Master thesis : Topology optimization in nonlinear vibrations

Auteur : Trillet, Denis
Promoteur(s) : Kerschen, Gaetan
Faculté : Faculté des Sciences appliquées
Diplôme : Master en ingénieur civil en aérospatiale, à finalité spécialisée en "aerospace engineering"
Année académique : 2017-2018
URI/URL : http://hdl.handle.net/2268.2/5496

University of Liège - Faculty of Applied Sciences

Topology optimization in non-linear vibrations

Master's thesis conducted by

Denis Trillet

in order to obtain the degree of Master in Aerospace Engineering

Under the supervision of

Prof. Pierre Duysinx
Prof. Gaëtan Kerschen

Academic Year 2017-2018

Topology optimization in nonlinear vibrations

Trillet Denis
Promoters: Prof. P. Duysinx
Prof. G. Kerschen

University of Liège: Faculty of Applied Science
Master in Aerospace Engineering
Academic Year 2017-2018

Abstract

The latest investigations on vibrations has highlighted the importance of their nonlinear behaviour. These new theories have brought new challenges for structural engineering. Indeed, the way to control their consequences can be derived from the linear theory by using devices called absorbers. This extended theory states that the nonlinear absorber has to present the same restoring force form than the excitation. This is the aim of this thesis, create nonlinear absorber by imposing a force/displacement curve with the help of topology optimization. Thus, the different type of nonlinearities are cited and the way to identify them is explained. Then, a nonlinear optimization model is built. The main problem of nonlinear optimization is the mesh distortion appearing in low density elements. To solve this problem, the energy density interpolation has been chosen between several methods. This model has been validated by solving a classical end-compliance problem. After that, the prescribed curve problem is investigated. The problem takes the form of an error minimization. After comparison, the $L1$ normalized error gives the best results. Some linear curves are imposed taking the stiffest beam behaviour as a reference. The code shows its limit when the imposed behaviour is under 30% of the reference curve. Then nonlinear curves are imposed and converge under certain conditions. Finally, the current model shows interesting results.

Acknowledgements

First, I would like to acknowledge both of my supervisor Prof. Pierre Duysinx and Prof. Gaëtan Kerschen for their constructive critiques and advice all along the thesis. I also thank Prof. Olivier Brüls who accepted to be member of my jury.

Then I want to thank Kevin who read and corrected this work.

I am also grateful to all my friends that I met either during lessons or while running, who made these six years go by too fast.

I also thanks my parents and my two brothers who supported me all along my studies and provide me a peaceful situation to work.

Lastly, I would like to thank Eva who has supported me during this thesis and who has invested a lot to help me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In structural mechanics, vibrations are an important source of problems. More precisely, when a structure comes to resonance, this phenomenon can induce a degradation of the structure or have an impact on its performance. Indeed, a perfect example is the famous Tacoma bridge which collapses because of aeroelastic induced vibration. In order to avoid these effects, researchers have worked on device that can reduce the effect of resonance. In 1934, Den Hartog proved that a resonance peak can be reduced into two peaks of lower intensity using a vibration absorber ([9]). This device, composed of a spring and a damper, has been largely studied for the linear mode of vibration. The scientific community has recently intensified the research about the nonlinear vibrations. These are more representative of the reality and they could be present in any structure. Moreover, it is of special interest in the aerospace domain where the performance and the security are the most important. In fact, new materials often show a nonlinear behaviour and the added devices increase the number of contacts in the structure. The feature of these vibrations is that the resonance peak is not associated to only one frequency. The resonance effect can be extended on a large bandwidth and it can highlight a frequency energy dependence. This dependence can be linear but not only, the nonlinear part can present any form. Obviously, all the assumptions made in the linear case such as the superposition principle and the invariance of frequency response function are no longer true in this domain. Even though nonlinear behaviours are less predictable than their linear counterparts, a complete way to identify them has been developed for the last decade. (G.Kerschen et al., 2006, [10]). The Den Hartog's theorem seems to be useless here since it focuses on a narrow bandwidth. Nevertheless, this theorem has been extended to the nonlinear domain. From this extension, some theories on nonlinear vibration absorber have appeared (Habib et al, 2015, [5]). It allows to design the same kind of devices but focused on the nonlinear type of vibration.

Parallel to these discoveries, topology optimization has become really popular these days, especially for the first design of new features. Indeed, topology optimization allows to explore more exotic model that will not come in mind of a human. However, with the expansion of the 3D printing, the shapes that were not machinable are now a reality. With the optimization, everything is possible, even the use of nonlinear models in order to explore new domain.

The main goal of this thesis is to design a nonlinear vibration absorber with the help of the topology optimization. This will be done using two types of domain shapes, a

cantilever and a doubly clamped beams. This work can be divided in three chapters.

In the first chapter, the nonlinear vibrations will be reviewed. More precisely, their origins and its different types will be explained, such as the way to identified it. Then, the absorber theory will be detailed for the linear and nonlinear cases.

The second chapter will focus on the nonlinear topology model. It will contain few theories about nonlinear mechanic and topology optimization. Then, the major issue related to the nonlinear optimization will be highlighted. Several methods to solve it will be explained and the most convenient will be performed. This chapter will be put in parallel with a end-compliance optimization which will allow to validate the result of the previously built model. Finally, the performance of the code will be discussed.

The last chapter will contain the first steps to design a nonlinear absorber with the topology optimization model built. Thus, some formulations will be compared and tested. Then, it will be applied to simple cases to see their limits. Finally, the effect of some parameters will be investigated to get the best results.

Finally, the perspectives and the possible improvements of this thesis will be discussed.

Chapter 2

Nonlinear vibrations and absorbers

2.1 Introduction

In engineering structures, vibrations are a common preoccupation. Most of the time the issue induced by vibration are solved using linear assumptions. However, in reality, this is more complex because most of the vibration has a nonlinear part. Consequently, the use of linear assumptions which allow to solve simple problems is useless. Obviously, by definition the superposition principle is not valid anymore, the uniqueness of the solution or the invariance of the frequency response function (FRF) neither. Thus, nonlinearities are present anywhere and their nature can be very different. Four main types of nonlinearity can be listed (Nam-Ho Kim, 2015, [1]): geometric, material, kinematic or force one. These types of nonlinearity are briefly explained below.

Geometric nonlinearities
Geometric nonlinearities are the nonlinearities induced by the deformation of a structure. This is specific to large displacements. In this case, the second order term of the Green-Lagrange strain tensor in Eq. 2.1 can not be ignored. It was possible with linear assumptions because this term is infinitesimal in these conditions.

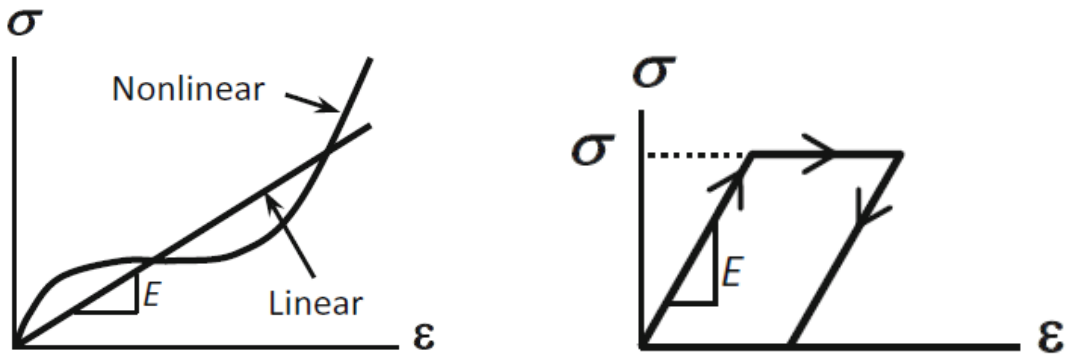
$$\varepsilon_{ij} = \left(\frac{\partial u_j}{\partial dx_i} + \frac{\partial u_i}{\partial dx_j} + \frac{\partial u_m}{\partial dx_i} \frac{\partial u_m}{\partial dx_j} \right) \tag{2.1}$$

where ε_{ij} is the strain, u_i the displacement in the i direction and x_i the i coordinate.

As it will be explained in the next chapter, geometrical nonlinearities are the most interesting in our case. Indeed, the purpose will be to find a geometry that shows interesting features with the help of topology optimization.

Material nonlinearities
Material nonlinearities are related to the constitutive law of the materials itself. Indeed, the stress strain relation is not always linear like for elastic materials but it can take various shapes such as the hyperelastic one as illustrated in Fig. 2.1a. Even for the elastic materials, when the strain becomes bigger, it enters in plastification and the relation is no longer linear. In this case, the deformation becomes permanent as shown in Fig. 2.1b.

In this type of nonlinearity, one can also mention the viscoelasticity which shows a time dependent behaviour that is modelled with a spring and a dashpot.



(a) For linear and non linear materials.

(b) For plasticity.

Figure 2.1: Stress-strain constitutive relation.

Kinematic nonlinearities
Kinematic nonlinearities or boundary nonlinearities are due to the boundary conditions which influence the mechanism. It happens when boundary conditions depend on the deformation of the structure. In other words, it means that either the location and/or the amplitude of these conditions are unknown. A perfect example is the case where two bodies enter in contact. It can induce some nonlinear behaviours.

Force nonlinearities
Force nonlinearities, like the kinematic one, occur when the force that acts on the system is related to the deformation. Contrary to the kinematic nonlinearities, it is the direction and/or the amplitude of the force that change. The obvious example is the pressure induced by a fluid on a beam that goes on large displacement as shown in Fig. 2.2.

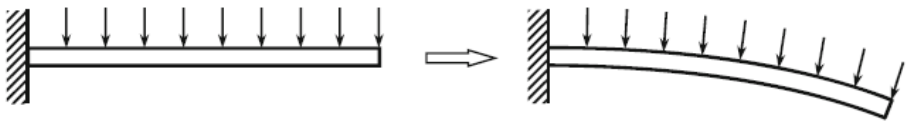


Figure 2.2: Example of force nonlinearity from [1]: Follow-up pressure load of a beam under large deformation.

All of these types of nonlinearies can be found in one system but it will be very complex to model. It is usually preferable to consider only one type.

2.2 Nonlinearity identification

A complete procedure has been developed through the years in order to identify nonlinearities in any system. The complete evolution of this procedure has been reviewed and improved by G.Kerschen et al. (2006, [10]), J.P. Noel and G. Kerschen (2017, [2]). The first step of the process is the detection for seeing if a nonlinearity is actually present or not. Then, the nonlinearity is characterized. Its localization, its physical type and the way to model it are found. Finally, the parameter to fit the mechanical model with the data is estimated (e.g. the stiffness of the nonlinear spring).

2.2.1 Detection

The first step is the most straightforward but probably the most important one because of its power of decision. Indeed, it will induce the engineer to use or not a nonlinear modeling. This is an experimental step and many methods could be used. Three of them will be cited. In order to detect the presence of any nonlinearity, the system is excited by a measured signal and its response is studied. The easiest signal to impose is the `sinesweep` signal ($\sin(\frac{\omega_1-\omega_2}{2n}t^2+\omega_1 t)$) given that it concentrates the energy. This concentration actives a lot the nonlinearities. A common method to detect the presence of a nonlinearity is the use of the superposition principle. If the system is linear, the response of the system to a sum of excitation $f_i(t)$ should be the sum of the response of each excitation $x_i(t)$. By homogeneity principle, these expressions can be multiplied. As example, for three excitations:

$$\begin{aligned} f_1(t) &\longrightarrow x_1(t) \\ f_2(t) &\longrightarrow x_2(t) \\ f_3(t)=\alpha f_1(t)+\beta f_2(t) &\longrightarrow x_3(t)=\alpha x_1(t)+\beta x_2(t) \end{aligned}$$

If it is not the case even for a set of parameters, there is no doubt that nonlinearities are present as mentioned by K. Worden and G.R. Tomlinson (2001, [11]). Another way of detection is the harmonic distortion which states that since the excitation signal is monoharmonic, the response has to be monoharmonic as well (K. Worden and G.R. Tomlinson, 2001, [11]).

One last method that can be cited is the violation of the invariance principle. It means that the response should not depend on the amplitude of the excitation. If nonlinearities are present, the range of frequency response often depends on the excitation amplitude and the jumps/bifurcations are more distinct.

2.2.2 Characterization

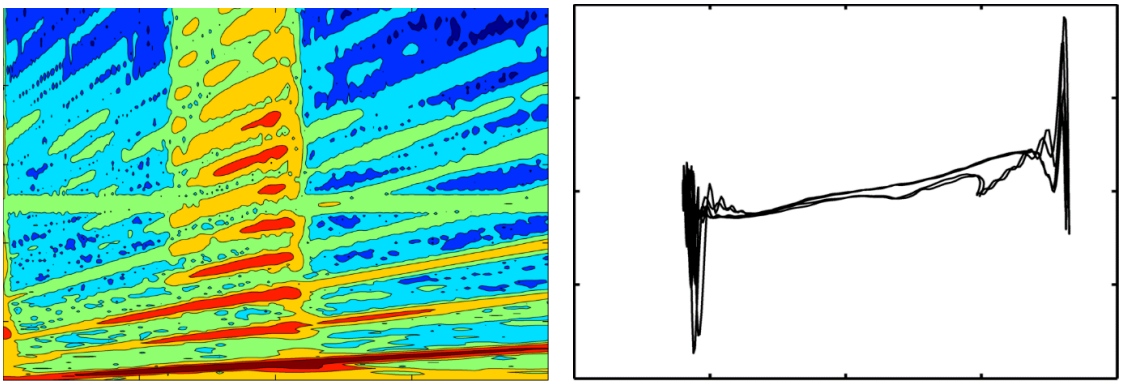
If a nonlinearity has been detected, it is important to characterize it, i.e. find its physical nature and its localization. This step is more complex that the previous one since the physical reason of the nonlinear behaviour can be very diverse (as mentioned, geometric, kinematic,...). Moreover, the characterization is important for the smooth running of the parameters estimation. Two methods can be used: the wavelet transform (WT) or the acceleration surface method (ASM).

The WT is based on a time-frequency analysis which highlights the frequency-amplitude dependence of the nonlinear vibration. The classical time-frequency analysis is based on the Short-Time Fourier Transform (STFT) but the fixed observation window is not the most convenient. It leads to the introduction of the WT which allows to use a variable resolution on the windows. This expression can be seen in Eq. 2.2

$$X(\omega,\tau)=\int_{-\infty}^{+\infty} x(t)\psi(\frac{t-b}{a})e^{-j\omega t}dt \tag{2.2}$$

where ψ is the mother wave wavelet. The observation window can be modified with the frequency ω, b locates the observation windows and a is the scaling factor of the resolution.

When the graph is plotted (instantaneous frequency as a function of the sweep frequency), it is possible to see if harmonics happen. Indeed, they will produce some lines (for linear sinesweep). As represented in Fig. 2.3a, the lines of higher amplitude appear in red. The one slope line is the fundamental response.



(a) For WT. (b) For ASM.

Figure 2.3: Typical graphs used for characterisation step from [2].

The ASM is derived from the restoring force plot (Fig. 2.3b) which directly shows the nonlinear stiffness or damping. It is based on the second law of Newton applied to a degree of freedom (DOF) adjacent to a nonlinearity. Mathematically, it is expressed as in Eq.2.3 and schematically represented as in Fig. 2.4.

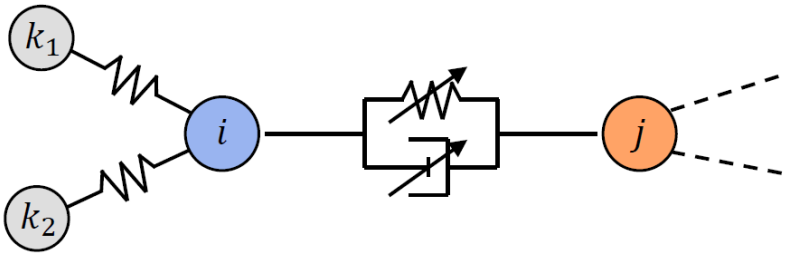


Figure 2.4: Scheme of the ASM from [3].

$$\sum_{n=1}^{n_p} m_{i,n}\ddot{x}_n + g_i(\mathbf{x}, \dot{\mathbf{x}}) = p_i \tag{2.3}$$

where i is the DOF studied, $m_{i,n}$ the mass matrix element, $\mathbf{x}$ the displacement, $\dot{\mathbf{x}}$ the velocity, $\ddot{\mathbf{x}}$ the acceleration, g_i the restoring force and p_i the excitation vector.

This equation can be simplified to only one DOF, the one with the highest amplitude response. It is also possible to get rid off the inertia terms and the restoring forces external to the studied connection. The approximated equation of motion is the next one:

$$m_{i,i}\ddot{x}_i + g_i\left(x_i - x_j, \ddot{x}_i - \ddot{x}_j\right) \approx p_i \tag{2.4}$$

where j is denoting a DOF next to the i^{th}. Lastly if no force are applied on the node considered it comes:

$$g_i\left(x_i - x_j, \ddot{x}_i - \ddot{x}_j\right) \approx -m_{i,i}\ddot{x}_i \tag{2.5}$$

As the Eq. 2.5 shown, after assumptions, the nonlinearity is related to the acceleration of the adjacent point. It allows to visualize the nonlinearity by plotting acceleration response as a function of restoring force ton obtain a graph as represented in Fig. 2.3b. Obviously, the two assumptions (external connections are linear or negligible function of the displacement inside the connection considered) used in this method have to be kept in mind otherwise it can lead to inaccurate results.

2.2.3 Parameters estimation

The last step should finalize the model used to represent the nonlinearity. It is important to estimate the parameters because this model will be the basis of the absorber. A method to do this is the Frequency-domain nonlinear subspace identification (FNSI)(J.P. Noël and G. Kerschen, 2013, [12]). The behaviour of the system can be expressed as follow:

$$M\ddot{q} + C\dot{q} + Kq + k_{nl}f_{nl} = p \tag{2.6}$$

where M, C, K are the mass, damping and stiffness matrices respectively, q and p are the generalized displacement and external forces and finally, k_{nl} is the nonlinear strength which has to be estimated and f_{nl} is the assumed nonlinear function form (e.g. a polynomial form).

Generally, the acceleration and the external force are measured. Eq. 2.6 can be reformulated as:

$$M\ddot{q} + C\dot{q} + Kq = p - k_{nl}f_{nl} \tag{2.7}$$

Doing so, the right member is seen as known, like additional inputs. It can represent the extended response function, it is used as a nonlinear feedback of the underlying linear system. Then, the system of Eq. 2.6 can be rewritten in state-space form with the state vector $x = [q^T, \dot{q}^T]^T$:

$$\begin{cases} \dot{x}(t) = Ax(t) + Be(t) \\ y(t) = Cx(t) + De(t) \end{cases} \tag{2.8}$$

where $B = \begin{pmatrix} 0 & 0 \\ M^{-1} & k_{nl}M^{-1} \end{pmatrix}$ contains the nonlinearity term to be estimated and $e = \begin{pmatrix} p(t) \\ -f_{nl}(t) \end{pmatrix}$ is known. The last step is to discretize and to apply the Fourier transform to this equation in order to save some computational resources:

$$\begin{cases} z_k X(k) = AX(k) + BE(k) \\ Y(k) = CX(k) + DE(k) \end{cases} \tag{2.9}$$

with the z-transform variable $z_k = e^{j2\pi k/N}$.

This method is useful because it allows a general representation for nonlinear systems. Moreover, it can be easily extend to multi-input/output case and the algorithm to solve it for linear case are already developed and so highly efficient. The simplest function forms that exist are the polynomials such as $f_{nl} = x^n$. It will be the first function forms used to design absorbers.

2.3 Vibration absorber

In order to protect the system from the outcome of vibration, some devices have been developed. These ones are the vibration absorbers. Firstly, they were focused on linear system but nowadays, nonlinear system could also be protected. The different types of absorbers are explained bellow.

2.3.1 Linear system

One way to minimize the perturbation induced by resonance is to use a device to absorb vibration. In a linear case, the device is called linear tuned vibration absorber (LTVA). The theory of this device has been developed by Den Hartog. He used a spring and a damper which increases the robustness of the device (1934, [9]). The method is known as the equal peak method. Schematically, if the primary system is linear as represented by a spring k_1 and a mass M_1, the absorber will be represented by an added system mass/spring as shown in Fig. 2.5, where M_2 is the mass of the device, k_2 the stiffness of the absorber and c_2 its damping.

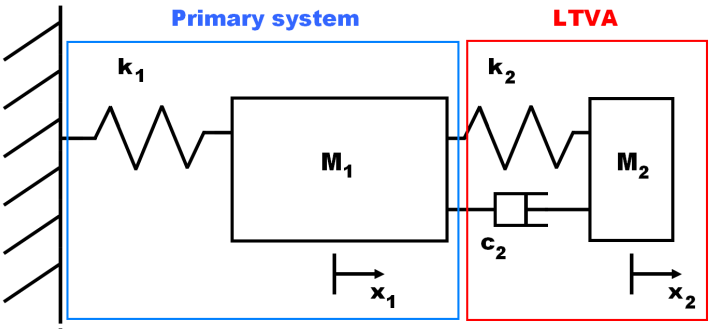


Figure 2.5: Scheme of the linear vibration absorber.

To understand this theory, the equations of motion of this system can be written as follow:

$$\begin{cases} M_1\ddot{x}_1 + k_1x_1 + c_2(\dot{x}_1 - \dot{x}_2) + k_2(x_1 - x_2) = F\sin\omega t \\ M_2\ddot{x}_2 + c_2(\dot{x}_2 - \dot{x}_1) + k_2(x_2 - x_1) = 0 \end{cases} \tag{2.10}$$

where F is the amplitude, ω the frequency of the force acting on the system and t the time. The displacement of the primary system is x_1 and x_2 for the absorber.

The main observation of Den Hartog was that two invariant points of the primary system displacement exist regardless of the damper value. The idea of this method is

to find the absorber stiffness that allows to have two peaks of equal amplitude. Indeed, this configuration is the optimal one. The device also works if a damper is added to the primary system but the expressions become more complex because the invariant points disappear. The response can still be minimized with more complex analytic expressions of the parameters. It can be noticed that the exact close form solution has been expressed many years later by T. Asami and O. Nishihara (2003, [13]). For the non damped linear system, it is stated as:

$$\begin{aligned} \lambda = \frac{w_{n1}}{w_{n1}} &= \sqrt{\frac{k_2 M_1}{k_1 M_2}} = \frac{2}{1+\epsilon}\sqrt{\frac{2(16+23\epsilon+9\epsilon^2+2(2+\epsilon)\sqrt{4+3\epsilon})}{3(64+80\epsilon+27\epsilon^2)}} \\ \mu_2 = \frac{c_2}{2\sqrt{k_2 M_2}} &= \frac{1}{4}\sqrt{\frac{8+9\epsilon-4\sqrt{4+3\epsilon}}{1+\epsilon}} \end{aligned} \tag{2.11}$$

with ϵ being the mass ratio $\frac{M_2}{M_1}$. λ is the frequency ratio and μ_2 the damping ratio of the absorber. These ratios allow to minimize the maximum response of the primary system displacement.

This theory is shown in Fig. 2.6 provided by Habib and Kerschen (2016, [4]) where the black dotted line is the response of the system without any absorber. The three other curves show the response for several values of the damper absorber, including the optimal according to the Eq. 2.11 (in black). It highlights the presence of the invariant points represented by black dots.

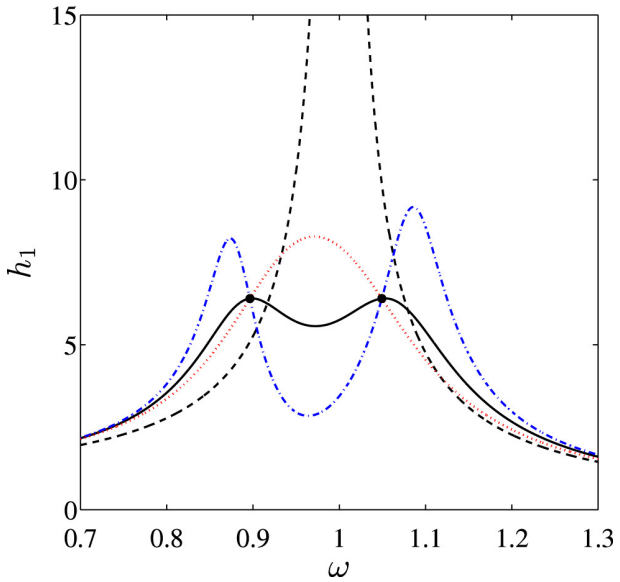


Figure 2.6: Frequency response function (h_1) of an undamped linear primary system with an attached linear vibration absorber from [4].

2.3.2 Nonlinear system

In the previous section, the principle of the LTVA has been explained. In this work, the purpose is to get an absorber for a nonlinear system. The first step is to analyze the reaction of the LTVA for a nonlinear system. The Duffing oscillator, a well known nonlinear

system (G. Duffing, 1918, [14]), is used. In addition to the linear spring, the primary system presents a nonlinear spring of stiffness k_{NL} and a damper c_1. The nonlinear spring is a cubic one and its reaction force is expressed as $k_{NL}x^3$. The absorber is attached to it as shown in Fig. 2.7.

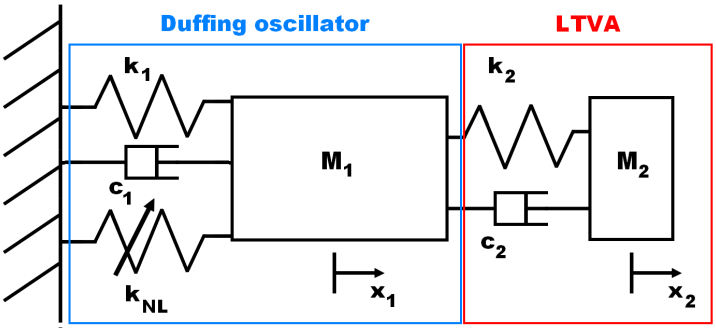


Figure 2.7: Scheme of the Duffing oscillator with LTVA.

When the solution of this system is computed, one can see that the response is detuned. It means that at least one of the two peaks is still nonlinear. This peak enters in resonance and it is not a convenient solution, the issue is just displaced. Thus, the linear absorber is ineffective for nonlinear system. This has been shown by Habib et al. (2015, [5]). Indeed, this detuned effect is visible in Fig. 2.8 where the right figure corresponds to a nonlinear system damped by a linear absorber. The different curves refer to different amplitudes in the excitation force which show the violation of the invariance principle.

