the definition of a higher-order function. In the case of a 2-D topology, a 3-D Level-Set function is defined.

Wang et al. (2003) point out the flexibility of the LSM in its ability to represent complex surfaces and join/split material during the optimization process. They approach the optimization procedure as a pseudo-temporal evolution.

Yulin & Xiaoming (2004) adapt the LSM to include multiple materials by imposing multiple LSFs to the same design domain. They allude to the possibility of combining LSFs to represent particular material properties, such as a multi-material state.

Amstutz & Andr  (2006) identify that LSMs that perform sensitivity analysis on the design variables with respect to the boundaries of the structure, can have difficulty creating new holes in the structure during the optimization process. They propose a revised version of the topological gradient to remedy this pitfall.

Wei et al. (2018) conducted a compliance-based optimization and used radial basis functions (RBFs). The topology boundaries created here reflect the qualities present in the level set functions chosen. Both the basis function as well as the topology it creates have rounded edges and smooth transitions, as seen in **Figure 1.3**.

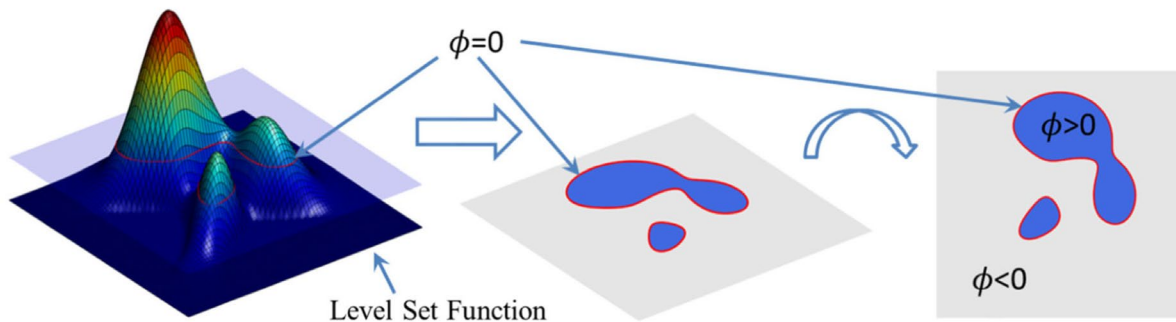


Figure 1.3: The description of a level set function for a 2-D problem (Wei et al., 2018)

TO approaches can be categorized by their methodologies, but also by their objective. Stress-based TO is especially relevant for this project, given the brittle nature of glass and the existing previous exploration of both the compliance and volume objectives in previous works (Koniari, 2022)(Schoenmaker, 2023).

(Le et al., 2010) point out the three main difficulties of stress-based optimization in comparison to its volume and compliance counterparts. These are: (i) the singularity problem, which refers to the high stress calculation in elements of low density; (ii) the local nature of stress, which requires it to be somehow condensed into one scalar for the objective function, and (iii) the highly non-linear relationship between the design and the stress level, which means that slight changes in the design can cause large changes in the stress.

According to Guo et al. (2011), stress is often the cause of structural failure, and should be included in TO. They apply the LSM with a discrete material distribution to a compliance objective in **Figure 1.4a**) and a global stress objective in **Figure 1.4b**). For the calculation of the global stress, the Von Mises Criteria and p-norm are used.

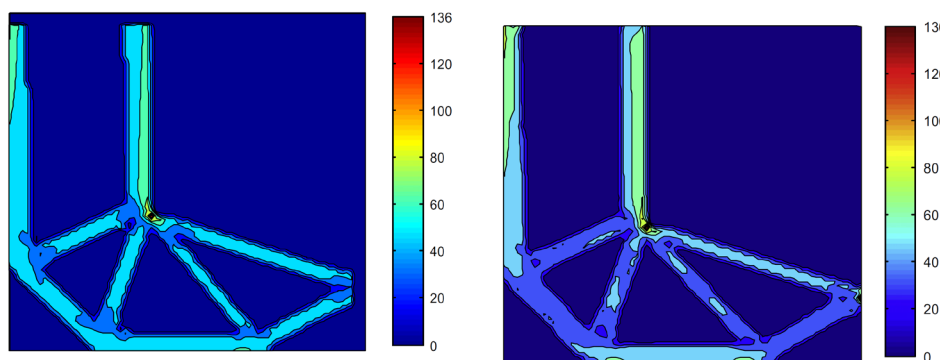


Figure 1.4: Optimal design of the L-shape beam problem with (a) on the left, compliance as objective function and the corresponding stress distribution; (b) global stress as objective function and the corresponding stress distribution. (Guo et al., 2011)

Another approach to including stress in TO is presented by Holmberg et al. (2013). They deal with the local nature of stress by clustering the stresses into groups during the TO process and applying an adapted p-norm to the clusters, the result of which are used as constraints in the TO process.

Kiyono et al. (2016) further explore possible solutions to the local nature of stress. They propose a method that does not rely on clustering, but combining different values of p for the p-norm to achieve a topology that is stiff, while also avoiding peak stresses. They use the Von Mises criteria in combination with the p-norm, and include stress in their objective function instead of as a constraint in the optimization process.

Zhang et al. (2017) also conduct a stress-based optimization. They provide an alternative to free-form topology optimization techniques (such as Level-Set or SIMP) by proposing the use of discrete geometric components to create the optimized geometry.

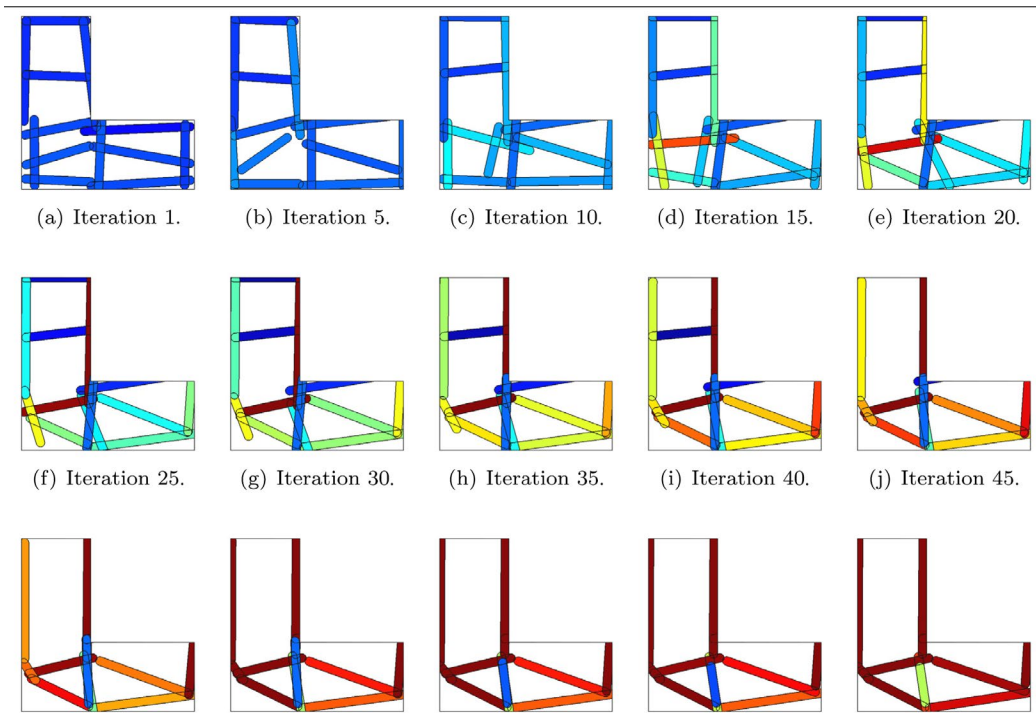


Figure 1.5: 2-D L-bracket design history for problem \mathcal{P}_σ^2 . The color denotes the penalized size variable for each geometric component. (Zhang et al., 2017)

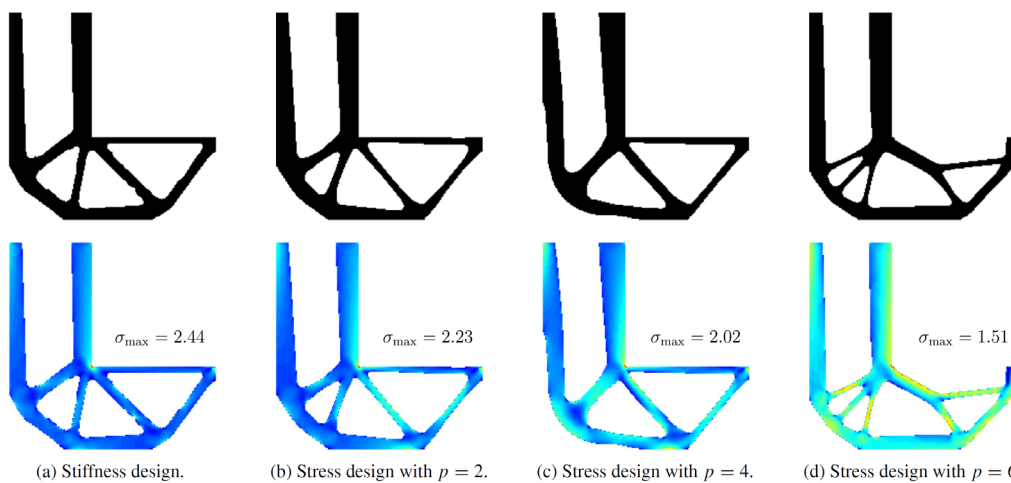


Figure 1.6: Comparison of topology designs of the L-bracket for different stress norm parameters (the color ranges from zero to the maximum von Mises stress for each case). (Xia et al., 2018)

Xia et al. (2018) use the bi-directional evolutionary structural optimization (BESO) method on a stress minimization problem. This method avoids the singularity problem present in density-based methods, due to its discrete topology definition. They include historical optimization information to stabilize the optimization process. They also employ the p-norm as a global stress measure. They remark that if the value of p is too large, then only the peak stress is targeted by the algorithm, while the others are not, this is visualized in **Figure 1.6** They use the Von Mises stress criteria.

Picelli et al. (2018) apply the LSM to TO. They explore comparisons of objective and constraints, including stress in both positions. Their update method relies on computing the sensitivity of the design variables with respect to the topology boundary. They also apply a least squares interpolation to increase the accuracy of the stress prediction.

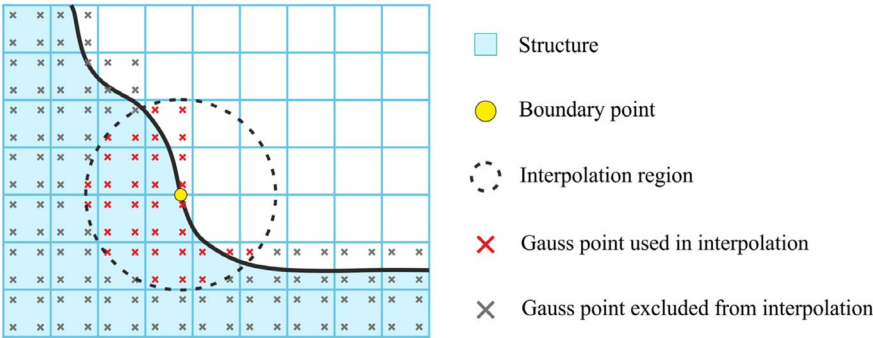


Figure 1.7: Gauss point sampling scheme for the least squares interpolation method. (Picelli et al., 2018)

Stress is especially important because of the material properties of glass, which include a vastly lower tensile strength than compressive strength. This requires a specific algorithm development because existing commercial software is not able to capture this difference (Koniari et al., 2023). The following works have investigated the application of TO to cast glass.

Previous work done at TU Delft on this subject includes a two-dimensional topology optimization algorithm specifically tailored to cast glass structures (Koniari, 2022), and its expansion to the third dimension with improved optimization speed (Schoenmaker, 2023). They focused on adapting a well-tested topology optimization methods (i.e. SIMP) to glass by including the brittle material strength differences in tension and compression, additional checks on thickness due to residual stresses formation during uneven cooling, gap width check for ensuring the minimum size needed for printing the mould, and post-processing for rounded edges, and aimed to reduce volume.

| Name | Optimization Result ^a | Volume (m ³) | Annealing time (hours : minutes) ^b |
|--------------------------|----------------------------------|--------------------------|---|
| BR-PN-30 | | 0,690 | 35:00 |
| SL-PN-30 | | 0,738 | 55:20 |
| BR-FX-30 | | 0,419 | 18:30 |
| BR-PN-40 | | 0,443 | 23:30 |
| Compact slab (h = 17 cm) | | 0,821 | 83:30 (BR) 249:50 (SL) |

Figure 1.8: Cast glass optimization (Koniari et al., 2023)

The work of Koniari et al., (2023) applied TO to the design of a cast glass bridge with a volume minimization objective. **Figure 1.8** showcases the results. It highlights how the support condition (either PN: pinned or FX: fixed) and the height of the design space yield different topologies. A vast reduction in the volume, and thus the cost of material and manufacturing, is achieved by applying TO. This encourages further exploration of TO possibilities.

Jewett et al. (2024) have used the SIMP methodology with three different minimization objectives: compliance, volume and stress, physical specimen of the optimized topologies can be seen on **Figure 1.9**. For the computation of the stress in the objective function, they have used the Drucker-Prager failure criterion, which is specifically tailored to brittle materials. They have produced the glass specimen by water-jet cutting float glass sheets and performed the four point bending test. Somewhat surprisingly (given that glass tends to fail due to tensile stresses), the stress-optimized topology had the lowest performance, while the stiffness-optimized topology performed the best. This may be due to many factors, and deserves further investigation.



Figure 1.9: Images of fabricated waterjet cut float glass beam specimen. The volume-based is on the left, the stress-based is in the middle, and the stiffness-based is on the right, on a separate photo. (Jewett et al., 2024)

The literature research shows some inspirational aspects, and also highlights gaps that could be further explored. The present research focuses on a specific problem. It is concerned with a particular brittle material (i.e. glass) that has very specific manufacturing constraints (i.e. even thickness and rounded transitions), and postulates whether its structural use can be informed by topology optimization. Therefore, it resides at the intersection of the three topics: (1) topology optimization, (2) brittle material, and (3) cast glass manufacturing constraints.

In terms of methodology, all of the TO approaches applied to cast glass have used the SIMP. The author hypothesises that perhaps another method, tailored to the manufacturing constraints of glass could be more suited. Additionally, to the best of the author's knowledge, only one cast-glass topology has been optimized including stress in the objective function (Jewett et al., 2024). Thus the investigation of a manufacturing-focused alternative method for cast-glass stress-based optimization could contribute to the field.

In terms of the brittleness of the material, there are two aspects to be considered. The first is the aforementioned inclusion of stress in the objective function, and the second is the adoption of the Drucker-Prager criteria, as done in Koniari et al. (2023) and Jewett et al. (2024).

This project explores the applicability of glass in the role of a sole structural element. It further explores a two-dimensional topology optimization approach, but with a novel workflow and a stress-based optimization algorithm. It combines different concepts of known optimization approaches in its algorithm. It attempts to create a topology definition that already incorporates soft constraints on the manufacturing requirements of glass (even thickness and rounded edges), significantly reduces the number of optimizable parameters, and minimizes the amount of effort needed during post-processing.

As seen in on the images of the different methodologies above, the type of material distribution definition has a large impact on the optimized design. The present work proposes the use of the level-set method with global RBFs basis functions to encourage rounded transitions. Furthermore, it explores the possibility

of double-slicing the LSF to create an even cross-sectional thickness throughout the design, as a way of embedding the even cooling manufacturing constraint into the optimization process. By doing so, the use of filters to tackle manufacturability is avoided.

The workflow uses an adapted sigmoid slicing function to project the LSF into a density-based material distribution, which is used for the structural calculations. It does so in order to ensure the differentiability of the entire optimization loop: from material definition to the objective function. This allows the application of automatic differentiation and a pre-defined optimizer in pytorch to update the weights of the model during the optimization process, exploiting the similarities between optimization and machine learning.

The case study is an indoor pedestrian bridge at the British Museum. The project will be evaluated by comparison with the works of Konari (2022) and Schoenmaker (2023), who have investigated the same case study, but with a volume objective and applying the SIMP methodology. The performance will be compared in terms of structural performance (maximum deflection, tensile and compressive stresses) and computational efficiency (number of parameters needed during the optimization, and time taken to achieve the optimized topology). Furthermore, the proposed topology will be visually evaluated by applying basic structural intuition, as well as a validation using ANSYS.

This project shows that the possibilities of constructing a tailored optimization process are vast, but also full of caveats. The workflow proposed performs well in terms of reducing the theoretical stress in the structure, and has comparable volume to the volume-based TO results achieved previously. However, the methodology still shines through the design and structural performance improvement can be visually identified, showing that the algorithm can be further improved.

The following section will outline the research questions/goals. It is followed by literature research chapters on: glass as a structural material (Chapter 2), optimization (Chapters 3 and 4), structural calculations using the Finite Element Method (FEM) (Chapter 5), stress in the context of optimization (Chapter 6), and the case study considered in this project (Chapter 7). The design part comes next with: the designed workflow and algorithm (Chapter 8), results (Chapter 9), and post-processing (Chapter 10). Finally, the closing chapters evaluate the work with a conclusion (Chapter 11) and discussion (Chapter 12).

1.1 Research

1.1.1 Problem statement

Design a computational workflow that minimizes both stress and volume to output an optimized structural 2-D cast glass topology that is manufacturable and time efficient.

1.1.2 Research goal

The research goal is to design:

- (1) The material distribution definition, specifically tailored to cast glass
- (2) The optimization algorithm, specifically tailored to brittle materials
- (3) A workflow that combines material constraints, topology (a.k.a. material distribution) definition, and optimization algorithm, using current computational tools.
- (4) Use the above to inform the design of an indoor cast glass pedestrian bridge at the British Museum.

1.1.3 Research question

How can a (two-dimensional) stress-based optimization algorithm be set up for designing a cast-glass structural element that takes into account manufacturing constraints and the brittle nature of cast glass?

1.1.4 Sub questions

1.1.4.1 Case Study

What have the previous explorations of this Case Study found?

What are the *boundary conditions* of the structural element?

1.1.4.2 Glass

Which *material properties* of glass are relevant for this project?

What are the *manufacturing constraints* of structural glass components?

What are the structural requirements (e.g. maximum deflection, safety factors) when designing a glass structural element?

1.1.4.3 Structural Calculations

How can *structural calculations* (stress, displacement) happen efficiently within an algorithm?

What are the possible *dimensions and shapes* of each finite element? How can the design space be discretised?

How can the load application in the structural model best *represent the real scenario* of the case study?

1.1.4.4 Optimization

What are the *types* of structural optimization?

What is the most suitable *combination of “objective” and “constraints”* for this project? And how do they relate to each other?

What is *stress*? How can it be quantified in the context of structural topology optimization?

What are the *challenges* specific to stress-based optimizations?

How are *material distributions* usually defined during the optimization process?

Is there a material distribution definition that could be *particularly suited to cast glass* elements in a museum setting?

What are the traditional *topology optimization methods* found in literature?

What *machine learning* concepts can be used for topology optimization and how?

Is there an alternative to traditional methods for topology optimization of structural cast glass?

1.1.4.5 Workflow

How can the ingredients mentioned above come together in a *workflow* that outputs optimised cast glass structures?

What do the individual algorithms look like that come together in the above mentioned workflow? Which requirements must they fulfill to form one cohesive whole?

1.1.4.6 Discussion

How does the optimised topology *perform* structurally and visually?

Did it perform as *expected*?

How well did the *algorithm* perform?

What are the possible areas for *future research*?

1.1.5 Methodology

The methodology is composed of four research components: Case Study, Glass, Structural Calculations and Optimization. It is followed by four design components, of which the first three are computational: (i) material distribution definition, (ii) optimization algorithm, (iii) overall workflow, and (iv) the Case Study design.

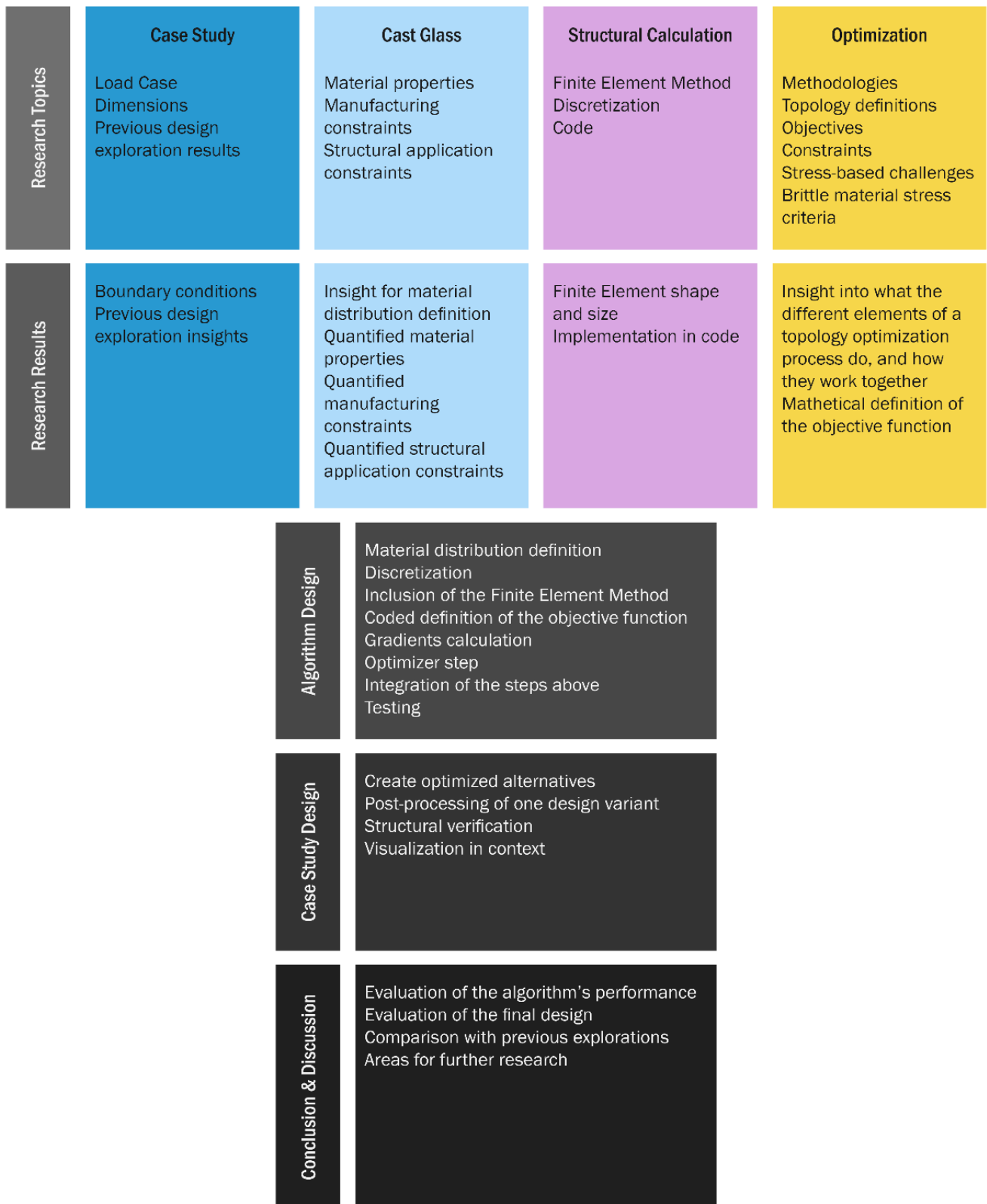


Figure 1.10: Research and Design Plan

2 Glass

Glass is a remarkable material from aesthetic, structural and sustainability points of view. Aesthetically, it has made it possible for us humans to create visual openings in buildings without allowing wind and water to enter, completely changing the feeling of our dwellings. Structurally, it has very particular qualities, with incredible compressive strength, contrasted with fragile brittleness. From the sustainability angle, it has the potential to be endlessly recycled, reshaped, repurposed, as long as the additives used are carefully considered.

Here, structural cast glass is considered. This project relies heavily on previous research done on the different types of glass by Oikonomopoulou (2019) and Bristogianni et al. (2020), and adopts Koniari's (2022) and Schoenmaker's (2023) decision of using Borosilicate glass, for its reduced annealing time, since for a very large glass component, the annealing time is critical.

2.1 Material properties

2.1.1 Brittleness

Glass has an amorphous atomic structure, the bonds are randomly arranged as opposed to following a repeated pattern. This is what gives glass its much beloved and distinctive transparency.

Glass is also a brittle material. Thus, unlike ductile amorphous materials (such as some metals), glass is unable to alleviate severe localized stress state through deformation, and instead experiences brittle failure (Steift, 1983). Brittle failure is the sudden rupture of the material at the end of its elastic deformation. Brittle materials are unable to enter the plastic deformation stage that could have helped to alleviate localized stresses.

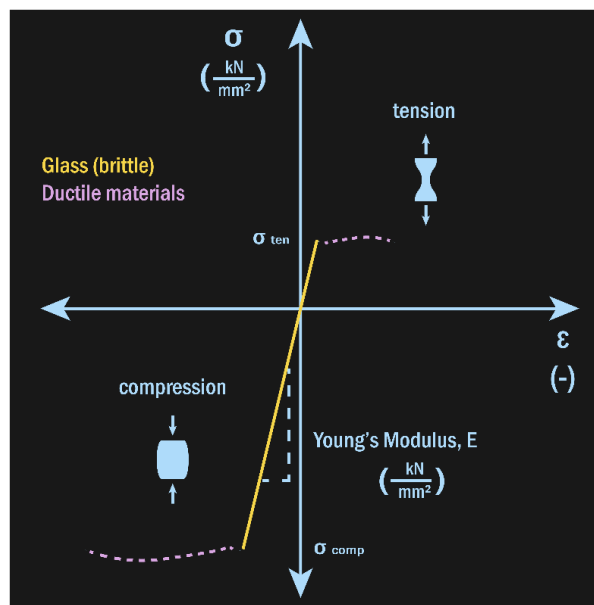


Figure 2.1: Qualitative stress-strain diagram for glass

Figure 1.1 shows a qualitative stress-strain diagram for glass, where the two properties mentioned above, 1) brittleness and 2) uneven strength and toughness in tension and compression, can be visualized. Firstly, the material experiences failure at the end of the elastic stage, without entering plastic deformation. Secondly, the material is able to endure a vastly greater amount of compressive ($500 \times 10^9 \text{ kN/mm}^2$) than tensile ($6.4 \times 10^9 \text{ kN/mm}^2$) stress. It is about 80 times stronger in compression than tension.

An important quality of brittle materials is that they tend to be stronger in compression than in tension. This is at least in part due to the fact that localized tensile stresses propagate, while compressive ones do not. This tensile stress propagation drives the structure to failure.

Given these properties, it is very pertinent in the design of brittle structures to consider the stress distribution in the material given the design load, in order to avoid sudden failure, which apart from rendering the structure unusable, could also cause injuries. It is also important to include both compressive and tensile stresses, given the vast difference in the material's ability to withstand them. This will be addressed in the stress criteria section.

2.1.2 Isotropic

Glass is an isotropic material: the material properties do not change depending on the direction being considered. This means that when calculating the deformation, displacement, and stresses on the structure, the values for Young's modulus, Poisson's ratio, and tensile and compressive strength remain the same regardless of the direction.

2.1.3 Young's Modulus

Young's Modulus is the ratio between stress and strain during elastic deformation. It depends on the strength of the bonds between the atoms, as opposed to plastic deformation, which depends on the ability of atoms to rearrange themselves. Glass has interatomic bonds, which are stronger than intermolecular bonds found in other materials, making it very stiff. Young's Modulus, E , of cast glass is 70 GPa or kN/mm² (Bristogianni et al., 2020).

2.1.4 Poisson's ratio

Poisson's ratio quantifies the ratio of deformation in two perpendicular directions when a material is under stress. It is the relationship of the deformation of the material in the direction of the force applied and in the direction perpendicular to it. For most materials, including glass, this ratio is below one, meaning that when a material is stretched in one direction, it becomes thinner in the other direction. Poisson's ratio for glass is 0.2 (unitless).

2.2 Manufacturing constraints

Two main contributors to manufacturing constraints are the annealing time and the internal stress created during cooling.

2.2.1 Mold

For a massive topology optimized glass structure, cast glass can be used. For the mould material, sand is very suitable, because it can be moulded to fine details, easily removed as long as there is a path to the outside of the structure, and even reused in future projects (Oikonomopoulou et al., 2023).

The mould can be 3-D printed if a complex geometry is desired. The gaps in the structure, which correspond to the material areas of the mould, should be large enough to be printed, and to be easily emptied after the glass has cooled.

Another inherited conclusion from Koniari (2022) and Schoenmaker (2023) is the choice of using secondary casting (Oikonomopoulou, 2019) since it only requires one kiln for melting and annealing. In the secondary casting method, the glass granulate is placed in the mould heated up to melting temperature, it then stays in the same kiln for the annealing process.

2.2.2 Cooling time, maximum thickness

Annealing is the process of slowing down the cooling rate. It minimizes internal residual stress build up.

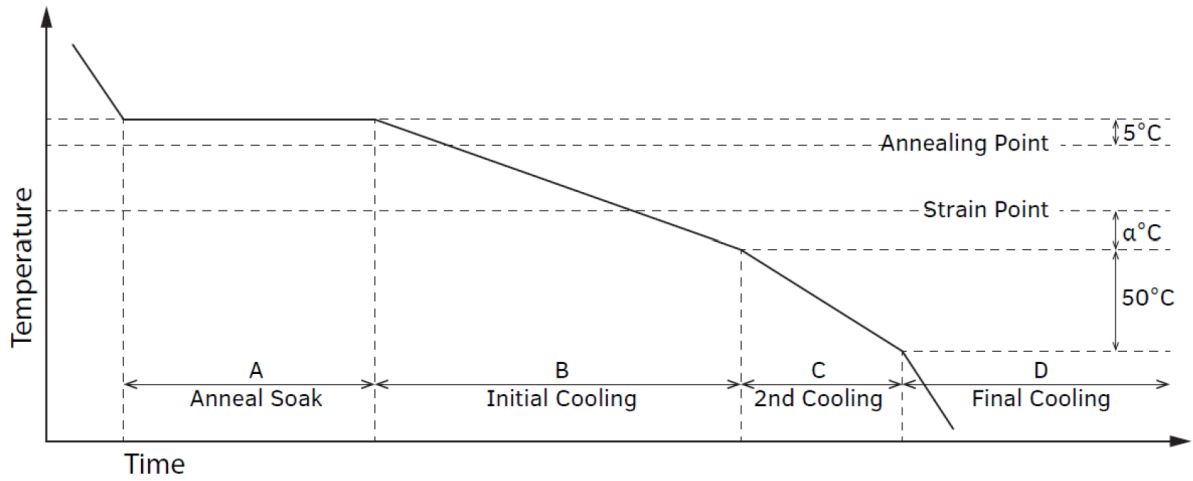


Figure 2.2: Typical annealing scheme for commercial soda-lime glass. (Oikonomopoulou, 2019)

Residual stress is created due to some of the glass cooling and hardening faster than other parts of the same element. For example, the middle cooling after the sides. This difference in hardening times causes internal stresses and strains to get captured within the glass. These stresses could take away available strength for the glass to support the applied loads and should thus be minimized. The allowable permanent stress is 1 MPa (Koniari, 2022). For cooling from the top, in an annealinglehr, the following formula is adapted from Hubert (2015):

Eq. 2.1

$$\frac{\Delta T}{t_{ann}} = \frac{\sigma_{res}}{\frac{E \alpha_T \rho c_p}{(1 - \nu) \lambda} \times d^2 \times b}$$

Where:

ΔT is the initial cooling range during annealing (K)

t_{ann} is the time the initial cooling takes (s)

$\Delta T / t_{ann}$ is the cooling rate (K/s)

σ_{res} is the maximum allowable residual stress (MPa), usually equal to 1.

d is the characteristic dimension

b is the shape factor

E is the Young's Modulus

α_T is the thermal expansion coefficient (K^{-1})

ρ is the density (kg/m^3)

c_p is the specific heat capacity ($J/(kg \cdot K)$)

ν is the Poisson's ratio (-)

λ is the thermal conductivity ($W/(m \cdot K)$)

This formula can be used to set a constraint on the maximum allowable cross section, by calculating if the amount of time that it would take it cool it down still falls within the amount of time that is deemed allowable. The values d and b of the cross section are the changeable factors here during the design exploration.

$$d^2 \times b = \frac{\sigma_{res}(1 - \nu)\lambda t_{ann}}{E \alpha_T \rho c_p \Delta T}$$

2.2.3 Even thickness

The next factor of importance is the range of thicknesses present in the cross section. Those should not differ too much in order to prevent uneven cooling and more residual stress formation. For this, the largest cross section should not be larger than thrice the smallest cross section (Schoenmaker, 2023).

Combining this requirement with the one from the section above, the following formulation arises:

$$d_{max} = \begin{cases} 3 \times d_{min} , & \text{if } t_{ann}(3 \times d_{min}) \leq t_{ann,max} \\ \sqrt{\frac{\sigma_{res}(1 - \nu)\lambda t_{ann}}{E \alpha_T \rho c_p \Delta T b}} , & \text{if } t_{ann}(3 \times d_{min}) > t_{ann,max} \end{cases}$$

The glass should also have a minimum thickness to prevent it from cooling too fast in comparison to the rest of the structure, and to ensure that it can be manufactured.

The thinner the section, the faster it will cool. However, it should also not be too thin, to avoid accidents. The minimum thickness as advised by Oikonomopoulou is 20 mm, with a corresponding maximum thickness up to 3 times the minimum thickness, as expressed in the formula above.

In case the manufacturing constraint applied during the optimization does so in a soft manner, the rule of thumb can be used. If the constraint is applied in a hard manner, the formula can be used.

2.2.4 Sharp edges

Sharp edges have two major downsides. Firstly, they tend to cool quicker than the rest of the structure and cause internal stresses, as described above. Secondly, sharp edges are at a higher risk for stress concentrations when the structural element is loaded, since internal sharp edges are sudden cross-sectional changes.

Smooth, rounded transitions are preferred for glass, so that no abrupt changes in cross section occur and thus the localized stresses can be better alleviated during loading, and also so that a gradient cooling occurs through the structure, reducing the risk of trapped residual stresses during manufacturing.

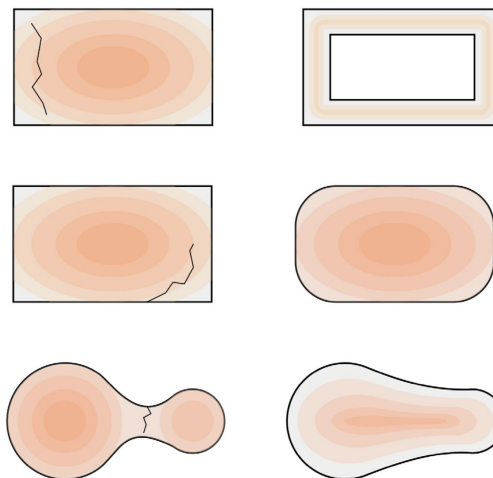


Figure 2.3: Visualization of the manufacturing constraints (Damen, 2019).

2.3 Structural constraints

2.3.1 Deflection

The constraint for serviceability requires that the maximum deflection be no larger than the span of the bridge divided by 500. In this case, the bridge is 4200 mm long, so the maximum allowable deflection is 8.4 mm.

2.3.2 Safety factors

For glass, the safety factors for permanent loads is 1.2 and for temporary loads is 1.5 (Schoenmaker, 2023).

2.4 Glass composition and support condition

The choice of glass composition, support condition and height of the design domain is based on the work of Koniari (2022). In Table 2.1, she has presented her findings of different combinations of the aforementioned structural design choices. It is apparent that a borosilicate glass composition with a fixed edge support and casting manufacturing method has had the best performance in terms of minimizing the volume, when compared to the use of soda lime or hinged supports. Thus, this same glass composition and support condition is chosen for the current project.

A hinged connection might have alleviated peak stresses at the top of the beam at the supports, and could be investigated in future research.

In terms of the structural height of the design domain, the table shows that a height of 30cm performs better than a height of 20cm. The current work will further facilitate a better performance by making a height of 40cm available to the optimization algorithm.









| Variations | | Result | Volume |
|------------|---|--|--------|
| | Casting Edge Supports Borosilicate |  | 935.8 |
| | Casting Edge Supports Soda Lime |  | 937.3 |
| | Stacking Edge Supports Borosilicate |  | 914.7 |
| | Casting Point Supports Borosilicate |  | 1500.2 |
| | Casting Point Supports Soda lime |  | 1604.5 |
| | Stacking Point Supports Borosilicate |  | 1381.3 |
| | Casting Edge Supports Borosilicate (20cm) |  | 1066.1 |
| | Total volume (cross section 30cm) |  | 3150 |

Table 2.1: Volume comparison of the different optimization results (Koniari, 2022).

2.5 Quantified properties and constraints

For this project – building onto the work done by the predecessors Koniari (2022) and Schoenmaker (2023) – borosilicate glass is modelled. It has the following material properties:

| Property | Symbol | Unit | Value |
|--|------------|-------------------|-----------------------|
| Young's Modulus | E | GPa | 70 |
| Poisson's Ratio | ν | - | 0.2 |
| Density | ρ | kg/m ³ | 2500 |
| Initial cooling range (annealing process) | ΔT | °C | 530 - 460 = 70 |
| Thermal expansion coefficient | α_T | 1/K | 3.25×10^{-6} |
| Thermal conductivity | λ | W/(m*K) | 1.15 |
| Specific heat capacity | c_p | J/(kg*K) | 800 |

Table 2.2: Glass Properties (Koniari, 2022).

| Property | Symbol | Unit | Value |
|--|--------------------|------|-----------------|
| Design tensile strength | $f_{t,des}$ | MPa | 6.4 |
| Design compressive strength | f_c | MPa | 500 |
| Deflection | δ | mm | $l/500 = 8.4$ |
| Maximum annealing time | $t_{ann,max}$ | s | 432000 (5 days) |
| Minimum element dimension | d_{min} | m | 0.06 |
| Ratio of maximum to minimum element dimension | r_{ann} | - | 2 |
| Maximum permanent residual stress | $\sigma_{res,max}$ | MPa | 1 |
| Permanent load safety factor | α_p | - | 1.2 |
| Temporary load safety factor | α_t | - | 1.5 |

Table 2.3: Design constraints (Koniari, 2022).

3 Optimization process

Optimization is the process of iteratively minimizing a function to find (local) minima. To set up a structural optimization, the following steps must be taken (which will be dealt with in the following section(s)):

- 1) Choosing the most suitable type of structural optimization (Section 3.1)
- 2) Defining the overall methodology, including the topology definition (This is an extensive topic and will be dealt with in Chapter 4)
- 3) Structural calculation (Chapter 5)
- 4) Objective & Constraints (Section 3.2)
- 5) Update parameters, which requires a gradients calculation (Section 3.3) and optimizer step (Section 3.4)

Figure 3.1 shows the main components of a topology optimization workflow, which includes the steps mentioned above and their interactions.

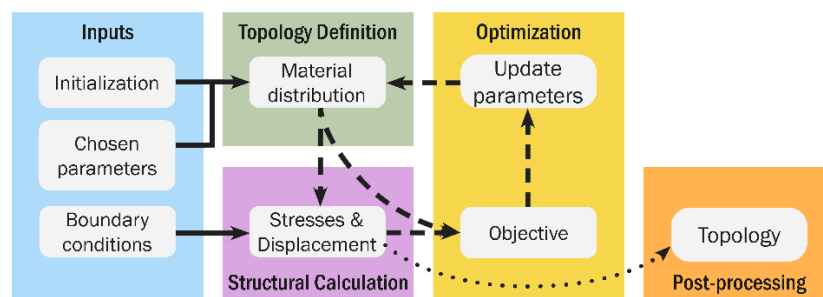


Figure 3.1: Compact workflow diagram. The dashed lines represent interactions that take place iteratively during the optimization process.

The Inputs quadrant refers to the choices made outside of the optimization cycle. These remain constant throughout the process, but still influence the outcome. They include: material properties, support condition and load case (i.e. magnitude and points of application of the temporary load, and magnitude of the self-weight of a single finite element), as well as an initial guess about the material distribution, such as the initialization method (e.g. randomized or evenly distributed) or chosen parameters (e.g. mesh density, parameters related to the desired maximum and minimum cross-sectional thickness).

The topology definition quadrant is in charge of outputting a material distribution to which the FEM can be applied to determine its structural performance. It is in this section that the design parameters are found. This can take many forms, as will be seen in the Optimization Methods Chapter, from the pseudo-densities of density-based methodologies to parameters of a LSF in discrete formulations, and even some combinations of the two. The important aspects are threefold: 1) the material distribution is defined in a parametric way (i.e. it can be changed at each iteration of the optimization process), 2) the material distribution can be used in the calculation of the structural performance, and 3) it is possible to calculate the effect of changing these parameters on the result of the objective function (i.e. the algorithm must be able to calculate in which direction to change each of the parameters in order to decrease the value of the objective function).

The Structural Calculations quadrant performs FEA given the material distribution of this particular iteration and the static input (material properties, boundary conditions, etc.), and outputs structural performance fields (displacement and stresses).

In the Optimization quadrant, two main operations take place: the objective and constraints calculation and the update of the design parameters.

The objective calculation may take the output from both the material distribution and the structural calculations, depending on the particular objective definition. The material distribution contributes to the volume and stiffness, while the structural performance defines the displacement and stress fields. These quantifications may be present in either the objective or constraint functions, and evaluate the design as a whole. The objective function condenses the performance into one scalar value. This may require additional mathematical operations to be performed on the raw outputs, especially with regards to outputs that are composed of a matrix of values (e.g. the displacement and stress fields).

The second part of the optimization quadrant, the update of the design parameters, takes the result of the objective function and its dependence on the current design parameters. It computes how the objective function changes with respect to the parameters (i.e. the gradients), and adapts those parameters to attempt to achieve a smaller value for the objective function in the next iteration. There are many methods for achieving this.

Then the cycle starts again, with the new material distribution being defined by the updated parameters. These four quadrants are performed in sequence for the total number of iterations. The number of iterations can be previously defined or can be decided during the optimization process, given the progress. Either way, after the optimization is completed, the material distribution exits the cycle and enters the post-processing phase.

Now that the general structural optimization workflow has been outlined, the following subsections provide specific ways of defining each of the steps above.

3.1 Type: size, shape, topology

There are three types of structural optimization: size, shape, and topology (Bendsøe & Sigmund, 2004).

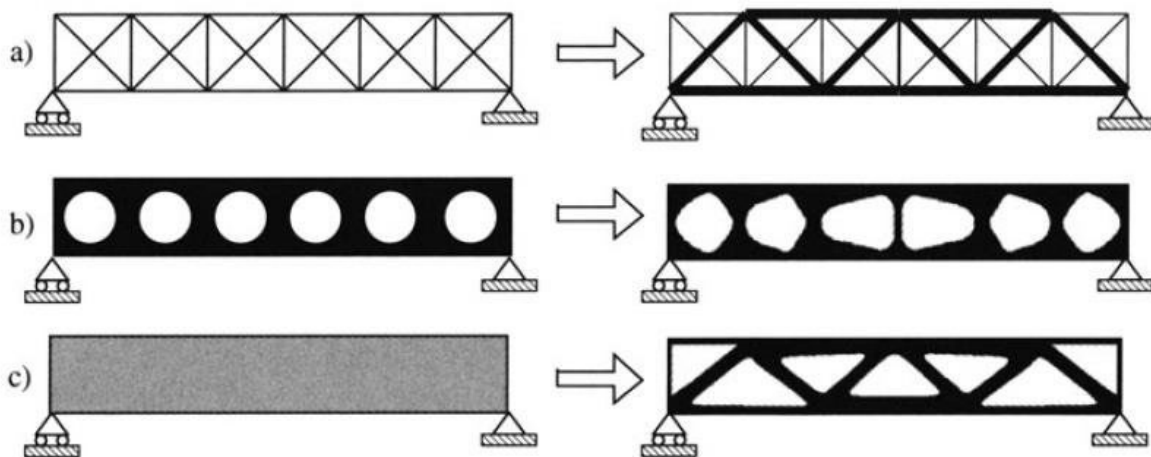


Figure 3.2: Structural optimization types: a) size, b) shape, and c) topology (Bendsøe and Sigmund, 2004)

In size optimization, only the dimensions of the element are optimized. This could be, for example, optimizing the height and width of a rectangular wooden beam spanning two walls, so that the space needed in the building for the wooden structure is minimized. In this case, only the dimensions are affected. The shape stays the same.

Shape optimization goes a step further and also takes into account holes in the structure. Both the external boundary of the element and the shape of the holes are optimized, but the positioning and amount of holes is predefined.

Topology optimization is the most comprehensive of the three. There are no predefined positions or shapes. It does not require an initial guess of the optimal structure (Van Dijk et al., 2013). There is a boundary where the structure can exist (i.e. the design domain), and – unless otherwise enforced – there are no imposed constraints inherent to the topology optimization process itself. It offers the most freedom of design, but is also the most computationally expensive of the three.

In order to be able to optimize a geometry (i.e. iteratively change it to achieve a pre-defined performance goal), this geometry has to be parametrically defined. In literature, there are two prominently used methods: pseudo-density and level-set approaches.

This paper will focus mainly on the traditional methods, but some preliminary research on machine learning (ML) is included in Appendix 14.3.

