Department of Construction Sciences
Solid Mechanics

ISRN LUTFD2/TFHF-20/TFHF-5239-SE(1-67)

Topology optimization of transient thermo-mechanical problems using multiple materials

Master's Dissertation by

Olov Günther-Hanssen

Supervisors:
Mathias Wallin, Division of Solid Mechanics

Examiner:
Håkan Hallberg, Division of Solid Mechanics

Printed by Media-Tryck AB, Lund, Sweden
For information, address:
Division of Solid Mechanics, Lund University, Box 118, SE-221 00 Lund, Sweden
Webpage: www.solid.lth.se

Abstract

Topology optimization is a powerful method for finding optimized designs for a variety of problems. In this work, thermo-mechanical problems are studied in particular and solved under transient conductive heat transfer using 2 materials plus void in the optimization.

In the first part of the thesis, the theoretical background for a generic transient thermo-mechanical topology optimization problem is introduced. The thermo-mechanical field is modelled using have a one-way coupling between the temperature field to the displacement field and is solved with the Finite Element Method for small deformations. To render designs with a minimal length scale and clear boundaries, design filtering is used together with a robust formulation, which is reliant on Heaviside projections.

The second part goes more into the details of the implementation in Matlab and the test cases used as well as a real-world application: optimizing a thermally actuated disassembly mechanism.

The results are verified against external previous results and are similar in the features. The optimal design for the real-world application is from a topology optimization perspective satisfactory but show very small displacements under the thermal load. As future work, large deformations are suggested to be implemented.

Acknowledgements

This master's thesis was done under the Division of Solid Mechanics at Lund University. I would like to thank my supervisor, professor Mathias Wallin, for the support and inspiration for research he has given me along the journey. I would also like to thank all the other colleagues at the department, especially everyone who's been in the master's thesis office during the spring and early fall, making the day to day life a blast and shaping fruitful discussions about topology optimization. A special thank you goes to Oscar Blomqvist and Matthis Schneider who held me company in the early suites of the COVID-19 pandemic when the whole wide worms seemed upside down.

My deepest gratitude also goes to Steven G. Johnson, the main author of the open-source projects *NLopt* and the authors of the open-source meshing software *GMSH* for making your useful routines available for anyone to use.

Notations

	Description	**Unit**
$\boldsymbol{q}$	Heat flux vector	$\mathrm{W\,m^{-2}}$
ρ	Density	$\mathrm{kg\,m^{-3}}$
T	Temperature	K
c_p	Specific heat capacity	$\mathrm{J\,kg^{-1}\,K^{-1}}$
s	Volumetric heat capacity	$\mathrm{J\,m^{-3}\,K^{-1}}$
Q	Internal heat generation	$\mathrm{W\,m^{-3}}$
κ	Heat conductivity	$\mathrm{W\,m^{-1}\,K^{-1}}$
t	Time	s
α	Coefficient of thermal expansion (CTE)	$\mathrm{K^{-1}}$
r	Thermal stress coefficient	$\mathrm{Pa\,K^{-1}}$
$\boldsymbol{u}$	Displacement vector	m
$\boldsymbol{\sigma}$	Stress tensor	Pa
$\boldsymbol{t}$	Surface traction vector	Pa
$\boldsymbol{\varepsilon}$	Strain tensor	-
$\boldsymbol{D}$	Material stiffness tensor	Pa
nn	Number of nodes	
nel	Number of elements	
nen	Number of element nodes	
N	Number of time steps	
M	Number of constraints	

Table 1: Descriptions and notations of the physical variables used in the report

Mathematical notations

Notation	**Description**
$\frac{\partial(\cdot)}{\partial x}$	Explicit derivative (a.k.a. partial derivative)
$\frac{D(\cdot)}{Dx}$	Implicit derivative (a.k.a. total derivative)
$\dot{x} = \frac{Dx}{Dt}$	Time derivative
$\nabla = \left(\frac{D(\cdot)}{Dx}, \frac{D(\cdot)}{Dy}, \frac{D(\cdot)}{Dz}\right)^T$	Vector differential operator
$\boldsymbol{I}$	Identity matrix
$\frac{D(\cdot)}{D\boldsymbol{x}} = \left(\frac{D(\cdot)}{Dx_1}, \frac{D(\cdot)}{Dx_2}, \dots\right)^T$	Vector derivative
$\cdot \big\vert_{\Omega}$	Evaluated at Ω
$\vert\vert \cdot \vert\vert$	Euclidean distance (L2 norm)
$\equiv$	Equality by definition

Table 2: Descriptions of the mathematical notation used in the report

Contents

Chapter 1

Introduction

Applied with a change in temperature, materials expand, and in some rare cases they contract. This property called thermal expansion most commonly causes headaches for engineers, as in the case with sun kinks on rail tracks where the heat of the sun causes the rails to buckle. But this intrinsic feature can also be used in ingenious designs. Example of this is the bimetallic thermometer, which uses two materials with different thermal expansion coefficients to cause a bending force in the instrument when subject to a temperature change. Another interesting application is micro grippers, where the joule heating from an applied electrical current causes a local temperature rise which in turn results in a bending force, as seen in figure 1.1. This force could then be used to grip a carbon nanofiber ($\sim$ 100 nm in diameter), for example.

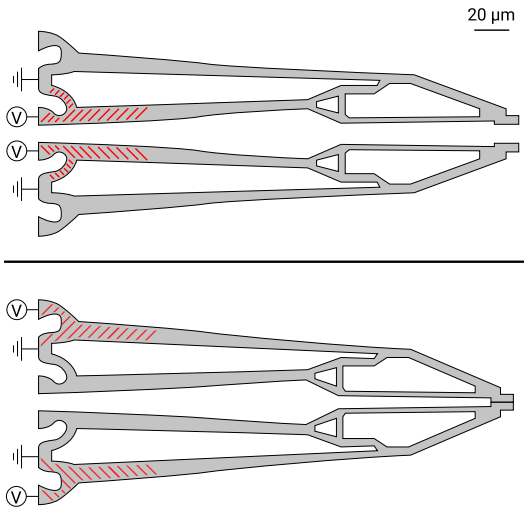


Figure 1.1: Working principle of an electrothermally actuated micro gripper. In the open state (above) the inner legs of the gripper are heated as marked in red. To reach the closed state (below), the outer legs are instead heated.

Both these designs are examples of thermal actuators. A thermal actuator uses thermally induced expansion as a mechanism for the creation of motion [1] and can be at the macro as well as the micro scale. The thermal expansion effect is what creates the coupling between the thermal field and the mechanical (displacement) field.

However, the design of a perfect thermal actuator may be non-intuitive and hard to find. A structured and powerful tool for finding the optimal design which increases the performance or have a special behavior is topology optimization. Since its birth in the late 1980's, topology optimization as a method has gained a lot of traction in both academia and industry. It most popular usage is within stiff structures and compliant mechanism, but the principles are just as applicable for thermal actuators. The general idea of topology optimization is to define a design domain with applied boundary conditions where the optimal design will be found, parametrize the domain in a suiting manner and insert the parameters into a non-linear optimization algorithm which minimizes a given objective function. The result is then interpreted for manufacturing. These steps are illustrated in figure 1.2.

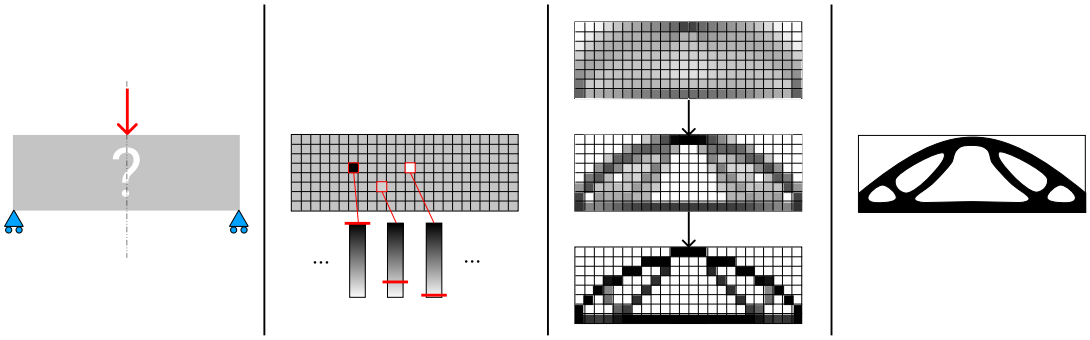


Figure 1.2: Overview of the topology optimization process. The first step is the definition of the design domain and the boundary conditions. The second step is the parametrization, followed by the third step where the optimizer iteratively finds a better and better design. The last step is the interpretation of the optimized design.

Topology optimization has successfully been used to design high-performance thermal actuators [2]. However, most of the optimization is done assuming steady-state conditions and single material designs. To take time-dependent effects into consideration when optimizing thermal actuators, transient analysis of the heat transfer must be considered and integrated into the optimization process. Multiple materials might as well extend the potential of the designs found.

The report is divided into 6 chapters. Chapter 2 covers all the theory relevant for the thesis, including the FE formulations for the thermal system and the mechanical system, the density based topology optimization method and the general method for acquiring the sensitivities for a problem with respect to the design parameters. Chapter 3 covers the implementation of a Matlab program and the modular structure

of the codebase. In this chapter, the cases studied are also presented. In chapter 4, the results from the cases described in the previous chapter are presented. Chapter 5 covers the evaluation and discussion of the results and chapter 6 ends the report with a short section about the potential future work that could be taken up on.

1.1 Purpose and objective

This master's thesis seeks to develop and study a general framework for solving topology optimization problems related to transient thermo-mechanical systems, including optimizing for multiple materials. The research in this thesis is therefore at a basic level, exploring the fundamental potential of this topic. However, as proof of concept, the framework will be applied on thermal actuators. The idea of the framework is however to be general and would be suitable to use for any other thermo-mechanical topology optimization problem with a time-transient effect, for example time-transient thermal stress optimization.

To develop and evaluate this framework in a structured way, the problem is divided into several sub-problems. The first would be to develop a basic framework for solving topology optimization problems for transient heat conduction. Secondly, another framework for using topology optimization for finding compliant mechanism actuated by a uniform temperature field would be developed. Once these two frameworks would successfully be in place, they would be coupled to finally form a framework optimizing topologies of time-transient thermal actuators. Lastly, multiple materials would have to be incorporated into the framework.

The main components needed for these frameworks are numerical models of the governing equations, normally Finite Element Method (FEM) models; the optimizer of the mathematical problem derived from the topology optimization problem; derivations and implementation of the sensitivities of these before mentioned models; preprocessing of the geometries and post-processing of the results.

1.2 Scope

The thesis will be limited to small deformations due to thermal expansion and use the linear Finite Element Method theory alongside linear isotropic material models. Materials properties are assumed to be temperature independent to simplify the solution of the governing equations. The final framework is set to work for 3D geometries, both structured and unstructured finite element discretizations.

1.3 Previous work

The concept of topology optimization was first introduced back in 1988 by Bendsøe and Kikuchi [3], which demonstrated that not only the shape of a structure could be optimized by systematic numerical methods, but the topology as well. In [3], the homogenization method was used to minimize the elastic energy of a design subject to external forces. Since then, the field of topology optimization has greatly evolved and is now used widely in industry to find new, non-intuitive designs of everything from the internals of airplane wings, to microscopical grippers.

Topology optimization has also proved useful in optimizing thermal systems, summarized in a survey paper by Dbouk [4]. Here the studies are divided by whether they model conductive, convective or conjugated heat transfer. Of the heat conduction systems mentioned in the paper, only one study by Zhuang et. al. have taken transient effects into consideration when trying to minimize the heat compliance [5]. The same author did later publish an article [6] where the objective function was the maximum temperature and multiple materials were also distributed in the design. Long and Wang have also studied the optimization with multiple materials and time-transient effect [7]. In a study by Wu and Zhang, the minimization of the maximum temperature is further studied along with the effect of the transient time period [8].

Looking at compliant mechanism, a recent survey paper was published which covers the state of the art related to compliant mechanism [9], including thermal actuators. The study of thermal actuators is quite extensive, however, papers taking the effect of time into consideration are relatively rare. What is especially interesting is a study done in 2004 by Li et. al which looks at thermal actuators under time-transient effects [10]. It is the first study to take transient effects into account. Moreover, Sigmund systematically studied the field of thermal actuators with non-linear effects and multi-materials and also derived a theoretical performance limit for any thermal actuator [11] [12]. In a conference paper [13], a transient thermomechanical system is optimized on and at the same time, a new penalization optimization method is described. The review paper on compliant mechanism also covers the usage of multiple materials in the optimization, showcasing the alternative techniques for interpolating the material properties of the models.

Chapter 2

Theory

2.1 Transient conductive heat transfer

2.1.1 Governing equations

The time-dependent heat equations are generally derived from the conservation of energy on an infinitesimal element, stating that the net power entering the element through its boundary in addition to the power being generated inside the element, should equal the change of internal energy in the element with respect to time. By denoting the internal heat generation as Q [$\mathrm{W\,m^{-3}}$], the heat flux vector as $\boldsymbol{q}$ [$\mathrm{W\,m^{-2}}$] and temperature as T [K], this statement can be expressed as the equation

$$-\nabla \cdot \boldsymbol{q} + Q = \rho c_p \dot{T}, \tag{2.1}$$

where $\dot{T}$ is the time derivative of T and ρ [$\mathrm{kg\,m^{-3}}$] and c_p [$\mathrm{J\,kg^{-1}\,K^{-1}}$] are the density and the heat capacity respectively. All quantities above are assumed to be functions of space and time, except the density and the heat capacity, which are assumed to only be functions of space.

For purely conductive heat transfer, the macroscopic relationship between the heat flux and the gradient of the temperature can be described with Fourier's law:

$$\boldsymbol{q} = -\boldsymbol{\kappa} \nabla T. \tag{2.2}$$

The minus sign shows that the heat flows to the regions with lower temperature, hence a negative temperature gradient. The second order positive definite tensor $\boldsymbol{\kappa}$ [$\mathrm{W\,m^{-1}\,{}^{\circ}C^{-1}}$] is called the thermal conductivity and in a Cartesian system, it is represented as

$$\boldsymbol{\kappa} = \begin{bmatrix} \kappa_{xx} & \kappa_{xy} & \kappa_{xz} \\ \kappa_{yx} & \kappa_{yy} & \kappa_{yz} \\ \kappa_{zx} & \kappa_{zy} & \kappa_{zz} \end{bmatrix}. \tag{2.3}$$

For the rest of this report, thermally isotropic material will be considered, where the matrix consists of only 1 independent component, κ, therefore making $\boldsymbol{\kappa} = \kappa \boldsymbol{I}$ where $\boldsymbol{I}$ denotes the identity matrix.

With Fourier's law inserted into (2.1) and with all the terms moved to the left hand side yields

$$\nabla \cdot (\boldsymbol{\kappa} \nabla T) + Q - \rho c_p \dot{T} = 0, \tag{2.4}$$

which is referred to as the *transient (conductive) heat equation*. As conduction is the only heat transfer mechanism considered, this is the single governing equation dictating how the temperature field evolves over time.

For the governing equations to be solvable, a body of interest must be specified alongside boundary conditions. For an arbitrary body, the physical space it occupies may be denoted Ω, see figure 2.1. The body is subject to a prescribed temperature at part of the boundary of Ω, $\partial\Omega_T$. A boundary condition of this type is essential to solve the system, and is therefore named *essential boundary condition* or *Dirichlet boundary condition*. At another part of the boundary, $\partial\Omega_q$, the heat flux is known. This kind of boundary condition is called *natural boundary condition* or *Neumann boundary condition*. As transient heat is studied, an initial condition must be stated. All together the problem is to find the temperature field $T(\boldsymbol{x}, t)$ which satisfies the following equations

$$\begin{cases} \nabla \cdot (\boldsymbol{\kappa} \nabla T(\boldsymbol{x}, t)) + Q(\boldsymbol{x}, t) - \rho c_p \dot{T}(\boldsymbol{x}, t) = 0 \quad \forall \boldsymbol{x} \in \Omega,\ t \in [0, t_f] \\ T(\boldsymbol{x}, t) = T'(\boldsymbol{x}, t) \quad \forall \boldsymbol{x} \in \partial\Omega_T,\ t \in [0, t_f] \\ q_n(t) = -\boldsymbol{\kappa} \nabla T(\boldsymbol{x}, t) \cdot \boldsymbol{n} = q_n'(\boldsymbol{x}, t) \quad \forall \boldsymbol{x} \in \partial\Omega_q,\ t \in [0, t_f] \\ T(\boldsymbol{x}, 0) = T_0(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega \end{cases}, \tag{2.5}$$

where the prime on T' and q_n' denotes that these are prescribed quantities on the boundaries $\partial\Omega_T$ and boundary $\partial\Omega_q$, respectively. T_0 is the initial temperature distribution and t_f denotes the time duration for which the temperature field will be solved.

Strong formulation to weak formulation

The transient conductive heat equation as it is expressed in (2.4), is given in a strong (differential) formulation. To solve the thermal problem with the Finite Element

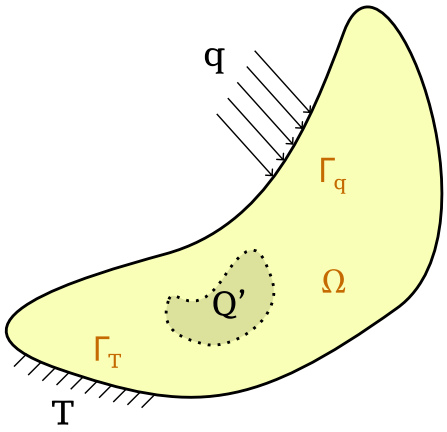


Figure 2.1: Description of the body and the boundaries for the thermal problem.

Method, it must instead be given in a weak (variational) formulation. What follows are the steps for transforming the strong formulation to a weak formulation.

First, the notion of a *virtual temperature* $\delta T(\boldsymbol{x}, t)$ is introduced, which is an imaginal variation of T satisfying the Dirichlet and Neumann boundary conditions. The first step is then to multiply (2.4) by the virtual temperature, namely

$$\delta T(\nabla \cdot (\boldsymbol{\kappa} \nabla T) + Q - \rho c_p \dot{T}) = 0. \tag{2.6}$$

As (2.6) holds for every point in Ω it may be integrated over the entire body.

$$\int_\Omega \delta T(\nabla \cdot (\boldsymbol{\kappa} \nabla T) + Q - \rho c_p \dot{T}) \, d\Omega = 0 \tag{2.7}$$

With the goal of removing the divergence operator from the equation above, the term related to the divergence is integrated with the Gauss-divergence theorem, stating that

$$\int_\Omega \delta T \nabla \cdot (\boldsymbol{\kappa} \nabla T) \, d\Omega = \int_{\partial\Omega} \delta T (\boldsymbol{\kappa} \nabla T) \cdot \boldsymbol{n} \, d(\partial\Omega) - \int_\Omega \nabla(\delta T) \cdot (\boldsymbol{\kappa} \nabla T) \tag{2.8}$$

The symbol $\boldsymbol{n}$ denotes the normal of the body's surface. Inserting (2.8) into (2.7) and denoting the normal heat flux at the surface $\boldsymbol{q} \cdot \boldsymbol{n}$ as q_n results in the weak, or variational, formulation

$$\int_{\partial\Omega} \delta T q_n \, d(\partial\Omega) + \int_\Omega (\nabla(\delta T))^T (\boldsymbol{\kappa} \nabla T) - \int_\Omega \delta T Q \, d\Omega + \int_\Omega \delta T \rho c_p \dot{T} \, d\Omega = 0 \tag{2.9}$$

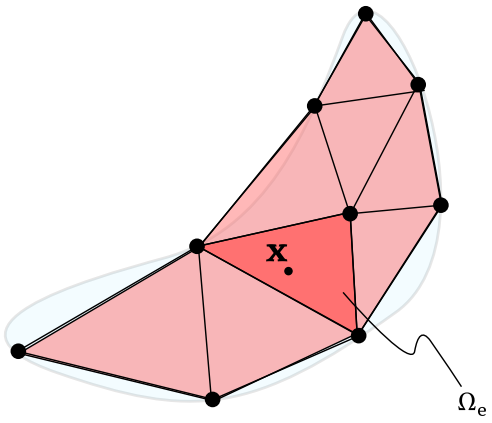


Figure 2.2: Spatial approximation of the body Ω.