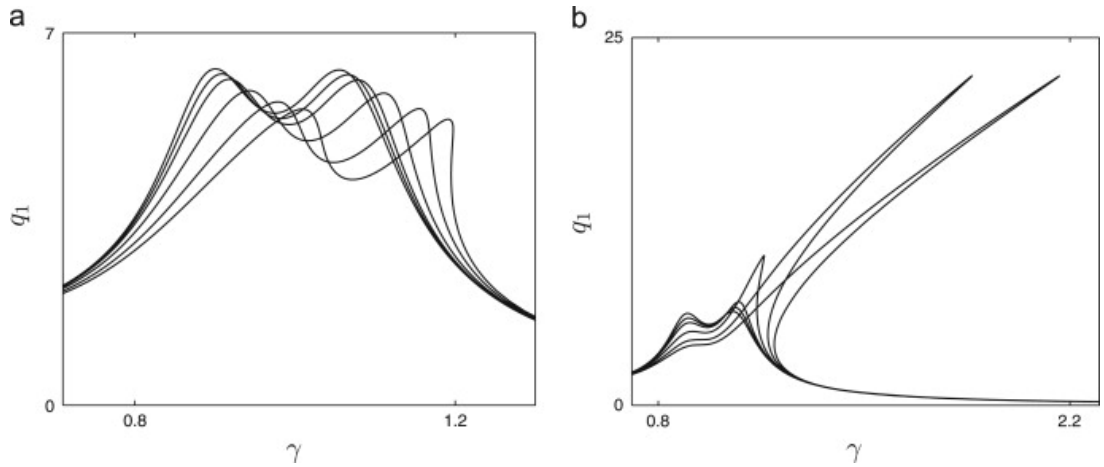


Figure 2.8: Damping of a nonlinear system using a) NLTVA; b) LTVA from [5]. q_1 is the dimensionless amplitude and γ the dimensionless frequency.

To create a nonlinear absorber, the equal peaks method has to be extended to the nonlinear system. Habib et al (2015, [5]) proposed to change the classical spring by a device which shows an arbitrary expression of its restoring force, expressed as $g(x)$. This device is called non linear vibration absorber (NLTVA) and can be schematized as in Fig. 2.9.

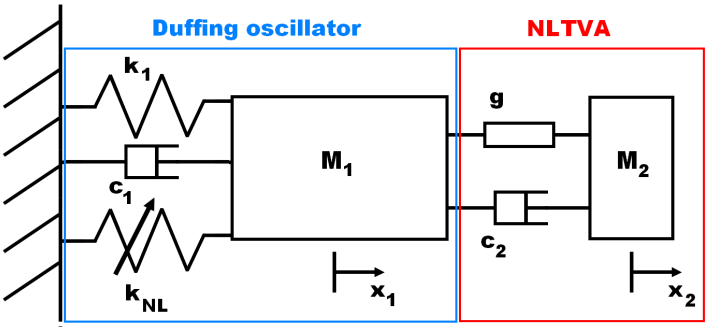


Figure 2.9: Scheme of the Duffing oscillator with NLTVA.

The equations of motion of the system can be easily derived:

$$\begin{cases} m_1\ddot{x}_1 + k_1x_1 + k_{nl}x_1^3 + c_2(\dot{x}_1 - \dot{x}_2) + g(x_1 - x_2) = F\sin\omega t \\ m_2\ddot{x}_2 + c_2(\dot{x}_2 - \dot{x}_1) + g(x_2 - x_1) = 0 \end{cases} \tag{2.12}$$

Some transformations are needed to get a solution. The study lead to an interesting tuning rules. It states that "*to choose the mathematical form of the NLTVA's restoring force so that it is a 'mirror' of the primary system*" (Habib et al, 2015, [5]). This means that for the Duffing oscillator, the restoring force of the absorber g should be a cubic spring. A linear spring should be also used to absorb vibrations at low regime where the nonlinear part is not excited.

As for the LTVA in Eq. 2.11, the parameters that minimize the response have been defined. It can be seen in Fig. 2.8 that the minimization is not unique. Indeed, for different load values, the two peaks change slightly in value and position. Nevertheless, the general form remains the same with two low peaks and seems to be a good solution.

2.4 Conclusion

Nonlinearities come from different sources but the most interesting for this work are the geometrical nonlinearities. The overall process to identify the nonlinearities has been presented. It is an important point since it allows to model any nonlinearities. To avoid resonance peak, vibration absorbers can be used. Firstly, these devices were focused on linear behaviour but Habib et al. managed to extend this theory to nonlinear case (2015, [5]). The key principle resulting from this study is the tuning rule. Indeed, using this new rule for model nonlinear absorber, topology optimization can be used to design nonlinear spring.

Chapter 3

Nonlinear topology optimization

3.1 Introduction

Topology optimization is a numerical method which allows to design the best material distribution for a specific feature under some constraints in a given domain. This is an interesting way to design anything. Indeed, it does not depend on predefined configuration like in a computer aided design (CAD) approach where topology modifications are impossible. The range of applications is very large because the type of optimization is multidisciplinary. However, it is mostly used in the conceptual stage of a design process. Indeed, exotic layouts are not always easy to manufacture. Recently, the additive manufacturing with 3D printers takes a lot of development and should allow the engineer to have much more freedom in the design as it is explained in many articles (T. Norton, 2013; S. Wasserman, [15], 2015, [16]). Another difficulty could be the intermediate density elements induced by some methods. These elements can have a big impact on the behaviour of the piece. Furthermore, they have no physical meaning and are impossible to manufacture. An example of topology optimization is the latest work of B. S. Lazarov et al (2018, [6]) on the heat sink for LED light sources. It highlights the multidisciplinary of the method because this is a thermal application. Moreover, it shows the new possibilities provided by additive manufacturing. Otherwise it would have been impossible to test it due to the complexity of the geometry shown in Fig. 3.1.

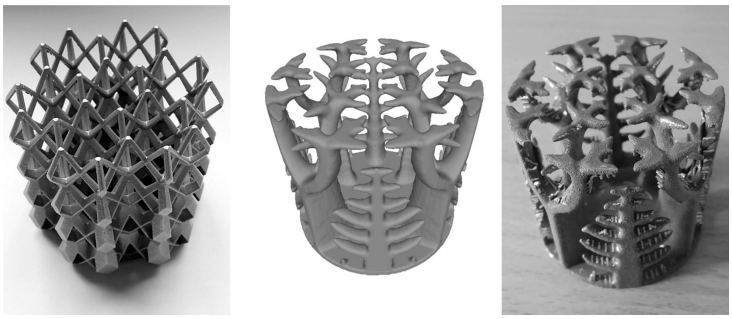


Figure 3.1: Example of thermal optimization and additive manufacturing from B.S. Lazarov et al (2018, [6]). From left to right: the industrial design solution, Topology-optimized LED heat sinks for horizontal orientations with 1/2 and 1/8 symmetries and 3D printed topology-optimized post-processed design with removed support.

The topology optimization method is based on the optimal material distribution. A fixed mesh will be used and the local density is the design variable. In order to have

an interpolation of the properties, the solid isotropic microstructure with penalization (SIMP) for intermediate densities or power law model is used (Bendsoe, 1989, [17]). For example in structural optimization, the property to interpolate is the Young modulus as shown in Eq. 3.1.

$$E_e = x_e^P(E_0 - E_{min}) + E_{min} \tag{3.1}$$

where E_e is the Young modulus of one mesh element, x_e its density, P the penalization and E_0 the reference Young modulus of a full density element.
The purpose of the penalization is to minimize the importance of the low density element and avoid the intermediate density one.

As it has been mentioned before, the optimization objective is to fit a prescribed force/displacement curve based on geometrical nonlinearities. Nonlinearities change a lot the optimization because many applications are made for linear case and do not care about large displacements. It has already been the topic of some studies by T. Buhl, C.B.W. Pedersen, O. Sigmund (2000, [18]) and T. E. Bruns and D. A. Tortorelli (2001, [19]). As a starting point, the open source code written by Sigmund et al (2010, [20]) is used. This code is known as "Efficient topology optimization in MATLAB using 88 lines of code" and has become a reference in topology optimization. The modified codes for this nonlinear study are available in the annex C.

3.2 Nonlinear mechanics

First, the basics of nonlinear mechanics will be remembered. In this study, the exploited feature is the geometrical nonlinearity. The basic expressions are derived from the continuum model and will be extended to the discretized model in section 3.4. Initially, the elements are in position $\mathbf{X}$ as represented hereunder (Fig. 3.2).

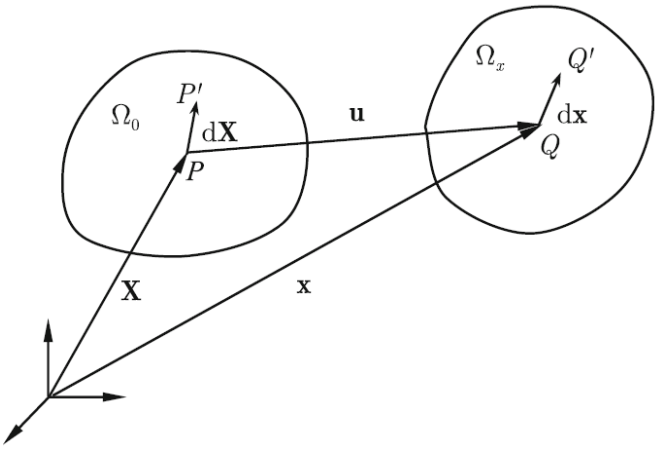


Figure 3.2: Deformation of a body from Ω_0 to Ω_x. The point P has only one related point in the converged shape (Q). Scheme from [1].

The global coordinates can be written as:

$$\mathbf{x} = \mathbf{u} + \mathbf{X} \tag{3.2}$$

where $\mathbf{u}$ and $\mathbf{x}$ are the global displacement and coordinates. From this expression, one can derive the displacement gradient $\nabla\mathbf{u}$:

$$\nabla\mathbf{u} = \frac{\partial\mathbf{u}}{\partial\mathbf{X}} \tag{3.3}$$

From this expression, the deformation gradient can be expressed as $\mathbf{F} = \nabla\mathbf{u} + \mathbf{I}$. The strain is related to the displacement gradient. If the Lagrangian strain presented in Eq. 2.1 is called $\mathbf{E}$, it comes:

$$\mathbf{E} = \frac{1}{2}\left(\frac{\partial\mathbf{u}}{\partial\mathbf{X}} + \frac{\partial\mathbf{u}^T}{\partial\mathbf{X}} + \frac{\partial\mathbf{u}^T}{\partial\mathbf{X}}\frac{\partial\mathbf{u}}{\partial\mathbf{X}}\right) = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T + \nabla\mathbf{u}^T\nabla\mathbf{u}) \tag{3.4}$$

If the assumptions of the infinitesimal strains are verified:

$$\varepsilon = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T) \tag{3.5}$$

Then, the stresses have to be defined. In nonlinear analysis, the second Piola-Kirchhoff stress tensor is often chosen:

$$\mathbf{S} = J\mathbf{F}^{-1}\sigma\mathbf{F}^{-T} \tag{3.6}$$

where $J = \det(\mathbf{F})$ and σ is the Cauchy strain tensor.

Cauchy and Piola-Kirchhoff stresses are equivalent in linear assumptions.

3.3 Model

It is also important to choose the constitutive model that will be used to perform the optimization. Since geometrical nonlinearities are the subject of this thesis, the model should be able to take into account large deformations. The most common one is the St-Venant Kirchhoff model. This one is used in this work and it takes form as Eq. 3.7. Some other models are used in the literature as the Neo-Hookean model used by M. Wallin et al (2018, [21]) which takes better care of compression. However the results are very similar.

$$\Phi = \frac{1}{2}\lambda E_{k,k}^2 + \mu E_{i,j}E_{i,j} \tag{3.7}$$

where Φ is the strain-energy density and λ and μ are the Lamé coefficient which depend on the Young modulus E and the Poisson's coefficient ν:

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)} \tag{3.8}$$

$$\mu = \frac{E}{2(1+\nu)} \tag{3.9}$$

By differentiating the strain-energy density, it comes the second Piola-Kirchhoff:

$$\mathbf{S} = \frac{\partial\Phi}{\partial\mathbf{E}} = \lambda\mathrm{tr}(\mathbf{E}) + 2\nu\mathbf{E} \tag{3.10}$$

3.4 Nonlinear Finite Element Method

In this work, the 2D case will be considered so the 4-nodes parametric element is used to discretize the geometry. The positions and displacements are interpolated using parametric shape function N_i. Thus, the nonlinear finite element method (FEM) procedure has to be explained. It is not as simple as the linear case where the equilibrium equation $\mathbf{Ku} = \mathbf{F}$ is direct. In the nonlinear case, the problem has to be solved iteratively passing through several equilibrium states. Indeed, large displacements induce some geometrical nonlinearities that will be exploited later but it complicates a the procedure. First, some theories have to be remembered. Note that the complete explanation can be found in "Introduction to Nonlinear Finite Element Analysis" (2015, [1]). Thus, one has to discretize the model. The displacements and coordinates are re-expressed using interpolation as follows:

$$\mathbf{u} = \sum_{I}^{N} N_I \mathbf{u}_I \tag{3.11}$$

$$\mathbf{x} = \sum_{I}^{N} N_I \mathbf{x}_I \tag{3.12}$$

where $\mathbf{u}_I$ are the displacements of the element's nodes and $\mathbf{x}_I$ its coordinates. With this model, if N_I are the shape functions, it comes:

$$\nabla \mathbf{u} = \frac{\partial \mathbf{u}}{\partial \mathbf{X}} = \sum_{i}^{N} \frac{\partial \mathbf{N}_i}{\partial \mathbf{X}} \mathbf{u}_i \tag{3.13}$$

These discretized values allow to compute the Lagrangian strain $\mathbf{E}$. Then, the variation of Lagrangian strain can be found:

$$\delta \mathbf{E} = \mathbf{B} \delta \mathbf{u} \tag{3.14}$$

where $\mathbf{B}$ is the nonlinear strain-displacement matrix that can be decomposed into two parts:

$$\mathbf{B} = \mathbf{B}_L + \mathbf{B}_N \tag{3.15}$$

It allows to find the expression of the internal force as in Eq. 3.16.

$$\mathbf{f}_{int} = \sum \int_{\Omega_e} \mathbf{B}(\mathbf{u}) \mathbf{S} \partial v \tag{3.16}$$

The FEM analysis is based on a Newton-Raphson scheme, the internal force is computed from the previous expression and a residual can be expressed:

$$\mathbf{r} = \mathbf{f}_{ext} - \mathbf{f}_{int} \tag{3.17}$$

The purpose of the Newton-Raphson scheme is to set the residual to 0. To do this, the tangent stiffness matrix has to be used:

$$\mathbf{K}_T = -\frac{\partial \mathbf{r}}{\partial \mathbf{u}} \tag{3.18}$$

Then thanks to the Taylor expansion, it can be linearized and the displacement increment $\Delta\mathbf{u}^i$ can be found easily:

$$\mathbf{K}_T \Delta\mathbf{u}^i = \mathbf{r} \tag{3.19}$$

where the stiffness tangent matrix is found by:

$$\mathbf{K}_T = \sum \int_{\Omega_e} (\mathbf{B}_N^T \mathbf{D} \mathbf{B}_N + \mathbf{B}_G^T \mathbf{Y} \mathbf{B}_G) \partial v \tag{3.20}$$

where $\mathbf{B}_G$ contains the shape functions' gradients, $\mathbf{D}$ is the constitutive matrix and $\mathbf{Y}$ contains the second Piola-Kirchhoff stresses. This allows to iteratively approach the equilibrium by updating the displacement vector as in Eq. 3.21. The internal force can be updated and afterwards the new residual. The procedure will run until the convergence is reached.

$$\mathbf{u}^{i+1} = \mathbf{u}^i + \Delta\mathbf{u}^i \tag{3.21}$$

The main problem of the Newton-Raphson scheme is its convergence. This one depends a lot of the initial conditions which have to be near the solution. This way, the scheme presents a quadratic convergence. To control the convergence, two options are available either by controlling the load or the displacement.

Load control

In load controlled procedure, the applied force on the system is gradually incremented until the wanted value. For each increment, the Newton-Raphson procedure is performed to find the intermediate equilibrium. Then, the intermediate displacement is used has a guess for the next increment. The load step can be either linear or not but by simplicity, it has been chosen linear in this work. The procedure scheme is shown in Fig.3.3. Between each load increment, the Newton-Raphson method is used to find the related displacement.

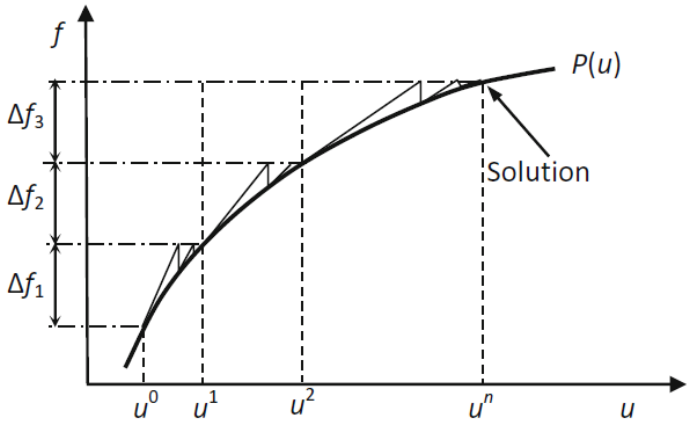


Figure 3.3: Example of load-controlled nonlinear FEM procedure from [1]. f is the external force and $P(u)$ the internal one.

Displacement control

As it was the case for the load, the displacement can also be controlled to find the internal force related. Since there is a relation between the displacement and the force, the result

should be the same. Practically, the displacement controlled procedure could be more stable (N.H. Kim, 2015, [1]).

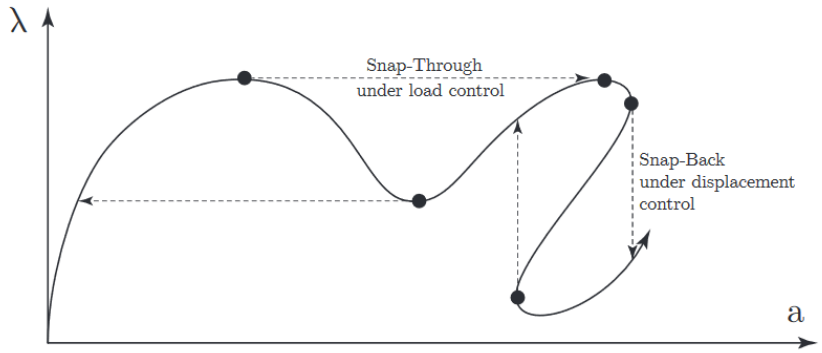


Figure 3.4: Example of complex force/displacement curve (2015, [7]). λ corresponds to the force and a the displacement.

Nevertheless, for each method, some behaviours are impossible to design with such procedure. As it is shown in Fig.3.4, snap-through is impossible to perform under load control. The same conclusion is available for the snap-back under displacement control. According to the nature of the problem one should be preferred to the other. In a first instance, the purpose is to fit simple curve, thus any of these behaviours will be studied in this work.

3.4.1 Validation

To verify the validity of the implemented code, the result of an analysis are compared to theoretical result. Since it will be the domain use after, the cantilever beam is chosen. The simplified deflection expression (2007, [22]) of a cantilever beam writes:

$$f = \frac{PL^3}{3EI} \tag{3.22}$$

where P is the load, L the length of the beam, E its Young modulus and I the inertia of the section. A simple rectangle is thus considered:

$$I = \frac{bh^3}{12} \tag{3.23}$$

where h is the high of the section and b its base.

For the numerical solution a 20×80 mesh is used. The Young modulus E is set to $1[Pa]$ and the force applied goes until $1e-4$[N]. The resulting force/displacement curves are plotted in Fig. 3.5. It shows that the FEM analysis seems to fit the theoretical approximation. Indeed, the mean error is about 2.6%, this allows to validate the model.

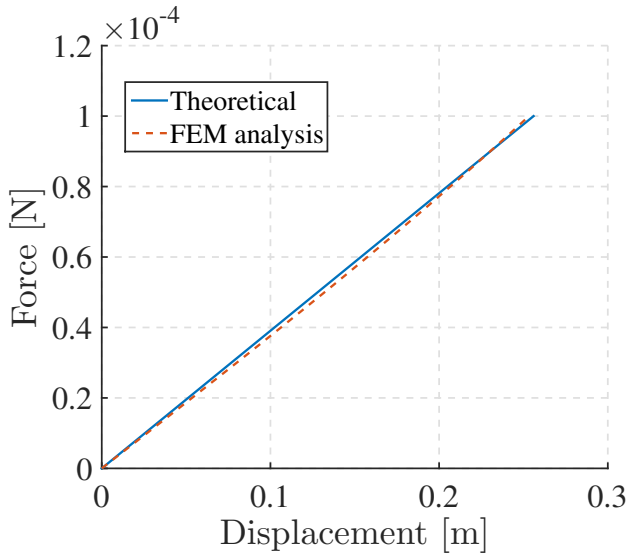


Figure 3.5: Comparison of theoretical and numerical deflection of a cantilever beam.

3.4.2 Stop criteria

The relative error is taken as a convergence criterion. While the maximum absolute value of the normalized residual vector is bigger than an error threshold, the Newton-Raphson scheme will continue to update the variables:

$$\frac{\max(\mathbf{R_{i,j}})}{F_i} \geq e_{limit} \tag{3.24}$$

where $R_{i,j}$ is the residual at the j^{th} iteration of the Newton-Raphson scheme and the i^{th} of the control iteration, F_i is the load related and e_{limit} is the threshold error.

Some threshold limits have been tried in order to find the best compromise between time and accuracy. To do this, the previous example of the cantilever beam loaded until $1e-4$[N] is chosen. As a reference, a threshold error about $1e-10$ is taken. To compare the threshold errors, the displacement at the tip of the beam, the time and the mean number of iterations are measured. The results are shown in Tab. 3.1. According to these values, the most interesting threshold error is about $1e-3$. Indeed, it does not take more time to compute compare to the bigger and give the same solution than the smaller. One can note that the values of the displacement were the same in this simple case and the error compare to the reference is very small.

e_{limit}	Time [s]	u_{tip}	Iterations
1e-2	15.311	0.2532950078	3
1e-3	15.818	0.2532950078	3
1e-4	20.952	0.2532950078	3.2
1e-10	31.748	0.25329500517	5

Table 3.1: Comparison of the iterative parameters for different threshold errors.

3.4.3 Step size verification

The nonlinear FEM analysis is a slow process due to its iterative nature. Therefore it is important to choose the right step for the incremental procedure. This is an important parameter for the convergence and the speed of the code. Few have been tested for the load incremental procedure. The benchmark is the same than previous and the cantilever beam has a 20×80 mesh. The force is increased from 0 to $1e-4$[N] by ΔF and the results are shown in Tab. 3.2.

ΔF [N]	Steps	Time [s]	Iterations
2e-5	5	8.6	4
1e-5	10	14.59	3
0.667e-5	15	22.99	3
0.5e-5	20	35.25	3

Table 3.2: Comparison of the iterative parameters for different step sizes.

In this table, the time has been taken into account but also the number of internal iterations i.e., the number of iterations taken by the Newton-Raphson scheme to meet the stop criteria from Eq. 3.24. This parameter is interesting because it quantify the distance between the guess and the solution. It shows that even if the first time step is faster, it takes more iterations to perform the analysis. It is not a problem here but for more complex problems induced by topology optimization, it can be very important to be close enough. For the three other steps, the number of iterations is the same. Thus the time factor becomes more important and the choice is made on $1e-5$[N]. This step has been used during all simulations and was not taken in default. One can note that for the nonlinear analysis, parallel functions of MATLAB have been used such as `parfor`. On my computer, it allows to use two workers. It obviously increases the speed. For example, for the chosen time step, the time taken is reduced by 20%. The specifies of the computer used for this study are referred on Appendix A.

3.5 Compliance optimization

To validate the model, it is interesting to try it with the most common example of topology optimization, i.e. the compliance minimization. This problem allows to find the stiffest shape for the given design domain. This well known problem takes form as in Eq. 3.25. It is important to notice that the load control has been chosen because of the form of the problem where the force is fixed.

$$\begin{aligned} \min \quad & f_0(x) = \mathbf{f}_{ext}^T \mathbf{u} \\ \text{s.t.} \quad & \mathbf{r} = \mathbf{0} \\ & \mathbf{v}^T \mathbf{x} \leq v^* \\ & \mathbf{x}_{min} \leq \mathbf{x} \leq \mathbf{x}_{max} \end{aligned} \tag{3.25}$$

where $\mathbf{u}$ is the generalized displacement, $\mathbf{f}_{ext}$ the external force, $\mathbf{r}$ the residual and v^* the maximum volume accepted.

The best optimizer to use is the method of moving asymptote (`MMA`) proposed by Svanberg (1998 [23]) which is a gradient-based algorithm. Thus, the sensitivities of the cost function and the constraint are needed.

3.5.1 Parameters

The next step is to choose the right parameters of `MMA`. Indeed, an adequate selection of parameters allows to run the optimization and validates this model. Given the classical use of the solver (see Appendix B), the parameters are the usual ones as proposed by Svanberg (1998, [23]):

$$\begin{aligned} a_j &= 0 && j = 1...m \\ d_j &= 0 && j = 1...m \\ c_j &= \text{big} && j = 1...m \end{aligned}$$

where m is the number of constraints.

After some verifications, the value of c has been set to 10000 which is a classical value found in the literature (Svanberg, 1994, [23]). It should not be too big because it could ill conditioned the problem. Some other techniques can be implemented in order to get the best solution and avoid convergence problem. For this model, a filter, a move limit, a continuation method and a projection have been chosen. This problem can be just seen as a displacement minimization and then, in the expression of the problem Eq. 3.25, $\mathbf{f}_{ext}$ can be replaced by $sign(\mathbf{f}_{ext})\mathbf{l}$ where $\mathbf{l}$ is the localization vector of the force.