3.2 Objective: compliance, volume, stress

The objective of the optimization is the quantity being minimized in the optimization process. The algorithm employed aims at finding the lowest value possible while tweaking the changeable parameters of the system. The constraints pose a limit on certain calculated quantities. The algorithm has to stay within the permitted boundaries. It is indifferent to where within the allowable range that value lies. In other words, the objective is relative, the constraint is absolute.

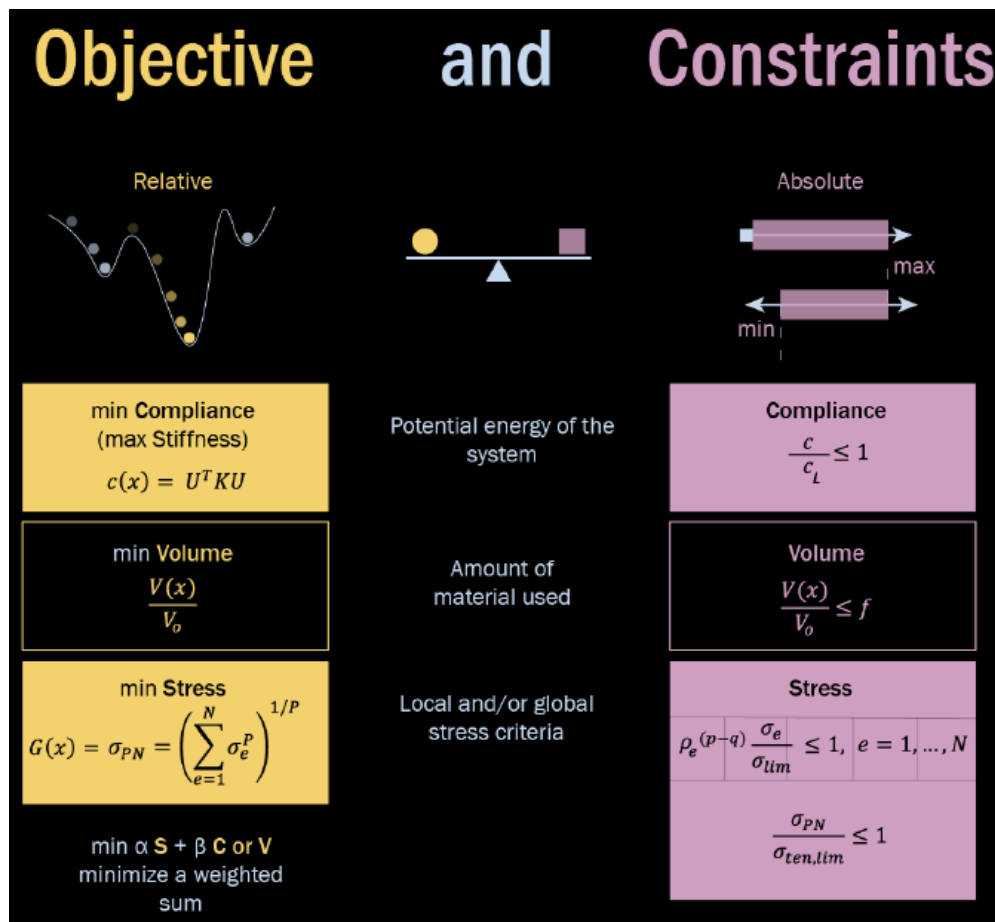


Figure 3.3: Overview of the three most common types of structural optimization objectives and constraints.

The three most common objectives for structural topology optimizations are: compliance, volume, and stress. Usually, these three goals are present in the optimization formulation in either the objective or constraints. **Figure 3.2** shows an overview of the objectives and constraint formulations.

Minimizing compliance is equivalent to maximizing stiffness. If an algorithm is purely given the task of minimizing compliance, it will tend to completely fill the domain, as that would give the stiffest possible structure. So compliance optimizations are usually balanced by a volume constraint. The algorithm must distribute a pre-defined ratio of the total volume of material in the design space.

This balancing act is crucial in an optimization, as having just one goal without constraints leads to extremes of either full or empty material. An example of empty material is to minimize the volume (aiming for a lightweight structure). This, by itself, would lead to a completely empty domain unless otherwise constrained. This highlights the thoughtfulness needed to craft pertinent objective and constraint equations.

The previous explorers of this Case Study have each run two optimizations, one minimizing compliance with a volume constraint, and the other minimizing volume with stress constraints (Koniari, 2022; Schoenmaker, 2023). All of their optimizations also included glass manufacturing constraints. Stress was considered as a constraint, but not as an objective. This is the last of the three common objectives to remain unexplored. Thus a contribution of this thesis is to include stress as an optimization objective. The intricacies of such a task will be discussed in Chapter 6.

3.3 Gradient calculation

This section discusses which information can be used to drive the optimization's update procedure. Gradient calculation is analogous to sensitivity analysis. It is the search for how the objective function changes with respect to the input parameters. In the past, the derivatives of the objective function had to be provided by the designer. Now, with computational advances such as auto-differentiation (in pytorch or TensorFlow), these derivatives can be automatically calculated by functions downloaded in packages in different programming languages. The work of Hoyer et al. (2019), which will be outlined in the following chapter, is one example of the use of automatic differentiation in the structural optimization context.

Here, the boundary of optimization and machine learning is blurred, as terms such as forward pass and backward pass become applicable. A machine learning algorithm optimizes its network parameters, just like a simple optimization optimizes its material distribution parameters. During the forward pass (where the model calculates the objective function given its current input parameters), it also keeps a gradient map of each mathematical operation used. When the backward pass is called, it runs backwards through this map, applying the chain rule, to calculate the derivative of the objective function with respect to the input parameters (a.k.a. the gradients).

The gradient is a key step in optimization, for it points out which direction the inputs need to change in order to decrease the value of the objective function.

No matter which method is used for the calculation of the gradients, it is always required that the process between the input parameters and the objective function be differentiable. That is, all functions must be smooth. This excludes the use of “bigger than”, “max”, or “if” statements, for those are not smooth/differentiable, but rather cause abrupt changes in result depending on the input.

3.4 Optimizer step

The function of the optimizer is to “take a step”. In other words, to decide what change to apply to each of the design parameters/weights at each iteration given the information provided by the gradient and other (optional) inputs. The ultimate goal is to have the lowest possible objective function evaluation.

In machine learning, the objective/loss function often calculates the discrepancy between a predicted value by the neural network and a ground truth, and usually uses batches of data. Here, the objective function is the weighted sum of the structural performance evaluation and area ratio, and only contains one example per iteration.

The simplest form of optimizer, and thus perhaps a good starting point, is called gradient descent. This optimizer always takes a step in the direction of the steepest slope downwards. The size of the step is equal to the learning rate times the gradient of the old weight with respect to the objective function. The new weight is equal to the old weight minus the step size (to go in the direction of steepest descent, not ascent), as seen in Eq. 3.1.

Eq. 3.1

$$W_{t+1} = W_t - \alpha \nabla W_t$$

Where

W_{t+1} is the weight for the next iteration

∇W_t is the gradient of W_t with respect to the objective function

α is the learning rate

However, this is likely to lead to local minima. One can visualize this by imagining that if one is on a mountain and wants to find the lowest valley point, strictly always taking a step downwards that is proportional to gradient and a static learning rate will likely lead to getting stuck in a puddle at high altitude, or taking a long time to reach the bottom. In order to remedy this pitfall, one can use stochastic gradient descent (SGD) with momentum. It considers the result of previous iterations and accelerates movement if they have been in the same direction.

Eq. 3.2

$$V_{t+1} = \beta V_t + (1 - \beta) \nabla W_t$$

$$W_{t+1} = W_t - \alpha V_{t+1}$$

$$\beta = 0.9$$

Where:

V is the velocity

W is the weight being updated

α is the learning rate

β controls how much influence the past velocity has on the current weight update, always less than one, commonly 0.9.

Even with SGD, optimization algorithms are likely to yield a local optimum instead of a global one. Since escaping from a shallow valley, over a mountain, to the next valley might be difficult even with momentum.

The drastic growth of machine learning (ML), which also uses optimizers to change the parameters of the network (e.g. weights, biases), has led to the creation of an array of optimizers. Each of them with their own strategy for looking for the highly desired and ever elusive global minimum.

One of the most popular optimizers developed for ML is Adam (Soydaner, 2020). It has been used in a wide range of applications and achieved faster convergence than SGD. It was first introduced by Kingma and Ba (2014).

Adam calculates the first and second moment of the gradients, and uses individual adaptive learning rates based on those moments (Soydaner, 2020). It is generally more successful at finding global (or at least lower local) minima than SGD. Kiyono et al. (2016) have stated that a powerful optimizer is needed to deal with the high non-linearity of stress.

In the present work, the focus has been in setting up a manufacturing-inspired and fully differentiable material distribution definition and applying automatic differentiation, so a deep dive into different optimizers has not been conducted. However, by implementing the workflow within pytorch, the application of different optimizers is straight forward, as they are included and pre-programmed in the package.

For further research, it would be very interesting to look into the intricacies of the different optimizers, hypothesise about, and test their performance on this particular problem. For example, an aspect that could play a role in choosing the ideal optimizer could be the contribution of different parameters to the objective function. If a parameter has a much larger impact on the objective function than another, it might make sense to use an optimizer that has different learning rates for different parameters, like Adam does. There might be other optimizers which would be ever better suited to the problem at hand.

For now, it suffices to know that different optimizers are available, that there is a hierarchy of performance, and to test a couple of them given the possibilities already set up in the pytorch code.

4 Optimization methods

Optimization methods are highly intertwined with their topology definition. As such, they affect the number of design variables. These variables require a gradient calculation and an optimizer update at each iteration cycle. Thus, they have a direct impact on the computational power needed to perform the optimization.

4.1 SIMP

Of the traditional methods, the most famous and widely applied is the Solid Isotropic Material with Penalization (SIMP) method. It works with a discretised domain, where each finite element is assigned a pseudo-density ranging between 0 and 1. The optimization starts with a predefined, usually evenly distributed density, and changes the pseudo density value based on the results of the sensitivity analysis of each variable in relation to the objective function, and the given constraints. It outputs a material field of values between 0 and 1 (Bendsøe & Sigmund, 2004).

In order to avoid singularities later on in the structural calculations, the smallest density is actually slightly above 0. Thus the topology is defined in grey scale with elements' material ranging from nearly zero to 1. One advantage of this is that optimization algorithms can manipulate those densities and perform sensitivity analysis to slowly inch closer to the optimized topology. A disadvantage is that intermediate densities are not an accurate representation of the physical reality, and thus optimized results using this method require some extra techniques (such as stress relaxation and intermediate density penalization) during the optimization process, as well as significant post-processing time to define clear boundaries in the field of grey-scale topology.

The penalization factor punishes output with intermediate values, but those are still unavoidable. Thus a decision needs to be made with regards to translating these intermediate densities to the binary real world of material-no material. This discrepancy between the model and reality is one of the main pitfalls of the SIMP method, and significantly adds to the post-processing time and effort required (Schoenmaker, 2023).

On the positive side, SIMP allows for any material configuration that can be represented by a combination of the elements in the mesh. So it is not constrained by any previous assumptions on what the optimized shape could be. Also, it is relatively intuitive to see how it works. The 99 line code by Sigmund has inspired and aided many optimization projects to date, and is also used to generate data for training machine learning models (Sigmund, 2001).

The optimizer's direct accessibility to each finite element's density can also create problems. These are: checkerboard patterns (i.e. when filled elements are only connected to each other through nodes), point flexures (i.e. when finite elements are only connected by each other diagonally), and floating material (i.e. when filled elements are disconnected from the supported part of the structure) (Kumar, 2017).

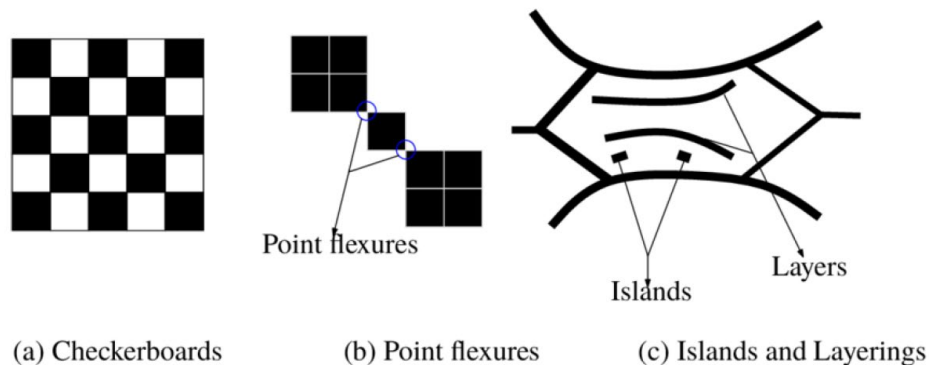


Figure 4.1: Challenges of the SIMP method (Kumar, 2017)

4.2 Indirect definition of pseudo-densities

There have been multiple attempts at defining the pseudo densities indirectly in order to avoid or minimize these drawbacks without the need for filters, or enforce desirable geometric qualities in the design. Some fall into the category of Level-Set methods, explained in Section 4.4, others will be presented in the current Section.

Guest et al. (2004) developed a methodology where the nodes instead of the elements are given a density, and those nodal densities get projected onto the finite element grid. **Figure 4.2** shows the nodes marked in green that contribute to the density definition of element e . This eliminates the possibility of the creation of the small features that are the building blocks of checkerboard patterns and allows for the definition of a minimum length of the structural member's cross section.

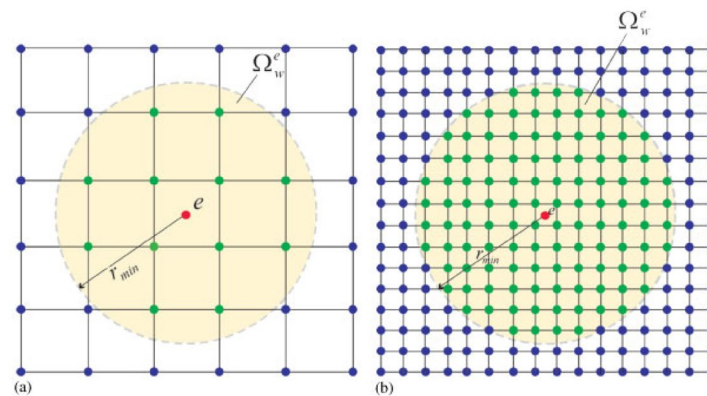


Figure 4.2: Nodes located inside the domain Ω_w^e are used in projection scheme for element e . Ω_w^e does not change as mesh (a) is refined to (b). (Guest et al., 2004)

The methodology uses a linear weight function to project the nodal weights onto the elements, so that nodes close by have a larger influence than those further away, as seen on **Figure 4.3**.

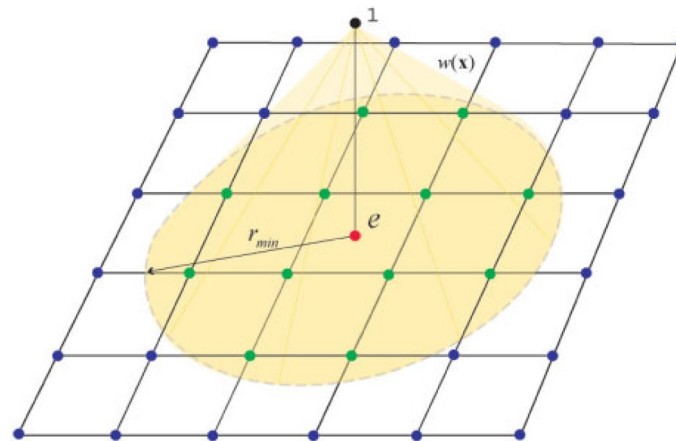


Figure 2. The weight function $w(x)$ for the linear projection scheme.

Figure 4.3: The weight function $w(x)$ for the linear projection scheme. (Guest et al., 2004)

It is interesting and innovative that in this method the design parameters have an effect on the topology that is less local than when the design parameters are the element densities.

Figure 4.4 shows optimization results with different mesh densities. The minimum length, shown in black on the figure, is kept constant and independent of the mesh. The results show that this is a successful approach for restricting the minimum thickness of the structural member. This is of special interest in the present work since one of the main challenges of working with cast glass is controlling the annealing time

which is defined by the maximum element thickness as well as the residual stresses which are defined by the ratio of largest to smallest cross sectional thickness. In the work of Guest et al. (2004), the minimum thickness is controlled, while for glass both the maximum and minimum thicknesses should be controlled. The idea of a projection function or another indirect definition of the pseudo-densities might be useful here.

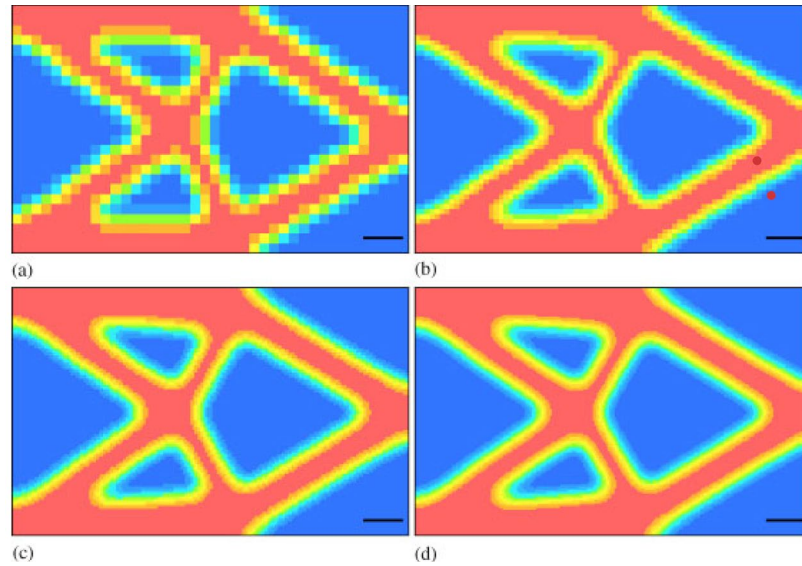


Figure 4.4: Optimal topology for different levels of mesh refinement when $d_{min} = 4$: (a) 40×25 ($d/h = 4$); (b) 80×50 ($d/h = 8$); (c) 160×100 ($d/h = 16$); and (d) 240×150 ($d/h = 24$). The black bars represent the length d_{min} . (Guest et al., 2004)

Figure 4.4 also shows intermediate values for the densities along the edges of the topology. The authors propose the use of a Heaviside function instead of a linear function (as shown in **Figure 4.5**) to create a crisper boundary edge. The Heaviside function changes value more abruptly, making the intermediate density band narrower. They use a regularized Heaviside function to ensure that gradient of the element densities with respect to the nodal densities (i.e. the design variables) are continuous.

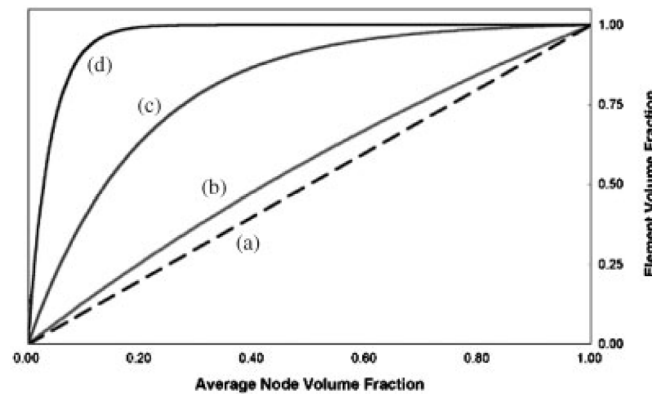


Figure 4.5: The regularized Heaviside step function for various magnitudes of β : (a) $\beta = 0$ (linear); (b) $\beta = 1$; (c) $\beta = 5$; and (d) $\beta = 25$.

Another work that aims at indirectly defining the pseudo-densities, but in a completely different manner than the aforementioned authors did, is the work by Hoyer et al. (2019). They propose the idea of Neural Reparameterization in structural optimization. It leverages the fact that machine learning is also a form of optimization (*Which neural weights yield the minimum value for the loss function?*) and apply the framework of neural networks to output the densities used for topology optimization. They use the main

ideas of SIMP, but combine it with a convolutional neural network to set the pseudo-densities of the topology. This is a hybrid SIMP-ML approach.

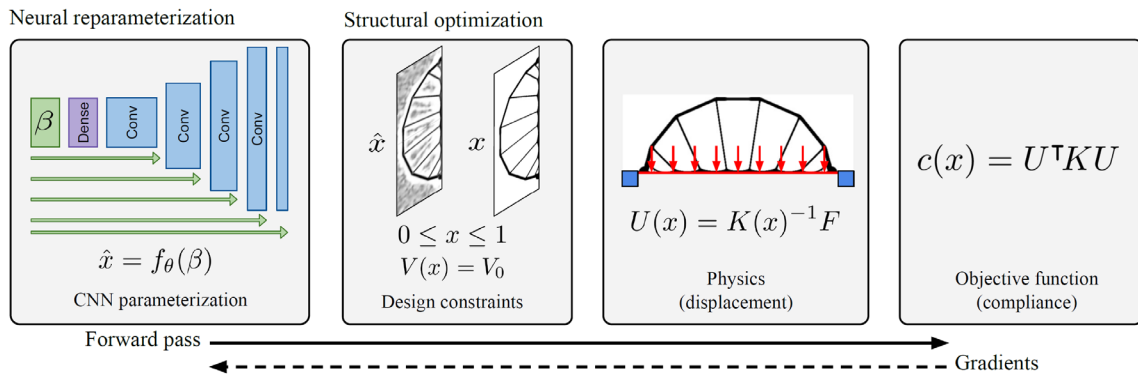


Figure 4.6: Schema of [Hoyer et al.'s (2019)] approach to reparametrizing a structural optimization problem with a neural network. Each of these steps – the CNN parameterization, the constraint step, and the physics simulation is differentiable. [They] implement the forward pass as a TensorFlow graph and compute gradients via automatic differentiation.

The core strategy of this work is to use a readily-available automatic differentiation package to write the model. This allows for the hands-off calculation of the gradients of the objective function with respect to the design variables (Hoyer et al., 2019). The objective function includes the desired structural performance criteria to be minimized. The learnt parameters are the convolutional layers in the CNN reparameterization, which was inspired by the U-net architecture, and the first activation vector β .

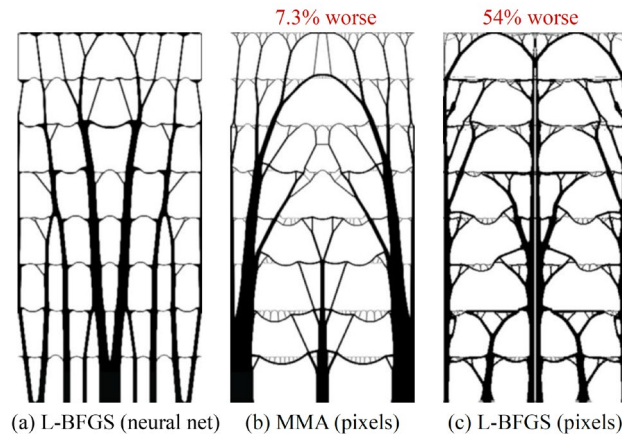


Figure 4.7: A multi-storage building task. Figure (a) is a structure optimized in CNN weight space. Figures (b) and (c) are structures optimized in pixel space.

Figure 4.7 shows that this methodology does indeed promote a clarity of design, in the sense that the structural material tends to be distributed in more discrete configuration than the optimizations performed in pixel space.

The downside of their approach was that, by training the model to optimize all the parameters in the CNN model, there were actually more design parameters than the SIMP methodology would have had for the same problem.

The applicability of these ideas to the current work could be to indeed leverage the current computational tools (such as automatic differentiation) that have been developed for the training of neural networks (NN) to the topology optimization problem.

4.3 Discrete topology

Some methodologies avoid the drawbacks from intermediate densities by relying on a discrete topology definition. Some of such approaches fall into the Level-Set category, and are explained in Section 4.4.

Carstensen et al. (2023) have proposed a frame-based, Mixed Integer Linear Program (MILP) topology optimization problem. It uses a ground structure of user-defined elements that the optimization algorithm can pick from to give form to the optimized topology. This ground structure creates restrictions on the available parts that can be used as components to the final structure. This can be a positive feature if those restrictions reflect inherent manufacturing constraints. Although restrictions can be positive, the ground structure should still be dense enough to allow for enough variation, and thus the composition of a high-performing structural element. The need to find a balance between restriction and freedom manifests in different ways depending on the chosen method, but to the best of the author's knowledge, it is present in all of them.

The design variable in this case includes a binary vector that states either the absence or the presence of each component of the ground structure in the topology. **Figure 4.8** shows a possible configuration of a ground mesh that includes both frame and plate elements.

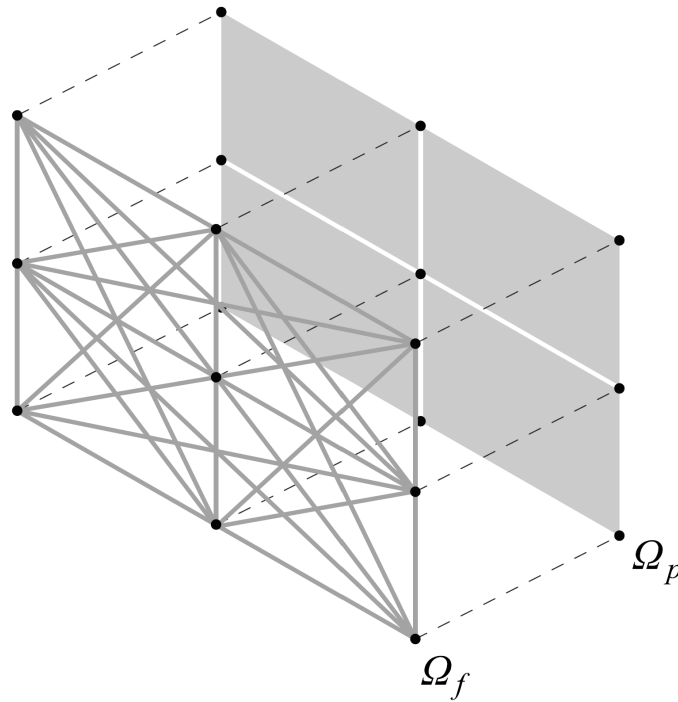


Figure 4.8: Illustration of how a design domain Ω is decomposed into two parts $\Omega = \Omega_f \cup \Omega_p$ when using a hybrid ground structure mesh. Potential frame elements in Ω_f are discretely selected from the user-specified set of cross-sections S_f , whereas quadrilateral thicknesses are selected from S_p . In [their] work, nodes of the frame elements coincide with quadrilateral node locations. (Carstensen et al., 2023)

Their project is of interest in the context of the present thesis because it also has a heavy focus on creating a TO method that is specific to a manufacturing process. Their goal is to create a topology that can be materialized using Additive Manufacturing (AM), whilst the goal of the present work is to produce a topology that can be cast in glass. Thus while the manufacturing methods and constraints differ, the focus on embedding the manufacturing constraint into the optimization process itself is present in both works.

The idea of a parameter that chooses from a set of available thicknesses could be useful for enforcing the requirement for minimum and maximum dimensions in the cast glass structure that originate from the annealing, residual stress constraints, and the fact that very thin cross sections are undesirable for being too fragile.

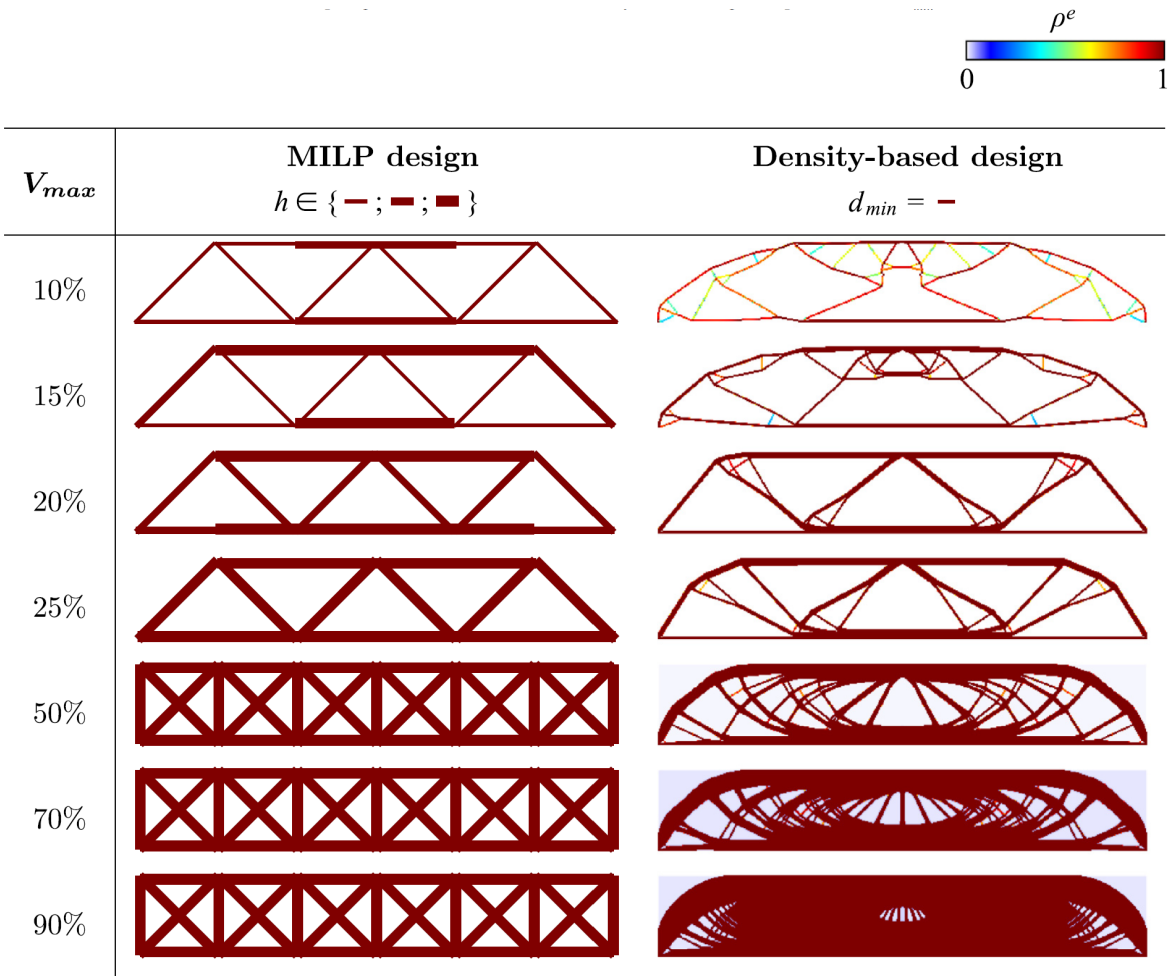


Table 4.1: Comparison of simply supported beams design by solving the MILP [methodology proposed in Carstensen et al., (2023)] and by solving the density-based problem (...). The MILP designs are obtained on the ground structure (...) and the available set of heights for the rectangular cross-section is $h \in \{0.8; 1.6; 2.4\}$ mm. The minimum topological feature size in the density-based designs is prescribed as $d_{min} = 0.80$ mm. (Carstensen et al., 2023)

From the results presented on **Table 4.1**, one can see the effect of the methodology on the topologies created. While the work of Carstensen et al. (2023) ensures the creation of a topology with sharp, angular transitions, the present work searches for a methodology that encourages rounded, smooth transitions instead.

4.4 Level-Set

In general, the Level Set Method (LSM) works by comparing the value of a 3-D function with a constant, c (Osher & Sethian, 1988). Visually, this looks like slicing the three dimensional function (i.e. the Level Set Function (LSF)) with a horizontal plane, which is usually at $z = 0$, as seen in **Figure 4.9**. The boundary of the topology is defined by the intersection of the two (Osher & Fedkiw, 2001). It is the iso-contour of the LSF on the level-set plane (Van Dijk et al., 2013), defined by the formulas:

$$\begin{aligned}
\phi(x) &> c, & x &\in \Omega \text{ (material)} \\
\phi(x) &= c, & x &\in \Gamma \text{ (boundary)} \\
\phi(x) &< c, & x &\notin \Omega \text{ (void)}
\end{aligned}$$

Where:

ϕ is a LSF

x is a point in the design domain

c is a constant (usually $c = 0$)

This allows for an unambiguous (i.e. clear boundaries, no intermediate densities as in SIMP) and mesh-independent topology definition. The design parameters are those used in the definition of the LSF. That is, changing the LSF can change the topology, as seen on **Figure 4.9**. There are many variations of this approach.

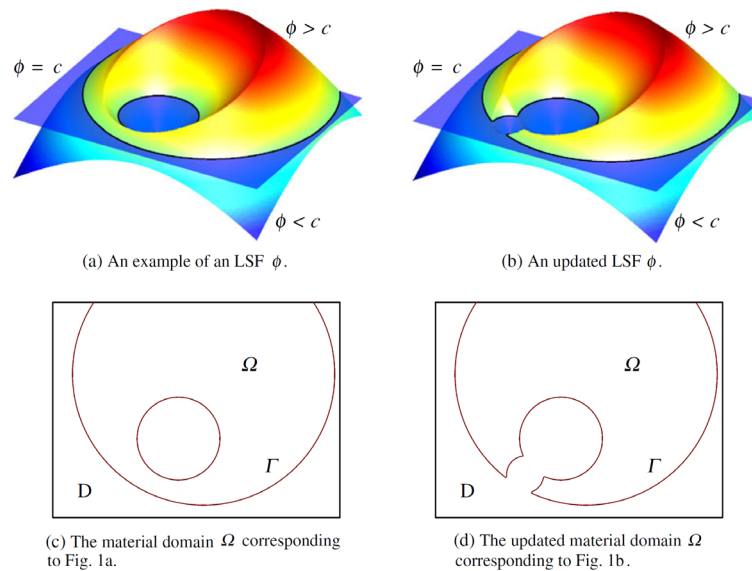


Figure 4.9: An example of the LSF ϕ and corresponding material domain Ω before and after a design update (Van Dijk et al., 2013).

In their review paper, Van Dijk et al. (2013) categorized different Level-Set methods according to their choice of: (1) function parameterization, (2) geometry mapping, (3) update information, (4) update procedure, (5) applied regularization. The information they collected is presented in the following subsections.

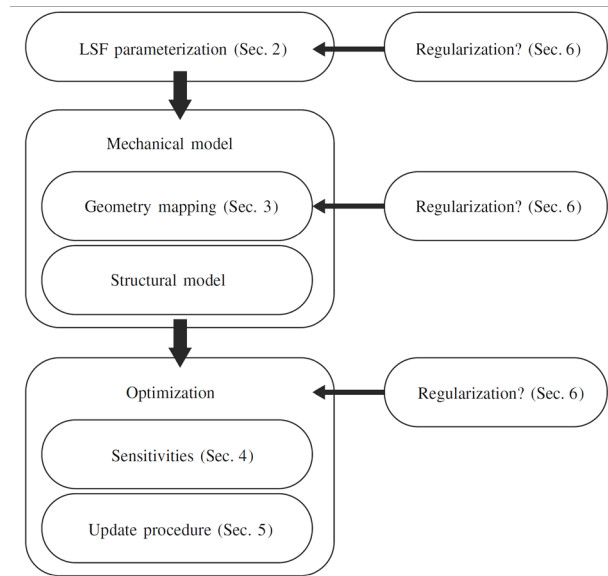


Figure 4.10: The components of a level-set-based TO method and corresponding sections (Van Dijk et al., 2013)

4.4.1 Function parameterization

A LSF, in its most general form is given by

$$\phi = \phi(X, \psi(X, s))$$

It is defined by an auxiliary function ψ that takes as input the design parameters \mathbf{s} , as well as the spatial coordinates \mathbf{X} (Van Dijk et al., 2013). The spatial coordinates are the discretization of the LSF (i.e. the points at which its values are calculated and compared to c). The design parameters \mathbf{s} are independent of the discretization \mathbf{X} .

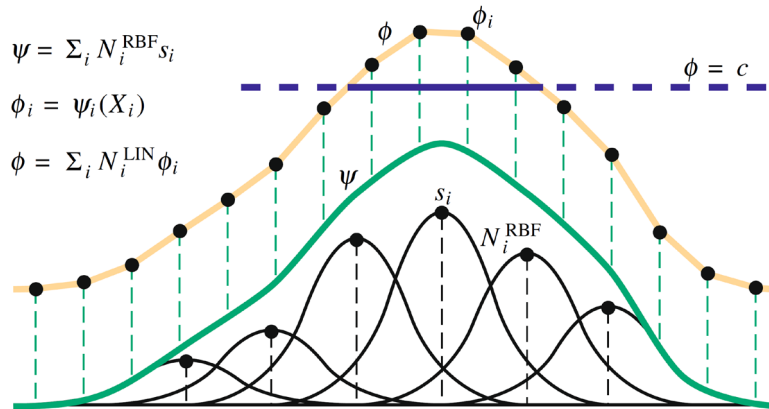


Figure 4.11: An example of a one-dimensional LSF ϕ discretized with linear shape functions of which the nodal values ϕ_i depend on an auxiliary field ψ . The auxiliary field ψ is discretised with RBFs and controlled by the optimization variables s_i (Van Dijk et al., 2013).

The auxiliary function is often a sum of a type of basis function:

$$\phi(X, s) = \sum_i^n \phi_i(X, s_i)$$

There are two most common ways to use the design variables \mathbf{s} to define the basis functions: scaling and translating (Van Dijk et al., 2013), as seen in **Figure 4.12**. Scaling can be disadvantageous because it gives less control over the spatial gradient (i.e. the slope at the topological boundary). Translating uses

the design variables to define the location of the basis functions. It can provide some control over the length scale of the results and spatial gradient. However, it can also lead to some underpopulated areas. Additionally, swapping two functions would give the same result, and this made lead to convergence issues (ibid).

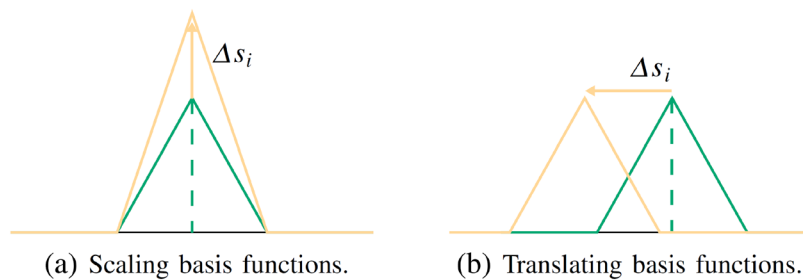


Figure 4.12: Examples of possible relationships between the design variables and the basis function parameters [own title] (Van Dijk et al., 2013)

These basis functions can take many forms, including Radial Basis Functions (RBFs). RBFs have a vertical symmetry axis at their core, and their value depends solely on the distance of the point in question to this axis.

“The choice of parameterization determines the kind of shapes (design freedom) and the level of detailedness (smallest features) of the zero-level contours of the LSF and may restrict the design space to prevent numerical artifacts caused by the spatial resolution of the discretised structural model. The choice of parameterization of the LSF directly influences the nature of the optimization problem, i.e., the nonlinearity and or/monotonicity and the dimensionality of the optimization problem, and therefore the effort needed to solve the optimization problem.” (Van Dijk et al., 2013, p. 441)

The quote above highlights the how impactful the choice of parameterization is in the process as well as outcome. From it, one can extract some important factors to consider when choosing a specific parameterization:

- 1) What kinds of *shapes* are desired in the design? This could be of special interest in the present work in terms of the manufacturing constraints of cast glass.
- 2) What could be the smallest feature of the design? i.e. What could be the building blocks of the topology?
- 3) Is it possible to reduce the dimensionality (i.e. the number of design parameters) of the problem in order to decrease the computational burden, while still providing enough design freedom to the algorithm? If so, how?

Van Dijk et al. (2013) also point out that, on top of the parameterization of the LSF itself, the discretization also has a great effect on the outcome. Discretization refers to the points at which the LSF is evaluated and eventually translated into a material distribution (that is used in the topological and structural calculations used in the objective function). Therefore, discretization affects the accuracy of the structural calculations and of the mesh used for calculating area/volume. This accuracy must be balanced against the efficiency of the computational calculations. They advocate for decoupling the parameterization of the LSF from the discretization of the structural model, so that the discretization can be adapted according to that balance between accuracy and efficiency (ibid).

Another great insight presented in the same review paper refers to the impact of the steepness of the slope of the LSF along the material boundary (Van Dijk et al., 2013). It affects the sensitivity of the design to changes in the LSF, and the convergence rate of the algorithm. In cases where a pseudo density field is being created by the discretization of the LSF, the slope also affects the thickness of the intermediate density band along the material boundary. (This is similar to the observations made by Guest et al. (2004),

who opted for the use of a Heaviside function to decrease the intermediate density band thickness.) Van Dijk et al. (2013) suggest that, for the reasons mentioned above, control over the spatial gradient (a.k.a. “steepness”) at the material boundary is desired.

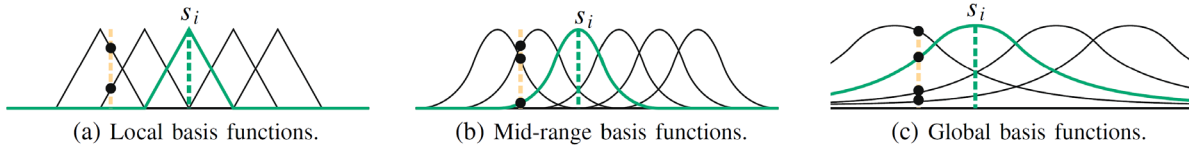


Figure 4.13: Examples of different support sizes of basis functions. The *black dots* indicate how many basis functions are non-zero at an arbitrary location (Van Dijk et al., 2013).

Going back to the idea of the LSF being defined by an auxiliary function ψ , which is itself a sum of many base functions, **Figure 4.13** shows three types of basis functions: (a) local, (b) mid-range, and (c) global. Their names refer to the domain over which they are non-zero, and thus also the likelihood of mutual overlap.

Global basis functions are non-zero everywhere, and thus all overlap with each other. This implies that *all* of the design variables have an effect on the entire design domain. Given the high complexity of this interaction, a larger computational burden can be created when compared to local functions (Van Dijk et al., 2013). On the other hand, having a large influence on the design domain can also mean that all design variables have sensitivity information (i.e. the result changes with respect to their change) and this can lead to faster design evolution (ibid).

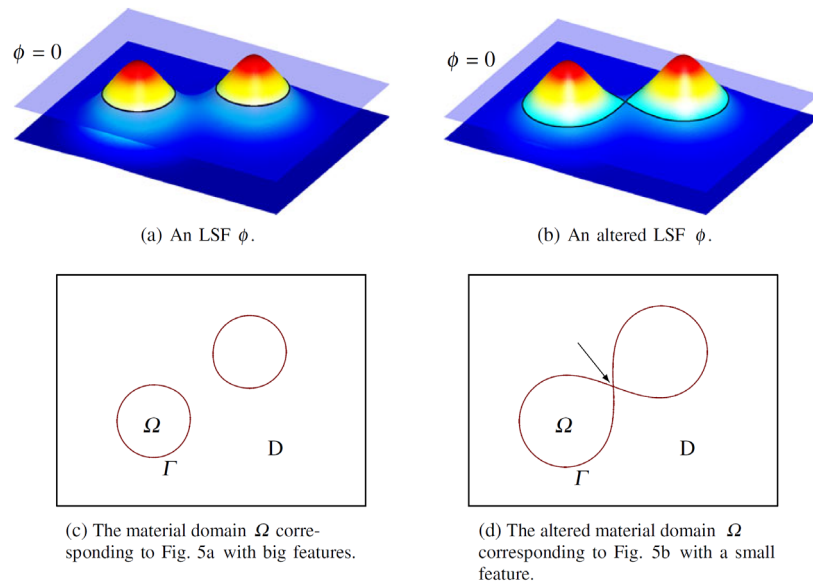


Figure 4.14: An LSF ϕ of just two Gaussian basis functions with a large support size and the corresponding material domain Ω (Van Dijk et al., 2013)

Figure 4.14 shows the features created by using Gaussians as basis functions. Gaussians are global RBFs, and capable of creating big as well as small features.

4.4.2 Geometry mapping

In order to perform the structural calculations, the LSF has to be mapped onto a discretised structural model (assuming the application of the Finite Element Method). The translation of the LSF to a material distribution can create either a discrete (strictly material or no-material) topology or a pseudo-density material distribution (Van Dijk et al., 2013).

Figure 4.15 gives an indication of the different types of information that can be used by the structural model to evaluate the design. For stress-based optimization, where small changes in the boundary make a big difference, a more accurate mesh is desirable (Van Dijk et al., 2013).

However, using a structured and uniform grid, which remains constant at every optimization iteration, can be computationally advantageous (Van Dijk et al., 2013). This is because some information can be computed during pre-processing, outside the optimization loop, such as the shape functions of the finite elements, the connectivity matrix, and even the *indexes* of contribution of each node of each finite element to the global stiffness matrix (the value of the contribution still has to be iteratively computed as it depends on the material distribution). It does require the approximation of the curvatures to a structured scheme, which by definition means some loss of accuracy, but the advantage is that a new mesh does not need to be calculated each time. Keeping the mesh constant also provides consistency of the baseline of the problem (i.e. the structural results are only changing because the topology is changing, not because the size and/or shape of the finite elements are changing as well).