The strong and weak form are equivalent, but the weak formulation is beneficial in the sense that no second order derivative of the temperature is present. It is therefore the weak formulation that is the basis for the approximation and transformation to a finite set of algebraic equations, which will be the subject of the next section.

2.1.2 Solving the heat equations with the Finite Element Method

Discretizing in space

The variational formulation is very difficult to solve analytically for more than simple geometries. To handle complex geometries, numerical methods must therefore be employed. Without knowing how the field looks like, it is then discretized in space, and approximated as the interpolation of these discrete values. At the same time, the body is approximated to be made up by finite number of element, cf. figure 2.2. These elements are to no surprise called *finite elements*. The discrete points inside the body and on its boundary are called *nodes* and the nodes together with the wireframe connecting the nodes and defining the finite elements, form a *mesh*.

At the nodes, the field variables are defined as scalar values, or nodal temperatures. The information about the temperature field is replaced by a vector of nodal values, namely

$$\boldsymbol{T} = \begin{bmatrix} T_1 \\ T_2 \\ \vdots \\ T_{nn} \end{bmatrix}, \tag{2.10}$$

where T_i denotes the temperature at node i and nn is the total number of nodes.

The temperature field is then approximated as an interpolation of these values

$$T(\boldsymbol{x}, t) \approx \sum_{i=1}^{nn} N_i(\boldsymbol{x}) T_i(t) = \boldsymbol{N}(\boldsymbol{x})\, \boldsymbol{T}(t), \quad \boldsymbol{N} = \begin{bmatrix} N_1 & N_2 & \dots & N_{nn} \end{bmatrix} \tag{2.11}$$

$\boldsymbol{N}$ is the global shape function matrix and N_i shape functions. The shape function N_i has the property of equaling 1 at node i and 0 at all other nodes, and interpolates the values in between. Notice also how (2.11) separates the approximation in space and time into the shape functions and nodal values respectively. A consequence is that the same interpolation can be used for the time derivative, $\dot{T} = \boldsymbol{N}(\boldsymbol{x})\, \dot{\boldsymbol{T}}(t)$, where $\dot{\boldsymbol{T}}(t)$ is the time derivative of the nodal temperatures.

In between nodes, the interpolation from the shape functions decides how to interpolate the nodal temperature values. In figure 2.2, the position $\boldsymbol{x}$ is inside a finite element. The interpolation will then only take the nodal temperature values of that element into consideration. A special trick is then to reduce the globally sized shape function matrix $\boldsymbol{N}$ and temperature vector $\boldsymbol{T}$ into a locally sized shape function matrix and temperature vector.

$$T \approx \boldsymbol{N}(\boldsymbol{x})\, \boldsymbol{T}(t) = \boldsymbol{N}(\boldsymbol{x}) \boldsymbol{E}_e \boldsymbol{E}_e^T \boldsymbol{T}(t) = \boldsymbol{N}^e \boldsymbol{T}^e, \quad \forall \boldsymbol{x} \in \Omega_e \tag{2.12}$$

The expansion/reduction matrix for element e maps the local node numbering for element e to the global numbering/vice versa, see more in appendix A. In (2.12), $\boldsymbol{N}\boldsymbol{E}_e$ is substituted with the local shape function matrix for element e, $\boldsymbol{N}^e$, and $\boldsymbol{E}_e^T \boldsymbol{T}$ with the local nodal temperature vector, $\boldsymbol{T}^e$. In the local region of an element, these vectors hold all the relevant information for interpolating inside that element. $\boldsymbol{N}^e$ and $\boldsymbol{T}^e$ are have the dimension nen, number of element nodes. Using the local numbering of the nodes, these vectors may be written as

$$\boldsymbol{N}^e = \begin{bmatrix} N_1^e & N_2^e & \dots & N_{nen}^e \end{bmatrix} \quad \boldsymbol{T}^e = \begin{bmatrix} T_1^e \\ T_2^e \\ \vdots \\ T_{nen}^e \end{bmatrix}. \tag{2.13}$$

The superscript e marks that the numbering is local. The expansion matrix can later

map this numbering to the global numbering used in (2.10) and (2.11). Another important equality from (2.12) is

$$\boldsymbol{N}(\boldsymbol{x}) = \boldsymbol{N}^e(\boldsymbol{x})\boldsymbol{E}_e^T \quad \forall \boldsymbol{x} \in \Omega_e \tag{2.14}$$

The same discretization and interpolation method is used on the virtual temperature, yielding

$$\delta T(\boldsymbol{x}, t) = \boldsymbol{N}(\boldsymbol{x})\boldsymbol{\delta T}(t) \tag{2.15}$$

Inserting these approximations, (2.11) and (2.15), into the weak formulation (2.9) results in

$$\begin{aligned}\int_{\partial\Omega} \boldsymbol{\delta T}^T \boldsymbol{N}^T q_n \; d(\partial\Omega) + \int_\Omega \nabla(\boldsymbol{N}\boldsymbol{\delta T})^T(\boldsymbol{\kappa}\nabla(\boldsymbol{N T})) \\ - \int_\Omega \boldsymbol{\delta T}^T \boldsymbol{N}^T Q \; d\Omega + \int_\Omega \boldsymbol{\delta T}^T \boldsymbol{N}^T \rho c_p \boldsymbol{N}\dot{\boldsymbol{T}} \; d\Omega = 0\end{aligned} \tag{2.16}$$

As the nodal vectors, $\boldsymbol{\delta T}$, $\boldsymbol{T}$ and $\dot{\boldsymbol{T}}$ are independent on position, they may be moved outside the integrals.

$$\begin{aligned}\boldsymbol{\delta T}\Bigg[\int_{\partial\Omega} \boldsymbol{N}^T q_n \; d(\partial\Omega) + \int_\Omega (\nabla \boldsymbol{N})^T \boldsymbol{\kappa} \nabla \boldsymbol{N} d\Omega \, \boldsymbol{T} \\ - \int_\Omega \boldsymbol{N}^T Q' \; d\Omega + \int_\Omega \boldsymbol{N}^T \rho c_p \boldsymbol{N} d\Omega \, \dot{\boldsymbol{T}}\Bigg] = 0\end{aligned} \tag{2.17}$$

As the virtual temperature is arbitrary, leads to the equality

$$\int_{\partial\Omega} \boldsymbol{N}^T q_n \; d(\partial\Omega) + \int_\Omega \boldsymbol{B}^T \boldsymbol{\kappa} \boldsymbol{B} d\Omega \, \boldsymbol{T} - \int_\Omega \boldsymbol{N}^T Q \; d\Omega + \int_\Omega \boldsymbol{N}^T \rho c_p \boldsymbol{N} d\Omega \, \dot{\boldsymbol{T}} = 0 \tag{2.18}$$

Here he gradient of the shape functions, $\nabla \boldsymbol{N}$ is denoted with $\boldsymbol{B}$. To make the notation of the integrals simpler, some symbols are introduced, as seen below. $\boldsymbol{f}_l$ is called the boundary load vector, $\boldsymbol{f}_v$ the volume load vector, $\boldsymbol{K}_{TT}$ the thermal stiffness matrix and $\boldsymbol{C}$ the heat capacity matrix. (2.18) may then be written as in (2.23).

$$\boldsymbol{f}_l \equiv \int_{\partial\Omega} \boldsymbol{N}^T q_n \; d(\partial\Omega) \tag{2.19}$$

$$\boldsymbol{f}_v \equiv \int_\Omega \boldsymbol{N}^T Q \; d\Omega \tag{2.20}$$

$$\boldsymbol{K}_{TT} \equiv \int_\Omega \boldsymbol{B}^T \boldsymbol{\kappa} \boldsymbol{B} d\Omega \tag{2.21}$$

$$\boldsymbol{C} \equiv \int_\Omega \boldsymbol{N}^T \rho c_p \boldsymbol{N} \; d\Omega \tag{2.22}$$

$$\boldsymbol{f}_l + \boldsymbol{K}_{TT}\boldsymbol{T} - \boldsymbol{f}_v + \boldsymbol{C}\,\dot{\boldsymbol{T}} = 0 \tag{2.23}$$

Rearranging and adding the two load vectors together to a single load vector $\boldsymbol{f}$, makes

$$\boldsymbol{C}\,\dot{\boldsymbol{T}} + \boldsymbol{K}_{TT}\boldsymbol{T} = \boldsymbol{f} \tag{2.24}$$

This is the finite element formulation discretized in space. However, it also needs to be discretized in time to end up with a set of algebraic equations to solve. The next section covers that.

Discretizing in time

In the last section, the field variables were discretized into finite points in space. The result was equation (2.24). This equation has however still a time derivative present. To acquire a finite set of algebraic equations, the field variables are discretized in time as well. The domain $[0, t_f]$ is discretized into a set of time steps, $[t_0,\ t_2,\ \ldots\ ,\ t_N]$. At each time step, we define the quantities at that time as follows.

$$\boldsymbol{T}^{(n)} \equiv \boldsymbol{T}(t = t_n), \quad \dot{\boldsymbol{T}}^{(n)} \equiv \dot{\boldsymbol{T}}(t = t_n), \quad \boldsymbol{f}^{(n)} \equiv \boldsymbol{f}(t = t_n), \quad \forall n = 0, 1, \ldots, N \tag{2.25}$$

To eliminate the time derivative, equation (2.24) is integrated with respect to time between two arbitrary time steps t_{n-1} and t_n.

$$\int_{t_{n-1}}^{t_n} \boldsymbol{K}_{TT}\boldsymbol{T}(\tau)\; d\tau + \int_{t_{n-1}}^{t_n} \boldsymbol{C}\,\dot{\boldsymbol{T}}(\tau)\; d\tau = \int_{t_{n-1}}^{t_n} \boldsymbol{f}(\tau)\; d\tau \tag{2.26}$$

As the material parameters are assumed to be constant over time, leads to the matrices $\boldsymbol{K}_{TT}$ and $\boldsymbol{C}$ moving outside the integrals. The integration of the time derivative equals a difference in temperature, $\int_{t_{n-1}}^{t_n} \dot{\boldsymbol{T}}(\tau)\; d\tau = \boldsymbol{T}(t_n) - \boldsymbol{T}(t_{n-1})$, which together with (2.26) becomes

$$\boldsymbol{K}_{TT}\int_{t_{n-1}}^{t_n}\boldsymbol{T}(\tau)\;d\tau+\boldsymbol{C}(\boldsymbol{T}^{(n)}-\boldsymbol{T}^{(n-1)})\;d\tau=\int_{t_{n-1}}^{t_n}\boldsymbol{f}(\tau)\;d\tau \tag{2.27}$$

As the temperature function with respect to time, $\boldsymbol{T}(\tau)$, is sought after, it is unknown and so also the integral over it. The integral is therefore approximated as a weighted sum of the temperatures at the two bounds of the integral multiplied by the time difference:

$$\int_{t_{n-1}}^{t_n}\boldsymbol{T}(\tau)\;d\tau\approx(\theta\,\boldsymbol{T}^{(n)}+(1-\theta)\,\boldsymbol{T}^{(n-1)}))(t_n-t_{n-1}),\quad\theta\in[0,1] \tag{2.28}$$

The integration over the load vector $\boldsymbol{f}$ in (2.27) does not necessarily have to be approximated, as the full loading history should be known beforehand. It is however common that the load vector is approximated in the same fashion as the nodal temperatures. To keep it general, it is kept as is and denoted $\bar{\boldsymbol{f}}^{(n)}\equiv\frac{1}{\Delta t_n}\int_{t_{n-1}}^{t_n}\boldsymbol{f}(\tau)\;d\tau$. By using the notation Δt_n for the time difference at step n: t_n-t_{n-1}, and by inserting (2.28), (2.27) can take its final form

$$\boldsymbol{K}_{TT}(\theta\,\boldsymbol{T}^{(n)}+(1-\theta)\,\boldsymbol{T}^{(n-1)}))\Delta t_n+\boldsymbol{C}(\boldsymbol{T}^{(n)}-\boldsymbol{T}^{(n-1)})\;d\tau=\Delta t_n\bar{\boldsymbol{f}}^{(n)} \tag{2.29}$$

Arranging all the terms with $\boldsymbol{T}^{(n)}$ on the left hand side and all the other terms on the right hand side and dividing by Δt_n yields

$$(\theta\boldsymbol{K}_{TT}+\frac{1}{\Delta t_n}\boldsymbol{C})\,\boldsymbol{T}^{(n)}=(\boldsymbol{K}_{TT}(\theta-1)+\frac{1}{\Delta t_n}\boldsymbol{C})\,\boldsymbol{T}^{(n-1)}+\bar{\boldsymbol{f}}^{(n)} \tag{2.30}$$

The equation above follows the format where all the quantities on the right hand side are known, and the only unknown in the left hand side is $\boldsymbol{T}^{(n)}$. The temperatures in time can therefore be retrieved in an simple recursive matter, where only the initial temperatures $\boldsymbol{T}^{(0)}$ have to be known. In a residual format, the equation is instead written as

$$\begin{aligned}\boldsymbol{R}_T^{(n)}(\boldsymbol{T}^{(n)},\boldsymbol{T}^{(n-1)})&\equiv(\theta\boldsymbol{K}_{TT}+\frac{1}{\Delta t_n}\boldsymbol{C})\,\boldsymbol{T}^{(n)}-(\boldsymbol{K}_{TT}(\theta-1)+\frac{1}{\Delta t_n}\boldsymbol{C})\,\boldsymbol{T}^{(n-1)}-\bar{\boldsymbol{f}}^{(n)}\\&\equiv\boldsymbol{A}\,\boldsymbol{T}^{(n)}-\boldsymbol{B}\,\boldsymbol{T}^{(n-1)}-\bar{\boldsymbol{f}}^{(n)}=\boldsymbol{0}\end{aligned} \tag{2.31}$$

The most common choices of θ are 0, 1 or 1/2. These integration schemes are then called the *forward Euler*, *backward Euler* and *Crank-Nicolson* scheme, respectively.

Stiffness and heat capacity matrices

The thermal stiffness matrix and heat capacity matrix defined in (2.21) and (2.22) are integrals of the shape function matrix or the gradient of the shape function matrix, over the domain Ω. If the integrals' domain instead is divided into elements, the integral may be calculated as a sum of smaller integrals. The heat capacity matrix can be expressed as

$$\boldsymbol{C} \equiv \int_{\Omega} \boldsymbol{N}^T \rho c_p \boldsymbol{N} \; d\Omega = \sum_{e=1}^{nel} \int_{\Omega_e} \boldsymbol{N}^T \rho c_p \boldsymbol{N} \; d\Omega. \tag{2.32}$$

The equality from (2.14) may then be used on the integral, which gives

$$\boldsymbol{C} = \sum_{e=1}^{nel} \boldsymbol{E}_e \int_{\Omega_e} \boldsymbol{N}_e^T \rho c_p \boldsymbol{N}_e \; d\Omega \boldsymbol{E}_e^T = \sum_{e=1}^{nel} \boldsymbol{E}_e \boldsymbol{c}_e \boldsymbol{E}_e^T. \tag{2.33}$$

The equality states that the global shape function matrix in the local domain Ω_e equals the expansion matrix times the local shape function matrix. This gives that the heat capacity matrix can be seen as made up of the sum of local heat capacity matrices $\boldsymbol{c}_e$, which are then expanded to the global size. This process is also called assembly and is the same for the thermal stiffness matrix as well.

$$\boldsymbol{K}_{TT} = \sum_{e=1}^{nel} \boldsymbol{E}_e \int_{\Omega_e} (\nabla \boldsymbol{N}_e)^T \boldsymbol{\kappa} (\nabla \boldsymbol{N}_e) \; d\Omega \boldsymbol{E}_e^T = \sum_{e=1}^{nel} \boldsymbol{E}_e \boldsymbol{k}_e \boldsymbol{E}_e^T \tag{2.34}$$

The benefit of the assembly process is that the local stiffness matrices $\boldsymbol{c}_e$ and $\boldsymbol{k}_e$ are much smaller in size and are identical for congruent elements.

Final formulation

With the residual equation formulated in (2.31), the finite element formulation with boundary conditions can be compactly expressed as

$$\begin{cases} \boldsymbol{R}_T^{(n)}(\boldsymbol{T}^{(n)}, \boldsymbol{T}^{(n-1)}) = \boldsymbol{0} \quad n = 1, \ldots, N \\ T_i^{(n)} = T'(\boldsymbol{x}_i, t_n) \quad \forall \boldsymbol{x}_i \in \partial\Omega_T^{(n)}, \; n = 1, 2, \ldots, N \\ q_n(\boldsymbol{x}, t) = q_n'(\boldsymbol{x}, t) \\ T_i^{(0)} = T_0(\boldsymbol{x}_i) \quad i = 1, 2, \ldots, nn \end{cases} \tag{2.35}$$

This serves as the state equations for the temperature field. Whenever $\boldsymbol{R}_T^{(n)}(\boldsymbol{T}^{(n)}, \boldsymbol{T}^{(n-1)}) = \boldsymbol{0}$ is used later in the report, this actually refers to the whole system in (2.35).

2.2 Thermo-elasticity

2.2.1 Governing equations

The governing equation for the mechanical system stems from the law of mechanical equilibrium. In a continuum, it may be stated as three coupled equations, one for each dimension.

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} + b_x = 0 \tag{2.36}$$

$$\frac{\partial \sigma_{yx}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} + b_y = 0 \tag{2.37}$$

$$\frac{\partial \sigma_{zx}}{\partial x} + \frac{\partial \sigma_{zy}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} + b_z = 0 \tag{2.38}$$

To write these three equations into one matrix equation, the second order stress tensor is written in Voigt notation.

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} & \sigma_{yy} & \sigma_{zz} & \sigma_{xy} & \sigma_{xz} & \sigma_{yz} \end{bmatrix}^T \tag{2.39}$$

As the stress tensor is symmetric, makes it possible to write the above equations in a more compact manner

$$\tilde{\boldsymbol{\nabla}}^T \boldsymbol{\sigma} + \boldsymbol{b} = \boldsymbol{0}, \tag{2.40}$$

where the body force vector $\boldsymbol{b} = [b_x, b_y, b_z]^T$ and the matrix differential operator $\tilde{\boldsymbol{\nabla}}$ is defined as

$$\tilde{\boldsymbol{\nabla}} \equiv \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \end{bmatrix} \tag{2.41}$$

The vector equation (2.40) is the strong formulation of the governing equation for the elasticity problem. To connect (2.40) with the primary unknown $\boldsymbol{u}(\boldsymbol{x}, t)$, a constitutive relationship and a kinematic relation must be established. The kinematic relation is stated as

$$\boldsymbol{\varepsilon}(\boldsymbol{x}, t) = \tilde{\boldsymbol{\nabla}} \boldsymbol{u}(\boldsymbol{x}, t), \tag{2.42}$$

where the second order strain tensor is written in Voigt notation as $\boldsymbol{\varepsilon}$, just as the stress tensor in (2.39). The displacements are denoted $\boldsymbol{u} = [u_x, u_y, u_z]^T$ and are functions of space and time.