3.5.2 Filter

To avoid checkerboard patterns, the use of filters is recommended. Many filters exist and have been tested (O. Sigmund, 2007, [24]) but none of them is perfect. Even if it is not the best, a density filter has been chosen, it takes form as in Eq. 3.26. From this paper, it seems to make less iterations than other method which is a real issue in nonlinear optimization considering the time taken by each iteration.

$$\bar{x}_e = \frac{1}{\sum_{i \in N_e} H_{ei}} \sum_{i \in N_e} H_{ei} x_i \tag{3.26}$$

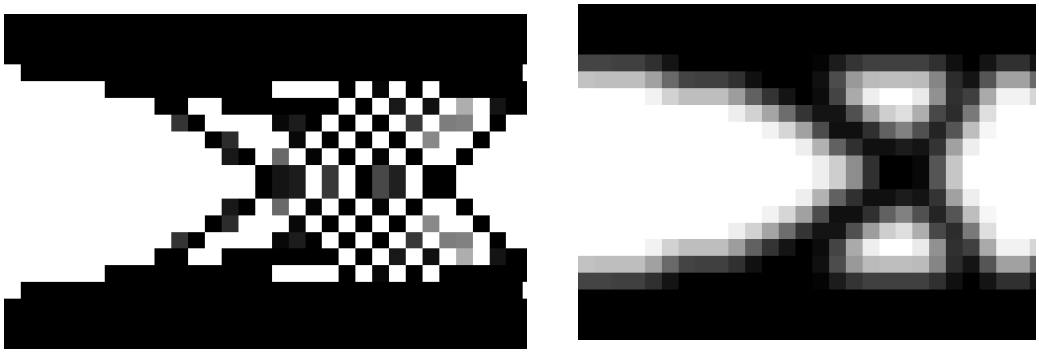
where $\bar{x}_e$ is known as the physical density. N_e is the set of elements for which the distance with the element e is less than the filter radius r_{min}. H_{ei} is a weight factor and can be expressed as:

$$H_{ei} = max(0, r_{min} - \Delta_{ei}) \tag{3.27}$$

This new variable is used to compute the sensitivities and thus it slightly changes their expressions as shown below for the objective function:

$$\frac{\partial c}{\partial x_j} = \sum_{e \in N_j} \frac{\partial c}{\partial \bar{x}_e} \frac{\partial \bar{x}_e}{\partial x_j} = \frac{1}{\sum_{i \in N_e} H_{ei}} \sum_{e \in N_j} H_{je} x_i \frac{\partial c}{\partial \bar{x}_e} \tag{3.28}$$

The effect of the filter can be seen in the next figure. In Fig. 3.6a, the checkerboard patter is obvious and one can understand that it has no physical sense in reality.



(a) Without filter, checkerboard pattern.

(b) With filter, $r_{min} = 2$.

Figure 3.6: Effect of the filter on a cantilever beam.

3.5.3 Move limit

Another defensive way to ensure convergence is the move limit. It sets a box limit to the design variable in order to avoid large change and oscillating convergence. It will slightly slow down the code since the limit is chosen between the box and the side constraints:

$$\max(x_{min,i}, x_i - \alpha) \leq x_i \leq \min(x_{max,i}, x_i + \alpha) \tag{3.29}$$

Usually the size of the box α is chosen between 0.15 and 0.3 according to the complexity of the problem. After some trials, 0.2 seems to be the best solution. A variable strategy can also be implemented but it does not work with a nonlinear analysis. Indeed, the variable strategy works well for some monotone optimizations. In nonlinear optimization, some elements are subject of sudden changes which are not well taken in charge by the variable strategy.

3.5.4 Continuation method

If the penalty is directly set to its highest value, the solution can present some numerical difficulties like being trapped in a local optima. In order to smooth the layout, a continuation procedure is implemented as Wallin (2018, [21]). The penalty begins lower (at 2) and is increased each 5 iterations by 0.1 until its maximum value (at 3).

3.5.5 Projection

One last useful part of the procedure is the projection. The purpose is to define a "0/1" distribution of density. It means that the distribution will tend to a perfect void/solid and intermediate densities will be avoided. Moreover, it allows to have more manufactarable layout. To perform this, an Heaviside projection can be used. F. Wang, B.S. Lazarov and O. Sigmund (2011, [25]) provide an efficient formulation for this projection:

$$\tilde{x}_e = \frac{tanh(\beta\omega) + tanh(\beta(\bar{x_e} - \omega))}{tanh(\beta\omega) + tanh(\beta(1 - \omega))} \tag{3.30}$$

where $\bar{x_e}$ are the filtered densities and β, ω are the projection parameters.

When β tends to infinity, the projection gives a perfect distribution. A continuation method is implemented for this parameter to go from smoothed to strict difference between empty and full elements. It increases by 1 after 10 iterations until 10. The other parameter, ω, is set to 0.5 to have a mean projection has shown in Fig.3.7. The elements above ω are projected towards 1 and the elements under towards 0. This projection is not submitted to any constraints and little violation of the volume constraint are induced when β changes. This change is often limited and solved by the optimizer itself. Thus, the volume is an interesting convergence parameter and is inspected along the iterations.

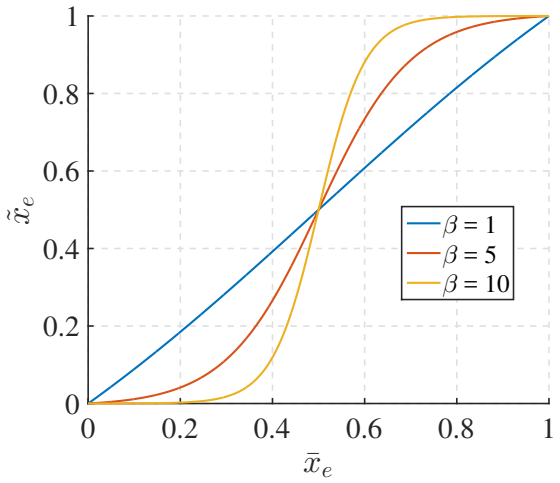


Figure 3.7: Effect of the Heaviside projection for different β with $\omega = 0.5$.

As for the density filter, the Heaviside projection induces few changes in the expression of the sensitivity:

$$\frac{\partial c}{\partial \tilde{x}_e} = \frac{\partial c}{\partial \bar{x}_e}\frac{\partial \bar{x}_e}{\partial \tilde{x}_e} \tag{3.31}$$

where the new term is derived from Eq. 3.30 with $sech(x) = \frac{1}{cosh(x)}$

$$\frac{\partial \bar{x}_e}{\partial \tilde{x}_e} = \frac{\beta sech(\beta(\bar{x_e} - \omega))^2}{tanh(\beta\rho_0) + tanh(\beta(1-\omega))} \tag{3.32}$$

The results obtained with the projection are better than without. Indeed, the full elements are stronger than the intermediate one. For the same linear optimization of a cantilever beam which is the subject of the next section, the compliance goes from 446.6 to 378.53.

3.5.6 Sensitivity computation

The cost function sensitivity can be computed, since the external force is constant, its derivative is null and it can be written as:

$$\frac{\partial f_0}{\partial x_e} = \mathbf{f}_{ext}^T \frac{\partial \mathbf{u}}{\partial x_e} \tag{3.33}$$

However, in this form, it will be computationally too expensive. Indeed, the displacement has to be computed for a variation of each design variable. It means that n FEM

analysis should be done, which is too costly. Instead, the Lagrangian augmented cost function is introduced in order to use the adjoint method.

$$f_0 = \mathbf{f}_{ext}^T \mathbf{u} + \lambda^T \mathbf{r} \tag{3.34}$$

where λ is the Lagrangian multiplier and $\mathbf{r}$ the residual vector. This expression is obviously the same than the initial since $\mathbf{r} = \mathbf{0}$. If it is this expression that is derived, it comes:

$$\frac{\partial f_0}{\partial x_e} = \mathbf{f}_{ext}^T \frac{\partial \mathbf{u}}{\partial x_e} + \lambda^T \left(\frac{\partial \mathbf{r}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial x_e} + \frac{\partial \mathbf{r}}{\partial x_e}\right) \tag{3.35}$$

which could be rewritten:

$$\frac{\partial f_0}{\partial x_e} = \left(\mathbf{f}_{ext}^T + \lambda^T \frac{\partial \mathbf{r}}{\partial \mathbf{u}}\right) \frac{\partial \mathbf{u}}{\partial x_e} + \lambda^T \frac{\partial \mathbf{r}}{\partial x_e} \tag{3.36}$$

Since the Lagrangian multiplier can be chosen arbitrarily, it should be taken in order to eliminate the displacement sensitivity which is computationally very heavy. From Eq. 3.36 and if one remember the expression of the tangent matrix $\mathbf{K}_t = -\frac{\partial \mathbf{r}}{\partial \mathbf{u}}$ it comes:

$$\lambda = -\frac{\partial \mathbf{r}}{\partial \mathbf{u}}^{-1} \mathbf{f}_{ext} = \mathbf{K}_t^{-1} \mathbf{f}_{ext} \tag{3.37}$$

Then, the sensitivity of the residual vector has to be computed, the residual is defined as the difference between external and internal force $\mathbf{r} = \mathbf{f}_{ext} - \mathbf{f}_{int}$ and as it has been mentioned before, the external force is constant so the residual sensitivity becomes:

$$\frac{\partial \mathbf{r}}{\partial x_e} = -\frac{\partial \mathbf{f}_{int}}{\partial x_e} \tag{3.38}$$

This equation can be derived from the constitutive equation of the internal force (Eq. 3.16):

$$\frac{\partial \mathbf{r}}{\partial x_e} = -\frac{\partial}{\partial x_e} \int_V \mathbf{B}^T \mathbf{S} \, \mathrm{d}V = -\frac{\partial}{\partial x_e} \int_V \mathbf{B}^T x_e^p \mathbf{D} \varepsilon \, \mathrm{d}V = \int_V \mathbf{B}^T p x_e^{p-1} \mathbf{D} \varepsilon \, \mathrm{d}V \tag{3.39}$$

The complete procedure can be explicitly expressed in the form of a flow chart. This allows a better understanding of the method, it is represented in Fig. 3.8. It clearly shows that the process contains two parts: the nonlinear FEM and the topology optimization. The FEM analysis is presented in blue while the green is the topology optimization step.

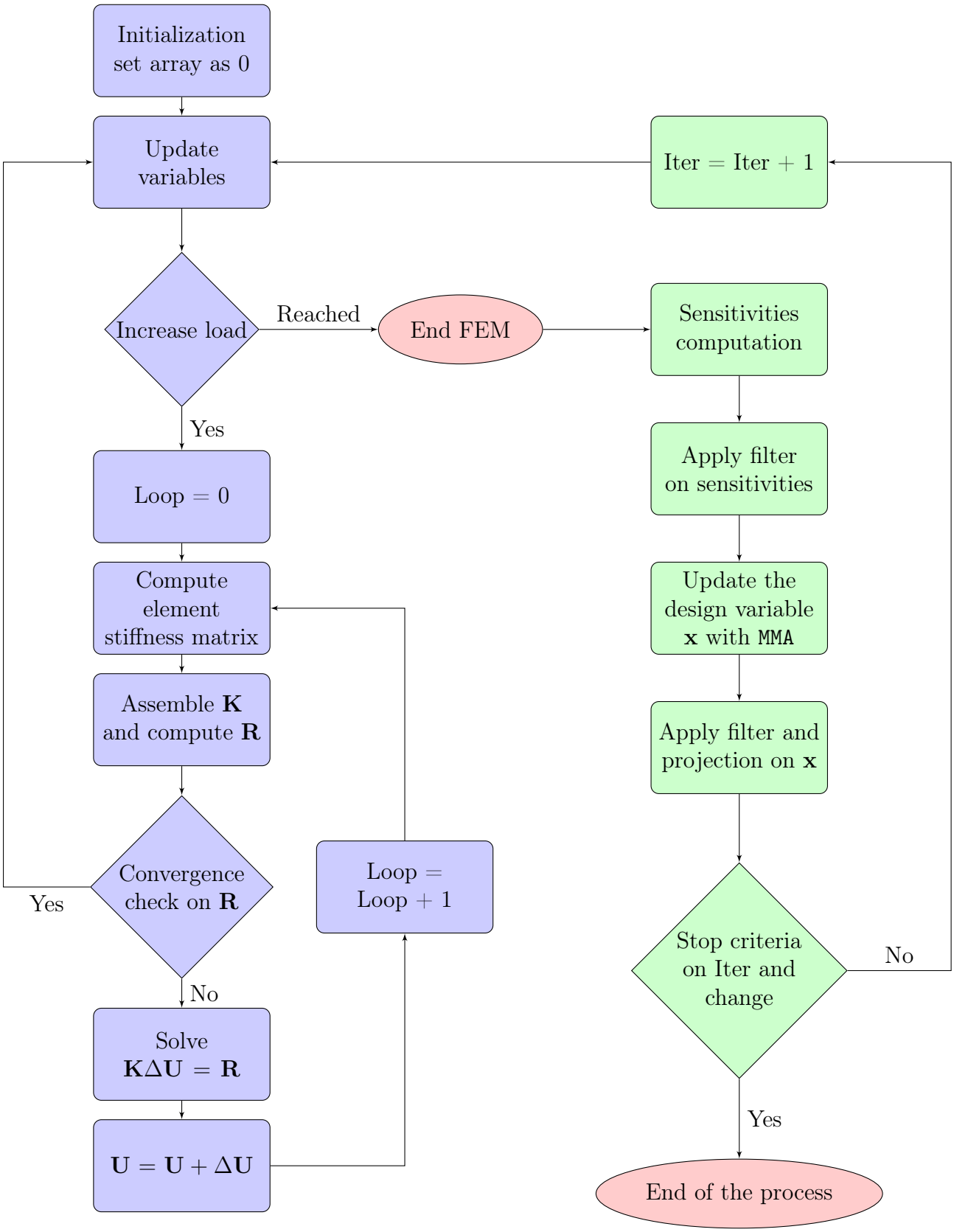


Figure 3.8: Flowchart of the overall procedure.

3.6 Cantilever beam

The cantilever beam is a usual problem in topology optimization, it is interesting to see how the nonlinear optimization differs from the linear one. Physically, this problem takes form as in Fig. 3.9 where the material and dimensional parameters have to be fixed. The practice in topology optimization is to choose a material that presents a unite Young modulus $E_0 = 1[Pa]$ and $nu = 0.4$. For nonlinear problem, some studies (O. Sigmund et al., 2000, [18]; B.S. Lazarov et al., 2014, [26]) use Nylon ($E_0 = 3[GPa]$ and $\nu = 0.4$) but it does not change the layout. The left of the beam is totally clamped and a force F is applied at the middle of its right side as shown in Fig. 3.9. Arbitrarily, the dimensions are taken to have 4:1 ratio ($a = 1$[m] and $b = 0.25$[m]). This is discretized by a 80×20 elements beam. The volume fraction is taken to 0.5 and the radius filter taken is 3.

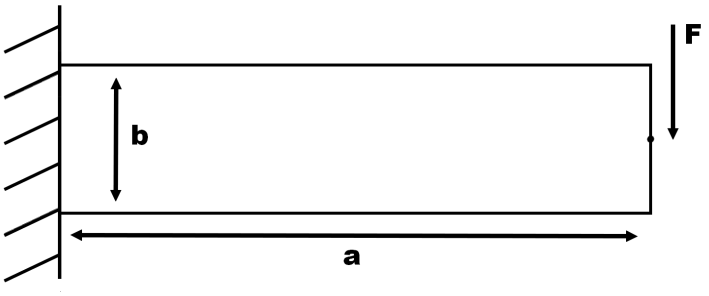


Figure 3.9: Load case of the cantilever beam.

The optimal layout is going to change for different values of the applied load. To highlight this, the optimization will be run for two different loads F_1 and F_2.

$$F_1 = 3e - 5[\text{N}]$$
$$F_2 = 5e - 5[\text{N}]$$

3.6.1 Linear optimization

The first step to have a standard layout is to perform a linear optimization on the domain. All the parameters and methods are the same than for the nonlinear optimization. The resulting layout is shown in Fig. 3.10.

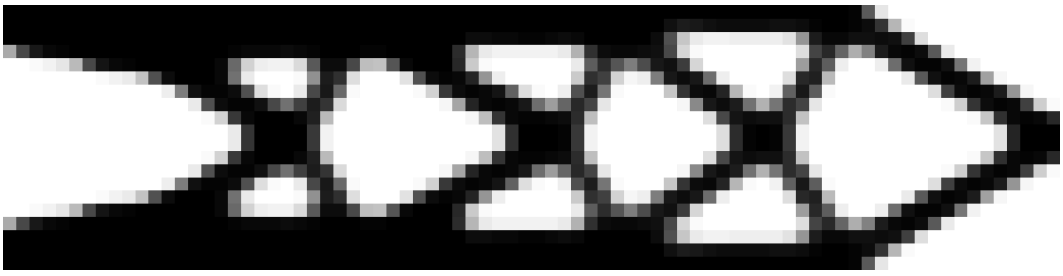


Figure 3.10: Optimal layout with linear analysis.

Unsurprisingly, this layout is symmetric and independent of the load intensity. A nonlinear FEM analysis is performed to find the displacement for the two loads. It comes that displacements are equal to 0.1192 [m] and 0.1963 [m] for F_1 and F_2 respectively.

3.6.2 Sensitivity validation

In order to validate the use of the adjoint method, the sensitivity should be computed in another way. An easy but inefficient method is the central finite difference method that could be written as Eq. 3.40. This method computes the objective function for a specific perturbed element e.

$$\frac{\partial f_0(x)}{\partial x_e} = \frac{f_0(\mathbf{x} + \Delta\mathbf{x}^e) - f_0(\mathbf{x} - \Delta\mathbf{x}^e)}{2\Delta x_e} \tag{3.40}$$

where $\mathbf{x}$ is the vector of the optimised density, Δx_e is the perturbation imposed on the element e and $\Delta\mathbf{x}^e$ the vector containing the perturbation at the right index.

Due to the truncation and the condition errors, the step size of the perturbation has to be chosen carefully. The effect of the step size has been studied by J.Iott, R. Haftka, H.M.Adelman (1985, [27]). It comes that the truncation error increases with Δx_e while the condition error decreases and vice-versa. According to this, the value Δx_e equal to 0.001 proposed a good compromise and consequently is used in this work. To compare the result of each method, four elements are arbitrarily chosen. Their sensitivities $\frac{\partial f_0(x)}{\partial x_e}$ are computed according to each method. This procedure is done after the first iteration. The results are available in Tab. 3.3.

Element	Finite difference sensitivity	Adjoint sensitivity	Relative error [%]
150	-1.7051e-4	-1.7059e-4	0.045
460	-2.3635e-3	-2.3577e-3	0.2453
970	-1.3542e-4	-1.3523e-4	0.1403
1550	-3.8110e-4	-3.8161e-4	0.1331

Table 3.3: Comparison between finite difference and adjoint method sensitivies.

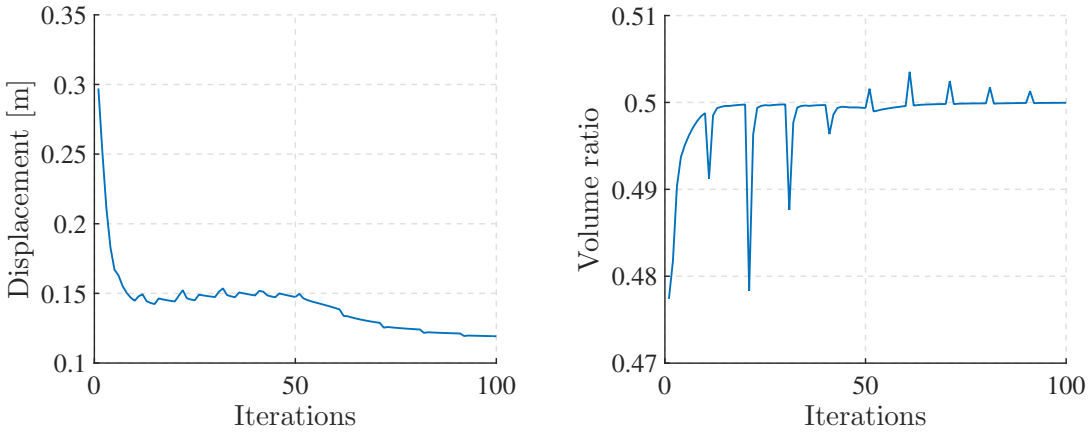
These results validate the method of adjoint sensitivities. Indeed, the relative error does not exceed 0.25% on each element which is low enough.

3.6.3 Results

The first optimization is performed with the load F_1 and it leads to the layout represented in Fig. 3.11. It is quite similar to the linear one but the longitudinal symmetry does not exist anymore. Indeed, it makes sense since it is made for one force direction. Thus the solver tends to stiffen especially this side.

Figure 3.11: Optimal layout for F_1 with nonlinear analysis.

Furthermore, the optimized displacement (since $C = \mathbf{F}^T\mathbf{U}$ and the force is constant) has a good convergence as it is shown in Fig. 3.12a. It begins very high for the uniformly reparted beam but it drops very fast as soon as the optimization begins. Then, it alternates some jumps when the penalty and/or the β parameter increase for the continuation method with some slight decreases during the optimization. It is a classical shape for the optimized variable. To verify the convergence, the volume ratio has been plotted in Fig. 3.12b. It can be seen that there is a jump when β changes but it is directly corrected by the solver. The final value after 100 iterations is equal to 0.1192[m]. It means that the nonlinear layout does not present an improvement compared to the linear case. A FEM analysis can be done on the new design until F_2. It shows that the behaviour is quite linear and the final displacement is equal to 0.1957[m]. For this second load, a slight decrease of the displacement is seen.



(a) Displacement.

(b) Volume.

Figure 3.12: Optimized displacement and volume as a function of the number of iterations for F_1 .

For the second test case, the load is increased to F_2. This time, the layout has completely changed compared to the linear reference and the previous shape. As shown in Fig. 3.13, it is very asymmetric. At a first sight, this shape may seem to be non stiff at all with the solo bar end. The optimized displacement after 100 iterations is about 0.1955[m] which is slightly smaller than the previous shape for the same load. A FEM analysis can be performed on this geometry and the displacement at F_1 can be found, it comes that is is about 10% higher than previously with 0.133[m].

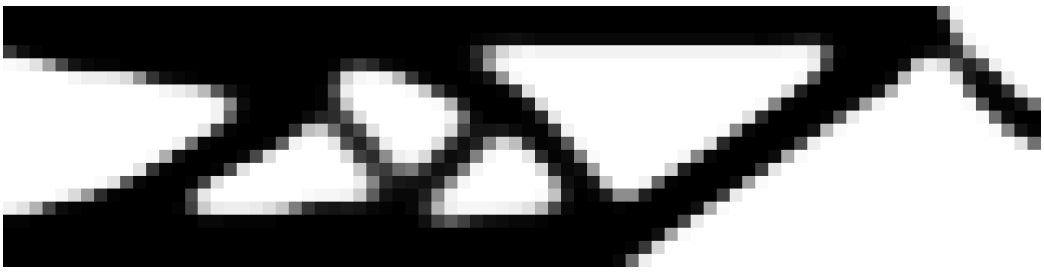
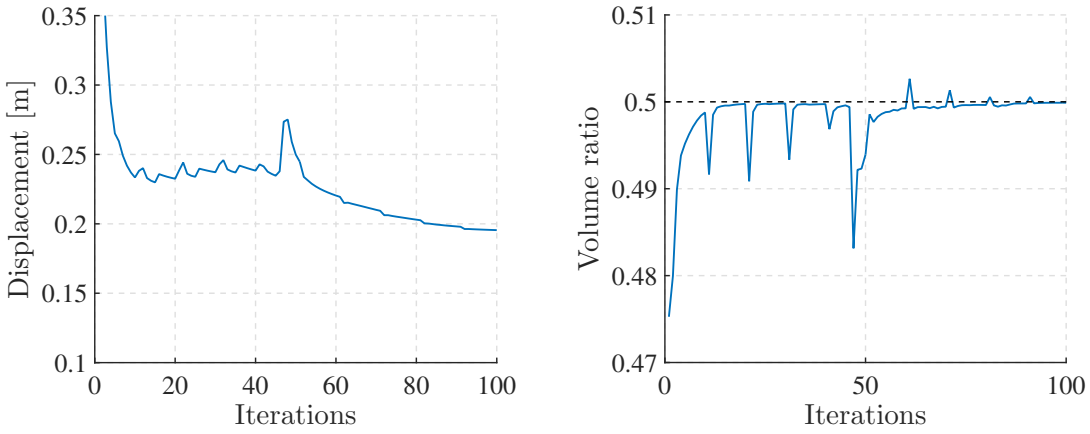


Figure 3.13: Optimal layout for F_2 with nonlinear analysis.

Then if one takes a closer look to the deformed shape in Fig. 3.15, the bar-end appears to be vertical which is the best way to take a charge with minimum displacement. It makes sense but it also means that the layout is optimized only for one precise load since the bar-end will be in bending in other case. In this same figure, the mesh, also plotted, seems to be distorted under the bar-end. Moreover, it is important to notice the presence of a big jump in the evolution of the optimized variable in Fig. 3.14a around the 50^{th} iteration. It corresponds to the biggest jump in the volume ratio of the beam as Fig.3.14b shows. With some inspections between each iteration, it comes that this jump occurs when the mesh first crashes. This issue is common in nonlinear optimization and will be investigated in section 3.7.



(a) Displacement.

(b) Volume.

Figure 3.14: Optimized displacement and volume as a function of the number of iterations for F_2 .

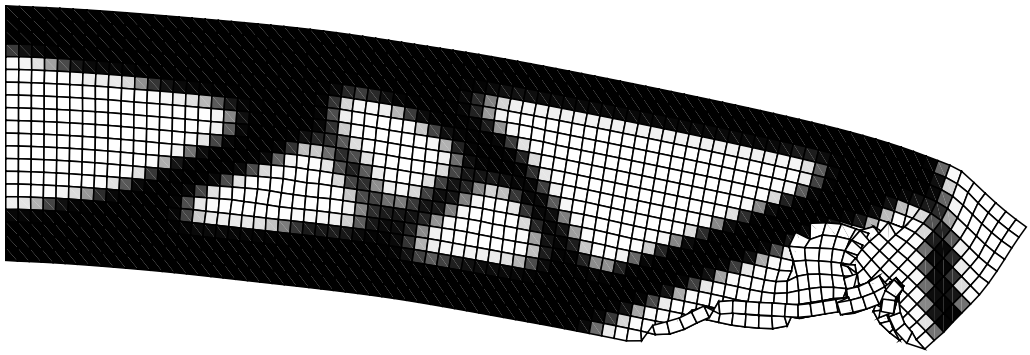


Figure 3.15: Deformed shape for F_2.

3.7 Mesh distortion

For the compliance problem of a cantilever beam, it has been shown that some crashes happen in the mesh. Obviously, this can be a problem for the convergence of the opti-

mization. Thus it is important to implement some techniques to avoid crashes. Two of them will be examined in the next section.

3.7.1 Energy interpolation

As mentioned before, a common problem that comes with nonlinear optimization is the mesh distortion. This issue is due to low density elements which crash under large displacements. It is a purely numerical problem since physically low density elements represent void. The problem causes by large displacements on these elements is that their tangent stiffness can be zero or negative. Thus, it can induce the non-convergence of the equilibrium iteration. The problem was already known and some solutions have been investigated like the relaxation of the convergence criteria during the Newton-Raphson scheme (Buhl et al., 2000, [18]; Pedersen et al., 2001, [28]). This method ignores the instability but the number of iterations is still very high and the results can be inaccurate. Another developed method was the element removal and reintroduction method of the low density elements (Bruns and Tortorelli, 2003, [29]) which works quite well. More recently F. Wang et al. (2014, [26]) found a better way to avoid this problem, it is the one chosen for this work. The idea of their method is to interpolate the density energy stored in any element between the nonlinear energy from Eq. 3.7 and the infinitesimal strain energy: $\Phi_L = \frac{1}{2}\lambda\varepsilon_{k,k}^2 + \mu\varepsilon_{i,j}\varepsilon_{i,j}$. It can be noticed that for the best results, the Heaviside projection is needed in order to avoid intermediate elements. The interpolation is written as follows:

$$\Phi_e(\mathbf{u}_e) = [\Phi(\gamma_e\mathbf{u}_e) - \Phi_L(\gamma_e\mathbf{u}_e) + \Phi_L(\mathbf{u}_e)]E_e \tag{3.41}$$

where γ_e is the threshold parameter of the SIMP density of the element e such as it is equal to 1 when the element is solid an 0 if it is void. It is found using an Heaviside projection:

$$\gamma_e = \frac{tanh(\beta_1\rho_0) + tanh(\beta_1(\tilde{x}_e^p - \rho_0))}{tanh(\beta_1\rho_0) + tanh(\beta_1(1-\rho_0))} \tag{3.42}$$

where $\tilde{x}_e$ is the projected density. In the same paper ([26]) the two parameters β_1 and ρ_0 have been studied and it has been shown that the use of $\beta_1 = 500$ and $\rho_0 = 0.01$ seems to provide good convergence and these values can be used for many applications.