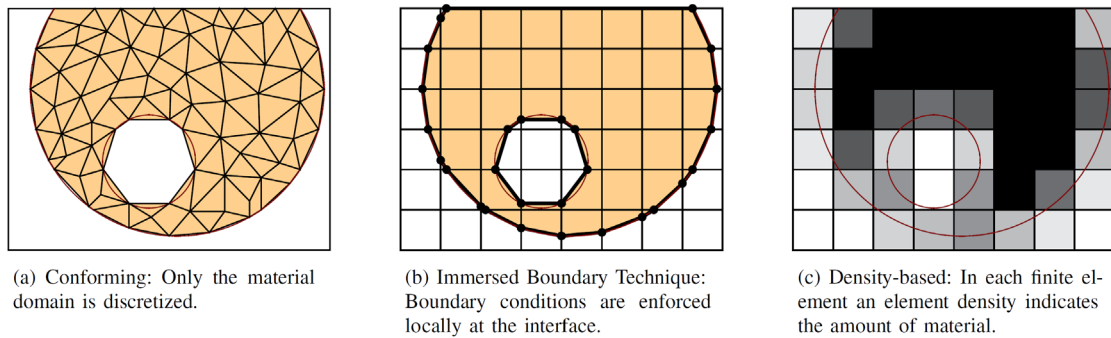


Figure 4.15: Examples of different types of geometry mapping corresponding to the topology depicted in **Figure 4.9** (Van Dijk et al., 2013)

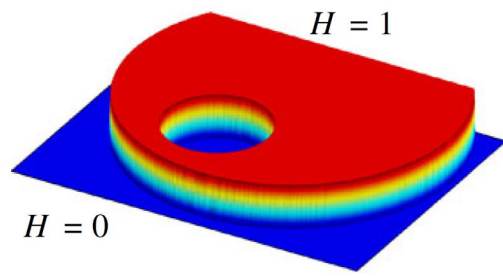
When using a pseudo density, the choice still needs to be made about how exactly the LSF is to be translated into a density field. One could create a sharp boundary by testing where the LSF is larger than c and assigning material, and where it is smaller than c , assigning no material. This is done using the exact Heaviside function, as seen in **Figure 4.16 (a)** and **(c)**. In this case, intermediate densities are only present where the isocurve cuts through a finite element. Although advantageous in terms of accuracy, the problem is that such a binary split is non-differentiable, and optimizations normally rely on gradient calculations to effectively change a design. Given this challenge, one possible solution is to use an approximate Heaviside function to create a gradual (and thus differentiable) intersection of material and no-material, as seen in **Figure 4.16 (b)** and **(d)**. Here, the intermediate density band is wider than in the exact Heaviside application, but the material distribution definition is differentiable.

Point-wise mapping is the most simple form of establishing the density of a finite element given the LSF and an additional, differentiable function, such as the aforementioned approximate Heaviside function. It is given by the formula (Van Dijk et al., 2013):

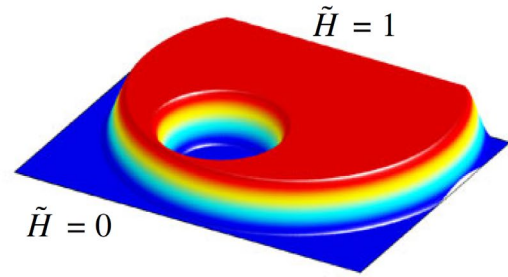
$$\rho_e = (1 - \varepsilon)\tilde{H}(\phi(X_e))$$

Where X_e is the centre of finite element e .

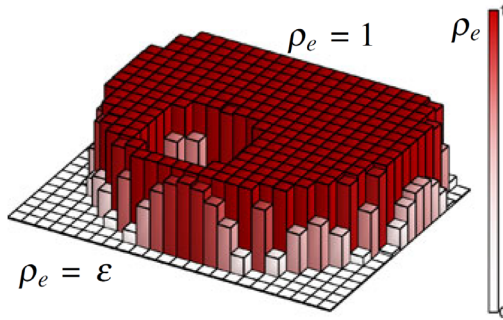
Going back to the importance of the spatial gradient along the material boundary, it is worthwhile to mention Van Dijk et al.'s (2013, p. 450) emphasis that “the size of the band of intermediate densities depends on the bandwidth h of the approximate Heaviside function and the local gradient of the LSF in the neighbourhood of the boundary.” That is to say that *both* (1) the gradient of the LSF at the boundary, as well as (2) the gradient of the function used to translate the LSF into a material distribution will affect the intermediate density band thickness.



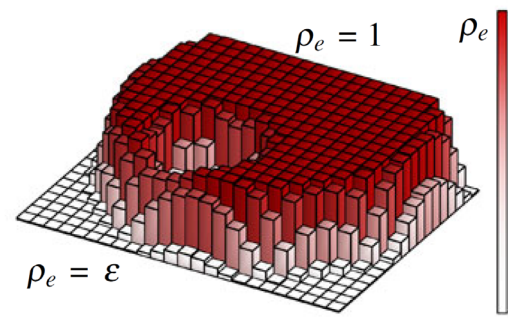
(a) The exact Heaviside function $H(\phi)$ corresponding to Fig. 1a.



(b) An approximate Heaviside function $\tilde{H}(\phi)$ corresponding to Fig. 1a.



(c) Element densities ρ_e corresponding to the exact Heaviside function $H(\phi)$ of Fig. 10a.



(d) Element densities ρ_e corresponding to the approximate Heaviside function $\tilde{H}(\phi)$ of Fig. 10b.

Figure 4.16: Geometry mapping a LSF to a material distribution (Van Dijk et al., 2013)

Yulin & Xiaoming (2004) explore the idea of using multiple level sets to define their material distribution in the design domain. Their goal is to offer a multi-material level-set approach. They achieve this by coupling each material to a different level set.

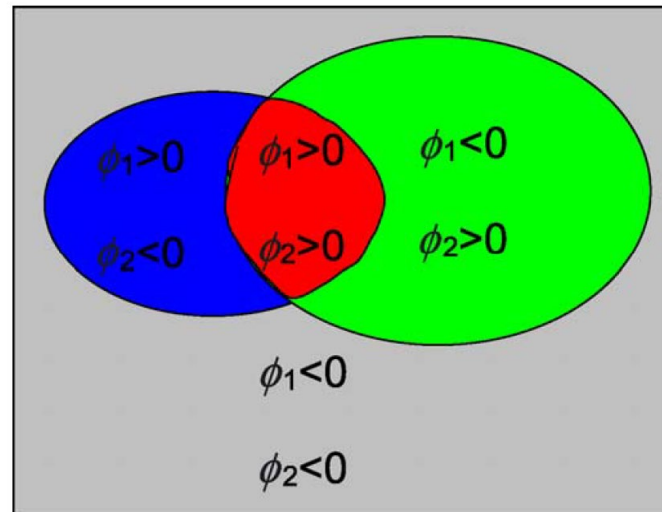


Figure 4.17: Vector level set representation model I of multi-materials (Yulin & Xiaoming, 2004)

4.4.3 Update information and process

In the context of Level-Set methodologies, the shape of the topology is what exerts influence on its structural performance. Many methodologies within the Level-Set approach calculate the sensitivity with respect to the boundary of the topology, i.e. shape sensitivity. In this case, the update procedure occurs through pseudo-time, and a pseudo-velocity of the boundary, which affects its change, is calculated.

$$\mathbf{X}_\tau = \mathbf{X} + \tau v_n \mathbf{n}$$

Where

\mathbf{n} is the normal vector to the boundary

τ is the pseudo-time, i.e. the design iterations

v_n is the pseudo-velocity field, which replaces the boundary

Alternatively, if a pseudo-density model is used, where the LSF is translated to a material distribution with the aid of a differentiable function (such as the approximate Heaviside function discussed previously), then one can compute material parameter sensitivities using the Chain-Rule (Van Dijk et al., 2013):

$$\frac{\partial R}{\partial s_i} = \sum_e \frac{\partial R}{\partial \rho_e} \frac{\partial \rho_e}{\partial s_i}$$

Where

R is the response function (a.k.a. objective function)

s_i are the design variables

ρ_e are the element pseudo densities

This entirely skips the need to calculate the exact boundaries, or any sensitivities related to it. The sensitivity is calculated between the design parameters \mathbf{s} and the structural performance present in the objective function, passing through the LSF, the geometry mapping, structural calculations, and any manipulations done to condense the results into a scalar objective.

According to Van Dijk et al. (2013), there are two classes of update procedures used in Level-Set methodologies, which are closely linked to the two classes of update information explained above. The first is the pseudo-temporal, and relies on the definition of sensitivities in relationship to the boundary, the direction and magnitude of the change is conceptualized as a pseudo-velocity towards steepest descent. The second relies on mathematical programming and can also apply the steepest descent method (SDM) but in this case the sensitivity used is of the pseudo-density in relation to the objective function.

4.4.4 Applied regularization

Regularization can be used to “control geometric properties of the resulting designs”, “obtain a well-posed optimization problem”, or “avoid convergence to sub-optimal local minima” (Van Dijk et al., 2013, p. 438).

Regularization can be used anywhere in the LS process. For example, one of the areas where regularization can be useful is in ensuring an even spatial gradient along the contour of the topology. For more information, refer to the work of Van Dijk et al. (2013).

As a final remark on the Level-Set Method, Van Dijk et al. (2013) point out that this method has a high dependency on the initial guess, and is likely to converge to local minima. Thus, if used in the optimization process, different starting configurations should be tested and compared.

4.5 (Bidirectional) Evolutionary Structural Optimization

Evolutionary Structural Optimization (ESO) works intuitively by performing structural calculations, and then progressively removing material where it is less used until the removal of more material would cause the structure to go beyond the allowed constraints (Xie & Steven, 1997). Bidirectional Evolutionary Structural Optimization (BESO) was proposed as improvement to ESO that can also add material as well as remove it (Querín et al., 1998).

5 Structural calculations

A Finite Element Analysis (FEA) is conducted in each loop of the optimization process to evaluate the structural performance of the current material distribution state. The results of the structural analysis are then processed and used in the objective function to quantify the structural performance. This chapter offers a walk-through the Finite Element Method (FEM).

5.1 Finite Element Analysis

The FEM is based on discretization of the design domain: splitting up the continuum of the structural element into small segments called finite elements (FE). This makes it possible to calculate the general behaviour of the structural element through equilibrium at the nodes by considering the global stiffness matrix (K), and interpolation of the nodes to deduce the behaviour of the area between them through the shape function. This introduces a degree of inaccuracy. The size and shape of the elements represent the trade-off between accuracy and computational power. The smaller and more specific the shape of each element, the larger the accuracy of the structural calculation and also the computational cost.

FEA consists of a series of steps. The first one is the problem definition. This includes defining the material (including Young's Modulus and Poisson's ratio), loads, supports, and overall dimensions. The data needed here is collected from Chapter 2 on Glass and Chapter 7 on the Case Study.

The second decision to be made when applying the FEM is on the size and type of finite elements during discretization (Chandrupatla, 2002). There are line elements in one dimension. These are most suited to truss analysis. In two dimensions there are triangular and square elements. These can have nodes just at the corners or double that amount of nodes by introducing extra nodes between the corners. Elements can also vary in shape and be defined iteratively according to the topology at each step of the optimization. However, more computational power is then needed to compute the discretization each time, and its corresponding shape functions, element stiffness matrix, connectivity matrix, and global stiffness matrix. In three dimensions, similar versions to the two dimensions are possible: pyramids and cubes, with each nodes just at the corners, or also in the middle of each edge.

For this project, four node square elements have been chosen for simplicity. Keeping the optimization process in 2 dimensions gives more hope of achieving the overall goal of a stress-based optimization which is already quite computationally intensive. While keeping the elements square create a good similarity between the quantified structural characteristics of each element and the values of pixels in a picture. This analogy could be used to train a machine learning algorithm, if that turns out to be the chosen optimization method.

The FEM is derived from the principal of minimum potential energy, which states that equilibrium is found at the configuration where the potential energy is minimized. The variational method using Galerkin's method of weighted residuals leads to the calculation of the element stiffness matrix $[k]$. This matrix is square, with dimensions equal to the degrees of freedom of the element. For the four nodes square element, that is a total of 8 degrees of freedom, 2 degrees of freedom (x and y) on each of the four nodes.

The Shape Functions interpolate the behaviour of the nodes to define the field inside the elements (Chandrupatla, 2002). They are defined in natural coordinates. The shape functions define the contribution of each node to the field in between them. It is defined by equating the contribution of the node for which the function is being defined to 1, when the point being evaluated is also at that same point, and the contribution of all other functions (regarding the other three nodes) to zero.

The connectivity matrix has dimensions *number of elements* by *number of nodes per element*. It states, for each element, which four global nodes it is connected to. The connectivity matrix is used to specify which elements' stiffness contribute to which global degrees of freedom.

The global stiffness matrix (K) is assembled by adding the stiffness contribution of each element to the degrees of freedom of the nodes that they are connected to (Chandrupatla, 2002). The stiffness of each element is multiplied by its pseudo-density, to reflect the current material distribution. Finite elements with material have a pseudo-density close to one, and thus retain (close to) their full stiffness, while “empty” finite elements have a pseudo-density close to (but not equal to) zero, thus reflecting the fact that they are impaired for carrying loads without large deformations and stresses. A value of zero may not be used, because then the displacement would be infinite, and so a very low value for the minimum density is chosen.

The global stiffness matrix is square with dimensions of the *global number of degrees of freedom*. Continuity requires that the displacement of a node be the same for all the elements that connect to that node. The global stiffness matrix (K) is sparse, since most elements are not connected to each other, and banded since the degrees of freedom present at the beginning of the structural element (wherever that is defined) tend not to be connected to the degrees of freedom at the end of the element.

Once all of this is set up, one has the K of the equation $F = KU$.

The next step is calculating F . F is a vector of the nodal forces (for each degree of freedom of each node). The load applied to the structure is modelled here. Everywhere a load is applied gets the value of the load, all others get the value of zero. In this project, the load on each node is proportional to the pseudo-densities of the four finite elements attached to it.

U is the vector of displacements at each degree of freedom. Some displacements are set to zero to represent the supports. All other displacements are calculated by solving the equation $F = KU$ for U . In this project, $U = K \backslash F$ is used.

Displacement is then used to solve for stress, by combining it with the displacement-strain matrix and material matrix. For more information, refer to Chandrupatla (2002) for general applications, and the python code provided with this report for this specific application.

5.2 Fine tuning the structural model

The structural calculations should be performed as efficiently as possible to decrease the computational burden in terms of both time and computational power needed. In order to achieve this, some measures are taken. Firstly, the global stiffness matrix is assembled without loops, as it was done by Schoenmaker (2023). Additionally, all required calculations (such as the indexes of the stiffness matrix needed for its assembly) are done outside of the optimization loop.

Every structural model is a simplification of the reality it aims to represent, and thus not infinitely accurate. For reasonable accuracy of the structural model here, the stress is measured at the Gauss points are used, as in Schoenmaker (2023).

In previous explorations of this case study, the load is assumed to be that of the entire domain during the entire optimization and applied together with the temporary load evenly throughout the whole structure. In this project, the load applied on the structure is updated at every iteration to reflect the current material distribution. This load is also applied at the nodes corresponding to where the material is found. With this setup, the values used in the objective and constraints of the optimization algorithm come closer to representing reality.

A validation of the FEA with a full-domain can be found on Appendix 14.1. To ensure that the final optimized topology does not exceed the stress limits, it should be checked using ANSYS.

6 Stress

This project is a stress-based topology optimization. But, what is stress exactly? Stress is the distribution of *internal* forces. As such, its units are force per area. In standard units this is Newton per meter squared [N/m^2], or the unit of pressure: Pascal [Pa].

There are two types of stress: normal stress and shear stress. Normal stress exists in the material parallel to the force at that point. It either stretches (tension) or compresses (compression) the material. It is calculated by dividing the force [N] by the cross sectional area where it is applied [m^2]. Shear stress is present in a cross section if there is a force (component) perpendicular to it.

Principal stresses are the stresses on the finite element obtained when the element is rotated such that there are no shear stresses. Principal stresses offer a way of evaluating the stress present in a finite element condensed to just two values (for a 2 dimensional element): the first principal stress and the second principal stress.

By convention, tensile stresses are represented with positive signs, and compressive stresses are represented with negative signs. Thus if an element has a tensile stress in one direction and a compressive stress in the perpendicular direction, then the first principal stress will be the tensile one, and the second principal stress is the compressive one. However, it is also possible for finite elements to be experiencing either compression or tension in both directions, in which case both the first and second principal stresses would be found to one of the sides of the shear axis in a Mohr's circle.

6.1 Why is stress important in the specific case of structural glass elements?

Structural glass differs from other glass applications in the sense that its failure – almost by definition – causes damage beyond itself. Thus, it is the task of the designer and engineer together to ensure that the final architectural expression of the element does not come at the cost of its structural integrity. In other words, the structure should be within the safety limits. To ensure this, particular attention should be paid to the weaknesses of the material. In this case, the brittle nature of glass.

If a massive glass component breaks, it has to be manufactured from scratch again – assuming that solutions such as glueing do not measure up to the structural, visual, and ideological standards of a fully glass-made structure. Thus it might be more efficient to prevent failure by using some more material, than to reduce the volume while stress concentration areas which might lead to failure are still present in the design.

If a glass element cracks, the most likely origin is tensile failure. Tensile stress can be exacerbated by manufacturing defects (such as an air bubble), surface irregularities (such as a scratch), or unexpected localized loads. With so many possible contributions to the weakness of glass, it could be interesting to include the minimization of predictable tensile stresses in the optimization process. Designing an element that reduces the possibility of tensile failure is the ultimate goal of this project.

6.2 Stress criteria

A particular structural element can be deemed unsuitable for the proposed task due to a number of failure criteria, including experiencing a deformation that is too large (impeding its usability) or lack of redundancy (which can be remedied by the inclusion of appropriate safety factors). Stress criteria are used to definite the point of failure due to stress. The ultimate strength of materials are tested in laboratories and the findings are used to inform structural engineering decisions for untested specimen and configurations.

In order to assess the goal of minimizing stress, a stress criterium can be useful. Stress criteria combine the stress values in one element into a single quantity. This means that, per finite element, there is only one parameter to be considered in the optimization. This Section explores some alternatives, with a specific focus on whether or not they are applicable to brittle materials.

6.2.1 Rankine

Rankine is the most simple of the stress criteria. The material has failed when the ultimate stress is exceeded (The Efficient Engineer, 2021). The plane stress field shows the yield surface for the Rankine criteria with respect to the principal stresses.

This method could be applied to materials with different ultimate strengths in tension and compression by shifting the threshold as seen on the figure.

Although at first glance just considering the ultimate strength might seem reasonable, this criteria is overly simplistic and does not reflect experimental data for specimen in situations other than simple compressive/tensile tests.

6.2.2 Von Mises

The Von Mises stress criteria is the most widely applied in topology optimization workflows. It is relatively easy to calculate and represents experimental data well for ductile materials.

However, it is only applicable to ductile materials (Chandrupatla, 2002), not brittle ones, making it less suitable to the project at hand. This is because it only includes the *differences* between the principal stresses in its calculation, and thus ignores hydrostatic stresses. Most commercial structural software use this stress criteria, and are thus also unsuitable for glass optimization.

6.2.3 (Modified) Coulomb-Mohr

Coulomb-Mohr is a stress criteria specifically made for brittle materials, as it takes into account their sensitivity to hydrostatic stresses (i.e. stresses that attempt to change the volume, but not the shape of an element) (The Efficient Engineer, 2021).

Modified Coulomb-Mohr offers a better representation of experimental data.

6.2.4 Drucker-Prager criteria

Drucker-Prager criteria originates from the study of soil mechanics (Drucker & Prager, 1952) . It provides a smooth boundary of acceptable stress states. It offers extra penalization for tension and greater allowance for compression. According to Labuz et al. (2018), who conducted a literature review on failure theories for rock (i.e. brittle material), rounded failure envelopes are a better representation than straight lines for a large normal stress range.

Drucker-Prager criteria has been used in the previous glass optimization projects (Koniari, 2022; Schoenmaker, 2023), as proposed by Duysinx (2012a). The equivalent stress is calculated by the formula:

$$\sigma^{eq} = \frac{s+1}{2s} \sqrt{3J_{2D}} + \frac{s-1}{2s} I_1$$

Where

s is the asymmetry ratio, given by $s = \frac{\sigma_{comp,lim}}{\sigma_{ten,lim}}$

I_1 is the first stress invariant, given by $I_1 = \sigma_{11} + \sigma_{22}$

J_{2D} is the second deviatoric stress invariant, given by $3J_{2D} = \sigma_{11}^2 + \sigma_{22}^2 - \sigma_{11}\sigma_{22} + 3\sigma_{12}^2$

Using principal stresses, it becomes:

$$\sigma^{eq} = \frac{s+1}{2s} \sqrt{\sigma_{11}^2 + \sigma_{22}^2 - \sigma_{11}\sigma_{22}} + \frac{s-1}{2s} (\sigma_{11} + \sigma_{22})$$

To use Drucker-Prager as a constraint, the equivalent stress, σ^{eq} , should be smaller than the tensile limit stress, $\sigma_{ten,lim}$.

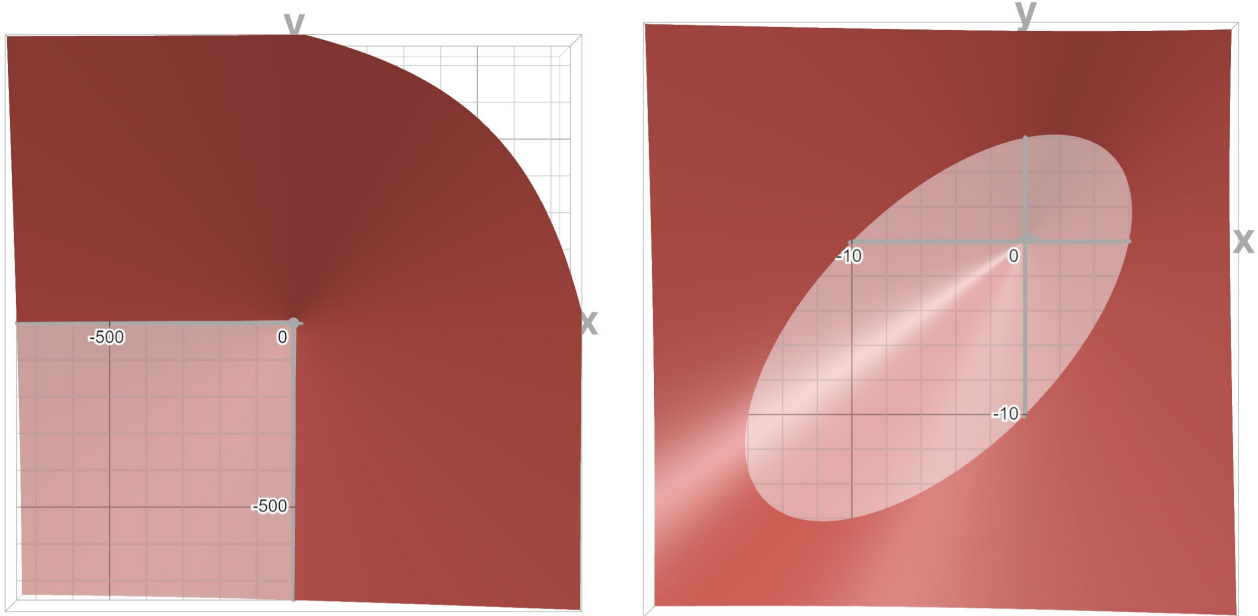


Figure 5.2: The Drucker-Prager stress criteria in 2-D, where the x and y axis represent the principal stresses. Left: compressive and tensile strengths applied in the present work (500 and 6.8 MPa, respectively). Right: compressive strength of 10 and tensile strength of 6 MPa.

The criteria crosses both axis at each of the strength limits. Notice that the shape created by the stress constraint based on this criteria is different when the compressive strength is much higher than the tensile strength. It is no longer oval (right), but open ended on the third quadrant (left).

6.3 Challenges and possible solutions

The three main challenges to stress-based topology optimization are: (1) the singularity problem, (2) the local nature of stress, and (3) non-linearity (Le et al., 2010).

6.3.1 Singularity

Singularity in the context of stress-based optimization means that the optimization tends to converge to local minima. One of the approaches to remedy this is the relaxation of the stress constraint, such as using ϵ -relaxation (Bruggi & Duysinx, 2012a; Duysinx & Bendsøe, 1998). This problem is related to methods using pseudo-density. It arises out of the fact that regions with low density are modelled with a low Young's Modulus and thus calculated to experience very high stresses. Meanwhile, regions with no material should not be considered at all for the stress constraint. There is a discontinuity arising from low density regions to no material regions, which create degenerate spaces in the optimization process (Duysinx & Sigmund, 1998). This causes the algorithm to be unable to either create or remove holes.

In order to apply the FEM, all finite elements must have a pseudo-density above zero (otherwise the displacement of the nodes dependent on that element would be infinite). The lower bound of this density

is set as a very low number, approaching zero, the upper bound is one. Thus if a method is used where all finite elements remain the same through the optimization, but their pseudo-densities change (ranging between the minimum density and one), then some stress relaxation is needed.

One form of relaxation that has been used in conjunction with the Drucker-Prager criteria is the p - q approach (Bruggi & Duysinx, 2012):

$$\langle \sigma_{ij} \rangle = \frac{\sigma_{ij}}{\rho_e^q}, \text{ where } q > 1$$

$$\langle \sigma_e^{eq} \rangle = \rho_e^{(p-q)} \bar{\sigma}_e^{eq}(s)$$

In their paper, they have used $q = 2.8$ and $p = 3.0$.

6.3.2 Non-linearity

The second problem refers to the non-linear relationship between the stress constraints and the design (Le et al., 2010). Stress states are highly dependent on a small region of the design directly surrounding the point in question. Small, local changes in design can have a large impact on the maximum stress value. This challenge seems unavoidable and is part of the reason why stress-based optimizations are computationally more expensive than their compliance and volume alternatives.

6.3.3 Local nature

The third challenge is that stress is different at each point in the continuum of the structure. The struggle arises in how to consider this multitude of values in the optimization process. When using the finite element method, the stress values can be made finite by using a stress criteria (such as the ones described above) to condense the stresses within the element into one quantity. Nonetheless, there are still then as many constraints as there are finite elements. In order to decrease the amount of parameters considered in the optimization, a global value for stress can be introduced. This global quantity combines the equivalent stresses of all finite elements into one value, such as the p -norm (Duysinx & Sigmund, 1998).

An alternative could have been to just use the maximum stress value found in the finite elements, but the $\max()$ function is non-differentiable, and thus a smoother function is more useful for optimization (Le et al., 2010). Two options that have been tested are the (previously mentioned) p -norm and the Kreisselmeier-Steinhauser (KS) function. The p -norm constraint/objective is calculated by:

$$\sigma_{PN} = \left(\sum_{e=1}^N \sigma_e^P \right)^{1/P}$$

Where

P is the stress norm parameter, Le et al. (2010) have achieved best results with 6 and 8.

ρ_e is the pseudo-density of element e

σ_e is the stress at element e once a stress criterion has been applied

The p -norm can be used either as a constraint or as an objective. If used as an objective, the value of the stress norm parameter, P , is of less importance, as the algorithm only needs a direction to move into, not an absolute value (Le et al., 2010). As P approaches infinity, the p -norm approaches the maximum stress, as it approaches 1, the p -norm approaches the sum of all stresses. A larger P value produces a design with more even stress distribution, and a lower P value generates a design close to a compliance minimization problem (Le et al., 2010). If used as a constraint, it provides a conservative quantification of the stress constraint, being always slightly larger than the maximum stress.

7 Case Study

7.1 Context

This project conducts research through design. The case study is an indoor pedestrian bridge at the British Museum, as was explored by Koniari (2022) and Schoenmaker (2023). The choice of the same case study can offer great insight by allowing for the comparison of the results achieved through different methodologies/objectives.

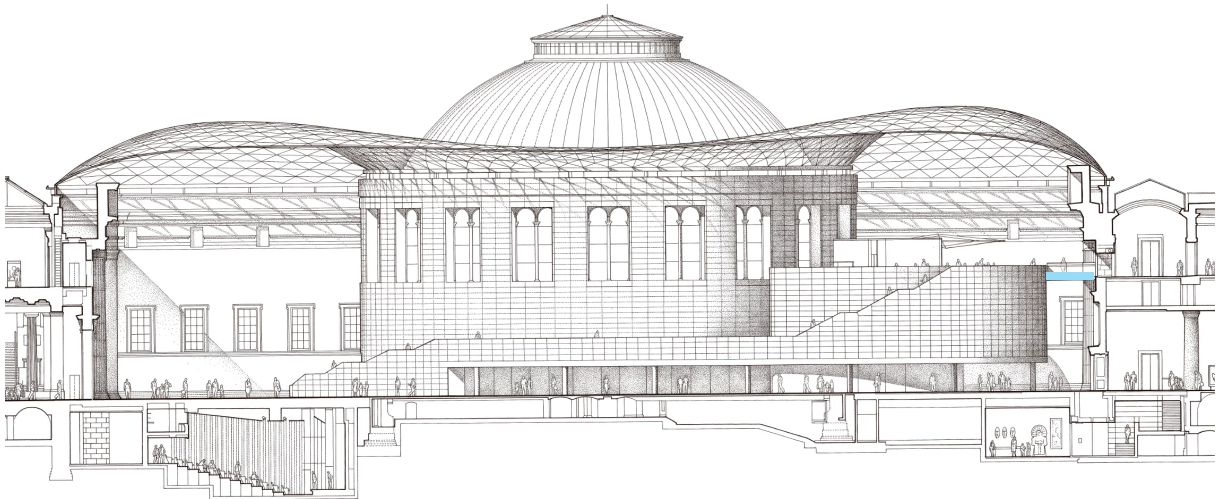


Figure 7.1: Case Study context, inner courtyard of the British Museum. Bridge marked in blue.



Figure 7.2: Previous explorations of the Case Study: Koniari (2022) (left) and Schoenmaker (2023) (right)

The fact that the location of the structural element is indoors reduces the complexity of the project. It does so firstly by allowing the reasonable assumption of minimal temperature fluctuations. This removes the necessity for including expansion and contraction of the element due to temperature changes in the structural calculations. Secondly, the effect of lateral loads due to wind pressure can also be neglected, meaning that all loads to be considered are vertical, and thus making two-dimensional optimization possible. These simplifications make it a good entry point for the already complex task of optimizing structural glass.

7.2 Design domain

Schoenmaker (2023) pointed out that the domain of the case study might be responsible for the similarity in tensile and compressive stresses found in the optimized structure for this very same case study. The shallow height compared to the span might have caused it to behave similarly to an I-beam. Koniari (2022) suggested adding a non-design domain on the upper part of the design domain to evenly support the float glass sheets, since she observed that in the structural verification of her final proposed topology, the largest deformations were found on the places where these top elements were unsupported by the underlying cast glass structure.

For this project, a similar design space will be used as in the works of Koniari (2022) and Schoenmaker (2023) to provide a basis for comparison, since the methodology will already provide extensive explorative ground, and their suggestions for changes will be incorporated.

The pedestrian bridge is 2400 mm wide, 4200 mm long. Koniari (2022) and Schoenmaker (2023) used a height of 300 mm, but this project will use a height of 400 mm. This adaptation is an application of Schoenmaker's observation that the height of the domain, compared to its length, might be responsible for the beam-like topology, and Koniari et al.'s (2023) quantified results which shows that a 300 mm design domain structural height allowed for a better performing design than its 200 mm counterpart. The idea is that a yet larger height will perform even better.

The cast glass structural element is covered by 2 sheets of laminated glass for a smooth horizontal walking surface and as a safety measure. There are handrails on each side of the structure, but those are not included in the optimization. There will also be a non-design domain, in accordance with Koniari's suggestion, with a thickness of 2 finite elements (which is equivalent to 2 cm) at the top of the design domain. The non-design domain will be included in the FEA, but cannot be altered by the optimization algorithm.

In line with a second insight of Schoenmaker (2023), the actual design domain used during optimization will be half of the length of the bridge. This is based on the assumption that a bridge with the same support conditions on both sides, and an even, distributed load at the top, should be symmetrical. Thus a symmetry axis is created in the middle, halving the design domain during optimization and thus greatly reducing the amount of parameters that need to be optimized as well as the number of structural calculations that need to be done (assuming the application of the Finite Element Method, where each node's displacement gets calculated).

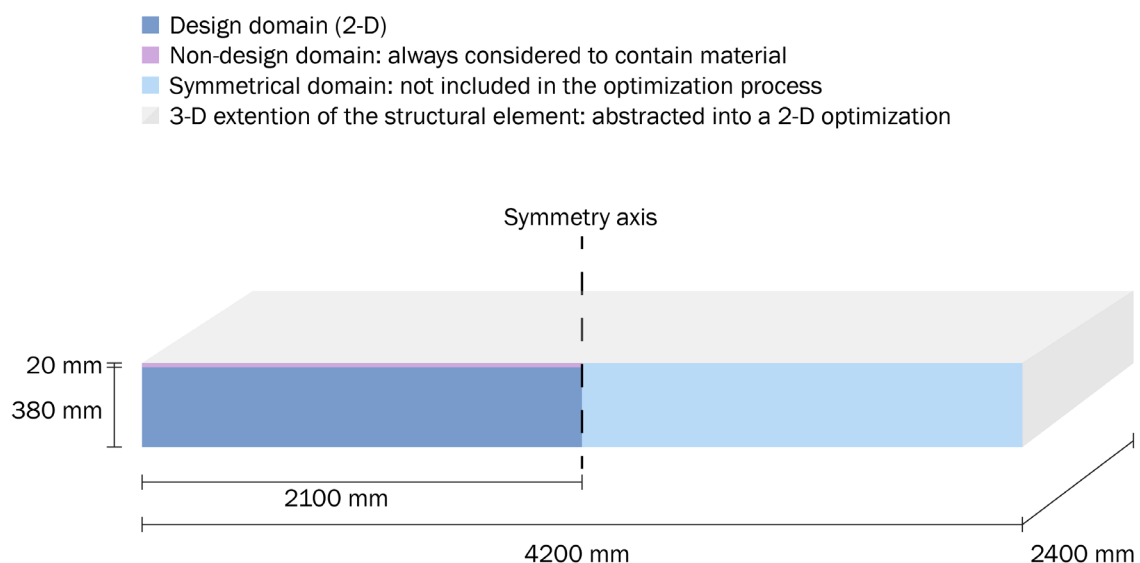


Figure 7.3: Design domain

7.3 Structural problem definition

The structural problem definition include the loads applied and the connection of the structure at the supports. It has a large impact on the final outcome of the optimization process. Damen (2019) found that different load cases yield vastly different topologies after optimization. It seems reasonable to assume that load cases will also greatly impact the stress distribution throughout the element and thus its stress-based optimized topology. For example, point loads generate high, localized stress zones, while distributed loads produce a more consistent stress field. Thus the Case Study definition is a crucial step for the results of the project.

7.3.1 Loads

For the loads, there are two important distinctions. The first regards the duration of the load and affects the safety factor used. Permanent loads include: the cast glass optimized structure itself and the 2 sheets of laminated glass on top. The permanent loads' value is calculated by multiplying their volume (which changes in each iteration) by the density of glass ($2500 \times 10^9 \text{ kg/mm}^3$). Temporary loads are the users and the maintenance loads. These are $5/10^6$ and $0.4/10^6 \text{ kN/mm}^2$ respectively.

The second distinction is about where those loads are applied. The temporary loads are applied to the very top surface of the structure, where people and maintenance crews walk. The permanent loads are applied to where the material is present. (This might be more clear in Chapter 5 about the Finite Element Method used for structural calculation). This second distinction is the only difference between this thesis and the two previous theses in terms of the load application, as they applied all loads evenly across the entire structure. This distinction is important because the point of application of the loads will lead to different stresses in the structure, and thus might also lead to different optimized topologies. The application of the temporary loads on top of the structural is actually beneficial for a material that is stronger in compression than in tension, and thus this conceptualization of the load applied might provide room for the optimization algorithm to lead the loads to the supports using more compressive than tensile strength.

7.3.2 Support

The supports are considered fixed on both ends of the bridge. Detailed attention to the supporting mechanism is beyond the scope of this project. It can be assumed to be a metal profile with some padding to accommodate the glass, and prevent translation as well as rotation.

A hinged connection was also considered and tested in the optimization algorithm, but not extensively, and could be the subject of future explorations. The connection of the structure at only one point (in x and y), together with the particular material distribution algorithm used here caused the optimization to be unstable. Given that feasible topologies are achieved with the fixed connection, this path is chosen.

Support in the horizontal direction is placed at the symmetry axis to model the resistance of the other half of the bridge. Vertical displacement is left free at the symmetry axis.

The fixed support is modelled as fixed translation in both the vertical and horizontal directions at every node present in the first column.

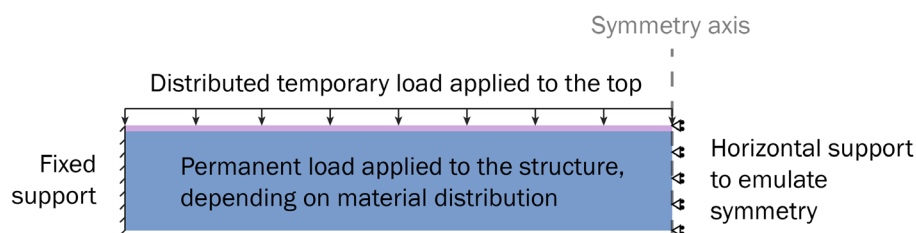


Figure 7.4: Boundary conditions

8 Workflow & Algorithm

The workflow proposed here did not come to be all at once. There were many trials and errors before the version presented here. Since it is already quite a lot of information, only the final version will be presented. Alternative paths considered can be found in **Appendix 14.4**.

One important decision made, was about which kind of model should be used for this project: a more traditional approach with a parametric topology or a ML method. Regarding the relationship between the dataset and the model, neural networks tend to perform well for interpolation (i.e. generating data that are a combination of the training set data), but it is more difficult to generate completely new examples that lie outside the boundaries of the training dataset (Burton, 2024). It was this quality that led the current work towards optimization (in order to create feasible examples) instead of directly applying ML, despite the large potential of the latter in topology optimization. Since without a fairly large dataset with at least 10,000 examples that represents feasible options, it would be difficult to train a model to output a structurally high-performing topology. There is currently no large dataset of feasible cast-glass topologies, and thus the alternative of creating an optimization workflow which outputs structurally optimized topologies was chosen. This workflow could potentially be used in the future to produce a dataset, which in turn could be used to train a ML model, if that is desired.

The workflow proposed for this thesis is inspired by the Level Set approach, but it also includes elements of the SIMP methodology (i.e. pseudo-densities) and tools/methods usually applied in machine learning (i.e. autograd pytorch functionality, sigmoid curve, stochastic gradient descent optimizer), and thus there are some terms which could be clarified. For example, in the optimization context, the goal of the process is modelled by the so called *objective* function or *fitness* function, which gets minimized as the optimization runs. In machine learning, when training a model one aims to decrease the *loss* function. As the workflow proposed here blends these two approaches, the terms *objective*, *fitness*, and *loss* are all used to mean the same thing: the value that should be minimized during the optimization by adapting the weights.

The algorithm is coded in python, version 3.12.9 and makes extensive use of pytorch version 2.6.0. Images were generated using matplotlib, and some minimal numpy has been used as well.

In this section, the steps included in the workflow and their motivations will be outlined, as well as their mathematical implementation in the algorithm. Firstly, it is important to be reminded of the general form of a structural optimization workflow, as seen in **Figure 8.1**.

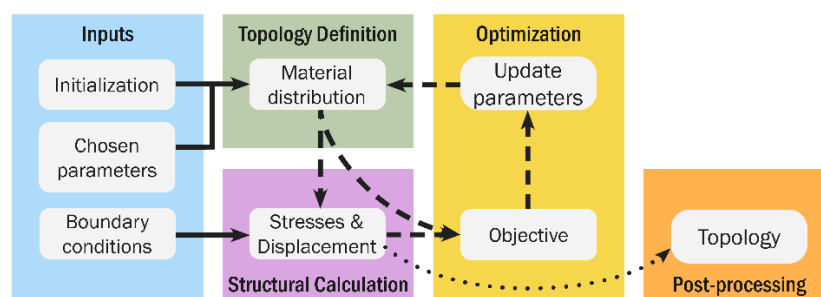


Figure 8.1: Compact workflow diagram

8.1 Inputs

The inputs related to the material properties of cast glass, domain discretization and boundary conditions required for the algorithm are inherited from the literature research done in Chapter 2, on Glass, Chapter 4, on the FEM, and Chapter 5, on the Case Study.

The chosen parameters for the material distribution will be discussed in **Subsection 8.2.2**.

The initialization of the material distribution is done randomly. This strips away preconceptions about the ideal design, and also allows for different optimization runs to be performed, each of which will yield its own topology. This is especially the case given that the method used here tends to result in local minima.

8.2 Topology Definition

The workflow starts by creating a design domain. In this case, it is a rectangular, 2-D surface, abstracted from the final desired 3-D element, as seen in the Case Study Chapter. It is a continuous domain at first.

The material distribution is created by the application of the LSM. This method creates a three-dimensional function which – when sliced twice horizontally – provides a two-dimensional material distribution instance. The first step towards the material distribution is the definition of this 3-D function.

It was very important in this project to decrease the computational burden wherever possible. Stress-based optimizations are notoriously computationally intensive. As discussed previously, all the local stress measures need to be calculated, somehow transformed into a value that can be minimized, and then all the parameters that affect it need their gradients and updates to be calculated as well. This differs from a volume or compliance-based optimizations, which use a global measure in their objective function.

Thus, apart from the manufacturing constraints, there was an extra incentive in using the LSM instead of SIMP to reduce the number of design parameters. This would in turn reduce the computational power needed to run the optimization process.

8.2.1 Function parameterisation: Gaussian Landscape

The current work proposes the use of Gaussians as the basis functions that define the LSF. As seen in the literature research, the basis functions are functions which are added up to form the LSF. Gaussians are global functions, and this allows them to create smooth boundary transitions when added up. Since glass is a brittle material, smooth transitions are desired, as explained in Chapter 2. Thus Gaussian functions, which provide those smooth transitions, are postulated to be a fitting choice for the definition of the cast-glass topology.

A single 3-D Gaussian function is defined by: (i) its origin point in the x, y plane (x_i, y_i), (ii) its amplitude, A , which is its height in the z-direction, and (iii) its spread (a.k.a. standard deviation), s , which defines its width. In this particular definition, the Gaussian function is radially symmetrical about the vertical axis that passes through its origin point, making it a Radial Basis Function (RBF).

Eq. 8.1

$$z = Ae^{-\left(\frac{(x-x_i)^2+(y-y_i)^2}{s^2}\right)}$$

Where

z is the height of the landscape at the coordinates (x, y)

(x_i, y_i) are the coordinates of the origin of each Gaussian peak

A is the Amplitude of the Gaussian peaks

s is the spread of the Gaussian peaks

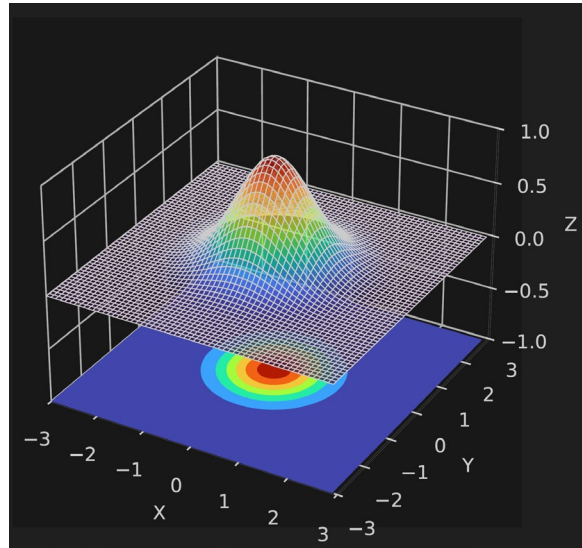


Figure 8.2: One 3-D Gaussian peak and its corresponding contour map underneath.
Where $A = 1$, $s = 1$, and $(x_i, y_i) = (0, 0)$.

The LSF is then defined as the sum of a given number of Gaussian functions. The landscape is a summation of the altitudes (i.e. z -values) of the given peaks, such that:

Eq. 8.2

$$z = \sum_{n=1}^N A e^{-\left(\frac{(x-x_n)^2 + (y-y_n)^2}{s^2}\right)}$$

Where:

(x_n, y_n) are the coordinates of the origin of each Gaussian peak

n is the number of Gaussian peaks, which in this project is 80, unless otherwise specified.

The Gaussian functions are placed in the design domain given their (x_n, y_n) coordinates. These coordinates are the design variables which will be updated throughout the optimization process. In order to form the first guess of the material distribution, they are randomly placed within the design domain boundary. This randomness is used in order to avoid preconceptions about what the final optimized structure should look like. It is known that the results achieved through the LSM are highly dependent on the initial material configuration (i.e. local minima). The use of randomness of the initial condition allows for virtually infinite trials, which can later be compared and chosen by the designer. On the negative side, none of the tried random initializations might be the actually optimal one. Each of them have their own bias towards a certain result. So this becomes a numbers game: producing a quantity of designs to find quality amongst them.

