Universidade de Aveiro Departamento de Engenharia Mecânica
2018

Riccardo Oliveira

Topology optimization in structures with nonlinear behaviour

Otimização topológica em estruturas de comportamento não-linear

Universidade de Aveiro Departamento de Engenharia Mecânica
2018

Riccardo Oliveira

Topology optimization in structures with nonlinear behaviour

Otimização topológica em estruturas de comportamento não-linear

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Mecânica, realizada sob orientação científica de António Gil D'Orey de Andrade Campos, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro, e de João Alexandre Dias de Oliveira, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro.

Apoio financeiro dos projetos
UID/EMS/00481/2013-FCT e CENTRO-01-0145-FEDER-022083

O júri / The jury

Presidente / President	**Prof. Doutor Rui António da Silva Moreira** Professor Auxiliar da Universidade de Aveiro
Vogais / Committee	**Doutor Hugo Filipe Pinheiro Rodrigues** Professor Adjunto do Instituto Politécnico de Leiria
	Prof. Doutor João Alexandre Dias de Oliveira Professor Auxiliar da Universidade de Aveiro (co-orientador)

agradecimentos / acknowledgements

À minha família por todo o apoio demonstrado desde sempre, especialmente à minha mãe por tornar isto possível a todos os níveis. À Paula pela paciência, por estar sempre presente e por tantas coisas mais. Ao Professor Doutor António Gil D'Orey de Andrade Campos e ao Professor Doutor João Alexandre Dias de Oliveira, pela orientação, disponibilidade, compreensão e motivação no decorrer deste trabalho. A todos os professores pelos ensinamentos transmitidos quer a nível profissional como a nível pessoal. Aos meus amigos pela amizade, apoio, por todos os momentos vividos e ainda por viver e por me acompanharem sempre neste percurso. Aos meus colegas pelo companheirismo dentro e fora desta academia. A todos o meu muito obrigado!

keywords Topology optimization, non-linear elasticity, finite element method, Matlab

abstract Optimization in engineering is the search for the optimal solution, taking into account the defined design objective and the imposed constraints. There is a wide scope of applications in engineering, ranging from management and planning to structural design problems. The development of computers and programming has boosted this technology by enabling a faster and more efficient search for optimal solutions through computational methods. Topology optimization combines the Finite Element Method (FEM) with optimization methods, in order to obtain the best distribution of material within a given domain. In this work, two distinct design objectives were analyzed in the scope of evaluation of non-linear structural problems: (i) minimization of compliance and (ii) minimization of complementary work. The material of the structure is described by a multi-linear behaviour (nonlinear elasticity) and a Solid Isotropic Material with Penalization (SIMP) approach. In order to solve the problem of topology optimization, a computational tool, enclosing a FEM analysis and a topology optimization algorithm, was developed using Matlab programming. The validation of the FEM was done using the numerical simulation program Abaqus as a reference, and, for the topology optimization problems, simulations were performed for several boundary conditions.

palavras-chave

Otimização topológica, elasticidade não-linear, método dos elementos finitos, Matlab

resumo

A otimização em engenharia é a busca pela solução ótima tendo em consideração o objetivo de projeto e as restrições impostas. Existe uma panóplia de aplicações desta temática em engenharia, que vão desde problemas de gestão e planeamento até problemas de projeto de estruturas. O desenvolvimento dos computadores e da programação impulsionaram esta tecnologia permitindo uma procura mais rápida e eficaz de soluções ótimas através de métodos computacionais. A otimização topológica combina o Método dos Elementos Finitos (MEF) com métodos de otimização, de forma a obter a melhor distribuição de material para um determinado domínio. Neste trabalho foram analisados dois objetivos distintos no contexto de problemas estruturais não-lineares: (i) minimização do trabalho das forças externas e (ii) minimização do trabalho complementar. O material da estrutura é descrito por um comportamento isotrópico multilinear (elasticidade não-linear). De modo a solucionar o problema de otimização topológica foi desenvolvida uma ferramenta computacional, que engloba um código de análise pelo MEF e um algoritmo de otimização topológica, usando o Matlab como plataforma de programação. A validação do método dos elementos finitos foi feita usando o programa de simulação numérica Abaqus como referência, e, para o problema de otimização topológica, foram realizadas simulações para diversas condições de fronteira.

Contents

List of Tables

Intentionally blank page.

List of Figures

Nomenclature

$\boldsymbol{\rho}$	Densities vector
$\boldsymbol{\sigma}$	Stress tensor
$\boldsymbol{\varepsilon}$	Strain tensor
ν	Poisson's ratio
$\mathbf{B}$	Displacement-strain matrix
$\mathbf{C}$	Constitutive tensor
$\mathbf{f}$	Load vector
$\mathbf{I}$	Identity matrix
$\mathbf{J}$	Jacobian matrix
$\mathbf{K}$	Global stiffness matrix
$\mathbf{K}_e$	Element stiffness matrix
$\mathbf{u}$	Displacement vector
$\mathbf{u}_e$	Element displacement vector
E	Young's modulus
N	Shape functions

Intentionally blank page.

Part I

Introduction and Background

Chapter 1

Introduction

1.1 Theoretical Framework

Optimization means finding an optimal solution with the most cost-effective or highest achievable performance under the given constraints. It is achieved by maximizing desired factors and minimizing undesired ones. Thus, it is important to define what is intended to be optimized and what are the constraints to take into account. For instance, to maximize the number of objects inside a container it is essential to know aspects like the size or shape of the container and objects because these will limit the variable that is being optimized in order to find a possible solution. In a technical sense, optimization is the process of maximizing or minimizing the required objective function respecting a set of constraints. Optimization can be applied to many engineering areas such thermodynamics, fluid mechanics or electronics, in problems such as design of aerospace and aeronautic structures, design of structures (bridges, towers, dams) with minimum cost, inventory and stock control, design of control systems, planning market strategies, pump and turbine design with maximum performance, and many others. [1]

The optimization of a structure with the objective of finding the maximum stiffness with minimum weight or cost is called Structural Optimization (SO). The basic idea of Topology Optimization (TO) is to find the optimal layout of a given amount of material within a prescribed design domain in order to obtain the best design performance. Typically, TO is carried out using the numerical framework of finite element analysis (FEA), with the FEA model parametrized by design variables [2]. Commonly the variables used are density or magnitude of the material properties. They can vary between different phases of the material or merely empty, assuming values between 1 and 0, respectively. It was first discussed in problems of civil engineering by Maxwell [3] with the objective of reducing the material used in a bridge. Years later Michell [4] studied optimum structures with the minimum possible material, naming the structure called Michell truss that is the optimum structure for a cantilever beam with a point load at the end. In this structure, the elements are subject to nominal stress with zero shear stress. Since its introduction a decade ago, the topology optimization method [5] has gained popularity in academia and industry. The increased use of SO is also a consequence of the advances in computational mechanics. The conventional topology optimization method is concerned in finding the best design of a linear structure that yields the stiffest structure by minimizing the compliance. In linear analysis, the abnormal deformation is not a critical matter because the consequence of the analysis, i.e., deformation, is never further used.

On the other hand, in a nonlinear incremental analysis, the results from the previous load step are again used to proceed to the next load step [6].

Due to a large number of variables in this type of problems, it is necessary to do some simplification regarding the constraints and formulation. For example, in the Solid Isotropic Material with Penalization (SIMP) or “power-law approach”, introduced by Bendsøe [7] and then presented by Rozvany et al. [8], the material properties are assumed constant within each finite element used to discretize the design domain and the variables are the element relative densities. This approach is one of the most used methods in topology optimization.

Most of the research in TO focus on linear problems like linear elasticity and only a few research take into consideration non-linear elasticity. The first attempt to incorporate inelasticity into topology optimization was in 1995 made by Yuge and Kikuchi [9], who optimized the layout of frame structures subject to plastic deformation and used a direct sensitivity analysis. Maute et al. [10] extended the density-based topology optimization method to elastoplastic materials by utilizing a sensitivity analysis in which simplifications were made by neglecting several derivative terms. An alternative sensitivity analysis method was proposed by Kato et al. [11] for elastoplastic multiphase composite topology optimization and was also demonstrated to be accurate as well as efficient. However, this method is not general and is limited to specific individual cases.

1.2 Objectives

The main objective of this work is to develop a computational tool capable of solving topology optimization problems for materials with nonlinear behaviour. In order to do that it is essential to know the physics behind this material behaviour, its mathematical interpretation and the use of numerical methods to implement it using computational resources. The programming platform used was Matlab, once that is a very flexible tool with a lot of potential in matrix computations that is helpful in the realization of this work. To simulate a nonlinear material behaviour it is necessary to develop a function with the implementation of the Finite Element Method (FEM). Once the material behaviour is implemented is used a common methodology for topology optimization problems. This methodology consist in doing a Finite Element Analysis (FEA), compute the objective function and sensitivities, use a specific methods to solve the optimization problem (e.g. Optimality Criteria (OC), Sequential Linear Programming (SLP) or the Method of Moving Asymptotes (MMA)) and also the use of filtering techniques for the sensitivities. Having the computational tool developed, it can be used to solve different objectives for various boundary conditions and parameters.

1.3 Layout

This document is divided in three different parts. The present one is **Introduction and Backgroud**, where a theoretical framework is presented along with essential concepts to understand the basics and methods used for the realization of this assignment. It is divided in the following chapters:

- **Chapter 1** - Is introduced the problematic, the state of the art and also the main goals to achieve with its realization.

- **Chapter 2** - Basic concepts of the material behaviour are approached.
- **Chapter 3** - Some concepts of the finite element method are discussed.
- **Chapter 4** - Some types of structural optimization are presented, focusing on the topology optimization problem and its methodology.

In the second part, **Methodology and Implementation**, it is described the methods used and its implementation using Matlab, presenting all the practical work done in this assignment and the final structure of the computational tool developed.

- **Chapter 5** - The methodology used in this assignment is discussed.
- **Chapter 6** - Is presented some implementation details and the final structure of the developed tool.

Finally, in the last part, **Results and Discussion**, it is shown all the topologies resulting from the implementation of various problems using the final computational tool, the validation of the developed functions and the discussion of this results ending with some final remarks.

- **Chapter 7** - Validation of the developed functions and results obtain with the final computational tool.
- **Chapter 8** - Conclusion and some suggested future works.

Intentionally blank page.

Chapter 2

Material Behaviour

2.1 Plane Stress and Plane Strain

In two-dimensional structural problems there are two different states that can be considered, the plane stress assumption in which one of the dimensions of the body in analysis (the thickness) is much smaller than the others and is subjected to loads that generate dominant stresses in the plane Oxy ($\sigma_{zz} = 0$) and the plane strain assumption in which has to be considered contributions from the thickness ($\sigma_{zz} \neq 0$). In the plane strain assumption the strain perpendicular to the plane Oxy can be considered null ($\varepsilon_{zz} = 0$), in the plane stress assumption this strain is not zero ($\varepsilon_{zz} \neq 0$). In this last case the strain ε_{zz} has to be calculated using the stress field (table 2.1) [12]. In engineering the plane stress assumption is applicable in the analysis of plates subjected to in plane loads and thin membranes with negligible stiffness in bending and transverse shear. To analyse a gas pipeline, the strain variation in the length direction is very small and negligible when compared to its strain variation in radial direction (figure 2.1). The constitutive relation between the stress and the strain can be represented as

$$\boldsymbol{\sigma} = \mathbf{C} : \boldsymbol{\varepsilon}, \tag{2.1}$$

or using the Voigt notation, this relation can be written in the matrix form considering the Cartesian components of the stress and strain states

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \tau_{xy} \end{Bmatrix} = \mathbf{C} \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \gamma_{xy} \end{Bmatrix}, \tag{2.2}$$

where C represents the constitutive tensor and it is function of the material proprieties, the Young's modulus (E) and the Poisson's ratio (ν)(linear isotropic material). For the plane stress assumption the constitutive tensor is represented by

$$\mathbf{C} = \frac{E}{(1+\nu^2)} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{(1-\nu)}{2} \end{bmatrix}, \tag{2.3}$$

and for the plane strain assumption the constitutive tensor is stated as

$$\mathbf{C} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} (1-\nu) & \nu & 0 \\ \nu & (1-\nu) & 0 \\ 0 & 0 & \frac{(1-2\nu)}{2} \end{bmatrix}. \tag{2.4}$$

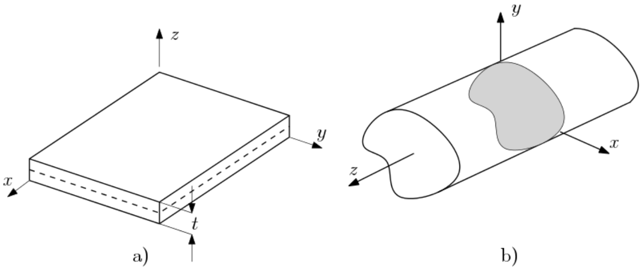

Figure 2.1: Representation of a) plane stress assumption in the plane Oxy and b) plane strain assumption in the plane Oxy.

Table 2.1: Differences between plane stress and plane strain assumption.

	Plane Stress	Plane Strain
Stress	$\sigma_{zz} = \tau_{xz} = \tau_{yz} = 0$	$\tau_{xz} = \tau_{yz} = 0$ $\sigma_{zz} \neq 0$
Strain	$\gamma_{xz} = \gamma_{yz} = 0$ $\varepsilon_{zz} \neq 0$	$\varepsilon_{zz} = \gamma_{xz} = \gamma_{yz} = 0$

2.2 Nonlinear behavior

In order explain the nonlinear behavior the plasticity can be used. The theory of plasticity is concerned with solids that, after being loaded may conserve a permanent (or plastic) deformation when unloaded. Materials whose behaviour can be described by this theory are called plastic (or rate-independent plastic) materials. In engineering, a vast number of materials, such as metal, rocks, concrete, clays, and soils may be modelled using the plastic theory under many circumstances of practical interest. The origins of the theory of plasticity can be traced back to the middle of the nineteenth century, and nowadays there are a lot of models that can mathematically describe the behaviour of plastic materials. To understand the phenomenological behaviour of this type of materials the uniaxial (one-dimensional) tension experiment with a metallic bar is commonly used. This experiment with ductile metals produces a stress-strain curve that is used to describe the material's behaviour (Figure 2.2). There the axial stress, σ, is plotted against the axial strain, ε, the bar is subject to a load that increases the axial stress from zero to a prescribed value, σ_0. After the bar is unloaded back to an unstressed point and then reloaded to a higher stress level, σ_1, this can be translated by the path $O_0Y_0Z_0O_1Y_1Z_1$ shown. The initial line segment O_0Y_0 is linear, meaning that if the bar is unloaded in the last point or before, it returns to O_0, this behavior is regarded as linear elastic. Beyond Y_0, the slope of the stress-strain curve changes dramatically and if the loading is reversed at, say, point Z_0, the bar returns to an unstressed state via path Z_0O_1. The new unstressed state O_1 differs from the original unstressed state O_0 and a permanent change in the bar is observed. This change represents the permanent (or plastic) axial strain ε^p. Reloading the bar again to a stress level σ_1 will follow the path $O_1Y_1Z_1$ where still the O_1Y_1 line segment is linear, similar to the initial elastic section O_0Y_0. Note that the points Y_0 and Y_1 are different. Thus, is essential to identify some proprieties of the uniaxial test:

1. The existence of an elastic domain or a range of stresses within which the behavior

of the material can be considered as purely elastic, without the evolution of permanent strains delimited by the yield stress. The portions O_0Y_0 and O_1Y_1 represent the elastic domain at two different states with the corresponding yield stresses Y_0 and Y_1.

2. Beyond the yield stress, the plastic yielding (or plastic flow) takes place and consequently evolution of plastic strains.

3. Accompanying the development of plastic strain evolution of the yield stress itself is observed (growth from Y_0 to Y_1). This phenomenon is known as hardening [13].

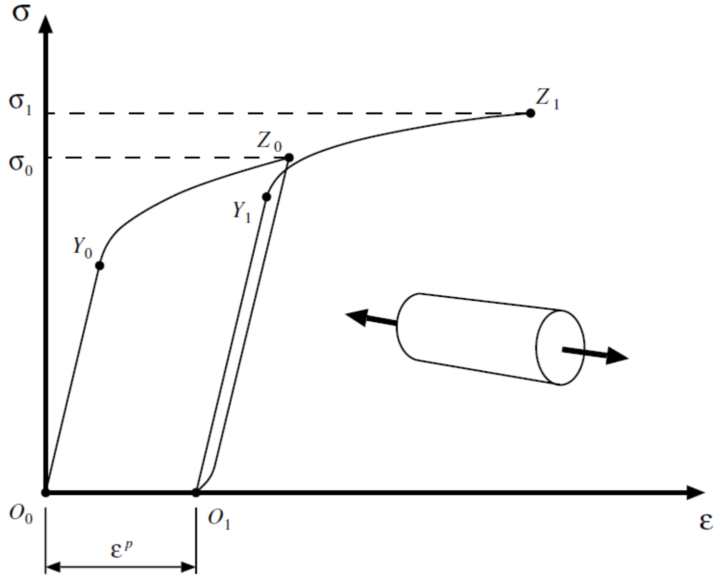

Figure 2.2: Uniaxial tension experiment with ductile metals. Adapted from [13].

Intentionally blank page.

Chapter 3

The Finite Element Method

Nowadays the Finite Element Method (FEM) is a widespread technique used in the industry and analysis in engineering, with applications not only in solid and structure analysis but also in fluid mechanics and heat transfer problems. This method appeared in the 50's/60's and was boosted with the developments of computers and mathematical programming that permitted a quick and efficient solution for algebraic equation systems [14]. The development of the FEM was impulsed, initially, by scientists and investigators of the Civil and Mechanical Engineering field. The application of the FEM in structural analysis began with linear elastic problems, and later the non-linear effects such as plasticity were included. Is important to emphasise that two major numerical approximations are necessary for the finite element solution:

1. *A time discretization of the underlying constitutive initial value problem*, where a numerical integration scheme is introduced to solve the initial value problem defined by the constitutive equations of the model that relate stresses to the history of deformations (for nonlinear analysis).