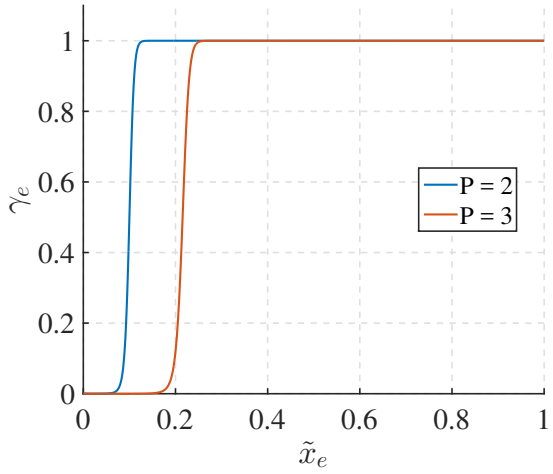


Figure 3.16: Threshold parameter γ_e as a function of the element density $\tilde{x}_e$ for different P.

The energy interpolation has been implemented in `MATLAB`. For load F_1, the shape changes compare to the previous optimization as shown in Fig. 3.17. Indeed, the load has to be bigger to observe a solo bar shape. It can be assumed that the mesh is more efficiently used with the interpolation. Indeed, when some mesh elements crash, they become useless and the solver is not able to fill them. These structure is called "Structure A".

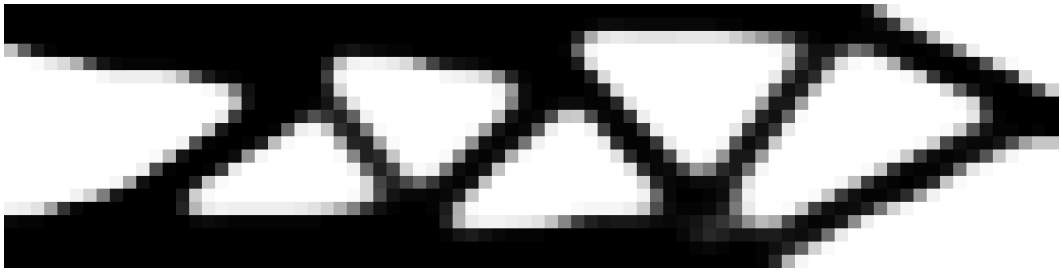


Figure 3.17: Optimal layout for F_2 with interpolation energy density. Structure A.

When the load is increased to $7e-5$[N], the shape in Fig. 3.18 presents the bar end. This second structure is named "Structure B". The deformed mesh is completely different since any element seems to crash even if these are still distorded. Consequently, this method is not perfect and if the displacement is too large, uncontrolled distortions or crashes will occur again. At the moment it seems to be the best solution for the SIMP model. Indeed, this energy interpolation method allows to optimize shape for bigger load. Even if it looks to made the optimization a slightly slower. The optimization is run for 100 iterations. A FEM analysis is performed on each layout and the results are shown in Tab. 3.4.

Load	$3e-5$[N]	$5e-5$[N]	$7e-5$[N]
Displacement [m] for *Structure* A.	0.1181	0.1917	0.2611
Displacement [m] for *Structure* B.	0.1328	0.1938	0.2524

Table 3.4: Comparison of the displacement for different design loads.

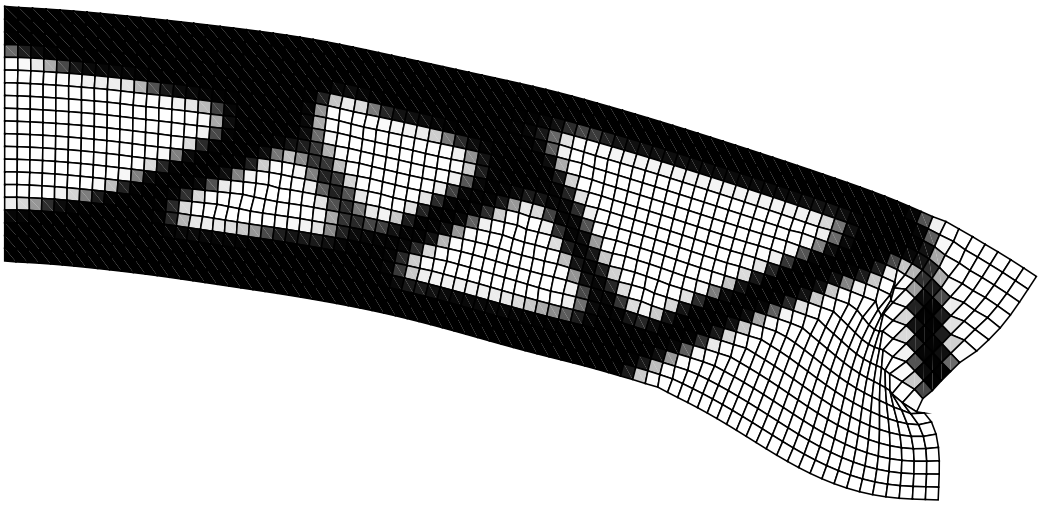
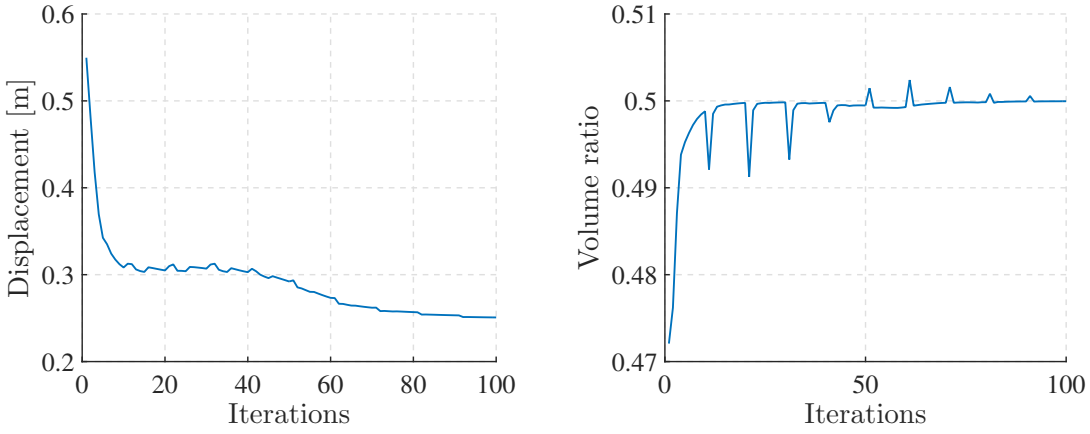


Figure 3.18: Deformed shape for 7e-5 [N] with interpolation energy density. *Structure B*.

One can notice that in addition of the crashed mesh, the displacement avoids big jumps in Fig. 3.19a. Note that some small jumps appear but these ones are due to the increment of β, like for the penalization. As expected, the volume ratio curve presents some variations when β changes but it is fastly solved as shown in Fig. 3.19b



(a) Displacement. (b) Volume.

Figure 3.19: Optimized displacement and volume as a function of the number of iterations for $7e-5$[N] with interpolation energy density.

3.7.2 Element connectivity parametrization

Energy interpolation is one of the best way to avoid non-convergence issues for the SIMP method. It is also possible to completely change the model and use the element connectivity parameterization (ECP) introduced by Yoon et al. (2005, [30]). Indeed, the mesh distorsion and instabilities that can induce non-convergence of the analysis are due to the low density elements. Based on it, Yoon et al. proposed a new type of modeling. Instead of using the density as a design variable, all the elements will be considered as solid. The

new model is based on the connection between these elements. They introduced the zero-length link between each element where the stiffness is variable. The new design variable γ will act on the stiffness. It means that a low density element in SIMP is modeled by a low stiffness link. Therefore, the tangent stiffness of the element does not lose its positive definiteness. Note that it can be done from two ways, either external (E-ECP) or internal (I-ECP) connections. The I-ECP is more often used in the mentioned literature and proposes interesting features as it is explained hereunder. In order to highlight the differences between the SIMP model and the I-ECP one, both have been schematically drawn in Fig. 3.20.

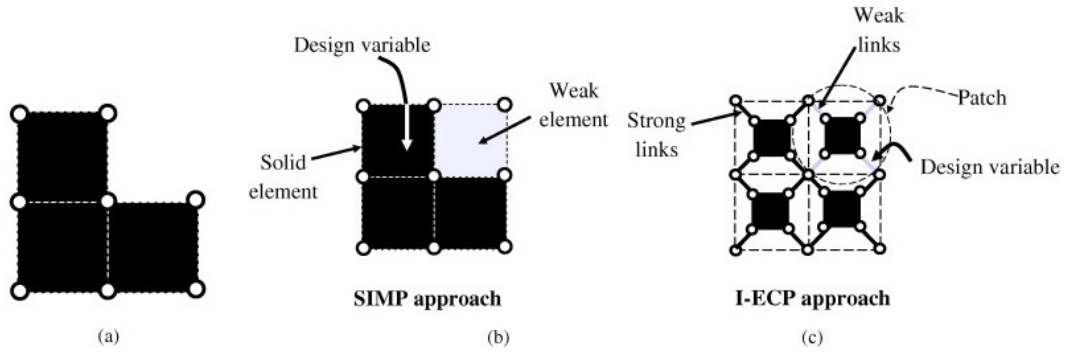


Figure 3.20: Comparison of the SIMP and I-ECP model (from [8]).

The drawback of the ECP method was the increase of variable and thus, the computational time. However, a few years later, Yoon et al. (2007, [31]) proposed a condensation method that allows to reduce the number of variables for the I-ECP method. Whit this, the size of the system is the same than for the SIMP method. This static condensation has been study for the linear case by Yoon et al. (2008, [32]) and still used for other type of nonlinear optimization by Yoon (2010, [8]). In addition of the computational gain, the condensation provides an easy way to represent the solution since the parameter γ can be used as the density to show the result.

3.8 Doubly clamped beam

Another common benchmark in topology optimization is the doubly clamped beam. This domain is also interesting for developing a nonlinear absorber. Thus, the same compliance test could be done for this design domain.

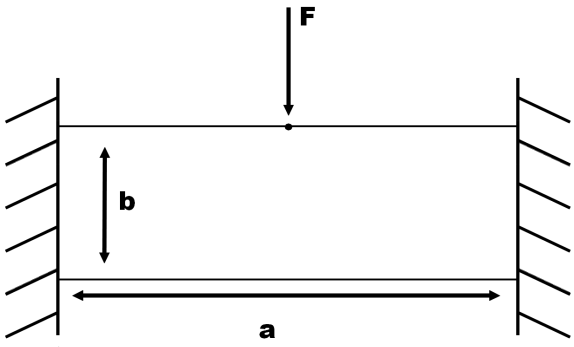


Figure 3.21: Load case of the doubly clamped beam.

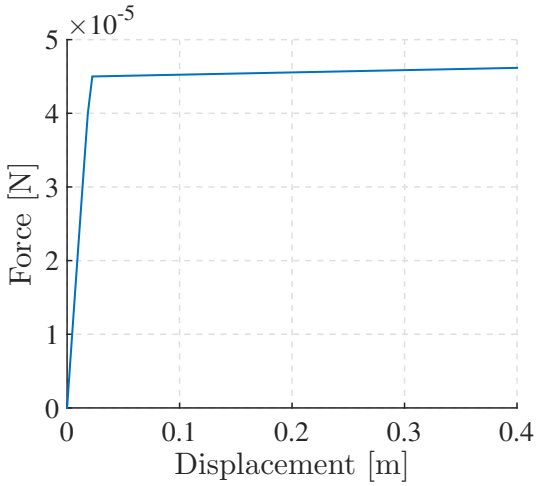
The same material is used (i.e. $E = 1[Pa]$ and $\nu = 0.4$) but the two sides of the beam are completely clamped and the force is applied at the centre of the beam as shown in Fig. 3.21. Arbitrarily, a 3:1 ratio is used for this case and is discretized by a 120×40 elements mesh. The volume fraction used is 0.1 which is really lower than for the cantilever beam. The minimum radius is set to 3 and the two same loads are used (i.e. $3e - 5$[N] and $5e - 5$[N]). To avoid mesh distortion, the interpolation scheme is adopted. Adjoint sensitivities have already been successfully validated for the cantilever beam.

3.8.1 Linear optimization

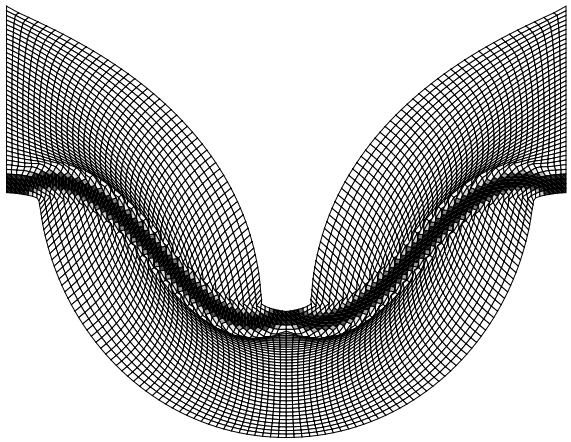
As for the cantilever beam, a standard layout is needed. To this pupose, the linear optimization is performed on the design domain. The related shape is found in Fig. 3.22.

Figure 3.22: Optimal layout for F_1 [N] with linear analysis.

This structure allows a very small displacement for the design force, about 0.0137[m]. When the force increases, it seems to resist but when a nonlinear analysis is done, it shows that this layout is subject to snap-through effect. Here, it happens for F_2, the deformed shape can be seen in Fig. 3.23b and the related displacement blows up. Indeed, for this load, the displacement reaches 1.637[m].



(a) Force-displacement curve.



(b) Snap-through effect at F_2.

Figure 3.23: Snap-through effect under a load of F_2 on the optimized beam for F_1.

3.8.2 Results

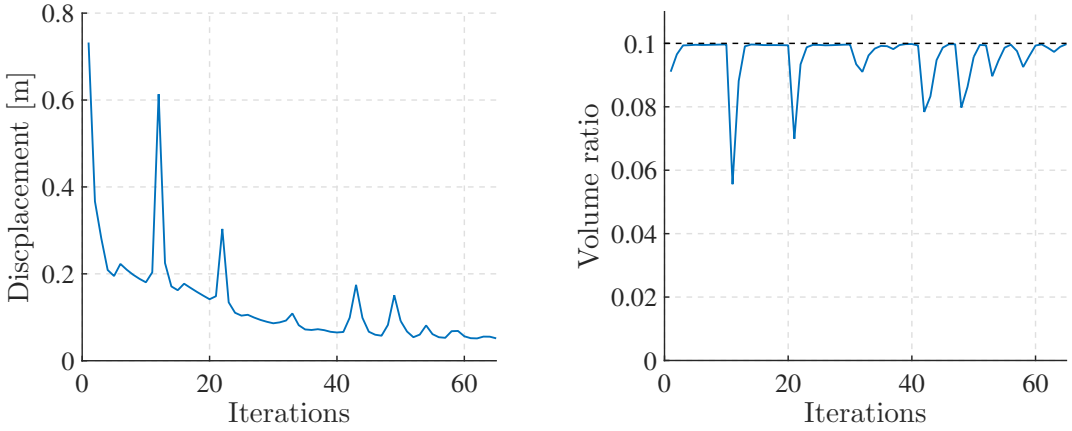
Unlike the cantilever beam, for small loads, the nonlinear layout is the same than the linear one as it is shown in Fig. 3.24. It is probably due to the fact that the force is directed along the axis of symmetry.

Figure 3.24: Optimal layout for F_1 [N] with nonlinear analysis.

Thus, the conclusion is the same than before. Then the second analysis can be done with the load F_2 which is critical for the linear result. As it has been mentioned, the volume is very low, it has for effect to made some sparse matrices which could induce non convergence at the early stages of the optimization for bigger loads. One way to circumvent this problem is to make one iteration with a lower load and then change its value. This could work because the first step tends to stiffen a lot the structure. With this, a new design has been found for bigger loads which avoid snap-through. As an example, for the critical load of the linear design, it presents a displacement about 0.04219[m]. For the previous design load F_1, it leads to a displacement of 0.02584[m] which is higher than for the linear optimization (0.0137[m]).

Figure 3.25: Optimal layout for F_2 with nonlinear analysis.

It can be noticed that this problem is much more sensitive than the cantilever beam. Indeed, if the convergence parameters are analyzed in Fig. 3.26a and Fig. 3.26b, one can see that the convergence is less stable. This is probably due to the fact that the fraction volume is lower. Thus, when the Heaviside projection changes its β value, the volume proportionally changes more than for the cantilever. These changes in the volume have an effect on the stiffness of the structure.



(a) Displacement.

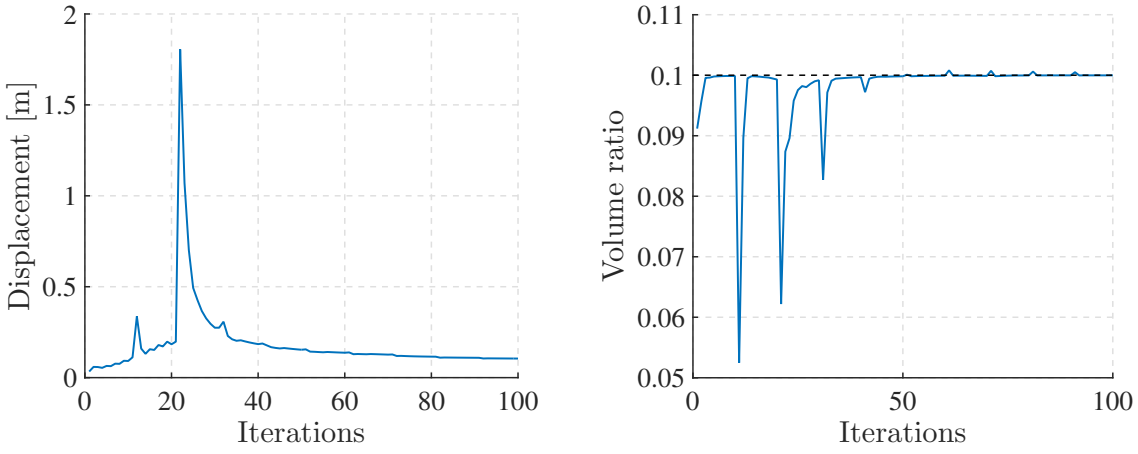
(b) Volume.

Figure 3.26: Optimised displacement and volume as a function of the number of iterations for F_2 .

Surprisingly, even if the layout does not present snap-through for the previous critical load, it undergoes the same effect for a higher load at $7e-5$ [N]. It shows the sensitivity of the optimization and the need to be careful. Thus, some conditions of this benchmark have to be modified. By changing the penalty continuation, going from 1 to 3 by an increment of 0.1, each 2 iterations until the penalty reaches 2 and then, each 5 iterations instead of the previous. It allows to increase the design load, it results in another resulting shape. The layout is shown in Fig. 3.27.

Figure 3.27: Optimal layout for $2 \times F_2$ with nonlinear analysis and more progressive continuation method.

This shape is quite similar to the previous one but it is tighter which provides more stability. One can note that it is closer to the result obtained by different studies (M. Wallin, 2018, [21]; Pedersen et al., 2001, [28]). The convergence parameters are plotted in Fig. 3.28a and Fig. 3.28b. The displacement in Fig. 3.28a shows that after 20 iterations, the structure suffers from buckling. Thus, from this point it optimizes the geometry for this type of behaviour. The volume ratio still changes when the parameter β increases but it seems more stable with this continuation procedure.



(a) Displacement.

(b) Volume.

Figure 3.28: Optimized displacement and volume as a function of the number of iterations for $2 \times F_2$.

Concerning the displacement, a FEM analysis can be done on the new shape, it results that displacements are 0.0367[m] and 0.0576[m] for F_1 and F_2 respectively. These results are not as good as the first nonlinear case for the same loads. The important point is that for a load of $1e-4$[N] it presents a displacement about only 0.1048[m] which is much better. Indeed, if the force/displacement curves are plotted for this shape and the linear one as in Fig. 3.29, it can be seen that it is not subjected to buckling.

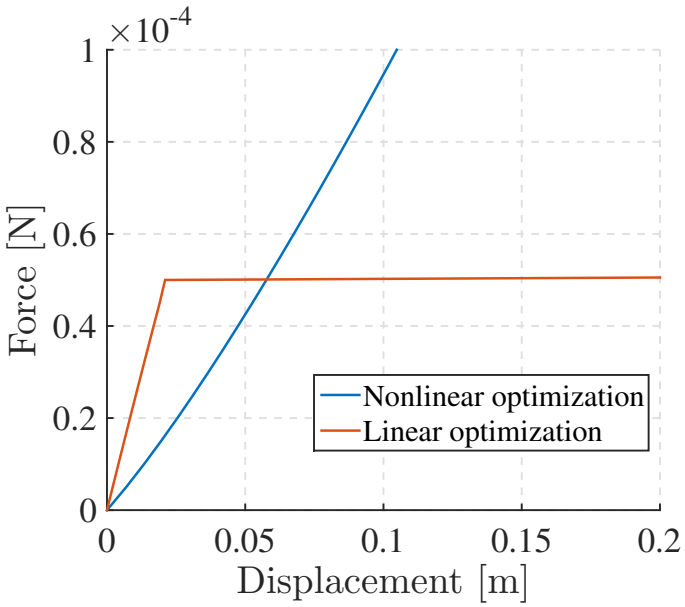
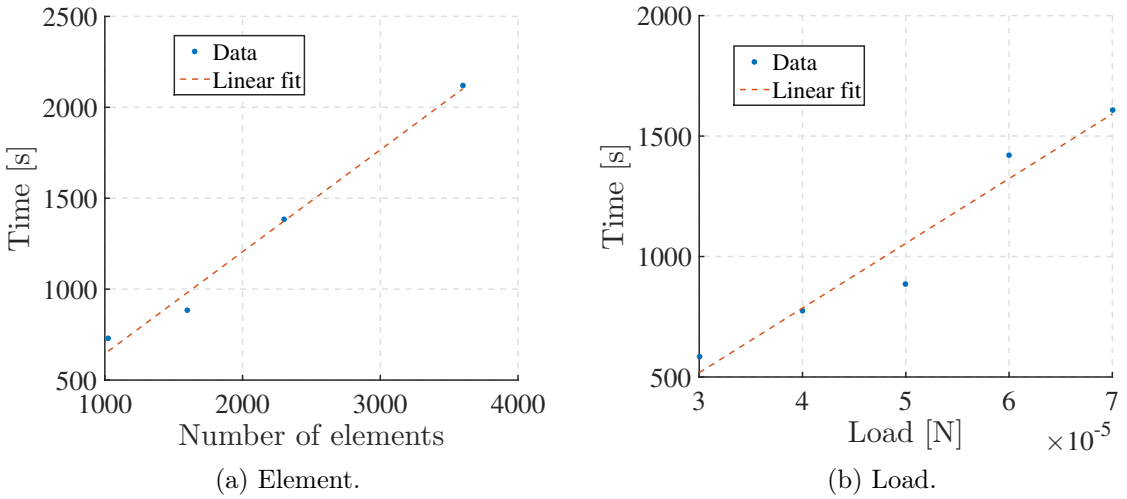


Figure 3.29: Force/Displacement curve for linear and nonlinear optimization.

3.8.3 Performance

It is often mentioned that the several simulations are quite slow. Indeed, the nonlinear analysis takes more time as it has been explained in section 3.4.3. Topology optimization needs a FEM analysis on every optimization step. As a consequence, it should be quite long and it is interesting to see how long it takes for different cases. The benchmark is the cantilever beam with 20×80 mesh (1600 elements) optimized for $5e-5$[N] with interpolation scheme. The optimization is run for 100 iterations. The results are presented in Fig. 3.30a and 3.30b.



(a) Element.

(b) Load.

Figure 3.30: Performances of the code: time as a function of simulation parameters.

In Fig. 3.30a, the number of elements increases. As a result, the time increases as well. The obtained layouts for each optimization have been verified and the general geometry are the same. Thus, one can see that the classical optimization takes almost 15 minutes

to compute on my computer (see Appendix A for details) which is already quite long. As a comparison, a linear optimization in the same conditions takes 12.38 seconds to run. If a most accurate layout is wanted, one can change the mesh to have a 30×120 one. In this configuration, the simulation runs in 35 minutes. This increase in time is quite linear compare to the increase in elements. For the compliance test, it is not very useful but it can provide more freedom for more complex optimization. Fig. 3.30b shows how the computational time changes with the load. Indeed, the FEM analysis takes more time to reach the equilibrium value and thus increases the running time.

For each simulation, the time of each iteration has been saved. It allows to see how it changes and what is the time distribution. Indeed, the time taken by the nonlinear FEM t_{FEM} and by the optimization t_{MMA} have been measured. The time proportion is defined for each part of the process as $p_{FEM} = \frac{t_{FEM}}{t_{FEM}+t_{MMA}}$. The results are presented in Tab. 3.5. From these numbers, one can see that the nonlinear FEM is very time consuming compare to the optimization step. Moreover, the proportion between the two remains quite stable for all the meshes.

Mesh	Element	t_{FEM} [s]	p_{FEM} [%]	t_{MMA} [s]	p_{MMA} [%]
16×64	1024	7.17	98.9	0.0881	1.1
20×80	1600	8.73	98.8	0.0972	1.2
24×96	2304	13.52	98.9	0.1403	1.1
30×120	3600	20.67	98.9	0.2410	1.1

Table 3.5: Time distribution for one iteration with different meshes.

In the same way, the computational times have been estimated for different load values. The results are presented in Tab. 3.30b. The time taken by the FEM analysis is increasing with the load but the optimization time remains almost constant all along. Thus, the proportion of the optimization time decreases with the load.