The strain can be divided into several contributions. In this thesis, elastic strains, $\boldsymbol{\varepsilon}^{el}$, induced by a stress field and thermal strain, $\boldsymbol{\varepsilon}^0$, induced by temperature change, are the only strains considered, making the total strain $\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}^{el} + \boldsymbol{\varepsilon}^0$. The thermal strain relates to a temperature change and do only contribute to the normal strains

$$\boldsymbol{\varepsilon}^0 = \begin{bmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \\ 0 \\ 0 \\ 0 \end{bmatrix} \Delta T = \boldsymbol{\alpha} \Delta T \tag{2.43}$$

α denotes the coefficient of thermal expansion (CTE) and ΔT is the temperature difference against a reference temperature for which no thermal strain is present: $\Delta T = T - T_R$. For an isotropic material, the coefficients of thermal expansions are all equal, giving $\boldsymbol{\alpha} = \alpha[1, 1, 1, 0, 0, 0]^T$.

Just as with the thermal problem, some kind of constitutive relationship must be introduced, in this case to couple the stresses with the strains. In this report, the constitutive relationship that will be used is Hooke's law, stating that there is a linear relationship between the stresses and strains, expressed in matrix format as

$$\boldsymbol{\sigma} = \boldsymbol{D} \boldsymbol{\varepsilon}^{el} \tag{2.44}$$

The constant material stiffness matrix $\boldsymbol{D}$ (note the difference towards the stiffness matrix $\boldsymbol{K}$) is in the general case of an anisotropic material a 6 x 6 matrix with 36 independent components. However, for a linear isotropic material, the number of independent components are 2, for example given with the engineering parameters Young's modulus E and Poisson's ratio ν. Combining (2.42), (2.43) and (2.44) together with the contribution to the total strain yields

$$\boldsymbol{\sigma} = \boldsymbol{D}(\tilde{\boldsymbol{\nabla}} \boldsymbol{u} - \boldsymbol{\alpha} \Delta T). \tag{2.45}$$

The body for which the governing equations are applied are denoted Ω, displayed in figure 2.3 together with the boundary conditions. The body is also subject to a prescribed displacement at part of the boundary of Ω, $\partial\Omega_u$. At another part

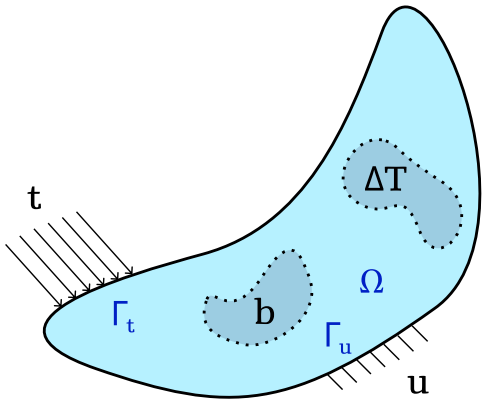


Figure 2.3: Description of the body and the boundaries for the mechanical problem.

of the boundary, $\partial\Omega_t$, the traction $\boldsymbol{t}$ is known. In addition to these Dirichlet and Neumann boundary condition, another boundary condition called *Robin boundary condition* is considered. This condition specifies how the traction is dependent on the displacements at a certain part of the boundary, called Ω_s. The governing equations together with the boundary conditions are presented below. The primary unknown to solve the system for is $\boldsymbol{u}(\boldsymbol{x}, t)$. Notice how no initial condition is needed as none of the governing equations are history dependent.

$$\begin{cases} \tilde{\boldsymbol{\nabla}}^T \boldsymbol{\sigma}(\boldsymbol{x}, t) + \boldsymbol{b} = \boldsymbol{0} \quad \forall \boldsymbol{x} \in \Omega,\ t \in [0, t_f] \\ \boldsymbol{\sigma}(\boldsymbol{x}, t) = \boldsymbol{D}(\tilde{\boldsymbol{\nabla}} \boldsymbol{u}(\boldsymbol{x}, t) - \boldsymbol{\alpha} \Delta T(\boldsymbol{x}, t)) \\ \boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{u}'(\boldsymbol{x}, t) \quad \forall \boldsymbol{x} \in \partial\Omega_u,\ t \in [0, t_f] \\ \boldsymbol{t}(\boldsymbol{x}, t) = \boldsymbol{\sigma}(\boldsymbol{x}, t) \cdot \boldsymbol{n} = \boldsymbol{t}'(\boldsymbol{x}, t) \quad \forall \boldsymbol{x} \in \partial\Omega_t,\ t \in [0, t_f] \\ \boldsymbol{t}(\boldsymbol{x}, t) = \boldsymbol{\sigma}(\boldsymbol{x}, t) \cdot \boldsymbol{n} = \boldsymbol{f}(\boldsymbol{u}(\boldsymbol{x}, t)) \quad \forall \boldsymbol{x} \in \partial\Omega_s,\ t \in [0, t_f] \end{cases} \tag{2.46}$$

Strong formulation to weak formulation

In a similar fashion as for the heat equation, including the introduction of a virtual displacement $\delta\boldsymbol{u}$, the weak form is derived for (2.40). For the full calculations, the reader is referred to [14]. The resulting weak form is then

$$\int_\Omega (\tilde{\boldsymbol{\nabla}} \delta \boldsymbol{u})^T \boldsymbol{\sigma}\ d\Omega = \int_{\partial\Omega} (\delta \boldsymbol{u})^T \boldsymbol{t}\ d(\partial\Omega) + \int_\Omega (\delta \boldsymbol{u})^T \boldsymbol{b}\ d\Omega, \tag{2.47}$$

where $\boldsymbol{t} = [t_x, t_y, t_z]^T$ is the surface traction on $\partial\Omega$. Inserting the constitutive relationship into (2.47) yields the final weak formulation for the elasticity problem.

$$\int_\Omega (\tilde{\boldsymbol{\nabla}}\delta\boldsymbol{u})^T \boldsymbol{D}(\tilde{\boldsymbol{\nabla}}\boldsymbol{u} - \boldsymbol{\alpha}\Delta T)\ d\Omega = \int_{\partial\Omega} (\delta\boldsymbol{u})^T \boldsymbol{t}\ d(\partial\Omega) + \int_\Omega (\delta\boldsymbol{u})^T \boldsymbol{b}\ d\Omega \tag{2.48}$$

2.2.2 Solving the equilibrium equations with the Finite Element Method

As with the thermal problem, the mechanical problem must be discretized in space and time to be solved. The same partitioning into finite elements is done for the mechanical problem.

In contrast to the thermal problem, where the unknown temperature field is a scalar field, the displacements field is a vector field. Every node in the mesh then have 3 degrees of freedom, one for each component of the displacement. When ordering all these into a vector, one option is to stack the components for each nodes on top of each other and insert into a column vector. The nodal displacement vector is denoted with a bar to differentiate it from the continuous displacements field variable.

$$\bar{\boldsymbol{u}} = \begin{bmatrix} u_{1,x} \\ u_{1,y} \\ u_{1,z} \\ u_{2,x} \\ \vdots \\ u_{nn,z} \end{bmatrix} \tag{2.49}$$

, where $u_{i,x}$ denotes the displacement in the x-direction at node i et cetera.

The interpolation of these values is then done similarly to what was done in (2.11).

$$\boldsymbol{u}(\boldsymbol{x},t) \approx \sum_{i=1}^{nn} N_i(\boldsymbol{x}) \begin{bmatrix} u_{i,x}(t) \\ u_{i,y}(t) \\ u_{i,z}(t) \end{bmatrix} \tag{2.50}$$

Writing this as a matrix product, results in the following global shape function matrix

$$\boldsymbol{u}(\boldsymbol{x},t) = \underline{\boldsymbol{N}}(\boldsymbol{x})\bar{\boldsymbol{u}}(t), \quad \underline{\boldsymbol{N}} = \begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & \\ 0 & N_1 & 0 & 0 & N_2 & 0 & \dots \\ 0 & 0 & N_1 & 0 & 0 & N_2 & \end{bmatrix} \tag{2.51}$$

The underlining of the shape function matrix, $\underline{\boldsymbol{N}}$, is used to distinct it from the one used for the temperature in (2.11). Similarly, the global shape function matrix and nodal displacement vector may be reduced to local size as with the temperature. The only difference is then that $\boldsymbol{E}_e$ now maps a larger set of degrees of freedom. The resulting shape function matrix and nodal element displacement vector look like

$$\underline{\boldsymbol{N}}^e = \begin{bmatrix} N_1^e & 0 & 0 & N_2^e & 0 & 0 & \dots \\ 0 & N_1^e & 0 & 0 & N_2^e & 0 & \dots \\ 0 & 0 & N_1^e & 0 & 0 & N_2^e & \dots \end{bmatrix} \quad \bar{\boldsymbol{u}}^e = \begin{bmatrix} u_{1,x}^e \\ u_{1,y}^e \\ u_{1,z}^e \\ \vdots \\ u_{nen,z}^e \end{bmatrix} \tag{2.52}$$

The same discretization and interpolation method is used on the virtual displacement

$$\delta \boldsymbol{u}(\boldsymbol{x}, t) = \underline{\boldsymbol{N}}(\boldsymbol{x}) \delta \bar{\boldsymbol{u}}(t) \tag{2.53}$$

The approximations for the displacement and virtual displacement is inserted into (2.48).

$$\begin{aligned} \int_\Omega (\tilde{\boldsymbol{\nabla}}(\underline{\boldsymbol{N}} \delta \bar{\boldsymbol{u}}))^T \boldsymbol{D} (\tilde{\boldsymbol{\nabla}} \underline{\boldsymbol{N}} \bar{\boldsymbol{u}} - \boldsymbol{\alpha} \Delta T) \, d\Omega = \\ \int_{\partial\Omega} (\underline{\boldsymbol{N}} \delta \bar{\boldsymbol{u}})^T \boldsymbol{t} \, d(\partial\Omega) + \int_\Omega (\underline{\boldsymbol{N}} \delta \bar{\boldsymbol{u}})^T \boldsymbol{b} \, d\Omega \end{aligned} \tag{2.54}$$

All terms are moved to the left hand side and the nodal virtual displacement vector may be moved outside the integrals, as it is independent on position.

$$\begin{aligned} \delta \bar{\boldsymbol{u}} \Bigg[\int_\Omega (\tilde{\boldsymbol{\nabla}} \underline{\boldsymbol{N}})^T \boldsymbol{D} \tilde{\boldsymbol{\nabla}} \underline{\boldsymbol{N}} \, d\Omega \bar{\boldsymbol{u}} - \int_\Omega (\tilde{\boldsymbol{\nabla}} \underline{\boldsymbol{N}})^T \boldsymbol{D} \boldsymbol{\alpha} \Delta T \, d\Omega - \\ \int_{\partial\Omega} \underline{\boldsymbol{N}}^T \boldsymbol{t} \, d(\partial\Omega) - \int_\Omega \underline{\boldsymbol{N}}^T \boldsymbol{b} \, d\Omega \Bigg] = \boldsymbol{0} \end{aligned} \tag{2.55}$$

As the virtual displacements are arbitrary leads to the equality

$$\begin{aligned} \int_\Omega \underline{\boldsymbol{B}}^T \boldsymbol{D} \underline{\boldsymbol{B}} \, d\Omega \bar{\boldsymbol{u}} - \int_\Omega \underline{\boldsymbol{B}}^T \boldsymbol{D} \boldsymbol{\alpha} \Delta T \, d\Omega - \\ \int_{\partial\Omega} \underline{\boldsymbol{N}}^T \boldsymbol{t} \, d(\partial\Omega) - \int_\Omega \underline{\boldsymbol{N}}^T \boldsymbol{b} \, d\Omega = \boldsymbol{0} \end{aligned} \tag{2.56}$$

Here the gradient of the shape functions, $\nabla \underline{\boldsymbol{N}}$ is denoted with $\underline{\boldsymbol{B}}$. To make the notation of the integrals simpler, some symbols are introduced, as seen below. $\boldsymbol{f}_l$ is called the boundary load vector, $\boldsymbol{f}_v$ the volume load vector, $\boldsymbol{f}_0$ the thermal load vector, $\boldsymbol{K}_{uu}$ the stiffness matrix. (2.56) may then be written as in (2.61).

$$\boldsymbol{f}_l \equiv \int_{\partial\Omega} \underline{\boldsymbol{N}}^T \boldsymbol{t} \; d(\partial\Omega) \tag{2.57}$$

$$\boldsymbol{f}_v \equiv \int_{\Omega} \underline{\boldsymbol{N}}^T \boldsymbol{b} \; d\Omega \tag{2.58}$$

$$\boldsymbol{f}_0 \equiv \int_{\Omega} \underline{\boldsymbol{B}}^T \boldsymbol{D}\boldsymbol{\alpha}\Delta T \; d\Omega \tag{2.59}$$

$$\boldsymbol{K}_{uu} \equiv \int_{\Omega} \underline{\boldsymbol{B}}^T \boldsymbol{D} \underline{\boldsymbol{B}} \; d\Omega \tag{2.60}$$

$$\boldsymbol{K}_{uu}\bar{\boldsymbol{u}} = \boldsymbol{f}_l + \boldsymbol{f}_v + \boldsymbol{f}_0 \tag{2.61}$$

The stiffness matrix $\boldsymbol{K}_{uu}$ is assembled in the same fashion as the thermal stiffness matrix, as seen in (2.34). (2.61) is the Finite Element formulation for the elasticity problem. As no unknown quantity is time dependent, there is no need to discretize in time. In residual format, the equation is written as

$$\boldsymbol{R}_u(\bar{\boldsymbol{u}}) \equiv \boldsymbol{K}_{uu}\bar{\boldsymbol{u}} - \boldsymbol{f}_l - \boldsymbol{f}_v - \boldsymbol{f}_0 = \boldsymbol{0} \tag{2.62}$$

Together with the boundary conditions, the final formulation of the discretized mechanical problem is as below

$$\begin{cases} \boldsymbol{R}_u(\bar{\boldsymbol{u}}) = 0 \\ \quad \boldsymbol{u}_i = \boldsymbol{g}(\boldsymbol{x}_i) \quad \forall \boldsymbol{x}_i \in \partial\Omega_u, \\ \boldsymbol{t}(\boldsymbol{x}) = \boldsymbol{t}'(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \partial\Omega_t, \\ \boldsymbol{t}(\boldsymbol{x}) = k'(\boldsymbol{x})\boldsymbol{u}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \partial\Omega_s, \end{cases} \tag{2.63}$$

2.3 Coupled thermo-mechanical system

Until now, the thermal system and the mechanical have been studied independently of each other. In the mechanical problem, the temperature field was assumed to be known. Now, an unknown temperature field is instead considered. The field is set to be time dependent and is retrieved as the solution of the thermal problem (2.35).

This solution would then be the input as temperature change in the mechanical problem (2.46). In the coupled system, Ω refers to the same body for both weak formulations, (2.9) and (2.48). However, the boundary conditions in the thermal problem and the mechanical problem are independent of each other.

The thermal problem is approximated with the FE formulation as derived in section 2.1.2 and presented in (2.18). For the mechanical problem, the result is a bit different. As the temperature is discretized into $N+1$ time steps, leads to the existence of $N+1$ nodal displacement vectors $\bar{\boldsymbol{u}}^{(n)}, n = 0, 1, \ldots, N$. In (2.56) the temperature difference is approximated by

$$\Delta T = \boldsymbol{N}(\boldsymbol{T}^{(n)} - \boldsymbol{T}_R) \tag{2.64}$$

for each time step $n = 0, 1, \ldots, N$. This implies that there are $N+1$ solutions to (2.56), just as mentioned above. The vector $\boldsymbol{T}_R$ consists of the reference temperature in the nodes. They are assumed to all be 0 to simplify the following expressions. All this results in (2.62) instead being written as

$$\boldsymbol{R}_u(\bar{\boldsymbol{u}}^{(n)}, \boldsymbol{T}^{(n)}) = \boldsymbol{K}_{uu}\bar{\boldsymbol{u}}^{(n)} - \boldsymbol{f}_l - \boldsymbol{f}_v - \boldsymbol{K}_{uT}\boldsymbol{T}^{(n)} = \boldsymbol{0}, \tag{2.65}$$

where

$$\boldsymbol{K}_{uT} = \int_\Omega \underline{\boldsymbol{B}}\boldsymbol{D}\boldsymbol{\alpha}\boldsymbol{N}\ d\Omega. \tag{2.66}$$

This thermo-mechanical stiffness matrix $\boldsymbol{K}_{uT}$ is assembled just like the other stiffness matrices.

Since the displacements depend on the thermal problem, but the temperature does not depend on the mechanical problem, the problem described here is said to be one-way coupled. The full coupled system can now be written as

$$\begin{cases} \boldsymbol{R}_T^{(n)}(\boldsymbol{T}^{(n)}, \boldsymbol{T}^{(n-1)}) = \boldsymbol{0} \quad n = 1, 2, \ldots, N \\ \boldsymbol{R}_u(\bar{\boldsymbol{u}}^{(n)}, \boldsymbol{T}^{(n)}) = \boldsymbol{0} \quad n = 0, 1, \ldots, N \\ T_i^{(n+1)} = T'(\boldsymbol{x}_i, t_n) \quad \forall \boldsymbol{x}_i \in \partial\Omega_T^{(n)},\ n = 1, 2, \ldots, N \\ q_n(\boldsymbol{x}, t) = q_n'(\boldsymbol{x}, t) \quad \forall \boldsymbol{x} \in \partial\Omega_q^{(n)} \\ T_i^{(0)} = T_0(\boldsymbol{x}_i) \quad i = 1, 2, \ldots, nn \\ \boldsymbol{u}_i = \boldsymbol{u}'(\boldsymbol{x}_i) \quad \forall \boldsymbol{x}_i \in \partial\Omega_u, \\ \boldsymbol{t}(\boldsymbol{x}) = \boldsymbol{t}'(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \partial\Omega_t, \\ \boldsymbol{t}(\boldsymbol{x}) = k'(\boldsymbol{x})\boldsymbol{u}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \partial\Omega_s, \end{cases} \tag{2.67}$$

2.4 Density based topology optimization

Density based topology optimization is one of several methods used to solve the material layout design problem presented in the introduction. Other commonly used methods are the *level-set method* and *evolutionary structural optimization*. As the name suggest, the density based method introduces a density field on the design domain to determine where material should appear and disappear. As the Finite Element Method is heavily incorporated into topology optimization methods, the densities are defined on the finite elements. Generically speaking, the densities are design parameters to the topology optimization problem, as they are the tuning properties to finding the optimal design. With one design parameter per finite element, denoted ϕ_e for element e, the whole set of design parameters can be collected in a vector, as follows

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_1 & \phi_2 & \dots & \phi_{ne} \end{bmatrix}^T, \tag{2.68}$$

where ne is the number of elements. This vector parametrizes the design domain and take any value in the design space Φ which will be defined in a following section. The role of the design parameters are to determine the material to use in a certain element for the optimal design.

To distribute more than two materials, it is common to introduce multiple design parameters per element (this depends on the chosen material interpolation model). In the case of three materials, two design parameters per element are introduced, ϕ_I and ϕ_{II}. ϕ_I determines whether an element consists of material 1 or the other materials. ϕ_{II} then determines whether an element consists of material 2 or material 3.