2. *A finite element discretization*, which comprises a standard finite element approximation of the virtual work statement where the domain of the body and the associated functional sets are replaced with finite-dimensional counterparts generated by finite element interpolation functions (figure 3.1) [13].

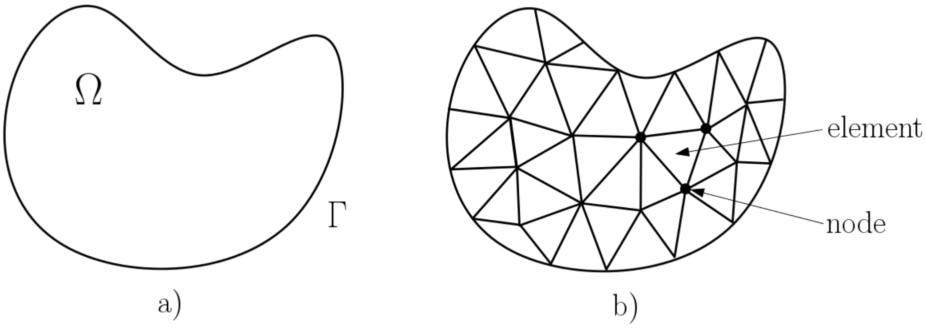

Figure 3.1: Discretization of the domain: a) Domain Ω, b) Discretization of Ω.

The finite elements can assume different geometries with some examples shown in figure 3.2. To solve one-dimensional problem line segment elements are used. In two-dimensional problems are generally used triangular or quadrilateral elements. In three-dimensional analyses are commonly used tetrahedrons, hexahedrons or pentahedrons, but there are many more configurations for the elements in this method [12]. The basis of the finite element method is to assign nodes to the elements and to assume that shape or interpolation functions can be determined to enable interpolation in order to obtain values of displacement or temperature, depending on the analysis type, at any point within the element in terms of nodal values. The shape functions are defined in the natural coordinate system $O\xi\eta$, and an important feature of the shape functions in the finite element method is that they take a value of 1 at their own node and zero at all others, so the sum of all shape functions is equal to unity [15].

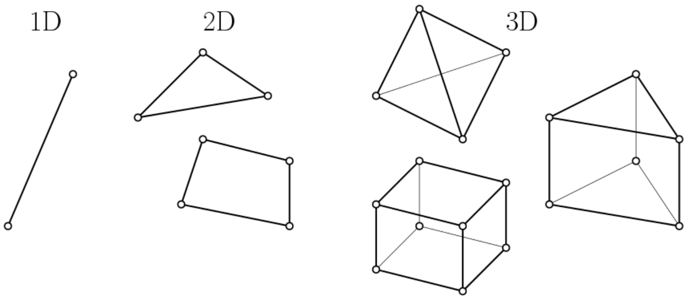

Figure 3.2: Some possible geometries for finite elements to one, two and three-dimensional problems.

3.1 Shape Functions and Jacobin Matrix

The shape functions used in the finite element method for Lagrange quadrilateral elements are based in Lagrange polynomials defined in the natural coordinate system $O\xi\eta$. The generic equations for unidimensional Lagrange polynomials in therms of the natural coordinates can be expressed as

$$N_i(\xi) = \prod_{j=1(j\neq i)}^{n_{\text{nodes}}} \frac{(\xi - \xi_i)}{\xi_i - \xi_j}, \tag{3.1}$$

$$N_i(\eta) = \prod_{j=1(j\neq i)}^{n_{\text{nodes}}} \frac{(\eta - \eta_i)}{\eta_i - \eta_j}, \tag{3.2}$$

with $\xi \in [-1, +1]$ and $\eta \in [-1, +1]$. For a isoparametric four node quadrilateral element the shape functions can be expressed by

$$\begin{cases} N_1(\xi,\eta) = \frac{1}{4}(1-\xi)(1-\eta) \\ N_2(\xi,\eta) = \frac{1}{4}(1+\xi)(1-\eta) \\ N_3(\xi,\eta) = \frac{1}{4}(1+\xi)(1+\eta) \\ N_4(\xi,\eta) = \frac{1}{4}(1-\xi)(1+\eta) \end{cases} . \tag{3.3}$$

In order to calculate de derivatives of the shape function, the chain rule is used

$$\begin{cases} \frac{\partial N_i(\xi,\eta)}{\partial \xi} = \frac{\partial N_i(\xi,\eta)}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial N_i(\xi,\eta)}{\partial y}\frac{\partial y}{\partial \xi} \\ \frac{\partial N_i(\xi,\eta)}{\partial \eta} = \frac{\partial N_i(\xi,\eta)}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial N_i(\xi,\eta)}{\partial y}\frac{\partial y}{\partial \eta} \end{cases}, \tag{3.4}$$

or, in the matrix form,

$$\begin{bmatrix} \frac{\partial N_i(\xi,\eta)}{\partial \xi} \\ \frac{\partial N_i(\xi,\eta)}{\partial \eta} \end{bmatrix} = \mathbf{J} \begin{bmatrix} \frac{\partial N_i(\xi,\eta)}{\partial x} \\ \frac{\partial N_i(\xi,\eta)}{\partial y} \end{bmatrix}, \tag{3.5}$$

where $\mathbf{J}$ is the jacobian matrix of the element responsible for the first order partial derivatives mapping from the global Oxy coordinate system to the natural $O\xi\eta$ coordinate system (Figure 3.3), i.e.

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}. \tag{3.6}$$

Finally, the inverse of the jacobian matrix is used to compute the derivatives present in the strain-displacement matrix

$$\begin{bmatrix} \frac{\partial N_i(\xi,\eta)}{\partial x} \\ \frac{\partial N_i(\xi,\eta)}{\partial y} \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \frac{\partial N_i(\xi,\eta)}{\partial \xi} \\ \frac{\partial N_i(\xi,\eta)}{\partial \eta} \end{bmatrix}. \tag{3.7}$$

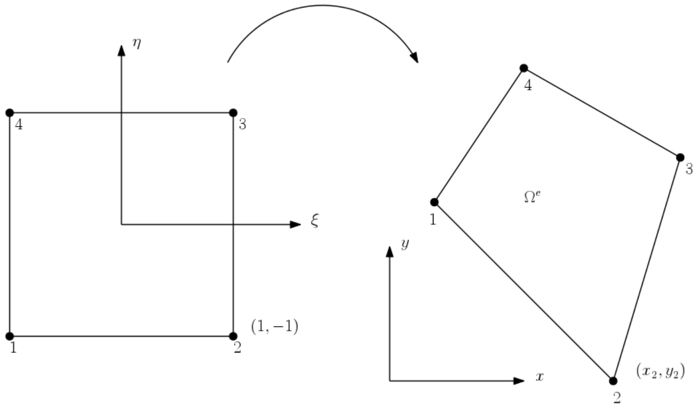

Figure 3.3: Transformation of isoparametric element from natural to cartesian coordinates.

3.2 Strain-Displacement relation

Consider a domain Ω in a bi-dimensional plane with the global axis x, y. The variables of the problem are the displacement, and the strain is obtained by deriving them, as

$$\mathbf{u}(x,y) = \begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix}, \tag{3.8}$$

and

$$\varepsilon(x,y) = \begin{bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \end{bmatrix}. \tag{3.9}$$

The domain is discretized in elements where Ω_e is defined for the element domain. The generic finite element with n nodes has $2n$ degrees of freedom that can be organized in the form

$$\mathbf{u}_e = \begin{bmatrix} u_1 & v_1 & u_2 & v_2 & ... & u_n & v_n \end{bmatrix}^{\mathrm{T}} \tag{3.10}$$

The relation between the displacements and the strain is expressed by

$$\boldsymbol{\varepsilon} = \mathbf{B}\mathbf{u}_e, \tag{3.11}$$

where $\mathbf{B}$ represents the strain-displacement matrix with dimension $3 \times 2n$, as

$$\mathbf{B} = \begin{bmatrix} \frac{\partial N_1^e}{\partial x} & 0 & \frac{\partial N_2^e}{\partial x} & 0 & ... & \frac{\partial N_n^e}{\partial x} & 0 \\ 0 & \frac{\partial N_1^e}{\partial y} & 0 & \frac{\partial N_2^e}{\partial y} & 0 & ... & \frac{\partial N_n^e}{\partial y} \\ \frac{\partial N_1^e}{\partial y} & \frac{\partial N_1^e}{\partial x} & \frac{\partial N_2^e}{\partial y} & \frac{\partial N_2^e}{\partial x} & ... & \frac{\partial N_n^e}{\partial y} & \frac{\partial N_n^e}{\partial x} \end{bmatrix} \tag{3.12}$$

This matrix contains the derivatives of the shape functions $N_i^e(i = 1, ..., n)$.

3.3 Stiffness Matrix and Load Vector

After calculating the $\mathbf{B}$ matrix is now possible to determine the stiffness of the element as

$$\mathbf{K}_e = \int_{\Omega_e} \mathbf{B}^{\mathrm{T}}\mathbf{C}\mathbf{B}\mathrm{d}\Omega_e. \tag{3.13}$$

This matrix has the dimension $(2n \times 2n)$ and the element volume can be expressed by

$$\mathrm{d}\Omega_e = h\mathrm{det}\mathbf{J}\mathrm{d}\xi\mathrm{d}\eta, \tag{3.14}$$

where h is the thickness and $\mathrm{det}\mathbf{J}$ is the determinant of the Jacobian. The integration can be done in terms of the natural coordinate system using four Gauss points (Figure 3.4) considering the weight of each integration point (Table 3.1). The element load vector can be expressed as

$$\mathbf{f}_e = \int_{\Omega_e} h\mathbf{N}^{\mathrm{T}}\mathbf{b}\mathrm{d}\Omega_e + \int_{\Gamma_e} h\mathbf{N}^{\mathrm{T}}\mathbf{t}\mathrm{d}\Gamma_e. \tag{3.15}$$

This components can be assembled in the global form and used to obtain the displacement vector trough the equilibrium equation stated in the optimization problem

$$\mathbf{K}\mathbf{u} = \mathbf{f}, \tag{3.16}$$

$$\mathbf{u} = \mathbf{K}^{-1}\mathbf{f}. \tag{3.17}$$

where $\mathbf{K}$ is the global stiffness matrix assembled using the element stiffness matrices $\mathbf{K}_e$ and $\mathbf{f}$ the global load vector.

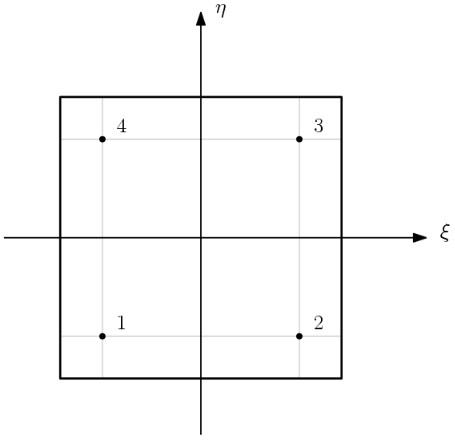

Figure 3.4: Quadrilateral finite element using 4 Gauss-Legendre integration points.

Table 3.1: Natural coordinates and weights for four Gauss integration points.

Point	ξ	η	Weight
1	-0.5773502...	-0.5773502...	1
2	0.5773502...	-0.5773502...	1
3	0.5773502...	0.5773502...	1
4	-0.5773502...	0.5773502...	1

3.4 Stress, Strain and Equivalent von Mises Stress Tensors

After solving the equilibrium equation (equation 3.16), it is possible to use the displacements and calculate the strain tensor using equation 3.11 and the stress by

$$\boldsymbol{\sigma} = \mathbf{C}\mathbf{B}\mathbf{u}^e = \mathbf{C}\boldsymbol{\varepsilon} \tag{3.18}$$

The stress tensor is composed by three components σ_{xx}, σ_{yy} and σ_{xy}. Using this tensor it is possible to calculate the equivalent stress using the von Mises criteria (figure 3.5)

$$\overline{\sigma}(\boldsymbol{\sigma}) = \sqrt{\frac{3}{2}\mathrm{s}(\boldsymbol{\sigma}) : \mathrm{s}(\boldsymbol{\sigma})}, \tag{3.19}$$

with $\mathbf{s} = \mathbf{s}(\boldsymbol{\sigma})$ being the deviatoric stress tensor

$$\mathrm{s}(\boldsymbol{\sigma}) = \boldsymbol{\sigma} - \frac{1}{3}tr(\boldsymbol{\sigma})\mathbf{I}. \tag{3.20}$$

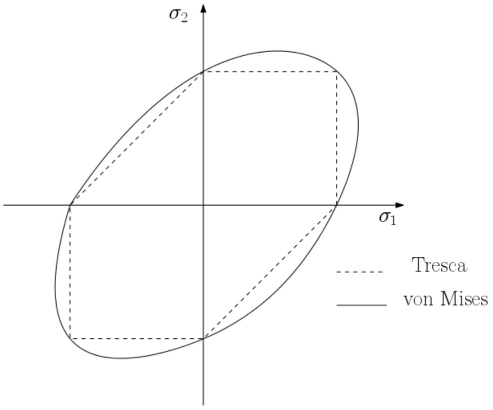

Figure 3.5: Representation of the vonMises and Tresca criteria.

Chapter 4

Topology Optimization

4.1 Structural Optimization

Structural Optimization (SO) is an old thematic in engineering, and the primary objective is to obtain the best relation between characteristics such as stiffness, weight or cost. However, the first scientific evaluation of SO problems was in 1870 by Maxwell [3] and later in 1904 by Michell [4]. The field of application was civil engineering where Maxwell studied ways to minimize the material used in a bridge project. To achieve that he analysed the stress field and suggested that the optimal structure is composed of discrete elements oriented in the principal directions of the stress field. This way the elements only have nominal stresses without shear component. Later Michell also studied optimal structures with minimum material. An example of these structures is shown in Figure 4.1. These structures are commonly used as an analytical benchmark for frame structures in topology optimization.

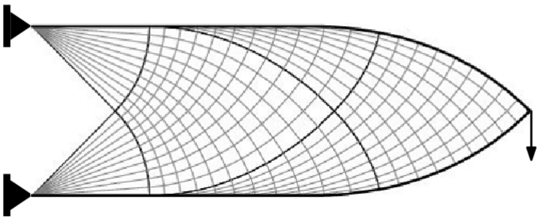

Figure 4.1: Example of a Michell frame structure (Michell-cantilever). Adapted from [16].

Structural optimization is commonly divided into three fundamental approaches: (i) sizing optimization, (ii) shape optimization and (iii) topology optimization. These types of SO are illustrated in Figure 4.2. In sizing optimization, the design variables are essential parameters to the structural behaviour. These parameters are related to the definition of the transversal section, dimensions, constitutive parameters, and others. The shape optimization allows to change the geometry of the structure in order to obtain the optimal configuration. The design variables are point coordinates and relevant parameters to the geometric definition. One of the typical approach it is the manipulation of the curves coefficients such as splines or Non-Uniform Rational Basis Spline (NURBS). Finally, in topology optimization, the objective function is optimized manipulating the topology of the structure inside the prescribed domain. This type of SO is concerned with the distribution of the material inside the domain in order to minimize/maximize the objective function respecting the primary constraints. The design variables are densities that vary between different phases of the material or void, assuming values of 1 or 0 respectively [1].

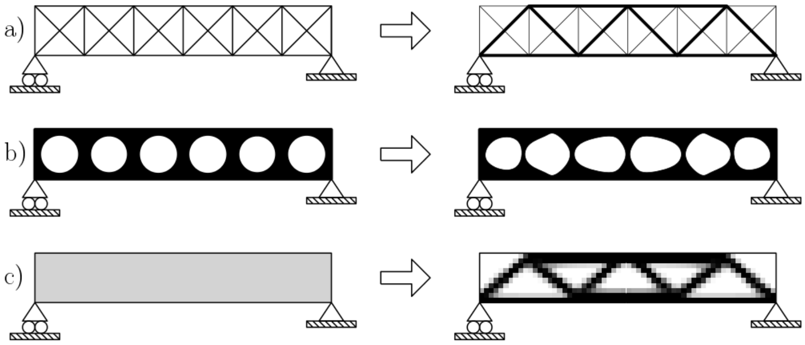

Figure 4.2: Three types of structural optimization. a) Sizing optimization, b) shape optimization and c) topology optimization. At the left hand side are shown the initial problems and at the right the respective solutions. Adapted from [17]

4.2 Topology Optimization for Minimum Compliance Design

In order to maximize the stiffness or minimize the compliance of a structure for the linear elastic problem it is used the strain energy as a measure of the compliance.

$$S = \frac{1}{2}\int_{\Omega} C_{ijkl}(\mathbf{x})\varepsilon_{ij}(\mathbf{u})\varepsilon_{kl}(\mathbf{u})\mathrm{d}\Omega, \tag{4.1}$$

where $\mathbf{u}$ represents the displacement field at the equilibrium, ε_{ij} the strains at the equilibrium and C_{ijkl} the elastic tensor of the material in $\mathbf{x} \in \Omega$.

Other way is to use the compliance, stated as

$$W = \int_{\Omega} b_i u_i \mathrm{d}\Omega + \int_{\Gamma} t_i u_i \mathrm{d}\Gamma. \tag{4.2}$$

b_i and t_i are the external forces by unit of volume and area, respectively. These equations can be used to define the total potential energy of the system (P) as,

$$P = S - W. \tag{4.3}$$

The potential energy is minimized by the displacement field $\mathbf{u}$ at the equilibrium ($\delta P = 0$).