Load [N]	t_{FEM} [s]	p_{FEM} [%]	t_{MMA} [s]	p_{MMA} [%]
$3e-5$	5.72	98.2	0.1035	1.8
$4e-5$	7.62	98.7	0.1009	1.3
$5e-5$	8.73	98.8	0.0972	1.2
$6e-5$	13.91	99.3	0.0995	0.7
$7e-5$	15.75	99.49	0.0994	0.6

Table 3.6: Time distribution for one iteration with different loads.

3.8.4 Implementation

All the code used in this work are available in the appendices. I started these codes with the "Topology Optimization Codes for Geometrical Nonlinearty" by Yu Li (Technical University of Denmark). Even if the global structure of the procedure was well implemented, this code did not give convenient result. Thus, I needed to be familiar with it in order to find what was necessary to modify. Since the code was already inspired from the famous "Efficient topology optimization in MATLAB using 88 lines of code" (2010, [20]), it was not too hard to see the ins and outs. Then the errors have to be corrected. After that, some other parts were needed to be implemented, a move limit, continuation scheme,

Heaviside projection, interpolation energy. It was also important to make it clearer and add a benchmark case.

3.9 Conclusion

Most of the applications using optimization only focuses on linear case. Doing so, the model is quite simple and the FEM analysis is direct. Since geometrical nonlinearities involve large displacements, the linear assumptions are useless. The new model includes a Newton-Raphson scheme to perform the FEM analysis. To validate the nonlinear optimization, a well-known problem is chosen, the end-compliance minimization. The main issue of nonlinear modeling is the mesh distortion that appears in some low-density elements. To avoid that, several methods exist in the literature. The most convenient has been chosen: the energy interpolation method. This method uses a linear model in the sensitive elements. The layout of the cantilever beam and the doubly clamped one are compared to existing results. These ones allow to validate the model. However, some results on the doubly clamped beam show that the parameters choice is quite important just as the verification. Indeed, a bad choice can lead to a structure that is still limited (e.g. subject of buckling). Obviously, the computational time is bigger for the nonlinear optimization. This is mostly due to the iterative FEM analysis. For example, the nonlinear model takes 15 minutes to solve the cantilever optimization against 12 seconds for the linear one.

Chapter 4

Prescribed curve

Given the nonlinear optimization model has just been built, it is thus possible to investigate the way to impose a prescribed curve. Compare to the compliance model, some parameters could change in order to give more freedom on the optimization. Indeed, this problem is not classical and there is no reference on the shape that could be obtained. More freedom could lead to better results.

4.1 Reference curve

In order to evaluate this, it is important to have a reference on the chosen shape. For this purpose, in one hand, the minimum compliance linear optimization is used and a nonlinear FEM analysis is made. In the other hand, the displacement of the uniformly distributed can be computed both numerically and theoretically in order to see the natural behaviour of the domain.

4.1.1 Cantilever beam

The first domain studied is the cantilever beam. The result of the optimization is plotted in Fig. 3.10. The theoretical expressions for the uniformly distributed domain are expressed in Eq. 3.22 and Eq. 3.23.

For the theoretical value of the mid density beam, the Young modulus is taken as in the SIMP method, i.e. multiplied by the density. As it is shown in Fig. 4.1 the behaviour of the optimal cantilever beam is almost linear while the one of the uniform beam presents a slight curvature.

Then, the characteristic numbers k for these behaviours as $F = kx$ can be found. These are computed for each model in Tab. 4.1, k_{opt} and k_{uni} represent the stiffness of the optimized and the uniform beam respectively. It will be an important parameter for the design of a nonlinear spring.

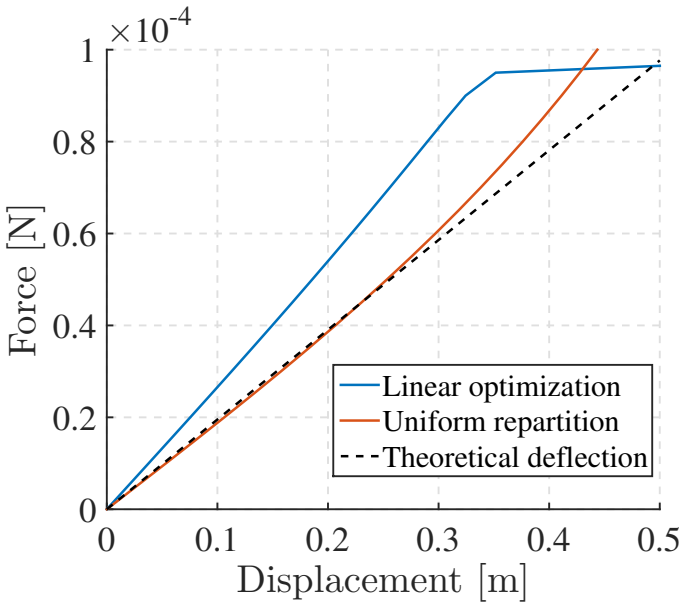


Figure 4.1: Force/Displacement curve of the references for the cantilever beam.

Force [N]	k_{opt} [N/m]	k_{uni} [N/m]	k_{th} [N/m]
1e-5	2.644e-4	1.869e-4	1.953e-4
2e-5	2.650e-4	1.882e-4	1.953e-4
3e-5	2.660e-4	1.905e-4	1.953e-4
4e-5	2.674e-4	1.936e-4	1.953e-4
5e-5	2.692e-4	1.974e-4	1.953e-4
6e-5	2.713e-4	2.019e-4	1.953e-4
7e-5	2.736e-4	2.071e-4	1.953e-4

Table 4.1: Characteristic number table.

According to these results, it seems that the cantilever beam has a slight stiffening behaviour for both the linear optimized and the uniformly divided beams. For the reference, the mean number is used that means $\bar{k}_{opt} = 2.697e-4[\mathrm{N/m}]$ and $\bar{k}_{uni} = 1.987e-4[\mathrm{N/m}]$.

4.1.2 Doubly clamped beam

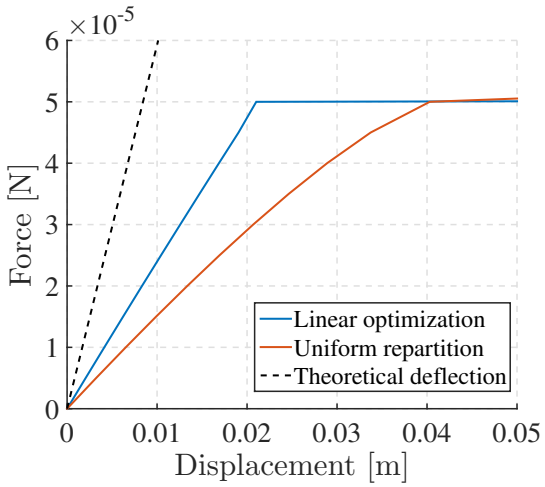
The same procedure can be done for the doubly clamped beam. For this configuration, the theoretical formula writes (C. Livemore, 2007, [22]):

$$f = \frac{PL^3}{6 \times 32EI} \tag{4.1}$$

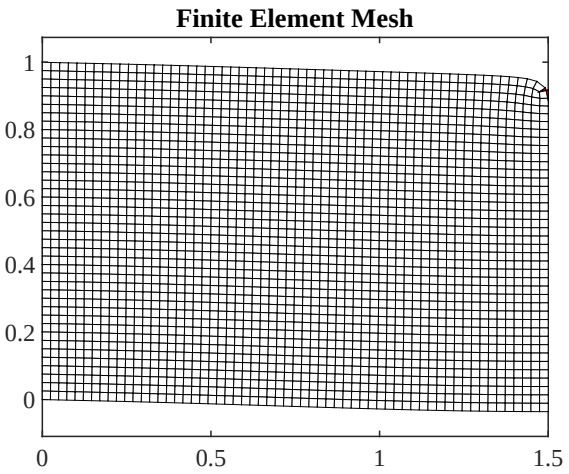
with the same variables as for the cantilever beam. The section is also rectangular so its inertia presents the same expression as Eq. 3.23.

The different curves are plotted in Fig. 4.2a. In this case, the analytic solution does not fit very well the numerical result. This is probably due to the fact that there is a lot of compression on some elements. Indeed, as it has been mentioned, the used model does not perform well in compression. This can be seen in Fig. 4.2b, the element in red is

completely crushed and the elements around also. One can also notice that the nonlinear design (section 3.8) was much more stable and could made a better reference from this point of view. As shown in Fig. 3.26a, the behaviour is almost linear but one can see a slight stiffening tendency. If a characteristic number should be taken from that, it comes $k = 8.672e - 04[N/m]$.



(a) Force/displacement.



(b) Half-mesh.

Figure 4.2: Force/displacement curve and half-mesh related of the doubly clamped beam.

4.2 Problem statement

The first step of a topology optimization process is to choose a cost function. The idea is to force the force-displacement behaviour to have a precise shape. First, one will try only some linear and simple polynomial functions (e.g. kx^n) but the main purpose is to impose any behaviour that could have been characterized. To do this, some target points that come from the imposed law are chosen and the error between these points and the one of the beam has to be minimized. It makes more sense to use the relative errors in order to be dimensionless. Mathematically, the problem can be stated as:

$$\begin{aligned} \min \quad & f_0(x) \\ \text{s.t.} \quad & \mathbf{r} = \mathbf{0} \\ & \mathbf{v}^T\mathbf{x} \leq v^* \\ & \mathbf{x}_{min} \leq \mathbf{x} \leq \mathbf{x}_{max} \end{aligned} \tag{4.2}$$

where f_0 is the cost function to be determined. The other variables are the same as before.

It would be interesting to try different forms of errors, the most used is the euclidean norm:

$$f_0(x) = \sum_{i=1}^{p} w_i \frac{(u_i - u_i^*)^2}{u_i^{*2}} \tag{4.3}$$

where u_i^* the prescribed displacement, u_i the computed displacement of the beam and w_i the weight of each point.

It is possible to use the maximum norm (or L_∞ norm) with the same parameters:

$$f_0(x) = \max \left| \frac{u_i - u_i^*}{u_i^*} \right| \tag{4.4}$$

Lastly, the L_1 norm could be used as well:

$$f_0(x) = \sum_{i=1}^{p} w_i \left| \frac{u_i - u_i^*}{u_i} \right| \tag{4.5}$$

Like for the compliance, this problem is solved using `MMA` solver proposed by Svanberg (1998, [23]). This means that the sensitivities of the objective function and the constraints are required. The sensitivity of the constraint does not change from the compliance problem since these are the same. The sensitivity of the cost function changes in the standard use of `MMA`:

$$\frac{\partial f_0}{\partial x} = \sum_{i=1}^{p} 2w_i(u_i - u_i^*)\frac{\partial u_i}{\partial x} \tag{4.6}$$

This is the same method as used by F. Wang, O. Sigmund, JS. Jensen (2014, [33]). Eq. 4.2 follows the same kind of procedure as for the compliance and the parameters a_j, d_j, c_j are identical but in this paper, Svanberg (1998, [23]) also proposed an alternative formulation for the least squared problems solved by `MMA`:

$$\min \quad \sum_{i=1}^{p} (h_i(x))^2 \tag{4.7}$$

If the objective can be rewritten as in Eq. 4.7, which is the case here, `MMA` can take a new set of parameters to solve it:

$$\begin{array}{rlrl}
m = 2p + q & & a_j = 0 & j = 1...m \\
f_0(x) = 0 & & d_k = 2 & k = 1...2p \\
f_k(x) = h_k(x) & k = 1...p & d_{2p+j} = 0 & j = 1...q \\
f_{p+k}(x) = -h_k(x) & k = 1...p & c_k = 0 & k = 1...2p \\
f_{2p+j}(x) = g_j(x) & j = 1...q & c_{2p+j} = \text{big} & j = 1...q
\end{array}$$

The problem to solve can be easily expressed in this form:

$$h_i(x) = w_i \frac{(u_i - u_i^*)}{u_i^*} \tag{4.8}$$

This formulation uses h_i as a double constraint instead of a cost function and its sensitivity will be needed. With some slight changes in these parameters, `MMA` can be optimized for the p-norm or the maximum one.

4.2.1 Sensitivity

For both methods, the sensitivities are based on the sensitivity of the displacement. It means that it only needs to perform the adjoint method on $u_i = \mathbf{l}^T \mathbf{u}_i$:

$$\frac{\partial u_i}{\partial x} = \mathbf{l}^T \frac{\partial \mathbf{u}}{\partial x} \tag{4.9}$$

The procedure for finding the adjoint sensitivity is no other than for the compliance presented from Eq. 3.34 to Eq. 3.38. Doing so, it comes:

$$\frac{\partial u_i}{\partial x} = \lambda^T \frac{\partial \mathbf{r}}{\partial x} \tag{4.10}$$

with

$$\lambda = \mathbf{K}_t^{-1} \mathbf{l} \tag{4.11}$$

$$\frac{\partial \mathbf{r}}{\partial x_e} = -\frac{\partial \mathbf{f}_{int}}{\partial x_e} \tag{4.12}$$

Then, for the cost function sensitivity of the classical use as in Eq. 4.2 the sum has to be performed.

4.2.2 Sensitivity validation

Likewise the compliance problem, the adjoint sensitivity should be validated by the finite difference as in Eq. 3.40. The same step size is used since it provides good results. Because in each use of `MMA`, the sensitivity of the cost function (or replacement constraint function) starts from the displacement sensitivity, the validation is made for the classical use. For testing the method, two points from the linear optimization of the cantilever beam are imposed. Four elements of the mesh are arbitrarily taken and their sensitivities are computed at the first iteration. The results are shown in Tab. 4.2 below.

Element	Finite difference sensitivity	Adjoint sensitivity	Relative error [%]
100	-0.30716	-0.30718	4.39e-3
401	-0.19752	-0.19751	5.26e-3
850	-0.004802	-0.004803	2.64e-3
1500	-7.6052e-4	-7.6036e-4	2.16e-2

Table 4.2: Comparison between finite difference and adjoint method sensitivies.

As for the compliance optimization, the adjoint method has been well implemented and leads to accurate results. In this case, the error does not exceed $3e-2\%$. This result can be extended to each type error and each use of `MMA`.

4.3 Method assessment

First, to assess the method, one can try to impose some easy points. The simplest is to try to get the behaviour of the stiffest beam. It means just trying to impose some values with the stiffness number extract from Tab. 4.1, i.e. $\bar{k}_{opt} = k_{top} = 2.697e-4[\mathrm{N/m}]$.

The cantilever beam case is chosen, two imposed points are taken, since the beam should always pass by the point $(0, 0)$, two imposed points are enough to see a behaviour. The euclidean norm is used for these simulations but this one will be compared to the other norms in the next section. This will allow to compare the classical and the special least square formulation of `MMA` and see if one is better than the other. All the parameters are identical, the weight factors are set to 1, the continuation method has the same procedure as for the compliance. The projection continuation on β is also the same. Arbitrarily, the two displacements chosen are 0.05[m] and 0.1[m] and their related force $1.35e-5$[N] and $2.69e-5$[N].

4.3.1 Classical `MMA`

The classical use of `MMA` is the first tried. After 100 iterations, the relative error is equal to 2.5% and the obtained layout is shown in Fig. 4.3.

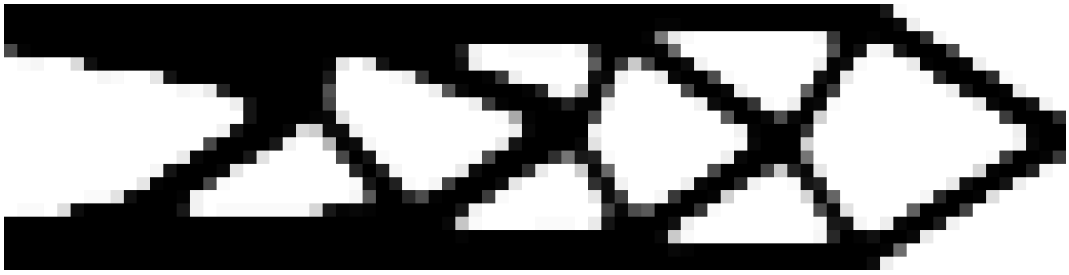


Figure 4.3: Layout obtained with classic use of `MMA` for the optimal stiffness.

This shape is similar to the one resulting from the linear optimization (Fig. 3.10). It just presents a little asymmetry compare to this one. As a convergence indicator, the relative error can be plotted along the iterations as in Fig. 4.4.

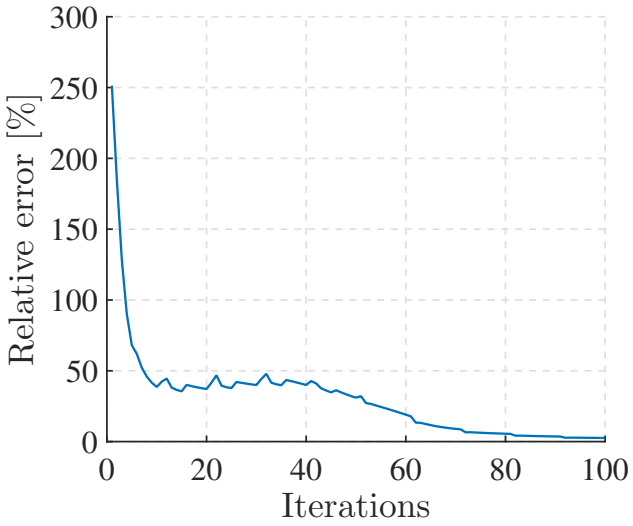


Figure 4.4: Relative error as a function of the number of iterations.

It shows that the formulation is stable in this case, the error decreases without too many jumps. The small jumps still represent the change of P or β.

4.3.2 Least square formulation of `MMA`

The same optimization is performed with the alternative formulation. The relative error resulting is equal to 2.56% and the layout is presented in Fig. 4.5.

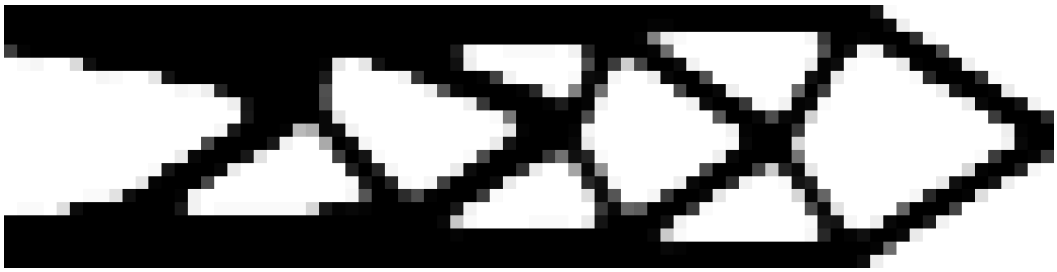


Figure 4.5: Layout obtained with alternative use of `MMA` for the optimal stiffness.

The evolution of the error along iterations is shown in Fig. 4.6. It proves the convergence of the alternative formulation.

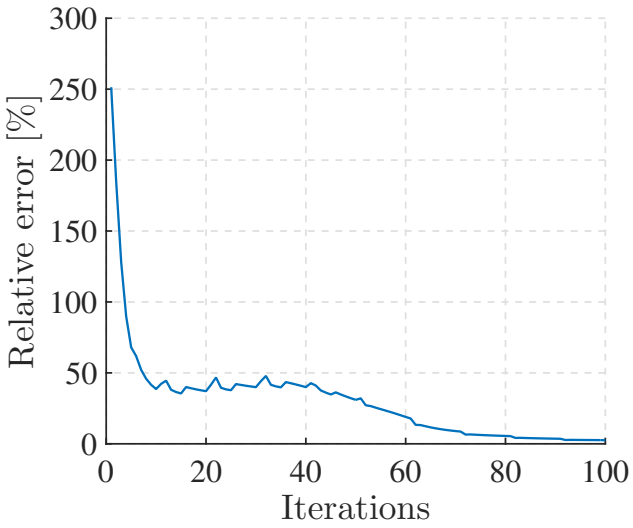


Figure 4.6: Relative error as a function of the number of iterations.

After the two simulations, one can see that both formulations give close results in the shapes. The resulting errors are also very close as shown in Fig. 4.4 and 4.6. As it can be seen in Fig. 4.7, the new force/displacement curves pass by the imposed points and keep this behaviour until $7e-5$[N]. Knowing that, the choice can not be made between these two formulations. Thus, other simulations are needed to have a better differentiation.

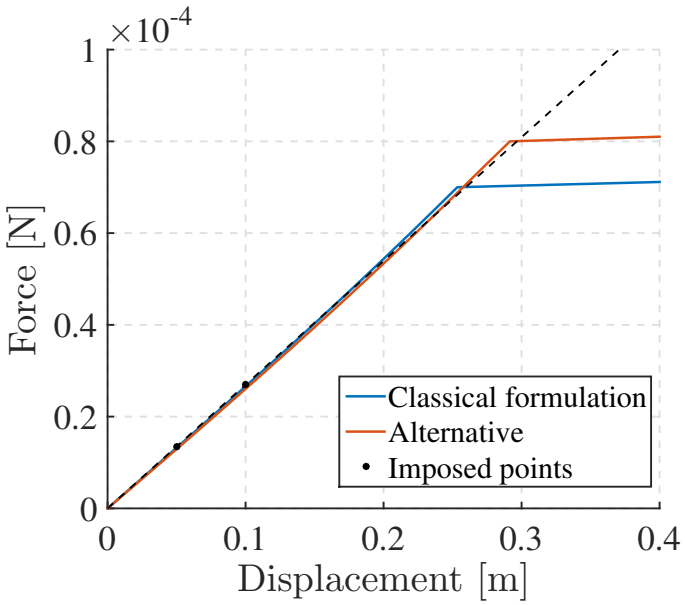


Figure 4.7: Force/Displacement curves for the optimal beam problem. $F = k_{top}x$ in black dotted line.

4.4 Other linear behaviour

For the next step, one can try to impose a behaviour a little more complex. Indeed, due to the fact that the points were reachable for only one shape, the precedent example was quite easy. Therefore, one will try to impose a linear behaviour by two points where the stiffness coefficient is smaller than the optimal one $k = 0.8 \times k_{top}$. The two points imposed are arbitrarily chosen and presented in Tab. 4.3. In all the simulations, the forces have been chosen in such a way that the related displacements are equal to 0.05[m] or 0.1[m] arbitrarily.

Point	Force [N]	Displacement [m]
A	$1.08e-5$	0.05
B	$2.16e-5$	0.1

Table 4.3: Imposed points.

The two formulations are tried to complete the comparison. Due to the more complex problem, the solver had some difficulties to reach a well defined shape. Indeed, the classical formulation does not converge at all and the shape is too much undefined. The weight of the points are increased in order to be more restrictive. Even with this restriction, the classical formulation shows its limit, the result of this optimization is found in Fig. 4.8. It is obvious that this layout is not good, the shape is not defined, it remains a lot of intermediate elements.

The convergence parameters also show the unstable character of this optimization. Indeed, one can see huge peaks either in the relative error and the volume ratio in Fig. 4.9a and Fig. 4.9b. The simulation has been stopped at 52 iterations because it took too much time to solve only one iteration.

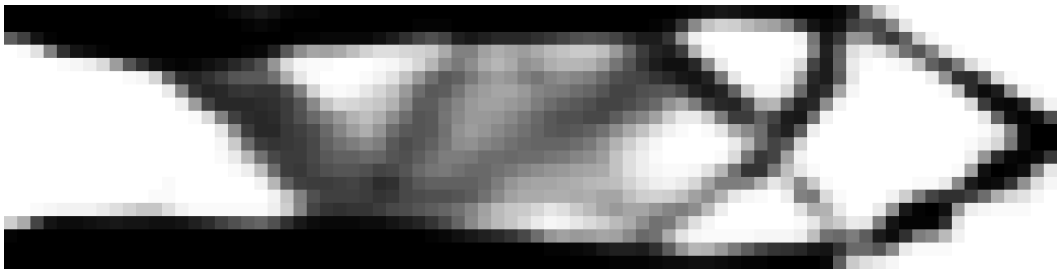
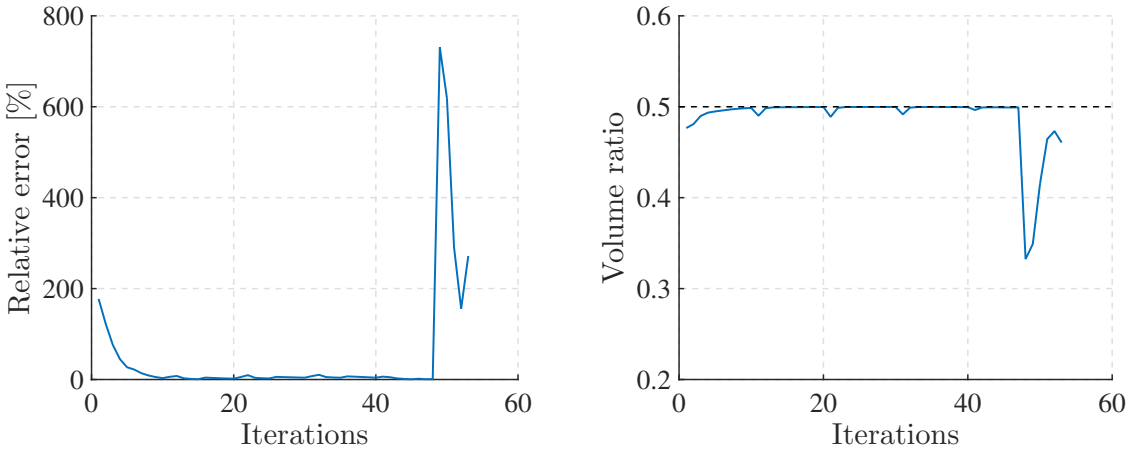


Figure 4.8: Unconverged layout obtained with classical use of `MMA`.



(a) Relative error.

(b) Volume.

Figure 4.9: Optimized error and volume as a function of the number of iterations for $0.8 \times k_{top}$.

The same optimization is made using the alternative formulation. In this formulation, the solutions tend to more interesting layout. The issue of the non convergence of the layout is also noted but only on small areas. Thus to avoid it, the Heaviside projection is increased to force the void/solid distribution. Instead of an incremental increasing of the β parameter, an exponential increasing is chosen with a factor of 1.5, i.e. $\beta = 1.5 \times \beta$. However, even with this improvement, the layout keeps some intermediate zones as seen in Fig. 4.10.