As the design parameters determine the material distribution, it governs the material properties in the elements. For a thermo-mechanical problem, the material properties in element e are functions of the design parameter over the same element, ϕ_e, accordingly

$$E = E(\phi_e), \quad \kappa = \kappa(\phi_e), \quad s = \rho \cdot c_p = s(\phi_e), \quad r = E \cdot \alpha = r(\phi_e) \tag{2.69}$$

The density and heat capacity, and Young's modulus and CTE are gathered to a single property to simplify the formulation. The product of the density and heat capacity, here denoted s, is called the volumetric heat capacity. The product of Young's modulus and the coefficient of thermal expansion, here denoted r, is called the thermal stress coefficient. As these material properties now are considered functions of the design parameters, leads to the residual equations to also be functions of the design parameters

$$\boldsymbol{R}_T^{(n)}(\boldsymbol{\phi}, \boldsymbol{T}^{(n)}, \boldsymbol{T}^{(n-1)}) = \boldsymbol{A}(\boldsymbol{\phi})\,\boldsymbol{T}^{(n)} - \boldsymbol{B}(\boldsymbol{\phi})\,\boldsymbol{T}^{(n-1)} - \bar{\boldsymbol{f}}^{(n)} \tag{2.70}$$

$$\boldsymbol{R}_u(\boldsymbol{\phi}, \boldsymbol{T}^{(n)}, \bar{\boldsymbol{u}}^{(n)}) = \boldsymbol{K}_{uu}(\boldsymbol{\phi})\bar{\boldsymbol{u}}^{(n)} - \boldsymbol{K}_{uT}(\boldsymbol{\phi})\,\boldsymbol{T}^{(n)} - \boldsymbol{f}_l - \boldsymbol{f}_v \tag{2.71}$$

Next in the optimization process is to formulate what quantity to optimize and under which conditions.

2.4.1 Mathematical representation

No matter which topology optimization problem considered, there must be a measure to represent the performance of a design. This measure is stated so the problem is to minimize the measure. When for example optimizing for stiffness, the measure is compliance which should be minimized. For the optimization of a compliant mechanism, however, the goal is to maximize the output at a certain point, which is measured as the negative displacement of that point, as minimizing the negative displacement actually maximizes the displacement. When subject to minimization, the measure is commonly referred to as an objective function, in this report denoted g_0.

Alongside the objective function are the constraints of the problem. Examples of constraints are that the total mass of the design must not surpass a certain limit. An optimization problem may be subject to several constraints, here denoted $g_i \leq 0,\ i = 1, 2, \ldots, M$ where M is the total number of constraints. The objective function and constraint functions depend in general on the design parameters and the state variables. As the state equations are also dependent on the design parameters, they must be included to implicitly couple the design parameter to the state variables.

For a general transient thermo-mechanical topology optimization problem $\mathcal{P}$, where the state variables are discretized as in section 2.3, the mathematical representation is as follows.

$$\mathcal{P}: \begin{cases} \min\limits_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \; g_0(\boldsymbol{\phi}; \boldsymbol{T}^{(0)}, \boldsymbol{T}^{(1)}, \ldots, \boldsymbol{T}^{(N)}; \bar{\boldsymbol{u}}^{(0)}, \bar{\boldsymbol{u}}^{(1)}, \ldots, \bar{\boldsymbol{u}}^{(N)}) \\ \text{subject to} \quad \begin{cases} \boldsymbol{R}_T^{(n)}(\boldsymbol{\phi}; \boldsymbol{T}^{(n)}, \boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u(\boldsymbol{\phi}; \bar{\boldsymbol{u}}^{(n)}, \boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \\ g_1(\boldsymbol{\phi}; \boldsymbol{T}^{(0)}, \boldsymbol{T}^{(1)}, \ldots, \boldsymbol{T}^{(N)}; \bar{\boldsymbol{u}}^{(0)}, \bar{\boldsymbol{u}}^{(1)}, \ldots, \bar{\boldsymbol{u}}^{(N)}) \leq 0 \\ \vdots \\ g_M(\boldsymbol{\phi}; \boldsymbol{T}^{(0)}, \boldsymbol{T}^{(1)}, \ldots, \boldsymbol{T}^{(N)}; \bar{\boldsymbol{u}}^{(0)}, \bar{\boldsymbol{u}}^{(1)}, \ldots, \bar{\boldsymbol{u}}^{(N)}) \leq 0 \end{cases} \end{cases} \tag{2.72}$$

Design space

For a material design layout problem with two materials, the design parameters are usually defined in such a way that a value of $\phi_e = 0$ should be interpreted as only

material 1 in that element and $\phi_e = 1$ as only material 2 in that element. Allowing $\boldsymbol{\phi}$ to only take the values 0 or 1, i.e. discrete values, seems then to be a natural choice. However, if gradient based methods are to be used by the optimizer later in the process, the design space is required to be continuous. A solution is then to define the design space as 0 and 1 plus everything in between.

$$\Phi = \{ \ \boldsymbol{\phi} \ | \ \boldsymbol{0} \leq \boldsymbol{\phi} \leq \boldsymbol{1} \} \tag{2.73}$$

This type of relaxation of the problem (allowing more designs to be feasible) do however introduce other problems discussed below.

2.4.2 Material interpolation models

The first and obvious consequence of the relaxation of the topology optimization problem, is that a solution may consists of intermediate design values, i.e. values where the design parameter are in the gray area between 0 and 1. The cumbersome question is: how should a design with intermediate values be interpreted, as a material with intermediate properties normally can not be manufactured? At the same time the question arises, how should an intermediate design value affect the material properties? The answers lies primarily in the choice of material interpolation model.

A *material interpolation model* decides how a material property should be interpolated in the interval of intermediate design values. Two of the most used interpolation model are *Solid Isotropic Material with Penalization* (SIMP), first suggested in 1989 [15] and *Rational Approximation of Material Properties* (RAMP), proposed in 2001 [16]. The idea behind these interpolation models is to penalize intermediate designs. Together with a resource constraint on the density (e.g. a mass constraint), it will drive the optimizer to prefer values of either 0 or 1. For the Young's modulus, the SIMP interpolation model is

$$E(\phi_e) = E_1 + \phi_e^p (E_2 - E_1), \tag{2.74}$$

where E_1 and E_2 are the Young's modulus of material 1 and 2, respectively. The parameter p in (2.74) is called the penalty factor, and is usually chosen such that $p \geq 3$ [17]. RAMP on the other hand interpolates with a rational expression

$$E(\phi_e) = E_1 + \frac{\phi_e}{1 + q(1 - \phi_e)} (E_2 - E_1) \tag{2.75}$$

Both interpolation models can be seen in figure 2.4. For multi-material optimization, the material interpolation models are represented by functions with several design parameters as input. Using SIMP or RAMP for optimization with 3 materials (and 2 design parameters per element), the interpolation model is just applied twice to

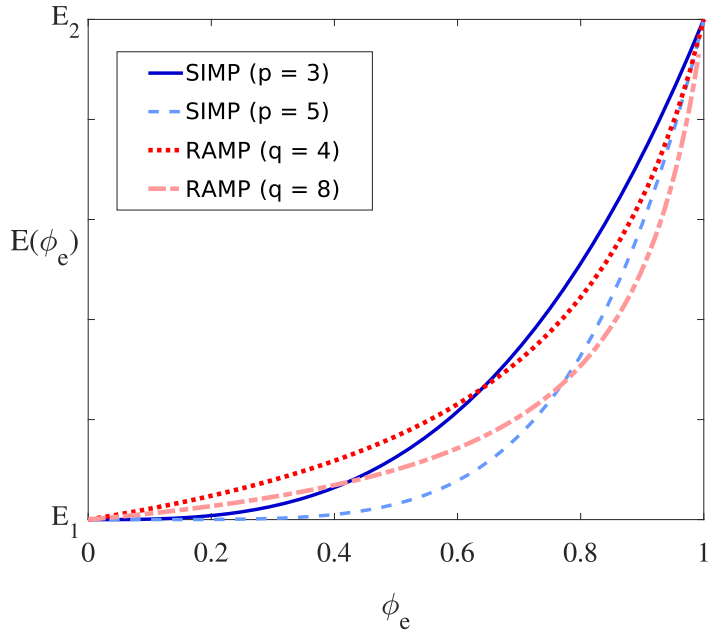


Figure 2.4: Example of SIMP and RAMP interpolation for different choices of parameters.

calculate the material properties. For the case of Young's modulus and SIMP, it would look like

$$E(\phi_{e,I}, \phi_{e,II}) = E_1 + \phi_{e,I}^{p_I} \left[E_2 + \phi_{e,II}^{p_{II}} (E_3 - E_2) - E_1 \right] \tag{2.76}$$

With this kind of interpolation, the first design parameter, ϕ_I, has precedence over ϕ_{II} for determining the material. The other material properties are interpolated in the same way, by replacing E in the equations above with the material property of interest.

2.4.3 Regularization

It is a well known fact for density based topology optimization that SIMP introduces numerical instabilities such as checkerboard designs. A popular solution to this problem is to regularize the problem via filtering techniques, which also sets a length scale for the smallest features of the optimal design.

The filtering techniques employed in this report is extensively described in the literature, for example by Bruns & Tortorelli [18], and is called density filtering. The main idea of the filtering is to introduce a filtered design parameter field $\widehat{\phi}$ that is the weighted unfiltered design ϕ in the surrounding elements. The interpretation, is that $\tilde{\phi}$ represents the physical design and ϕ can instead be seen as the raw input

(a) Unfiltered design with checkerboard pattern (b) Density based filtered design (c) Filtered and projected design

Figure 2.5: Showing the effect of the filtering technique and projection on a simple linear design

parameter to the optimization without any physical meaning. A general description of the density filtering technique follows.

$$\widehat{\phi_e} = \frac{\sum_{i=1}^{ne} w_e(\boldsymbol{x}_i)\phi_i v_i}{\sum_{i=1}^{ne} w_e(\boldsymbol{x}_i)v_i}, \tag{2.77}$$

where v_i denotes the volume of element i and the position $\boldsymbol{x_i}$ is normally taken to be the geometric center of element i. In matrix notation, the weighting can be written as

$$\widehat{\boldsymbol{\phi}} = \boldsymbol{W}\boldsymbol{\phi}, \quad W_{ij} = \frac{w_i(\boldsymbol{x}_j)v_j}{\sum_{k=1}^{ne} w_i(\boldsymbol{x}_k)v_k} \tag{2.78}$$

It can be seen in figure 2.5b how the filtering affects the design. The weighting normally only takes the closest elements into consideration. A simple and common weighting function is a linearly decaying radial function.

$$w_e(\boldsymbol{x}) = \begin{cases} R - ||\boldsymbol{x} - \boldsymbol{x}_e|| & \text{if } ||\boldsymbol{x} - \boldsymbol{x}_e|| < R \\ 0 & \text{otherwise} \end{cases} \tag{2.79}$$

R here denotes the weighting radius.

The filtering techniques do however unavoidably re-introduce intermediate designs as seen in figure 2.5b.

2.4.4 Projection

Even if the filtering solves the checkboard pattern problem and introduces a mesh-independent length scale on the design, it also reintroduces the problem with the intermediate design values. A solution which produces more "black and white"

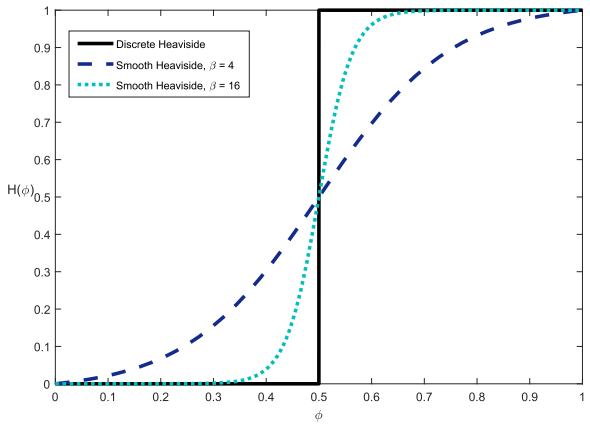


Figure 2.6: Heaviside projections with the threshold at 0.5. Notice how the smooth approximation of the Heaviside function approaches the discrete function when β is increased.

designs and pertains the length scale is Heaviside projection together with a robust formulation of the topology optimization problem.

The principle of Heaviside projection is to define a threshold between 0 and 1, where all the intermediate design parameters below the threshold are projected to 0 and all the ones above the threshold are projected to 1. A Heaviside projection function $H(\phi_e)$ according to this principle is seen as a solid line in figure 2.6. This function is however not continuous, which is necessary for the gradient based optimizers. A smooth approximation of the Heaviside projection function can be created as

$$H(\phi_e) = \frac{\tanh(\beta\eta) + \tanh(\beta(\phi_e - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1 - \eta))} \tag{2.80}$$

The approximated function has two parameters: η, which sets the threshold and β, which controls the steepness of the approximation as seen in figure 2.6. The steepness parameter β is usually set low in the beginning of the optimization to avoid making the system too sensitive, and gradually increased throughout the optimization. The Heaviside projection is preferably used together with some kind of regularization technique.

Robust formulation

Together with the Heaviside projection, a robust formulation of any mass constrained topology optimization problem can be defined. A robust formulation creates a new problem to which the solution should be more robust to errors in the manufacturing process. This technique has the benefit of preserving the minimal length scale of the optimal design, which is otherwise lost when applying the Heaviside

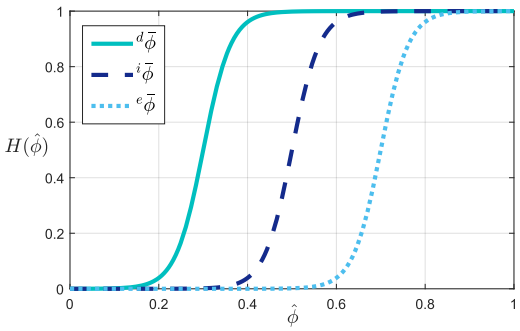


Figure 2.7: The different projections for $\Delta\eta = 0.2$ and $\beta = 16$

Figure 2.8: Example of a projected eroded, intermediate and dilated design in the order from left to right.

projection to the design and also the benefit of avoiding one-node hinges. [19]

The steps of a robust formulation are outlined in [19]. First, three different projected designs are defined, one eroded, one intermediate and one dilated.

$$^e\overline{\phi} = H(\widehat{\phi}, \eta = 0.5 - \Delta\eta) \tag{2.81}$$

$$^i\overline{\phi} = H(\widehat{\phi}, \eta = 0.5) \tag{2.82}$$

$$^d\overline{\phi} = H(\widehat{\phi}, \eta = 0.5 + \Delta\eta) \tag{2.83}$$

$\Delta\eta$ decide the margin of the threshold. In figure 2.7 and 2.8 examples of the different projections functions and resulting designs can be seen. Moreover, a new objective function is defined as the maximum of the objective function with the three different designs as input.

$$\begin{aligned}\bar{g}_0 \equiv \max \Big(& g_0(^e\bar{\boldsymbol{\phi}}; {}^e\boldsymbol{T}^{(0)}, \ldots, {}^e\boldsymbol{T}^{(N)}; {}^e\bar{\boldsymbol{u}}^{(0)}, \ldots, {}^e\bar{\boldsymbol{u}}^{(N)}, \\ & g_0(^i\bar{\boldsymbol{\phi}}; {}^i\boldsymbol{T}^{(0)}, \ldots, {}^i\boldsymbol{T}^{(N)}; {}^i\bar{\boldsymbol{u}}^{(0)}, \ldots, {}^i\bar{\boldsymbol{u}}^{(N)}, \\ & g_0(^d\bar{\boldsymbol{\phi}}; {}^d\boldsymbol{T}^{(0)}, \ldots, {}^d\boldsymbol{T}^{(N)}; {}^d\bar{\boldsymbol{u}}^{(0)}, \ldots, {}^d\bar{\boldsymbol{u}}^{(N)}) \Big)\end{aligned} \tag{2.84}$$

The left-sided superscript of the state variables (e.g. $^e\boldsymbol{T}^{(0)}$) denotes that they are the solution to the corresponding state equation with either the eroded, intermediate

or dilated design as input. There are therefore three sets of state equations to solve for the robust formulation:

$$\begin{cases} \boldsymbol{R}_T^{(n)}({}^e\overline{\boldsymbol{\phi}}; {}^e\boldsymbol{T}^{(n)}, {}^e\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^e\overline{\boldsymbol{\phi}}; {}^e\boldsymbol{T}^{(n)}, {}^e\bar{\boldsymbol{u}}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \tag{2.85}$$

$$\begin{cases} \boldsymbol{R}_T^{(n)}({}^i\overline{\boldsymbol{\phi}}; {}^i\boldsymbol{T}^{(n)}, {}^i\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^i\overline{\boldsymbol{\phi}}; {}^i\boldsymbol{T}^{(n)}, {}^i\bar{\boldsymbol{u}}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \tag{2.86}$$

$$\begin{cases} \boldsymbol{R}_T^{(n)}({}^d\overline{\boldsymbol{\phi}}; {}^d\boldsymbol{T}^{(n)}, {}^d\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^d\overline{\boldsymbol{\phi}}; {}^d\boldsymbol{T}^{(n)}, {}^d\bar{\boldsymbol{u}}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \tag{2.87}$$

The intermediate design is taken as the design to actually manufacture and use. The eroded and dilated designs are interpreted as designs with production errors. The maximum function of the robust formulation makes sure these designs have a considerably good performance, sort of acting as a safe margin for manufacturing errors. This also prevents non-manufacturable one-node hinges to appear in the optimal design.