The most common form is to use the external work as a measure of the compliance. Thus the optimization problem can be stated as

$$\begin{aligned} \text{minimize} \quad & \int_{\Omega} \mathbf{bu}\mathrm{d}\Omega + \int_{\Gamma} \mathbf{tu}\mathrm{d}\Gamma, \\ \text{subject to} \quad & \mathbf{C} = \mathbf{C}_{\mathrm{adm}}, \qquad\qquad , \\ & \mathbf{Ku} = \mathbf{f}, \end{aligned} \tag{4.4}$$

or, in the most practical form,

$$\begin{aligned} \text{minimize} \quad & \int_{\Omega} \mathbf{f}^{\mathrm{T}}\mathbf{u}\mathrm{d}\Omega = \int_{\Omega} \mathbf{u}^{\mathrm{T}}\mathbf{K}\mathbf{u}\mathrm{d}\Omega, \\ \text{subject to} \quad & \mathbf{C} = \mathbf{C}_{\mathrm{adm}}, \\ & \mathbf{K}\mathbf{u} = \mathbf{f}. \end{aligned} , \tag{4.5}$$

Here $\mathbf{u}$ and $\mathbf{f}$ are the displacement and load vectors, respectively. The stiffness matrix $\mathbf{K}$ depends on $\mathbf{C}_e$ in element $e = 1, ..., N$.

$$\mathbf{K} = \sum_{e=1}^{N} \mathbf{K}_e(\mathbf{C}_e), \tag{4.6}$$

where $\mathbf{K}_e$ is the element stiffness matrix.

4.3 Solid Isotropic Material with Penalization (SIMP)

In 1988 Bendsøe and Kikuchi [5] presented topology optimization as a computational tool applied to continuum structures, where a so-called microstructure or homogenization based approach was used, based on studies of the existence of solutions. The homogenization has the disadvantage that the determination and evaluation of optimal microstructures and their orientations are cumbersome if not unresolved (for non-compliance problems). However, the homogenization approach to topology optimization is still relevant in the sense that it can provide bounds on the theoretical performance of structures. One year later an alternative approach to topology optimization, the so-called "power-law approach" or SIMP approach was introduced by Bendsøe [7] and later presented by Rozvany et al. 1992 [8], being today one of the most used approaches in topology optimization. In this approach, material properties are assumed constant within each element used to discretize the design domain, and the variables are the element relative densities. The material properties are modelled as the relative material density raised to some power times the material properties of a solid material. This approach has been criticized since it was argued that no physical material exists with properties described by the power-law interpolation. However, in 1999 Bendsøe and Sigmund proved that the power-law approach is physically permissible as long as simple conditions on the power are satisfied (e.g., $p \geq 3$ for Poisson's ratio equal to $\frac{1}{3}$) [18].

Thus it can be now introduced the penalization parameter p and the topology optimization problem rewrite in terms of the element density ρ_e as

$$\begin{aligned} \underset{\rho_e}{\text{minimize}} \quad & \mathrm{c}(\rho_e) = \mathbf{f}^{\mathrm{T}}\mathbf{u} \\ \text{subject to} \quad & \sum_{e=1}^{N} \rho_e^p \mathbf{K}_e \mathbf{u} = \mathbf{f}, \\ & \sum_{e=1}^{N} v_e \rho_e \leq V, \\ & 0 < \rho_{\min} \leq \rho_e \leq 1, \end{aligned} \tag{4.7}$$

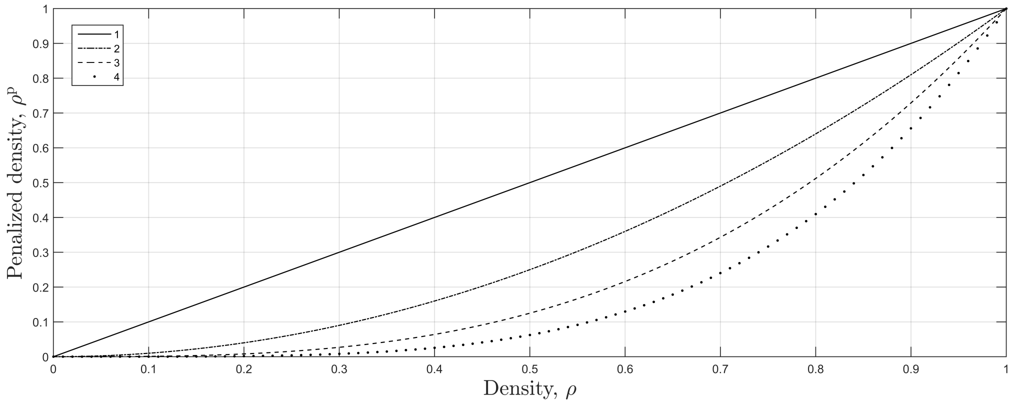

Figure 4.3: Representation of the power-law approach (SIMP) for different penalties.

where v_e represent the element volume and V the prescribed volume fraction. Note that the element densities can not be zero, to avoid singularity, so it is defined a minimum density value, $\rho_{\min}$.

4.4 Sensitivity Analysis

The analysis of the objective function's derivatives is critical to the solution of the optimization problem. The derivatives in a certain point permits to evaluate the behaviour of the function in its proximity. This type of analysis is called sensitivity analysis. In topology design is usual to work with a moderate number of constraints, so the most effective method for calculating derivatives is to use the adjoint method, where the derivatives of the displacement are not calculated explicitly. Thus the objective function $c(\rho)$ can be rewrite by adding the zero function:

$$c(\rho) = \mathbf{f}^{\mathrm{T}}\mathbf{u} - \tilde{\mathbf{u}}^{\mathrm{T}}(\mathbf{K}\mathbf{u} - \mathbf{f}), \tag{4.8}$$

where $\tilde{\mathbf{u}}$ is any arbitrary, but fixed real vector. The zero function is chosen using the equilibrium $\mathbf{K}\mathbf{u} = \mathbf{f}$. By deriving the new objective function is obtained the follow derivative

$$\frac{\partial c}{\partial \rho_e} = (\mathbf{f}^{\mathrm{T}} - \tilde{\mathbf{u}}^{\mathrm{T}}\mathbf{K})\frac{\partial \mathbf{u}}{\partial \rho_e} - \tilde{\mathbf{u}}^{\mathrm{T}}\frac{\partial \mathbf{K}}{\partial \rho_e}\mathbf{u}. \tag{4.9}$$

This can in turn be written as

$$\frac{\partial c}{\partial \rho_e} = -\tilde{\mathbf{u}}^{\mathrm{T}}\frac{\partial \mathbf{K}}{\partial \rho_e}\mathbf{u}, \tag{4.10}$$

when $\tilde{\mathbf{u}}$ satisfies the adjoint equation:

$$\mathbf{f}^{\mathrm{T}} - \tilde{\mathbf{u}}^{\mathrm{T}}\mathbf{K} = 0. \tag{4.11}$$

Knowing that

$$\mathbf{f}^{\mathrm{T}} = \mathbf{u}^{\mathrm{T}}\mathbf{K}, \tag{4.12}$$

it is verified that in this case is directly obtained that $\tilde{\mathbf{u}} = \mathbf{u}$. Finally using the $\mathbf{K}$ derivative

$$\frac{\partial \mathbf{K}}{\partial \rho_e} = p(\rho_e)^{p-1}\mathbf{K}_e, \tag{4.13}$$

the final derivative of the objective function for the minimum compliance problem is

$$\frac{\partial c}{\partial \rho_e} = -p(\rho_e)^{p-1}\mathbf{u}^{\mathrm{T}}\mathbf{K}_e\mathbf{u}. \tag{4.14}$$

It is interesting to notice that the derivative is “localized” once that it uses information of each element and is negative for all elements wich confirms that additional material in any element decreases compliance [17].

4.5 Mesh-independency Filter

In topology optimization many restriction methods can be used in order to reduce the mesh dependency, *checkerboard* phenomena (figure 4.4), for a better post-processing of the final geometries and to ensure existence of solutions. A great example of that is filter techniques [Sigmund and Petersson 1998] that can be divided in filters that are applied directly on the problem variable and filters that use gradients or the objective function sensitivities. The mesh-independency filter works by modifying the element sensitivities as follow:

$$\widehat{\frac{\partial c}{\partial \rho_e}} = \frac{1}{\rho_e \displaystyle\sum_{f=1}^{N} \hat{H}_f} \sum_{f=1}^{N} \hat{H}_f \rho_f \frac{\partial c}{\partial \rho_f}, \tag{4.15}$$

where $\hat{H}_f$ is the convolution operator and is written as

$$\hat{H}_f = r_{\min} - \operatorname{dist}(e, f), \tag{4.16}$$

$$\{f \in N \mid \operatorname{dist}(e,f) \leq r_{\min}\}, \quad e = 1, \ldots, N.$$

The filter is applied to the element e and to the elements that have their centers in a distance delimited by the filter radius $r_{\min}$. The convolution operator is zero outside the filter area and decays linearly with the distance from element f. Thus the sensitivity used in the optimization problem will be the modified (4.15) instead of the original one (4.14).

4.6 Optimality Criteria Method

The optimality criteria is a method that uses a fix point iteration scheme. Is a very efficient method for problems with reduced number of restrictions compared with the number of design variables. The update scheme has the objective of updating the variable respecting optimum criteria established with the Karush-Kuhn-Tucker or KKT conditions. The generic problem can be stated as

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}), \\ &\text{subject to} && g_j(\mathbf{x}) = 0, \quad j = 1, ..., p, \end{aligned} \tag{4.17}$$

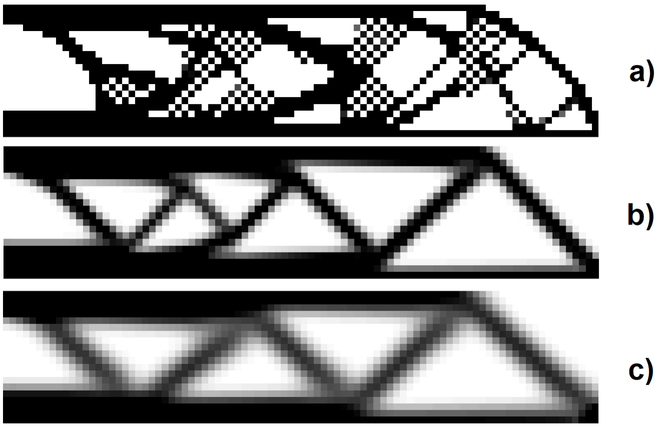

Figure 4.4: Influence of the filter for different radius: a) $r_{\text{min}} = 1$, b) $r_{\text{min}} = 1.5$ and c) $r_{\text{min}} = 3$.

$$\nabla\mathcal{L}(\mathbf{x}^*) = \nabla f(\mathbf{x}^*) + \boldsymbol{\lambda}^{*\text{T}}\nabla\mathbf{g}(\mathbf{x}^*) = 0 \tag{4.18}$$

where $\boldsymbol{\lambda}$ is a Lagrange multiplier vector to the restriction $\mathbf{g}$. Considering a problem with one restriction only the equation 4.18 can be rewrite as

$$-\frac{\partial f}{\partial x_i} = \lambda\frac{\partial g}{\partial x_i}, \qquad i = 1, ..., n, \tag{4.19}$$

from where the Lagrange multiplier can be deduced as

$$\lambda = -\frac{\frac{\partial f}{\partial x_i}}{\frac{\partial g}{\partial x_i}}. \tag{4.20}$$

Thus the fix point iteration scheme can be presented as

$$x_i^{(k+1)} = x_i^{(k)}\left(-\frac{\frac{\partial f}{\partial x_i}}{\lambda\frac{\partial g}{\partial x_i}}\right)^{\frac{1}{\gamma}}, \tag{4.21}$$

where γ is an iteration step control parameter.

In terms of the topology optimization problem the optimality criteria can be stated using the heuristic updating scheme

$$\rho_e^{\text{new}} = \begin{cases} \max(\rho_{\text{min}}, \rho_e - m) \\ \quad \text{if} \quad \rho_e B_e^\eta \leq \max(\rho_{\text{min}}, \rho_e - m), \\ \\ \rho_e B_e^\eta \\ \quad \text{if} \quad \max(\rho_{\text{min}}, \rho_e - m) < \rho_e B_e^\eta < \min(1, \rho_e + m), \\ \\ \min(1, \rho_e + m) \\ \quad \text{if} \quad \min(1, \rho_e + m) \leq \rho_e B_e^\eta, \end{cases} \tag{4.22}$$

where m is a positive move-limit and η is a numerical damping coefficient. The optimality condition to be used in the fixed point iteration (equation 4.21) is defined as

$$B_e = \frac{-\frac{\partial c}{\partial \rho_e}}{\lambda \frac{\partial V}{\partial \rho_e}}. \tag{4.23}$$

In this procedure the density is increased when $B_e > 1$ and decreased when $B_e < 1$. The optimal point is when the optimality condition equals to unity ($B_e = 1$) for all density variables. In this problem the Lagrange multiplier satisfies the volume restriction. The function $g(\lambda) = V(\rho(\lambda)) - f_{\mathrm{v}}\overline{V}$ has a monotonous decreasing dependency with λ, due to that the Lagrangian multiplier can be found by a bi-sectioning algorithm to do an iterative update of λ. This iterative cycle is represented in the figure 4.5 and converge when $g(\rho) = 0$. When the optimality condition equals one for all values ($\rho^{new} \approx \rho$) the optimization problem has converged [18], [1].

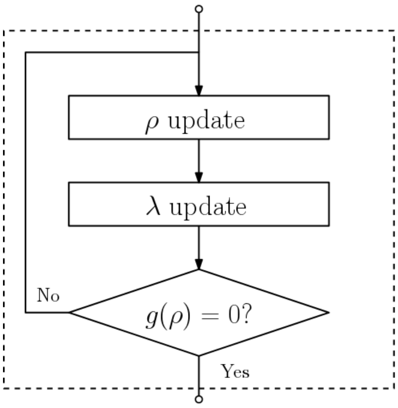

Figure 4.5: Iterative cycle in the OC. Adapted from [1]

Intentionally blank page.

Part II

Methodology and Implementation

Chapter 5

Methodology

5.1 Non-linear Material

In order to define a non-linear material behaviour two simple approaches were applied. The first approach consists in make a variation of the Young's modulus with some prescribed parameters. These parameters have the intuit of simulate the plastic phenomenon. The first one is related with a load limit (f_{y}) that defines at which percentage of the maximum load ($f_{\max}$) this phenomenon occurs, its calculated by

$$\mathrm{H1} = \frac{f_{\mathrm{y}}}{f_{\max}}. \tag{5.1}$$

The second one a parameter to the variation of the Young's modulus in each load step ($\mathrm{H2} = \alpha_{\mathrm{E}}$). In this approach the proprieties of the material are assumed to be constant in all elements. Thus the Young's modulus is updated in each load step by

$$E_{i+1} = \alpha_{\mathrm{E}} E_i, \tag{5.2}$$

so for a α_{E} of 0.5 in each load step the Young's modulus will decay to an half of the previous one.

The second approach is similar to the first with the differences that the "plastic" phenomenon is not related with the applied load but with the stress levels of each individual element and the material proprieties are not constant for all elements. To achieve that a vector containing the element's elastic modulus was created and only the elements with stress levels greater than a defined yield stress have their proprieties changed.

5.2 Objective Function and Sensitivities

In this work two different objectives were considered. Minimization of the end-compliance and minimization of the complementary work. The main difference between these two objectives is that the minimization of end-compliance only takes into account the last values of displacement and stiffness matrices ignoring the history behind the last load increment and since only requires the finding of one equilibrium no incremental procedure is necessary [17]. On the other hand the minimization of the complementary work requires an incremental procedure since that the objective is calculated using all the information

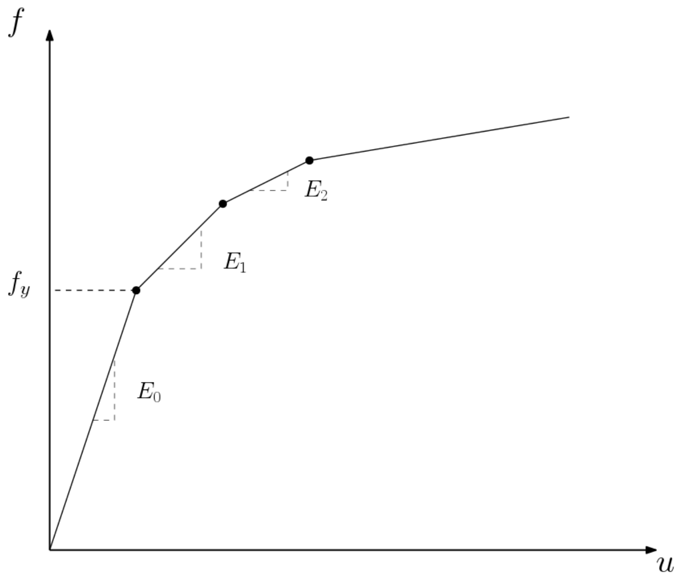

Figure 5.1: Force displacement curve obtained with the Young's modulus variation.

of all load increments. The objective function and sensitivities of the minimization of end-compliance were already discussed (Equations 4.7 and 4.14).