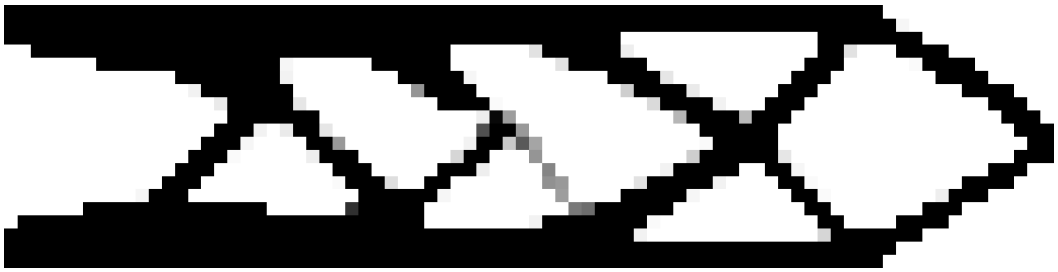
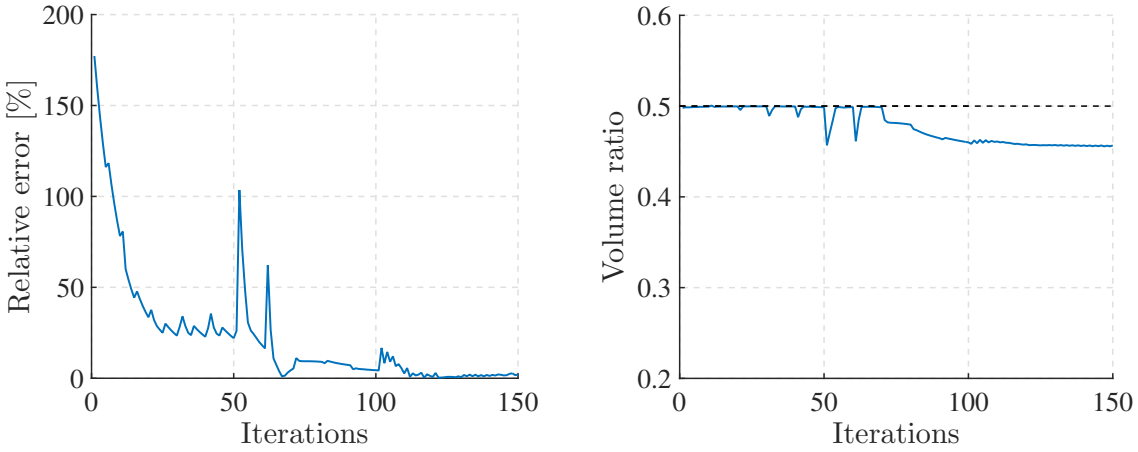


Figure 4.10: Converged layout obtained with alternative use of `MMA`.

The convergence parameters are more stable. The error in Fig. 4.11a shows an overall decrease along the iterations even if a peak happens around 50 iterations. The volume ratio in Fig. 4.11b remains stable and under the 0.5 limit plotted in dotted line. However, it is probably on this parameter that the difference between the two formulations is. Indeed, in order to get a less stiff structure, the most simple solution is to remove a little bit of matter. In both cases, the solver tries to do it but in the classical it does not stabilize, due to the fact that it is the only constraint. In the second formulation, one can see that the final volume ratio is equal to 0.4566, the solver is able to stabilize this ratio because of the presence of the objective function constraints. After this, it seems logical to prefer the alternative formulation instead of the classical one.



(a) Relative error.

(b) Volume.

Figure 4.11: Optimized error and volume as a function of the number of iterations for $0.8 \times k_{top}$.

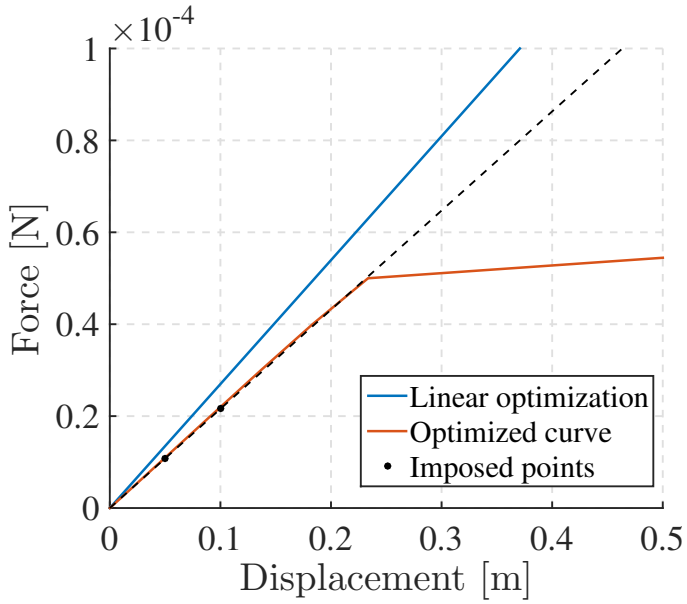


Figure 4.12: Force/Displacement curve for the sub-optimal beam problem. $F = 0.8 \times k_{top} x$ in black dotted line.

The resulting force/displacement curve can be verified. This is done in Fig. 4.12, one can see that the resulting curve passes through the imposed points. However, after a force of $5e-5$[N], the structure breaks. To avoid this issue, a wider range of imposed points can be tested or considered a predefined interval use of this structure.

As mentioned, some zones keep an intermediate character. In order to change that, one can try to vary some of the optimization parameters. Nevertheless, some parameters are sometimes very sensitive with the new Heaviside projection. As an example, the radius filter, if it goes from 2 to 3, the layout will never converge. Thus, now that the formulation is chosen, one can investigate its behaviour for this optimization. The relative error can be compared to the classical one. Then, one can study the other errors presented in Eq. 4.4 and 4.5. At first, the mesh is changed (which increases the running time) to get a more accurate result. A technique when the results are undefined could be a post processing step. As an example, a threshold value can be chosen and a cut off can be performed. Then, the new structure could be studied to see how its behaviour changes.

4.4.1 Mesh dependency

First, the size of the mesh is changed. It goes from 20×80 to 30×120. Thus the radius filter is set to 3 now. It will obviously increase the computational time as it has been shown in Tab. 3.5. However, for such optimization where the layout is not known in advance, it offers more freedom. For the same points as before, it leads to the layout in Fig. 4.13. This shape is a little bit different than the previous one and allows an improvement in the error going to only $4.6e-3\%$. The volume is quite similar and tends to a ratio about 0.479.

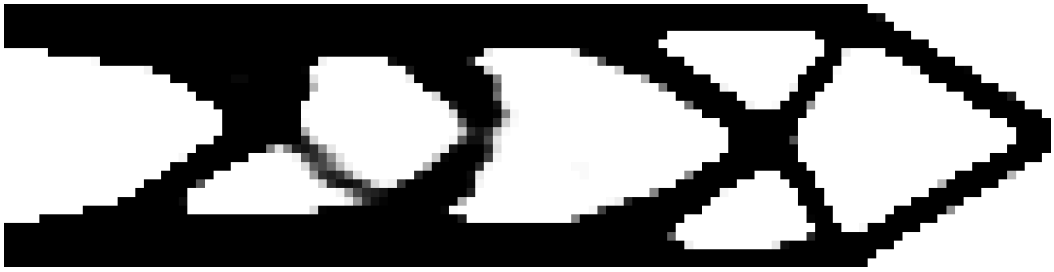


Figure 4.13: Converged layout obtained with alternative use of `MMA`.

In addition of the error improvement, the breakdown of the structure occurs later. It can be seen in Fig. 4.14, the behaviour remains quite linear until $4e-5$[N] but it breaks at $6e-5$[N]. The values of some points are taken in Tab. 4.4 for illustration. The displacements wanted are expressed as u_{Th} and those of the computed beam u_{opt}. One can see that until $5e-5$[N], the error remains under 5% which seems still quite accurate.

Force [N]	u_{opt}	u_{Th}	Relative error [%]
1e-5	0.04635	0.04634	0.0245
2e-5	0.09265	0.09269	0.0395
3e-5	0.13967	0.13904	0.4514
4e-5	0.18834	0.18539	1.5920
5e-5	0.24013	0.23173	3.6232
6e-5	0.30053	0.27808	8.0709
7e-5	0.89012	0.32443	174.362

Table 4.4: Comparison between prescribed behaviour and obtained one.

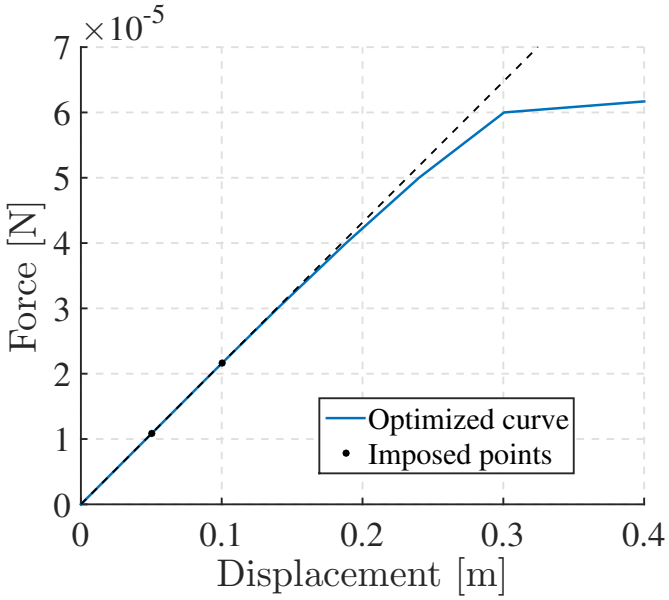


Figure 4.14: Force/Displacement curve for the sub-optimal beam problem with new mesh. $F = 0.8 \times k_{top}x$ in black dotted line.

4.4.2 Radius filter

After several trials, it seems that changing the radius filter can have an important impact. Unlike the first mesh, with the new one, if the radius is increased, it converges and the resulting layout is different. This one is drawn in Fig. 4.15.

Figure 4.15: Converged layout with $r_{min} = 4$.

The error is equal to $3.3e-3\%$ which is the same order as with a smaller filter. However, after a FEM analysis, the structure is more stable and does not brake until $1e-4$[N]. It is shown in its force displacement curve in Fig. 4.16. If the displacements are compared to the one of the prescribed curve, it comes that the mean error is about 1.73% while the maximum one is equal to 4.5%. This model seems robust and it will be interesting to deepen the study with. However, it shows how much the result can be sensitive to a parameter.

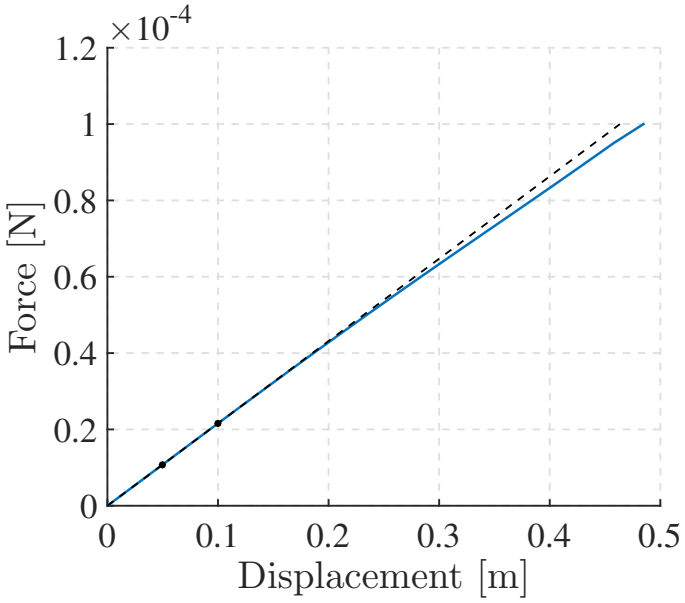


Figure 4.16: Force/Displacement curve for the sub-optimal beam problem with $r_{min} = 4$. $F = 0.8 \times k_{top}x$ in black dotted line.

4.5 Error form

As mentioned, the error can take several forms. The first one chosen was the euclidean norm since it is a widely used and because a special formulation is proposed. However, `MMA` is also optimized for other errors. It is interesting to see if these errors have an impact on the converged layout. Also, the comparison between the error and the relative error has been made and leads to the same result thus, the relative errors are chosen here.

4.5.1 L_1 norm

For this error, presented in Eq. 4.5, the parameters of `MMA` change compare to the euclidean norm. The parameters become:

$$\begin{aligned} a_j &= 0 & j &= 1...m \\ d_j &= 0 & j &= 1...m \\ c_k &= 1 & k &= 1...2p \\ c_j &= \text{big} & j &= 1...q \end{aligned}$$

Using this norm, the shape changes a little as it is shown below (Fig. 4.17). The relative error is equal to $1.3e-5\%$ which represents an improvement. The relative error

can be computed all along the force/displacement curve plotted in Fig. 4.18. Then, the average of these values can be computed. It comes that the mean relative error is about 1.25% and no error exceeds 2.6%.

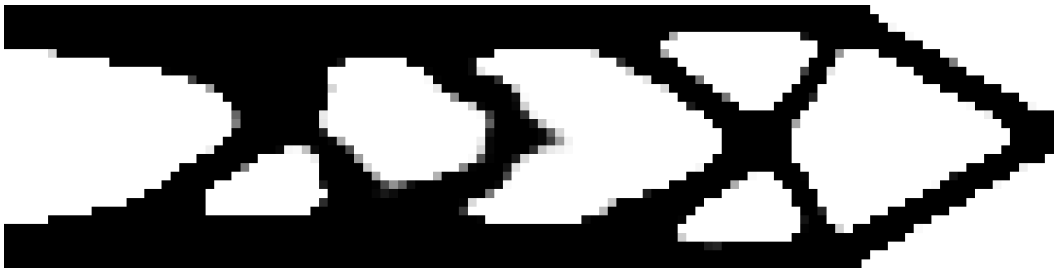


Figure 4.17: Converged layout with $L1$ norm.

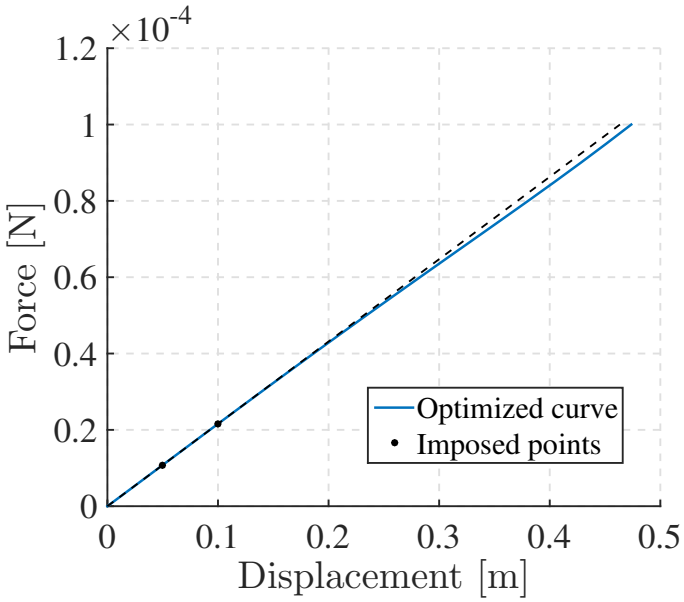


Figure 4.18: Force/Displacement curve for the sub-optimal beam problem with L_1 norm. $F = 0.8 \times k_{top}x$ in black dotted line.

4.5.2 L_∞ norm

For this error, presented in Eq. 4.4, the parameters of `MMA` change again compare to the others. These parameters are:

$$\begin{aligned} a_k &= 1 & j &= 1...2p \\ a_{2p+j} &= 0 & j &= 1...q \\ d_j &= 0 & j &= 1...m \\ c_j &= \text{big} & j &= 1...m \end{aligned}$$

The identical process is used for this error type. It leads to an error about $1.7e-5\%$. From the FEM analysis, the force/displacement curve is found and this one is represented in Fig. 4.20. Using those points, the mean error comparing to the prescribed behaviour

can be computed. It comes that the mean error is about 1.6% and the maximum one is equal to 3.4%.

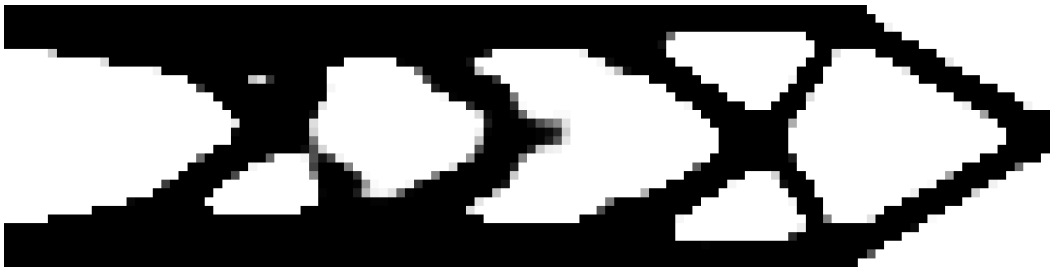


Figure 4.19: Converged layout with L_∞ norm.

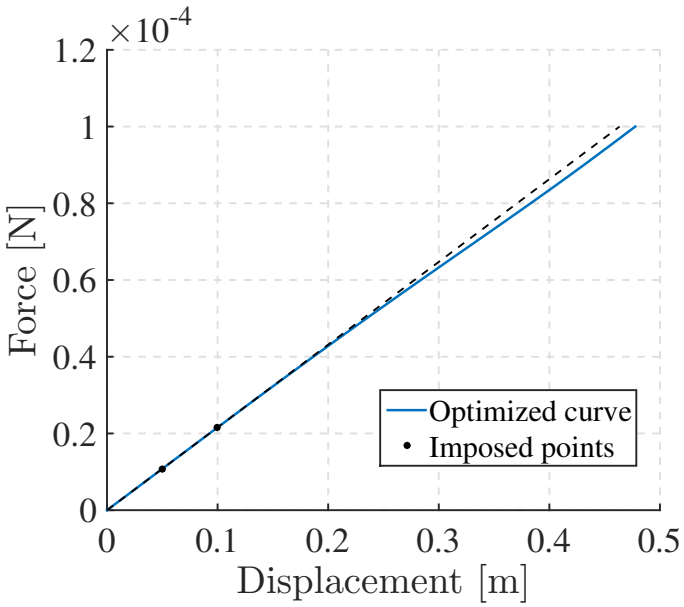


Figure 4.20: Force/Displacement curve for the sub-optimal beam problem with L_∞ norm. $F = 0.8 \times k_{top}x$ in black dotted line.

The exact values of the point A and B for each type of error have been saved and are shown in Tab. 4.5. It allows to end the comparison between all the types. For each error, the resulting error are very small. However, according to these values, the euclidean norm seems to be the less accurate. The square norm decreases a lot with such a small differences. Then, the two other norms are very close on these two points. Thus, from the other points computed by a FEM analysis, the mean error and the max one were smaller for the L_1 norm. It leads to use this error type for the other simulations. From a geometric point of view, the euclidean layout seems to be the smoothest but they are all very similar. The main difference is about the wavy bar in the middle of the structure that presents a more pronounced peak for the L_1 and the L_∞.

Error type	u_a	Relative error [%]	u_b	Relative error [%]
Euclidean	0.050000085604	$1.7e-4$	0.1000002822	$2.8e-4$
L_1	0.049999999618	$7.6e-7$	0.0999999988	$1.2e-6$
L_∞	0.049999999166	$1.6e-6$	0.0999999994	$6e-7$

Table 4.5: Comparison between different error types.

4.6 Extreme linear behaviour

After proving that the model can handle a slight decrease of the stiffness. It is possible to try to reach the limit of the code by decreasing more the stiffness. It has been noticed that this type of simulation takes more time to converge. As a solution, the Heaviside continuation can be sped up, going from 1.5 to 2. As an example, $0.3 \times k_{top}$ is chosen, the two imposed points are presented in Tab. 4.6. The forces used for this optimization are very small because these are chosen to give feasible displacements. The same mesh is used and the filter radius is equal to 3.

Point	Force [N]	Displacement [m]
A	$4.04e-6$	0.05
B	$8.09e-6$	0.1

Table 4.6: Imposed points for $0.3 \times k_{top}$.

The optimization leads to an error about $1.16e-6\%$ and the layout is presented in Fig. 4.21. The volume ratio stabilizes near 0.48. The layout seems very weak but it is the idea of such a design. Indeed, it is an envelop only, there is any bar that can stiffen the structure inside.

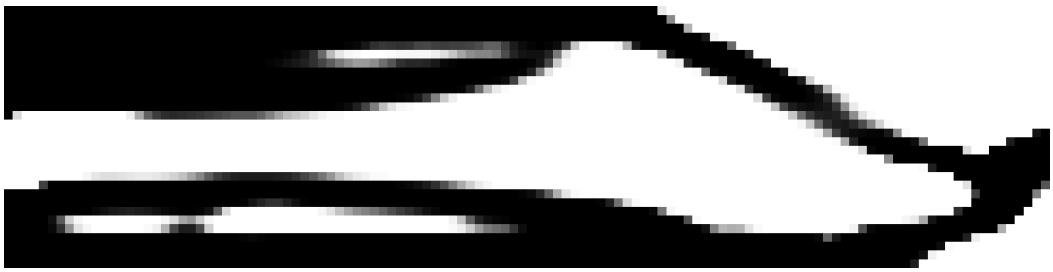


Figure 4.21: Converged layout for $0.3 \times k_{top}$.

One can see that the force/displacement curve remains in the right behaviour until 0.18[m]. After that, the structure breaks and the displacement increases fast.

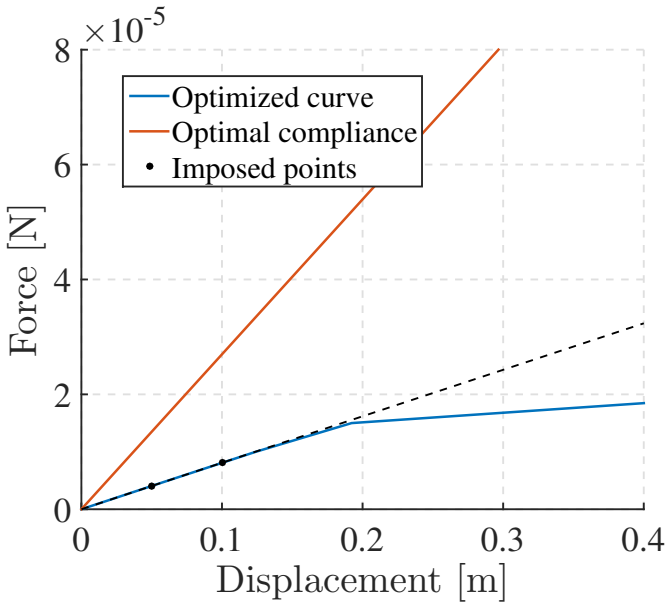


Figure 4.22: Force/Displacement curve for the linear limit problem. $F = 0.3 \times k_{top}x$ in black dotted line.

It is the limit of this optimization, when one try to get a structure even weaker, the convergence is hard to reach and the structure has no sense, presenting some separated parts which is not the purpose of the optimization. As an example, the $0.2 \times k_{top}$ case is shown in Fig. 4.23. A solution to reach this behaviour could be to iterate the objective function. It means starting from an almost converged layout about 30% and then try the 20%.

Figure 4.23: Converged layout for $0.2 \times k_{top}$.

4.7 Nonlinear behaviour

Since the linear cases performed well under a certain limit, some nonlinear functions can be imposed. The first nonlinear behaviour are the polynomials function as $F = kx^n$. The choice of the points is important but there is no real reference for this kind of behaviour. The only way to find the best point is by trials and errors method. As for the extreme linear behaviour, the filter radius is set to 3 and the continuation methods are the same.

Since it is an often cited example (J.P. Noël and G. Kerschen, 2018 [3]), the cubic behaviour is chosen. The problem when the exponent increases is that the slope at the first points is tiny. The bigger the exponent, the smaller the slope. The induced problem

is that the tangent matrix tends to be null which can lead to a convergence problem. To avoid this problem, two ideas have been tried. First, the k factor can be increased. Then, one can add a linear part to the behaviour as $F = k_L x + k_{NL} x^n$. The combination of the two can be interesting as well. From the previous section, the optimization reaches a limit around $0.3 \times k_{top}$. Thus, this limit is used as the linear part.

4.7.1 Pure nonlinear behaviour

First, a pure nonlinear behaviour can be imposed. By pure, it means that there will not be any linear part, i.e. $F = kx^n$. With the formulation, the cubic exponent never converges when the parameters change. However, to try this formulation, the exponent has been decreased to a square case. The imposed points are $(0.05, 0.337e-5)$ and $(0.1, 1.349e-5)$. Doing so, with the factor k equal to $5 \times k_{top}$, the solver converges. It leads to the layout plotted in Fig. 4.24 and the error is about $5.4e-7\%$.

Figure 4.24: Converged layout for $5 \times k_{top} x^2$.

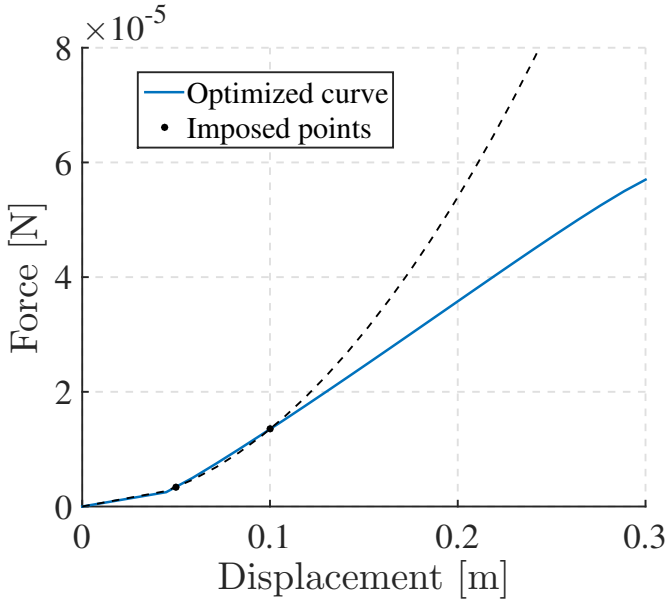


Figure 4.25: Force/Displacement curve for the pure nonlinear problem. $F = 5 \times k_{top} x^2$ in black dotted line.

When the FEM analysis is done, the force displacement curve can be computed, it is shown in Fig. 4.25. One can see that this resulting curve passes through the points by

two lines. The first point acts like an flexion point. Obviously, after the second point, the slope does not change and the curve moves away from its reference behaviour (in black dotted line).

This optimization is possible from a factor equal to 3, i.e. $k = 3 \times k_{top}$. Under, the layout can converge but lead to bad results. If there is a lower limit, there is also an upper one. Indeed, if the factor goes too high, one imposed point could present a better stiffness than the optimal stiffness behaviour. In this condition, the point would be unreachable and the solver will not converge. For the chosen points and behaviour, this limit arrives at a factor 10.