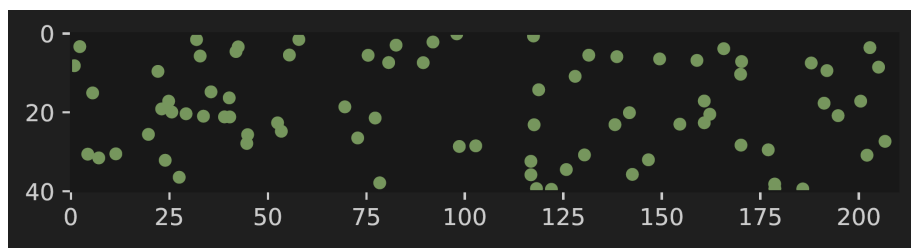


Figure 8.3: Placing centroids (x_n, y_n) randomly in the design domain at the start of the optimization.

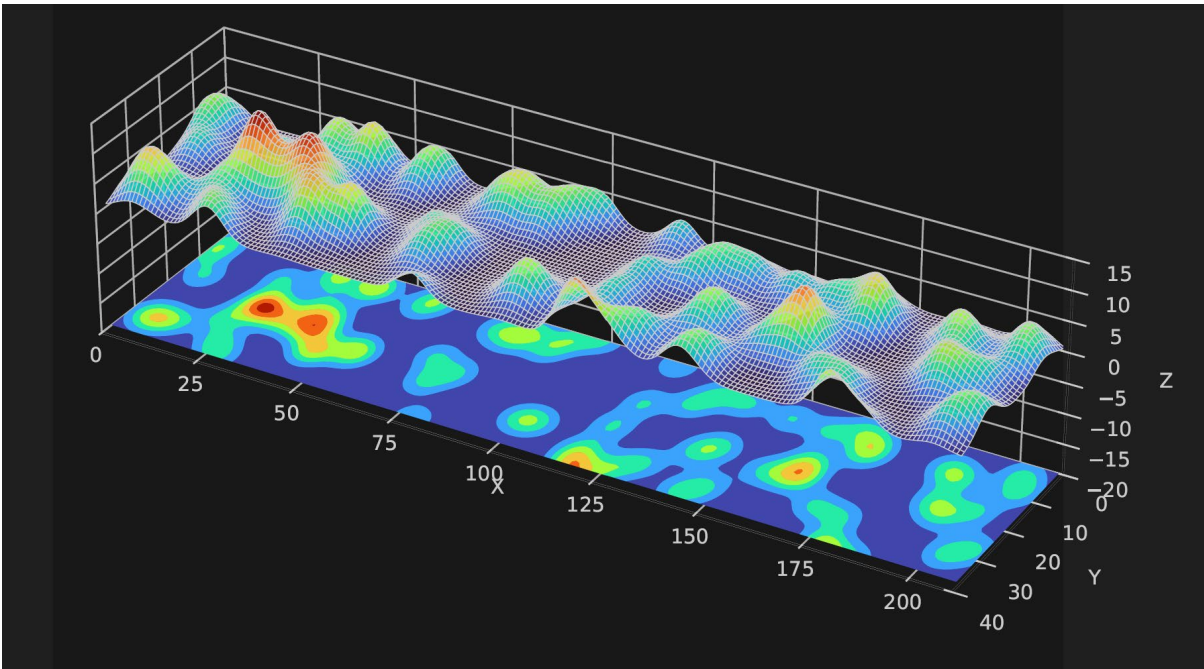


Figure 8.4: 3-D visualization of the Gaussian Landscape created by the addition of Gaussian peaks in accordance with Eq. 8.2 placed on the centroids shown in **Figure 8.3**. The contour map is projected onto the plane at $z = -20$.

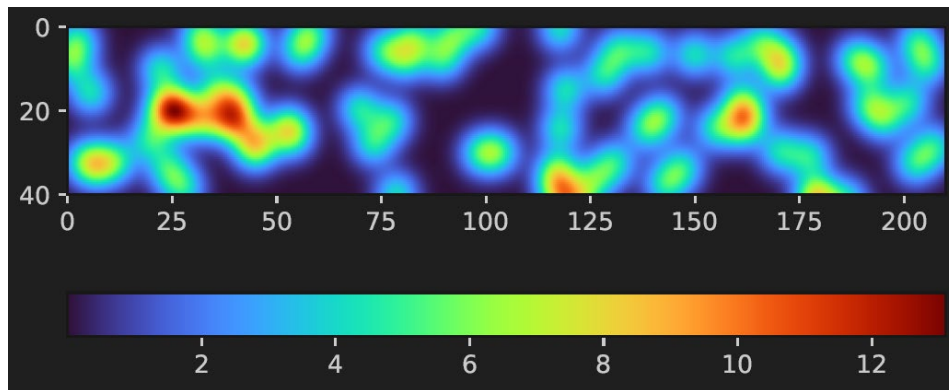


Figure 8.5: 2-D visualization of the Gaussian Landscape created by the addition of Gaussian peaks in accordance with Eq. 8.2 placed on the centroids shown in **Figure 8.3**.

In this paper, this 3D function will be referred to as a *landscape*, due to its resemblance to a mountain range. More specifically, since it is composed of a sum of Gaussian 3-D peaks, it will be called a Gaussian landscape. For the optimization runs which are randomly initialized, this landscape is analogous to a Gaussian Random Field, as referred to in literature.

This concludes the creation of the LSF. During optimization, the design parameters (i.e. the position of the centroids of each Gaussian peak individually, and the spread of all of them in unison) are changed by the optimizer.

8.2.2 Geometry mapping: slicing the Landscape twice with a sigmoid

This section deals with the translation of the landscape into a topological geometry that can be evaluated by the objective function. The main idea follows the LSM of slicing the 3-D landscape horizontally and defining the topology based on whether the landscape is above or below that horizontal plane. However, there is a novel approach here of slicing the landscape not with one horizontal plane, but with two. This

is done to test the idea of creating, not blocks of material that connect to each other, but paths of stress flow instead. Let us first introduce the simpler *discrete* slicing function.

The typical LSM creates a discrete material distribution as shown in Eq. 4.1. The double slicing explored in the present work is given by:

Eq. 8.3

$$\begin{aligned} l > \phi(x) < u, \quad x &\in \Omega \text{ (material)} \\ \phi(x) = l \vee \phi(x) = u, \quad x &\in \Gamma \text{ (boundary)} \\ \phi(x) < l \vee \phi(x) > u, \quad x &\notin \Omega \text{ (void)} \end{aligned}$$

Where:

ϕ is a LSF

x is a point in the design domain

l is the z-value of the lower plane (confusingly marked as level set in yellow in Figure 8.6)

u is the z-value of the upper plane (marked in blue in Figure 8.6)

Slicing the 3-D function once creates a chunk of material. Going one step further and slicing it again with a plane that is shifted upwards from the first one in accordance with Eq. 8.3, creates a hole within that chunk, and a material path along its boarder.

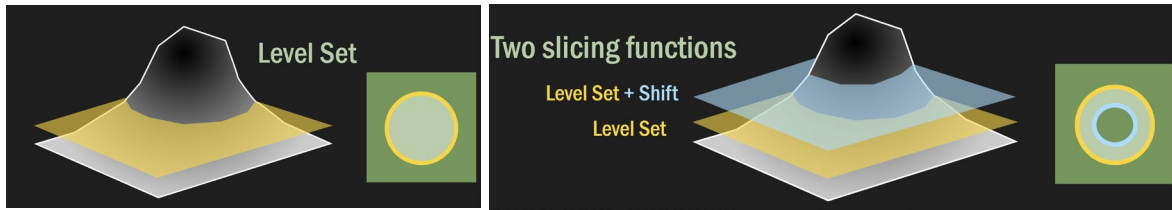


Figure 8.6: Slicing a single Gaussian peak with one (left) or two (right) horizontal planes and their corresponding mapped geometry. The dark green area represents no material, the light green represents material. The line circles represent the intersection between the planes and the Gaussian (i.e. the boundary). The outer boundary is defined by the lower plane (yellow) and the inner boundary is defined by the upper plane (blue).

Figure 8.6 shows, on the left side, a chunk of material created (the filled circle), and on the right a path created (the donut/empty circle). Given that for this project an even cross-sectional area is desired, the amplitude and spread of the peaks does not vary between them. Allowing them to differ would create a significant difference in the slopes of the landscape, which translates into a changing thickness of the cross sectional area after the horizontal slicing. That is why it was chosen to keep them the same for all peaks. The spread is also kept the same in both directions (x and y) for the same reason. This has the consequence that the building blocks of the topology are circular (not oval). Those circular paths can be combined to form more complex geometries.

Parameters used in the definition of the Gaussian Landscape and slicing function “are the design variable in this method but do not carry much physical meaning on their own. They must be converted into element [pseudo densities] before topology, stiffness and volume of material can be determined. This conversion takes place through” discretization and the application of the slicing function to the Gaussian Landscape (Guest et al., 2004, p.240).

When mapping the landscape to a material distribution, the continuous design domain gets discretised. The definition of the LSF is decoupled from the discretization. That is to say, with the exact same LSF (and

thus the same number of design parameters) one can choose different levels of structural calculation accuracy through the choice of type and size of finite element during discretization.

In the present work, the design domain gets discretised into square elements of 10 x 10 mm, meaning that the design domain has 40 elements in the vertical (y) direction and 210 elements in the horizontal (x) direction. The centroids of each finite element (FE) are the (x, y) coordinates used to define the material distribution.

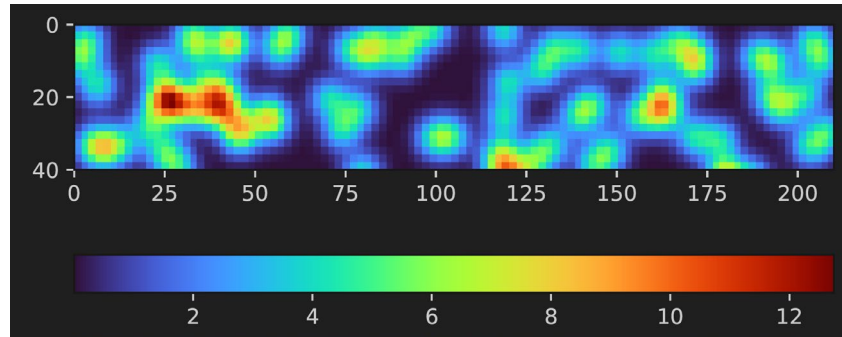


Figure 8.7: Rough discretization of the design domain with square elements of 20 x 20 mm and the evaluation of the landscape height (z) at each finite element. Similar to Figure 8.5, but now discretised instead of continuous.

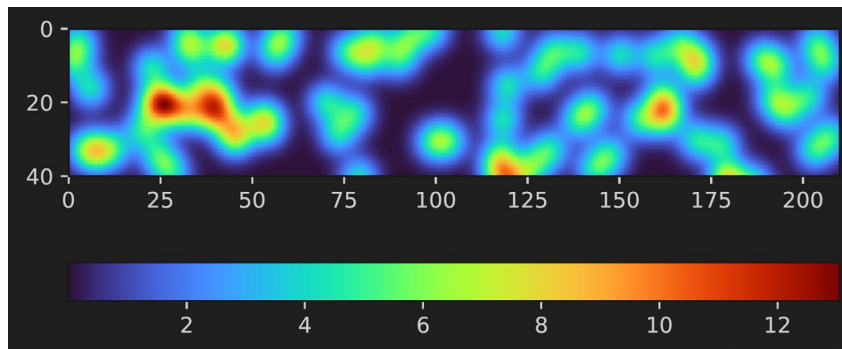


Figure 8.8: Discretization of the design domain with square elements of 10 x 10 mm. Visually indistinguishable from the continuous design domain.

Figure 8.7 shows the discretization of the design domain into larger finite elements for visualization purposes. **Figure 8.8** shows the discretization of the domain with the finite element size used in the present work, and evaluation of the landscape at each of the finite elements' centroid. The previous explorations of this Case Study have used finite elements with 20 mm side dimension: square in Koniari (2022) and cube in Schoenmaker (2023). The present work used a 10 mm side dimension to try to capture local stresses more precisely, since this is a stress-based optimization workflow.

Once the domain has been discretised and the value of the landscape for each element has been calculated in accordance with Eq. 8.2, the slicing of the landscape takes place. At every (x, y) coordinate for which the altitude of the landscape (z) falls in between the two planes, there is material. Conversely, where the altitude falls outside of the gap between those two planes, there is no material.

The lowest plane is defined by the value of the level set variable (in honour of the level set method, but honestly in hindsight that is just confusing). This is a scalar value which defines the plane's height. The second plane is higher up, or shifted up from the level set. It is defined by the level set plus a shift value. This shift is the gap between the two planes and is one of the variables that defines the cross-sectional thickness of the topology.

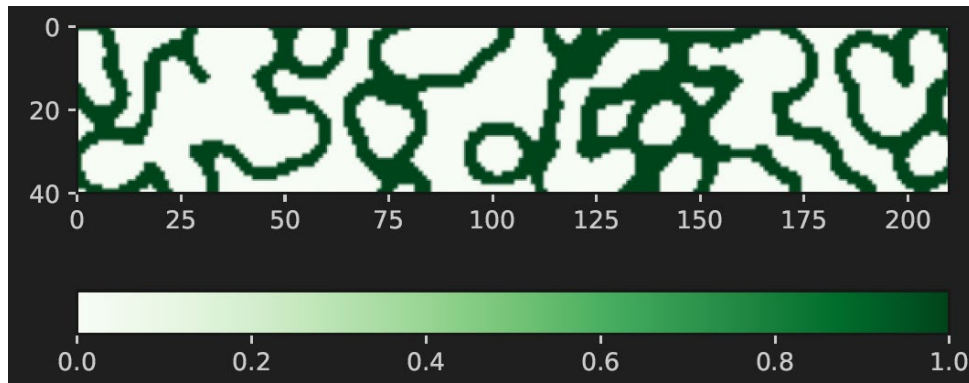


Figure 8.9: Discrete geometry mapping. White means no material, dark green means material.

Figure 8.9 shows the discrete material distribution that arises from slicing the landscape presented previously by the two planes. It should be emphasised that this is a completely random initialization and thus not optimized at all at this stage.

At this point, each pixel has a material, no-material definition that can be used for the structural calculations. In order to avoid singularity, the minimum density (i.e. the density in the no-material area) is set to a very small value, close to zero, but not zero. The density in the areas with material is set to one. This use of pseudo-densities (albeit not as the design parameters) resembles the SIMP approach.

However, an important distinction between the workflow proposed in this paper and the SIMP approach that is also a consequence of how those pseudo densities get defined, is that it needs to optimize less parameters. For example, the case study explored here has a structural height of 400 mm and a structural length of 2100 mm with a finite element (FE) size of 10mm. This means 40 x 210 FEs, or 8400 FEs. This is the amount of parameters used in a SIMP optimization with this problem definition. In this workflow, however, the pseudo-densities are a consequence of the landscape and slicing function, which together can be defined with about 162 parameters. The exact value is not set in stone. The trials run so far had 162 parameters: 2 * 80 (x, y coordinates of each of the 80 Gaussian peaks), plus one spread value, plus one shift value. That is about 2% of the amount of parameters found in SIMP for the same discretization. This makes it possible to run the optimization with much less computing power. Additionally, since the topology definition is inspired by the manufacturing constraints of glass, less checks need to be performed during the optimization. On the other hand, the fact that the topology here is constrained by the geometry of the system also means that it has less freedom than with SIMP and that the optimal results might not be possible to construct in the present system.

Initially, the horizontal sliding proposed was as black-and-white as the one shown in **Figure 8.9**. However, once the optimization stage is reached, it became clear that a differentiable topology definition is needed. That is to say, a way of translating the landscape into a topology which had no abrupt jumps in value. The discrete check of either between the horizontal planes or not between them did not qualify, since it is not differentiable.

In order to apply automatic differentiation, the slicing function must be smoothly defined so that the gradient of the pseudo-density, ρ_e , with respect to the design variables is continuous. The function needed here is one that outputs a value between 0 and 1: no-material and material, but does so with a gradual change between the two. In the context of machine learning, there is a well-known activation function called sigmoid, which does exactly that. So this function was chosen as the intermediary between the landscape and the topology. It translates the three-dimensional landscape into a topology with densities ranging between zero and one.

In the use of these pseudo-densities, it resembles the SIMP approach. However, because the value of those densities is not defined per pixel, but rather by the landscape and sigmoid combination, the

challenge faced by SIMP of intermediate densities can be mitigated by a careful selection of the parameters of the landscape and sigmoid functions, so that no punishment of intermediate densities is needed. More specifically, the slope of the sigmoid function has a very large impact on avoiding intermediate densities, as does the slope of the LSF at the height of the cutting planes.

The sigmoid outputs values between 0 and 1, but it could not be used in its raw form. It can create a single boundary between material and non-material. But for this project, the idea is to have two cut-off thresholds (which previously were the two horizontal planes). So a modification had to be made. The adaptation procedure will be outlined below. As a basis for the following adaptation, it is useful to know that the raw sigmoid function is given by:

Eq. 8.4

$$S(x) = \frac{1}{1 + e^{(-x + a) \times b}}$$

Where:

a is a horizontal shift in the positive x direction

b is the slope of the curve (i.e. how “quickly” it goes from zero to one)

x is the value being tested (i.e. the height of the landscape, z at each finite element)

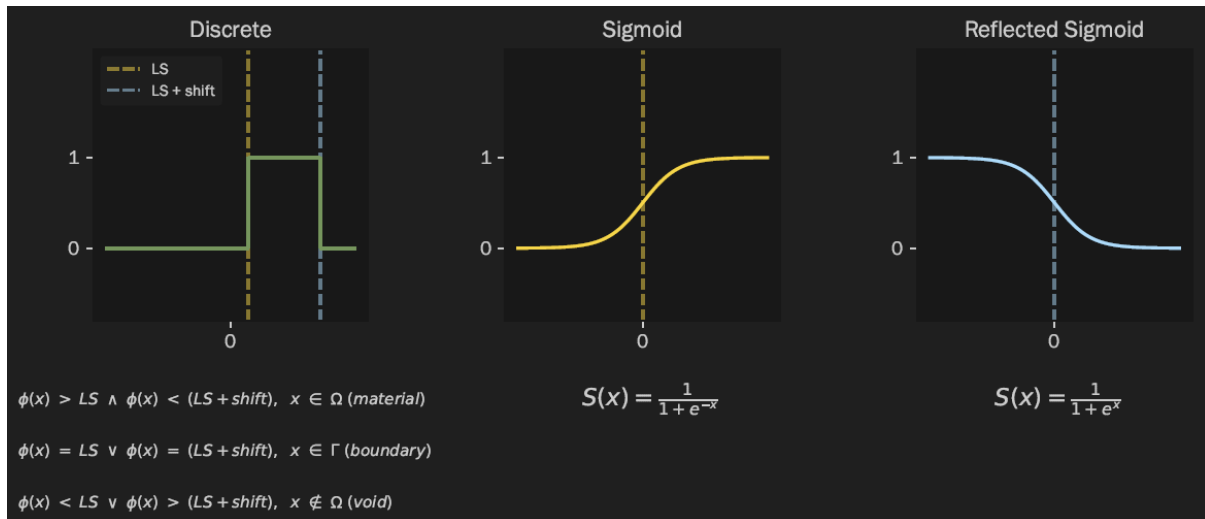


Figure 8.10: Construction of the sigmoid slicing curve, part 1.

Figure 8.10 shows the discrete slicing function (left), previously explained with the two cutting planes. The goal is to achieve a similar function, but with a smooth transition between 0 and 1. IT also shows the raw sigmoid (middle), which goes up from zero to one smoothly, and its reflection over the y-axis, which goes back to zero from one. In these two sigmoids, a = 0, meaning no horizontal translation, while b = 1 (middle) and b= -1 (right), meaning that they have slopes in opposite directions (values a and b as seen in Eq. 8.4).

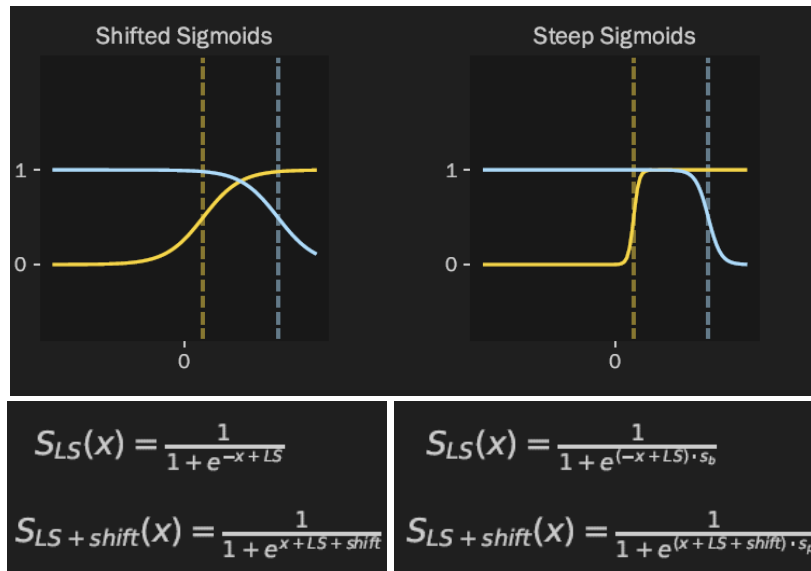


Figure 8.11: Shifting the sigmoid functions and creating steeper slopes.

On Figure 8.11, the sigmoid functions are added to the same graph. They are shifted (left graph) so that their central axis aligns with the lower plane (yellow) and upper plane (blue). Their slopes are also increased (right graph) to create a faster increase from zero to one, which translates to a sharper boundary between material and no-material, or a narrower intermediate density belt around the topology boundary.

The definition of the slopes is very tricky. On the one hand, they should be as steep as possible to best approximate the discrete definition shown in Figure 8.10 (left). On the other hand, they should not be so steep that they cause numerical instabilities during the optimization process.

As if that balancing act was not enough, there is an additional consideration. They interact with the Gaussian landscape at different heights of the landscape. Just as the two horizontal planes also interacted with the landscape at different heights. These different heights of the landscape have themselves different slopes. It is the interaction of the slope of the landscape and the slope of the sigmoid that creates the intermediate material band around the topology. Since the landscape has a lower slope around the basis, where it is “cut” by the lower plane, this is counterbalanced by a steeper slope on the lower (“left”) side of the sigmoid. The landscape is steeper where it is cut by the upper plane, so the slope of the sigmoid there can be lower. This interaction will be further discussed at the end of this subsection.

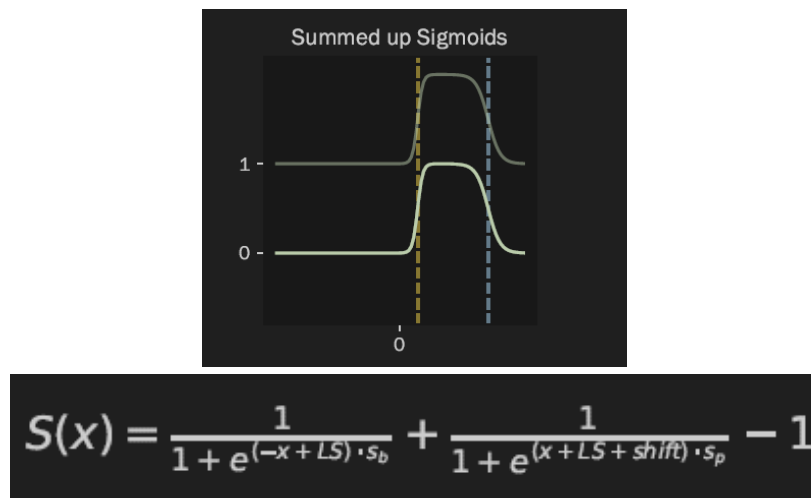


Figure 8.12: Combining the two sigmoid functions to create a single slicing function.

The final step in the creation of the differentiable slicing function is to combine the two sigmoid functions into one. This is done by simply adding them up and then subtracting 1. Thus, the final formulation reads:

Eq. 8.5

$$S(x) = \frac{1}{e^{(-x+a) \times b}} + \frac{1}{e^{(-x+c) \times -d}} - 1$$

Where:

a is the height of the lower cutting plane (previously named level set, in yellow)

b is the slope that defines the sharpness of the outer boundary of the topology

c is the height of the upper cutting plane (previously named level set plus shift, in blue)

d is the slope that defines the sharpness of the inner boundary of the topology

The height of the landscape (z) (at each coordinate (x, y)) is the (x) input for the sigmoid function. And the output of the sigmoid function then defines the pseudo density of the topology at that particular coordinate. Since the height value (i.e. vertical value) of the landscape serves as the x-input for the sigmoid, the later might be presented with its x-axis vertically aligned to make their connection clearer. The adapted sigmoid is shown vertically in pink in **Figure 8.13**.

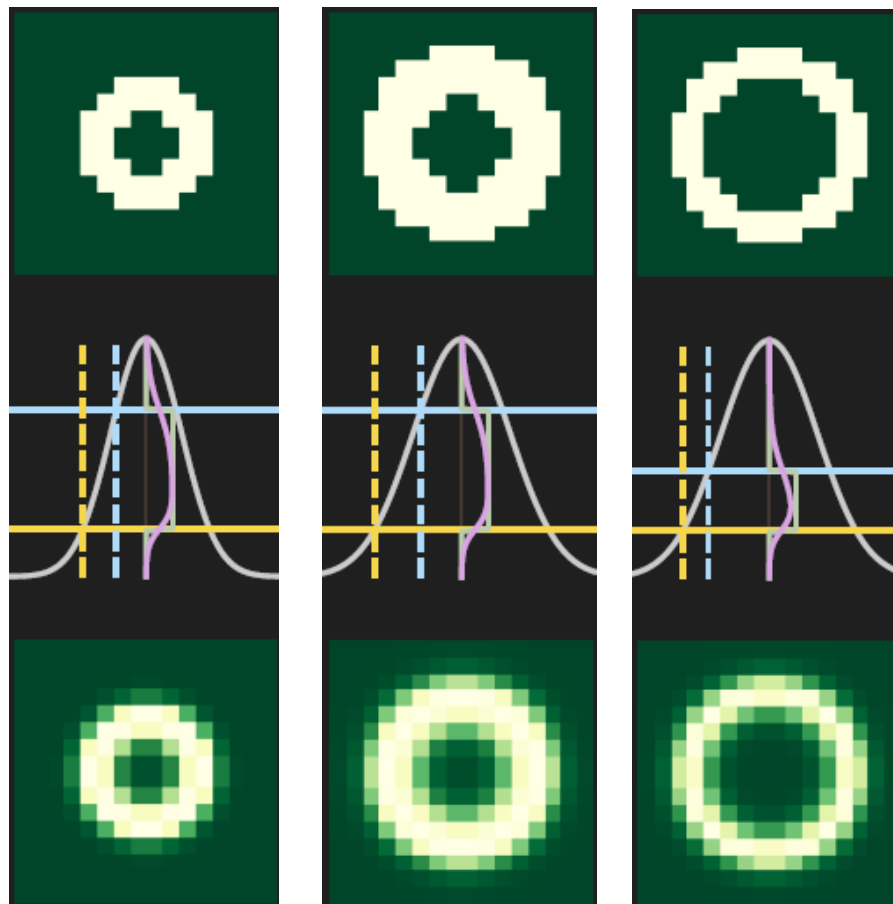


Figure 8.13: Different combinations of shift and spread, and their effect on the material distribution. Left: shift = 2, spread = 3.3; Middle: shift = 2, spread = 4.5; Right: shift = 1, spread = 4.5. At the top, the discrete material distribution, and at the bottom the differentiable material distribution in accordance with Eq. 8.5.

As previously mentioned, the design parameters are the centroids of the Gaussian peaks, the shift, and the spread. **Figure 8.13** shows the effect of changing the shift and spread on the material distribution. For clarity, this is done on a single Gaussian peak. The Gaussian peak is shown in 2-D, looking at plane x-z (or y-z, they look the same), while the material distributions are shown from the top, looking at plane x-y.

If the example in the *middle* of **Figure 8.13** is taken as reference, then to the left only the spread changes, which has an effect on the overall size of the material doughnut. Changing the spread is the only way to affect the outer boundary size, but also has an effect on the inner boundary size (i.e. the size of the hole). It affects both because it changes the actual shape of the entire Gaussian peak.

Conversely, when the example on the right of **Figure 8.13** is compared to the middle, then only the shift changes. This affects solely the size of the inner void, leaving the outer boundary unchanged. This is because the shift is a parameter of the slicing function. More specifically, it defines the cutting height of the second plane, so it does not affect the cutting done by the lower plane nor the Gaussian Landscape as a whole.

By introducing these two design parameters on top of the placement of each Gaussian peak, the algorithm can control the size and thickness of the basic shape that composes the topology.

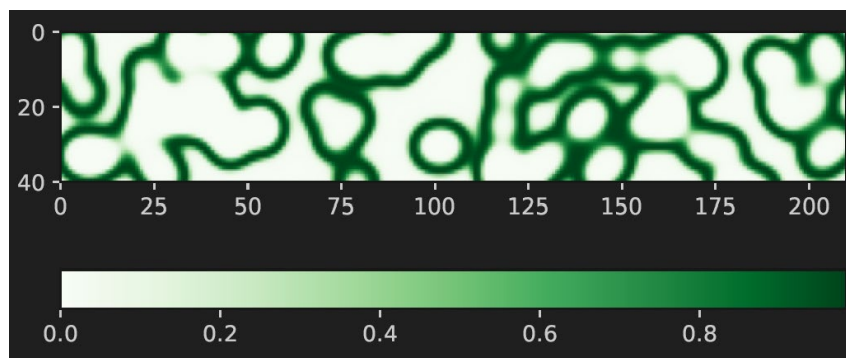


Figure 8.14: Differentiable material definition (comparable to the previously shown discrete material definition on Figure 8.9)

Figure 8.14 shows the application of the differentiable material distribution definition to the same landscape that the discrete distribution had been previously applied. Here the image seems “blurry”, which is the effect caused by the intermediate densities along the topology boundaries.

The material distribution is now almost ready for its structural evaluation. The last step is to add material to the non-design domain. In **Figure 8.15** the final output of the material distribution process can be seen, including the two layers of filled finite elements at the top, which cannot be affected by the optimization algorithm. After this addition, the material distribution enters the structural calculation process (i.e. FEA) for this iteration.

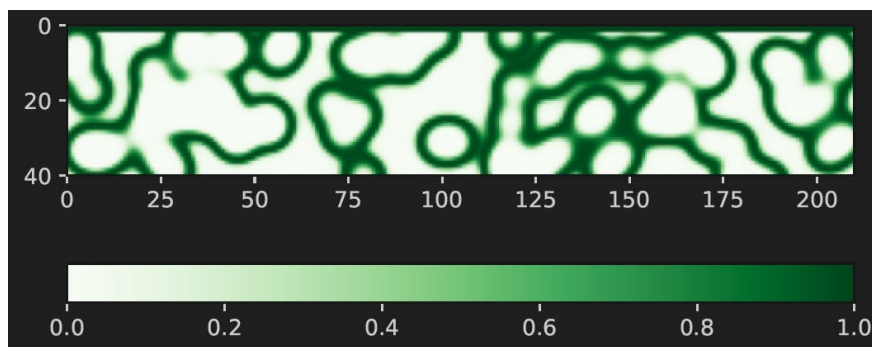


Figure 8.15: Differentiable material definition with insertion of the non-design domain at the top

This material distribution process is repeated at each iteration, given the updated design parameters and produces the topology for the next structural evaluation.

In this workflow, some of the problems of SIMP are avoided: the checkerboard pattern and discontinuity (e.g. islands, disconnected parts of the structure). Checkerboard is avoided by the level-set inspired approach of producing the topology by slicing a 3-D function. Thus the algorithm does not have direct access to each FE density, and it becomes very difficult to create a checkerboard.

Discontinuity is still an issue, but it can be remedied by including a p-norm of the displacement in the objective function. By working towards minimizing the maximum displacement, the algorithm is encouraged to create a structure that is connected to itself, since disconnected chunks experience a very large displacement. (They are still technically connected by elements with a density close to zero.)

There is another issue, arising out of the pseudo-density definition of the landscape: the intermediate densities around the boundary of the topology. In SIMP, those are punished by raising the density to a power bigger than 1 when performing the stiffness matrix assembly. However, for the method proposed here, intermediate densities are by definition unavoidable at the topology boundaries. The sigmoid curves, which make the code differentiable, are also responsible for those intermediate densities. However, by carefully choosing the slopes of the sigmoid, those intermediate densities can be limited to exist in a narrow band around the topology. The algorithm cannot be asked to learn the level set and sigmoid slopes, for that does lead to the spread of intermediate densities, which then in turn would need to be punished. So that algorithm is limited to learning the position of the Gaussian peaks in the x-y plane, the spread of those peaks, and the shift (the height of the second cutting plane above the level set).

Just as in the work of Guest et al. (2004), intermediate densities are always present at the boundary of the structure and cannot be removed through penalization as it is an inherent part of the topology definition. The exact boundary can be found by interpreting the intermediate densities and/or polishing during post-processing. Alternatively, in the present work, the intermediate densities allow for the formation of stress paths in the stress field. It is this stress field that is used in post-processing to determine the final, discretised topology.

8.3 Structural Calculations

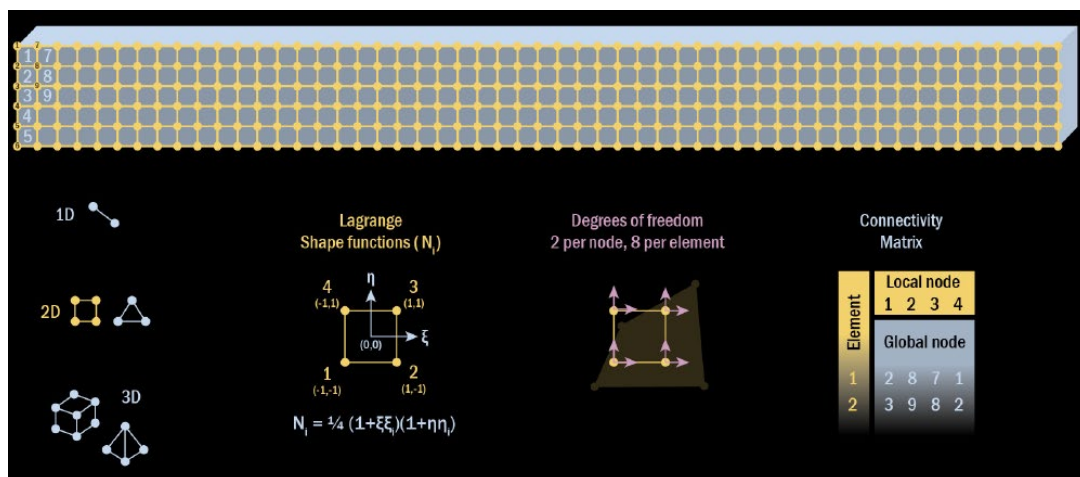


Figure 8.16: Discretised domain with its element type (square elements with four nodes), the shape function to interpolate the displacement inside each element, the 8 degrees of freedom of the square elements, and the Connectivity Matrix, which connects the four local element nodes to their global numbers. The left upper local nodes are shown in light blue with their global values in their respective elements.

The pseudo-density matrix, visualized in **Figure 8.15**, enters the structural calculations. It is the basis for the scaling of three matrices: 1) the stiffness matrix, 2) the self-weight matrix, and 3) the stress field matrix. The first two happen before the FEA, and the last one happens afterwards.

The FEM is used for the structural calculations, which also entails the discretization of the design domain (already performed during the geometry mapping section of material distribution process). For FEA, the elements *and* their nodes are needed, they are connected through the Connectivity Matrix.

8.3.1 Stiffness matrix

Each finite element contributes its stiffness to the global stiffness matrix. The material distribution matrix is used to scale that stiffness contribution by the value of the pseudo-density.

Eq. 8.6

$$E_e = E_{min} + \rho_e \times (E_{mat} - E_{min})$$

Where:

E_e is the element's Young's Modulus (E)

E_{min} is the minimum stiffness, here equal to 10^{-11}

ρ_e is the element's pseudo-density

E_{mat} is the Young's Modulus of the material, here 70 MPa

Each finite element has a stiffness matrix, k . This matrix contains the relationship between all of the elements degrees of freedom. Since here square elements are used, with vertical and horizontal freedom on each of the four nodes, their stiffness matrix is 8×8 . The k_e for a full finite element is computed before the start of the optimization process, since it remains the same at every iteration. This is a benefit of using finite elements that all have the same size and shape. The pseudo-densities scale the stiffness of the k_e , which get assembled into the global stiffness matrix, K .

The optimization process was being cut short due to lack of memory of the laptop. Two changes were implemented. The stiffness matrix is calculated all at once, without saving intermediate steps. This change, together with saving images of the results more sparsely, leads to a vast increase in the number of iterations that are possible to compute using a laptop's CPU. The optimizing power increased from about 140 to 1200 iterations with those two changes. This makes sense, since the stiffness matrix is the largest matrix in this problem.

8.3.2 Self-weight

The self-weight is also scaled by the material distribution matrix. Firstly, the downwards force due to one-fourth of a finite element is calculated before the optimization process begins. One-fourth because the forces are applied to the nodes, and each element has four nodes that carry its weight.

During the optimization, a fourth of each element's weight is applied to each of the four nodes that the element is connected to.

The temporary loads are applied as point loads to all of the on the first row. The permanent load, the self-weight, is adjusted by the material distribution. For example, if one node has three full elements connected to it, and one empty element, as seen in Figure 8.17, it will get 3 times $\frac{1}{4}$ of the force of one full finite element applied to it.

In general, the force, F , applied to each node is given by the sum of: the pseudo-densities of the four elements connected to it multiplied by the force of gravity of a quarter of an element.

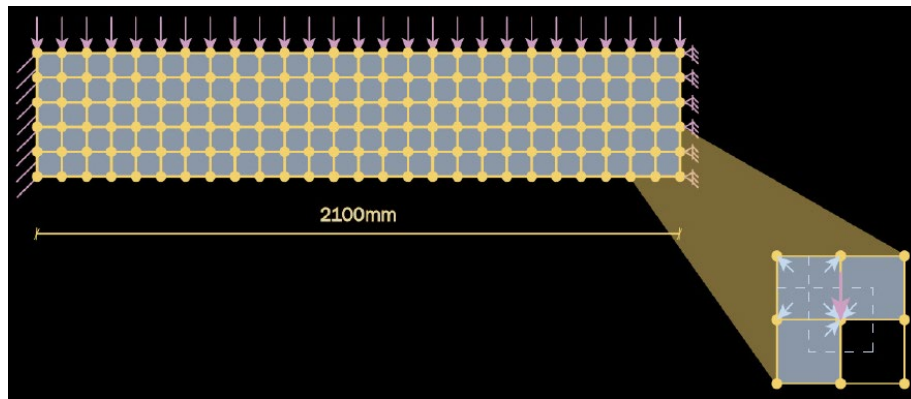


Figure 8.17: Load application to nodes.

Once the vector F and the matrix K have been scaled and assembled, one solves the equation $F=KU$ for U , which is the displacement.

8.3.3 Stress

Sometimes, the FEA ends here. However, for the purposes of this stress-based optimization, the stress is also needed. Using the material stiffness matrix D , the strain-displacement matrix B , and the nodal displacements q (obtained from U), the stress is calculated as $\sigma = DBq$. In 2-D, this gives a 2 by 2 matrix.

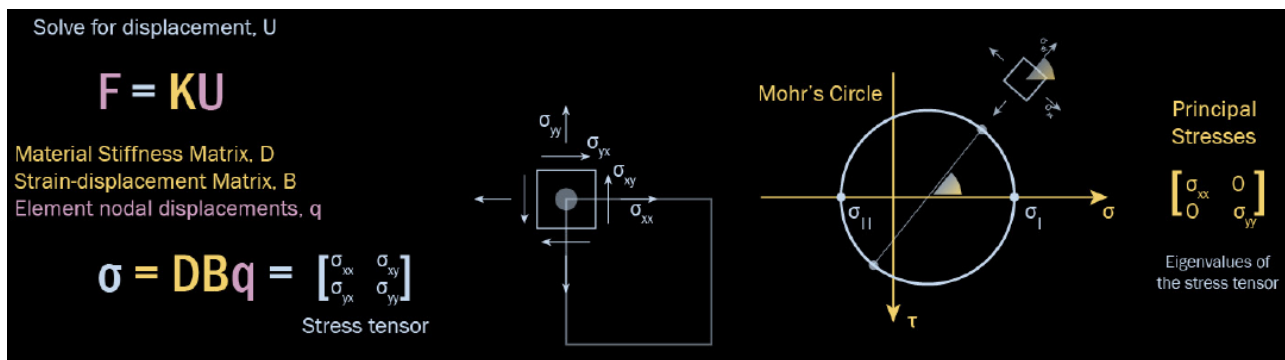


Figure 8.18: FEA and stress

Figure 8.18 shows the calculation of the stress tensor, which corresponds to the stress evaluation at one point. This tensor contains both normal and shear stresses. For each finite element, the stress is calculated at each of the four Gauss points. The Eigenvalues of the stress tensor are calculated to obtain the two principal stresses, which are used in the calculation of the objective function. For each finite element, there are four stress calculations, each with two principal stresses.

A caveat here is that elements with extremely low densities, will have extremely high stresses. This makes it so that the algorithm finds it difficult to remove material. (As it attempts to remove it, the stresses become very large, so it is inclined to put it back again.) For this reason, a stress relaxation can be applied.

The present work does this in an (over-) simplistic way: the calculated stresses are multiplied by the element pseudo-density. Thus the extremely high stresses of almost empty finite elements are multiplied by a tiny number and regain their appropriate proportion. Investigation into more advanced relaxation techniques, such as p-q relation or ϵ -relaxation could be conducted.

8.4 Optimization

8.4.1 Objective

The objective of this structural optimization is to minimize a weighted sum of stress, volume and displacement. Since this is a 2D optimization, volume translates to a minimization of area. The inclusion of displacement as part of the objective function encourages a *connected* material distribution (as

opposed to it having islands, or being composed of physically separate bodies). For the weighted objective function, both the functions as well as the weights of each component need to be defined.

$$f(x) = \alpha \sigma + \beta V + \gamma \delta$$

Where:

α , β , and γ are the weights of the stress, volume and displacement, respectively

σ is the stress

V is the volume

δ is the displacement

Firstly, the term for *stress* included in the objective function is the p-norm of the Drucker-Prager criteria of the relaxed principal stresses of each finite element.

As previously mentioned, each finite element is rotated to find the principal stresses. It is assumed that the first principal stress is in tension and that the second one is in compression. (This is an important assumption because the p-norm implementation in pytorch uses only absolute values, so the distinction between positive and negative is lost.) In case this assumption does not hold true, it is not a big problem, since the first principal stress is then the highest of the two tensile stresses anyway, and for the optimization, it is the peak stresses that define the objective.

For each of the finite elements, there are four Gauss points, and thus for stress evaluations. At this point, a p-norm is applied to each finite element, to find the largest absolute value of the principal stresses of each element. Now each finite element has two stress values: maximum principal stress and minimum principal stress.

For the stress relaxation, the following simplistic formula is applied:

$$\sigma_{ij,e} = \langle \sigma_{ij,e} \rangle \times \rho_e$$

Where

$\sigma_{ij,e}$ is the relaxed stress at element e

$\langle \sigma_{ij,e} \rangle$ is the calculated stress at element e from the FEA

ρ_e is the pseudo-density of element e

ij refers to their position in the stress matrix, if principal stresses are considered (so that 12 and 21 equal zero), then 11 is the first principal stress and 22 is the second principal stress.

Both principal stresses are relaxed. Once they have been relaxed, they enter the Drucker-Prager brittle failure criteria. When expressed in terms of principal stresses, it is:

$$\sigma_{eq,e} = \frac{s+1}{2s} \sqrt{\sigma_{11}^2 + \sigma_{22}^2 - \sigma_{11}\sigma_{22}} + \frac{s-1}{2s} (\sigma_{11} + \sigma_{22})$$

Where

$s = \frac{\sigma_{comp,lim}}{\sigma_{tens,lim}}$, is a constant material property equal to the ratio of the limit compressive stress over the limit tensile stress

σ_{11} is the (relaxed) first principal stress

σ_{22} is the (relaxed) second principal stress

After the Drucker-Prager criteria, there is only one equivalent stress per finite element. These local stresses are now combined into one global measure by using the p-norm.

$$\sigma_{PN} = \left(\sum_{e=1}^N \sigma_{eq}^P \right)^{\frac{1}{P}}$$

Where

σ_{eq} is the equivalent stress calculated using the Drucker-Prager Criteria

e are the finite elements

N is the total number of finite elements

P is the p-value chosen for the p-norm

Pytorch has its own implementation of p-norm, where the inclusion of a p-value is not necessary, one only needs to specify that the p-norm should be used to find a maximum. Thus a choice of p-value does not need to be made by the user/designer.

σ_{PN} is the final formulation of the stress component of the objective function. It is multiplied by the weight, α . The value of alpha used in this project is 70. This is chosen because of the difference in magnitude between the stresses in kN/mm² and the area ratio. A relatively large α makes the quantities have a more similar order of magnitude and thus both become relevant for the algorithm when minimizing the objective function.

The next component of the objective function is area. It is measured as a ratio of the total design domain, giving a result that is between zero and one, zero meaning empty and one meaning a full domain. The individual pseudo-densities of each finite element are added up and then divided by the full domain.

$$A = \frac{\sum_{e=1}^N \rho_e}{N}$$

Where (same as above, and)

ρ_e is the pseudo-density of element e

The value of the area ratio is multiplied by β . The values for β tried in this project range between 0.3 and 0.5. All of the optimization runs presented in the Results section used 0.5.