For the minimization of complementary work, using the trapezoidal method for numerical integration, the complementary work of the external forces can be calculated as

$$c(\rho) = W^{\mathrm{C}} \approx \Delta\mathbf{f}^{\mathrm{T}} \left[\frac{1}{2}\mathbf{u}(\mathbf{f}_0) + \sum_{i=1}^{n-1} \mathbf{u}(\mathbf{f}_i) + \frac{1}{2}\mathbf{u}(\mathbf{f}_n) \right], \tag{5.3}$$

where n is the number of load increments and $\Delta\mathbf{f}$ the size of the increments determined by

$$\Delta\mathbf{f} = \frac{(\mathbf{f}_n - \mathbf{f}_0)}{n}, \tag{5.4}$$

where $\mathbf{f}_0$ and $\mathbf{f}_n$ are the zero and maximum load vectors, respectively. The sensitivity analysis for the complementary work also uses the adjoint method as described before, resulting in the following derivative

$$\frac{\partial c}{\partial \rho_e} = \Delta\mathbf{f}^{\mathrm{T}} \left[\frac{1}{2}\boldsymbol{\lambda}_0^{\mathrm{T}} \frac{\partial \mathbf{r}_0}{\partial \rho_e} + \sum_{i=1}^{n-1} \boldsymbol{\lambda}_i^{\mathrm{T}} \frac{\partial \mathbf{r}_i}{\partial \rho_e} + \frac{1}{2}\boldsymbol{\lambda}_n^{\mathrm{T}} \frac{\partial \mathbf{r}_n}{\partial \rho_e} \right], \tag{5.5}$$

where $\mathbf{r}$ is the residual defined as the error in obtaining the equilibrium that corresponds to the difference between the external and internal forces

$$\mathbf{r}(\mathbf{u}) = \mathbf{f} - \int_V \mathbf{B}^{\mathrm{T}} \boldsymbol{\sigma} \mathrm{d}V. \tag{5.6}$$

This finite element equilibrium can be found using the iterative Newton-Raphson method, but for a multi-linear approach as the used in this work the partial derivative of the residual is $-\frac{\partial \mathbf{K}}{\partial \rho_e}\mathbf{u}$ and the adjoint variable $\boldsymbol{\lambda}$ equal to the original displacement vector $\mathbf{u}$. Thus the sensitivity for the minimization of complementary work is

$$\frac{\partial c}{\partial \rho_e} = -\Delta\mathbf{f}^{\mathrm{T}} p\rho_e^{p-1}\left[\frac{1}{2}\mathbf{u}_0^{\mathrm{T}}\mathbf{K}_0\mathbf{u}_0 + \sum_{i=1}^{n-1}\mathbf{u}_i^{\mathrm{T}}\mathbf{K}_i\mathbf{u}_i + \frac{1}{2}\mathbf{u}_n^{\mathrm{T}}\mathbf{K}_n\mathbf{u}_n\right]. \tag{5.7}$$

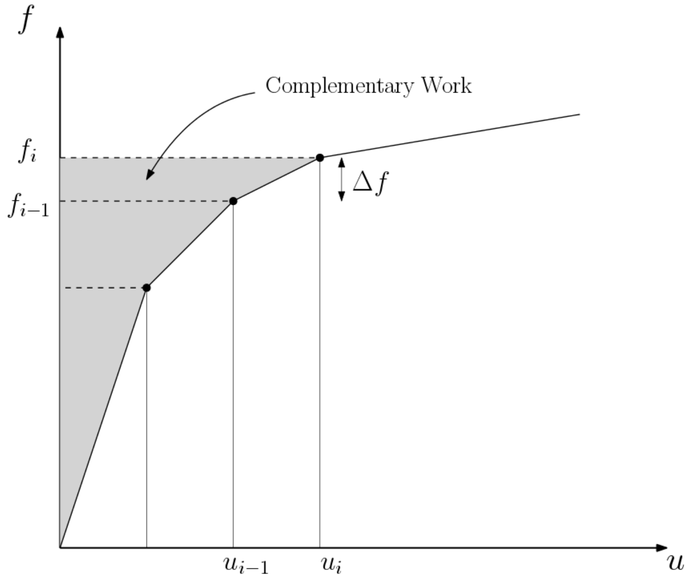

Figure 5.2: Representation of the complementary work.

Intentionally blank page.

Chapter 6

Software Arquitecture and Implementation

6.1 Implementation Details

6.1.1 Mesh Generation

The discretization of the initial domain it was implemented with a function that generates a mesh with quadrilateral elements. This function permits to define the domain dimensions and the number of elements in the $x - axis$ and $y - axis$. The outputs are the variables **node**, the nodal coordinate matrix ($n_{nodes} \times 2$), which contains the nodes coordinates and **element**, the element connectivity matrix ($n_{elem} \times 4$), that contains the connectivity between nodes of all elements. The numbering of the elements start from the bottom left corner and end at the top right corner, same as the software ABAQUS. The connectivity of the nodes is done in counter-clockwise fashion to avoid negative Jacobian's and consequently a singular stiffness matrix.

$$\mathbf{node} = \begin{bmatrix} 0.0 & 0.0 \\ 1.0 & 0.0 \\ 0.0 & 1.0 \\ 1.0 & 1.0 \end{bmatrix} \tag{6.1}$$

$$\mathbf{element} = \begin{bmatrix} 1 & 2 & 4 & 3 \end{bmatrix} \tag{6.2}$$

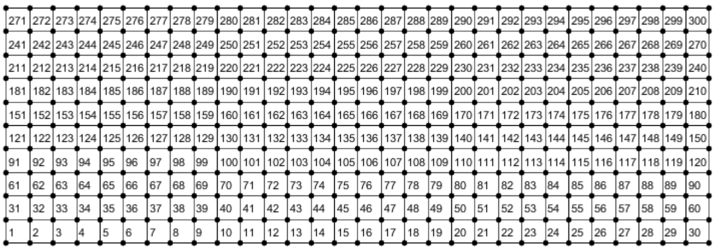

Figure 6.1: Mesh generated with RectangularMesh function.

6.1.2 The Constitutive Tensor

For the constitutive tensor was implemented for the two assumptions discussed before, the plane stress and plane strain assumptions. This function has as input arguments the Young modulus E, the Poisson ration ν, a variable **state** that is a string that contains the chosen assumption (*plane-stress* or *plane-strain*), but also the variable used in the topology problem, the density **x**, and the penalization used in the SIMP approach, **p**. Also has the number of elements, **numelem**, in order to access to the densities of each element. The variable and the penalization is applied to the constitutive tensor giving in this way the output of this function that is an hyper-matrix with dimension ($ntens \times ntens \times numelem$), where $ntens$ is the size of stress and strain component array (= 3 in 2D analysis).

$$\mathbf{C} = \rho_e^p \mathbf{C}_e \tag{6.3}$$

In this way all the constitutive matrix for all elements are stored in one variable.

6.1.3 Global Stiffness Matrix Assemble

The global constitutive tensor will be than used as an input to compute the global stiffness matrix **K** through this function. This function also has as input arguments the nodal coordinate matrix, **node**, and the connectivity matrix, **element**. After computing the **B** matrix using the methodology described in chapter 5 the element stiffness is computed using equation 3.13 for all four integration points. These matrices are then assembled in the global stiffness matrix. In order to do that the connectivity matrix is used to create a vector with the degrees of freedom of the corresponding element, **eldof**. Thus for an element with the connectivity [1 2 4 3] is obtained the following degrees of freedom

$$\mathbf{eldof} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 7 \\ 8 \\ 5 \\ 6 \end{bmatrix}, \tag{6.4}$$

6.1.4 The Finite Element Function

To solve the finite element problem is used the function *FEM.m*. In this function are established boundary conditions for 3 different problems, determining which degrees of freedom are active (**activeDof**) and also the node where the load is applied. After begins the iterative cycle where is added the load increment to the load in each iteration. In this cycle is calculated the constitutive tensor then used to determine the stiffness matrix and solve the equilibrium equation ($\mathbf{Ku} = \mathbf{f}$) using only the active degrees of freedom. With the resulting displacements the stress, strain and equivalent von Mises stress are calculated for each element. For the first approach after the load reach a percentage of the maximum load, defined by an hardening parameter the Young's modulus is changed in accord to the second hardening parameter. In the second approach the Young's modulus variation occurs only for elements with stress levels greater than the defined “yield stress”. Thus a scheme of this function it is presented in Figure 6.2.

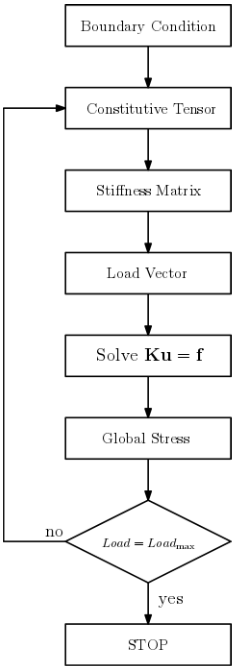

Figure 6.2: Scheme of the *FEM.m* function.

6.1.5 Element Numbering

The main function is and adaptation of “*A 99 line topology optimization code written in Matlab*” by O.Sigmund [18] with a finite element formulation. Since the element numbering in the Sigmund’s mesh is different a rearrangement was necessary in the design variable, **x**, and in the sensitivities vector, **dc**. One of the Matlab’s advantages is that permits an easy solution for this problem by its flexibility in reshaping vector and matrices.

```
X = flip(x',2);

dc = flip(reshape(dce,[nelx,nely])',1);
```

6.2 Final Computational Tool

The main function is composed by 5 functions. The input arguments are the number of elements in x, **nelx**, the number of elements in y, **nely**, the domain dimensions, **L** and **B**, the volume fraction, **volfrac**, this can be considered the domain proprieties. For the material proprieties is given the Young’s modulus, **E**, the Poisson’s ration, **nu**, the “hardening” parameters, **Hparams** and the penalization, **penal**. The last inputs are the filter radius, **rmin**, the maximum load, **fmax**, the load increment size, **step**, the assumption to use (plane-strain or plane-stress), **state**(*string*), the boundary condition,

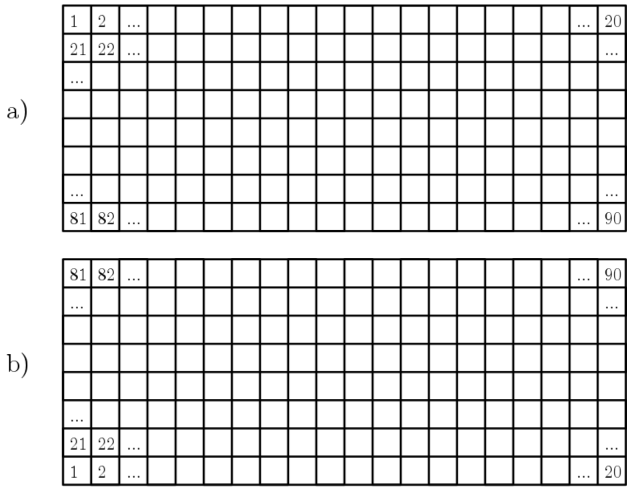

Figure 6.3: Difference in the element numbering between the Sigmund mesh and the generated mesh.

problem and finally, the objective function to be use (end-compliance or complementary-work), **objective**(*string*). The function can be run through the command window as for example

```
>> TopOpt(60,20,60,20,0.5,70000,0.3,[40  0.5],3,1.5,20,5,'plane-stress',4,'
    complementary-work')
```

The algorithm structure is:

- Construct a finite element mesh for the given domain dimensions and number of elements;
- Make the initial design with an homogeneous distribution of material (gives the selected volume fraction value for all elements);
- Start the iterative part of the algorithm (**loop**= 0, **change**=1);
- For the initial design, compute by the finite element method the resulting displacements and element stiffness matrices;
- Compute the objective function and sensitivities of the design;
- Modify the sensitivities using filtering techniques;
- Compute the update of the density variable based on the optimality criteria method
- If there is only a marginal improvement over the last design (**change**$\leq$ 0.01), stop the iterations.

The **change** variable is responsible for the convergence of the iteration cycle, when necessary condition of optimality are satisfied the design of the previous iteration is almost equal to the new one and then the cycle stop. The scheme of the final computational tool is shown in Figure 6.4.

```
change = max(max(abs(x-xold)));
```

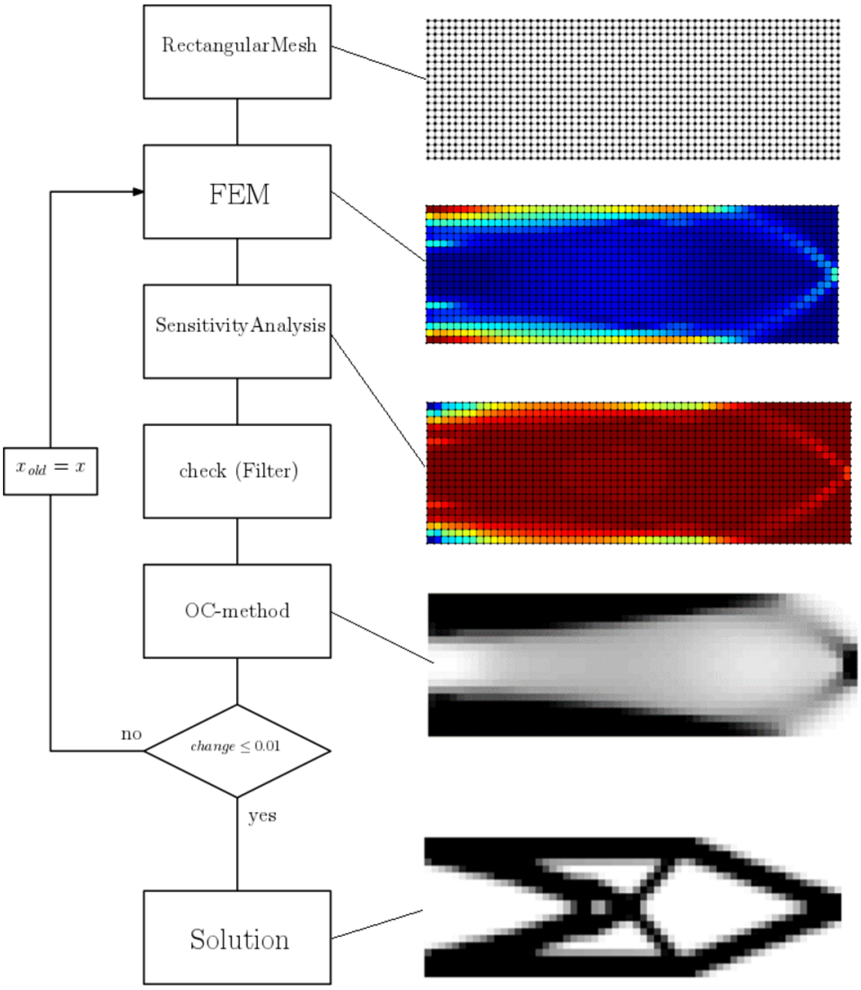

Figure 6.4: Scheme of the developed computational tool.

Intentionally blank page.

Part III

Results and Discussion

Chapter 7

Validation and Numerical Examples

In the developed computational tool 4 different problems are programmed: (1) cantilever beam with a load in the upper right corner, (2) cantilever beam with a load in the middle of the free end, (3) MBB-beam and (4) beam with both ends fixed (Figure 7.1). Problem number 3 corresponds to the MBB-beam problem (Figure 7.2) with a symmetry boundary condition in order to simplify the problem.

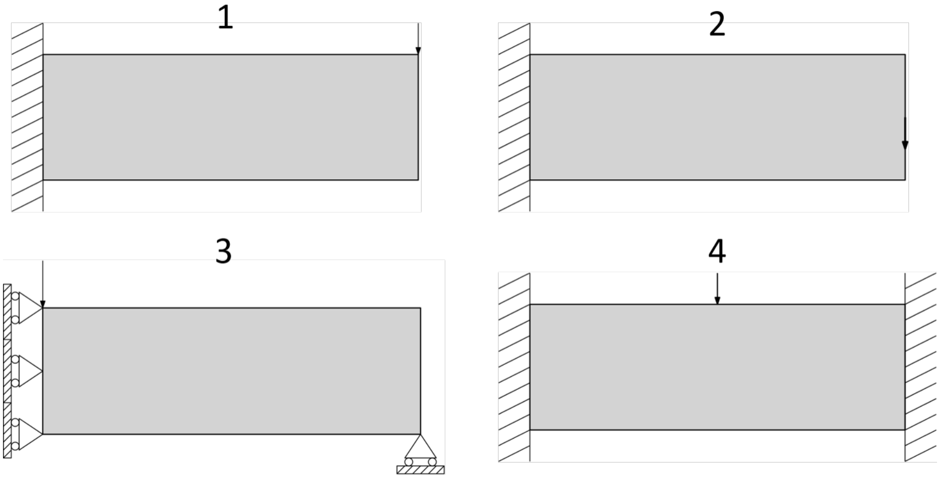

Figure 7.1: Different problems analysed with the developed computational tool.

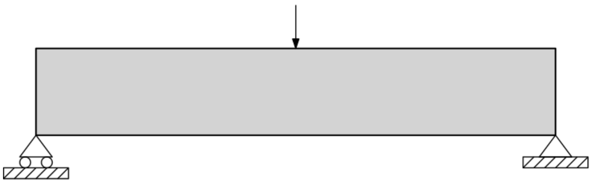

Figure 7.2: MBB-beam without the symmetry boundary condition.

7.1 Validation

For a good solution of the optimization problem it is essential to have a solid sensitivity analysis that is dependent of a good Finite Element Analysis (FEA). Thus, in order to validate the FEM function the numerical simulation software Abaqus was used. In this software the problem number 1 described before for the data presented in Table 7.1 was simulated. The graphical results (Figure 7.3) and the maximum values (Table 7.2) are similar, with the major difference in stress values. The Abaqus notation S stands for stress, E for strain and U for the displacement, where 1 and 2 are the x and y components, respectively. Another way to validate the displacement values was done by comparing them with the displacement obtained using the 99 line topology optimization code by Ole Sigmund [18] for problem 3 and the values are similar.

Table 7.1: Properties of the validation FEA.

Material properties		
Young's modulus	(E)	70 [GPa]
Poisson's ratio	(ν)	0.3
Domain dimension		
Length	(L)	30 [mm]
Width	(B)	10 [mm]
Number of elements	(nelx × nely)	30×10
Analysis conditions		
Boundary condition		Encastre (left side)
Load	$(f_{\max})$	40 [N]

Table 7.2: Comparison of the maximum values of the FEA using Abaqus and the developed finite element function.