It is assumed that any problem under the robust formulation has a mass constraint. Without a robust formulation, a simple mass constraint for a design would look like

$$g_m(\boldsymbol{\phi}) = \sum_{i=1}^{nel} \rho(\phi_i) v_i - m^* \leq 0 \tag{2.88}$$

The mass limit m^* limits how much material can be used in the optimal design. For a robust formulation, the input to the mass constraint is the worst projected design, which in this thesis will be the dilated design for the material interpolation model chosen on the density. However, the original constraint should hold for the final design, which for the robust formulation is the intermediate design. To make sure the mass constraint holds for the intermediate design, a dilated mass limit is used, m_d^*.

$$g_m({}^d\overline{\boldsymbol{\phi}}) = \sum_{i=1}^{nel} \rho({}^d\overline{\phi}_i) v_i - m_d^* \leq 0 \tag{2.89}$$

The dilated mass limit is then updated regularly over the course of the optimization, to make sure the constraint holds for the intermediate design.

$$m_d^* = \frac{\sum \rho({}^d\overline{\phi}_i) v_i}{\sum \rho({}^i\overline{\phi}_i) v_i} m^* \tag{2.90}$$

The general problem $\mathcal{P}$ stated as in (2.72) is with a robust formulation instead written as

$$\mathcal{P}_r : \begin{cases} \min\limits_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \ \bar{g}_0 \\ \text{s.t.} \quad \begin{cases} \begin{cases} \boldsymbol{R}_T^{(n)}({}^e\overline{\boldsymbol{\phi}}; {}^e\boldsymbol{T}^{(n)}, {}^e\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^e\overline{\boldsymbol{\phi}}; {}^e\bar{\boldsymbol{u}}^{(n)}, {}^e\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \\ \begin{cases} \boldsymbol{R}_T^{(n)}({}^i\overline{\boldsymbol{\phi}}; {}^i\boldsymbol{T}^{(n)}, {}^i\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^i\overline{\boldsymbol{\phi}}; {}^i\bar{\boldsymbol{u}}^{(n)}, {}^i\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \\ \begin{cases} \boldsymbol{R}_T^{(n)}({}^d\overline{\boldsymbol{\phi}}; {}^d\boldsymbol{T}^{(n)}, {}^d\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^d\overline{\boldsymbol{\phi}}; {}^d\bar{\boldsymbol{u}}^{(n)}, {}^d\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \\ g_m({}^d\overline{\boldsymbol{\phi}}) \leq 0 \\ g_1({}^i\overline{\boldsymbol{\phi}}; {}^i\boldsymbol{T}^{(0)}, {}^i\boldsymbol{T}^{(1)}, \ldots, {}^i\boldsymbol{T}^{(N)}; {}^i\bar{\boldsymbol{u}}^{(0)}, {}^i\bar{\boldsymbol{u}}^{(1)}, \ldots, {}^i\bar{\boldsymbol{u}}^{(N)}) \leq 0 \\ \vdots \\ g_M({}^i\overline{\boldsymbol{\phi}}; {}^i\boldsymbol{T}^{(0)}, {}^i\boldsymbol{T}^{(1)}, \ldots, {}^i\boldsymbol{T}^{(N)}; {}^i\bar{\boldsymbol{u}}^{(0)}, {}^i\bar{\boldsymbol{u}}^{(1)}, \ldots, {}^i\bar{\boldsymbol{u}}^{(N)}) \leq 0 \end{cases} \end{cases} \tag{2.91}$$

2.4.5 Optimizers

With the topology optimization problems formulated as mathematical optimization problems such as $\mathcal{P}$ or $\mathcal{P}_r$, the problem of finding a material distribution design has turned into minimizing a mathematical function under certain constraints. Traditional mathematical optimizers may be employed to solve these problems. However, since solving the state equations to evaluate the objective function for a certain set of design parameters is very expensive from a computational point of view, special optimizers are used in topology optimization. The common ground for these optimizers is that they approximate the non-linear, non-convex problem $\mathcal{P}$ with a convex, separable problem $\mathcal{P}'$, which can be exactly solved. Iteratively approximating $\mathcal{P}$ at the current design and solving $\mathcal{P}'$ to find a new, better design will eventually converge into a local minimum using these methods. Another common feature of most optimizers used in topology optimization, is that they are gradient based, meaning that the gradient of the objective function and constraints must be given to approximate the problem. To fully approximate $\mathcal{P}$ around an arbitrary design $\boldsymbol{\phi}_k$ the following first order derivatives must be known:

$$\left.\frac{D g_i}{D \boldsymbol{\phi}}\right|_{\boldsymbol{\phi}=\boldsymbol{\phi}_k}, \quad \forall i = 1, 2, \ldots, M \tag{2.92}$$

One popular optimizer for topology optimization is the Method of Moving Asymptotes (MMA), which was introduced by Svanberg in 1987 [20]. For more details about MMA and optimizers, the reader is referred to [21].

2.4.6 Sensitivity analysis

In order to determine the derivatives in (2.92), a sensitivity analysis of the functions $g_i, i = 0, 1, \ldots, M$ with respect to $\phi_j, j = 1, 2, \ldots, nel$ is required. As g_i in general is a function of the design parameters but also the state variables, gives that the chain rule must be used to fully determine the derivative.

$$\frac{Dg_i}{D\phi_j} = \frac{\partial g_i}{\partial \phi_j} + \sum_{n=0}^{N} \frac{\partial g_i}{\partial \boldsymbol{T}^{(n)}} \frac{D\boldsymbol{T}^{(n)}}{D\phi_j} + \sum_{n=0}^{N} \frac{\partial g_i}{\partial \bar{\boldsymbol{u}}^{(n)}} \frac{D\bar{\boldsymbol{u}}^{(n)}}{D\phi_j} \tag{2.93}$$

The explicit derivatives $\frac{\partial g_i}{\partial \phi_j}$, $\frac{\partial g_i}{\partial \boldsymbol{T}}$ and $\frac{\partial g_i}{\partial \bar{\boldsymbol{u}}}$ are directly given from the expression of g_i. The implicit derivatives of the state variables, $\frac{D\boldsymbol{T}}{D\phi_j}$ and $\frac{D\bar{\boldsymbol{u}}}{D\phi_j}$, are not as easily found. They may be calculated by implicitly deriving the state equations, but this method, called the direct method, has to be done for every $\phi_j, j = 1, 2, \ldots, nel$ and is thus computationally expensive for any case with a high number of design parameters. A better suited method for the problems encountered in this thesis is the adjoint method. The following derivations closely follows those done in [22]. The first step in the adjoint method is to define an augmented version of the function g_i

$$G_i \equiv g_i - \sum_{n=1}^{N} (\boldsymbol{\lambda}_i^{(n)})^T \boldsymbol{R}_T^{(n)} - \sum_{n=0}^{N} (\boldsymbol{\gamma}_i^{(n)})^T \boldsymbol{R}_u \tag{2.94}$$

The variables $\boldsymbol{\lambda}_i$ and $\boldsymbol{\gamma}_i$ are dependent on $\boldsymbol{\phi}$ and are called the adjoint variables. They are arbitrary vectors with the same dimensions as $\boldsymbol{T}$ and $\bar{\boldsymbol{u}}$ respectively. The augmented function G_i equals g_i since the state equations always equals zero. Finding the sensitivity of G_i is thus equivalent to finding the sensitivity of g_i. Differentiating (2.94) with respect to ϕ_j result in

$$\frac{DG_i}{D\phi_j} = \frac{Dg_i}{D\phi_j} - \sum_{n=1}^{N} (\boldsymbol{\lambda}_i^{(n)})^T \frac{D\boldsymbol{R}_T^{(n)}}{D\phi_j} - \sum_{n=0}^{N} (\boldsymbol{\gamma}_i^{(n)})^T \frac{D\boldsymbol{R}_u}{D\phi_j} \tag{2.95}$$

Notice how the derivatives of the adjoint variables $\frac{D\boldsymbol{\lambda}}{D\phi_j}$ and $\frac{D\boldsymbol{\gamma}}{D\phi_j}$ disappear, since the state equations equal zero, as mentioned before. The expansion of $\frac{Dg_i}{D\phi_j}$ is already given in (2.93). The expansion of $\frac{D\boldsymbol{R}_T}{D\phi_j}$ and $\frac{D\boldsymbol{R}_u}{D\phi_j}$ are similarly expanded as

$$\frac{D\boldsymbol{R}_T^{(n)}}{D\phi_j} = \frac{\partial \boldsymbol{R}_T^{(n)}}{\partial \phi_j} + \frac{\partial \boldsymbol{R}_T^{(n)}}{\partial \boldsymbol{T}^{(n)}} \frac{D\boldsymbol{T}^{(n)}}{D\phi_j} + \frac{\partial \boldsymbol{R}_T^{(n)}}{\partial \boldsymbol{T}^{(n-1)}} \frac{D\boldsymbol{T}^{(n-1)}}{D\phi_j} \tag{2.96}$$

and

$$\frac{D\boldsymbol{R}_u}{D\phi_j} = \frac{\partial \boldsymbol{R}_u}{\partial \phi_j} + \frac{D\boldsymbol{R}_u}{\partial \bar{\boldsymbol{u}}^{(n)}}\frac{D\bar{\boldsymbol{u}}^{(n)}}{D\phi_j} + \frac{D\boldsymbol{R}_u}{\partial \boldsymbol{T}^{(n)}}\frac{D\boldsymbol{T}^{(n)}}{D\phi_j} \tag{2.97}$$

Inserting (2.96), (2.97) and (2.93) into (2.95) yields the lengthy expression

$$\begin{aligned}\frac{DG_i}{D\phi_j} = \frac{\partial g_i}{\partial \phi_j} + \sum_{n=0}^{N}\left(\frac{\partial g_i}{\partial \boldsymbol{T}^{(n)}}\right)^T \frac{D\boldsymbol{T}^{(n)}}{D\phi_j} + \sum_{n=0}^{N}\left(\frac{\partial g_i}{\partial \bar{\boldsymbol{u}}^{(n)}}\right)^T \frac{D\bar{\boldsymbol{u}}^{(n)}}{D\phi_j} - \\ \sum_{n=1}^{N}(\boldsymbol{\lambda}_i^{(n)})^T\left(\frac{\partial \boldsymbol{R}_T^{(n)}}{\partial \phi_j} + \frac{\partial \boldsymbol{R}_T^{(n)}}{\partial \boldsymbol{T}^{(n)}}\frac{D\boldsymbol{T}^{(n)}}{D\phi_j} + \frac{\partial \boldsymbol{R}_T^{(n)}}{\partial \boldsymbol{T}^{(n-1)}}\frac{D\boldsymbol{T}^{(n-1)}}{D\phi_j}\right) - \\ \sum_{n=0}^{N}(\boldsymbol{\gamma}_i^{(n)})^T\left(\frac{\partial \boldsymbol{R}_u}{\partial \phi_j} + \frac{\partial \boldsymbol{R}_u}{\partial \bar{\boldsymbol{u}}^{(n)}}\frac{D\bar{\boldsymbol{u}}^{(n)}}{D\phi_j} + \frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(n)}}\frac{D\boldsymbol{T}^{(n)}}{D\phi_j}\right)\end{aligned} \tag{2.98}$$

This expression still has the implicit derivatives of the state variables present, just as with (2.93). However, if the adjoint variables $\boldsymbol{\lambda}_i$ and $\boldsymbol{\gamma}_i$ are chosen in a specific way, the implicit derivatives may be annihilated from the expression. To see how this is done, the expression in (2.98) is divided into an explicit part and an implicit part.

$$\frac{DG_i}{D\phi_j} = \underbrace{\frac{DG_{i,E}}{D\phi_j}}_{\text{Explicit part}} + \underbrace{\sum_{n=1}^{N}\left(\frac{DG_{i,I}}{D\phi_j}\right)_{\boldsymbol{T}}^{(n)} + \sum_{n=0}^{N}\left(\frac{DG_{i,I}}{D\phi_j}\right)_{\bar{\boldsymbol{u}}}^{(n)}}_{\text{Implicit part}} \tag{2.99}$$

The first term collects all the explicit derivatives. The sums $\sum_{n=1}^{N}\left(\frac{DG_{i,I}}{D\phi_j}\right)_{\boldsymbol{T}}^{(n)}$ and $\sum_{n=0}^{N}\left(\frac{DG_{i,I}}{D\phi_j}\right)_{\bar{\boldsymbol{u}}}^{(n)}$ collects all the terms with the implicit derivatives $\frac{D\boldsymbol{T}^{(n)}}{D\phi_j}, n = 1, 2, \ldots, N$ and $\frac{D\bar{\boldsymbol{u}}^{(n)}}{D\phi_j}, n = 0, 1, \ldots, N$, respectively. The explicit part is defined as

$$\begin{aligned}\frac{DG_{i,E}}{D\phi_j} \equiv \frac{\partial g_i}{\partial \phi_j} - \sum_{n=1}^{N}(\boldsymbol{\lambda}_i^{(n)})^T\frac{\partial \boldsymbol{R}_T^{(n)}}{\partial \phi_j} - \sum_{n=0}^{N}(\boldsymbol{\gamma}_i^{(n)})^T\frac{\partial \boldsymbol{R}_u}{\partial \phi_j} - \\ (\boldsymbol{\lambda}_i^{(1)})^T\frac{\partial \boldsymbol{R}_T^{(1)}}{\partial \boldsymbol{T}^{(0)}}\frac{D\boldsymbol{T}^{(0)}}{D\phi_j} - (\boldsymbol{\gamma}_i^{(0)})^T\frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(0)}}\frac{D\boldsymbol{T}^{(0)}}{D\phi_j}\end{aligned} \tag{2.100}$$

The implicit derivative $\frac{D\boldsymbol{T}^{(0)}}{D\phi_j}$ is included in the explicit part, as this derivative is known (most of the time it is $\boldsymbol{0}$ as the initial temperatures are most often set to be constant). The implicit part related to the temperature is a bit more difficult to express. Below follows the expressions for every term in $\sum_{n=1}^{N}\left(\frac{DG_{i,I}}{D\phi_j}\right)_{\boldsymbol{T}}^{(n)}$

$$
\left.
\begin{aligned}
\left(\frac{DG_{i,I}}{D\phi_j}\right)^{(1)}_{\boldsymbol{T}} &\equiv \left[\left(\frac{\partial g_i}{\partial \boldsymbol{T}^{(1)}}\right)^T - (\boldsymbol{\lambda}^{(1)})^T \frac{\partial \boldsymbol{R}_T^{(1)}}{\partial \boldsymbol{T}^{(1)}} - (\boldsymbol{\lambda}^{(2)})^T \frac{\partial \boldsymbol{R}_T^{(2)}}{\partial \boldsymbol{T}^{(1)}} - (\boldsymbol{\gamma}^{(1)})^T \frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(1)}}\right] \frac{D\boldsymbol{T}^{(1)}}{D\phi_j} \\
\left(\frac{DG_{i,I}}{D\phi_j}\right)^{(2)}_{\boldsymbol{T}} &\equiv \left[\left(\frac{\partial g_i}{\partial \boldsymbol{T}^{(2)}}\right)^T - (\boldsymbol{\lambda}^{(2)})^T \frac{\partial \boldsymbol{R}_T^{(2)}}{\partial \boldsymbol{T}^{(2)}} - (\boldsymbol{\lambda}^{(3)})^T \frac{\partial \boldsymbol{R}_T^{(3)}}{\partial \boldsymbol{T}^{(2)}} - (\boldsymbol{\gamma}^{(2)})^T \frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(2)}}\right] \frac{D\boldsymbol{T}^{(2)}}{D\phi_j} \\
&\vdots \\
\left(\frac{DG_{i,I}}{D\phi_j}\right)^{(N-1)}_{\boldsymbol{T}} &\equiv \left[\left(\frac{\partial g_i}{\partial \boldsymbol{T}^{(N-1)}}\right)^T - (\boldsymbol{\lambda}^{(N-1)})^T \frac{\partial \boldsymbol{R}_T^{(N-1)}}{\partial \boldsymbol{T}^{(N-1)}} - (\boldsymbol{\lambda}^{(N)})^T \frac{\partial \boldsymbol{R}_T^{(N)}}{\partial \boldsymbol{T}^{(N-1)}} - \right. \\
&\qquad \left. (\boldsymbol{\gamma}^{(N-1)})^T \frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(N-1)}}\right] \frac{D\boldsymbol{T}^{(N-1)}}{D\phi_j} \\
\left(\frac{DG_{i,I}}{D\phi_j}\right)^{(N)}_{\boldsymbol{T}} &\equiv \left[\left(\frac{\partial g_i}{\partial \boldsymbol{T}^{(N)}}\right)^T - (\boldsymbol{\lambda}^{(N)})^T \frac{\partial \boldsymbol{R}_T^{(N)}}{\partial \boldsymbol{T}^{(N)}} - (\boldsymbol{\gamma}^{(N)})^T \frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(N)}}\right] \frac{D\boldsymbol{T}^{(N)}}{D\phi_j}
\end{aligned}
\right\} \tag{2.101}
$$

For the implicit part related to the displacement, the expression is much simpler to write out

$$
\left(\frac{DG_{i,I}}{D\phi_j}\right)^{(n)}_{\bar{\boldsymbol{u}}} \equiv \left[\left(\frac{\partial g_i}{\partial \bar{\boldsymbol{u}}^{(n)}}\right)^T - (\boldsymbol{\gamma}^{(n)})^T \frac{\partial \boldsymbol{R}_u}{\partial \bar{\boldsymbol{u}}^{(n)}}\right] \frac{D\bar{\boldsymbol{u}}^{(n)}}{D\phi_j}, \quad \forall n = 0, 1, \ldots, N \tag{2.102}
$$

To annihilate the implicit derivatives, the expressions inside the square brackets in (2.101) and (2.102) must equal zero. This creates a system of equations to solve for $\boldsymbol{\lambda}$ and $\boldsymbol{\gamma}$, namely

$$
\left.
\begin{aligned}
\left(\frac{\partial \boldsymbol{R}_T^{(1)}}{\partial \boldsymbol{T}^{(1)}}\right)^T \boldsymbol{\lambda}_i^{(1)} &= \frac{\partial g_i}{\partial \boldsymbol{T}^{(1)}} + \left(\frac{\partial \boldsymbol{R}_T^{(2)}}{\partial \boldsymbol{T}^{(1)}}\right) \boldsymbol{\lambda}_i^{(2)} + \left(\frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(1)}}\right) \boldsymbol{\gamma}_i^{(1)} \\
\left(\frac{\partial \boldsymbol{R}_T^{(2)}}{\partial \boldsymbol{T}^{(2)}}\right)^T \boldsymbol{\lambda}_i^{(2)} &= \frac{\partial g_i}{\partial \boldsymbol{T}^{(2)}} + \left(\frac{\partial \boldsymbol{R}_T^{(3)}}{\partial \boldsymbol{T}^{(2)}}\right) \boldsymbol{\lambda}_i^{(3)} + \left(\frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(2)}}\right) \boldsymbol{\gamma}_i^{(2)} \\
&\vdots \\
\left(\frac{\partial \boldsymbol{R}_T^{(N-1)}}{\partial \boldsymbol{T}^{(N-1)}}\right)^T \boldsymbol{\lambda}_i^{(N-1)} &= \frac{\partial g_i}{\partial \boldsymbol{T}^{(N-1)}} + \left(\frac{\partial \boldsymbol{R}_T^{(N)}}{\partial \boldsymbol{T}^{(N-1)}}\right) \boldsymbol{\lambda}_i^{(N)} + \left(\frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(N-1)}}\right) \boldsymbol{\gamma}_i^{(N-1)} \\
\left(\frac{\partial \boldsymbol{R}_T^{(N)}}{\partial \boldsymbol{T}^{(N)}}\right)^T \boldsymbol{\lambda}_i^{(N)} &= \frac{\partial g_i}{\partial \boldsymbol{T}^{(N)}} + \left(\frac{\partial \boldsymbol{R}_u}{\partial \boldsymbol{T}^{(N)}}\right) \boldsymbol{\gamma}_i^{(N)}
\end{aligned}
\right\} \tag{2.103}
$$

and

$$\left(\frac{\partial \boldsymbol{R}_u}{\partial \bar{\boldsymbol{u}}^{(n)}}\right)^T \boldsymbol{\gamma}_i^{(n)} = \frac{\partial g_i}{\partial \bar{\boldsymbol{u}}^{(n)}} \tag{2.104}$$

These equations are the adjoint variants of the first order approximation of the residual equations 2.31 and 2.65, except for the quantities $\frac{\partial g_i}{\partial \boldsymbol{T}^{(n)}}$ and $\frac{\partial g_i}{\partial \bar{\boldsymbol{u}}^{(n)}}$. As these quantities have the same form as the loads in the original state equations, they are usually called adjoint loads. The adjoint loads are in general different for each function g_i. That is why the adjoint system consisting of (2.103) and (2.104) must be solved for each function $g_i, i = 0, 1, \ldots, M$.

The solution to the adjoint system, the adjoint variables $\boldsymbol{\lambda}$ and $\boldsymbol{\gamma}$, are then inserted into (2.100), which is also the final expression for the sensitivity of (2.99), as the implicit parts have been annihilated.