4.7.2 Mixed behaviour

The pure nonlinear optimization has shown interesting results but it cannot optimize the layout for a cubic form. Thus, another formulation can be tested to get a more general program. A way to do that is to add a linear part on the expression, i.e. $F = k_L x + k_{NL} x^n$. Since the purpose is to have a nonlinear behaviour, the linear proportion has to be minimized. The first point is to use the less linear part as possible. From the previous section, one knows that the code reaches its limit for $0.3 \times k_{top}$ which seems a good value. The other point is to increase the nonlinear part with k_{NL}. Indeed, if $k_{NL} = k_{top}$, the nonlinear part will represent maximum 3.5% of the behaviour. From the previous result, it seems that $5 \times k_{top}$ is good. The optimized layout is plotted in Fig. 4.26. In this configuration, the nonlinear part is about 14.5% on the last imposed point. As mentioned, this type of optimization is not easy and it can be seen that some intermediate elements are still remaining. In addition of that, some almost solid elements are in void area which is a numerical error. In order to get a more machinable design, a simple post processing is applied, i.e. a cut off. It means that the element under a threshold density are decreased to 0. For this case, the threshold density is set to 0.9. It has not been used before because the optimization made until here were well defined. The new design is represented in Fig. 4.27.

Figure 4.26: Converged layout for $0.3 \times k_{top} x + 5 \times k_{top} x^3$ without cut off.

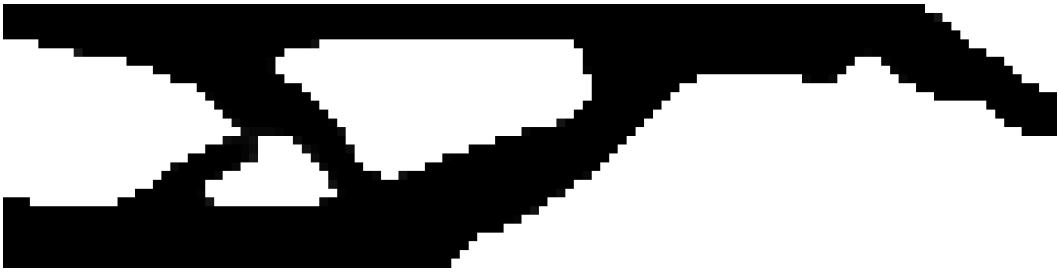


Figure 4.27: Converged layout for $0.3 \times k_{top}x + 5 \times k_{top}x^3$ with cut off.

The final error on the two points is about $5e-8\%$, once again, it is very small. A FEM analysis is performed on the resulting layout. Its force/displacement curve is illustrated in Fig. 4.28. The general shape of this curve is interesting, it shows a real stiffening of the structure. However, the prescribed curve separates from the prescribed behaviour (in black dotted line) and at a force of $1e-4$[N]. It presents an error about 20%. It seems big but it is far from the last prescribed point thus it is reasonable.

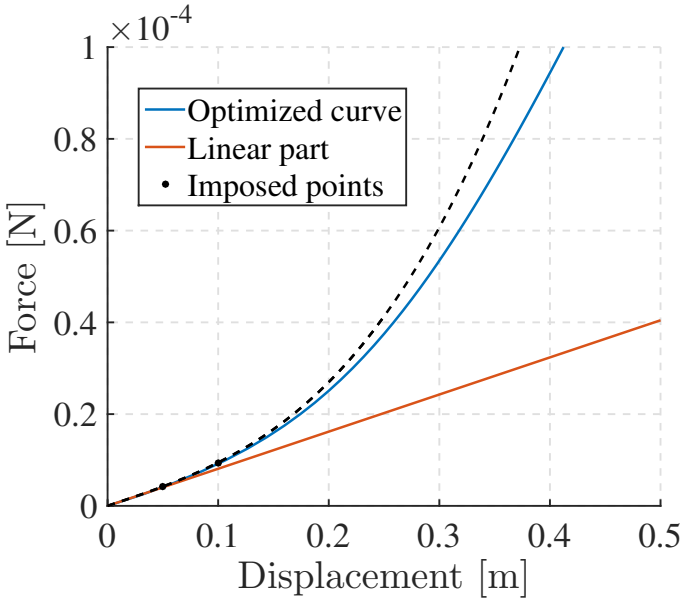


Figure 4.28: Force/Displacement curve for the mixed behaviour problem. $F = 0.3 \times k_{top}x + 5 \times k_{top}x^3$ in black dotted line.

After other simulations, it shows that these conditions are especially good. Indeed, for the other simulations, the results are not that good. It leads to the conclusion that the code gives interesting results in general. However, a good combination of parameters allow the result to be even better. However, this set of parameters is not direct and need some trials to be found.

As an example, for the factor $k_{NL} = 10 \times k_{top}$, the layout and the force/displacement curve related are represented in Fig. 4.29 and 4.30. These figures are more general and reflect the majority of the obtained results. In particular, the two points matched quite accurately but directly after the second one, the curve comes off the target behaviour (black dotted line). After that, the behaviour seems quite linear like for the pure nonlinear formulation.

Figure 4.29: Converged layout for $0.3 \times k_{top}x + 10 \times k_{top}x^3$.

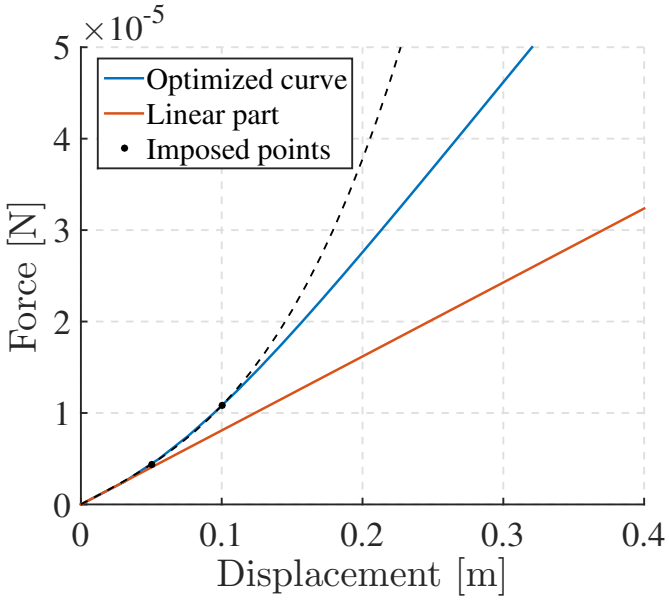


Figure 4.30: Force/Displacement curve for the mixed behaviour problem. $F = 0.3 \times k_{top}x + 10 \times k_{top}x^3$ in black dotted line.

4.8 Adding a new point

Lastly, some fast attempts have been made with the addition of a new point. Indeed, two points are enough to see the shape of a behaviour but the more points, the more accurate the method is. Therefore for nonlinear case, it will be useful to add many points. The first step is to try with only one added point for both method and see how it reacts. The added point is related to a displacement about 0.15[m]

4.8.1 Pure nonlinear

In order to have a point of comparison, the same function as in the previous section is taken, i.e. $F = 5 \times k_{top}x^2$. The results are illustrated in Fig. 4.31 and 4.32. The error related is about 1.54% which is bigger compared to the errors obtained before. It may be due to the fact that three points are harder to match that only two. On the layout, it seems to add some bars but the general shape is similar. However, on the force/displacement curve the last point is close but there is no matching. After this point, the curve directly changes its curvature.

Figure 4.31: Converged layout for $5 \times k_{top}x^2$ with 3 imposed points.

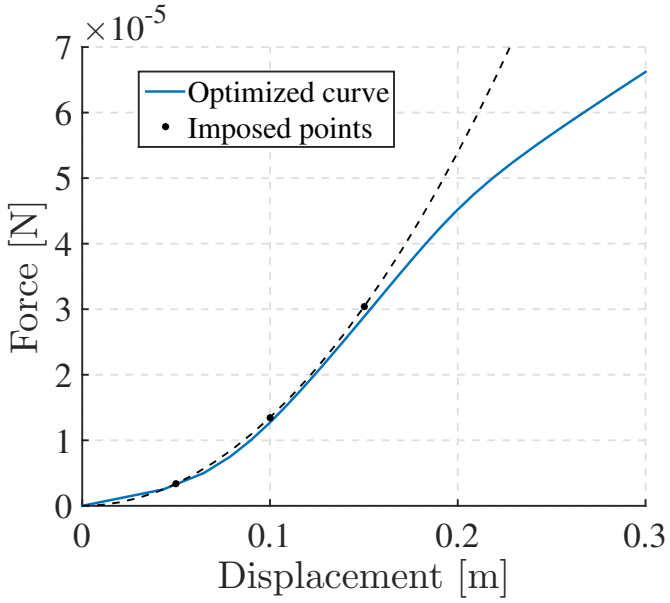


Figure 4.32: Force/Displacement curve for the pure nonlinear behaviour problem with 3 points. $F = 5 \times k_{top}x^2$ in black dotted line.

4.8.2 Mixed behaviour

The expression that gave general results is used, i.e. $F = 0.3 \times k_{top}x + 10 \times k_{top}x^3$. The layout is represented in Fig. 4.33 and as for the pure linear behaviour, the added point seems to add some bars on the structure. Unlike the pure behaviour, the force/displacement curve plotted in 4.34 is better and the behaviour keeps a close behaviour even with the last point. The error related is about 0.43% which is also big. It can be noted that the layout in Fig. 4.27 has not been found with the added point.

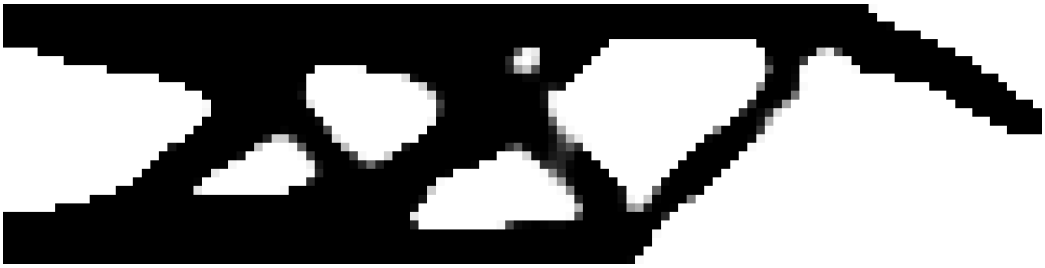


Figure 4.33: Converged layout for $0.3 \times k_{top}x + 10 \times k_{top}x^3$ with 3 imposed points.

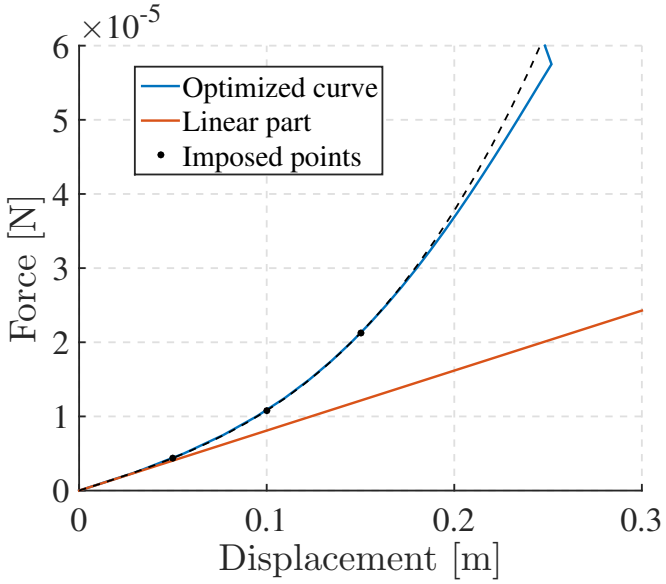


Figure 4.34: Force/Displacement curve for the mixed behaviour problem with 3 points. $F = 0.3 \times k_{top}x + 10 \times k_{top}x^3$ in black dotted line.

4.9 Conclusion

The prescribed curve optimization has been investigated. First, the problem has been stated in the form of an error minimization. To solve it, `MMA` can be used as usual. However, some special formulations of this solver allow to optimize its use for error problems. To assess the method, the linear optimized beam has been imposed by two points. The two formulations lead to satisfactory result and validate the problem statement. The next step was to impose a proportion of this behaviour. The first was 80%. In this condition, the special formulation of `MMA` has proven its interest and converges unlike the classical one. Then, the different types of norm have been studied on the same case, the results were similar but the L_1 norm shows small advantages. The linear behaviour has a limit for 30% of the reference. Under this limit, the layout does not converge. After, the nonlinear case is tested. The problem is that the polynomial functions tend to have a tiny slope on the first point. This issue leads to a null matrix and thus unconvergence of the process. Two methods have been tried to solve this issue. First, by increasing the stiffness number of the behaviour, which works but leads to an approximation of the behaviour by lines. The second idea is to add a minimum linear part to stabilize the problem. This method has

shown interesting result for the cubic spring with a pronounced nonlinear behaviour even after the imposed points. Nevertheless, the general result is not as good as this exception and a linear behaviour is observed after the last imposed point. Finally, some tests have been made with a third imposed point. It shows that the problem is more complex and the error resulting is bigger. However, for the mixed behaviour, the curve fits more the target behaviour even after the last point.

Chapter 5

Conclusion

The first purpose of this thesis was to extend an existing topology optimization model for prescribed force/displacement curve from 2D to 3D. Given the fact that any model was directly available, it has been chosen to restart from the beginning. It means that a new model of topology optimization has to be developed to impose force/displacement curve. The main goal of such optimization is that the resulting device could be used as a nonlinear spring in a vibration absorber. This goal could be achieved by two objective steps. The first one is to build a nonlinear topology optimization model. The second is to implement the prescribed problem to the model.

The first chapter has reviewed the types and modeling of the nonlinear vibrations. The main theory about the absorbers has been explained for the nonlinear case. To design this device, a nonlinear optimization model is needed. This is the objective of the second chapter. The first attempts were good but a convergence issue remained, i.e. mesh distortion. To solve that, the energy interpolation has been used and has shown promising results. However, the distortion is still present but controlled and if the displacement is really too big, mesh intersections can still happen. The simulations on the cantilever and the doubly clamped beams validate the model built. However, the doubly clamped beam reminds that the user should be careful with the parameters choice and always verify the converged layout. The performance of the code has shown that the nonlinear model is almost 70 times slower than the linear one. This model is used to impose some prescribed force/displacement curves. The linear case has been imposed with success but the optimizer can not go under 30% of the stiffest beam. Finally, the nonlinear case is imposed. A pure nonlinear behaviour does not give convenient results above the square. However, with the addition of a linear part, the problem is more stable and gives interesting results. A third point is imposed to see the behaviour. The resulting errors are bigger but especially for the mixed behaviour, the curve is better fitted.

In general, the results are promising, indeed, linear behaviour can be imposed until the limit. For the nonlinear behaviour, a linear part is needed but some stiffening devices have been designed. In each simulation, the user has to be careful with the parameters. There is no miracle formulation, a trial and error process will always be needed. Even if these results are interesting, it can be improved by several ways.

During all the prescribed curve optimization, the cantilever beam has been investigated. Due to the high computational requirements of these simulations, it was decided

to focus on one case. Bad results are obtained when testing the doubly clamped beam. This case is more sensitive and could highlight some weak points of the formulation. Indeed, during all the simulation, the load controlled procedure has been used and does not present an issue for the cantilever. The displacement controlled procedure has not been investigated and could be a first improvement of this thesis since it could be more stable (N.-H. Kim, [1], 2015). The Arc-length method is also mentioned sometimes and could solve this problem and take advantage of the doubly clamped domain.

The problem of the doubly clamped beam can be due to the used model. Indeed, there is a lot of compression in the center of the beam and the St-Venant Kirchhoff model is known to misbehave in compression. It does not seem a priority since the end-compliance problem was well solved with this model but the prescribed curve problem is more complex. Maybe a modified St-Venant Kirchhoff model or a Neo-Hookean one could solve this problem.

The error formulation has given all these results but the formulation can be asked. As an example, instead of some passage points, one can impose the slopes between the points or a combination of the two.

For the future, some ideas could be investigated. Obviously, the $3D$ generalization is still one and could allow to test in reality the obtained design.

Vibrations are always symmetric but in this work, a special focus was made on a single prescribed curve. Enforcing a symmetry condition could be another step in the realization of a real absorber. This could be done by applying a counter force but the problem formulation may not be optimized.

The main issue of nonlinear optimization has been well taken in charge by the energy interpolation method. As mentioned, this method is not perfect and the distortion is more controlled than avoided. Another way to solve this problem is the ECP method (Yoon et al., 2005, [30]). It can be an interesting way to change the model. The basic code was based on the SIMP method and the ECP method has been tested too late to give results. However, an ECP process could be derived from the SIMP version.

Another important aspect could be more related to the code itself. Indeed, the computational time is a huge problem and would become even more important in $3D$. To improve this, the parallel implementation could be more explored, even if MATLAB is not the best environment for that. Another way to reduce the time could be the implementation of a reduction method for nonlinear FEM.

Appendix A

Computer

All the simulations have been computed on my personal computer. Thus it is important to specify its important features. This is the Acer Aspire V 15 Nitro.

Operating system	Windows 10
Processor	Intel Core i5-6200U 2.3GHz
Memory	DDR4 8 GB
Storage	1 TB HDD 128 GB SSD

Table A.1: Features of the computer used for simulations.

Appendix B

MMA

The classical optimization problem described by Svanberg (1998, [23]) for the `MMA`, in its more general expression takes form as:

$$
\begin{aligned}
\min \quad & f_0(x) + a_0 z + \sum_{i=1}^{m} (c_i y_i + \frac{1}{2} d_i y_i^2) & \\
\text{s.t.} \quad & f_i(x) - a_i z - y_i \leq 0 & i = 1, ..., m \\
& x_{j,min} \leq x_j \leq x_{j,max} & j = 1, ..., n \\
& y_i \geq 0 & i = 1, ..., m \\
& z \geq 0 &
\end{aligned}
$$

where f_i are real functions, continuous and differentiable, $x_{j,min}$ and $x_{j,max}$ are real numbers and a_i, c_i, d_i are real non negative numbers.

Appendix C

MATLAB codes

C.1 Compliance optimization

```
%%%% Topology Optimization Codes for Geometrical Non-linearty %%%%
%%%% Liege 2018 %%%%
clear;clc;

%% Input parameters
beam = 2;

if beam ==1
    nelx = 80;
    nely = 20;%element numbers 80/20
    a = 1;
    b = 0.25;
    h = 0.1; % long beam size (m)
    volfrac = 0.5;
elseif beam == 2
    %nelx = 120;
    nelx = 60;
    nely = 40;
    %a = 3;
    a = 1.5;
    b = 1;
    h = 0.1; % long beam size (m)
    volfrac = 0.10;
elseif beam == 3
    nelx = 50;
    nely = 50;%element numbers
    a = 1;
    b = 1;
    h = 0.1; % long beam size (m)
    volfrac = 0.4;
end


penal = 3.;
if beam == 1
    P = 2.;
    detaP = 0.1;
else
    P = 1.;
    detaP = 0.1;
end
 % contuation method
if beam == 1
    rmin = round(nely*3/20); %radius: numbers of elements surround
else
    rmin = round(nely*4/40);
end
```

```
ft = 3; % filter
plane = 2; % plane stress condition

Total_elem = nely*nelx;
Total_node = (nely+1)*(nelx+1);
Total_dofs = 2*Total_node;

lambda = 5;
load = -lambda*1e-5;       % Load(N)
stepsIni = round(lambda);  % Load step
stop = 1e-3;               % Convergence criterion

%% Young Modulus (Pa)
E0 = 1; %3e9;
Emin = 1e-9*E0;
nu = 0.4;

%% PREPARE FINITE ELEMENT ANALYSIS
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+...
    repmat([0 1 2*nely+[2 3 0 1] -2 -1],nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);

%% node coordinates
detaxx = linspace(0,a,nelx+1);  % x>0
detayy = -linspace(0,b,nely+1); % y<0

[coordinates, nodes,nel,nnode] = MeshBeam2D(a,b,nelx,nely);
xx = coordinates(:,1);
yy = coordinates(:,2);          %% OFFSET y>0 %%

%% element size
aa = detaxx(2);
bb = detayy(2);

%% DEFINE LOADS AND SUPPORTS (LONG BEAM)
if beam ==1
    F = sparse(2*(nely+1)*(nelx)+nely+2,1,load,2*(nely+1)*(nelx+1),1);
    fixeddofs =[1:2*(nely+1)];
    xinit = volfrac;
    posF = 2*(nely+1)*nelx+nely+2;
elseif beam == 2
%     F = sparse(2*(nely+1)*(nelx/2)+2,1,load,2*(nely+1)*(nelx+1),1);
%     fixeddofs = union([1:2*(nely+1)], 2*(nelx)*(nely+1)+1:2*(nelx+1)*(nely+1));
%     xinit = min(1., volfrac);
%     posF = 2*(nely+1)*(nelx/2)+2;
    F = sparse(2*(nely+1)*(nelx)+2,1,load,2*(nely+1)*(nelx+1),1);
    fixeddofs = union([1:2*(nely+1)],...
        2*(nelx)*(nely+1)+1:2:2*(nelx+1)*(nely+1)-1);
    xinit = min(1., volfrac);
    posF = 2*(nely+1)*(nelx)+2;
elseif beam == 3
    posF1 = nelx+1;
    F = sparse(posF1,1,-load,2*(nely+1)*(nelx+1),1);
    posF = 2*(nely+1)*(nelx+1)-nelx-1;
    %F(posF) = load;
    fixeddofs = union(1:1:4, 2*(nely+1):-1:2*(nely+1)-3);
    xinit = volfrac;
    Kspring = sparse(Total_dofs,Total_dofs);
    Kspring(posF, posF) = Kspring(posF, posF) +0.1;
    Kspring(posF1, posF1) = Kspring(posF1, posF1) + 0.1;
elseif beam ==4
    posF = 2*(nely+1)*nelx+nely+1;
    F = sparse(posF,1,load,2*(nely+1)*(nelx+1),1);
    fixeddofs =[1:2*(nely+1)];
    xinit = volfrac;
    PerimLimit = 1;
end
```

```
118
119 U = zeros(2*(nely+1)*(nelx+1),min(size(F)));
120 alldofs = [1:2*(nely+1)*(nelx+1)];
121 freedofs = setdiff(alldofs,fixeddofs);
122
123 %% PREPARE FILTER
124 [dy,dx] = meshgrid(-ceil(rmin)+1:ceil(rmin)-1,-ceil(rmin)+1:ceil(rmin)-1);
125 H = max(0,rmin-sqrt(dx.^2+dy.^2));
126 Hs = conv2(ones(nely,nelx),H,'same');
127 beta1 = 500;
128 rho0 = 0.01;
129
130 %% INITIALIZE ITERATION
131
132 x = repmat(xinit,nely,nelx); %volfrac or 1. to stiffen
133
134 if beam == 1
135     beta = 1;
136 else
137     beta = 1;
138 end
139 omega = 0.5;
140 if ft == 1 || ft == 2
141   xPhys = x;
142 elseif ft == 3
143   xTilde = x;
144   xPhys = (tanh(beta*omega) + tanh(beta*(xTilde - omega)))...
145       /(tanh(beta*omega) + tanh(beta*(1 - omega)));
146 end
147
148 loop = 0;
149 loopbeta = 0;
150 change = 1;
151
152 %% Iinitiation of MMA
153 %   m    = The number of general constraints.
154 %   n    = The number of variables x_j.
155 m = 1;
156 if beam == 4
157     m = 2;
158 end
159 n = nelx*nely;
160
161 onen    = ones(n,1);
162 onem    = ones(m,1);
163 zeron   = zeros(n,1);
164 zerom   = zeros(m,1);
165 %  a     = Column vector with the constants a_i in the terms a_i*z.
166 %  c     = Column vector with the constants c_i in the terms c_i*y_i.
167 %  d     = Column vector with the constants d_i in the terms
168 a_mma   = zerom;
169 c_mma   = 10000*onem;
170 d_mma   = zerom;
171 a0      = 1;
172
173 xval    = x(:);
174 xold1   = xval;
175 xold2   = xold1;
176
177 iCont = 1;
178 ifint = edofMat';
179 ifint = ifint(:);
180
181 %bounds for design variables
182 move = 0.2;
183 alpha = move*onen; % For move limits
184
185 l = zeros(2*(nelx+1)*(nely+1),1);
186 l(posF) = 1;
187 %% START ITERATION
```

```
if beam == 1
    maxloop = 100;
    addBeta = 1;
else
    maxloop = 100;
    addBeta = 2;
end
while change > 0.01 && loop < maxloop
    tic % time start
    loop = loop + 1;
    loopbeta = loopbeta+1;
    xe = reshape(xPhys,nely*nelx,1);
    f = F/stepsIni; % step load

    %%%%%%%%%%%%% Preallocation %%%%%%%%%%%%%
    ue = zeros(8,1);
    ke = cell(Total_elem,1);% cell
    fint = cell(Total_elem,1);
    detaU = zeros(2*(nely+1)*(nelx+1),1);
    sK = zeros(64*Total_elem,1);
    sfint = zeros(8*Total_elem,1);

    %%%%%%%%%% Continuation scheme %%%%%%%%%%
    if(iCont>5 && P<penal)
      P = P + detaP;
      iCont = 1;
      fprintf('Penalty increased to %7.3f ', P);
    elseif(iCont>2 && P<2)
      P = P + detaP;
      iCont = 1;
      fprintf('Penalty increased to %7.3f ', P);
    end
    iCont = iCont+1;

    gamma_e = (tanh(beta1*rho0) + tanh(beta1*(xe.^P - rho0)))...
        /(tanh(beta1*rho0) + tanh(beta1*(1 - rho0)));
    %gamma_e = onen;

    if beam==2 && loop<2
        steps = 2;
    else
        steps = stepsIni;
    end

    %% Geometrica Nonlinearity FE-ANALYSIS
    for step=1:steps   % Incremental step
        Fext = f*step;% external nodal force
        Rmax=abs(load);
        iter=0;

        while (Rmax/abs(load)>stop)
            iter=iter+1;
            % Start Parpool:
            poolobj = gcp('nocreate'); % If no pool, do not create new one.
            if isempty(poolobj)
                parpool
            end
            % Using Parfor
            parfor ele=1:Total_elem%element number
                [ke{ele},fint{ele}] = CalKFint(ele,edofMat,xx,yy,aa,bb,...
                    h,U(:,1),nely,nu,gamma_e(ele), xe(ele));
            end

            for ele=1:Total_elem
                edof = edofMat(ele,:); % Element dofs
                % SIMP
                sK((ele-1)*64+1:(ele)*64) = ke{ele}*...
                    (Emin+xe(ele)^P*(E0-Emin));
                sfint((ele-1)*8+1:(ele)*8) = fint{ele}*...
                    (Emin+xe(ele)^P*(E0-Emin));
```

```
258             end
259
260             K = sparse(iK,jK,sK); % global stiffness matrix
261             Fint = sparse(ifint,ones(8*Total_elem,1),sfint);
262             R = Fext(:,1)-Fint;
263             if beam == 3
264                 %R = R + Kspring * U;
265                 K = K + Kspring;
266             end
267            %% Newton Raphson Algorithm
268             detaU(freedofs) = K(freedofs,freedofs)\R(freedofs);
269             U(:,1) = U(:,1)+detaU;
270             Rmax = max(abs(R(freedofs)));
271             blabla(iter) = Rmax;
272         end
273
274     end
275     Residu(loop) = Rmax;
276
277
278     %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
279     c = abs(l'*U);% end-compliance
280     cVec(loop) = c;
281
282     %% Ajoint method
283     L = zeros(2*(nely+1)*(nelx+1),1);
284     L(freedofs) = K(freedofs,freedofs)\(sign(load)*l(freedofs));
285     dc = zeros(nely*nelx,1);
286
287     for ele=1:Total_elem
288         edof = edofMat(ele,:);
289         le = L(edof);
290         dc(ele) = -P*(E0-Emin)*xe(ele)^(P-1)*le'*fint{ele};
291     end
292
293     if beam == 3
294         dc = -dc;
295     end
296     dc = reshape(dc,nely,nelx);
297     dv = ones(nely,nelx);
298
299     %% FILTERING/MODIFICATION OF SENSITIVITIES
300     if ft == 1
301         dc = conv2(dc.*xPhys,H,'same')./Hs./max(1e-3,xPhys);
302     elseif ft == 2
303         dc(:) = conv2(dc./Hs,H,'same');
304         dv(:) = conv2(dv./Hs,H,'same');
305     elseif ft == 3
306         dx = beta * (sech(beta*(xTilde - omega))).^2/...
307              (tanh(beta*omega) + tanh(beta*(1 - omega)));
308         dc(:) = conv2(dc.*dx./Hs,H,'same');
309         dv(:) = conv2(dv.*dx./Hs,H,'same');
310     end
311
312     %% MMA UPDATE OF DESIGN VARIABLES AND PHYSICAL DENSITIES
313
314     % Move limit strategy
315
316     xmin  = max(zeron, xval-alpha);
317     xmax  = min(onen, xval+alpha);
318     low     = xmin;
319     upp     = xmax;
320
321     % Objectif
322     f0val = c;
323     df0dx = dc(:);
324     df0dx2 = 0*df0dx;
325     % Constraint
326     fval = (sum(xPhys(:)))/ volfrac/(nelx*nely)-1;  % column vector
327     dfdx = dv(:)'/ volfrac/(nelx*nely) ;
```

```
328      dfdx2 = 0*dfdx;
329
330      %%%% The MMA subproblem is solved at the point xval:
331      [xmma,ymma,zmma,lam,xsi,eta,mu,zet,s,low,upp] = ...
332          mmasub(m,n,loop,xval,xmin,xmax,xold1,xold2, ...
333          f0val,df0dx,df0dx2,fval,dfdx,dfdx2,low,upp,a0,a_mma,c_mma,d_mma);
334      %%%% Some vectors are updated:
335      xold2 = xold1;
336      xold1 = xval;
337      xval  = xmma;
338      xnew = reshape(xval, nely, nelx);
339
340      if ft == 1
341          xPhys = xnew;
342      elseif ft == 2
343          xPhys(:) = conv2(xnew,H,'same')./Hs;
344      elseif ft == 3
345        xTilde(:) = conv2(xnew,H,'same')./Hs;
346        xPhys = (tanh(beta*omega) + tanh(beta*(xTilde - omega)))...
347             /(tanh(beta*omega) + tanh(beta*(1 - omega)));
348      end
349
350      change = max(abs(xnew(:)-x(:)));
351      x = xnew;
352      t = toc;% time finish
353
354      %% PRINT RESULTS
355      fprintf('It.:%4i Obj.:%8.4f Vol.:%1.3f ch.:%1.3f sec.:%3.3f\n',loop,c,...
356          (sum(xPhys(:)))/(nelx*nely),change,t);
357      VolVec(loop) = mean(xPhys(:));
358      %% PLOT DENSITIES
359      %[map] = brewermap(9,'*Blues');
360      colormap(gray); imagesc(1-xPhys); caxis([0 1]);
361      axis equal; axis off; drawnow;
362      Ulast(loop,:) = U;
363      if ft == 3 && beta < 100 && (loopbeta >= 10)
364          if beam == 1 && beta < 6
365              beta = beta+1;
366          else
367              beta = beta+2;
368          end
369          loopbeta = 0;
370          change = 1;
371          fprintf('Parameter beta increased to %g.\n',beta);
372      end
373
374  end
375
376  %%
377
378  if beam == 2
379      xPhys2 = fliplr(xPhys);
380      colormap(gray); imagesc(1-[xPhys, xPhys2]); caxis([0 1]);
381      axis equal; axis off; drawnow;
382  end
383
384   for ely=1:nely
385          for elx=1:nelx
386              nn1=(nely+1)*(elx-1)+ely;
387              nn2=(nely+1)*elx+ely;
388              Ue= U([2*nn1-1;2*nn1;2*nn2-1;2*nn2;2*nn2+1;2*nn2+2;2*nn1+1;2*nn1+2],1);
389              xxx=[Ue(1)+xx(nn1),Ue(3)+xx(nn2),Ue(5)+xx(nn2+1),Ue(7)+xx(nn1+1)]';
390              yyy=[Ue(2)+yy(nn1),Ue(4)+yy(nn2),Ue(6)+yy(nn2+1),Ue(8)+yy(nn1+1)]';
391              patch(xxx,yyy,[1-xPhys(ely,elx) 1-xPhys(ely,elx) 1-xPhys(ely,elx)])
392          end
393   end
394   axis equal;
395   drawnow;
```