	S11 [MPa]	S22 [MPa]	E11	E22	U1 [mm]	U2 [mm]
Abaqus	85.7300	25.7200	1.1100E-03	2.7000E-04	0.0168	-0.0690
FEM.m	78.4546	19.7449	8.0000E-04	3.2500E-04	0.0153	-0.0628

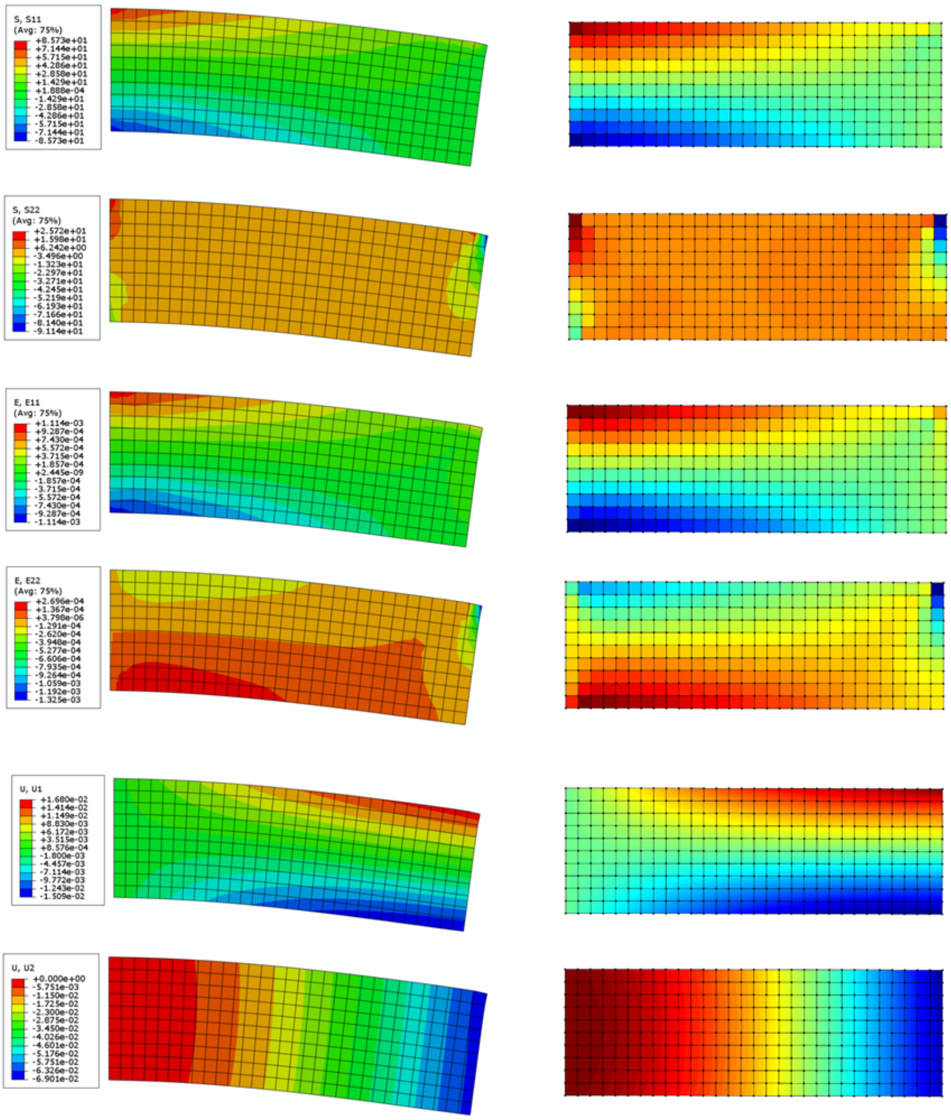

Figure 7.3: Comparison of the images for the FEA using Abaqus and the developed finite element function.

7.2 First Approach

7.2.1 Plane Strain vs Plane Stress

Using the first approach and the complementary work as objective function, the designs for the different problems were compared for both described assumptions, plane-stress and plane-strain. For all the results the von Mises equivalent stress, the sensitivities and the final designs were analyzed. The inputs used are presented in Table 7.3. In this analysis it was verified that the topologies for both assumptions do not have much variation. The only topology that has a small change is for the problem 3 (MBB-beam) in the lower right corner. It can be also noticed that the stress and sensitivities absolute values are higher were is defined the boundary condition and were the load is applied (Figures 7.4, 7.6, 7.8 and 7.10). Another interesting analysis is about the computational cost for the different problems and assumptions. It was verified that the plane-stress assumption has lower cost than the plane-strain assumption, and in terms of the defined problems the most expensive is problem 1 and the less expensive is the problem 4 (Table 7.4).

Table 7.3: Values used in the developed computational tool.

Material properties	Input argument	Values
Young's modulus	(E)	70 [GPa]
Poisson's ratio	(nu)	0.3
Young's modulus variation	(dE)	0.5
Domain dimension		
Length	(L)	60 [mm]
Width	(B)	20 [mm]
Number of elements	(nelx × nely)	60×20
FEM conditions		
Load	(fmax)	40 [N]
Load limit	(fy)	24 [N]
Load steps	(step)	5
Optimization conditions		
Volume fraction	(volfrac)	0.5
Penalization (SIMP)	(penal)	3
Filter radius	(rmin)	1.5

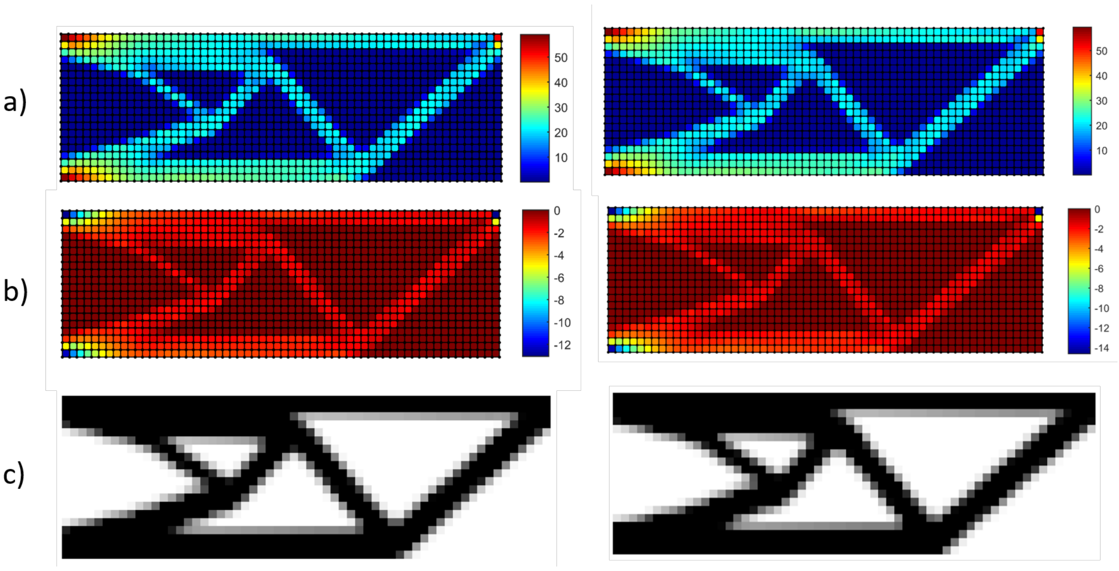

Figure 7.4: a) von Mises stress, b) sensitivity analysis and c) final design to the plane strain (left) and plane stress (right) assumptions for problem 1.

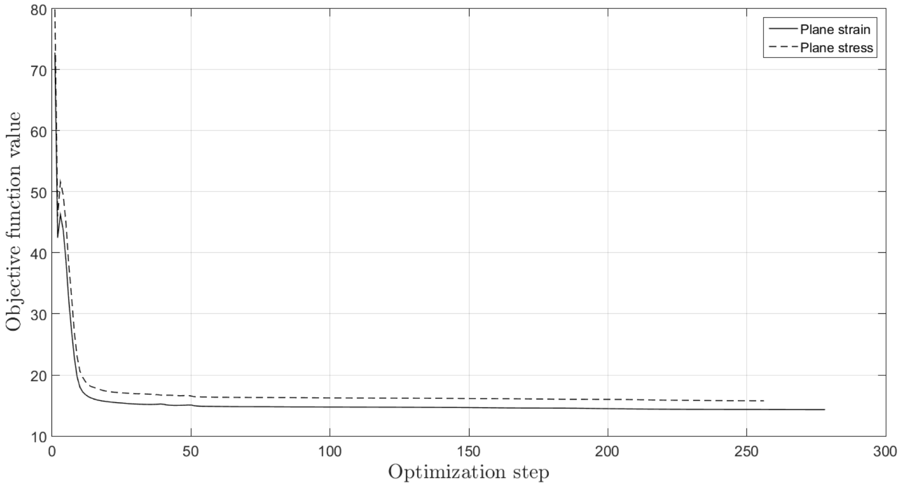

Figure 7.5: Evolution of the objective function value for problem 1.

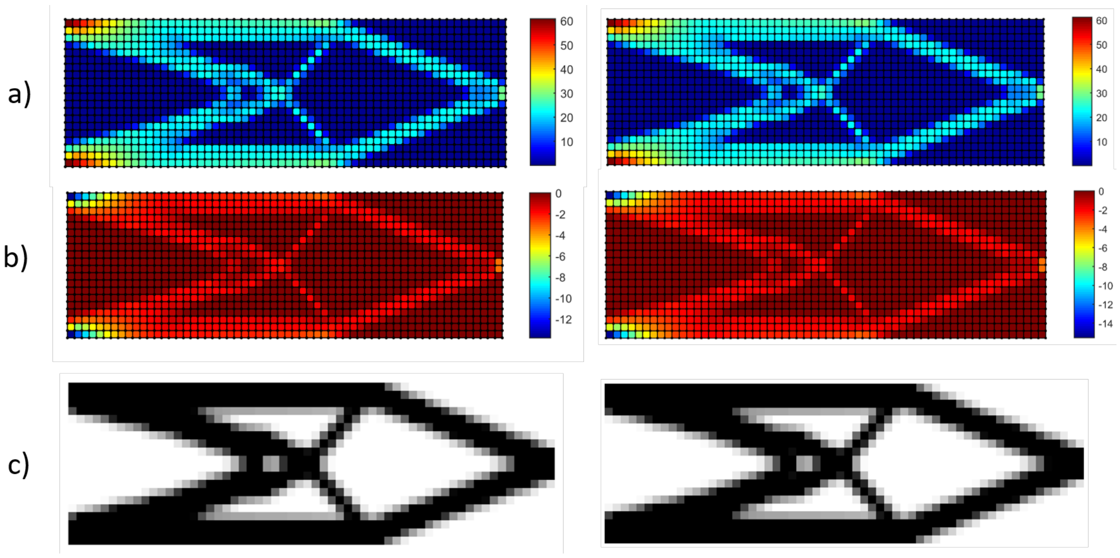

Figure 7.6: a) von Mises stress, b) sensitivity analysis and c) final design to the plane strain (left) and plane stress (right) assumptions for problem 2.

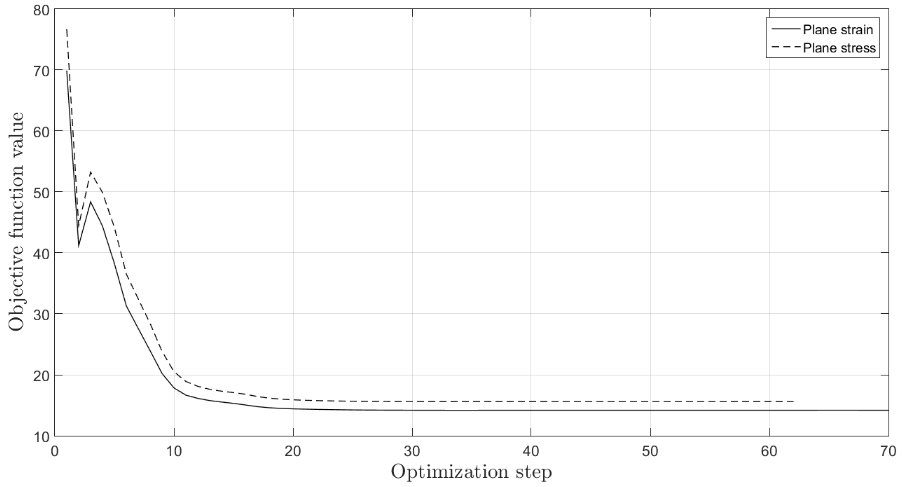

Figure 7.7: Evolution of the objective function value for problem 2.

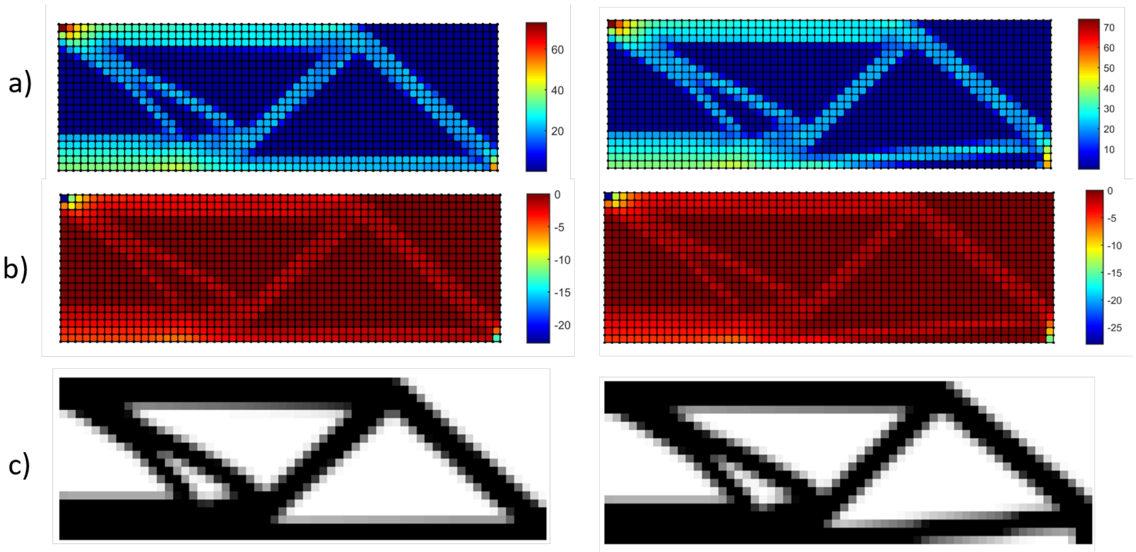

Figure 7.8: a) von Mises stress, b) sensitivity analysis and c) final design to the plane strain (left) and plane stress (right) assumptions for problem 3.

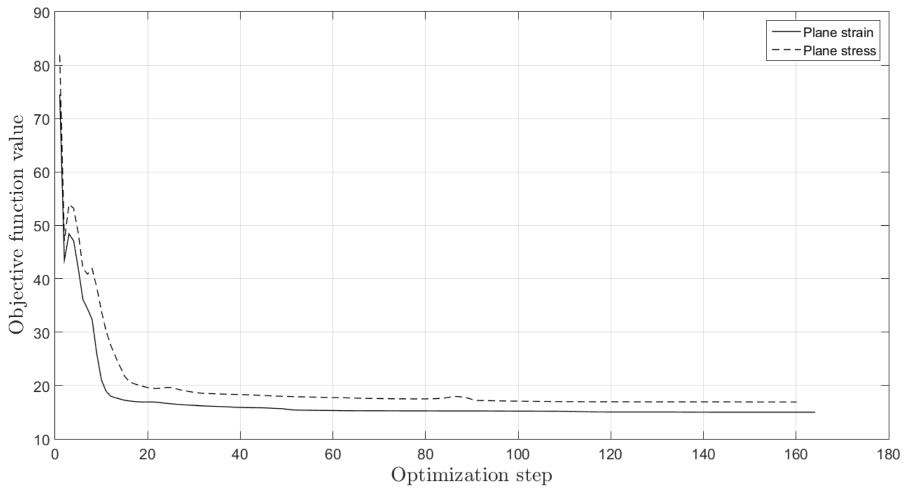

Figure 7.9: Evolution of the objective function value for problem 3.

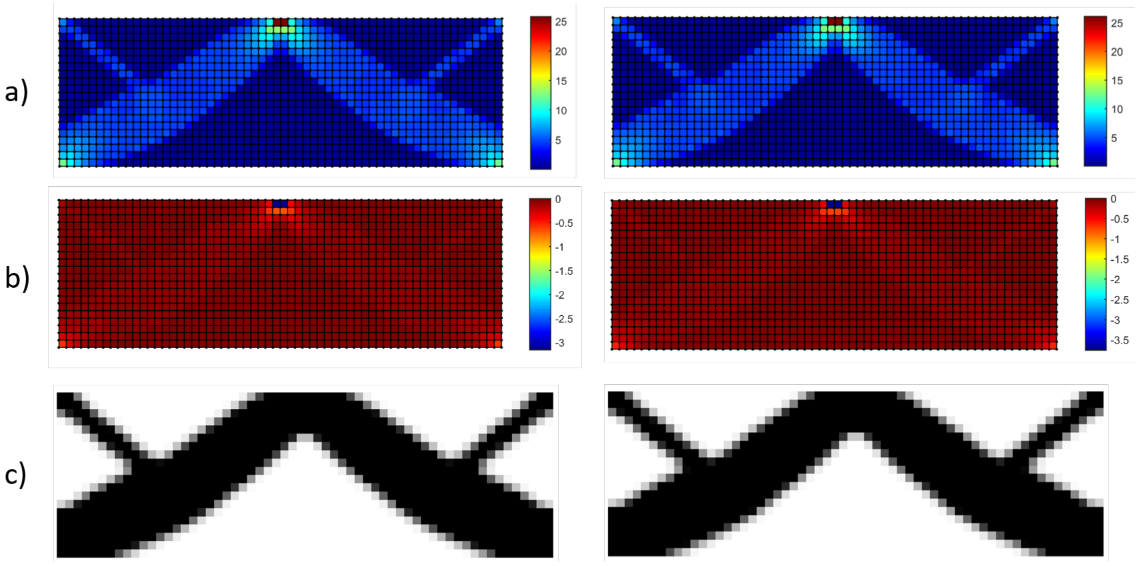

Figure 7.10: a) von Mises stress, b) sensitivity analysis and c) final design to the plane strain (left) and plane stress (right) assumptions for problem 4.

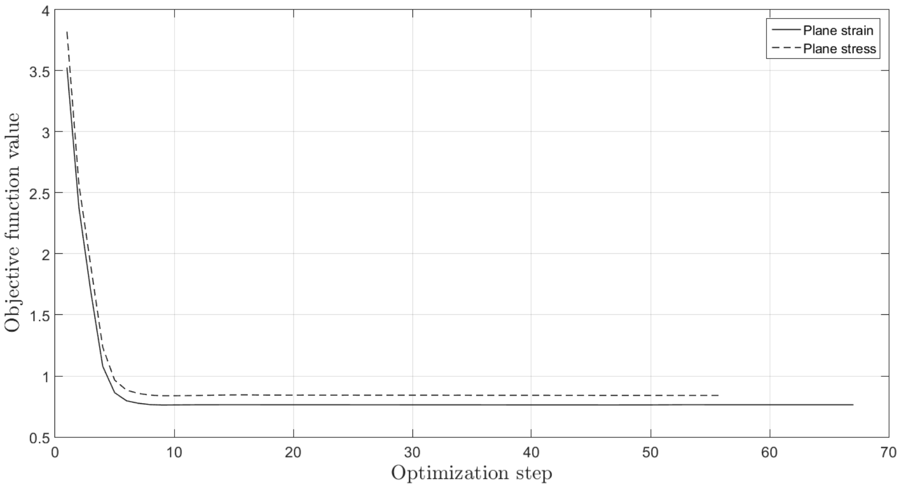

Figure 7.11: Evolution of the objective function value for problem 4.