Lastly, if a density filter and Heaviside projection is used on the design parameters which are the input to $g_i = g_i(\overline{\boldsymbol{\phi}}; \ldots)$, the derivatives with respect to ϕ_j in (2.100) are replaced with derivatives with respect to $\overline{\phi}_j$. $\frac{Dg_i}{D\phi_j}$ is still the input to the optimizer and the chain rule is therefore applied

$$\frac{Dg_i}{D\phi_j} = \sum_{k=1}^{nel} \frac{Dg_i}{D\overline{\phi}_k} \frac{D\overline{\phi}_k}{D\widehat{\phi}_k} \frac{D\widehat{\phi}_k}{D\phi_j} \tag{2.105}$$

The derivative of the filtered design parameter $\frac{D\widehat{\phi}_k}{D\phi_j}$ can be derived from (2.77) which gives

$$\frac{D\widehat{\phi}_k}{D\phi_j} = W_{kj} \tag{2.106}$$

where W_{kj} is defined in (2.78). The derivative of the projected density parameter $\frac{D\overline{\phi}_k}{D\widehat{\phi}_k}$ is derived from (2.80), and gives

$$\frac{D\overline{\phi}_k}{D\widehat{\phi}_k} = \frac{DH(\widehat{\phi}_k)}{D\widehat{\phi}_k} = \frac{\beta\left[1 - \tanh^2(\beta(\widehat{\phi}_k - \eta))\right]}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))} \tag{2.107}$$

Chapter 3

Implementation

In the previous chapter, the theory to construct a topology optimization algorithm was covered. In this chapter, the practical details are covered instead, presenting the specific details on how the program used in this thesis operates. This includes an overview of the solution process, from defining a geometry to plotting the result; the pseudo-code for the topology optimization algorithm and the overall design of the code. The program is written in Matlab and is open-sourced (retrievable at [23]). The code is highly modular and provides a simple way to add new topology optimization problems and FEM models, for example electromagnetic or non-linear mechanical FEM.

3.1 Implementation overview

The process of constructing and solving a general transient linear coupled thermo-mechanical topology optimization problem (referred to as a *run*), can be outlined by a number of steps. The steps follows, and are described more in detail below.

1. Pre-process:
   (a) Define the discretized geometry and time domain.
   (b) Select the FEM model(s) to use.
   (c) Set boundary conditions on the FEM model.
2. Define materials and set material interpolation models used in the FEM model.
3. Instantiate the TO problem with the FEM model as input.
4. Set an initial design.
5. Run the optimization algorithm and receive the optimal design.

6. Post-process:
    (a) Plot the optimal design.
    (b) Plot the state variables for the optimal design.

A run begins with a pre-processing step, which involves the definition of the geometry and the TO FEM model or models used. The boundary conditions are also defined in this step. Next, the materials to use and the material interpolation models are defined and inputted to the TO FEM model. The TO FEM model is input to a predefined topology optimization problem. The number of materials (single or multi-material) determines how many design parameters the optimization problem will have. The initial design to the problem is then set and inputted to the optimization algorithm. This algorithm is where the majority of the computational time is spent when performing a run, and is therefore outlined with algorithm 1. The termination criteria for the while-loop in the algorithm is generic and can for example be a limit on the number of optimization iteration (e.g. $k < 100$) or a limit on the iteration step (e.g. $||\boldsymbol{\phi}^{(k)} - \boldsymbol{\phi}^{(k-1)}|| < 10^{-6}$). The outputted design from the optimization is then plotted to show how the design looks like. The state equations are also solved for the final design and the state variables plotted.

3.2 Design of the modular Topology optimization solver

The implementation overview in the previous section describes how a single run is performed. However, several different topology optimization problems, FEM models, different number of materials and perhaps also different optimizers should be studied. In addition, a great control of different parameters in the process is needed to study the influence of these parameters. These criteria may be fulfilled with a modular codebase which also allows for easy control of the whole topology optimization process. What is described in this section is the codebase written alongside this master's thesis. The criteria to keep the codebase modular and the topology optimization controllable have constantly been incorporated into the code.

The entire TO solver is programmed in Matlab. This choice of programming language is motivated by its mathematical features, optimized routines, object oriented features and ease of use for the developer. For the optimization, a Matlab interface to a library called *NLopt* [24] has been used. The library offer several different gradient-based algorithms, such as the Method of Moving Asymptotes (MMA) and sequential quadratic programming (SQP).

The unstructured geometry input supported are `.GMSH`-files, as this format is open-sourced and lightweight. The output format are VTK-files, which is a lightweight and widely recognized format. The freely available program Paraview is used to post-process the results.

Algorithm 1: Topology optimization algorithm

Input: Initial design $\boldsymbol{\phi}^{\{0\}}$
Output: Final design $\boldsymbol{\phi}^{\{K\}}$

$k := 0$;
Set $\boldsymbol{\phi}^{\{k\}} := \boldsymbol{\phi}^{\{0\}}$;
if *filtering activated* **then**
  Calculate weight matrix $\boldsymbol{W}$;
else
  Set $\boldsymbol{W} := \boldsymbol{I}$
end
if *projection activated* **then**
  Set the Heaviside approximate function H;
else
  Set $H(x) = x$;
end
while ***not*** *termination criteria* **do**
  $\widehat{\boldsymbol{\phi}}^{\{k\}} := \boldsymbol{W}\boldsymbol{\phi}^{\{k\}}$;
  $\overline{\boldsymbol{\phi}}^{\{k\}} := H(\widehat{\boldsymbol{\phi}}^{\{k\}})$;
  for *timestep* $n \leftarrow 0$ **to** N **do**
    Solve the state problem $\boldsymbol{R}_T^{(n)}(\overline{\boldsymbol{\phi}}^{\{k\}})$;
    Set the temperature load $\boldsymbol{f}_0^{(n)} = \boldsymbol{K}_{uT}\boldsymbol{T}^{(n)}$;
    Solve the state problem $\boldsymbol{R}_u(\overline{\boldsymbol{\phi}}^{\{k\}})$;
  end
  Calculate the function values $g_0, g_1, \ldots, g_M$;
  for *timestep* $n \leftarrow N$ **to** 0 **do**
    Solve $\boldsymbol{\gamma}^{(n)}$ in the adjoint problem (2.104);
    Set the adjoint load from $\boldsymbol{\gamma}^{(n)}$;
    Solve $\boldsymbol{\lambda}^{(n)}$ in the adjoint problem (2.103);
  end
  Calculate the sensitivities $\frac{Dg_0}{D\boldsymbol{\phi}}, \frac{Dg_1}{D\boldsymbol{\phi}}, \ldots, \frac{Dg_M}{D\boldsymbol{\phi}}$;
  Insert the values and sensitivities to an optimizer to find $\boldsymbol{\phi}^{\{k+1\}}$;
  $k := k + 1$;
end
Set $\boldsymbol{\phi}^{\{K\}} := \boldsymbol{\phi}^{\{k\}}$

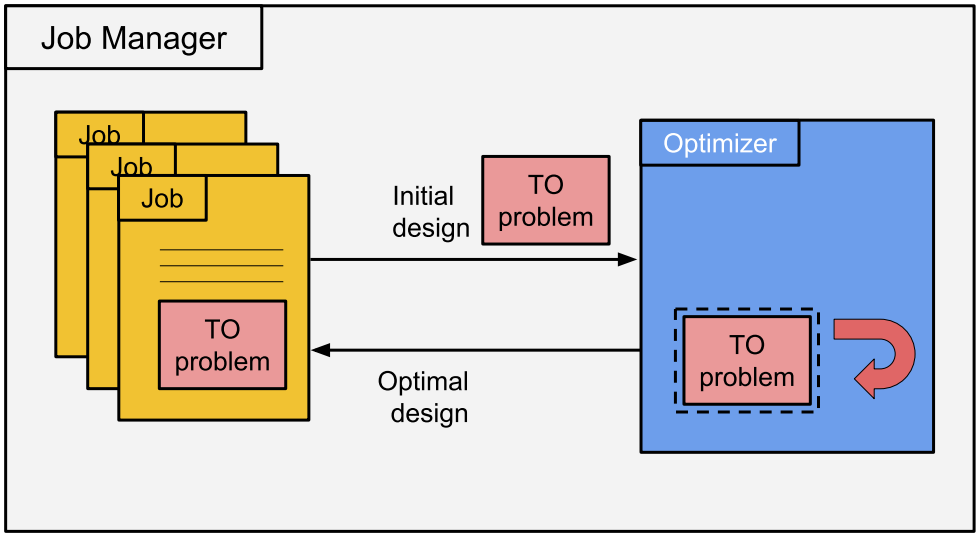


Figure 3.1: Schematics showing the principles of the job manager.

Job Manager

To performs several runs in a systematic way, the runs are described in so called *Jobs* and stored by a *Job Manager*. When a job is run by the job manager, the TO problem and initial design declared for that job are sent to the optimizer, which performs non-linear mathematical optimization on the TO problem. The optimizer therefore calls the functions of the TO problem each iteration to find the next design. When the termination criteria are met, the optimizer stops and outputs the optimal design back to the job to store. The results of a job may be plotted directly in Matlab, or exported to a VTK-file. A schematic view of a job manager is shown in figure 3.1.

TO Problem

The topology optimization (TO) problem described by the jobs are pre-defined and must be a subclass to an abstract class called `TopOpt`. A TO problem encapsulates the problem functions $g_0, g_1, \ldots, g_M$ and their first order derivatives $\frac{Dg_0}{D\boldsymbol{\phi}}, \frac{1g_0}{D\boldsymbol{\phi}}, \ldots, \frac{Dg_M}{D\boldsymbol{\phi}}$ along with the TO FEM models necessary to resolve the state variables present in the TO problem. The input to a TO Problem is a design in the form of a design parameter vector $\boldsymbol{\phi}^{\{k\}}$, with its size determined by the number of materials used in the problem. The output is then the function values and values of the first order derivatives for the provided design. A schematic view of a TO problem is shown in figure 3.2.

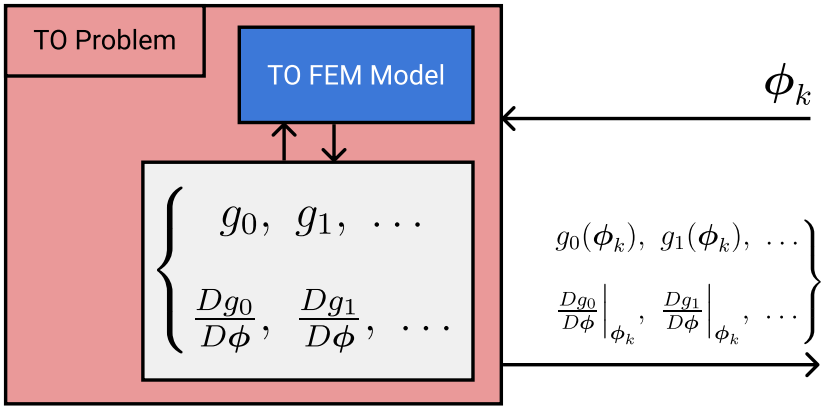


Figure 3.2: Schematic showing what is included in a TO problem object and what it takes as input and outputs when sent to an optimizer.

FEM Models

Several different TO problems are to be run by the program which leads to the need of several FEM models. Also, because structured mesh and unstructured mesh differ when implementing the methods used in a FEM model, there is a need of many different FEM classes. Much of the properties and methods can however be shared between the classes, so a family tree of the FEM classes was created. In figure 3.3 the class tree structure is visible. At the top of the tree is the common class of all FEM classes, `FEMBase`, holding the most common properties like the element connectivity mapping and node coordinates. For the thermal and mechanical problem there are separate common classes for these problems as well. They hold most of the properties and implements most of the methods related to the Finite Element Method. These common classes then have one subclass for the structured mesh and one for the unstructured mesh. Each concrete subclass then have a topology optimization related subclass which holds the design parameters for optimization and allows for material interpolation when calculating the material properties in different elements. Lastly, the coupled topology optimization class inherits from the thermal TO FEM model and has the mechanical one as a property. This technicality is needed as inheritance from both would have conflicts and inheritance from none would make the class not fulfill the interface of any FEM class.

3.3 Studied problems

To validate the correctness of the TO problems implemented and evaluate the results from the TO solver, 4 different test cases have been studied together with a real-life application.

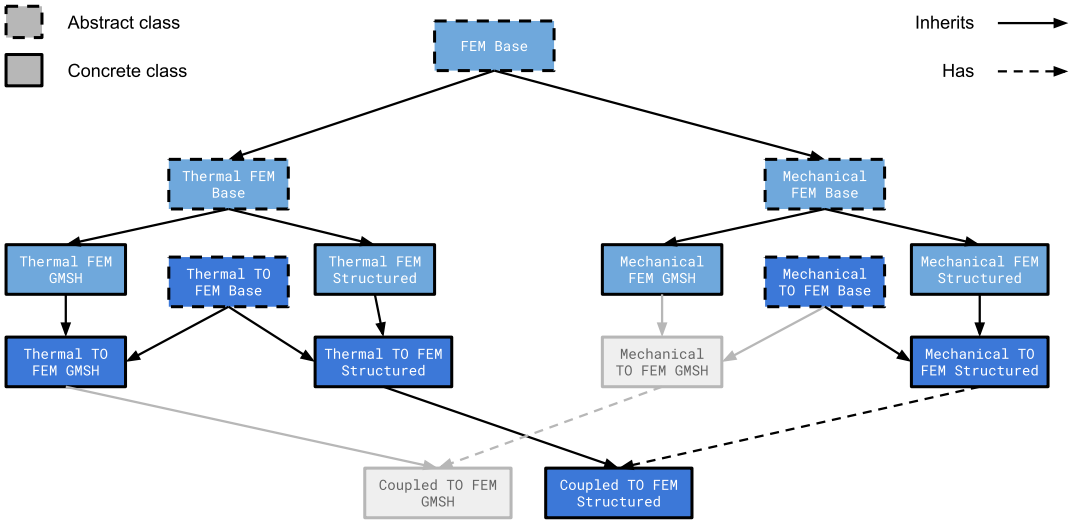


Figure 3.3: Class tree of the Finite Element Method classes and their topology optimization (TO) counterparts, marked with dark blue. The gray boxes mark classes not yet implemented.

If not stated otherwise, the meshes used in the test cases have consisted of isoparametric quadrilateral or hexahedron finite elements, ordered in a structured way. For the 2D cases, the thickness of the geometry is set to 1 m. For the time integration of the temperature, the backward Euler method has been used. The standard penalization parameter setting has been 3. To generate stiff thermal actuators, a linear spring has been connected to the output node for all cases on max displacement. The termination criterion has been $|g_0(\boldsymbol{\phi}^{\{k\}} - g_0(\boldsymbol{\phi}^{\{k-1\}})| < 10^{-7} \cdot |g_0(\boldsymbol{\phi}^{\{k\}})|$. The SIMP material interpolation model has been used. For the cases with multi-materials and 2 design parameters to interpolate, the penalization for ϕ_I and ϕ_{II} has been the same. In all cases, MMA has been used as the optimizer.

For all the details describing how the test cases were implemented, the reader is referred to the GitHub repository (commit `92e8600b0`) and the files describing the runs, `run1.m` - `run5.m`.

Test case 1

The first test case is the minimization of the maximum temperature in a 2D square body by distributing two material, one with low thermal conductivity and one with high thermal conductivity (see table 3.1 for exact property values). The body, seen in figure 3.4, is heated by a point load in the middle of the geometry, constant over time. In all four corners, the heat is able to flow out of the body, thanks to the temperature being prescribed at the corners. The initial temperature is uniformly 0 K all over the body, the same as the temperature at the corners.

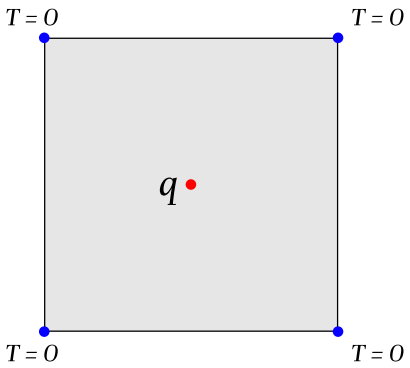


Figure 3.4: The geometry and boundary conditions for test case 1. The 2D domain has dimensions 0.1m × 0.1m.

To have a smooth objective function, the max function is approximated by a smooth maximum. The smooth maximum function used in this test case is the following, taken from [8]

$$\text{smax}(x_1, x_2, \ldots, x_N; a) = \frac{\sum_{i=1}^{N} x_i a^{x_i}}{\sum_{i=1}^{N} a^{x_i}} \tag{3.1}$$

Care must be taken when computing this function in the implementation, as it may very likely lead to floating point overflow. A simple solution to this is to normalize the temperatures when calculating the maximum temperature.

The smooth maximum function is applied in two steps to receive the maximum temperature, first taking the maximum of every nodal temperature for each time step, then taking the maximum over every time step. The only constraint included in the optimization problem is a volume constraint restricting material 2 to take up a maximum of 20 % of the volume.

Materials	ρ	c_p	κ	E	ν	α
Material 1	1	5E5	0.1	-	-	-
Material 2	1E3	1E3	10	-	-	-

Table 3.1: Properties of the materials used in test case 1 and 2.

$$\mathcal{P}_1 = \begin{cases} \min\limits_{\boldsymbol{\phi} \in \boldsymbol{\Phi}} \; g_0 = \text{smax}\left(\text{smax}(\boldsymbol{T}^{(1)}), \ldots, \text{smax}(\boldsymbol{T}^{(N)})\right) \\ \text{s.t.} \quad \begin{cases} \boldsymbol{R}_T^{(n)}(\widehat{\boldsymbol{\phi}}; \boldsymbol{T}^{(n)}, \boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ g_1(\widehat{\boldsymbol{\phi}}) = \dfrac{\sum_i \phi_i v_i}{V^*} - 1 \leq 0 \end{cases} \end{cases} \tag{3.2}$$

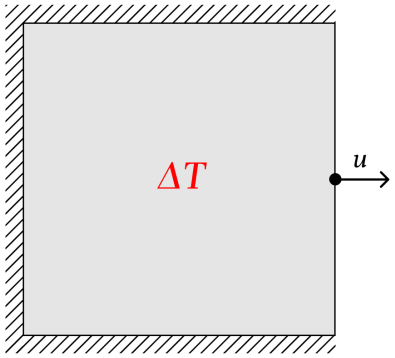


Figure 3.5: Geometry for test case 3. All side except the right one are fixed. The dimensions of the 2D domain are 400nm × 400nm.

Test case 2

The geometry in test case 1 is extended in test case 2 into 3D and into a cube with dimensions 0.1 m × 0.1 m × 0.1 m. The heat from the point heat is then able to dissipate through the 8 corners of the cube.

Test case 3

The third test case looks at a thermo-elastic problem, where the temperature change over the domain is uniform and fixed at $100K$. The objective function is to maximize the displacement at the center of the right side of the domain, as seen in figure 3.5. This is done by distributing 3 different materials. The first material represents void. The two others are low conducting metals, but differ in their CTE, see table 3.2.

Materials	ρ	c_p	κ	E	ν	α
Material 1 (void)	1E-3	1E9	1E-3	100E3	0.3	0
Material 2	1	1E6	10	100E9	0.3	1E-5
Material 3	1	1E6	10	100E9	0.3	2E-5

Table 3.2: Materials used in the optimization of test case 3 and 4.

The objective function can be calculated as the product of the nodal displacement vector due to the temperature change and an imaginary force $\boldsymbol{F}$. This force is usually referred to as a dummy load. As the goal is to maximize the displacement, the objective function takes the negative value of the displacement product. At the output is also a linear spring with stiffness $1 \times 10^7\,\mathrm{N\,m^{-1}}$.