The last component of the objective function is displacement. Displacement also makes use of the p-norm. It applies it to the displacement of each node, to find the maximum displacement.

$$\delta_{max} = \left(\sum_{n=1}^N \delta_n^P \right)^{\frac{1}{P}}$$

The calculated global displacement is weighted by γ . The value used in this project is 1. This is only based on empirical observation that it seemed to encourage a connected structure. No further investigation was carried out.

With these three components taken into account, the objective function is defined as:

$$f(x) = \alpha \left(\sum_{e=1}^N \sigma_{eq,e}^P \right)^{\frac{1}{P}} + \beta \frac{\sum_{e=1}^N \rho_e}{N} + \gamma \left(\sum_{n=1}^N \delta_n^P \right)^{\frac{1}{P}}$$

And the goal of the optimization is defined as the minimization of the above mentioned function with respect to the optimizable parameters, x :

$$\min_x : \alpha \left(\sum_{e=1}^N \sigma_{eq,e}^P \right)^{1/P} + \beta \frac{\sum_{e=1}^N \rho_e}{N} + \gamma \left(\sum_{n=1}^N \delta_n^P \right)^{1/P}$$

Where:

α is the stress weight

ρ_e is the pseudodensity of element e

$\sigma_{eq,e}$ is the equivalent stress of element e , according to the Drucker-Prager Criteria

β is the area weight

N is the total number of finite elements or nodes

γ is the displacement weight

δ_n is the displacement of node n

P is the power for the p -norm

And, last but not least,

x are the optimizable parameters, which in this case are the (x, y) coordinates of the Gaussian peaks, the spread of those peaks, and the shift value of the sigmoid function. The stress, volume and displacement terms all depends on all of the parameters.

8.4.2 Constraints

The constraint of following $F = KU$ is included in the calculation of the loss function. No additional constraints are included during the optimization.

The constraints of this optimization are the deflection for serviceability, which, given the stiff and brittle nature of glass, is met by all the topologies that also meet the stress constraints. It has only been checked afterwards, but not coded into the optimization.

The stress is minimized as part of the objective, and thus the most successful optimization runs also already fulfill the stress constraint.

The manufacturing constraints are partly embedded in the material distribution definition (using the Gaussian landscape for rounded edges and slicing it at two shifted elevations for even thickness) and partly evaluated during post processing. Thus they are only soft constraints in this workflow.

The minimum size for the mould is not enforced at all. The results will be visually inspected to check whether they meet this requirement.

8.4.3 Gradient

The gradient is calculated using automatic differentiation. This is done in pytorch, by setting `requires_grad = True` to the design parameters, and calling `backward()` on the objective function.

Pytorch has an inbuilt method for performing the gradient calculation. Once the `requires_grad` is set to `True`, it collects a map of the operations performed on those variables during the forward pass. The forward pass in this case are all the calculations from the material distribution (how the centroids and spread define the Gaussian Landscape, and how the shift affects the calculation of the pseudo-densities), through the FEA, and the calculation of the objective function.

Once the result of the objective function has been reached, `backward()` is called. Now pytorch goes backwards through this map applying the Chain Rule to find how the objective function result changes with respect to the design parameters. So for every design parameter, there is a gradient (i.e. a direction towards which it could be moved to decrease the value of the objective function).

8.4.4 Optimizer

The optimizers used are Stochastic Gradient Descent (SGD) and Adam. Both are pre-programmed in the pytorch package version 2.6.0.

SGD in this particular project was just the vanilla version, with no acceleration. In this case, the optimizer just always takes a step towards descent. It is the most straight-forward and easy to understand optimizer. However, it can take a long time to achieve results and get easily stuck in local minima. Regardless, it did produce some beautiful optimization runs and interesting topologies.

Adam is a more advanced optimizer. Since the setup of the entire code in the pytorch environment had already been done, it was easy to also try a different optimizer. Even though not enough research could be done to understand it, it was still valuable to use this opportunity to apply it.

In an optimizer, at the very least, a learning rate needs to be chosen. The learning rates tried in this project varied between 0.03 and 0.10. One forum said that getting NaN results during the optimization could be due to a learning rate that is too high. Thus the learning rate was lowered from the initial 0.10. This did help, as the same optimization could then be run for more iterations, but did not mean that eventually the NaN result would not pop up again. This was a struggle during this project, as due both to the complexity as well as the novelty of this approach, exacerbated by the lack of experience of the author, it was hard to pinpoint the major cause of NaN results. It could be the sigmoid function slopes, the learning rate, or anything else really.

The optimizer then decides how to change the design parameters. In the present work, these were actually weights that get multiplied by the starting configuration. Once these weights have been altered, they are used to define the material distribution for the next iteration. An overview of the entire workflow can be found on **Figure 8.19**.

8.4.5 Optimization inputs

Some of the parameters relating to the optimization are kept constant. They can be found on the following table:

| Stress weight | Displacement weight | # Gaussian Peaks | Amplitude | Level set | Slope in |
|---------------|---------------------|------------------|-----------|-----------|----------|
| 70 | 1 | 80 | 4 | 0.8 | 3 |

Table 8.1: Material distribution parameters kept constant

8.5 Summary of workflow

In a nutshell, the workflow consist of the following steps:

- Generate a Gaussian Field (3-D) defined by the x and y coordinates of each Gaussian peak (in the first iteration, this is done at random) and the same spread and amplitude applied to all of them (these are kept constant to encourage even thickness).
- Slice this field by two horizontal gradient planes by passing this field through an adapted sigmoid function. The adapted sigmoid is the sum of two sigmoid curves that slope in opposing directions. It is defined by the level set (lower plane), the shift (how much the higher plane is shifted upwards

from the lower plane) and the slopes of the two sigmoid functions that compose it. This generates a 2-D matrix of pseudo-densities.

- Use these pseudo-densities to scale the stiffness of the finite elements, and the self weight of the structure applied on the nodes of those elements.
- Apply the Finite Element Method to calculate the displacement field and tensile and compressive stress fields using principle stresses.
- Relax the stresses by multiplying them by the element density.
- Apply the Drucker-Prager criteria to combine the tensile and compressive stresses.
- Calculate the maximum stresses and displacement using p-norm on each of them.
- Combine the stress and displacement values with the volume for the weighted-sum objective function.
- Calculate the gradients of the objective function with respect to the input using backward() from pytorch to perform automatic differentiation.
- Take an optimizer step using Stochastic Gradient Descent (SGD) or Adam that changes the coordinates of the peaks as well as the spread (used in the landscape creation) and the shift (used in the slicing of the landscapes to form the topology)
- Iterate this process for a pre-defined number of iterations
- Post-process the structure to make it discrete (material or no-material) instead of pseudo-density based, while still retaining the desirable qualities of rounded edges, even thickness, and respect for the stress paths.

An expanded workflow diagram can be found on the next page.

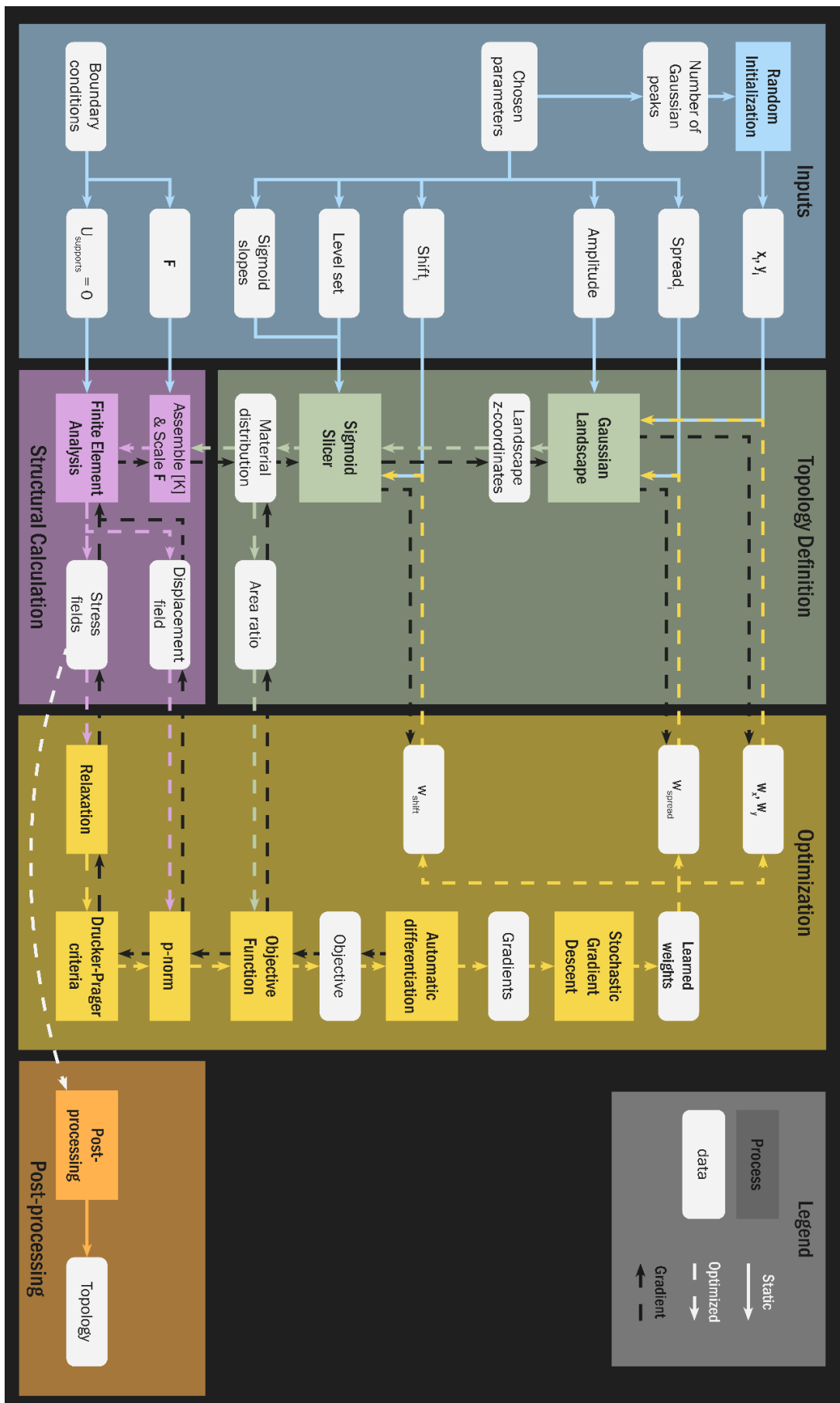


Figure 8.19: Workflow diagram

9 Results

The most promising results from the optimization are described in this chapter. One result from each optimizer is chosen for post-processing. The first subsection deals with the results achieved using the Stochastic Gradient Descent (SGD) optimizer, and the second deals with the results achieved using Adam.

9.1 SGD optimizer

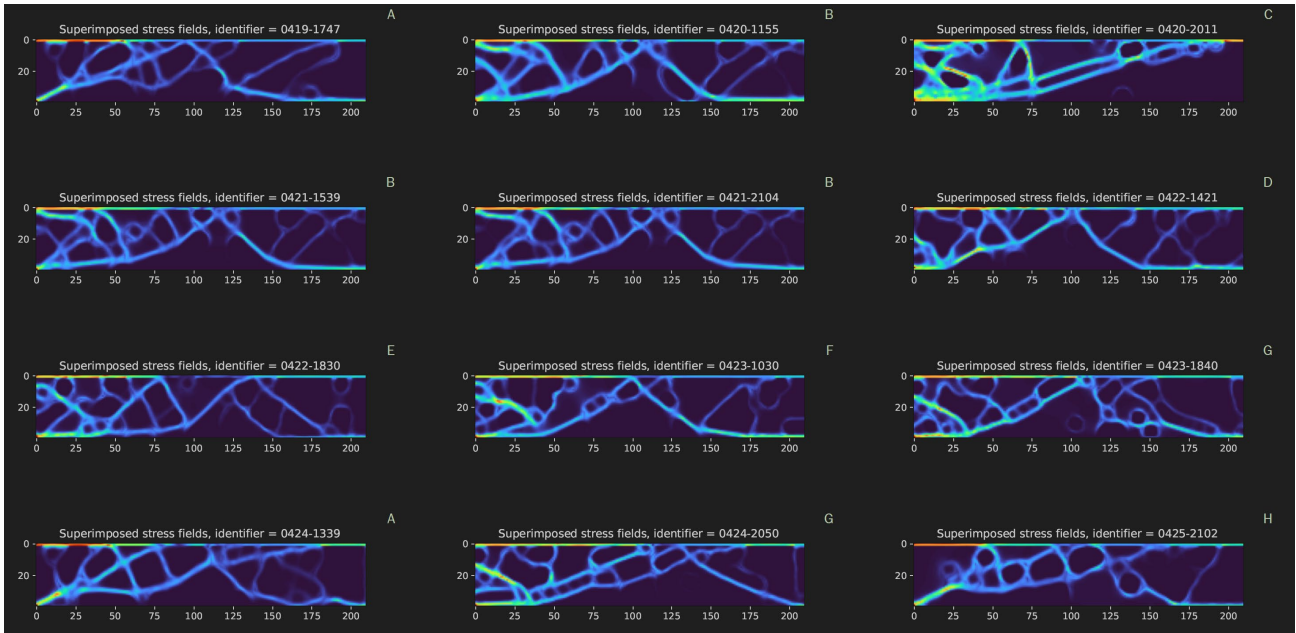


Figure 9.1: Visualization of the superimposed stress fields of the optimizations ran with SGD mentioned in **Table 9.1**.

Overall, the lowest objective result was achieved by optimization 0420-2011, masked in orange on **Table 9.1**. It also had the lowest maximum tensile stress. The superimposed stress fields of the optimization result are shown on the upper right corner of **Figure 9.1**. It was the only result achieved by SGD to resemble a pure cantilever (which, after the symmetry is applied, means a cantilever coming from each side and just *not* meeting in the middle), instead of a cantilever combined with a middle beam, as is seen in all the other optimization results.

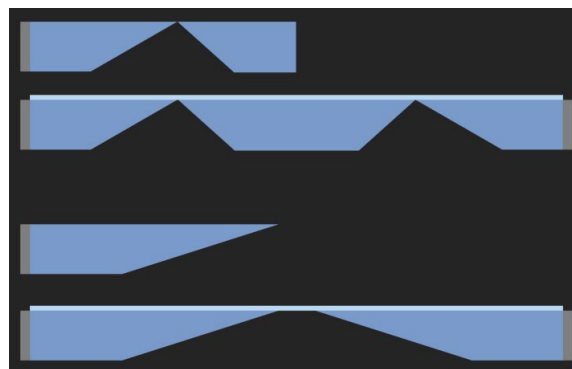


Figure 9.2: The two types of structural systems found in the optimized topologies. They are each shown halved first (as this is how the optimization was performed), and then reflected about the symmetry axis to cover the entire span. At the top, the cantilever plus middle beam. Below, the pure cantilever.

The table below offers an overview of the best performing optimization runs, sorted chronologically.

| Identifier | Objective | Area ratio | Comp. | Tension | Disp. | Run time | Seed |
|------------------|---------------|---------------|--------------------|--------------------|---------|----------|------|
| | [-] | [-] | kN/mm ² | kN/mm ² | mm | seconds | [-] |
| 0419-1747 | 0.3207 | 0.2187 | -0.0040 | 0.0018 | -0.1328 | 18409 | 03 |
| 0420-1155 | 0.2466 | 0.2904 | -0.0024 | 0.0011 | -0.0837 | 19337 | 06 |
| 0420-2011 | 0.2404 | 0.3155 | -0.0016 | 0.0008 | -0.0897 | 19233 | 07 |
| 0421-1539 | 0.2871 | 0.2500 | -0.0034 | 0.0013 | -0.0976 | 17978 | 06 |
| 0421-2104 | 0.3147 | 0.2263 | -0.0035 | 0.0014 | -0.1067 | 11583 | 06 |
| 0422-1421 | 0.3476 | 0.2135 | -0.0026 | 0.0016 | -0.1349 | 14780 | 12 |
| 0422-1830 | 0.3361 | 0.2419 | -0.0028 | 0.0015 | -0.1128 | 18001 | 13 |
| 0423-1030 | 0.3041 | 0.2255 | -0.0028 | 0.0012 | -0.1110 | 22602 | 14 |
| 0423-1840 | 0.3277 | 0.2333 | -0.0025 | 0.0013 | -0.1253 | 24224 | 16 |
| 0424-1339 | 0.3680 | 0.2306 | -0.0033 | 0.0017 | -0.1398 | 23620 | 03 |
| 0424-2050 | 0.3246 | 0.2352 | -0.0026 | 0.0012 | -0.1277 | 35926 | 16 |
| 0425-2102 | 0.3851 | 0.2251 | -0.0035 | 0.0017 | -0.1560 | 34817 | 18 |

Table 9.1: Optimization results using SGD optimizer from pytorch. Seed refers to the random initiation of the Gaussian Random Field. For runs with the same seed and different results, other input parameters were changed, such as the sigmoid slopes, initial Gaussian spread, or learning rate. Ideally, more structured trials would have been carried out. At least the boundary conditions, load case, and dimensions of the problem always remain the same.

The best performing optimization result in terms of having the lowest area ratio had a 0.2135 area ratio, marked in green on **Table 9.1**. (This might differ from the area ratio after post processing, since the optimization process works with pseudo-densities which still need to be translated into manufacturable discrete domains.)

It is interesting to notice that, for all optimization runs, the compressive stress is always higher (and roughly double) the tensile stress. This shows that the algorithm indeed prioritized the use of the compressive strength over the tensile strength. This was the intuition behind applying the Drucker-Prager criteria, and the results show its effect.

9.2 Adam optimizer

Further designs were generated using the Adam optimizer. This significantly reduced the computational time needed to achieve comparable results to those of SGD. It also created topologies that are more clear (i.e. where the separation of material and no-material is easier to discern) in comparison to those created with SGD.

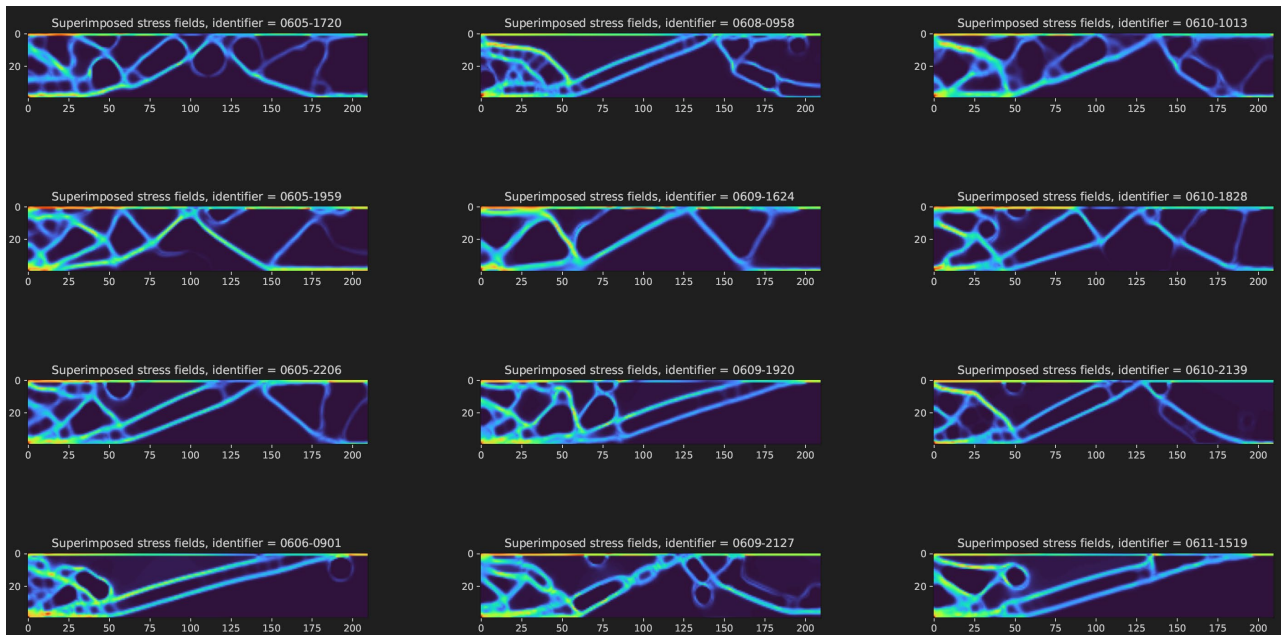


Figure 9.3: Visualization of the superimposed stress fields of the optimizations ran with Adam mentioned in **Table 9.2**.

| Identifier | Objective | Area ratio | Compression | Tension | Disp. | Run time | Seed |
|------------------|---------------|---------------|-------------|---------------|---------|----------|------|
| | [-] | [-] | kN/mm2 | kN/mm2 | mm | seconds | [-] |
| 0605-1720 | 0.3038 | 0.2118 | -0.0021 | 0.0013 | -0.1080 | 4059 | 13 |
| 0605-1959 | 0.2793 | 0.2369 | -0.0015 | 0.0010 | -0.0888 | 7508 | 03 |
| 0605-2206 | 0.2794 | 0.2236 | -0.0017 | 0.0010 | -0.0981 | 7694 | 04 |
| 0606-0901 | 0.2697 | 0.2222 | -0.0016 | 0.0008 | -0.1054 | 7422 | 05 |
| 0608-0958 | 0.2728 | 0.2388 | -0.0019 | 0.0008 | -0.0971 | 6882 | 06 |
| 0609-1624 | 0.2711 | 0.2149 | -0.0020 | 0.0010 | -0.0954 | 7729 | 07 |
| 0609-1920 | 0.2721 | 0.2203 | -0.0017 | 0.0009 | -0.1012 | 7518 | 08 |
| 0609-2127 | 0.3050 | 0.2910 | -0.0017 | 0.0009 | -0.0977 | 7484 | 09 |
| 0610-1013 | 0.2890 | 0.2461 | -0.0020 | 0.0010 | -0.0975 | 7292 | 12 |
| 0610-1828 | 0.2845 | 0.2188 | -0.0023 | 0.0011 | -0.1013 | 7212 | 20 |
| 0610-2139 | 0.3249 | 0.1919 | -0.0022 | 0.0010 | -0.1210 | 7152 | 07 |
| 0611-1519 | 0.2624 | 0.2263 | -0.0019 | 0.0008 | -0.0973 | 5357 | 07 |

Table 9.2: Results of the optimization using the Adam optimizer. Seed refers to the random seed used to define the initial Gaussian Random Field.

0610-2139 had exactly the same inputs (including the initialization of the random variables) as 0609-1624. The only difference between them was in the weight, α , of the area ratio in the objective function. For 0610-2139 $\alpha = 0.7$, while for all other optimization runs (including 0609-1624) $\alpha = 0.5$.

Quantitatively, this optimization run with a higher weight for the area ratio indeed yielded the lowest area ratio (0.1919) of all of the runs performed, as would be expected. It performed comparably to the other runs in terms of tensile stress, but did have the highest displacement value.

There was a second comparison done with 0609-1624. This was an alteration of the learning rate. Optimization 0611-1519 used a learning rate of 0.1 instead of the learning rate used in all other optimization runs of 0.05.

From the comparisons made above, it is apparent that changing one of the weights in the weighted objective function, or altering the learning rate, can have as drastic effects on the optimized topology as using a different random initialization for the (x, y) coordinates of the Gaussian peaks. The two comparison runs above would be difficult to distinguish from the other ones solely on a visual basis. That is to say, it would be hard to tell which three runs had the same random initialization just by looking at the results.

9.3 Proposed topologies

The objective function had three weighted terms. The topologies can be rated according to the result of the weighted sum, or any of its components: area ratio, stress, and displacement. The visual aspect of the final topology can also play a role in the topology chosen, since it should be fitting for its context. Additionally, structural and/or manufacturing concerns that are not included in the optimization process, can also affect the chosen topology.

The topologies chosen for post-processing are 0424-1339 (SGD, marked in pink on **Table 9.1**) and 0609-1624 (Adam, marked in pink on **Table 9.2**), one using each optimizer. They can be seen in **Figures 9.4 and 9.5**, respectively. Even though they were quantitatively not the best performing optimization runs, they did perform well in terms of ease of post-processing, and clarity of design. The loss graphs for both of them can be found in **Appendix 14.5**.

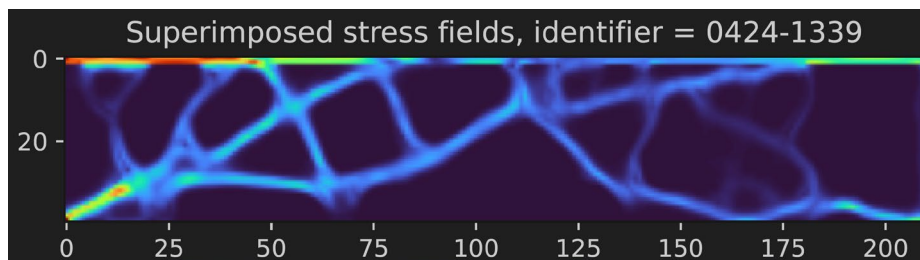


Figure 9.4: Superimposed stress fields resulting from optimization 0424-1339, which used SGD.

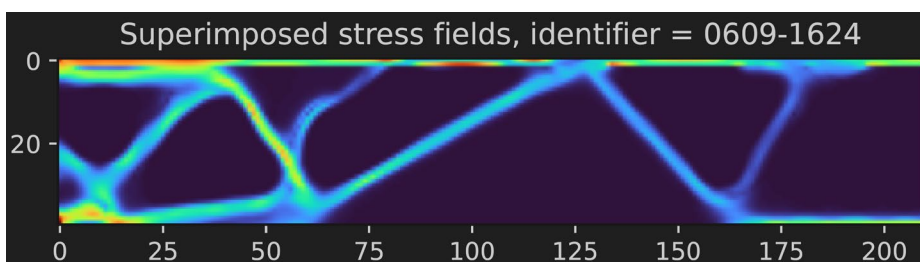


Figure 9.5: Superimposed stress fields resulting from optimization 0609-1624, which used Adam.

10 Post-processing

Some post-processing of the optimization output is required, since it does not output manufacturable, discrete domains. Some choices have to be made during the post-processing journey. They include: i) the basis for the final topology, ii) percentage threshold levels, iii) clean up of material residue, iv) check on manufacturing constraints (thickness ratio and gap size), and v) possible addition of material in fragile places.

Firstly, what should be the basis for the final topology? The first impulse might suggest that the basis for the final topology should be the material distribution output from the optimization. However, this is not the only way. During the exploration of the workflow of this project, images were generated of the evolution of the stress-fields during optimization. This was mesmerizing, structurally beautiful as well as informative. The Mentors for this thesis suggested using those stress fields as the basis of the final topology. This possibility was explored, and applied.

10.1 From superimposed stress fields to discrete topology

From the optimization process, there is an iteration which has the best performance (i.e. the lowest result for the objective function). This specific material distribution has corresponding compressive and tensile stress fields after the boundary conditions (supports and loads) are applied during FEA.

For each finite element, there is $\sigma_{comp,e}$, $\sigma_{tens,e}$.

These stress fields are post-processed as follows: firstly, they are each made into a percentage of the maximum stress of that field. So the range of each of them is between zero and one.

When dividing one negative compressive stress by the maximum negative compressive stress, it automatically becomes a positive ratio. The average is taken between the compressive and tensile ratios. This generates a field of superimposed stresses with values between zero and one. Since the value for compression is higher than tension, this implicitly overemphasises the amount of material in the final topology dedicated to tensile stresses, compared to just taking an average of the compressive and tensile stresses all together at once. **See Figure 10.1.**

Now, each finite element has superimposed stress value of:

$$\sigma_{sup,e} = \frac{\frac{\sigma_{comp,e}}{\sigma_{comp,max}} + \frac{\sigma_{tens,e}}{\sigma_{tens,max}}}{2}$$

However, peak stresses in both compression and tension are still very dominant, and skew the distribution, making it difficult to determine what a suitable boundary should be for the discrete topology. In order to make the distribution more even, the superimposed stress field is put through a function that raises the stress in each finite element to a power, p , that is less than one. This brings up the small values, while not increasing the larger values by much. The power used in this thesis was 0.13. $\sigma_{ega,e} = \sigma_{sup,e}^p$

The next step is to create a discrete topology from this field of stress ranges. This is done by choosing a percentage threshold to using as a cutting plane, c . This now ties back to a similar topology definition described in the Level-Set method.

$$\begin{aligned}\sigma_{ega,e} &> c, & e &\in \Omega(\text{material}) \\ \sigma_{ega,e} &< 0, & e &\notin \Omega(\text{void})\end{aligned}$$

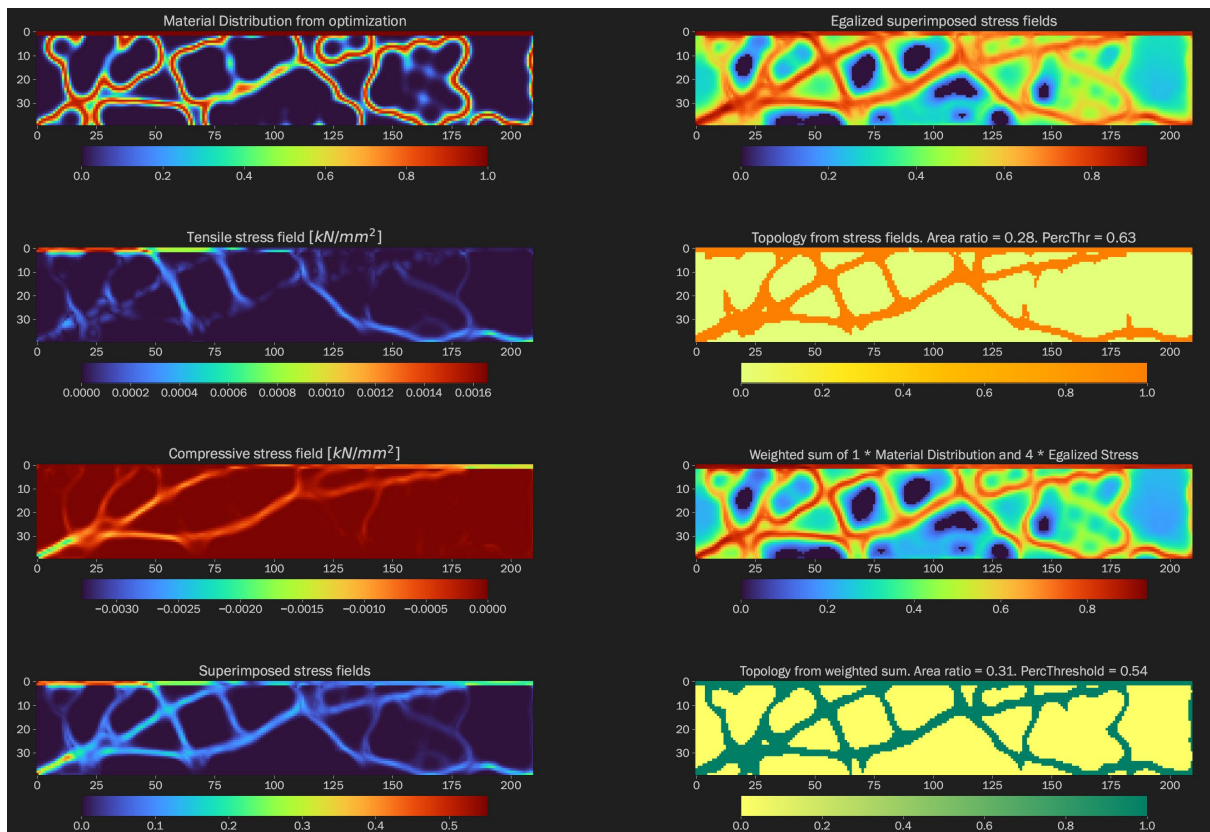


Figure 10.1: Creation of a discrete topology based on the superimposed stress fields. Optimization 0424-1339.

Once the discrete topology above was achieved, then some final adjustments were made by hand. The little sharp edges that were reminding of weak stress connections were removed, a couple of finite elements were filled in which provided some further improvement in the stress performance, and the manufacturing criteria was checked, with small adjustments being made to ensure that it was fulfilled. This gave the final discretised topology, with the corresponding stress and displacement performance.

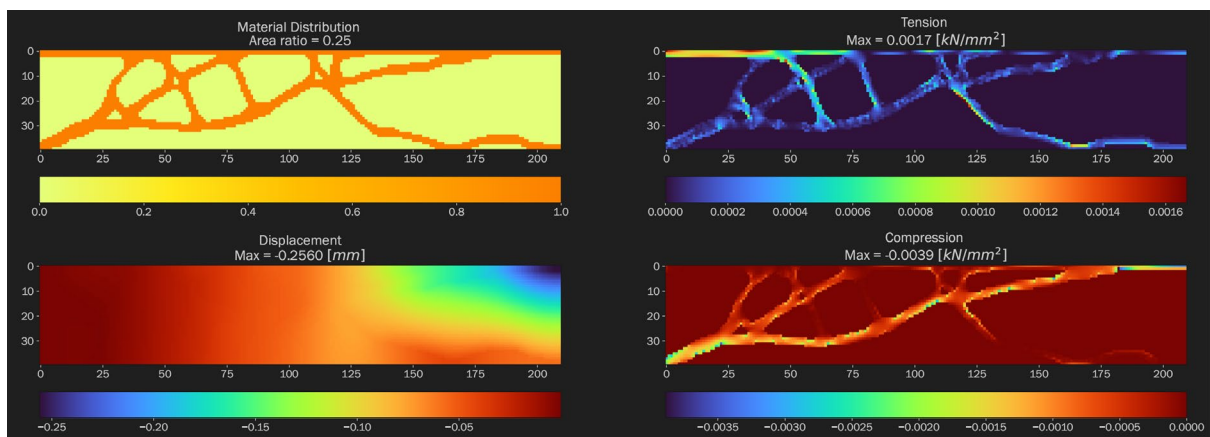


Figure 10.2: Clean discrete material distribution based on the superimposed stress fields and its corresponding structural performance.

A different way of achieving the discrete topology was considered. It relied on a weighted sum of the material distribution and the stress fields. It can still be seen on the right bottom corner of **Figure 10.1**, but was not used for further post-processing.

For the topology achieved with Adam, the same process was followed. **Figure 10.3** shows the creation of the discrete material distribution. What is noticeably different here is that the material distribution (top

left graph) is already relatively discrete. Thus in this case, two different versions of the discrete topology definition were explored. The first is based on the superimposed stress fields, as was done with the SGD optimization, shown on the second graph of the right column in orange. The second is using only the material distribution. It is shown on the bottom graph of the second column in green. **Figure 10.4** shows them side-by-side. They resemble each other greatly, and have the same area ratio. However, the one based on the stress fields actually has smoother transitions than the one based on the material distribution. It also has more linear members when compared to its counterpart. Thus, the final topology chosen is based on the stress field.

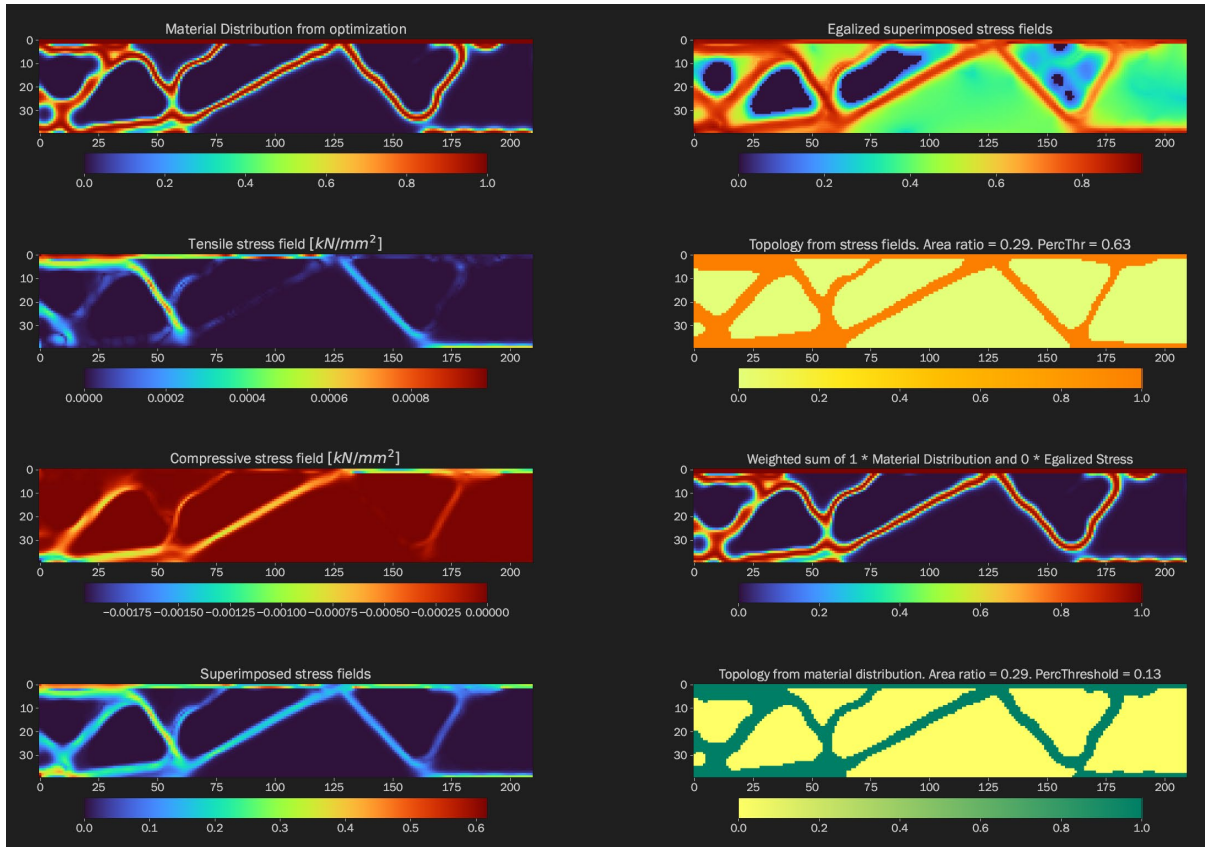


Figure 10.3: From material distribution and superimposed stress fields to discrete topology, optimization 0609-1624.

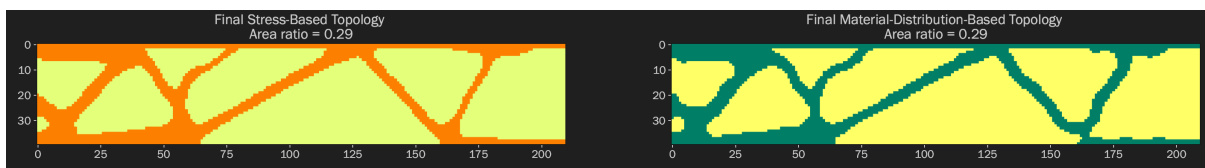


Figure 10.4: Comparison of stress-field-based and material-distribution-based discrete topologies with the same area ratio.

To verify that indeed that is the best choice, and to quantify the difference in performance between them, the stress check was performed for both. **Figure 10.5** shows the stress fields of the discrete topology achieved through manipulation of the stress fields gathered from the optimization process, while **Figure 10.6** shows the stress fields of the material distribution-based topology.

It is indeed so that the former performs better than the later. The curviness of the members in the material distribution-based topology causes more peak stresses throughout the structure (shown in dark red on the graph). However, their performance is comparable, with the stress-field-based outperforming the

material-distribution-based by only 0.0002 kN/mm^2 in both tension and compression. These theoretical differences seem too small though. In reality, creating the smooth transitions achieved by the stress-based in the detailing of the structural design would almost certainly be a *much* better choice for the longevity of the structure.

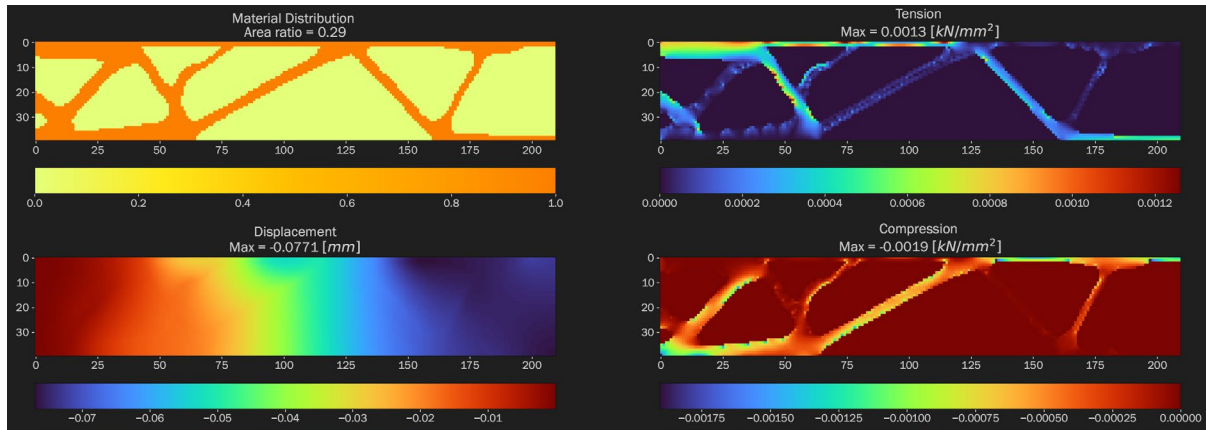


Figure 10.5: Performance of stress-field-based discrete topology.

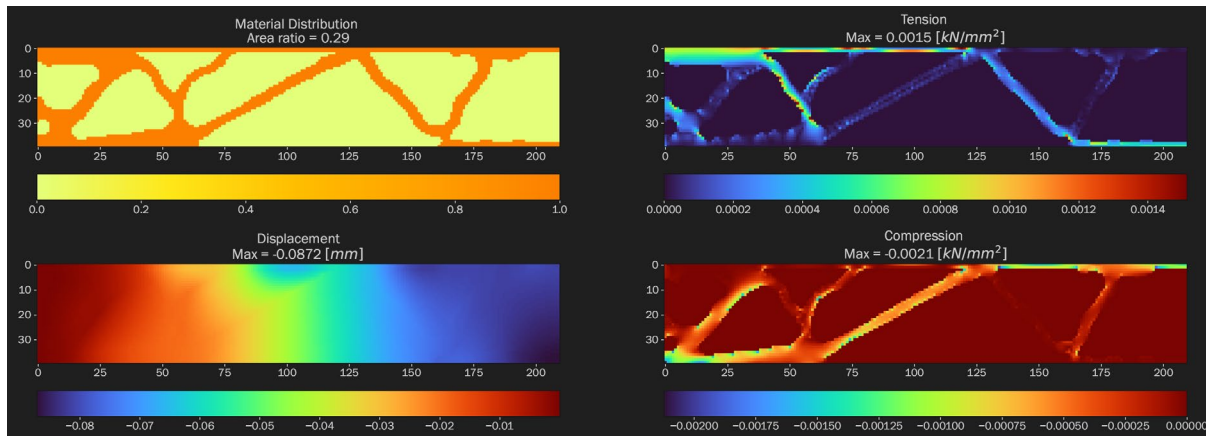


Figure 10.6: Performance of material-distribution-based discrete topology.

Another noticeable difference between the result achieved through SGD and the one achieved through Adam is that the former needed manual cleaning up of the discrete topology, while the later did not. There were no “loose ends” present in the structure. The only manual intervention needed to define the discrete topology after the optimization process what completed was to decide on what percentage threshold to use. (In this case 0.63 was used. See **Figure 10.3**.)

On the other hand, the topology produced with Adam does have a larger maximum cross section than its SGD counterpart. Thus the annealing constraint should be checked and might not be fulfilled in this case. The soft constraint was not enough here, as the intermediate densities were used, widening of the stress paths over the “edge” of the material distribution paths.

The last and final step was to smoothen out the edges to create the final topologies. This was done manually. The results can be seen in **Figure 10.7**.

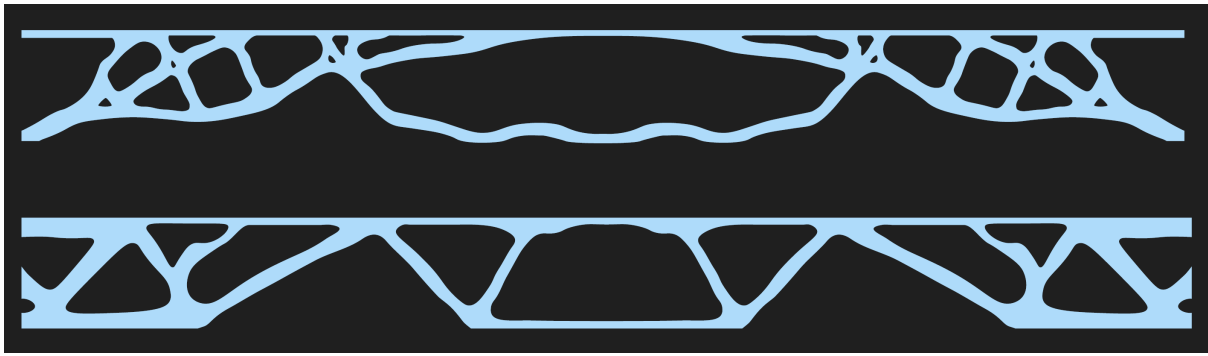


Figure 10.7: Final smooth and discrete topologies. Top: achieved through using SGD, optimization 0424-1339. Bottom: achieved through using Adam, optimization 0609-1624.