Table 7.4: Values of the von Mises stress, complementary work and number of iterations for plane stress and plane strain.

Problem	State	von Mises [MPa]	W^{C} [J]	Iterations
1	P-Strain	59.0177	14.3186	278
	P-Stress	59.4058	15.7701	256
2	P-Strain	60.7946	14.1921	70
	P-Stress	61.1743	15.5912	62
3	P-Strain	72.8979	15.0188	164
	P-Stress	73.9134	16.9297	160
4	P-Strain	25.7749	0.7641	67
	P-Stress	26.082	0.8404	56

7.2.2 Load Variation for Complementary Work

After comparing both assumptions, it was selected the problem 2 with plane-stress assumption and was made a variation of the load in order to analyse the evolution of the von Mises stress, complementary work values and final configurations. This study was made using the same values of the previous analysis (Table 7.3), changing only the load values. It was verified that the von Mises equivalent stress has a linear relation with the load as expected (Figure 7.14). In contrast, the evolution of the objective function value has a quadratic evolution when compared with the load variation. This can be explained by the relations between load-displacements and displacements-complementary work. The increase of the load also reflects an increase in displacements once that this physical measures present a linear dependence by the stiffness (Equation 3.16), this can be seen in Figure 7.12. In terms of the complementary work, it can be verified in its equation (Equation 5.3) that the displacements have a double contribution, which is equivalent to have $\mathbf{Ku}^2$. The final design configuration does not present changes, neither does the objective function evolution, and to explain that a study of the evolution of sensitivities was made.

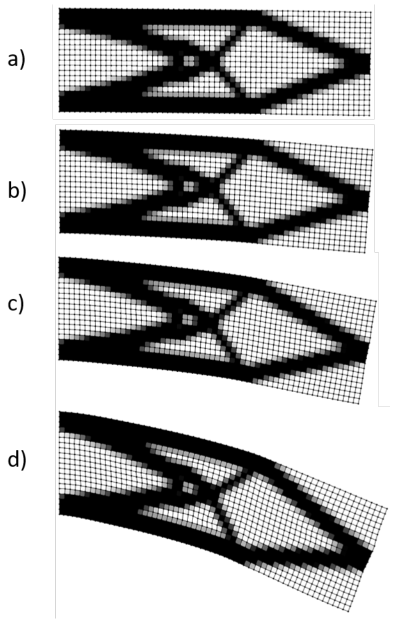

Figure 7.12: Deformed configurations of the topologies optimized for a) 40 N, b) 45 N, c) 50 N and d) 55 N.

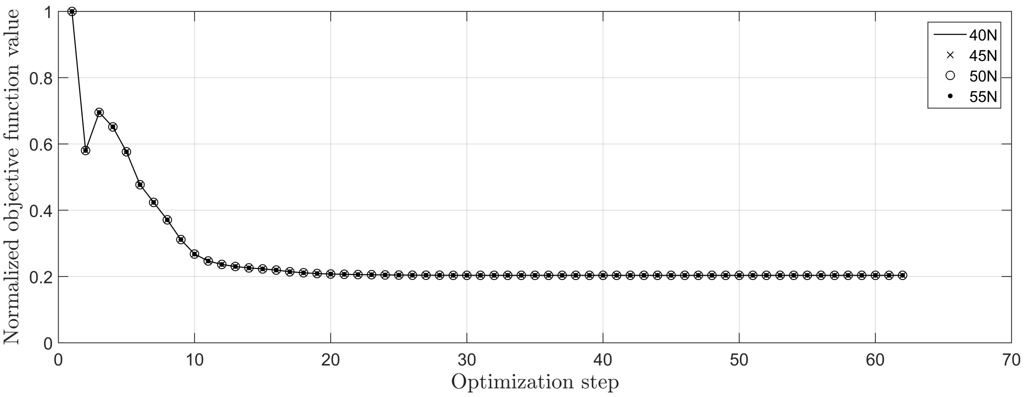

Figure 7.13: Evolution of the normalized objective function values for 40 N, 45 N, 50 N and 55 N load values.

Table 7.5: Values of the von Mises stress and complementary work for different loads.

Load [N]	von Mises [MPa]	W^{C}[J]
20	30.5804	0.6509
25	38.2329	1.3365
30	45.8791	2.9812
35	53.5280	6.8191
40	61.1743	15.5912
45	68.8211	35.3287
50	76.4679	79.1894
55	84.1146	175.6833

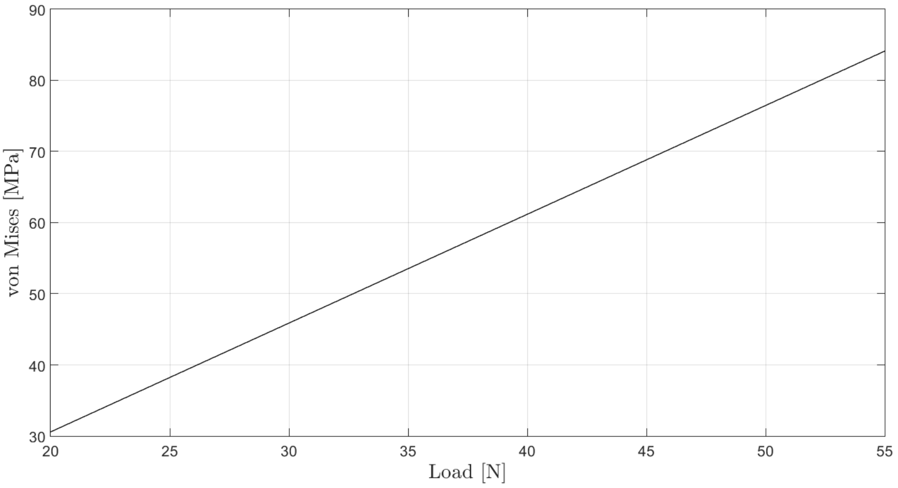

Figure 7.14: Evolution of the von Mises equivalent stress with load variation.

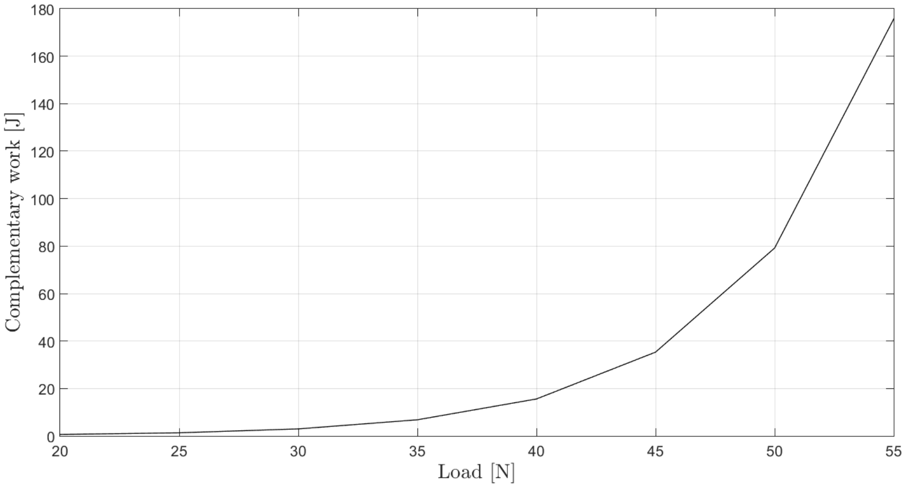

Figure 7.15: Evolution of the complementary work with load variation.

7.2.3 Sensitivity Analysis

In order to understand why the final configuration do not present significant changes, a sensitivity analysis was made. It was analyzed the evolution of the derivative's values for three cases: elastic material behaviour, multi-linear behaviour and bilinear behaviour (Figures 7.16, 7.17 and 7.18 respectively). For the elasticity it was applied a load of 24 N and for the other two cases the applied load was 40 N changing only the Young's modulus variation. From Figure 7.19 it can be verified that the evolution of the sensitivity for different optimization steps does not change, which means that the same problem is being solved but for different parameters that consequently result in different solution values. This can also be seen trough the normalized evolution of the objective function values in Figure 7.13.

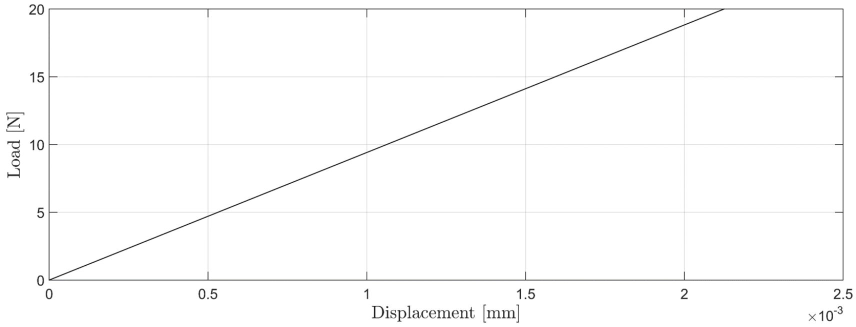

Figure 7.16: Load displacement curve for elastic behaviour (24 N).

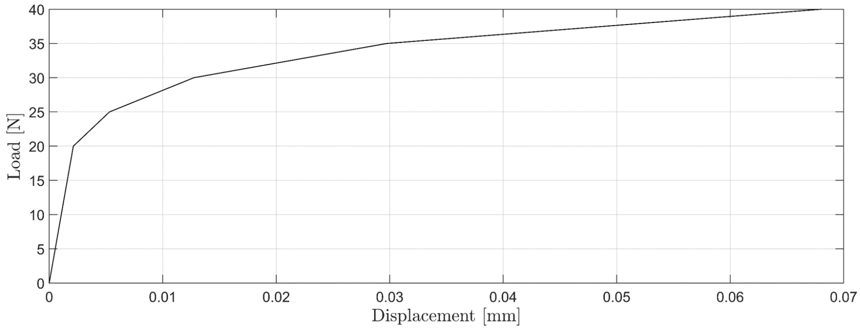

Figure 7.17: Load displacement curve for multi-linear behaviour (40 N).

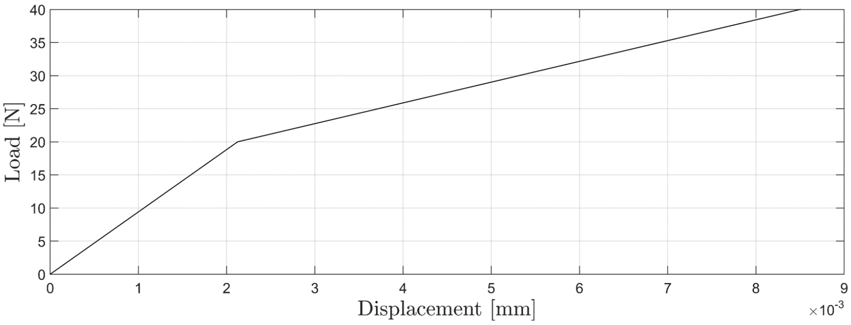

Figure 7.18: Load displacement curve for bilinear behaviour (40 N).

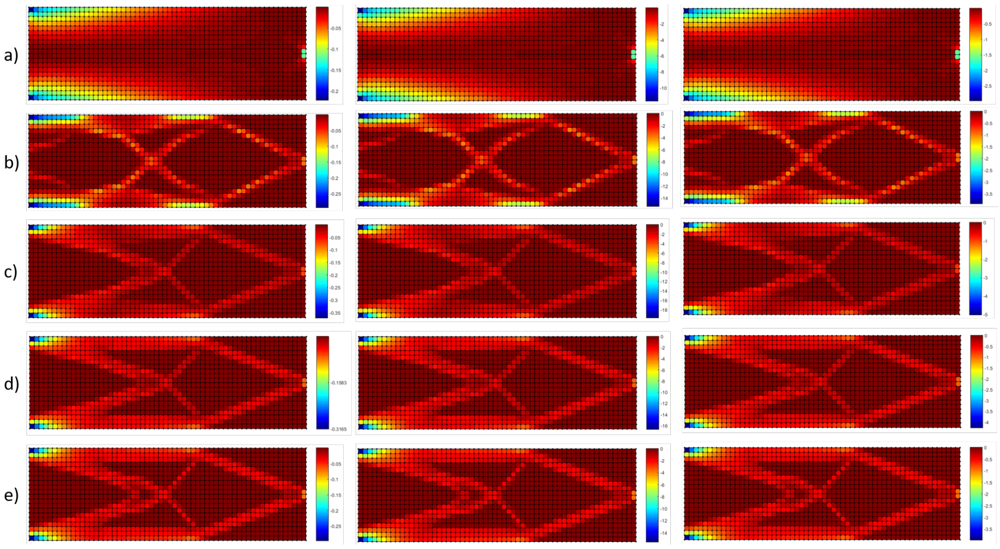

Figure 7.19: Evolution of sensitivities for elastic, multi-linear and bilinear behaviour respectively. a) first optimization step, b) 10 optimization steps, c) 20 optimization steps, d) 30 optimization steps and e) 62 (last) optimization steps.

7.3 Second Approach

Using the second approach, in which the elastic modulus is changed only for the yielded elements, it was solved the optimization problem with load variation. In this approach it can be verified that the final configuration changes in contrast to the first approach. An interesting result was that with the convergence of the optimization problem the final design evolves in a way that tends to minimize the number of elements that have reached the defined yield stress. Thus is equivalent to say that the algorithm tends to minimize elements with higher stress levels and also larger displacements, which makes sense once that the objective depends on the displacement values. In Figure 7.20 it is possible to see the evolution of the element's state (yielded or elastic), the von Mises equivalent stress and the sensitivities for different optimization steps using problem 2 with a load of 55 N and a yield stress of 40 MPa. The next study was to analyse the evolution of the solutions for the different problems with different loads.

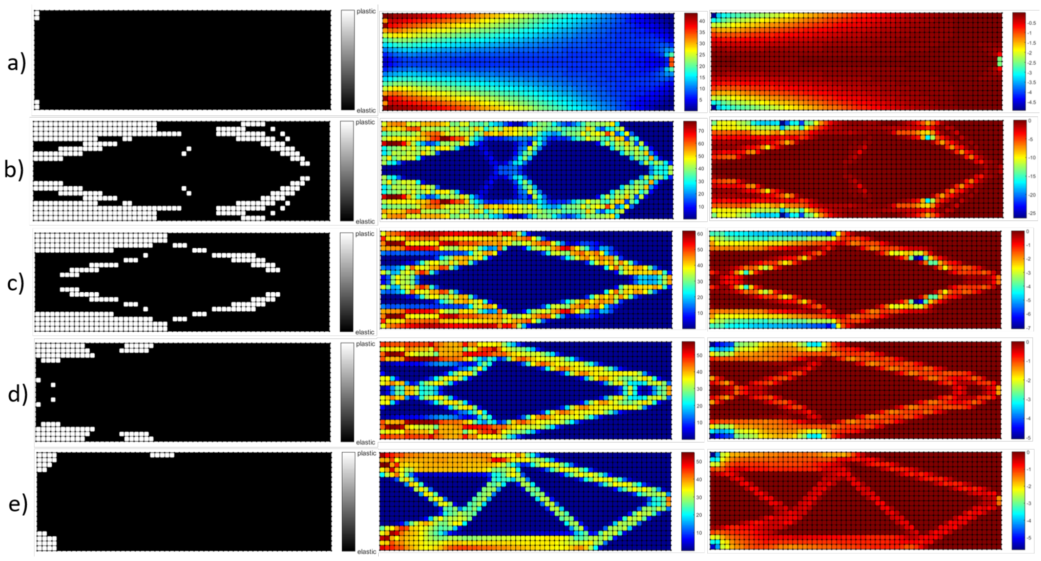

Figure 7.20: Element state (left), von Mises stress (middle) and local sensitivities (right) for: a) first optimization step, b) 10 optimization steps, c) 20 optimization steps, d) 30 optimization steps and e) last optimization step.

7.3.1 Load Variation and Final Topologies for Complementary Work

For the first boundary condition the optimization problem was solved for the load values of 35 N, 40 N, 45 N and 50 N. In this case it is not verified a big variation of the final designs and with the increase of the load the computational cost also increases. The evolution of the objective functions for the different loads can be seen in Figure 7.22. Analysing the second boundary condition it is verified that for a load of 55 N the final design presents a significant change and in this case the number of iterations increased a lot compared with the other loads. It is possible to see in Figure 7.24 that for this load value the evolution of the objective function value has a large oscillation. The evolution of the topologies in problem number 3 is also significant with the load variation (Figure 7.25). In this case, the problem with more iterations was the one with 35 N of load value, stabilizing for the next two loads (40 N and 45 N). Finally, for the problem 4 it was considered a yield stress of 20 MPa in order to reach the yield phenomenon with less load values once that this boundary condition generates a more stable structure. In this case there is not a significant variation of the final designs and the objective function values are again lower than the other problems.