A robust formulation is also used on this problem to avoid 1-node hinges in the

design. The threshold shift $\Delta\eta$ is set to 0.2 in the implementation. The sensitivity of the max function in $\bar{g_0}$ is replaced by the sensitivity of the objective function with the maximum value in the current iteration. The mass constraint takes the dilated design as input and limits the mass of non-void material to 20 % of the mass of a design taking up the entire domain with material 3. Material 2 and 3 have here the same density, which leads to no one of them being preferred over the other when it comes to the mass constraint.

$$\mathcal{P}_3 = \begin{cases} \min\limits_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \ \bar{g_0} = \max(g_0({}^d\overline{\phi}), g_0({}^i\overline{\phi}), g_0({}^e\overline{\phi})) \\ g_0 = -\boldsymbol{F}^T\bar{\boldsymbol{u}} \\ \text{s.t.} \quad \begin{cases} \boldsymbol{R}_u({}^d\overline{\boldsymbol{\phi}}; {}^d\bar{\boldsymbol{u}}) = \boldsymbol{0} \\ \boldsymbol{R}_u({}^i\overline{\boldsymbol{\phi}}; {}^i\bar{\boldsymbol{u}}) = \boldsymbol{0} \\ \boldsymbol{R}_u({}^e\overline{\boldsymbol{\phi}}; {}^e\bar{\boldsymbol{u}}) = \boldsymbol{0} \\ g_m({}^d\overline{\boldsymbol{\phi}}) = \dfrac{\sum_i \rho({}^d\overline{\phi}_{I,i})v_i}{m_d^*} - 1 \leq 0 \end{cases} \end{cases} \tag{3.3}$$

Test case 4

The fourth test case is a proper transient thermo-mechanical topology optimization problem, with the same objective function as in test case 3. The difference is that the temperature field is variable and controlled by the heat transfer equations. The heating is caused by a prescribed temperature of 100 K at the left corners, see figure 3.6. The initial temperature is uniformly set to 0 K. Depending on the simulated time, t_f, the temperature input will have different possibility to affect the whole domain, and in theory, this should affect the optimal design to the TO problem. A spring with the same stiffness as in test case 3 is connected to the output node.

The materials allowed in the design are the same as in test case 3, see table 3.2. Notice how the volumetric heat capacity s is set to be the same in the void and the non-void. This prevents the void to gain a too high temperature compared to if it had a lower volumetric heat capacity.

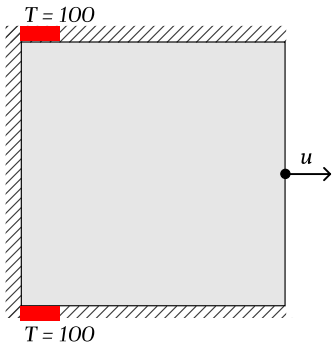


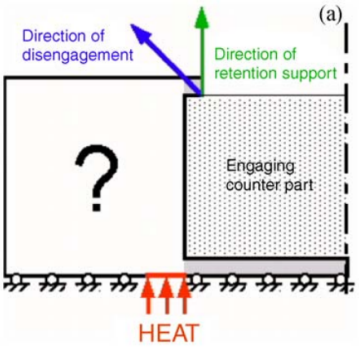
Figure 3.6: Geometry for test case 4. The displacements on all sides are fixed, except on the right side. In the left corners, the temperature is prescribed and everywhere else the boundary is adiabatic. The dimensions of the 2D domain are 400nm × 400nm.

$$
\mathcal{P}_4 = \begin{cases} \min\limits_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \ \bar{g}_0 = \max({}^{d}\overline{\phi}), g_0({}^{i}\overline{\phi}), g_0({}^{e}\overline{\phi})) \\ g_0 = -\boldsymbol{F}^T \bar{\boldsymbol{u}}^{\boldsymbol{N}} \\ \text{s.t.} \quad \begin{cases} \begin{cases} \boldsymbol{R}_T^{(n)}({}^{e}\boldsymbol{\phi}; {}^{e}\boldsymbol{T}^{(n)}, {}^{e}\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^{e}\boldsymbol{\phi}; {}^{e}\bar{\boldsymbol{u}}^{(n)}, {}^{e}\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \\ \begin{cases} \boldsymbol{R}_T^{(n)}({}^{i}\boldsymbol{\phi}; {}^{i}\boldsymbol{T}^{(n)}, {}^{i}\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^{i}\boldsymbol{\phi}; {}^{i}\bar{\boldsymbol{u}}^{(n)}, {}^{i}\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \\ \begin{cases} \boldsymbol{R}_T^{(n)}({}^{d}\boldsymbol{\phi}; {}^{d}\boldsymbol{T}^{(n)}, {}^{d}\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1, 2, \ldots, N \\ \boldsymbol{R}_u({}^{d}\boldsymbol{\phi}; {}^{d}\bar{\boldsymbol{u}}^{(n)}, {}^{d}\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0, 1, \ldots, N \end{cases} \\ g_v({}^{d}\overline{\boldsymbol{\phi}}) = \dfrac{\sum_i {}^{d}\overline{\phi}_{I,i} v_i}{V_d^*} - 1 \leq 0 \end{cases} \end{cases} \tag{3.4}
$$

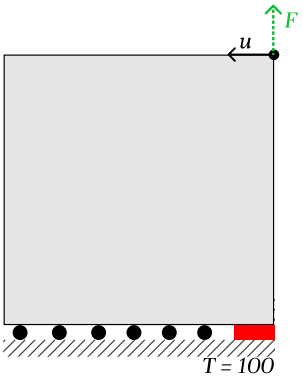
Thermally actuated disassembly mechanism

The last case is an optimization of a real-life application, namely a thermally actuated disassembly mechanism. The principle of the mechanism is described in [10], where it is also optimized. The thermally actuated mechanism should be able to snap-fit into place to assemble two different parts. Snap-fit mechanism are commonly made of elastic material like plastic and act as the working principle in side release buckles. To later disassemble the two parts, an applied heat increases the temperature in the mechanism and thermal expansion causes it to bend, allowing the two parts to release from each other.

The heating is assumed to be applied for a limited time, to prevent unnecessary thermal expansion, but also because reaching steady-state conditions take time, which is undesirable when disassembling.



(a) Disengagement and retention schematics.



(b) Design domain for the disassembly problem. The 2D geometry has the dimensions 10 cm × 10 cm.

Figure 3.7: Disassembly concept

This concept of an thermally actuated disassembly mechanism is transferred into a topology optimization problem, where the displacement in a certain part should be maximized. This formulation resembles test case 4, but an extra constraint must be formulated for this application. As the mechanism should be able to engage and retain the two parts without them releasing, it must have a certain stiffness in the retention direction. This constraint act as a limit on the compliance of the mechanism when applied with a force, see figure 3.7.

$$
\mathcal{P}_5 = \begin{cases} \min\limits_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \ \bar{g}_0 = \max({}^{d}\overline{g_0}({}^{d}\overline{\phi}), g_0({}^{i}\overline{\phi}), g_0({}^{e}\overline{\phi})) \\ g_0 = -\boldsymbol{F}^T\bar{\boldsymbol{u}}^{\boldsymbol{N}} \\ \text{s.t.} \begin{cases} \begin{cases} \boldsymbol{R}_T^{(n)}({}^{e}\boldsymbol{\phi}; {}^{e}\boldsymbol{T}^{(n)}, {}^{e}\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1,2,\ldots,N \\ \boldsymbol{R}_u({}^{e}\boldsymbol{\phi}; {}^{e}\bar{\boldsymbol{u}}^{(n)}, {}^{e}\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0,1,\ldots,N \end{cases} \\ \begin{cases} \boldsymbol{R}_T^{(n)}({}^{i}\boldsymbol{\phi}; {}^{i}\boldsymbol{T}^{(n)}, {}^{i}\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1,2,\ldots,N \\ \boldsymbol{R}_u({}^{i}\boldsymbol{\phi}; {}^{i}\bar{\boldsymbol{u}}^{(n)}, {}^{i}\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0,1,\ldots,N \end{cases} \\ \begin{cases} \boldsymbol{R}_T^{(n)}({}^{d}\boldsymbol{\phi}; {}^{d}\boldsymbol{T}^{(n)}, {}^{d}\boldsymbol{T}^{(n-1)}) = \boldsymbol{0}, \quad \forall n = 1,2,\ldots,N \\ \boldsymbol{R}_u({}^{d}\boldsymbol{\phi}; {}^{d}\bar{\boldsymbol{u}}^{(n)}, {}^{d}\boldsymbol{T}^{(n)}) = \boldsymbol{0}, \quad \forall n = 0,1,\ldots,N \end{cases} \\ g_v({}^{d}\overline{\boldsymbol{\phi}}) = \dfrac{\sum_i {}^{d}\overline{\phi}_{I,i} v_i}{V_d^*} - 1 \leq 0 \\ \bar{g}_1 = \max({}^{d}\overline{g_1}({}^{d}\overline{\phi}), g_1({}^{i}\overline{\phi}), g_1({}^{e}\overline{\phi})) \\ g_1 = -\boldsymbol{F}_2^T\bar{\boldsymbol{u}}_{\boldsymbol{2}} \\ \boldsymbol{R}_u({}^{d}\overline{\boldsymbol{\phi}}; {}^{d}\bar{\boldsymbol{u}}_{\boldsymbol{2}}) = \boldsymbol{0} \\ \boldsymbol{R}_u({}^{i}\overline{\boldsymbol{\phi}}; {}^{i}\bar{\boldsymbol{u}}_{\boldsymbol{2}}) = \boldsymbol{0} \\ \boldsymbol{R}_u({}^{e}\overline{\boldsymbol{\phi}}; {}^{e}\bar{\boldsymbol{u}}_{\boldsymbol{2}}) = \boldsymbol{0} \end{cases} \end{cases} \tag{3.5}
$$

Chapter 4

Topology optimized results

4.1 Test case 1

Test case 1 as described in the implementation chapter was implemented into the TO solver and the results of the topology optimization was studied for different choices of penalization for the SIMP material interpolation model together with different choices of simulated time, t_f. The results can be seen in table 4.1

4.2 Test case 2

Test case 2 is an extension of test case 1 into 3 dimensions. It is used mainly to showcase the possibility to optimize 3D geometries. Only one topology optimization test case 2 was performed, with $t_f = 500\,\mathrm{s}$ and the SIMP penalization parameters set to 3: $p_\kappa = 3$, $p_s = 3$.

4.3 Test case 3

For the third test case, the SIMP penalization parameter for the Young's modulus and the thermal stress coefficient have been varied. In table 4.2, the optimal designs for different penalization parameters are shown. The value of the objective function for these designs are given in table 4.3. One of the results are displayed with the optimal design undeformed and deformed in figure 4.2. In all these images the void is transparent and material 2 and 3 are depicted with yellow or blue material, or intermediate colors (e.g. green) for intermediate materials.

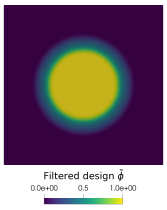
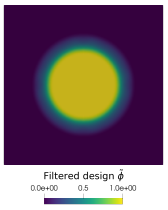
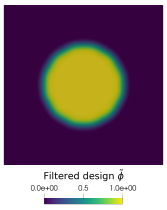
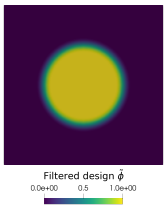
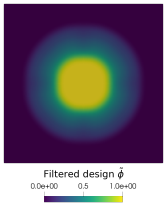
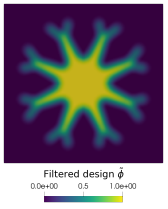
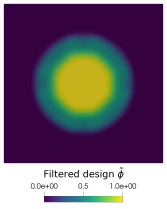
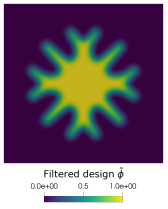
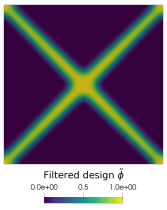
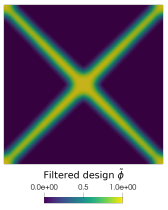
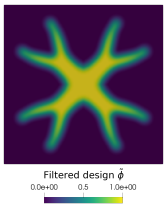
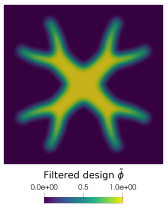
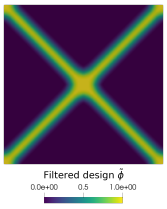
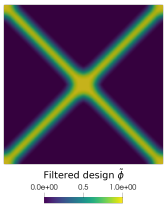
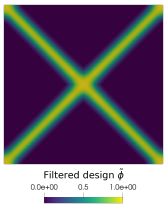
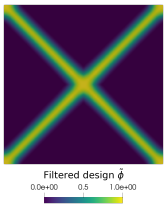
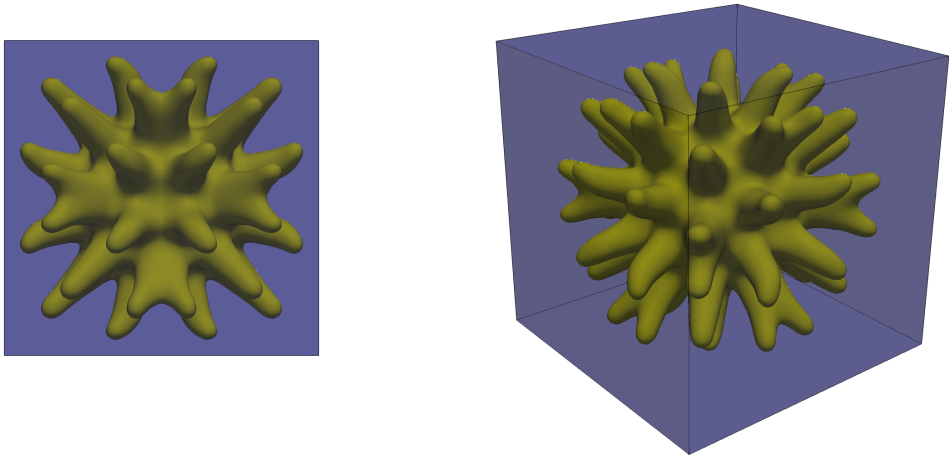
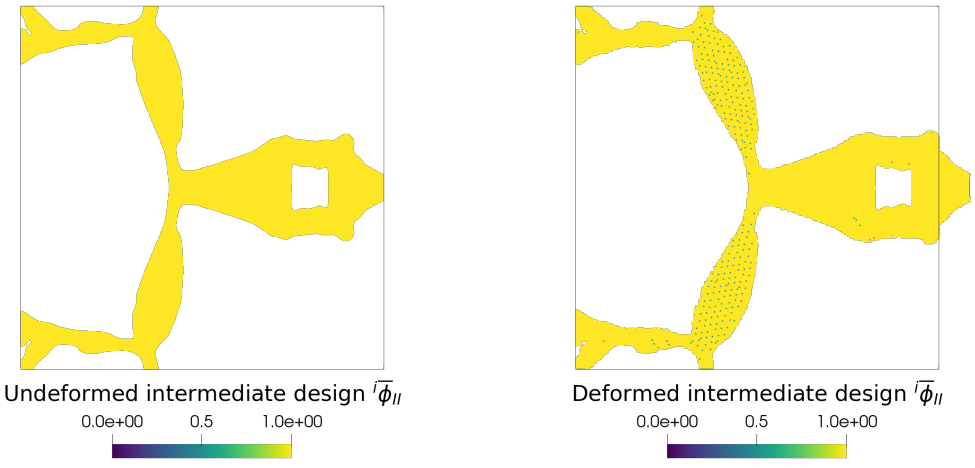
t_f	$(p_\kappa, p_s) = (2, 2)$	$(p_\kappa, p_s) = (2, 3)$	$(p_\kappa, p_s) = (3, 2)$	$(p_\kappa, p_s) = (3, 3)$
100 s				
500 s				
2'000 s				
10'000 s				

Table 4.1: Optimal design for different time intervals and penalization parameters. The yellow areas are filled with material 1 and the dark blue areas are filled with material 2.



(a) Orthographic top view (b) Perspective view

Figure 4.1: Optimal design for the 3D problem. To view the material distribution of material 1, a contour surface have been drawn in yellow at $\widehat{\phi} = 0.5$. Material 2 fills the rest of the volume of the cube.



(a) Orthographic top view (b) Perspective view

Figure 4.2: Scale factor 10

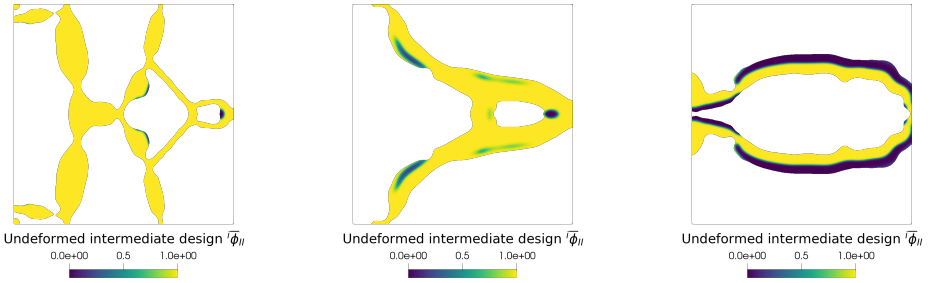
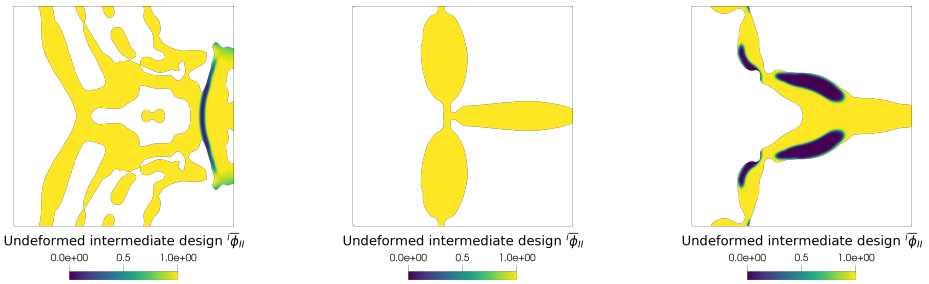
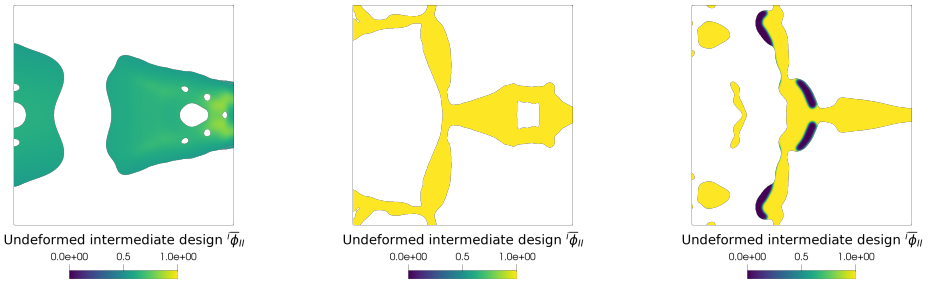
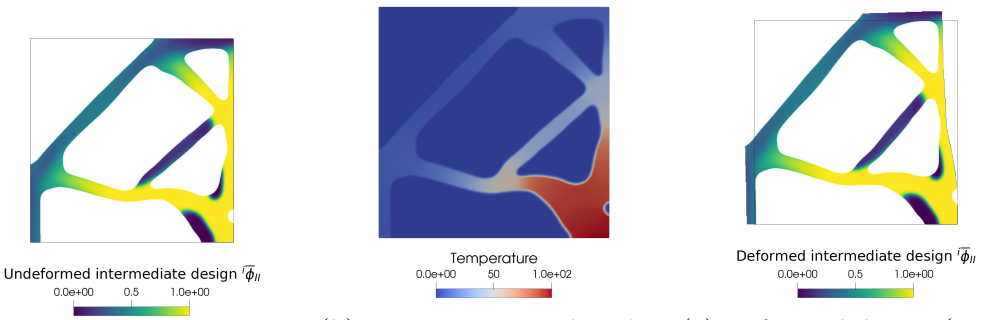
-	$p_r = 2$	$p_r = 3$	$p_r = 4$
$p_E = 2$			
$p_E = 3$			
$p_E = 4$			

Table 4.2: Optimal design for different penalization parameters.

g_0	$p_r = 2$	$p_r = 3$	$p_r = 4$
$p_E = 2$	-2.4984	-1.0926	-1.2900
$p_E = 3$	-1.3332	-3.1667	-1.6276
$p_E = 4$	-0.8860	-2.4815	-2.0685

Table 4.3: Values of the objective function for the designs in table 4.2, scaled by a factor 10^6.