10.2 Post-processed design performance

| | | SGD – 0424-1339 | | Adam – 0609-1624 | | Design constraints |
|----------------------|-----|------------------------|-----------------------|------------------------|-----------------------|--------------------|
| | | Before post-processing | After post-processing | Before post-processing | After post-processing | |
| Area ratio | [-] | 0.23 | 0.25 | 0.21 | 0.29 | NA |
| Max principal stress | MPa | 1.7 | 1.7 | 1.0 | 1.3 | 6.4 |
| Min principal stress | MPa | -3.3 | -3.9 | -2.0 | -1.9 | -50 |
| Max deflection | mm | 0.14 | 0.26 | 0.10 | 0.08 | 8.4 |

Table 10.1: Performance before and after post-processing.

Table 10.1 shows the performance of the two designs before and after post-processing. Adam came up with a better performing topology than SGD in terms of stresses and displacement. Adam's topology also looks more structurally intuitive. SGD has a smaller area ratio and conforms better to the annealing constraint.

The topologies are validated with ANSYS. The results can be found in **Appendix 14.2**.

11 Conclusion

11.1 Design in context

To conclude the project, the two final designs are presented in the context of the Case Study. In **Figures 11.1 and 11.3**, the result achieved with SGD can be seen, and in **Figure 11.2**, the one achieved with Adam. The research goals will be answered in the following subsection and the Discussion of the results and algorithm will follow in the next chapter.

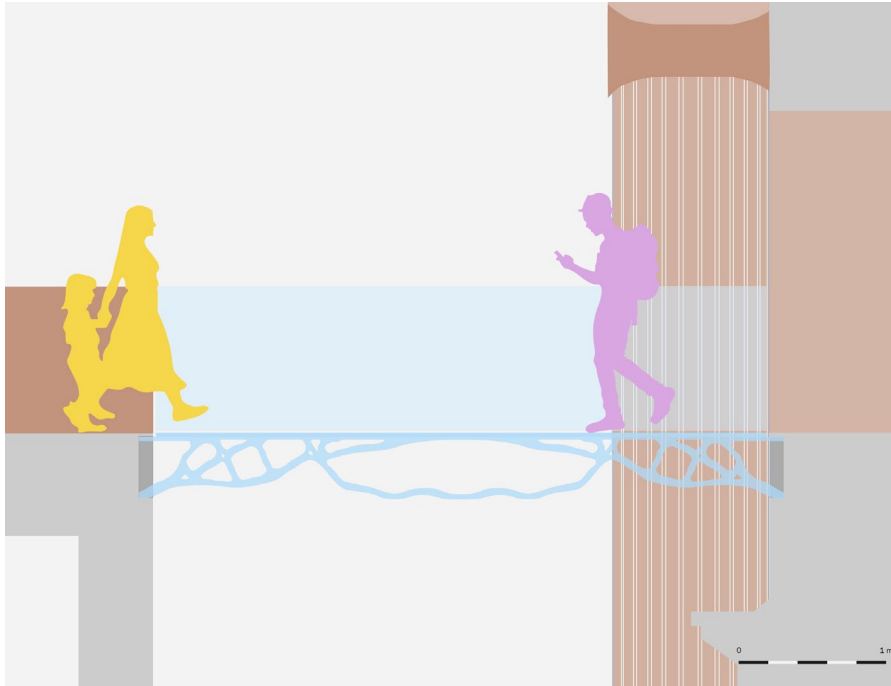


Figure 11.1: SGD optimized design in context.

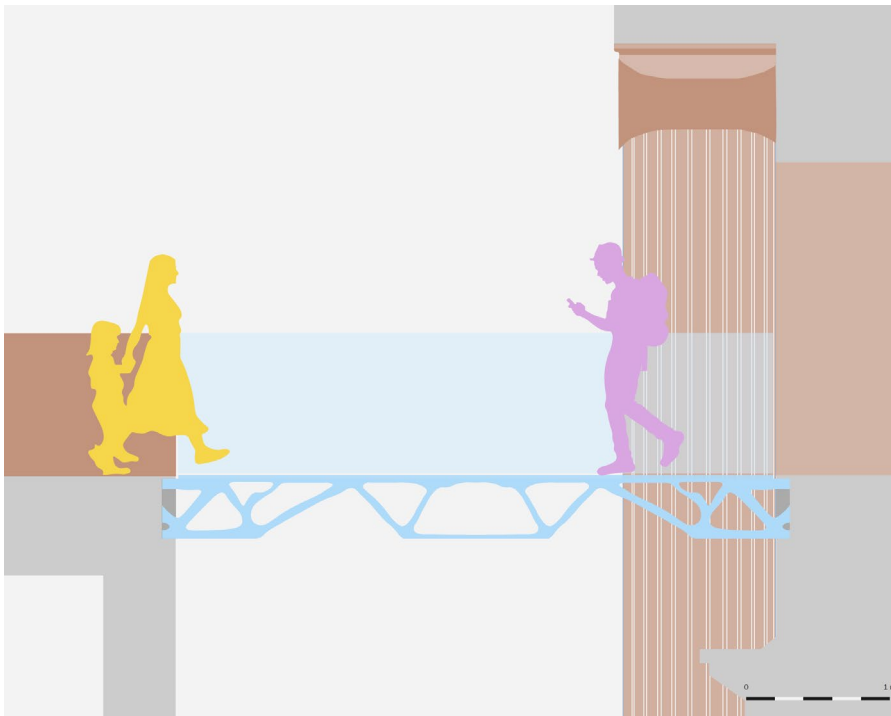


Figure 11.2: Adam optimized design in context.



Figure 11.3: Render of SGD in context, by Lara Neuhaus.

11.2 Research goals

The research goal is to design:

(1) *A material distribution definition, specifically tailored to cast glass.*

This design goal had some successes and some areas which could use improvement. Overall, the algorithm was indeed able to create paths that were used by the stresses during optimization. And the material distribution had the added benefit of being defined by only about 2% of the parameters that would be needed by the SIMP method for the same discretization accuracy. These are considerable wins!

On the other hand, the algorithm could also lump peaks together to form thicker blobs. This was possible because the algorithm was also given full control over the shift parameter of the material distribution, which allows it to fill up the blob. More detailed algorithm design could be implemented here to pose a constraint on the shift parameter and ensure more even sections, even when the Gaussian peaks are close together.

(2) *The optimization algorithm, specifically tailored to brittle materials*

Creating an algorithm specifically tailored to brittle materials involved not only the inclusion of a brittle material stress criterion (i.e. Drucker-Prager criteria) as was already done in the previous thesis, but also the inclusion of stress in the objective function, which reflects the fact that brittle materials tend to fail due to peak stresses. This part was achieved successfully, and the loss function graphs obtained during the optimization clearly show that the tensile peak stresses are getting minimized at each iteration. Furthermore, the comparison of the peak stresses with previous explorations of the same case study shows that the stress has indeed been greatly reduced.

(3) *A workflow that combines material constraints, topology (a.k.a. material distribution) definition, and optimization algorithm.*

Bringing it all together was also one of the major challenges. The switch to using the adapted sigmoid curve to define the topology tied the optimization loop together. It allowed the use of automatic differentiation for the optimization process and was a pivotal point in this thesis. The workflow is differentiable and through the optimization process one can see that the algorithm has access to the topology definition parameters and changes them in a reasonable direction.

The addition of the adapted sigmoid curve made it all possible, but also created a challenge that remained only partially explored. That is the precise definition of the two slopes of the adapted sigmoid. Some slope values were used and in the end results were obtained, but the algorithm did at times become numerically unstable and return NaN values. This might be due to the gradient calculation encountering slopes that tend to infinity, and the sigmoid slopes could be a contributing factor. More research here would definitely be advisable.

(4) Use the above to inform the design of an indoor cast glass pedestrian bridge at the British Museum.

The design of the indoor pedestrian bridge was also achieved. The final design chosen is a visual mix of structured material with an ode to the material definition method in the curvy middle span.

11.3 Research question

How can a (two-dimensional) stress-based optimization algorithm be set up for designing a massive cast glass structural element that takes into account manufacturing constraints and the brittle nature of glass?

The algorithm can be created as a differentiable workflow, which defines the material distribution during the optimization with a Gaussian field that gets sliced by an adapted sigmoid curve before undergoing FEA to quantify a weighted-sum objective function of stress, area, and displacement, to which automatic differentiation is applied and used by an optimizer (SGD or Adam) to adapt the parametric inputs of the material distribution with respect to the result of the objective function.

12 Discussion

12.1 Evaluation of design

In this section, the two topologies will be evaluated from an engineering intuition point of view. Firstly, Figure 12.1 shows the final results achieved with SGD.



Figure 12.1: Final smooth and discrete topology achieved through using SGD, optimization 0424-1339.

It is apparent that the methodology shines through the topology. The intention to create the desired rounded transitions also created structurally inefficient wavy parts in the structure.

The optimized topology could be made more efficient by implementing changes gathered by visual inspection/engineering intuition: the wavy connection at the bottom of the structural element could be made waveless. This is because when the shape of the element does not match the stress path, stress concentrations arise.

The reason for the algorithm not adjusting this itself might be that it is concerned with the *peak* stress minimization (obtained from the p-norm). Since the peak tensile stress calculated *during* the optimization is at the supports (upper part of the connection, see **Figure 10.1** second graph on the first column), it might have had little incentive to change the less-stressed middle section. However, the stress analysis with the discrete topology, does show peak stress concentrations at the waves (see **Figure 10.2**). Perhaps a punishment on the stiffness of intermediate densities might have helped to bring the pseudo-density and discrete stress fields (calculated during and after the optimization, respectively) closer together. Which in turn might have encouraged the algorithm to address the stresses created in the wavy part.

From another point of view, the unexpected shape of the structural member could also be perceived as a positive quality, depending on the context. For the case study presented here, the location is a museum, so a structure which is more expressive could be seen as a welcome addition. The motivation for the use of glass as a structural material tends to have a high decorative component. Thus, creating topologies that are more playful and unexpected than a strictly structurally efficient ones might be valuable.

Here, the decision of the designer comes in: what is the intention of the structure? How does it fit with the other architectural components in the space? Luckily, this workflow can generate plenty of options.

Still, it remains true that the topology could be further optimized to achieve better performance. It might be too constrained by the rounded geometry of the Gaussian, not have enough incentive in the objective function to alleviate stresses that are not the peak stresses, or there might be a mismatch between optimizer and problem.



Figure 12.2: Final smooth and discrete topology achieved through using Adam, optimization 0609-1624.

The following topology, the one produced by the Adam optimizer, can be seen in **Figure 12.2**. From a structural intuition point of view, it “looks better”. But it still presents a waviness in one diagonal member (when reflected, they are two).

Here, it becomes clear that the algorithm *can* create stiff triangles (even from the rounded Gaussians). This structure performs better than the previous one in terms of stress and displacement.

However, since the manufacturing constraint was only softly enforced, and no maximum thickness check was performed during the optimization, it might take too long to anneal. Its largest cross section might also be more than three times the smallest one, which might cause the creation of residual stresses beyond the allowable limit. To remedy this, perhaps there should be a hard enforcement of the manufacturing criteria. Or, at least, a post-processing automated check to verify the result by calculating the annealing time and ration between the thickest and thinnest cross sections.

On the positive side, the cross-sections did indeed remain fairly even (although more precise quantification of their thickness and annealing time could be performed), and the transitions were indeed smooth and rounded.

12.2 Comparison of results

Koniari (2022) and Schoenmaker (2023) also explored the same Case Study, their designs can be seen in **Figure 12.3**. The topologies obtained differ qualitatively and quantitatively from both the works of Koniari (2002) as well as Schoenmaker (2023). The workflow characteristics and structural performance of the results can be found on **Tables 12.1** and **12.2**, respectively.

| | unit | Koniari, 2022 | Schoenmaker, 2023 | SGD | Adam | Design constraints |
|----------------------|------|-----------------------------------|-----------------------------------|------------------------------------|------------------------------------|-----------------------|
| Dimensions | # | 2 | 3 | 2 | 2 | NA |
| Design domain height | mm | 300 | 300 | 400 | 400 | NA |
| Objective | [-] | Volume | Volume | Weighted sum | Weighted sum | NA |
| Constraints | [-] | Compliance, Stress, Manufacturing | Compliance, Stress, Manufacturing | Soft manufacturing | Soft manufacturing | Stress, Manufacturing |
| Methodology | NA | SIMP | SIMP | Level-Set | Level-Set | NA |
| Load application | NA | Same throughout process | Same throughout process | Depending on material distribution | Depending on material distribution | NA |

Table 12.1 Comparison of dimensions, objectives, constraints, methodology, and load for the different workflows. SGD and Adam refer to the results achieved in the present work. Weighted sum refers to the sum of weighted volume (i.e. area ratio), stress and displacement as discussed in Chapter 8.4.

Firstly, it is important to understand the differences in dimensions, objectives, constraints, methodology, and load application of each of the workflows. In terms of the number of dimensions, Schoenmaker’s (2023) work is the only one that goes beyond two dimensions, performing a 3-D optimization. The other difference in the dimensions is regarding the design domain height. Koniari’s (2022) and Schoenmaker’s

(2023) algorithm had access to a design domain that was 300 mm high, while the present work increased that value to 400 mm.

Secondly, the workflows can be split into two groups in terms of the objective and constraints. Koniari (2022) and Schoenmaker (2023) both used volume as the objective to be minimized, while the present work used the weighted sum of volume, stress and displacement, as discussed in Chapter 8.4. Additionally, Koniari (2022) and Schoenmaker (2023) enforced the manufacturing constraint during the optimization using filters and the calculation of annealing time and thickness ratios. This was a hard constraint. The present work, on the other hand, only enforced a soft constraint on the manufacturing criteria, which was embedded in the material distribution definition, as discussed in Chapter 8.2. All methodologies were constrained by $F = KU$, and pseudo-densities ranging between ρ_{min} and 1.

Thirdly, a major difference between the workflows is the methodology. Koniari (2022) and Schoemaker (2023) used SIMP, while the present work used the LSM.

In terms of the loads applied, the groups are split in the same way as for the objectives and constraints, as well as methodologies. With Koniari (2022) and Schoenmaker (2023) applying an even permanent load throughout the optimization process. This permanent load is added to the temporary load and applied evenly through the design domain. In contrast, the present work adjusts the permanent load to reflect the material distribution at each iteration, and applies it to the nodes surrounding the finite element where the material is found, as discussed in Chapter 8.3. It also distinguishes between temporary and permanent load by applying the former to the top of the structure. All methodologies used the same material properties for borosilicate cast glass, temporary loads and safety factors for the loads.

Now that the differences and similarities between the methodologies have been established, the structural performance of the results will be compared.

| | unit | Koniari, 2022 | Schoenmaker, 2023 | SGD | Adam | Design constraints |
|--------------------------|------|---------------|-------------------|------|------|--------------------|
| Volume ratio | [-] | 0.3 | 0.23 | 0.25 | 0.29 | NA |
| Maximum principal stress | MPa | 2.96 | 4.7 | 1.7 | 1.3 | 6.4 |
| Minimum principal stress | MPa | -5.56 | -4.7 | -3.9 | -1.9 | -50 |
| Deflection | mm | 0.12 | 0.24 | 0.26 | 0.08 | 8.4 |

Table 12.2 Comparison of the (structural) performance achieved through the different workflows.

In terms of the volume ratio, Schoenmaker's (2023) design performs the best, exhibiting a volume ratio of only 23%. This might be due to the inclusion of the third dimension, which makes it possible for the algorithm to allocate material differently for each layer of the design. Load paths can then travel not only on the x-y plane (vertical and horizontal) but also along the z-dimension, in the depth of the design. This makes it possible for material to be allocated to more layers in parts of the structure that really need it, but in less layers for the parts of the structure where less material is able to transfer all the load.

Schoenmaker's design is also qualitatively different from the others, exhibiting an I-beam-like shape with an upper and lower flange. However, it also has the diagonal bars close to the central symmetry axis connecting the upper and lower flanges, as the work of Koniari (2022) and the present work do. Another

similarity is the reduction of the material used about a third of the span from the support. At this position, the work of Schoenmaker shows the gaps on the top surface, and the present work and Koniari's show the hinge. The design domain of Schoenmaker's work was different from the other works considered here, which might account for the differences in result. The creation of the I-beam can be responsible for the equal tensile and compressive stresses. Still, the quantitative results are very comparable, and the addition of the third dimension only seems to provide an advantage in terms of the volume.

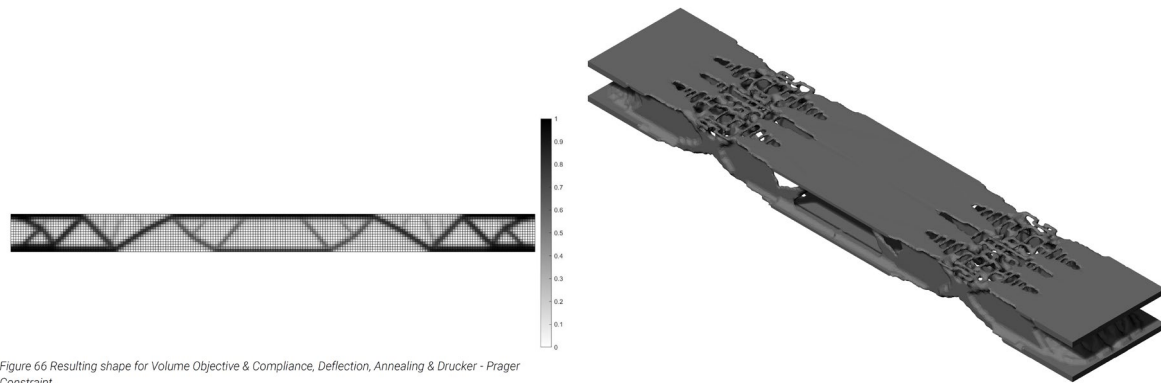


Figure 12.3: Optimized topologies by Koniari (2022) on the left, and Schoenmaker (2023) on the right.

Schoenmaker's design is also qualitatively different from the others, exhibiting an I-beam-like shape with an upper and lower flange. However, it also has the diagonal bars close to the central symmetry axis connecting the upper and lower flanges, as the work of Koniari (2022) and the present work do. Another similarity is the reduction of the structural height about a quarter of the span from the support. The design domain of Schoenmaker's work was different from the other works considered here, which might account for the differences in result. The creation of the I-beam can be responsible for the equal tensile and compressive stresses. Still, the quantitative results are very comparable, and the addition of the third dimension only seems to provide an advantage in terms of the volume.

At the other hand of the spectrum, Koniari's (2022) had the highest volume ratio of 30%. However, since Adam had a larger design domain, it actually was design outcome that uses the most material, even with the slightly lower volume ratio of 29%.

Further deepening into the comparison between Koniari (2022) and Adam, they had the stiffest designs. In **Figure 12.4**, one can see the similarity of the designs. The structural concept of an overhang from each side hinged to a beam in the middle applies to both of them. Even the positioning of the elements forming the truss-like structure is similar, as well as the distance between the hinge location and the support.

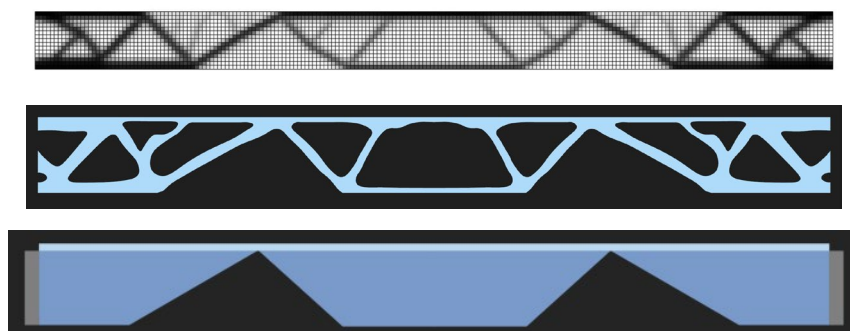


Figure 12.4: Comparison between Koniari (2022) (up) and Adam (middle). Structural scheme at the bottom.

The lower maximum displacement (which reflects an increased stiffness) of Adam in comparison with Koniari (2022) could be due to Adam having: 1) the larger structural height at its disposal, and/or 2) used a higher amount of material.

In terms of stress, the design created by applying the Adam optimizer significantly outperforms that of Koniari (2022), which makes sense given that Adam performs a stress-based optimization. The propensity to lower stress concentrations might be reflected in the physical structure in two ways: 1) again, the larger structural height, which provides more leverage to the structural element and could not be changed by the optimizer, and/or 2) the thicker connection between bars present in the Adam design, which *can* be altered by the optimizers.

Considering this second point in isolation, one can postulate that the methodology of the present work had positive effects on the result. Particularly, the creation of the material distribution with the rounded Gaussian fields, and the intermediate densities around the boundary of the topology, combined with the definition of the final, discrete topology based on the stress paths, has created these smooth transitions between the linear members. Stress peaks are usually found at abrupt cross sectional changes, and that is alleviated here.

Figure 12.5 suggests that this intuition is correct. It is the ANSYS validation of the design by Koniari (2022) and it shows the peak tensile stresses (marked in red) at the transition points between the linear members.

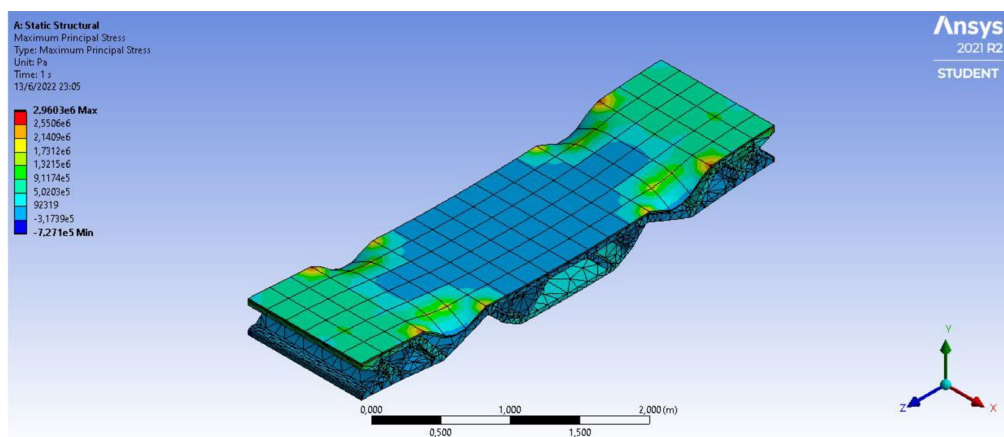


Figure 12.5: Maximum principal stress in the design by Koniari (2022).

Looking at the volume-stress relationship from another point of view, it could also be possible that the result achieved by Adam has more volume than that achieved by Koniari (2022) *because* of the stress term present in the objective of the former. Koniari (2022) could further reduce the volume without punishment, as long as it remained within the stress constraint, while Adam had to balance the volume and stress reductions, since they were both present in its objective. This indicates that, in case a structure with yet less volume is desired, perhaps the relationship between the weights applied to the volume and stress terms should be reconsidered, increasing the former in relation to the later.

The manufacturing constraint is very important for cast glass, given its annealing criteria. In **Figure 12.4**, the cross section of the work of Koniari (2022) does seem more even than the present work. The one curve diagonal element produced by Adam is perhaps the weakest point in the comparison of the two designs. In terms of the other parts of the design, it seems like the soft enforcement of the manufacturing criteria was enough. As mentioned in the previous subsection, the inclusion of tests to quantify its manufacturing performance would be a welcome addition.

12.3 Algorithmic performance

Apart from the performance of the design itself, there is also the performance of the algorithm used to calculate it. One can also look at the intersection of the two and ask the question: Is the algorithm a good representation of the structural design problem?

12.3.1 Computational complexity

Firstly, the performance of the algorithm itself. **Table 12.3** shows a comparative overview of the computational complexity. There is a startling reduction in the number of design parameters when compared to the works of Koniari (2022) and Schoenmaker (2023). This reduces the computational complexity.

| | Unit | Koniari, 2022 | Schoenmaker, 2023 | This thesis |
|--------------------|------|---------------|-------------------|--------------------------------|
| Structural height | mm | 300 | 300 | 400 |
| Dimensions | # | 2 | 3 | 2 |
| FE edge | mm | 20 | 20 | 10 |
| Parameters | # | 1 575 | 31 500 | 162 |
| Nodes | # | 1 696 | 35 616 | 8 651 |
| Degrees of freedom | # | 3 392 | 106 848 | 17 302 |
| Objective | type | Volume | Volume | Stress + Volume + Displacement |

Table 12.3: Computational complexity.

In terms of nodes and degrees of freedom, the current work sits somewhere in between the works of Koniari and Schoenmaker. It has more degrees of freedom than the work of Koniari, even though both are 2-D optimizations, because it has finite elements that are a quarter of the size (i.e. the side dimensions are halved), and the structural height is 33% taller, going from 300 to 400mm.

The objective function increases the computational complexity, when compared to the two other works, by containing the local quantities of stress and displacement.

The results were still achieved using less computational power, see **Table 12.4**. All optimizations were performed on a Laptop's CPU and in the same (SGD) or less (Adam) time than the other explorations.

| | Unit | Koniari, 2022 | Schoenmaker, 2023 | This thesis |
|----------------|-------|---------------|-------------------|----------------------|
| Computer used | name | DelftBlue | DelftBlue | Laptop's CPU |
| CPU cores | # | 17,000 | 17,000 | 8 |
| Computing time | hours | 46 | 8 | 7 (SGD), 2.14 (Adam) |
| Code language | name | MATLAB | MATLAB | Python |

Table 12.4: Computational power used.

In terms of the time taken for the optimization, the process ran for a pre-defined number of iterations, so it kept going even after being relatively stable. Furthermore, the making of pdf images to visualize the process (loss graphs and structural performance fields) was done within the loop, which also costs time.

With regards to the code written specifically for this project, the python code is slower than the MATLAB one for performing the material distribution and FEA processes. There might be many reasons for this. But one of them might be that in this particular python code, there are a lot of reshape/transpose operations performed on the tensors/matrices. That is because the code was written in MATLAB first, and MATLAB performs operations down the columns, while python does so along the rows. So when, for example, a matrix needed to be turned into a vector, the translation of the code into python first needed to transpose the matrix and then get each value to compose the vector, so that the result would be the same as the one achieved by MATLAB. If the python code were to be written from scratch again, the numbering of the elements and nodes for the FEA would be done along the rows instead of along the columns.

The new method enables the optimization to be run with less computational power. One does not need to rely on a supercomputer, which is usually an external, shared tool. But rather can use a laptop, and perform optimization runs in less time than the supercomputer did. This allows for experimentation with the optimization parameters (learning rates, optimizers, design domain dimensions, etc.), achievement of multiple design variations from the random initial guesses, and more independence in the process.

The drawbacks are the predisposition to achieve local instead of global minima, the wavy parts of the structural element.

12.3.2 Is the algorithm a good representation of the problem?

This section explores some relationships between the structural problem and the algorithm.

First of all, a look into the material distribution. During optimization, the idea is to create some structure to be evaluated at each iteration. This structure has some material parts (representing the actual structure) and some no-material parts (representing the unused sections of the design domain). As seen before, this material distribution is not discrete during optimization, but a range between 0 and 1. This distribution is defined by, amongst other factors, the slopes of the sigmoid curve. This is a tricky calibration: it cannot be too steep (or the algorithm will get very steep gradients and not be able to progress smoothly) nor too shallow (or there will be a large pool of intermediate densities around the topology).

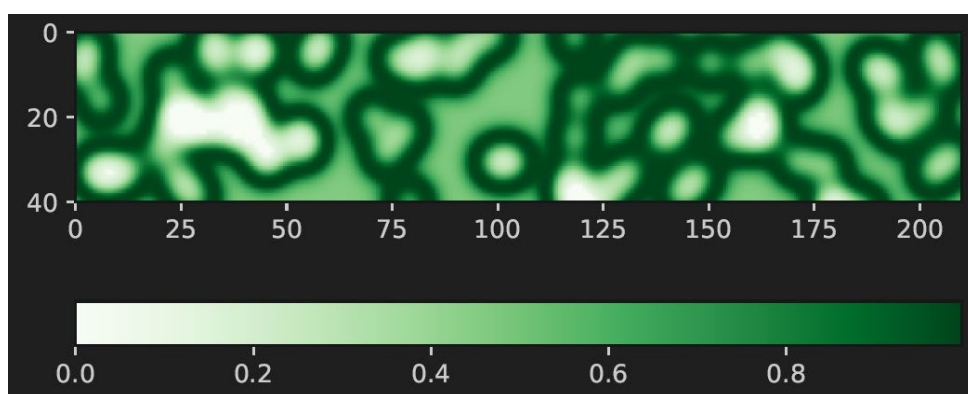


Figure 12.6: A random material distribution raised to the power of 0.13.

Empirical tests were done to find the steepest possible slopes for in and out which would still work smoothly with the algorithm. However, they still do not yield equal pseudo densities on the inside and on the outside of the topologies (i.e. those that are high up on the peaks, and those that are low on the valleys differ). This can also affect the optimization itself. Since the inner pseudo densities are lower, the algorithm (while minimizing the volume which is defined as a sum of all densities) might also tend towards

having more of the design domain on the inside than on the outside of the topology. In Figure 12.6, where a random material distribution has been raised to the power of 0.13, the white areas represent the lowest densities. They are found on the “inside” of the peaks. They are assigned “no material” because of the second slicing of the landscape, the cutting off of the peaks. The darkest areas are the material distribution itself. If one looks at those areas, one sees that to one side of them there are the white areas, and to the other side there are the light green areas. The “no material” zone defined on the “outside” of the peaks is not as empty as the one defined on the “inside”. But this does not represent any actual physical quality of the system. It is just an unfortunate consequence of the current workings of this workflow. Ideally, both “no-material” zones would have the same pseudo densities, regardless of whether they are on the inside or outside of the material paths.

This discrepancy between inside and outside, and the relatively “high” density of the “no material” zones on the outside of the peaks creates problems for the optimization. For example, optimization 0606-0901 had the lowest tensile stress, second best overall score, and area, which are the main goals here. So it was first chosen for post-processing. However, the discrete material distribution stress check suggests that it was relying too much on the low densities present in the triangle between the top layer and the reaching diagonal arm. **Figure 12.7** shows that the discrete topology presents high tensile stress at the top layer, that does not conform to the stress constraint of 0.0064 kN/mm², even though the stress calculations using pseudo-densities during the optimization had predicted a very low stress of 0.0008 kN/mm².

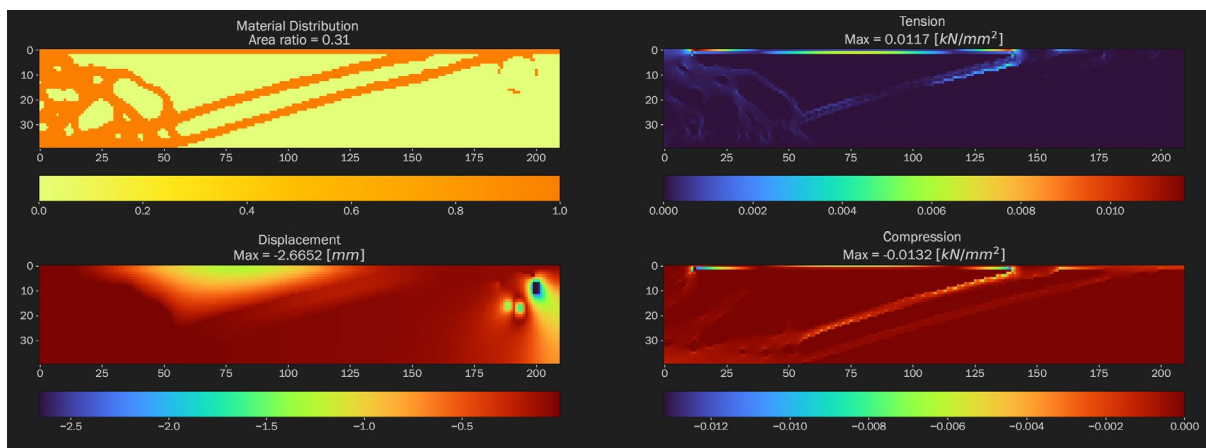


Figure 12.7: High tensile stress at the top of optimization 0606-0901.

The same holds for the other well-performing optimization in terms of stress, 0608-0958. This points to the need of working on the interaction between the lower part of the landscape and the sigmoid slope that slices through it, so that a yet lower density is achievable for the “empty” areas outside of the peaks.

Another undesirable consequence of this “high” pseudo density on the outside of the peaks is hanging material. This can be seen on **Figure 12.8**. This same figure also shows another problem with the workflow: sometimes it creates paths that are unused. We see on the Material distribution graph, that around point (45, 35) there is a circle of material. Of this circle, only the upper part is used to connect the diagonal arms. The rest is just “there”, unused. This is not ideal and also leads to the wrong quantification of how much material is actually needed in this design variation.

Now onto some positive observations on the same image. By looking at the image of the Gaussian landscape, one can see that the algorithm indeed uses the gaussians to draw a structural scheme. This means that the components of the workflow can indeed fulfill their function, and the algorithm “understands” how to use those parameters to provide alternatives of viable outcomes.

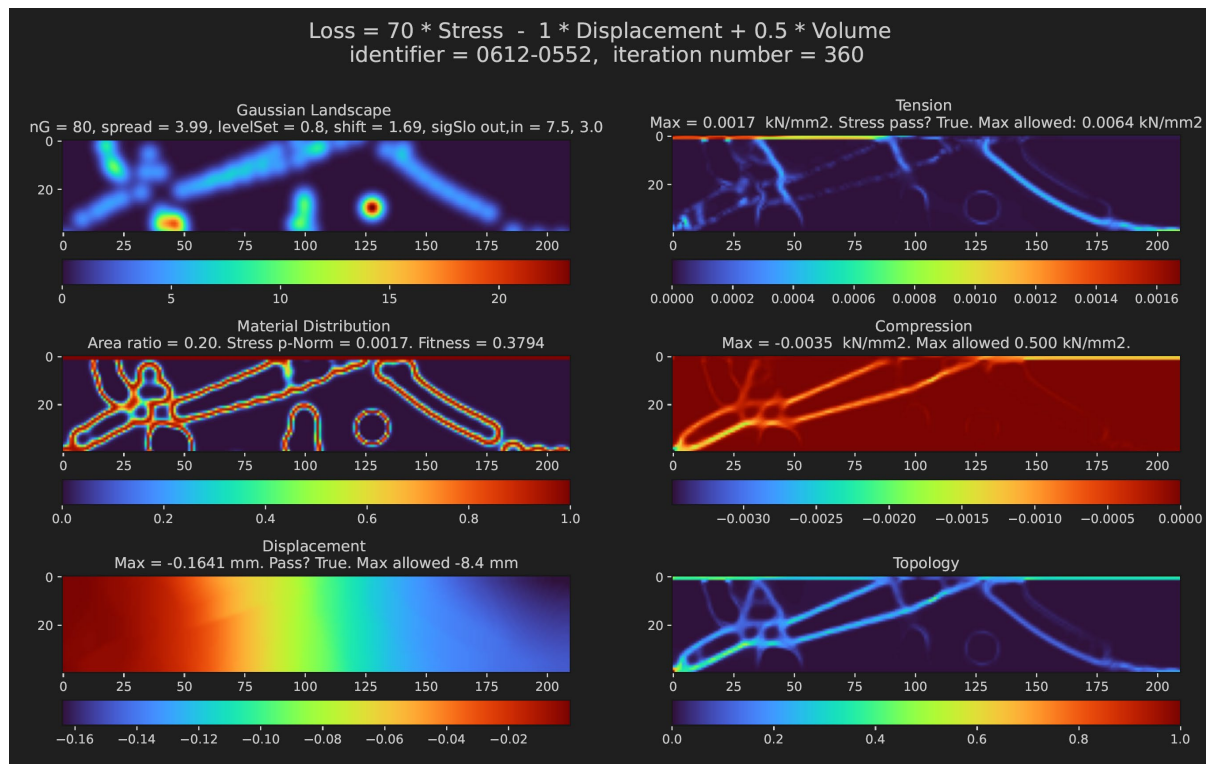


Figure 12.8: Visualization of two problems and two successes. P1) Hanging material: the circle and block disconnected from the rest of the structure on the second graph of the left column. P2) Unused paths: extra circular path centred around point (45, 35) on the bottom left of the material distribution, of which only one small part is used. S1) Using the Gaussians to draw the structure: relationship between Gaussian landscape and material distribution/stress fields. S2) Intermediate densities around the topology boundary allow the creation of smooth stress paths.

This first positive point confirmed the expectation of the behaviour of the algorithm. The second positive point, however, was completely unexpected. The intermediate densities around the boundary of the topology, which are usually seen as a problem/undesired, actually allowed the formation of very smooth stress paths, since the stress could use those intermediate densities to create “shortcuts” between the more obvious material areas. It would not have been able to do that if a discrete material distribution had been used.

The use of the intermediate density, however, also allowed the formation of thicker elements than expected, and led to the soft constraint on annealing becoming even softer.

The result of the objective function sometimes was NaN, which broke the optimization process. It was not clear what the cause was. Sometimes it seemed that increasing the sigmoid slope too much would cause a NaN (which could be explained by the derivative approaching infinity), or a learning rate that was too high, or the attempt to model a hinged support (which could be explained by the fact that in that the whole model is then hanging on through areas of extremely low density, and thus huge displacements and stresses are found) onto just two nodes on the corners. On most runs, the optimization did not return a NaN as long as a fixed support was used, and the sigmoid slopes were kept low.

To close this section, the focus point of this project was the material distribution definition. The main achievement was to create a new, manufacturing inspired, material distribution definition, through the double slicing of the Gaussian Field, and the adapted sigmoid, and implementing it using pytorch, which allowed for the automatic differentiation and the use of different optimizers.

12.4 Wider application possibilities

Zooming out and considering a wider application of this workflow begs the questions: How is this algorithm customized to cast glass? Could it be applied in other materials? What would change and why?

The material cast glass is embedded in the workflow in three main ways: 1) through the Drucker-Prager criteria, which takes its peculiar, brittle properties into account, 2) through the use of Gaussian peaks with *the same spread and amplitude* to define the material distribution with even cross-sections to decrease the chances of residual stress creation during annealing, and 3) the use of Gaussian peaks as basis functions to encourage rounded transitions in an attempt to reduce peak stresses in re-entrant corners.

The first, the stress criteria, could be tweaked to represent non-brittle materials by applying the Von Mises criteria instead. The second, however, has much more interesting possibilities.

The landscape definition has kept the spread and amplitude of the Gaussian peaks the same for every peak, so that, when sliced, they would form a topology with even thickness. This was due to the annealing constraint of glass, which needs to cool evenly. Other (brittle) materials do not have that same requirement. And thus the spreads and amplitudes can be freed, and optimized individually. That is to say, each Gaussian peak can have its own amplitude and spread, independently of each other. This will cause a larger variety of thicknesses throughout the topology.

The following figures show some examples of such optimizations. They were all performed using the Adam optimizer with the same objective function as the optimizations presented in the Results section.

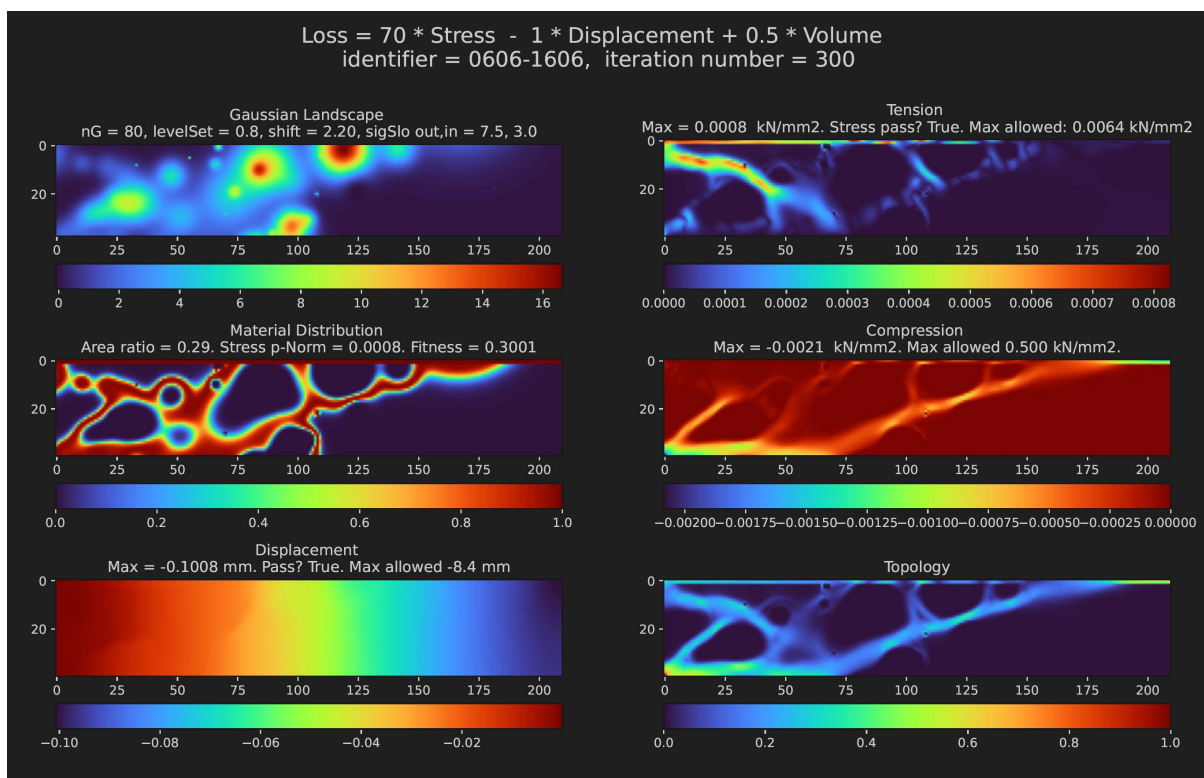


Figure 12.9: Example of optimizing spreads and amplitudes individually. It has 320 optimizable parameters. Four for each of the 80 Gaussian peaks: x-coordinate, y-coordinate, spread, and amplitude. Random seed = 04

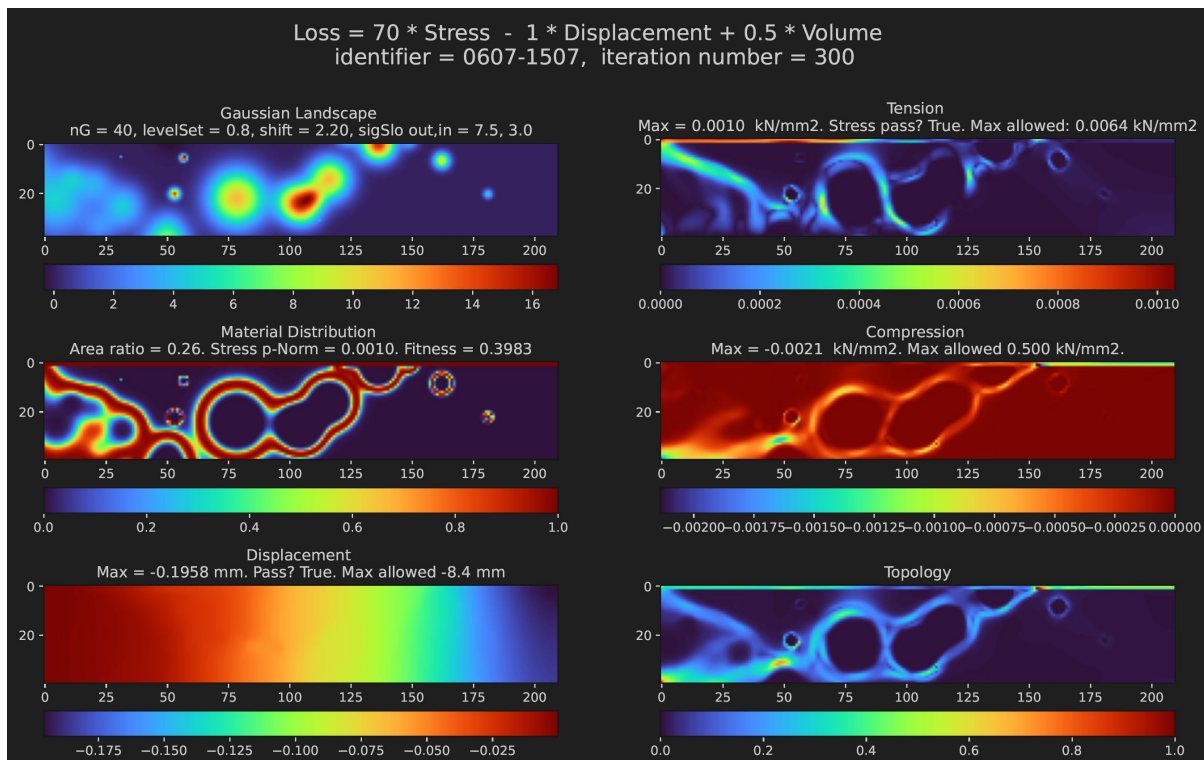


Figure 12.10: Example of optimizing spreads and amplitudes individually while keeping the number of design parameters the same (as the other optimization runs performed) by halving the number of gaussian peaks. So now there are 160 optimizable parameters. Four for each of the 40 Gaussian peaks. Random seed = 03

There is some floating material, and the boundary between material and no-material seems more blurred (wider spreads cause a larger intermediate material belt around the boundary).

The optimization on Figure 12.9 performs better than the one on Figure 12.10 in terms of a lower stress. While the later performs better than the former in terms of area. On both, there is a wide range of cross-sectional thicknesses in the design.