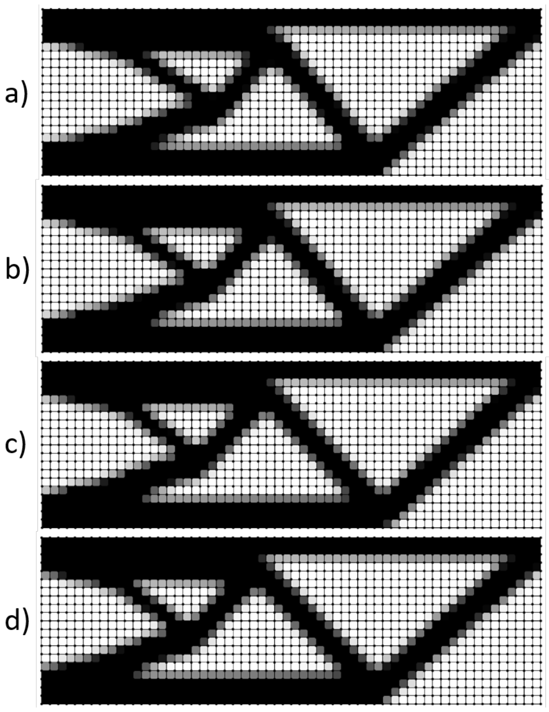

Figure 7.21: Solutions for problem 1 with load values of: a) 35 N, b) 40 N, c) 45 N and d) 50 N.

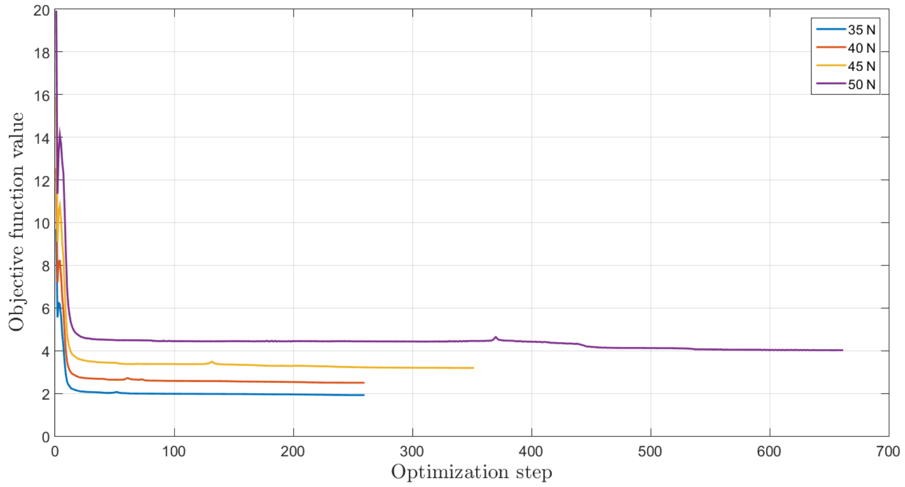

Figure 7.22: Evolution of the objective function values for the different load values in problem 1.

Table 7.6: Numerical results for different loads for problem 1.

Load	von Mises [MPa]	W^{C}[J]	Iterations
35	46.2654	1.9190	259
40	52.4706	2.4966	259
45	58.5900	3.1905	351
50	65.7391	4.0255	661

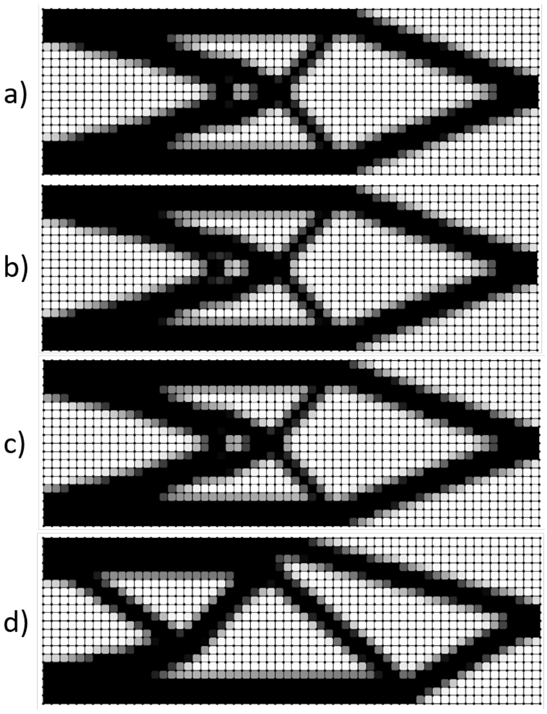

Figure 7.23: Solutions for problem 2 with load values of: a) 40 N, b) 45 N, c) 50 N and d) 55 N.

Table 7.7: Numerical results for different loads for problem 2.

Load	von Mises [MPa]	W^{C}[J]	Iterations
40	51.3967	2.4852	73
45	54.5835	3.1626	74
50	51.8558	3.9878	58
55	54.8948	4.7496	375

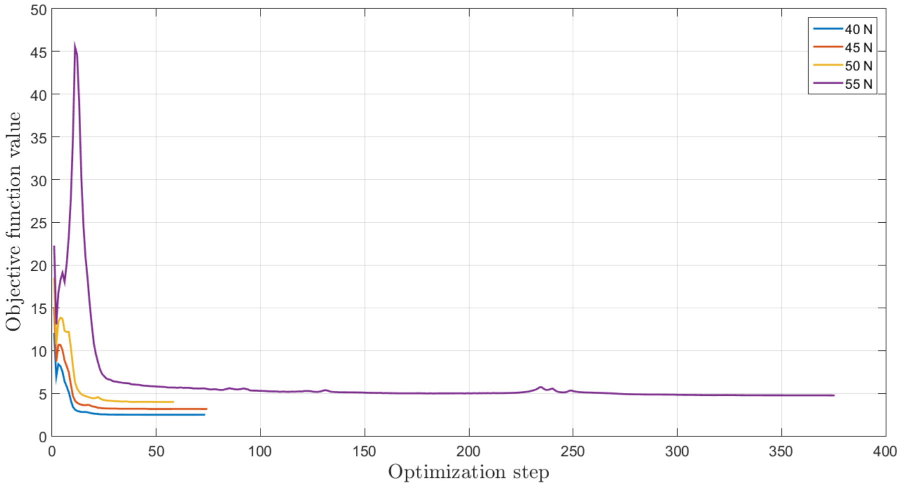

Figure 7.24: Evolution of the objective function values for the different load values in problem 2.

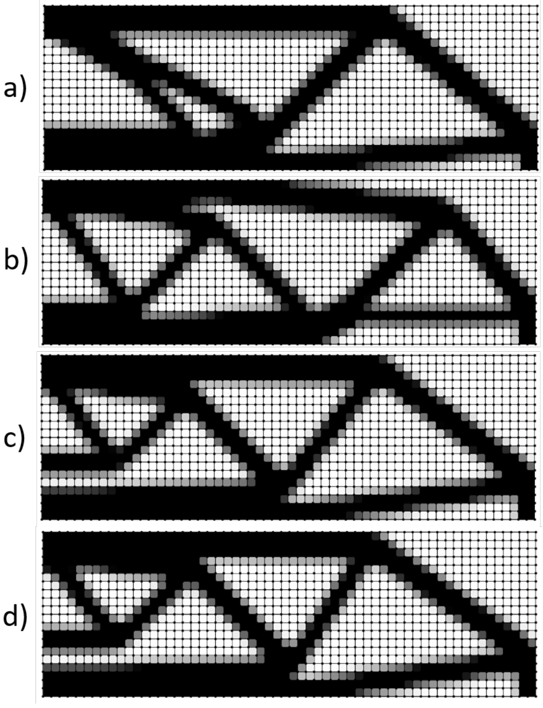

Figure 7.25: Solutions for problem 3 with load values of: a) 30 N, b) 35 N, c) 40 N and d) 45 N.

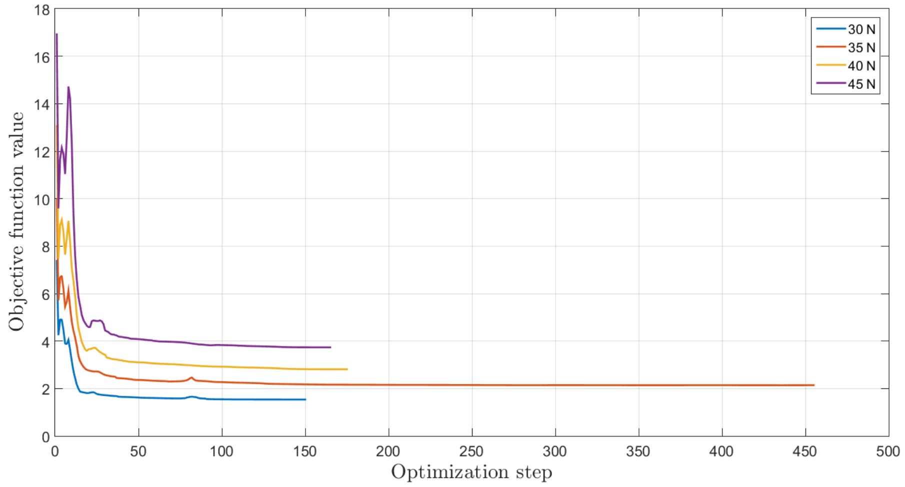

Figure 7.26: Evolution of the objective function values for the different load values in problem 3.

Table 7.8: Numerical results for different loads for problem 3.

Load	von Mises [MPa]	W^{C}[J]	Iterations
30	46.5413	1.5346	150
35	47.0529	2.1401	455
40	50.6925	2.8120	175
45	58.1473	3.7328	165

Table 7.9: Numerical results for different loads for problem 4.

Load	von Mises [MPa]	W^{C}[J]	Iterations
40	24.6741	0.1345	170
50	29.6640	0.2423	78
60	36.1474	0.5322	67
70	42.0953	1.5907	67

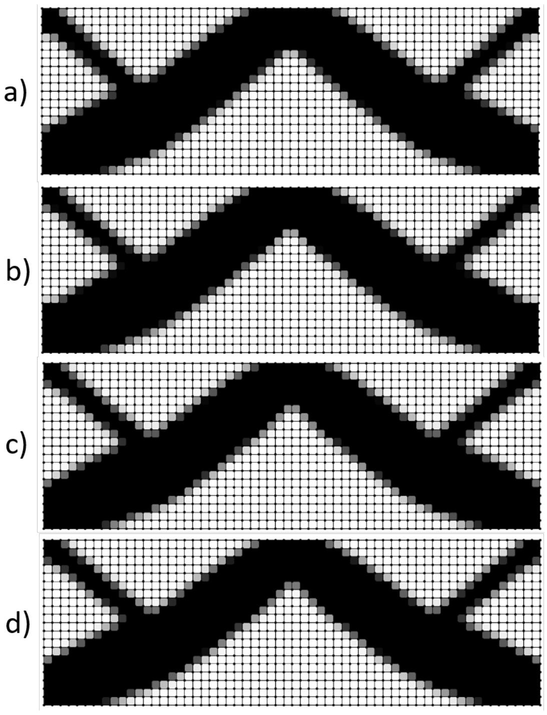

Figure 7.27: Solutions for problem 4 with load values of: a) 40 N, b) 50 N, c) 60 N and d) 70 N.

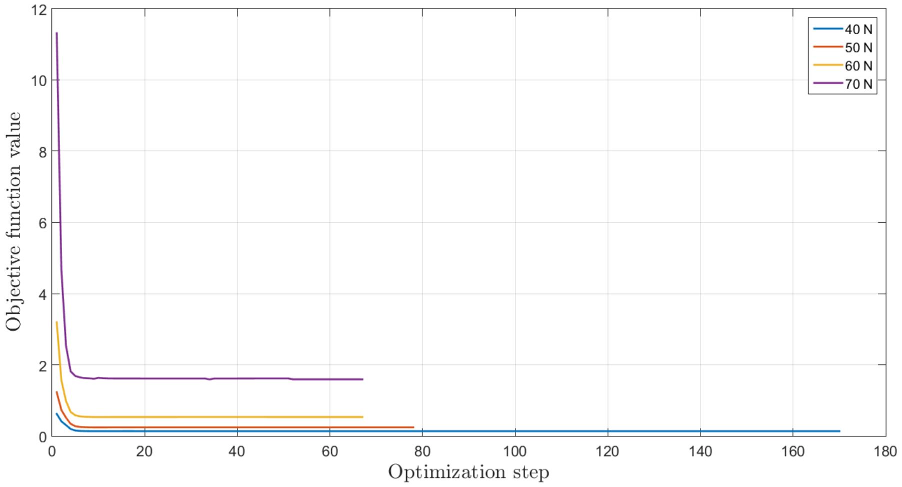

Figure 7.28: Evolution of the objective function values for the different load values in problem 4.

7.4 Final Topologies for End Compliance

For the minimization of end-compliance has objective function it was verified that the algorithm has the same behaviour (Figure 7.29). It is possible to note that the first design is a homogeneous distribution through the domain with the prescribed volume fraction value for all elements. The solution were obtained by solving the optimization problem for a load value of 40 N. The obtained results are shown in Table 7.10. In terms of the solution it can be verified that the topologies are similar to the ones using the complementary work as objective function (Figure 7.30). The evolution of the objective function values can be seen in Figure 7.31 for the first 3 problems and in Figure 7.32 for problem number 4. The plots are presented separately due to the difference of range in values.

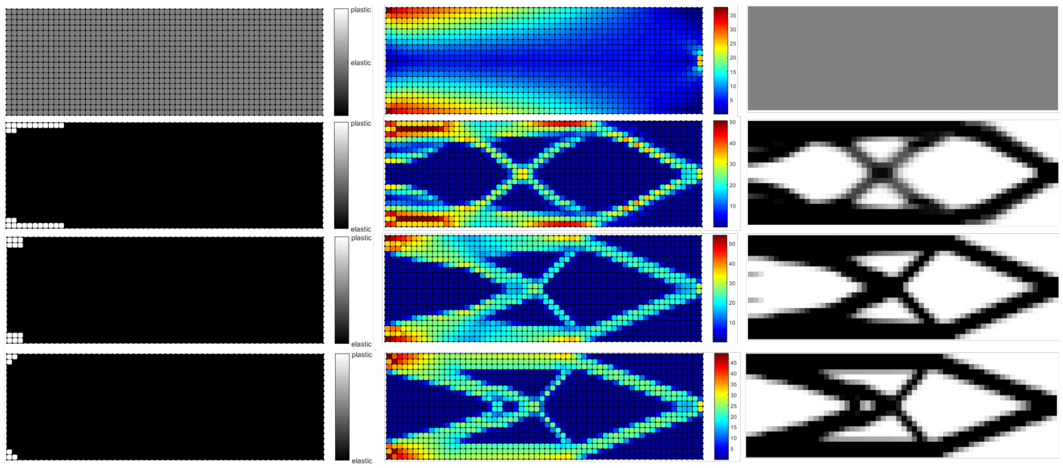

Figure 7.29: Element's state (left), von Mises stress (middle) and design configuration for 1, 10, 20 and last optimization steps, respectively.

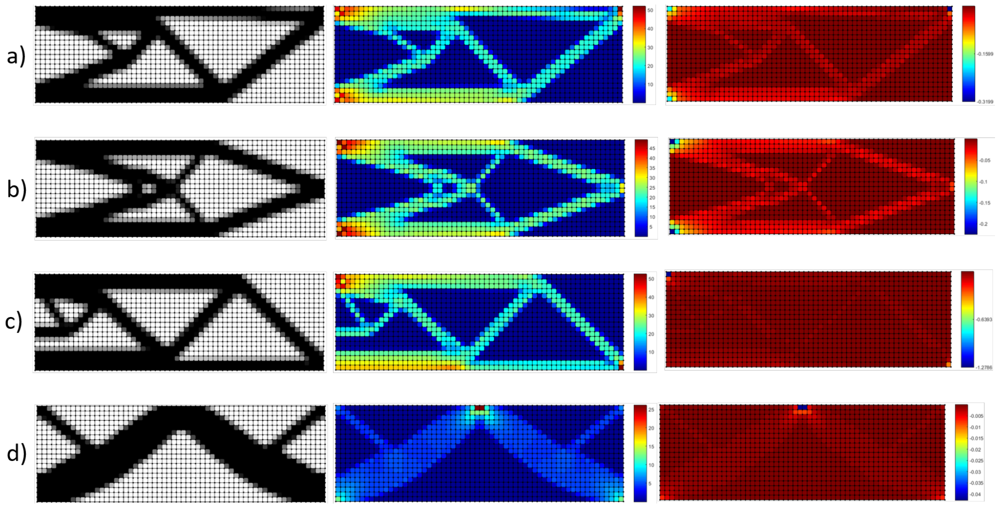

Figure 7.30: Solutions (left), von Mises stress (middle) and local sensitivities (right) for: a) problem 1, b) problem 2, c) problem 3 and d) problem 4.

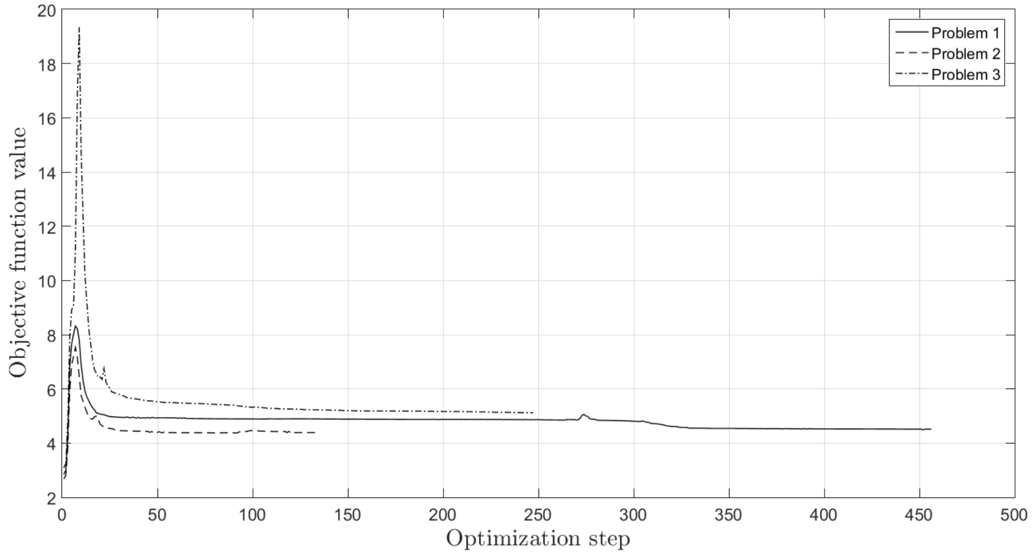

Figure 7.31: Evolution of the objective function values for problems 1,2 and 3.