(a) Undeformed design

(b) Temperature distribution after t_f seconds.

(c) Deformed design (with a scale factor 10).

Figure 4.3: Optimal design for the thermally actuated disassembly mechanism.

4.4 Test case 4

The results for the fourth test case show how the optimal design varies with the SIMP penalization parameter used, seen in table 4.4 and the corresponding objective function values in table 4.5. Note how a study for p_s was not performed, as the volumetric heat capacity was set to be the same for all materials in this test case.

In table 4.6, the optimal design for different t_f can instead be seen. In these runs, the SIMP penalization parameters were chosen to be $p_\kappa = 2, p_E = 3$ and $p_r = 4$. This choice was based on the best performing results of previous runs.

The last design in table 4.6 performs best in terms of displacement, where the objective function takes the value -1.4019 (scaled with a factor 10^6).

4.5 Thermally actuated disassembly mechanism

The thermally actuated disassembly mechanism problem was run for $t_f = 600\,\mathrm{s}$ at all penalization parameter set to 3. The result is seen in figure 4.3.

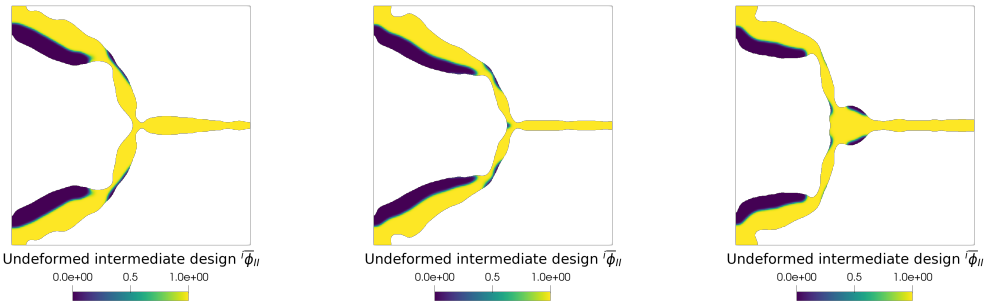
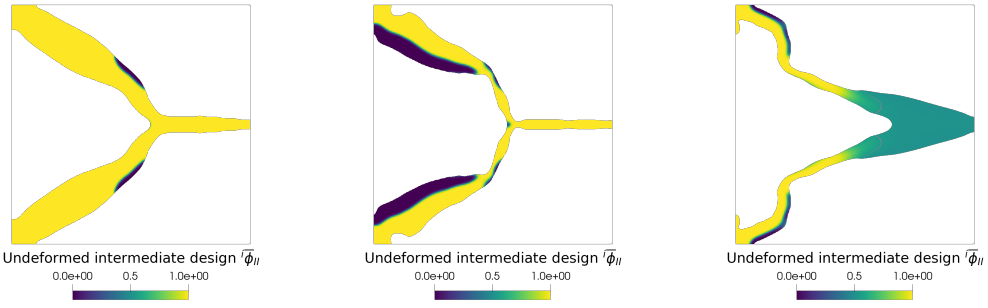
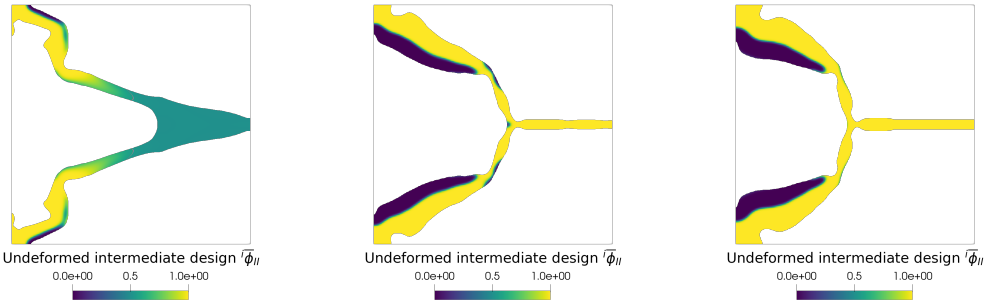
-	2	3	4
p_κ			
p_E			
p_r			

Table 4.4: Different optimal design when varying one penalization parameter at the time between 2 and 4. The parameter that has been varied are written in the first column. All other penalization parameters have then been set to 3.

g_0	2	3	4
p_κ	-1.2267	-1.0393	-1.1734
p_E	-0.8248	⋮	-0.6914
p_r	-0.7056	⋮	-1.1087

Table 4.5: The values of the objective function for the cases in table 4.4, scaled with a factor 10^6

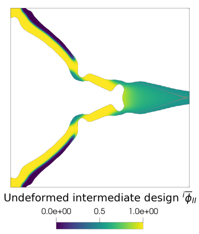
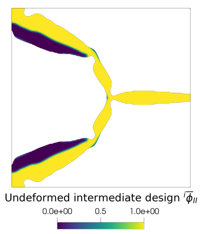
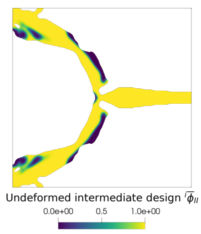
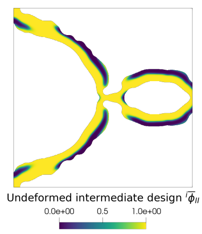
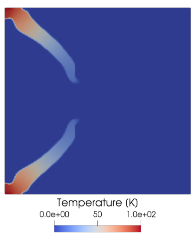
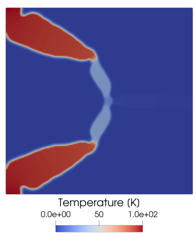
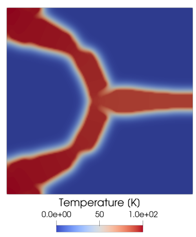
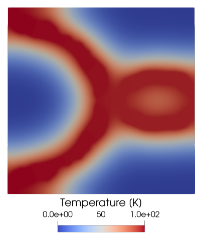
t_f	0.001 s	0.01 s	0.1 s	1 s
				
				

Table 4.6: Optimal design and final temperature for different time intervals

Chapter 5

Discussion

From a broad perspective, the main objectives of the thesis have been achieved through the results presented in this report, to develop a working thermo-mechanical TO solver for transient heat transfer. The real-life application clearly shows how the TO solver finds a design which successfully actuates due to thermal expansion, caused by a transient temperature field. Furthermore, all the test cases are in line with previous results (except for test case 4 for which there are not previous results) from other authors.

The robust formulation rendered satisfactory results, successfully preventing 1-node hinges. However, as this is not the main focus of the thesis, its effects will not be discussed here.

5.1 Verification against previous results

The results from test case 1 in table 4.1 tries to replicate the results in the paper by Wu et. al [8]. Several results differ from the replicated results, but the overall features are there and the transition from circular designs to cross-like designs happens in the same range for the simulated time, t_f. The reason for the differences in the design are thought to be from numerical settings, such as the mesh resolution or the normalization of the temperatures.

The second test case is an extension of test case 1 and have the same settings as for the designs with $t_f = 500\,\mathrm{s}$ and $(p_\kappa, p_s = (3, 3))$ in table 4.1. As expected the 3D design showcase the same type of features as the 2D design, only in three dimensions.

The results for the third test case in table 4.2 are best compared to the results by Sigmund [12]. Notice how there are several differences in material properties and spring stiffness between test case 3 and the cases Sigmund looked at. However, some features are still occurring in both cases, for example the slightly angled, thickened bars in $(p_E, p_r) = (3, 3)$ or the compliant structure with several hinges

and bars in $(p_E, p_r) = (2, 2)$. It is also noticed from the results for test case 3 that the penalization parameters have a great impact on the final optimal design and that the thermo-elastic problem seem to have local minimum where the designs are fundamentally different from each other.

5.2 Study of the transient thermo-mechanical problem

Test case 4 studies the topology optimization of transient thermal actuators. No previous work using this test case for transient thermal actuators was found and therefore there are no other results to compare to.

The first thing to study in the results is the influence of the elapsed time, t_f. This should hopefully have a significant effect on the design, as it is the reason why transient heat transfer is simulated at all. The optimal designs in table 4.6 show how the elapsed time influences the problem. The conducting paths are similar for all designs. For the case with the shortest elapsed time, the area of increased temperature is limited to the "legs" of design, and should be the reason why green, intermediate, material is present in the tip. The optimizer does not prefer any material over the other, thus leaving it to what was initially set. Another interesting observation to make it how all designs use both materials in the design to cause bending motions, even for the design where the elapsed time is 1 s. For a large t_f that in the end generates designs where the temperature field becomes uniform, the optimal design should resemble the ones from test case 3. The boundary conditions of test case 4 makes it however not possible to directly compare the results. If the boundary condition in test case 4 was extended to all sides of the domain except the right one, a more fair comparison could be made.

The best performing design for the uniform temperature field are all single-material. Why the designs in test case 4 are all instead multi-material design is not clear. Another important study is the one done on test case 4 regarding different penalization parameters, with the results in table 4.4. To notice from the results are how the relationship between the penalization parameter for E and r seems to be important. A high penalization on E compared to the penalization on r is correlated to low performing designs. The optimal designs for $p_E = 4$ and $p_r = 2$ in table 4.4 showcase this pattern. They are both very similar in design as well. A flaw with the combined interpolated material property r, is that it may induce different penalization on Young's modulus in the stiffness matrix $\boldsymbol{K}_{uu}$ and the thermal stiffness matrix $\boldsymbol{K}_{uT}$. It would be interesting to do a direct comparison with interpolating the CTE separately instead.

A reoccurring problem which is related to compliant mechanisms (and therefore also thermal actuators) is that the mass constraint is not always fulfilled. There is no direct correlation between the displacement and the volume of the structure,

making it difficult to penalize intermediate design values. The penalization simply have not the same effect as for a minimum compliance problem. The route chosen in this thesis was to use Heaviside projection and a robust formulation to force intermediate values to the extremes. Another methods, like direct penalization of intermediate values in the objective function, has not been explored properly.

5.3 Thermally actuating disassembly mechanism

As mentioned, the disassembly mechanism is supposed to act as a more applicable case. It is therefore interesting to discuss the performance and the final design in detail. From a topology optimization point of view, the design should be considered satisfactory, as the geometry is clearly defined and the the design actuates a noticeable distance. From a functional perspective however, it is disputable how useful the thermally actuating disassembly mechanism is. An issue is the performance, as the displacement compared to the size of the mechanism is small. For the materials chosen, the actuating time is also quite long, 10 min. A third issue is the multi-material. Even though manufacturing components with complex distribution of multiple material is possible via 3D-printing, no study has been done on the cost of this, nor how good the interface between two material tolerate stress, something which is generated by design in the optimal design.

A topic for the future could be to investigate the potential of thermal actuator and larger displacements. This should preferably also include a study on the most suitable materials to use to achieve large displacements in a short time.

Chapter 6

Future work

This thesis has explored a broad topic related to transient thermo-mechanical topology optimization. Because of the many parameters affecting the optimization and the design, a lot of questions are still unanswered, for example where thermal actuators may be used besides micro grippers.

In the paper by Sigmund on thermal actuators [11], Sigmund looks at geometrical non-linearities and also incorporates a non-linear model into the optimization. Through this, he showed how modelling non-linearities resulted in different design which were also performing better. To model the thermal actuators in a better way, large displacements and geometrical non-linearities must be taken into consideration. To implement this would simply mean to create a new non-linear mechanical FEM model, see figure 3.3 for the existing classes. Simulating non-linearities is however much more computationally expensive.

Another topic to study further has already been mentioned in the discussion, the potential of thermal actuators. Sigmund present in his paper [11] a theoretical limit on the work performed by a linear thermal actuator. In the second part [12], he also derives a similar limit for multi-material design which shows how single-material design for most material choices are superior to multi-material ones. These limits do however not consider transient heat and the global temperature differences achievable through this. Imagine for example a thermal actuator with two material, one conducting and one isolating. If it would be possible to distribute the material in such a way that only parts of the actuator was heated, large differences in strain would be achievable, resulting in bending forces.

Lastly, since the transient thermo-mechanical TO problem has been handled in such a general way throughout this thesis, much of the principle and implementations done in this thesis is applicable to any other transient thermo-mechanical TO problem, unrelated to thermal actuators. An example could be the minimization of the compliance in a structure, having a constraint on the maximum thermal stress induced by a time-dependent heat flow.

Appendix A

Expansion and reduction matrices

In the context of the finite element method, a expansion matrix $\boldsymbol{E}$, sometimes called kinematic matrix [21, p. 181], expands a local vector related to a finite element into the size of the global vector and maps the local numbering of the degrees of freedom to the global numbering, filling all the empty elements in the vector with zeros. The operation is exemplified below with the local shape function matrix, which is expanded to the size of the global shape function matrix. Note that no additional information from any other elements are added to the global matrix, but because of the behavior of shape functions, this global matrix is sufficient in the local domain of the element.

$$\boldsymbol{N}\Big|_{\Omega_e} = \boldsymbol{N}_{ee} = \boldsymbol{E}_e \boldsymbol{N}_e \tag{A.1}$$

Here $\boldsymbol{N}_{ee}$ have the matrix dimensions [ndof $\times$ 1], $\boldsymbol{N}_e$ [ndof$_e$ $\times$ 1] and $\boldsymbol{E}_e$ then having the dimensions [ndof $\times$ ndof$_e$], where ndof is the total number of degrees of freedom and ndof$_e$ is the number of degrees of freedom in element e.

Taking the transpose of the expansion matrix, $\boldsymbol{E}_e^T$, can then be denoted the reduction matrix, which takes a globally sized matrix and reduces it to the size of a local matrix, where only the information related to the finite element e is kept, and all other is discarded.

The elements of an expansion matrix are either 0 or 1, distributed so that the sum over each column equal 1 and the sum over each row equal either 0 or 1. A general structure is shown below.

$$\boldsymbol{E} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ & & \vdots & \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ & & \vdots & \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.2}$$

Bibliography

[1] *Encyclopedia of Nanotechnology*. Springer Netherlands, 2012.

[2] O. Sardan, D. H. Petersen, K. Mølhave, O. Sigmund, and P. Bøggild, "Topology optimized electrothermal polysilicon microgrippers," *Microelectronic Engineering*, vol. 85, no. 5-6, pp. 1096–1099, 5 2008.

[3] M. P. Bendsøe and N. Kikuchi, "Generating optimal topologies in structural design using a homogenization method," *Computer Methods in Applied Mechanics and Engineering*, vol. 71, no. 2, pp. 197–224, 11 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0045782588900862

[4] T. Dbouk, "A review about the engineering design of optimal heat transfer systems using topology optimization," 2017.

[5] C. Zhuang, Z. Xiong, and H. Ding, "Topology optimization of the transient heat conduction problem on a triangular mesh," *Numerical Heat Transfer, Part B: Fundamentals*, vol. 64, no. 3, pp. 239–262, 5 2013.

[6] C. Zhuang and Z. Xiong, "Temperature-constrained topology optimization of transient heat conduction problems," *Numerical Heat Transfer, Part B: Fundamentals*, vol. 68, no. 4, pp. 366–385, 10 2015.

[7] K. Long, X. Wang, and X. Gu, "Multi-material topology optimization for the transient heat conduction problem using a sequential quadratic programming algorithm," *Engineering Optimization*, vol. 50, no. 12, pp. 2091–2107, 12 2018.

[8] S. Wu, Y. Zhang, and S. Liu, "Topology optimization for minimizing the maximum temperature of transient heat conduction structure," *Structural and Multidisciplinary Optimization*, vol. 60, no. 1, pp. 69–82, 7 2019.

[9] B. Zhu, X. Zhang, H. Zhang, J. Liang, H. Zang, H. Li, and R. Wang, "Design of compliant mechanisms using continuum topology optimization: A review," *Mechanism and Machine Theory*, vol. 143, p. 103622, 1 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094114X19315538

[10] Y. Li, K. Saitou, and N. Kikuchi, "Topology optimization of thermally actuated compliant mechanisms considering time-transient effect," *Finite Elements in Analysis and Design*, 2004.

[11] O. Sigmund, "Design of multiphysics actuators using topology optimization - Part I: One-material structures," *Computer Methods in Applied Mechanics and Engineering*, vol. 190, no. 49-50, pp. 6577–6604, 10 2001.

[12] ——, "Design of multiphysics actuators using topology optimization - Part II: Two-material structures," *Computer Methods in Applied Mechanics and Engineering*, vol. 190, no. 49-50, pp. 6605–6627, 10 2001.

[13] E. . Van De Ven, E. . Hooijkamp, M. . Langelaar, and F. Van Keulen, "Topology optimization of a transient thermo-mechanical problem using material penalization," in *Proceedings of the 11th World Congress of Structural and Multidisciplinary Optimization (WCSMO-11)*. University of Sydney, 2015.

[14] N. S. Ottosen and H. Petersson, *Introduction to the finite element method*. Prentice Hall, 1992.

[15] M. P. Bendsoe, "Structural Optimization Optimal shape design as a material distribution problem," Tech. Rep., 1989.

[16] M. Stolpe and K. Svanberg, "An alternative interpolation scheme for minimum compliance topology optimization," *Structural and Multidisciplinary Optimization*, vol. 22, no. 2, pp. 116–124, 2 2001.

[17] M. P. Bendsøe and O. Sigmund, "Material interpolation schemes in topology optimization," *Archive of Applied Mechanics*, vol. 69, no. 9-10, pp. 635–654, 1999. [Online]. Available: https://link.springer.com/article/10.1007/s004190050248

[18] O. Sigmund, "Morphology-based black and white filters for topology optimization," *Structural and Multidisciplinary Optimization*, vol. 33, no. 4-5, pp. 401–424, 4 2007.

[19] F. Wang, B. S. Lazarov, and O. Sigmund, "On projection methods, convergence and robust formulations in topology optimization," *Structural and Multidisciplinary Optimization*, vol. 43, no. 6, pp. 767–784, 6 2011. [Online]. Available: https://link.springer.com/article/10.1007/s00158-010-0602-y

[20] K. Svanberg, "The method of moving asymptotes—a new method for structural optimization," *International Journal for Numerical Methods in Engineering*, vol. 24, no. 2, pp. 359–373, 2 1987. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/nme.1620240207https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.1620240207https://onlinelibrary.wiley.com/doi/10.1002/nme.1620240207

[21] P. W. Christensen and A. Klarbring, "An introduction to structural optimization," *Solid Mechanics and its Applications*, vol. 153, pp. 1–220, 2008.

[22] P. Michaleris, D. A. Tortorelli, and C. A. Vidal, "Tangent Operators and Design Sensitivity Formulations for Transient Nonlinear Coupled Problems with Applications to Elasto-Plasticity," Tech. Rep.

[23] O. Günther-Hanssen, "Golho / }Thermo-Mechanical-Topology-Optimization," 2020. [Online]. Available: https://github.com/Golho/Thermo-Mechanical-Topology-Optimization

[24] "Overview - NLopt Documentation." [Online]. Available: https://nlopt.readthedocs.io/en/latest/