On Figure 12.11 the creation of intermediate density is present around point (35,22) through the inclusion of a peak with relatively low amplitude, which just barely touched the lower-bound of the sigmoid slicing function. It indicates that perhaps a intermediate density punishment might be needed if the amplitudes are to be optimized as well. On the other hand, there might be enough results that do not show that problem (as the ones on Figures 12.09 and 12.10) so that such a change is not necessary.

Even with this amplification of the freedom of design, the topologies are still based on a rounded shape. Another possible adaptation could be the use of another basis function (not a Gaussian), which would provide a different building block for the topology. There could be as many variations of this as there are 3-D functions in the world, each of them would be more (or less) suited for the particular material and manufacturing method.

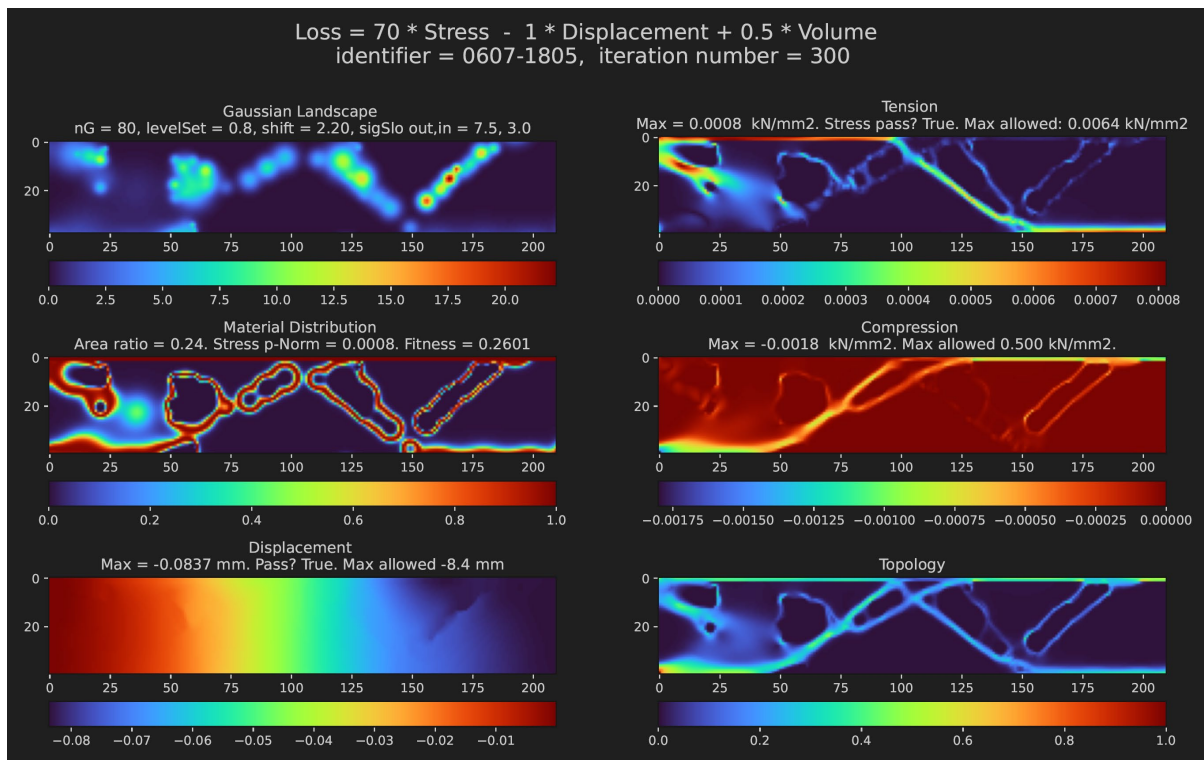


Figure 12.11: Here the same random seed is used as in the example above, but with 80 gaussian peaks again (and thus 320 optimizable parameters). Random seed = 03

12.5 Further research

The overall workflow successfully outputted a stress-based optimized topology. However, the algorithm also behaved in unforeseen ways. More structured tests to really see the effect of changing parameters, mesh density, loading and support conditions would be useful. And so would benchmark optimizations.

There are (overlapping) areas of future research: computational performance (Sections 12.5.1-3), parameters of the material distribution (12.5.4-8), and overarching topics (Sections 12.5.9-12)

12.5.1 Optimizer

The field of optimizers is vast. There are so many alternatives and intricacies. More investigation into other optimizers could be very useful to get better, faster results.

12.5.2 Learning rate

The learning rate during optimization could also use more experimentation. Additionally, one could explore a different learning rate for each type of parameter. That is, one learning rate for the (x, y) coordinates of the gaussian peaks, but a different one for the spread and perhaps yet another one for the shift. This goes hand-in-hand with the optimizer experimentation.

12.5.3 Parallel processing

Another area of improvement is to implement parallel processing. The optimizations presented here were all run on the CPU, but should be faster if run on the GPU, using CUDA. In this project, the images of the optimization process were generated during the optimization. To use matplotlib, the data needed to be on the CPU, so even though CUDA was set up, it ended up not being used, to avoid the movement of data back and forth from the CPU to the GPU. A different system, where the visualizations are not created during the optimization itself, could improve the computational time and allow for a smooth transition to using CUDA.

Next, some comments will be made on the parameters used for the material distribution.

12.5.4 Hard Manufacturing criteria

If a more specific material thickness is desired, a hard enforcement of the manufacturing criteria could be developed in one of the following ways:

- 1) by applying a filter constraint on the material distribution as was done in Koniari (2022) and Schoenmaker (2023)
- 2) by posing a constraint on the shift parameter given the spread,
- 3) by automatically calculating the shift parameter given a desired material thickness and the spread,
- 4) by removing the shift and spread from the optimizable parameters list and making them given parameters instead, which already reflect a certain desired thickness and accuracy (i.e. smallest building block size).

12.5.5 Initialization of shift and spread parameters

The starting values of the optimizable parameters (x, y) coordinates, shift, and spread, which can have a large influence on the final output. Only the coordinates were really varied in this thesis, by using Gaussian random fields as a starting point, but the shift and spread were initialized with around the same values every time. The shift and spread are hypothesized to have the highest influence on the *thickness* of the material. Thus experimenting with other sets of initial values for these two parameters might yield topologies that have either more slender or more chunky members. This is especially relevant in the cast-glass context, where annealing time is a crucial constraint.

12.5.6 Number of basis functions

Another area of improvement could be in experimenting with and fine tuning the choice of the number of Gaussian peaks used, which directly influences the amount of material available for the optimizer to allocate, as well as being the largest contributor to the number of design variables. This affects the level of design freedom and the computational power needed to solve the optimization problem.

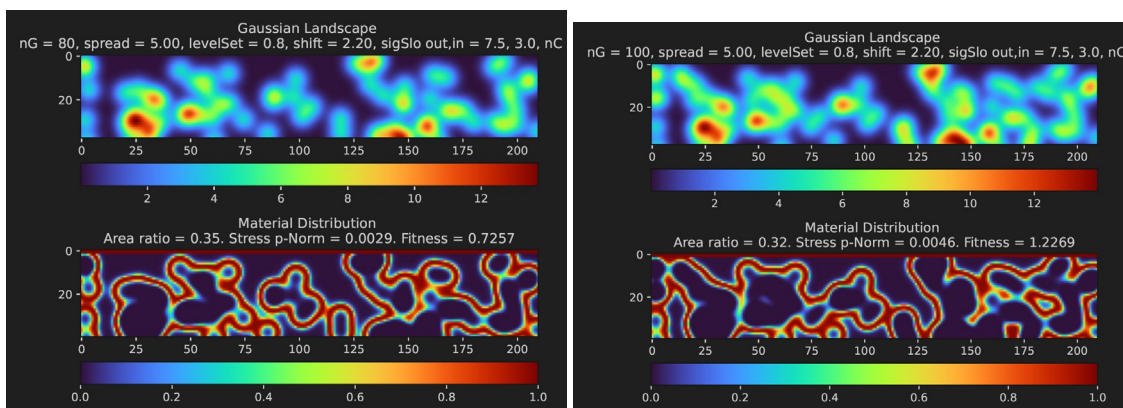


Figure 12.12: Left: Starting point of optimization 0606-0901, with 80 gaussian peaks, the same amount as all optimizations considered in the Results section. Right: Starting point of optimization 0606-2049, with the same inputs as identifier 0606-0901, but with 100 gaussian peaks instead of 80 (the coordinates of the first 80 are the same, the extra 20 are defined by 40 new random numbers).

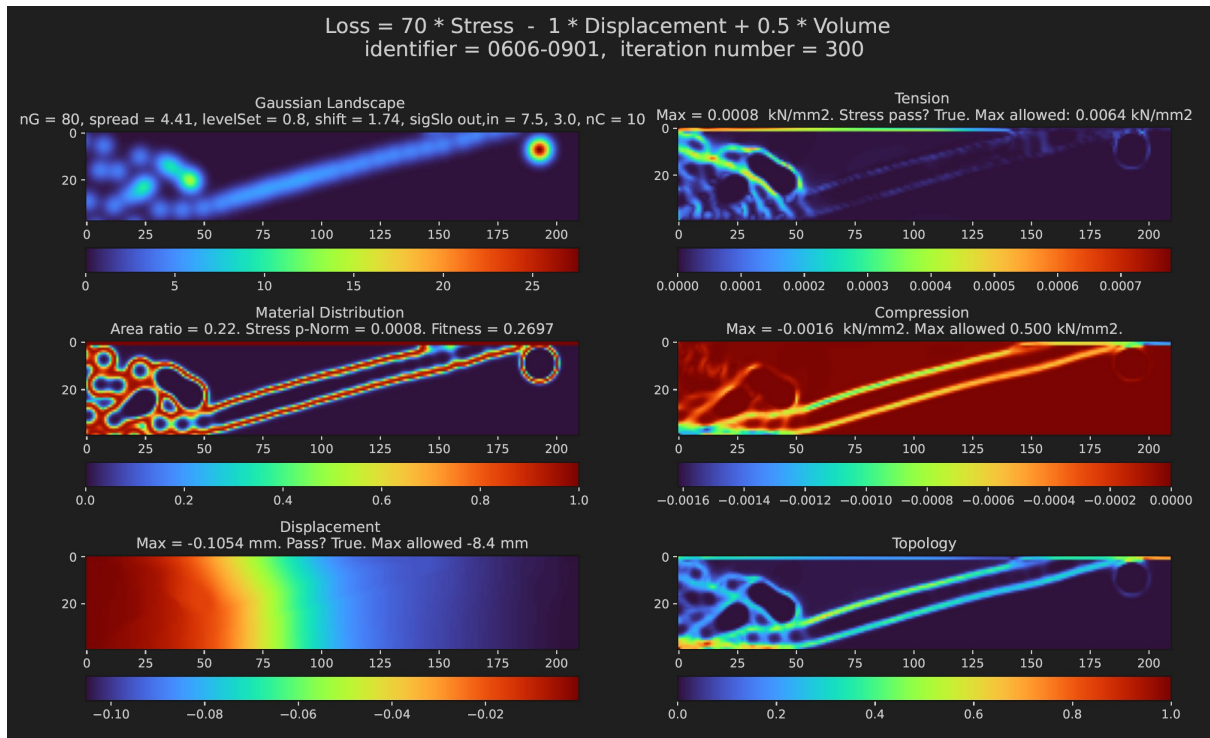


Figure 12.13: End point of optimization 0606-0901 (80 gaussian peaks).

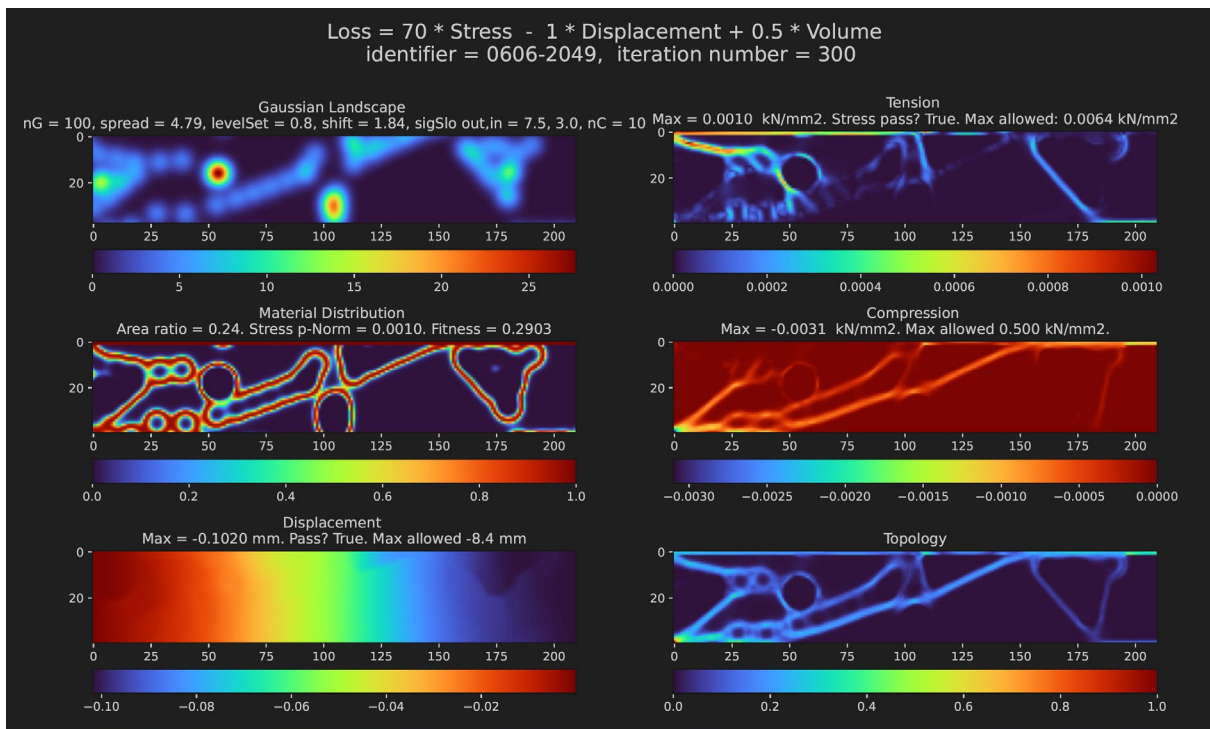


Figure 12.14: End results of optimization 0606-2049. Same inputs as identifier 0606-0901, but with 100 gaussian peaks instead of 80 (the coordinates of the first 80 are the same, the extra 20 are defined by 40 new random numbers)

Interestingly, the one with less Gaussian Peaks, on **Figure 12.13**, achieved a better overall performance (i.e. lower objective total) compared to the one with more peaks, on **Figure 12.14**. It performed better both in terms of the lowest tensile stress as well as the lowest area. It indicates that more design parameters (and thus higher computational burden) does not necessarily mean better results.

12.5.7 Sigmoid slopes

The slopes of the sigmoid curves were a particularly tricky part of the project. The slopes should be high enough that they do create a boundary between material and no-material within a couple of finite elements, but not so steep that they cause numerical instabilities.

Of additional importance regarding the sigmoid slopes is the fact that the boundary gets defined both on the outer (the valleys of the landscape) as well as inner (the peaks of the landscape) sections of the Gaussian landscape. The Gaussian Landscape itself has different slopes at those two positions. The slope close to the valley is lower (ascend is slower), while the slope close to the peak is higher (ascent is steeper). This means that on the outer boundary it takes longer (physically) for the Gaussian peak to pass the sigmoid transition area, creating a higher pseudo density in the material.

In this thesis, some initial experimentation was done, but this area could use improvement, and perhaps a more strict mathematical definition of the ideal values. With the slopes used in the project, the pseudo-densities on the outside of the slopes were still quite “high”. This allowed the algorithm to make use of this stiffness and sometimes material remained “hanging on” this “empty” area. A lower density would have probably caused a larger displacement and a removal of the material through the optimization given the presence of the displacement term in the objective function.

Another possible contributor to the hanging material phenomenon is that these material islands were formed by the combination of many concentric Gaussian peaks, so perhaps the cost of the displacement was lower than the cost of the movement of a single Gaussian peak away from the shared centroid, since this move would increase the area ratio. So an exploration of the weights of the objective function could also be helpful here.

One could also consider to optimize the slopes themselves. However, it is not as simple as just adding them as extra optimizable parameters within the existing workflow. **Figure 12.15** shows the result of doing so. The algorithm then behaves similarly as SIM without P. That is, it creates intermediate density as a way to gain stiffness at a low material cost.

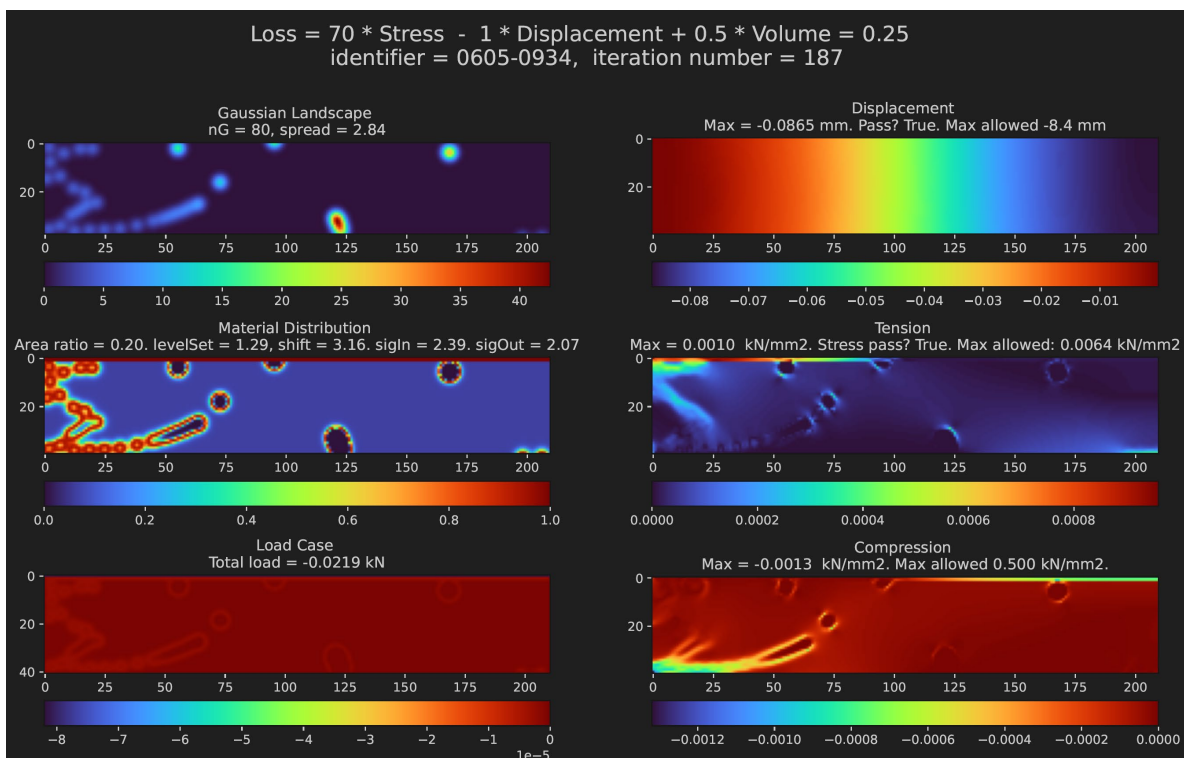


Figure 12.15: Sigmoid slope optimization.

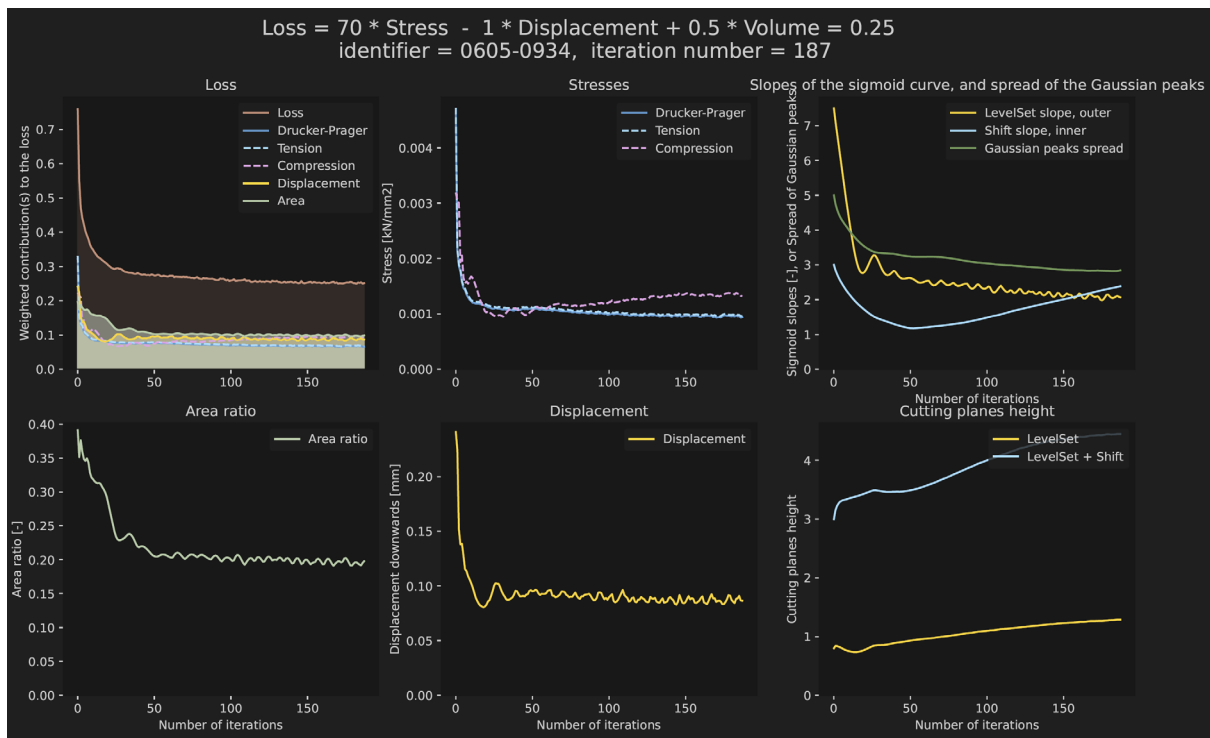


Figure 12.16: Loss evolution to achieve the optimization results presented in Figure 12.15

In **Figure 12.16**, on the upper right graph, the outer slope (in yellow) decreases through the optimization. That is what creates the intermediate density. On **Figure 12.15** On the second graph of the left column, Material Distribution, we see the relatively lighter purple background, which represents an intermediate density of around 10-15% material. It is noticeably lighter (i.e. higher density) than the backgrounds of the Material Distributions of the results where the slopes are not optimized, such as in **Figure 12.14**.

The sigmoid slope has a single purpose: to create a differentiable, but as sharp as possible, boundary between material and no-material. Thus it might not be beneficial to optimize it concurrently with the other parameters. But rather to either pre-define it with a static value (as was done in the work presented here), or make it a function of the other parameters which ensures that the “empty” areas of the design remain as empty as possible (i.e. tending towards zero). A minimum density is anyways added during the FEA to avoid singularity in the calculation.

12.5.8 Amplitude

The amplitude of each gaussian peak could also be experimented with. This was kept constant in this project in the hopes of softly enforcing a more even cross section through the design domain, but could potentially be experimented with. Even in this project, there were varying heights of the landscape anyway, given the addition of peaks placed close together. On the other hand, experimenting with this particular parameter might not make as large of a difference in comparison with some of the other, more impactful contributors to the topology and algorithmic performance (such as the spread and sigmoid curve slopes for the former, and optimizer and learning rates for the later).

12.5.9 Structured investigation

With respect to all of the chosen parameters above, more structured optimization runs should have been performed. It would also have been useful to apply the workflow to some well-known problems such as the shoe (L-shaped beam with a point load), as many of the research papers presented in the introduction did. Most of the optimization process was done on intuition, and each optimization run was set up based on the results of its precedent, but a broader overview could be made to decide what parameters to use

for the optimization and perhaps find a better topology. This was in part due to lack of time, and in part due to the need of gaining experience with the optimization algorithm in order to get a feel for what parameters are suitable. This applies to parameters pertaining to the optimization process itself (such as the learning rate, number of iterations and weights of the objective function), as well as those that define the material distribution (such as the sigmoid curve slopes, amplitude, spread, shift, and level set). Not all of them were tested, some were just used because it was working with the set value.

12.5.10 Further automation of post-processing

Post-processing could be further automated by:

- 1) Removing disconnected material using the displacement field.
- 2) Writing an algorithm that would pick the appropriate percentage threshold to ensure that the remaining, discrete material distribution is within the stress constraints after being subjected to the load. It would be an optimization whose objective is to minimize the mass given the stress constraint. The only design variable would be the percentage threshold, to be applied to the superimposed stress field that resulted from the structural optimization run. The removal of material islands through the displacement could be integrated in this workflow.
- 3) Use another program to find smooth lines around the density field.
- 4) For the results obtained with Adam, use the Material Distribution instead of the superimposed stress fields. Since this optimizer yielded results that were much more clear-cut than SGD. Those boundaries are infinitely accurate, since they are a direct calculation from the Gaussian Landscape. However, this would come at a cost of stress performance as seen in Chapter 10.1.

12.5.11 Expansion to the third dimension

The current workflow could also be expanded to the third dimension. This would require the LSF to have one extra dimension as well, and thus be defined in 4 dimensions. It would still be possible, and still have less parameters than the comparable SIMP definition. It would be fun!

A more substantial barrier to the expansion to the third dimension would be calculation of the stresses in 3-D finite elements, since adding a third dimension increases the number of degrees of freedom per finite element from 8 to 24 (assuming the use of cubic elements with 8 nodes).

12.5.12 Structural problem definition

More broadly, one could also consider the pertinence of the definition of the structural problem. Is an optimized topology generated through the application of a distributed load actually able to avoid peak stresses created during a less idealised loading situation (e.g. localized loads) in real life? Not all possible loading scenarios can be avoided, but given the safety factors and the fact that the stress limit is not even close to being reached for the optimized topology with the distributed load case, it is reasonable to assume that most realistic point load applications would still be well supported by this structure. The two top, float glass, laminated plates are also included to help distribute those point loads to the cast glass structure.

Additionally, this is a static optimization. However, real life is dynamic. The structure would deform (even if ever-so-slightly due to its high stiffness) after the application of the load. Does the new, deformed configuration behave and perform the same as the initial one?

Lastly, buckling has not been considered here. It could be a problem since it is the failure through tension of slender members loaded in compression. The algorithm tends to create members in compression (due to the material properties and Drucker-Prager criteria) and slenderness (due to the minimization of volume present in the objective function). It also tends to create members that are not completely straight, due to the use of Gaussian basis functions.

13 Bibliography

- Amstutz, S., & Andrä, H. (2006). A new algorithm for topology optimization using a level-set method. *Journal of Computational Physics*, 216(2), 573–588. <https://doi.org/10.1016/j.jcp.2005.12.015>
- Andreassen, E., Clausen, A., Schevenels, M., Lazarov, B. S., & Sigmund, O. (2011). Efficient topology optimization in MATLAB using 88 lines of code. *Structural and Multidisciplinary Optimization*, 43(1), 1–16. <https://doi.org/10.1007/s00158-010-0594-7>
- Bendsøe, M. P., & Sigmund, O. (2004). *Topology Optimization. Theory, Methods, and Applications*. Springer Berlin.
- Bristogianni, T., Oikonomopoulou, F., Yu, R., Veer, F. A., & Nijse, R. (2020). Investigating the flexural strength of recycled cast glass. *Glass Structures and Engineering*, 5(3), 445–487. <https://doi.org/10.1007/s40940-020-00138-2>
- Bruggi, M., & Duysinx, P. (2012a). Topology optimization for minimum weight with compliance and stress constraints. *Structural and Multidisciplinary Optimization*, 46(3), 369–384. <https://doi.org/10.1007/s00158-012-0759-7>
- Bruggi, M., & Duysinx, P. (2012b). Topology optimization for minimum weight with compliance and stress constraints. *Structural and Multidisciplinary Optimization*, 46(3), 369–384. <https://doi.org/10.1007/s00158-012-0759-7>
- Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., & Lerchner, A. (2018). *Understanding disentangling in β -VAE*. <http://arxiv.org/abs/1804.03599>
- Carstensen, J. V., Kim-Tackowiak, H., & Liang, M. Y. (2023). Improving the manufacturability of highly materially restricted topology-optimized designs with Mixed Integer Linear Programming. *Engineering Structures*, 284. <https://doi.org/10.1016/j.engstruct.2023.115955>
- Chandrupatla, T. R. (2002). *Introduction to Finite Elements in Engineering*.
- Chen, R. T. Q., Li, X., Grosse, R., & Duvenaud, D. (2018). *Isolating Sources of Disentanglement in VAEs*.
- Damen, W. (2019). *Topologically Optimised Cast Glass Grid Shell Nodes Exploring Topology Optimisation as a design tool for Structural Cast Glass elements with reduced annealing time*.
- Drucker, D. C., & Prager, W. (1952). Soil mechanics and plastic analysis or limit design. *Quart Appl Math*, 10, 157–165.
- Duysinx, P., & Bendsøe, M. P. (1998). Topology optimization of continuum structures with local stress constraints. *International Journal for Numerical Methods in Engineering*, 43(8), 1453–1478. [https://doi.org/10.1002/\(SICI\)1097-0207\(19981230\)43:8<1453::AID-NME480>3.0.CO;2-2](https://doi.org/10.1002/(SICI)1097-0207(19981230)43:8<1453::AID-NME480>3.0.CO;2-2)
- Duysinx, P., & Sigmund, O. (1998). New developments in handling stress constraints in optimal material distribution. *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1501–1509. <https://doi.org/10.2514/6.1998-4906>
- Grover, A. (2023). *Deep Generative Models*. <https://Deepgenerativemodels.Github.io/Notes/Introduction/>.
- Guest, J. K., Prévost, J. H., & Belytschko, T. (2004). Achieving minimum length scale in topology optimization using nodal design variables and projection functions. *International Journal for Numerical Methods in Engineering*, 61(2), 238–254. <https://doi.org/10.1002/nme.1064>
- Guo, X., Zhang, W. S., Wang, M. Y., & Wei, P. (2011). Stress-related topology optimization via level set approach. *Computer Methods in Applied Mechanics and Engineering*, 200(47–48), 3439–3452. <https://doi.org/10.1016/j.cma.2011.08.016>

- Harish, B., Eswara Sai Kumar, K., & Srinivasan, B. (2020). Topology optimization using convolutional neural network. *Lecture Notes in Mechanical Engineering*, 301–307. https://doi.org/10.1007/978-981-15-5432-2_26
- Holmberg, E., Torstenfelt, B., & Klarbring, A. (2013). Stress constrained topology optimization. *Structural and Multidisciplinary Optimization*, 48(1), 33–47. <https://doi.org/10.1007/s00158-012-0880-7>
- Hoyer, S., Sohl-Dickstein, J., & Greydanus, S. (2019). *Neural reparameterization improves structural optimization*. <http://arxiv.org/abs/1909.04240>
- Hubert, M. (2015). *IMI-NFG Course on Processing in Glass Lecture 1: Commercial glass compositions, properties and technical considerations-Raw materials*. <http://www.metmuseum.org>
- Jeong, H., Batuwatta-Gamage, C., Bai, J., Xie, Y. M., Rathnayaka, C., Zhou, Y., & Gu, Y. T. (2023). A complete Physics-Informed Neural Network-based framework for structural topology optimization. *Computer Methods in Applied Mechanics and Engineering*, 417. <https://doi.org/10.1016/j.cma.2023.116401>
- Jewett, J., Koniari, A. M., Andriotis, C., Oikonomopoulou, F., Bristogianni, T., & Carstensen, J. V. (2024). More with Less: Topology Optimization Strategies for Structural Glass Design. *Challenging Glass Conference Proceedings*, 9.
- Kingma, D. P., & Ba, J. (2014). *Adam: A Method for Stochastic Optimization*. <http://arxiv.org/abs/1412.6980>
- Kiyono, C. Y., Vatanabe, S. L., Silva, E. C. N., & Reddy, J. N. (2016). A new multi-p-norm formulation approach for stress-based topology optimization design. *Composite Structures*, 156, 10–19. <https://doi.org/10.1016/j.compstruct.2016.05.058>
- Koniari, A. M. (2022). *Development of a Topology Optimization Algorithm for a Mass-Optimized Cast Glass Component*.
- Koniari, A. M., Andriotis, C., & Oikonomopoulou, F. (2023). Minimum mass cast glass structures under performance and manufacturability constraints. *International Conference on Computer-Aided Architectural Design Futures*, 437–451.
- Kumar, P. (2017). *Synthesis of Large Deformable Contact-Aided Compliant Mechanisms using Hexagonal cells and Negative Circular Masks*.
- Labuz, J. F., Zeng, F., Makhnenko, R., & Li, Y. (2018). Brittle failure of rock: A review and general linear criterion. In *Journal of Structural Geology* (Vol. 112, pp. 7–28). Elsevier Ltd. <https://doi.org/10.1016/j.jsg.2018.04.007>
- Langelaar, M. (2016). Topology optimization of 3D self-supporting structures for additive manufacturing. *Additive Manufacturing*, 12, 60–70. <https://doi.org/10.1016/j.addma.2016.06.010>
- Le, C., Norato, J., Bruns, T., Ha, C., & Tortorelli, D. (2010). Stress-based topology optimization for continua. *Structural and Multidisciplinary Optimization*, 41(4), 605–620. <https://doi.org/10.1007/s00158-009-0440-y>
- Lew, A. J., & Buehler, M. J. (2021). Encoding and exploring latent design space of optimal material structures via a VAE-LSTM model. *Forces in Mechanics*, 5. <https://doi.org/10.1016/j.finmec.2021.100054>
- Loeber, P. (2021, February 21). *Deep Learning with Pytorch - Full Course*.
- Nie, Z., Lin, T., Jiang, H., & Kara, L. B. (2020). *TopologyGAN: Topology Optimization Using Generative Adversarial Networks Based on Physical Fields Over the Initial Domain*. <http://arxiv.org/abs/2003.04685>
- Oikonomopoulou, F. (2019). *Unveiling the third dimension of glass Solid cast glass components and assemblies for structural applications*.

- Oikonomopoulou, F., Ioannidis, M., Koniari, A. M., & Bristogianni, T. (2023). *Fabrication methods for topology-optimized massive glass structures*.
- Oikonomopoulou, F., Koniari, A. M., Damen, W., Koopman, D., Stefanaki, I., & Bristogianni, T. (2022). Topologically optimized structural glass megaliths: potential, challenges and guidelines for stretching the mass limits of structural cast glass. *8th Eighth International Conference on Structural Engineering, Mechanics and Computation*, 437–450.
- Osher, S., & Fedkiw, R. P. (2001). Level Set Methods: An Overview and Some Recent Results. *Journal of Computational Physics*, 169(2), 463–502. <https://doi.org/10.1006/jcph.2000.6636>
- Osher, S., & Sethian, J. A. (1988). Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations. In *JOURNAL OF COMPUTATIONAL PHYSICS* (Vol. 79).
- Picelli, R., Townsend, S., Brampton, C., Norato, J., & Kim, H. A. (2018). Stress-based shape and topology optimization with the level set method. *Computer Methods in Applied Mechanics and Engineering*, 329, 1–23. <https://doi.org/10.1016/j.cma.2017.09.001>
- Schoenmaker, E. (2023). *Adding a New Dimension to Glass Giants Development of a Three-Dimensional Topology Optimization Algorithm for Mass-Optimized Cast Glass Components*.
- Sigmund, O. (2001). A 99 line topology optimization code written in matlab. *Structural and Multidisciplinary Optimization*, 21(2), 120–127. <https://doi.org/10.1007/s001580050176>
- Soydaner, D. (2020). A Comparison of Optimization Algorithms for Deep Learning. *International Journal of Pattern Recognition and Artificial Intelligence*.
- Steift, P. S. (1983). Ductile versus brittle behaviour of amorphous metals. In *I. Mech. Phys. Solids* (Vol. 31, Issue 5).
- The Efficient Engineer. (2021, June 2). *Understanding Failure Theories*. <https://www.youtube.com/watch?v=xkbQnBAOFeg&t=1s>.
- Ulyanov, D., Vedaldi, A., & Lempitsky, V. (2018). *Deep Image Prior*. <http://www.cs.tut.fi/>
- Van Dijk, N. P., Maute, K., Langelaar, M., & Van Keulen, F. (2013). Level-set methods for structural topology optimization: A review. In *Structural and Multidisciplinary Optimization* (Vol. 48, Issue 3, pp. 437–472). Springer Verlag. <https://doi.org/10.1007/s00158-013-0912-y>
- Wang, M. Y., Wang, X., & Guo, D. (n.d.). *A level set method for structural topology optimization*. www.elsevier.com/locate/cma
- Wei, P., Li, Z., Li, X., & Wang, M. Y. (2018). An 88-line MATLAB code for the parameterized level set method based topology optimization using radial basis functions. *Structural and Multidisciplinary Optimization*, 58(2), 831–849. <https://doi.org/10.1007/s00158-018-1904-8>
- White, D. A., Arrighi, W. J., Kudo, J., & Watts, S. E. (2019). Multiscale topology optimization using neural network surrogate models. *Computer Methods in Applied Mechanics and Engineering*, 346, 1118–1135. <https://doi.org/10.1016/j.cma.2018.09.007>
- Xia, L., Zhang, L., Xia, Q., & Shi, T. (2018). Stress-based topology optimization using bi-directional evolutionary structural optimization method. *Computer Methods in Applied Mechanics and Engineering*, 333, 356–370. <https://doi.org/10.1016/j.cma.2018.01.035>
- Yulin, M., & Xiaoming, W. (2004). A level set method for structural topology optimization and its applications. *Advances in Engineering Software*, 35(7), 415–441. <https://doi.org/10.1016/j.advengsoft.2004.06.004>
- Zhang, S., Gain, A. L., & Norato, J. A. (2017). Stress-based topology optimization with discrete geometric components. *Computer Methods in Applied Mechanics and Engineering*, 325, 1–21. <https://doi.org/10.1016/j.cma.2017.06.025>

Zhang, Y., Peng, B., Zhou, X., Xiang, C., & Wang, D. (2019). *A deep Convolutional Neural Network for topology optimization with strong generalization ability.*

14 Appendix

14.1 Full domain MATLAB code validation

The validation uses the Case Study set up from Koniari (2022) and Schoenmaker (2023). Structural dimensions are a beam length of 2100 mm and height of 300 mm. The supports are: a fixed support on one end and displacement in the x-direction support (i.e. mirroring) on the other end. The loads are: a self-weight load with a factor of 1.2 plus a 5.4 kN/mm² distributed load with a factor of 1.5. The material properties are for borosilicate glass as defined in the Glass chapter.

| | Maximum deflection | Maximum principal stress | Minimum principal stress |
|-----------------------------------|--------------------|--------------------------|--------------------------|
| | mm | kN/mm ² | kN/mm ² |
| ANSYS, 3D (Schoenmaker, 2023) | -0.09436 | 0.002419 | -0.002099 |
| MATLAB, 3D (Schoenmaker, 2023) | -0.09443 | 0.001993 | -0.001976 |
| ANSYS, 2D (Koniari, 2022) | -0.09866 | 0.002179 | -0.002156 |
| MATLAB, 2D (own model) | -0.09176 | 0.001933 | -0.001923 |

Table 14.1: Results for maximum displacement and principal stresses from ANSYS and MATLAB models.

The discrepancy in principal stresses between the ANSYS and MATLAB calculations is just over 12%. The results obtained by Schoenmaker's (2023) MATLAB model had a very similar discrepancy from the ANSYS calculation.

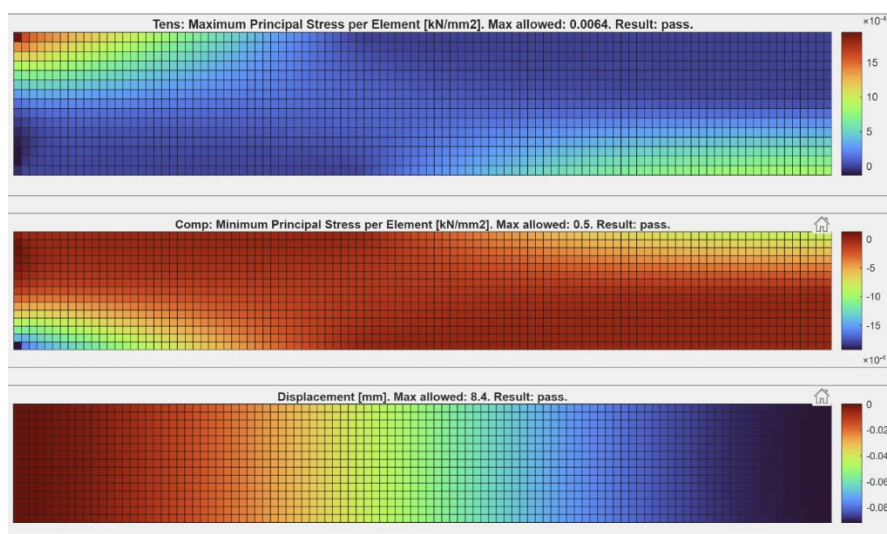


Figure 14.1: Stresses and displacement fields of the full domain. Fixed support on the left, symmetry axis on the right.

14.2 Design validation with ANSYS

The post-processed topology achieved with SGD optimizer has been validated using ANSYS. **Figure 14.2** shows the boundary conditions and loads of the simulation, and **Figure 14.3** Shows the mesh used by ANSYS. Note that the present work used a strictly square mesh with finite elements of equal dimensions, while ANSYS used a more precise mesh, adapted to the contour.

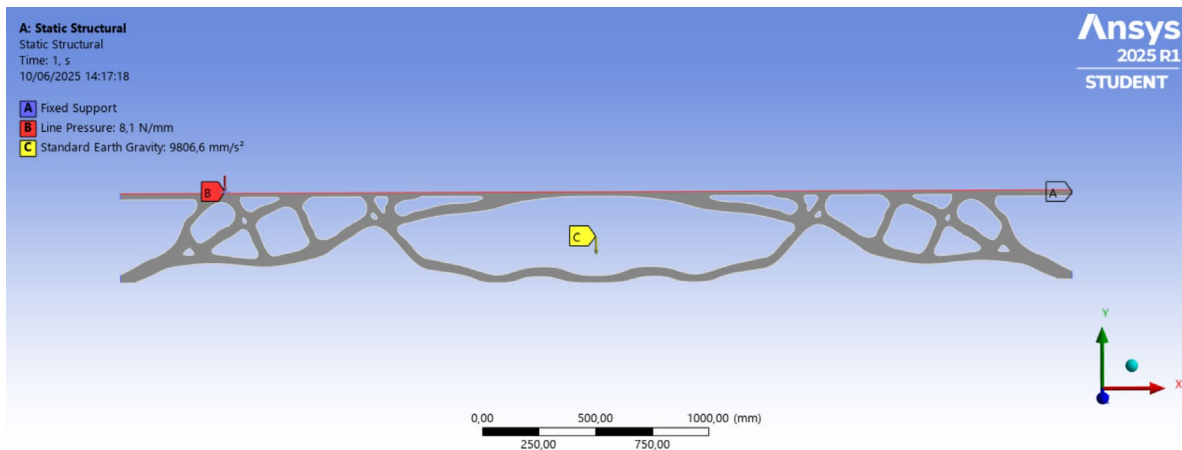


Figure 14.2: Boundary conditions and loads from ANSYS validation.

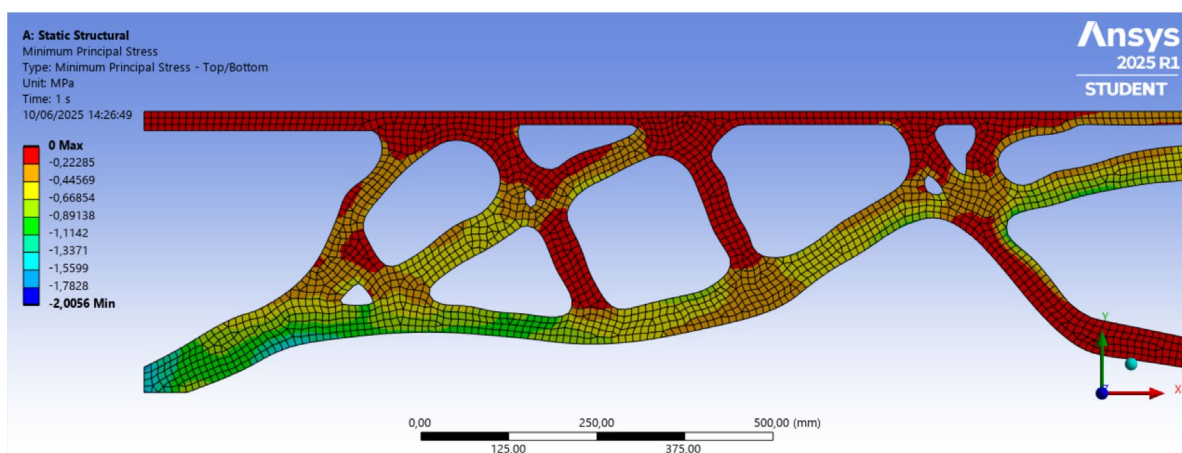


Figure 14.3: Mesh from ANSYS validation.