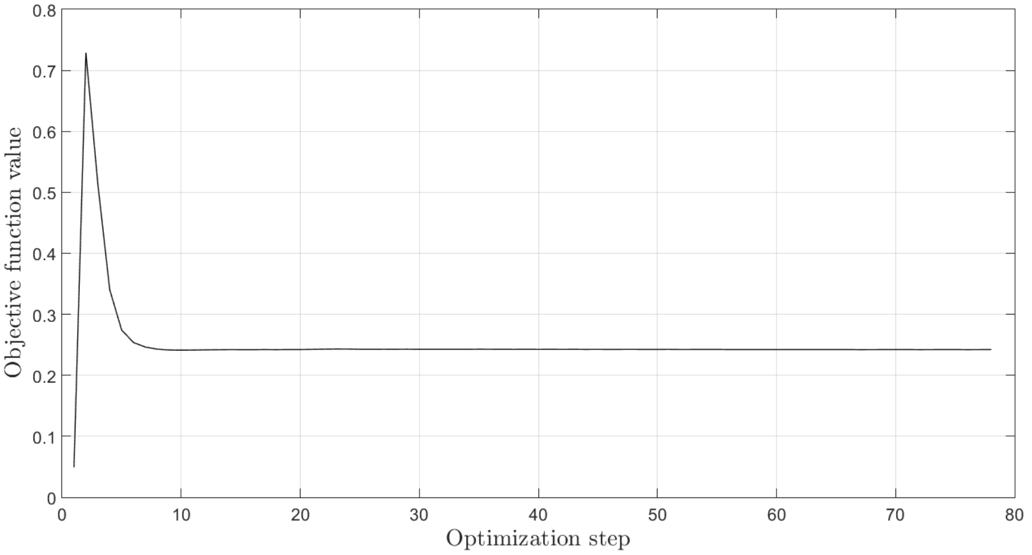

Figure 7.32: Evolution of the objective function value for problem 4.

Table 7.10: Numerical results for all problems

Problem	von Mises [MPa]	Compliance	Iterations
1	52.045	4.5125	456
2	49.2462	4.3865	134
3	52.5414	5.1199	247
4	26.0966	0.2320	69

Chapter 8

Final Remarks

8.1 Conclusion

In this work, a computational tool was developed using Matlab as a programming platform. The primary objective of the tool is to solve the topology optimization problem. However, a simulation module using the finite element method, that can be used individually or within the optimization procedures with minimal changes, was also developed. This tool has four different boundary conditions, but more conditions can be implemented in a straightforward manner with simple modifications.

Despite the use of simple approaches, some interesting conclusions about the operation of the topology optimization algorithm were retrieved, such as the reduction of plastic elements through the optimization steps and the distribution of material for different load cases.

Based on the results, it can be observed that, for the topology optimization problem in the 2D analysis, there are no significant differences between the use of plane stress and plane strain assumptions. The distribution of material and the numerical results are similar in both approaches.

It can also be concluded that the use of an explicit incremental scheme has a high computational cost. When implemented using programming languages such as Matlab the problem appears to be cumbersome to solve for a large number of design iterations and elements.

Therefore, for future works, it is suggested to use an iterative scheme such as the Newton-Raphson method. However, taking into account that the method can fail to converge for large displacements and in order to stabilize the convergence for the equilibrium iterations, it can be relaxed by eliminating the nodes surrounded by void.

8.2 Future Work

With the work done in this document some future works are proposed in order to continue and improve the study in this field:

- Elaboration of an interface using Matlab's GUI;

- Implementation using iterative schemes such as Newton-Raphson method;

- Implementation of constitutive models such as elastoplasticity or viscoplasticity;
- Resolution of the topology optimization problem using other objective functions (e.g. minimization of the strain energy, multiple load case).

Bibliography

[1] A. A. Campos, J. Oliveira, and J. Cruz, *Otimização não-linear em engenharia: cálculo estrutural e computacional multiescala.* ETEP - Edições Técnicas e Profissionais, 2015.

[2] R. Alberdi, G. Zhang, L. Li, and K. Khandelwal, “A unified framework for nonlinear path-dependent sensitivity analysis in topology optimization,” *International Journal for Numerical Methods in Engineering*, vol. 115, no. 1, pp. 1–56, 2018.

[3] J. Maxwell, “On Reciprocal Figures, Frames and Diagrams of Force,” *Transactions of the Royal Society of Edinburgh*, vol. 26, pp. 1–40, 1870.

[4] A. Michell, “The Limits of Economy of Material in Frame Structures,” *Philisophical Magazine*, vol. 8, no. 6, pp. 589–597, 1904.

[5] M. 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, 1988.

[6] S. Cho and H. S. Jung, “Design sensitivity analysis and topology optimization of displacement-loaded non-linear structures,” *Computer Methods in Applied Mechanics and Engineering*, vol. 192, no. 22-23, pp. 2539–2553, 2003.

[7] M. Bendsøe, “Optimal Shape Design as a Material Distribution Problem,” *Structural and Multidisciplinary Optimization*, vol. 1, no. 4, pp. 193–202, 1989.

[8] G. Rozvany, M. Zhou, and T. Birker, “Generalized Shape Optimization Without Homogenization,” *Structural and Multidisciplinary Optimization*, vol. 4, no. 3-4, pp. 250–252, 1992.

[9] K. Yuge and N. Kikuchi, “Optimization of a frame structure subjected to a plastic deformation,” *Structural Optimization*, vol. 10, no. 3-4, pp. 197–208, 1995.

[10] K. Maute, S. Schwarz, and E. Ramm, “Adaptive topology optimization of elastoplastic structures,” *Structural Optimization*, vol. 15, no. 2, pp. 81–91, 1998.

[11] J. Kato, H. Hoshiba, S. Takase, K. Terada, and T. Kyoya, “Analytical sensitivity in topology optimization for elastoplastic composites,” *Structural and Multidisciplinary Optimization*, vol. 52, no. 3, pp. 507–526, 2015.

[12] F. Teixeira-Dias, J. Pinho-da Cruz, R. F. Valente, and R. Sousa, *Método dos elementos finitos: técnicas de simulação numérica em engenharia.* 2010.

[13] E. A. de Souza Neto, D. Peri, and D. R. J. Owen, *Computational Methods for Plasticity.* 2008.

[14] A. J. M. Ferreira, *Problemas de elementos finitos em Matlab.* Fundação Calouste Gulbenkian. Serviço de Educação e Bolsas, 2010.

[15] P. Hartley, *Introduction to Computational Plasticity*, vol. 39. 2006.

[16] T. Sokół, "A 99 line code for discretized Michell truss optimization written in Mathematica," *Structural and Multidisciplinary Optimization*, vol. 43, no. 2, pp. 181–190, 2011.

[17] M. Bendsøe and O. Sigmund, *Topology Optimization. Theory, Methods and Applications.* Springer, 2003.

[18] O. Sigmund, "A 99 line topology optimization code written in matlab," *Structural and Multidisciplinary Optimization*, vol. 21, no. 2, pp. 120–127, 2001.

Appendix A

Developed functions

A.1 Main

```matlab
function TopOpt(nelx,nely,L,B,volfrac,E,nu,Hparams,penal,rmin,fmax,step,state,problem,objective)
% MESH GENERATION
[node, element] = RectangularMesh(L,B,nelx,nely);
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
change = 1.;
% START ITERATION
while change > 0.01
    loop = loop + 1;
    xold = x;
    % FINITE ELEMENT ANALYSIS
    X = flip(x',2);
    [U1, KE,stress,strain,smises,P,Plas] = FEM(node,element,X,penal,E,nu,fmax,step,Hparams,state,problem);
    % OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
    [c,dce] = SensitivityAnalysis(element,X,penal,step,U1,KE,P,objective);
    dc = flip(reshape(dce,[nelx,nely])',1);
    % FILTERING OF SENSITIVITIES
    [dc] = check(nelx,nely,rmin,x,dc);
    % DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
    [x] = OC(nelx,nely,x,volfrac,dc);
    % PRINT RESULTS
    change = max(max(abs(x-xold)));
    disp([' It.: ' sprintf('%4i',loop) ' Obj.: ' sprintf('%10.4f',c) ...
           ' Vol.: ' sprintf('%6.3f',sum(sum(x))/(nelx*nely)) ...
            ' ch.: ' sprintf('%6.3f',change )])
    % PLOT DENSITIES
    colormap(gray); imagesc(-x); axis equal; axis tight; axis off;pause(1e-6);
    % OUTPUTS
    ObjF(loop) = c;
    Dce(:,loop) = dce;
    it(loop) = loop;
    Stress(:,:,loop) = stress;
    Strain(:,:,loop) = strain;
    Smises(:,loop) = smises(:,1);
    GPlas(:,loop) = Plas;
    x2(:,:,loop) = x;
end
figure
plot(it,ObjF)
end
```

A.2 Mesh Generation

```
function [node, element, Nnodes]=RectangularMesh(L, B, Nx, Ny)
nel = Nx*Ny;     %number of elements
nnel = 4;    %number of nodes per element
nx = Nx+1;    %nodes in x
ny = Ny+1;    %nodes in y
nnode = nx*ny;     %number of nodes
Nnodes = 1:nnode;    %numbering of nodes
%node coordinates
nodex = linspace(0,L,nx);
nodey = linspace(0,B,ny);
[xx, yy] = meshgrid(nodey,nodex);
node = [yy(:) xx(:)];
%Connectivity
element = zeros(nel,nnel);
Nnodes = reshape(Nnodes,nx,ny);
element(:,1) = reshape(Nnodes(1:nx-1,1:ny-1),nel,1);
element(:,2) = reshape(Nnodes(2:nx,1:ny-1),nel,1);
element(:,3) = reshape(Nnodes(2:nx,2:ny),nel,1);
element(:,4) = reshape(Nnodes(1:nx-1,2:ny),nel,1);
end
```

A.3 Finite Element Method

```
function [ U2,KE2,stress,strain,smises,P,Plas] = FEM(node,element,x,penal,E
    ,nu,fmax,step,Hparams,state,Problem)
%Initialize variables
numnode = size(node,1);
numelem = size(element,1);
U1 = zeros(2*numnode,1);
f = zeros(2*numnode,1);
P = zeros(2*numnode,1);
Load = 0:step:fmax;
fy = Hparams(1);
dE = Hparams(2);
Eg(1:numelem) = E;
smises = zeros(numelem,3);
Plas = zeros(numelem,1);
%Boundary Condition
switch Problem
    case 1
        %Boundary Condition 1
        alldofs = 1:2*numnode;
        encastre = find(node(:,1)==0)*2;
        diff = encastre-ones(numel(encastre),1);
        fixeddofs(1:2:2*numel(encastre)-1) = diff;
        fixeddofs(2:2:2*numel(encastre)) = encastre;
        activeDof = setdiff(alldofs',fixeddofs);
    case 2
        %Boundary Condition 2
        nx = 2*sum(node(:,2)==0);
        halfend = nx*(((sum(node(:,1)==0)-1)/2)+1);
        alldofs = 1:2*numnode;
        encastre = find(node(:,1)==0)*2;
        diff = encastre-ones(numel(encastre),1);
        fixeddofs(1:2:2*numel(encastre)-1) = diff;
        fixeddofs(2:2:2*numel(encastre)) = encastre;
        activeDof = setdiff(alldofs',fixeddofs);
    case 3
        %Boundary Condition 3
        nelx = sum(node(:,2)==0)-1;
        alldofs = 1:2*numnode;
        encastre = find(node(:,1)==0)*2;
        diff = encastre-ones(numel(encastre),1);
        last = (nelx+1)*2;
        fixeddofs = union(diff,last);
        activeDof = setdiff(alldofs',fixeddofs);
    case 4
        %Boundary Condition 4
        nelx = sum(node(:,2)==0)-1;
        alldofs = 1:2*numnode;
        encastre = find(node(:,1)==0)*2;
        encastre2 = find(node(:,1)==nelx)*2;
        diff2 = encastre2-ones(numel(encastre2),1);
        diff = encastre-ones(numel(encastre),1);
        union1 = union(encastre,diff);
        union2 = union(encastre2,diff2);
        fixeddofs = union(union1,union2);
        activeDof = setdiff(alldofs',fixeddofs);
        halftop = 2*numnode-nelx;
end
%Young's modulus variation
for i = 1:size(Load,2)
    for e = 1:numelem
        if smises(e,1) > fy
           Eg(e) = dE*Eg(e);
           Plas(e) = 1;
        end
    end
    %Constitutive Tensor
    [C] = ConstitutiveTensor(Eg, nu, x, penal, numelem,state);
    %Stiffness Matrix
    [ K,~,~,KE ] = StiffnessMatrix( node, element, C);
    %Load
    switch Problem
        case 1
            f(numnode*2) = -Load(i);
            P(numnode*2) = -step;
        case 2
            f(halfend) = -Load(i);
            P(halfend) = -step;
        case 3
            f(max(encastre)) = -Load(i);
            P(max(encastre)) = -step;
        case 4
            f(halftop) = -Load(i);
            P(halftop) = -step;
    end
    %Solving KU=F
    U = K(activeDof,activeDof)\f(activeDof);
    U1(activeDof) = U;
    %Compute Stress, Strain, von Mises
    [stress, strain, smises] = GlobalStress(node,element,U1,C);
    %Output
    U2(:,i) = U1;
    KE2{i} = KE;
end
end
```

A.4 Constitutive Tensor

```
function [C] = ConstitutiveTensor(E, nu, x, penal, numelem, state)
ntens = 3;
C = zeros(ntens,ntens,numelem);
    for e = 1:numelem
        Ee = E(e);
        if strcmp(state,'plane-strain')
            dsde = Ee/((1+nu)*(1-2*nu))*[(1-nu) nu 0; nu (1-nu) 0; 0 0 (1-2*nu)/2];
        end
        if strcmp(state,'plane-stress')
            dsde = Ee/(1-nu^2)*[1 nu 0; nu 1 0; 0 0 (1-nu)/2];
        end
        C(:,:,e) = x(e)^penal*dsde;
    end
end
```

A.5 Stiffness Matrix

```
function [ K,B,C,KE ] = StiffnessMatrix( node, element, C )
numnode = size(node,1);
numelem = size(element,1);
%Quadrature
q = 0.577350269189626;
Q = [-q -q;
      q -q;
      q  q;
     -q  q]; %natural coordinates
W = [1;1;1;1]; %weights
K = sparse(2*numnode,2*numnode);
        for e = 1:numelem
        Ce = C(:,:,e);
        indice = element(e,:);
        nn = length(indice);
        eldof = [indice(1)*2-1;indice(1)*2;...
                 indice(2)*2-1;indice(2)*2;...
                 indice(3)*2-1;indice(3)*2;
                 indice(4)*2-1;indice(4)*2];
            for i = 1:size(W,1)
            xi = Q(i,1);
            eta = Q(i,2);
            wt = W(i);
            N = 1/4*[ (1-xi)*(1-eta);
                      (1+xi)*(1-eta);
                      (1+xi)*(1+eta);
```

```
                  (1-xi)*(1+eta)]; %Shape Variation
            dNdxi = 1/4*[-(1-eta) -(1-xi);
                      1-eta  -(1+xi);
                      1+eta    1+xi;
                     -(1+eta)   1-xi]; %Derivatives of Shape functions (dxi deta)
            %Jacobian
            J0 = node(indice,:)'*dNdxi;
            invJ0 = inv(J0);
            dNdx = dNdxi*invJ0;
            %B Matrix Assembly
            B = zeros(3,8);
            B(1,1:2:2*nn) = dNdx(:,1)';
            B(2,2:2:2*nn) = dNdx(:,2)';
            B(3,1:2:2*nn) = dNdx(:,2)';
            B(3,2:2:2*nn) = dNdx(:,1)';
            %K Matrix Assembly
            Ke(:,:,i) = B'*Ce*B*wt*det(J0);
            K(eldof,eldof) = K(eldof,eldof)+Ke(:,:,i);
            end
            KE(:,:,e) = Ke(:,:,1)+Ke(:,:,2)+Ke(:,:,3)+Ke(:,:,4);
        end
    K = full(K);  %Sparse Matrix to Full Matrix
end
```

A.6 Stress, Strain and von Mises Stress

```
function [ Stress, Strain, Smises ] = GlobalStress( node, element, U, C )
numelem = size(element,1);
Stress = zeros(numelem,3);
Strain = zeros(numelem,3);
Smises = zeros(numelem,3);
stressPoints = [0 0];
    for e = 1:numelem
        Ce = C(:,:,e);
        indice = element(e,:);
        nn = length(indice);
        eldof = [indice(1)*2-1;indice(1)*2;...
                 indice(2)*2-1;indice(2)*2;...
                 indice(3)*2-1;indice(3)*2;
                 indice(4)*2-1;indice(4)*2];
            for i = 1:1
                pt = stressPoints(i,:);
                xi = pt(1);
                eta = pt(2);
                N = 1/4*[ (1-xi)*(1-eta);
                          (1+xi)*(1-eta);
                          (1+xi)*(1+eta);
                          (1-xi)*(1+eta)]; %Shape Variation
                dNdxi = 1/4*[-(1-eta) -(1-xi);
                               1-eta  -(1+xi);
                               1+eta    1+xi;
                             -(1+eta)   1-xi]; %Derivatives of Shape functions (dxi deta)
                %Jacobian
                J0 = node(indice,:)'*dNdxi;
                invJ0 = inv(J0);
                dNdx = dNdxi*invJ0;
                %B Matrix Assembly
                B = zeros(3,8);
                B(1,1:2:2*nn) = dNdx(:,1)';
                B(2,2:2:2*nn) = dNdx(:,2)';
                B(3,1:2:2*nn) = dNdx(:,2)';
                B(3,2:2:2*nn) = dNdx(:,1)';
                %stress and strain
                strain = B*U(eldof);
                stress = Ce*strain;
                smises = stress(1)^2 - stress(1)*stress(2) + stress(2)^2+3*stress(3)^2;
                smises = sqrt(smises);
                Smises(e,:) = smises;
                Stress(e,:) = Ce*strain;
                Strain(e,:) = strain';
            end
    end
end
```