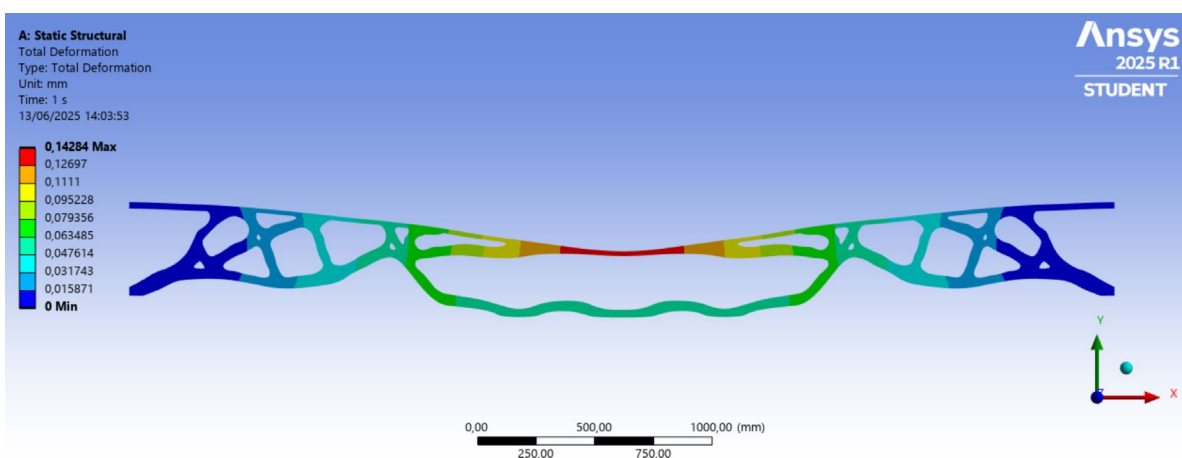


Figure 14.4: Deformation from ANSYS validation, SGD design.

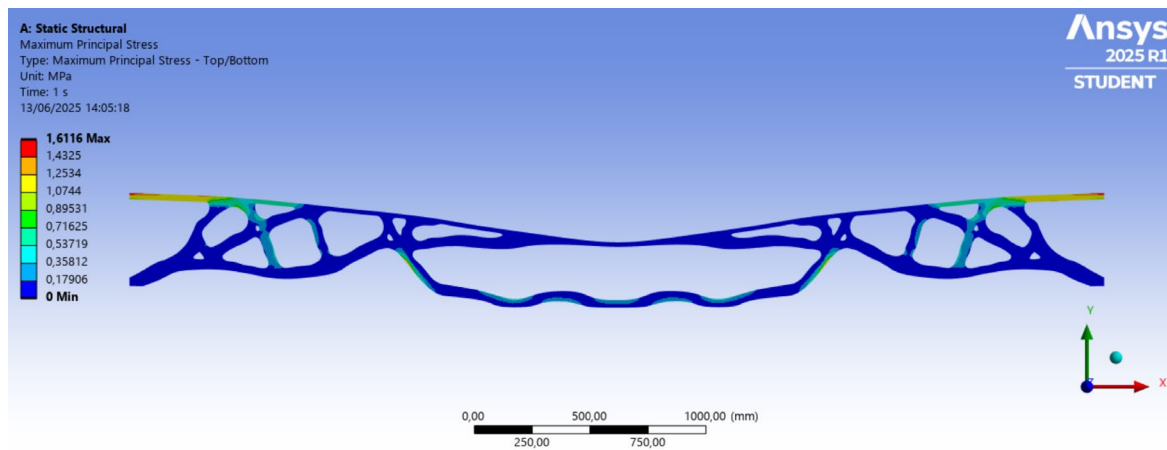


Figure 14.5: Maximum principal stresses from ANSYS validation, SGD design.

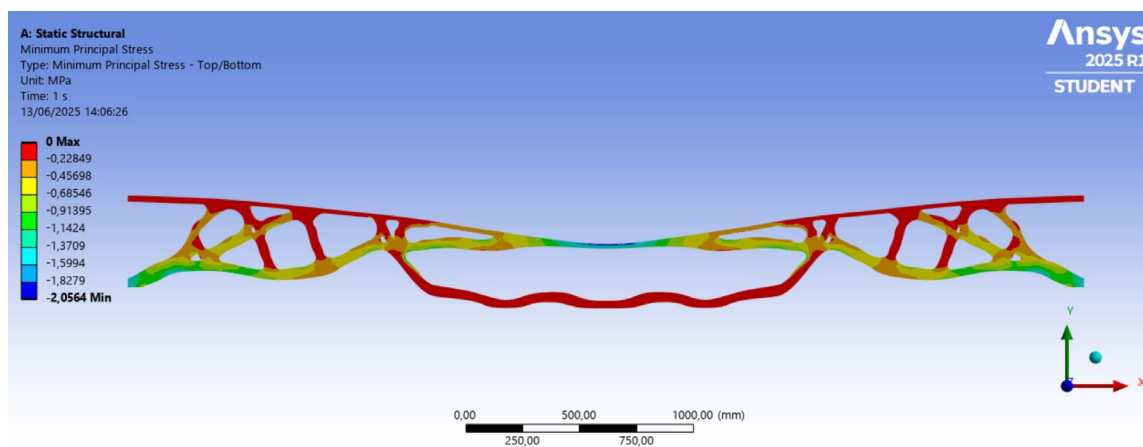


Figure 14.6: Minimum principal stresses from ANSYS validation, SGD design.

| | SGD | | Adam | |
|----------------------|--------|-------|--------|-------|
| | python | ANSYS | python | ANSYS |
| Max principal stress | 1.7 | 1.6 | 1.3 | 1.1 |
| Min principal stress | -3.9 | -2.1 | -1.9 | -1.4 |
| Max deformation | 0.26 | 0.14 | 0.08 | 0.07 |

Table 14.2: Validation of results with ANSYS.

The comparison of the results achieved through the present work and ANSYS for the SGD design show that for the maximum principal stress, the results are very comparable, with the present work giving a more conservative outcome than ANSYS for the maximum principal stress. The minimum principal stresses, however, differ significantly. The results also differ for the deformation. At least the python code was more conservative, which means that the proposed design would be safe. The different discretization of the topology can also have had an impact on the results. Since it is the tensile limit that is critical here, this is considered sufficient for now, but more investigation and validation should be performed if the project were to continue beyond this phase.

For the Adam design, the results are comparable for all measures, with python being more conservative.

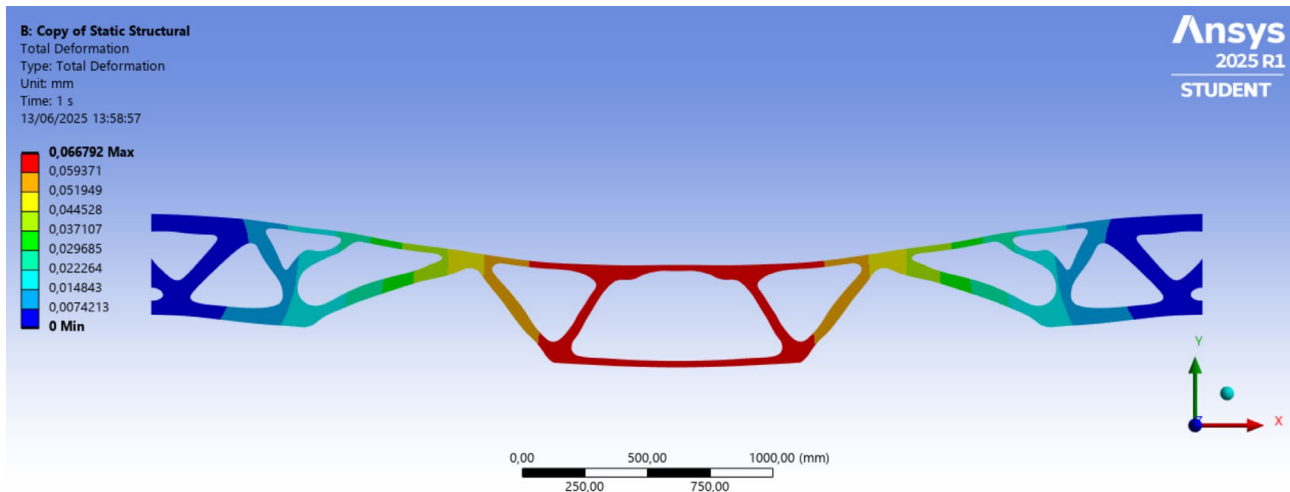


Figure 14.7: Deformation from ANSYS validation, Adam design.

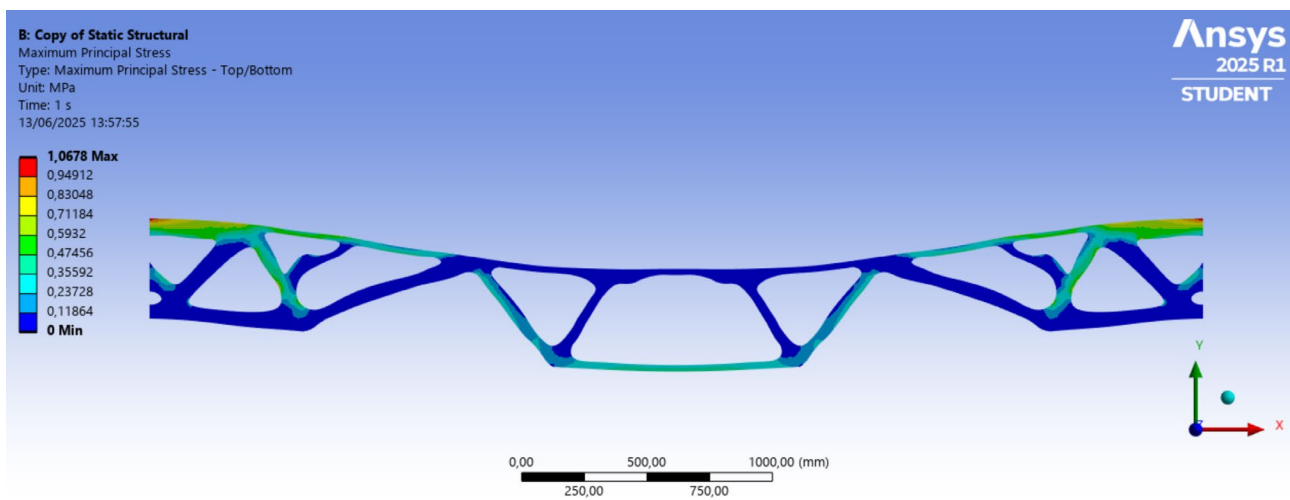


Figure 14.8: Maximum principal stresses from ANSYS validation, Adam design.

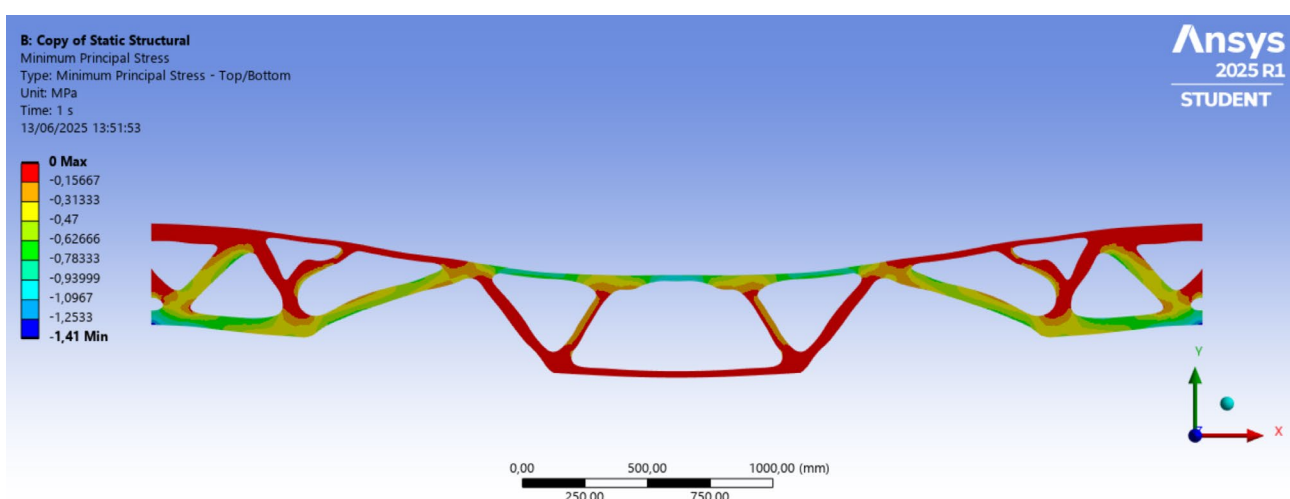


Figure 14.9: Minimum principal stresses from ANSYS validation, Adam design.

14.3 Preliminary Machine Learning Research

In recent years there has been a strong shift towards the use of artificial intelligence (AI) in many disciplines, including structural optimization. This means that the methods can be split into two groups: traditional methods which started being developed in the 1960s (including SIMP, Level-Set and (B)ESO), and machine learning methods which have been under exploration more recently. There are also some approaches which include a blend of the two (Hoyer et al., 2019).

Machine learning (ML) is a subset of AI. It refers to the capacity to generate, acquire, and process data (Grover, 2023). In this project, ML could be introduced in any of the aforementioned steps. For data generation, it can be used to create possible topologies that take into account the manufacturability of glass, to generate a stress map given a topology, or to generate a stress field with implicit topology (i.e. a FEA surrogate model) (Harish et al., 2020; Jeong et al., 2023; White et al., 2019; Y. Zhang et al., 2019).

When building a neural network, one must design (Loeber, 2021):

- 1) The model: includes input and output sizes, as well as the forward pass, which yields the model's prediction
- 2) The loss: evaluates the model's prediction
- 3) The optimizer: adapts the weights accordingly
- 4) The training loop: links them together, creating a cycle of...
 - a. Forward pass: use model to compute the prediction,
 - b. Loss: evaluate prediction,
 - c. Backward pass: compute the gradients of the loss with respect to the weights,
 - d. Update weights
- 5) Termination of the training

This is very similar to an optimization. An optimization could be seen as a two-layer neural network model: input layer and output layer, with optimizable weights in between, and the objective function of the optimization is analogous with the loss function in a ML neural network.

In traditional methodologies, the entire optimization workflow depends on being able to adjust the parameters of the topology definition to minimize the objective function. In machine learning, there are other possibilities, such as training a model to learn the features that define the objective function and then explore the trained model to find optimal results.

14.3.1 Layers

An important characteristic of neural networks is the structure of their layers. This is strongly related to the nature of the data and the task being performed. There are two major types of layer structures: nodes and convolutional layers. The learning parameters of a layer with nodes is one number per node, while for a convolutional layer they are in a matrix of defined dimensions (Burton, 2024).

Convolutional layers are particularly well suited to analysing spatially structured data, such as images. The deep image prior means that “generalization requires the structure of the network to “resonate” with the structure of the data” and “... a great deal of image statistics are captured by the structure of a convolutional image generator independent of learning.” (Ulyanov et al., 2018). Stress and displacement fields also follow a structure similar to an image, where each value (or pixel) is related to the value of its neighbours, and where information on the differences between these neighbouring values conveys something of importance about the meaning of the data. For example, an edge in a picture usually means a change of object being depicted, while an edge on a stress field could mean the boundary of the topology.

14.3.2 Deep Generative Models

Deep Generative Models in general map a dataset to a probability distribution. The enquiries that may be posed to this model are (Grover, 2023):

- 1) Density estimation: find the probability assigned by the model to a given point
- 2) Sampling: generate new data
- 3) Unsupervised representation learning: check if the model has found meaningful features

Due to the lack of a data set of already optimized glass structures, it is not possible to implement deep generative models alone to generate optimized topologies, as it has been done in literature (Nie et al., 2020; Oh et al., 2019). However, it could be possible to use them in combination with other ML algorithms or FEA to find optimized results.

Deep learning could be employed to encode information on stress and displacement fields and their implicit topologies. This way, during optimization, the optimizer has access to structural information at a much higher speed than using the Finite Element Method, as well as a differentiable path through the parameters that led to this structural field.

14.3.3 Variational Autoencoders (VAE)

Variational Autoencoders are a Machine Learning tool for dimensionality reduction and reparameterization (Lew & Buehler, 2021). By being trained to encode information into the bottleneck (a.k.a. latent space) and then decode it, they find new parameters with which to code the information, and these parameters necessarily have less dimensions than the original input. The aim of the VAE is to sample the probabilistic latent space after training, to generate previously unseen (i.e. not included in the training set) expressions of the learnt model.

Like other forms of neural networks, VAEs are an optimization problem. As such, they work towards minimizing a function. VAEs have a two-part loss function:

$$\mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z}) = -\mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{x}|\mathbf{z})] + D_{KL}(q_{\phi}(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z}))$$

Where the first term is the reconstruction loss. The expectation operator is present – in contrast with a simple Autoencoder – because a distribution is being sampled. The encoder is trained to find the values of μ and σ , the variables needed to define a Gaussian function.

The second term is the latent loss. Here the KL divergence is used to encourage the distribution to resemble a Gaussian distribution (a.k.a. normal distribution) with a mean of zero and a standard deviation of one.

These two terms compete with each other. If the first one is overly emphasized, the model will be very good at reconstructing the training data from the vectors \mathbf{z} which have been created by that same data. However, the latent space will tend to be sparser, meaning that random sampling will be unlikely to generate sensible previously unseen data (much like a normal Autoencoder). The clusters will be far apart, and the Gaussians will have little to no overlap with each other.

If the second term is overemphasized, then the representations in latent space will all be “too close” to a Gaussian with mean zero and sigma 1, and thus there will be too much overlap between all the representations. This means that the model will not be able to reconstruct the data accurately, as any sampling would lead to the same decoded data.

Thus these terms must be balanced out so that the model is able to have accurate enough reconstruction, while still also having a continuous latent space to generate new data. To balance these two factors, a

weighted loss can be used, where parameter α scales the reconstruction loss, and parameter β scales the latent loss.

An entire subset of VAEs has been created, called β VAEs, that focus on specifying the weighting factor of the latent loss to strike a good balance between regularization of the latent space and reconstruction ability (Burgess et al., 2018). Chen et al. (2018) propose a split of the latent loss KL divergence function into three separately weighted components for each of the following aspects: (i) index-code mutual information, (ii) total correlation, and (iii) dimension-wise KL divergence. This could give more control to the model designer with regards to the specific latent space properties that are being encouraged by the loss function during training.

To further break down the loss function, one must understand the prior and posterior probabilities, as used in Bayesian statistics. The posterior is $p_\theta(z|x)$, the probability density function of a vector z being generated given a data input x . It is intractable, and is thus approximated by $q_\phi(z|x)$, this is called variational inference.

After the theoretical workarounds and approximations, the loss function can be re-written as:

$$\mathcal{L} = (x - y)^2 - \frac{1}{2}[\log \sigma^2 - \mu^2 - \sigma^2 + 1]$$

Where the reconstruction loss has become the squared difference between the input to the encoder (x) and the output from the decoder (y). In practice, many models only train the mean, μ , and let σ equal one.

Another particularity of the VAE is the Reparameterization Trick. In the bottleneck of the network, there is the sampling of the distribution. Since backpropagation cannot run through a sampling node, a workaround is needed. Thus in order to be able to run backpropagation through the network in the training process, the Reparameterization Trick is used.

The reparameterization trick splits the latent vector z into learning components and a stochastic component. The stochastic component, ϵ , is normally distributed $(0,1)$.

$$z = \mu + \sigma \epsilon$$

A disadvantage of VAEs is that due to the use of the average difference in the loss function, it tends to produce blurrier images than GANs (discussed in the next section), since the network is not getting rewarded for sharpness, but rather for the smallest sum of individual errors, which makes a blurry image perform well. When images are blurry, the peaks of colours tend to be dissipated into its neighbouring pixels. In the context of a stress field, it can be a great disadvantage, since peak stresses (the defining feature for the optimization) could be downplayed by the decoder, making a design seem feasible when it is not.

An advantage of VAEs is that they allow the latent space to be relatively easily explored. A mean-normal distribution pair of vectors yields a new manifestation of the learnt model.

14.3.4 Conditional Generative Adversarial Network (cGAN)

A Generative Adversarial Network (GAN) finds features of images (i.e. its input and output are in the form of R(ed), G(reen), and B(lue) channels of $m \times n$ pixels)(Nie et al., 2020). Features are, for example, edges (in particular directions) or colour changes, which indicate useful information to solve the task that is being posed to the algorithm. If the task is segmentation, then features which pinpoint when one object stops and another begins will be useful. The network learns its filters during the training process. The effect of the first layer of filters on the input image can still be meaningful to the human eye (e.g. it highlights horizontal changes from dark to light, and darkens horizontal changes from light to dark, while blurring out the rest). As the layers get deeper, we are less able to discern their meaning by looking at

them. To the computer, however, they become more loaded in compressed information. Each pixel has information over a larger area of the original input with regards to a particular feature. A feature is useful if it provides the decoder with the information it needs in order to produce outcomes that can fool the discriminator.

GANs have been used extensively in image applications, but have also found their way in engineering in general and topology optimization specifically. TopologyGAN introduces the idea of using physical fields (e.g. Von Mises stress distribution) over the initial (i.e. unoptimized) domain as additional input (on top of the boundary conditions and loads) in a model that learns to output optimized topologies (Nie et al., 2020). Even though – due to the lack of a data set of optimized designs – it is not possible to utilize their entire framework for a glass application, the intuition that including the stresses and deformation fields of the entire domain could result in more accurate predictions of the stress fields for (partially) optimized topologies by the cGAN can still be adopted. Alternatively, in the present work, cGAN could be used to speed up the mapping of topology to stress field.

14.4 Brain-storming workflows/ The paths not taken

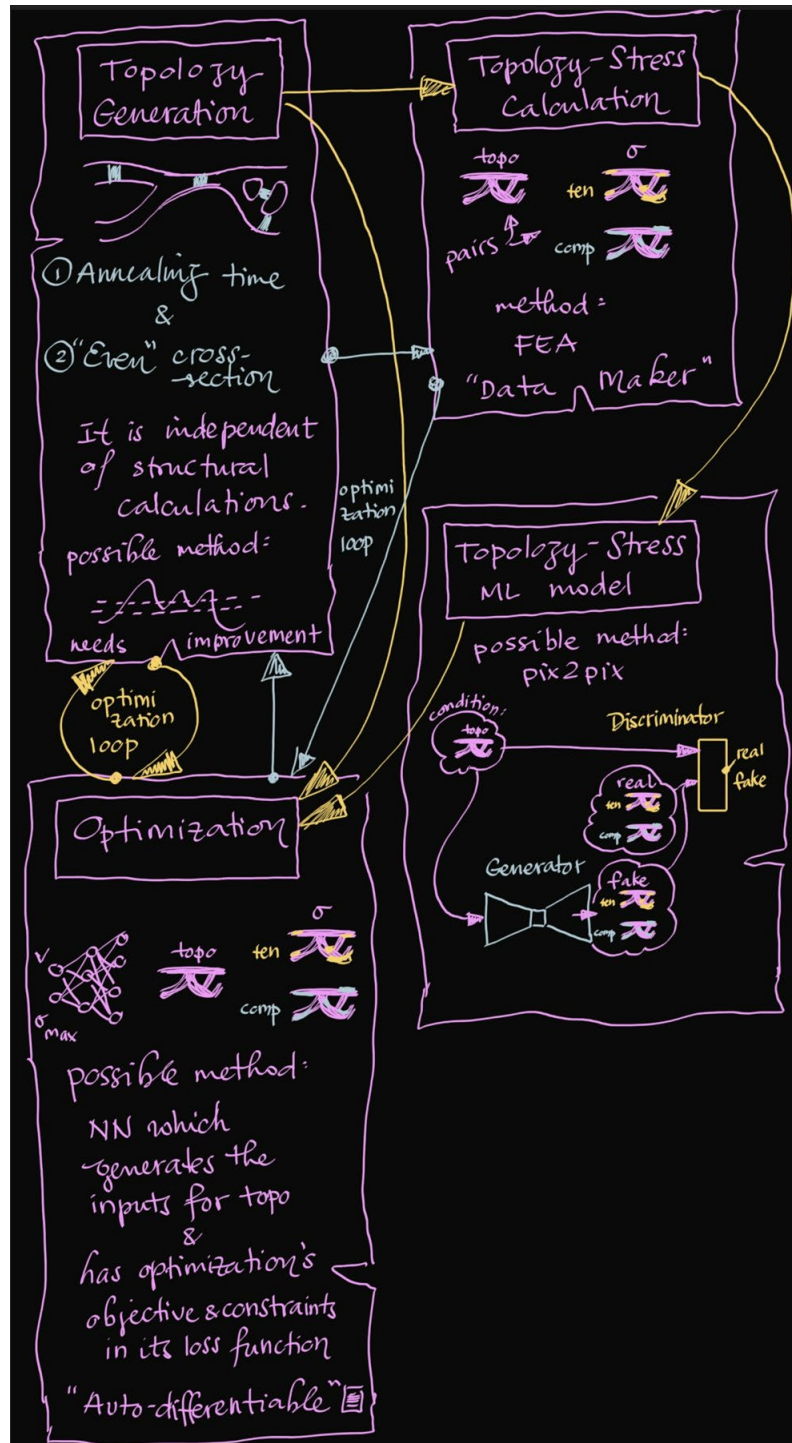


Figure 14.10: Brainstorming workflows

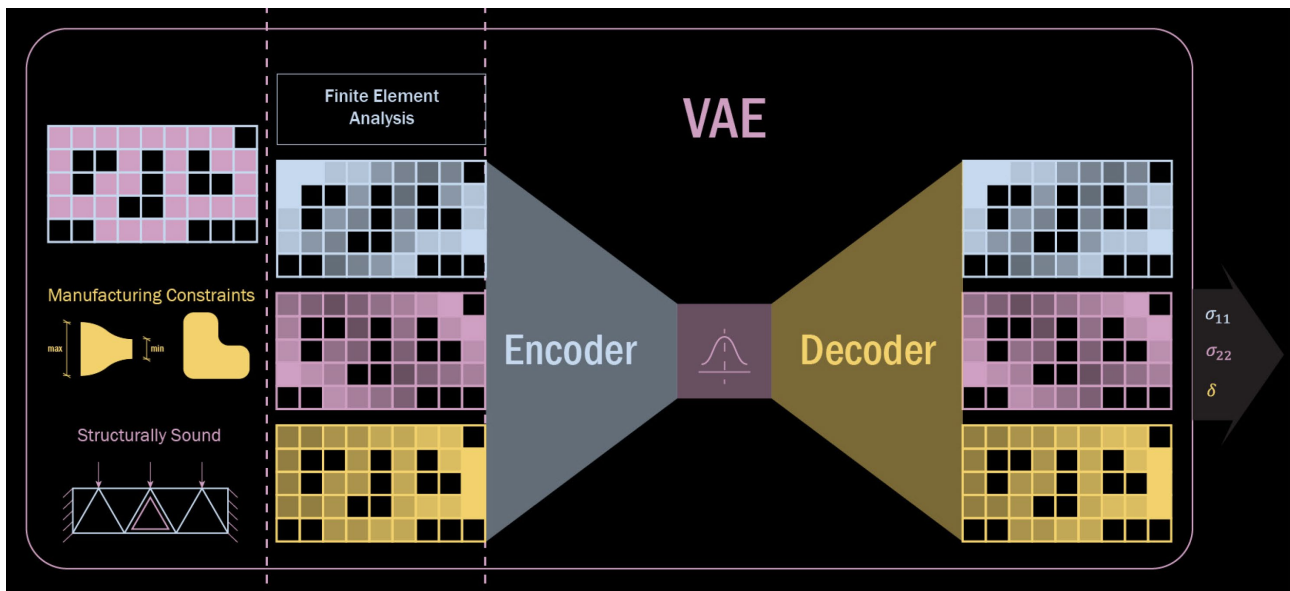


Figure 14.11: Training a VAE on stress and displacement fields

14.5 Loss graphs of the material distributions chosen for post processing

A note should be made that the final iteration is not necessarily the best iteration. But in the case of the examples presented here, it is pretty close. The results for the *best* iteration have been presented in the Results chapter.

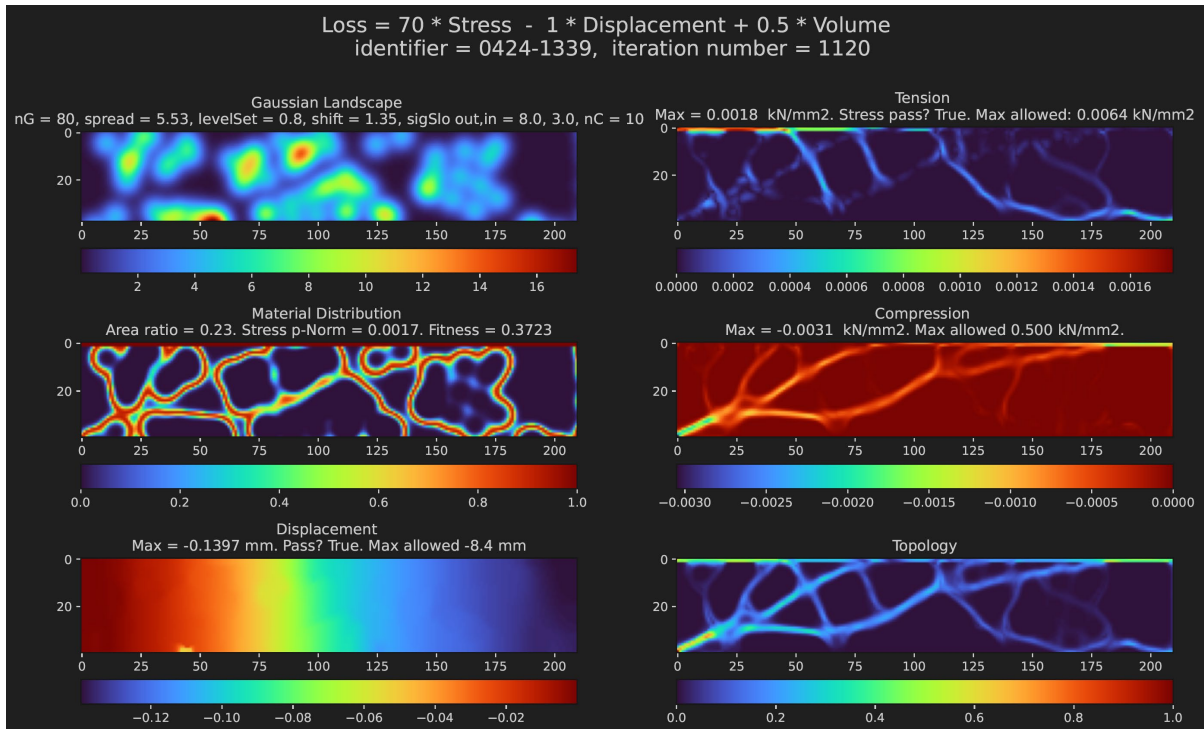


Figure 14.4.1: Results for 0424-1339, optimized with SGD.

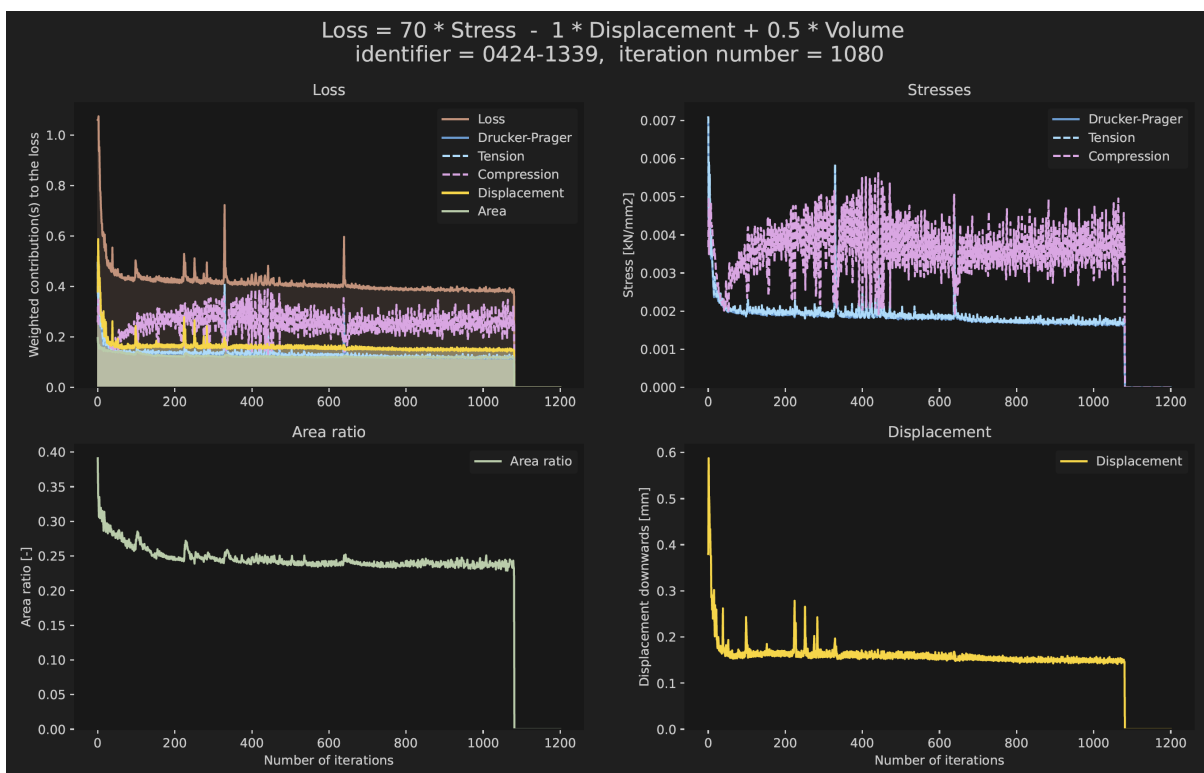


Figure 14.4.2: Loss graphs for 0424-1339, optimized with SGD.

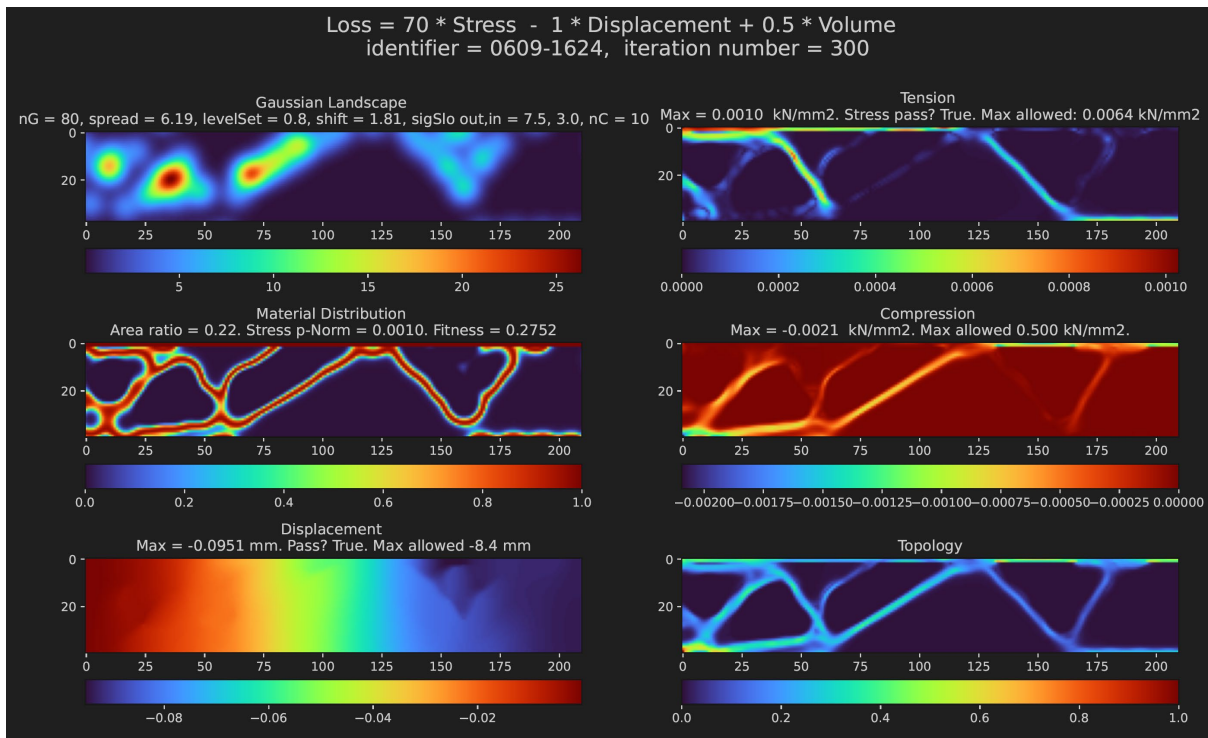


Figure 14.4.3: Final iteration results for 0609-1624, optimized with Adam.

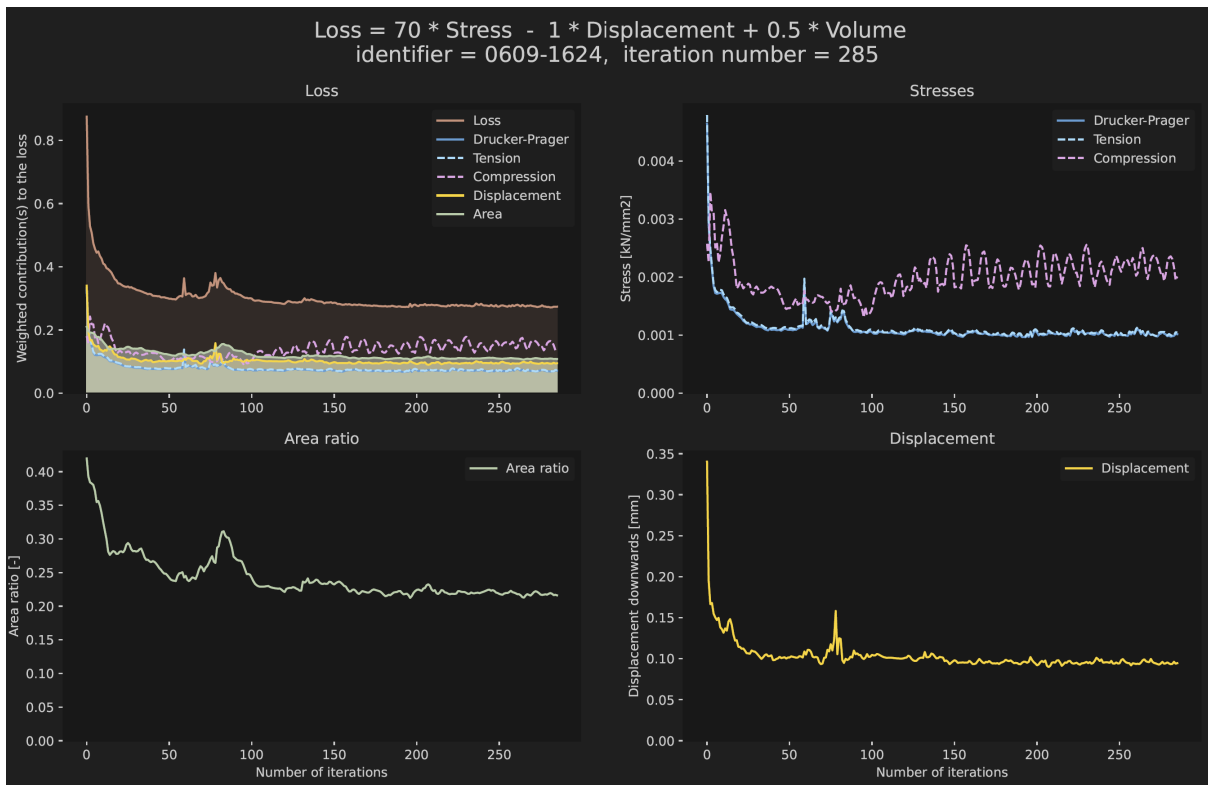


Figure 14.4.4: Loss graphs for 0609-1624, optimized with Adam.

14.6 Reflection

1) How are research and design related?

Intimately. Research informed, and provided a foundational groundwork for design. In turn, experimentation (which is a form of research!) was an intrinsic part of the design process. The insights gained from research of optimization methodologies was what allowed the creation of a new workflow, more adapted to the material and context at hand. Previous work done by Koniari and Schoenmaker was a guiding hand in the initial phase of the project, and the application of the finite element method. Furthermore, ideas gained from research were applied in unusual places. As when the concept of a sigmoid, most commonly used as an activation function when training neural networks, is applied as a slicing function for a topology definition.

2) How is the graduation topic positioned in the studio?

It is positioned more towards the computational side in the computational-design spectrum. The focus of the project was then the design of the algorithm, as opposed to the design of a final physical project. It is deeply connected to the studio in its use of glass as a structural material and the inclusion of glass' biggest weakness, tensile stresses, as part of the objective to be minimized.

3) How did the research approach work out & did it lead to the results you aimed for?

The research approach was very flexible. New ideas and findings were allowed to change the course of the project. This was, in my opinion, beneficial to the design of a creative workflow that was achievable within the timeframe of the master thesis.

Yes, I believe that it did work out. The optimization workflow indeed minimized the stresses, and was designed to take into account the needs of cast glass from the very beginning, creating soft constraints on manufacturing criteria, and thus also reducing the computational power needed.

The results were beautiful and structurally sound.

4) To what extent are the results applicable in practice?

Moderately. For the application of the results in practice, more rigorous structural analysis is advised. Only one load case was taken into consideration, and no peak loads were considered. This should definitely be done before any physical implementation.

Building with cast glass is a very niche application. It is not intended to be widespread, but more of a sculptural, artistic structural expression. And as such, this project does fulfill the requirement, especially considering the context of the case study application: a Museum.

5) To what extent has the project innovation been achieved?

The project is fairly innovative. The workflow is a combination of parts which did not come up in any of the literature research conducted. The material distribution definition was created from scratch, and the connection between this definition and the optimizer using the adapted sigmoid is also new. The implementation of the workflow using automatic differentiation in combination with the Level Set method was novel.

6) What is the impact of the project on sustainability & sustainable development?

Overall, there was a huge reduction in the number of parameters needed for the optimization, and this reduces the amount of computational power needed. Additionally, the use of cast glass as a structural material is on the one hand unsustainable because of the high temperatures needed for manufacturing, and on the other hand sustainable because of the endless recyclability potential.

7) What is the socio-cultural and ethical impact of the project and what is the relation between this project and the wider social context?

The project creates a very unique structural object. It can become a part of an experience and cause curiosity or admiration.

It provides the framework for running topology optimization with significantly less parameters, making it more accessible for people without access to a supercomputer.

8) How does the project affect architecture and the built environment?

It broadens the horizon of an important structural topic: topology optimization. It shows that one does not need to be performing such tasks in a fixed workflow, but that experimentation and critical thinking can provide new ways of optimizing structures, and that these different methods also leave a mark on the structures themselves.

14.7 Stress paths

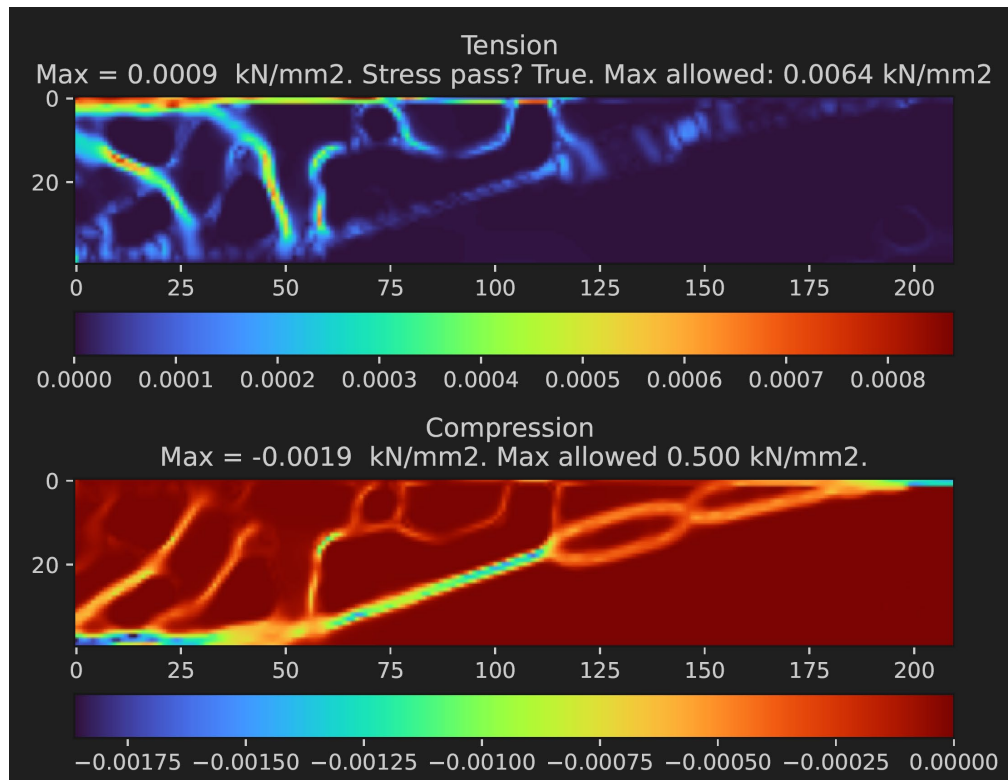


Figure 14.10: Stress paths

Cool to see the stresses on the main diagonal member, with the compression (long diagonal lines) perpendicular to the tension (short diagonal lines).