C.2 Prescribed curve

```
%%%% Topology Optimization Codes Prescribed Force/Displacement Curves %%%%
%%%% In Liege 2018 %%%%
clear;
clc;

%% Input parameters
beam = 1;

if beam ==1          % Cantilever beam
    nelx = 120;      % Element numbers
    nely = 30;
    a = 1;           % Length of the beam (m)
    b = 0.25;        % Width
    h = 0.1;
    volfrac = 0.5;   % Volume ratio
elseif beam == 2     % Doubly clamped beam
    %nelx = 120;
    nelx = 60;       % Element numbers
    nely = 40;
    a = 3;
    a = 1.5;         % Length of the beam (m)
    b = 1;           % Width
    h = 0.1;
    volfrac = 0.10; % Volume ratio
end

penal = 3.;          % Maximum penalty
if beam == 1
    P = 2.;          % Starting penalty
    detaP = 0.1;     % Penalty increment
else
    P = 1.;
    detaP = 0.1;
end

rmin = 4;        % Radius filter
% Type of filter (1 = sensitivity, 2 = density, 3 = density + Heaviside)
ft = 3;
plane = 2;

Total_elem = nely*nelx;
Total_node = (nely+1)*(nelx+1);
Total_dofs = 2*Total_node;

if beam == 1
    ktop = 2.697e-04; % Optimal stiffness number
else
    ktop = 4.7619e-04;
end

% Exponent of the prescribed behaviour
poly = 3;
% Prescribed displacements and forces

uPresc = [-0.05  -0.075 -0.1];
numImp = length(uPresc); %Number of passage points
load = 0.3*ktop*uPresc + 5.*ktop*sign(min(uPresc))^(poly+1)*uPresc.^poly;

maxStep = 4;                                        % Load step number
steps = sort([1:maxStep, load(1:end-1)/load(end)*maxStep]);% Add prescribed points
stop = 1e-3;                                        % Convergence threshold

%% Young Modulus (Pa)
E0 = 1;
Emin = 1e-9*E0;
nu = 0.4;
```

```
67
68  %% PREPARE FINITE ELEMENT ANALYSIS
69  nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
70  edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
71  edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],nelx*nely,1);
72  iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
73  jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
74
75  %% NODE COORDINATES
76  detaxx = linspace(0,a,nelx+1);  % x>0
77  detayy = -linspace(0,b,nely+1); % y<0
78
79  [coordinates, nodes,nel,nnode] = MeshBeam2D(a,b,nelx,nely);
80  xx = coordinates(:,1);
81  yy = coordinates(:,2);
82
83  %% ELEMENT SIZE
84  aa = detaxx(2);
85  bb = detayy(2);
86
87  %% DEFINE LOADS AND SUPPORTS (LONG BEAM)
88  if beam ==1
89      posF = 2*(nely+1)*nelx+nely+2;
90      F = sparse(posF,1,sign(load(1))*max(abs(load)),2*(nely+1)*(nelx+1),1);
91      fixeddofs =[1:2*(nely+1)];
92  elseif beam == 2
93  % posF = 2*(nely+1)*(nelx/2)+2;
94  % F = sparse(posF,1,sign(load(1))*max(abs(load)),2*(nely+1)*(nelx+1),1);
95  % fixeddofs = union([1:2*(nely+1)], 2*(nelx)*(nely+1)+1:2*(nelx+1)*(nely+1));
96      F = sparse(2*(nely+1)*(nelx)+2,1,sign(load(1))*max(abs(load)),2*(nely+1)*(nelx+1),1);
97      fixeddofs = union([1:2*(nely+1)], 2*(nelx)*(nely+1)+1:2:2*(nelx+1)*(nely+1)-1);
98      xinit = min(1., volfrac);
99      posF = 2*(nely+1)*(nelx)+2;
100 end
101
102 U = zeros(2*(nely+1)*(nelx+1),1);
103 alldofs = [1:2*(nely+1)*(nelx+1)];
104 freedofs = setdiff(alldofs,fixeddofs);
105
106 %% PREPARE FILTER
107 [dy,dx] = meshgrid(-ceil(rmin)+1:ceil(rmin)-1,-ceil(rmin)+1:ceil(rmin)-1);
108 H = max(0,rmin-sqrt(dx.^2+dy.^2));
109 Hs = conv2(ones(nely,nelx),H,'same');
110
111 % Parameters for interpolation energy method
112 beta1 = 500;
113 rho0 = 0.01;
114
115 %% INITIALIZE ITERATION
116 x = repmat(volfrac,nely,nelx);
117 % Parameters for Heaviside projection
118 beta = 1;
119 omega = 0.5;
120
121 if ft == 1 || ft == 2
122   xPhys = x;
123 elseif ft == 3
124   xTilde = x;
125   xPhys = (tanh(beta*omega) + tanh(beta*(xTilde - omega)))...
126       /(tanh(beta*omega) + tanh(beta*(1 - omega)));
127 end
128
129 loop = 0;
130 change = 1;
131
132 %% Position of prescribed displacement
133 l = zeros(2*(nelx+1)*(nely+1),1);
134 l(posF) = 1;
135 fImp = zeros(maxStep,1);
136
```

```
%% Iinitiation of MMA
%   m    = The number of general constraints.
%   n    = The number of variables x_j.

m = 2*numImp+1;
n = nelx*nely;

onen    = ones(n,1);
onem    = ones(m,1);
zeron   = zeros(n,1);
zerom   = zeros(m,1);
%  a     = Column vector with the constants a_i
%  c     = Column vector with the constants c_i
%  d     = Column vector with the constants d_i
normType = 2; % Norm type (1 = Euclidean, 2 = L1, 3 = Min max)

if normType == 1
    a_mma   = zerom;
    c_mma   = zerom;
    c_mma(m:end)   = 10000;
    d_mma   = 2*onem;
    d_mma(m:end)   = 0;
    a0      = 1;
elseif normType == 2
    a_mma   = zerom;
    c_mma   = onem;
    c_mma(m:end)   = 10000;
    d_mma   = zerom;
    a0      = 1;
elseif normType == 3
    a_mma   = onem;
    a_mma(m:end)   = 0;
    c_mma   = onem*10000;
    d_mma   = zerom;
    a0      = 1;
end

xval    = x(:);
xold1   = xval;
xold2   = xold1;

iCont = 1;
ifint = edofMat';
ifint = ifint(:);

% Bounds for move limit
move = 0.2;
alpha = move*onen;

loopbeta = 0;

%% START ITERATION
while loop < 150
    tic                                 % Time start
    loop = loop + 1;
    loopbeta = loopbeta + 1;

    xe = reshape(xPhys,nely*nelx,1);
    f = F/maxStep;                      % Step load

    % Preallocation
    ue = zeros(8,1);
    ke = cell(Total_elem,1);            % Cell
    fint = cell(Total_elem,1);
    fintImp = cell(Total_elem,maxStep);
    detaU = zeros(2*(nely+1)*(nelx+1),1);
    sK = zeros(64*Total_elem,1);
    sfint = zeros(8*Total_elem,1);

    % Continuation scheme
```

```
    if(iCont>5 && P<penal)
      P = P + detaP;
      iCont = 1;
      fprintf('Penalty increased to %7.3f ', P);
    elseif(iCont>3 && P<2)
      P = P + detaP;
      iCont = 1;
      fprintf('Penalty increased to %7.3f ', P);
    end
    iCont = iCont+1;

    % Energy interpolation variable
    gamma_e = (tanh(beta1*rho0) + tanh(beta1*(xe.^P - rho0)))...
        /(tanh(beta1*rho0) + tanh(beta1*(1 - rho0)));

    %% Ajoint method part 1
    L1 = zeros(2*(nely+1)*(nelx+1),1);
    L2 = zeros(2*(nely+1)*(nelx+1),1);
    L3 = zeros(2*(nely+1)*(nelx+1),1);
    dc = zeron;

    %% Geometrica Nonlinearity FE-ANALYSIS
    for iStep=1:length(steps)           % Incremental step
        Fext = f*steps(iStep);          % External nodal force
        Rmax=1;
        iter=0;
        while (Rmax/min(abs(load))>stop)
            iter=iter+1;
            % Start Parpool
            poolobj = gcp('nocreate');  % Create new one if none.
            if isempty(poolobj)
                parpool
            end
            % Using Parfor
            parfor ele=1:Total_elem     % Element number
                [ke{ele},fint{ele}] = CalKFint(ele,edofMat,xx,yy,aa,...
                    bb,h,U,nely,nu,gamma_e(ele),x(ele));
            end

            for ele=1:Total_elem
                edof = edofMat(ele,:);  % Element dofs
                % SIMP
                sK((ele-1)*64+1:(ele)*64) = ke{ele}*...
                    (Emin+xe(ele)^P*(E0-Emin));
                sfint((ele-1)*8+1:(ele)*8) = fint{ele}*...
                    (Emin+xe(ele)^P*(E0-Emin));
            end

            K = sparse(iK,jK,sK);       % Global stiffness matrix
            Fint = sparse(ifint,ones(8*Total_elem,1),sfint);
            R = Fext-Fint;
           %% Newton Raphson Algorithm
            detaU(freedofs) = K(freedofs,freedofs)\R(freedofs);
            U = U+detaU;
            Rmax = max(abs(R(freedofs)));
        end
        if steps(iStep)/steps(end) == load(1)/load(end)
            U1 = U;
            fint1 = fint;
            K1 = K;
        end
         if steps(iStep)/steps(end) == load(2)/load(end)
             U2 = U;
             fint2 = fint;
             K2 = K;
         end
    end
    U3 = U;
    fint3 = fint;
    K3 = K;
```

```
277
278     %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
279
280     uResult(1) = l'*U1;
281     uResult(2) = l'*U2;
282     uResult(3) = l'*U3;
283
284     weight = 100;
285     hi = weight*(uResult - uPresc)./abs(uPresc);
286     % Compute the objective function unused in this formulation of MMA
287     c =  sum(hi.^2);
288     cVec(loop) = c;
289
290     L1(freedofs) = K1(freedofs,freedofs)\l(freedofs);
291     L2(freedofs) = K2(freedofs,freedofs)\l(freedofs);
292     L3(freedofs) = K3(freedofs,freedofs)\l(freedofs);
293
294     %% Adjoint method
295     for ele=1:Total_elem
296         edof = edofMat(ele,:);
297         le1 = L1(edof,:);
298         le2 = L2(edof,:);
299         le3 = L3(edof,:);
300         dh(1,ele) = -weight/abs(uPresc(1))*P*(E0-Emin)*xe(ele)^(P-1)...
301             * (le1'*fint1{ele});
302         dh(2,ele) = -weight/abs(uPresc(2))*P*(E0-Emin)*xe(ele)^(P-1)...
303             * (le2'*fint2{ele});
304         dh(3,ele) = -weight/abs(uPresc(3))*P*(E0-Emin)*xe(ele)^(P-1)...
305             * (le3'*fint3{ele});
306
307     end
308
309     dv = ones(nely,nelx);
310
311     %% FILTERING/MODIFICATION OF SENSITIVITIES
312     if ft == 1
313         dc = conv2(dc.*xPhys,H,'same')./Hs./max(1e-3,xPhys);
314     elseif ft == 2
315         dv(:) = conv2(dv./Hs,H,'same');
316
317     elseif ft == 3
318         dx = beta * (sech(beta*(xTilde - omega))).^2/...
319             (tanh(beta*omega) + tanh(beta*(1 - omega)));
320         dh(1,:) = reshape(conv2(reshape(dh(1,:),nely,nelx).*dx...
321             ./Hs,H,'same'),1,nel);
322         dh(2,:) = reshape(conv2(reshape(dh(2,:),nely,nelx).*dx...
323             ./Hs,H,'same'),1,nel);
324         dh(3,:) = reshape(conv2(reshape(dh(3,:),nely,nelx).*dx...
325             ./Hs,H,'same'),1,nel);
326         dv(:) = conv2(dv.*dx./Hs,H,'same');
327     end
328
329     %% MMA UPDATE OF DESIGN VARIABLES AND PHYSICAL DENSITIES
330
331     % column vector
332     % Move limit strategy
333     xmin  = max(zeron, xval-alpha);
334     xmax  = min(onen, xval+alpha);
335     low     = xmin;
336     upp     = xmax;
337
338     % Objection
339     f0val = 0;
340     df0dx = dc(:)*0;
341     df0dx2 = 0*df0dx;
342
343     % Constraints
344     fval = [hi'; -hi'; (sum(xPhys(:)))/ volfrac/(nelx*nely)- 1];
345     dfdx = [dh; -dh; dv(:)'/ volfrac/(nelx*nely)];
346     dfdx2 = 0*dfdx;
```

```
347
348       %%%% The MMA subproblem is solved at the point xval:
349       [xmma,ymma,zmma,lam,xsi,eta,mu,zet,s,low,upp] = ...
350           mmasub(m,n,loop,xval,xmin,xmax,xold1,xold2, ...
351           f0val,df0dx,df0dx2,fval,dfdx,dfdx2,low,upp,a0,a_mma,c_mma,d_mma);
352       %%%% Some vectors are updated:
353       xold2 = xold1;
354       xold1 = xval;
355       xval  = xmma;
356       xnew = reshape(xval, nely, nelx);
357
358       if ft == 1
359           xPhys = xnew;
360       elseif ft == 2
361           xPhys(:) = conv2(xnew,H,'same')./Hs;
362       elseif ft == 3
363         xTilde(:) = conv2(xnew,H,'same')./Hs;
364         xPhys = (tanh(beta*omega) + tanh(beta*(xTilde - omega)))...
365             /(tanh(beta*omega) + tanh(beta*(1 - omega)));
366       end
367
368       change = max(abs(xnew(:)-x(:)));
369       changeVec(loop) = change;
370
371       x = xnew;
372       t = toc;                              % Time finish
373       %% PRINT RESULTS
374       fprintf('It.:%4i Obj.:%8.4f Vol.:%1.3f ch.:%1.3f sec.:%3.3f\n',loop,c,...
375           (sum(xPhys(:)))/(nelx*nely),change,t);
376       VolVec(loop) = mean(xPhys(:));
377       %% PLOT DENSITIES
378
379       colormap(gray); imagesc(1-xPhys);
380       caxis([0 1]); axis equal; axis off; drawnow;
381       Ulast(loop,:) = U;
382
383       if loop <60
384           maxLoopBeta = 10;
385       else
386           maxLoopBeta = 10;
387       end
388       if ft == 3 && beta < 200 && (loopbeta >= maxLoopBeta)
389           if  beta < 6
390               beta = beta+1;
391           elseif beam == 1 && beta > 5
392               beta = round(beta*2);
393           elseif beam == 2 && beta > 5
394               beta = beta +2;
395           end
396           loopbeta = 0;
397           change = 1;
398           fprintf('Parameter beta increased to %g.\n',beta);
399       end
400    end
```

C.3 Nonlinear FEM analysis

```
1  function [Uresult, Fresult, time, itInt, Ulast] = ...
2      Analysis(xPhys, a, b, h , nelx, nely, beam, P)
3  tic
4  nu = 0.4;
5  E0 = 1;
6  Emin = 1e-9;
7  Total_elem = nely*nelx;
8  Total_node = (nely+1)*(nelx+1);
```

```
Total_dofs = 2*Total_node;

nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);

detaxx = linspace(0,a,nelx+1);  % x>0
detayy = -linspace(0,b,nely+1); % y<0

[coordinates, nodes,nel,nnode] = MeshBeam2D(a,b,nelx,nely);
xx = coordinates(:,1);
yy = coordinates(:,2);          %% OFFSET y>0 %%

%% element size
aa = detaxx(2);
bb = detayy(2);

lambda = 10;
load = -lambda*1e-5;         %load (N)
steps = lambda*4;           % load step numbers
stop = 1e-2;          % convergence criterion

Total_elem = nely*nelx;
Total_node = (nely+1)*(nelx+1);
Total_dofs = 2*Total_node;

xe = reshape(xPhys,nely*nelx,1);

detaxx = linspace(0,a,nelx+1);  % x>0
detayy = -linspace(0,b,nely+1); % y<0

[coordinates, nodes,nel,nnode] = MeshBeam2D(a,b,nelx,nely);
xx = coordinates(:,1);
yy = coordinates(:,2);          %% OFFSET y>0 %%

%% element size
aa = detaxx(2);
bb = detayy(2);

%% DEFINE LOADS AND SUPPORTS (LONG BEAM)
if beam ==1
    F = sparse(2*(nely+1)*(nelx)+nely+2,1,load,2*(nely+1)*(nelx+1),1);
    fixeddofs =[1:2*(nely+1)];
    posF = 2*(nely+1)*nelx+nely+2;
elseif beam == 2
%     F = sparse(2*(nely+1)*(nelx/2)+2,1,load,2*(nely+1)*(nelx+1),1);
%     fixeddofs = union([1:2*(nely+1)], 2*(nelx)*(nely+1)+1:2*(nelx+1)*(nely+1));
%     posF = 2*(nely+1)*(nelx/2)+2;
    F = sparse(2*(nely+1)*(nelx)+2,1,load,2*(nely+1)*(nelx+1),1);
    fixeddofs = union([1:2*(nely+1)], 2*(nelx)*(nely+1)+1:2:2*(nelx+1)*(nely+1)-1);
    posF = 2*(nely+1)*(nelx)+2;
end
f = F/steps; % step load
l = zeros(2*(nelx+1)*(nely+1),1);
l(posF) = 1;

U = zeros(2*(nely+1)*(nelx+1),min(size(F)));
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);

% Preallocation
ue = zeros(8,1);
ke = cell(Total_elem,1);% cell
fint = cell(Total_elem,1);
detaU = zeros(2*(nely+1)*(nelx+1),1);
sK = zeros(64*Total_elem,1);
sfint = zeros(8*Total_elem,1);

```

```
ifint = edofMat';
ifint = ifint(:);

beta1 = 500;
rho0 = 0.01;
gamma_e = (tanh(beta1*rho0) + tanh(beta1*(xe.^P - rho0)))...
    /(tanh(beta1*rho0) + tanh(beta1*(1 - rho0)));
%gamma_e = gamma_e*0;

U = zeros(Total_dofs,1);
Uresult = zeros(steps,1);
    for step=1:steps   % Incremental step
        Fext = f*step;% external nodal force
        Rmax=1;
        iter=0;
        itInt(step) = 0;
        while (Rmax/abs(step/steps*load)>stop)
            itInt(step) = itInt(step)+1;
            iter=iter+1;
            % Start Parpool:
            poolobj = gcp('nocreate'); % If no pool, do not create new one.
            if isempty(poolobj)
                parpool
            end
            % Using Parfor
            parfor ele=1:Total_elem %element number
                [ke{ele},fint{ele}] = CalKFint(ele,edofMat,xx,yy,aa,...
                    bb,h,U,nely,nu,gamma_e(ele),xe(ele));
            end

            for ele=1:Total_elem
                edof = edofMat(ele,:);% element dofs
                % SIMP
                sK((ele-1)*64+1:(ele)*64) = ke{ele}*(Emin+xe(ele)^P*(E0-Emin));
                sfint((ele-1)*8+1:(ele)*8) = fint{ele}*(Emin+xe(ele)^P*(E0-Emin));
            end

            K = sparse(iK,jK,sK); % global stiffness matrix
            Fint = sparse(ifint,ones(8*Total_elem,1),sfint);
            R = Fext-Fint;
           %% Newton Raphson Algorithm
            detaU(freedofs) = K(freedofs,freedofs)\R(freedofs);
            U = U+detaU;
            Rmax = max(abs(R(freedofs)));
        end
        Uresult(step+1) = (l'*U);
        Ulast(step,:) = U;
    end
time = toc;
Uresult(1) = 0;
Fresult = 0:load/steps:load;
%%
figure
hold on
grid on;
ax = gca;
ax.GridLineStyle = '--';
plot(-Uresult, -Fresult, 'linewidth', 2)
%scatter(-Uresult(1:steps/10:end), -Fresult(1:steps/10:end)/1000,'filled')
xlabel('Displacement [m]','Interpreter','latex')
ylabel('Force [N]','Interpreter','latex')
set(gca,'fontsize',28,'fontname','Times','LineWidth',1.5);

% le=legend('Linear optimization','Uniform repartition');
% set(le,'Fontsize',24,'Location','Best');

%%

PlotMesh([xx, yy]+ [U(1:2:end), U(2:2:end)], nodes)
figure
```

```
149   for ely=1:nely
150           for elx=1:nelx
151               nn1=(nely+1)*(elx-1)+ely;
152               nn2=(nely+1)*elx+ely;
153               Ue= U([2*nn1-1;2*nn1;2*nn2-1;2*nn2;2*nn2+1;2*nn2+2;2*nn1+1;2*nn1+2],1);
154               xxx=[Ue(1)+xx(nn1),Ue(3)+xx(nn2),Ue(5)+xx(nn2+1),Ue(7)+xx(nn1+1)]';
155               yyy=[Ue(2)+yy(nn1),Ue(4)+yy(nn2),Ue(6)+yy(nn2+1),Ue(8)+yy(nn1+1)]';
156               patch(xxx,yyy,[1-xPhys(ely,elx) 1-xPhys(ely,elx) 1-xPhys(ely,elx)])
157           end
158   end
159   axis equal;
160   drawnow;
```

C.4 Constitutive model

```
1   %% MATERIAL PROPERTIES
2   function [D]=C(nu)
3   % nu=0.4;
4   % if plane==1 % plane strain
5   %     d=(Emin+xe(ele)^penal*(E0-Emin))/((1+nu)*(1-2*nu));
6   %     D=d*[1-nu  nu    0
7   %           nu  1-nu   0
8   %           0    0  (1-2*nu)/2];
9   % elseif plane==2 % plane stress
10      d=1/(1-nu*nu); % unit elastic matrix
11      D=d*[ 1  nu    0
12            nu  1    0
13            0  0 (1-nu)/2];
14  % end
15  end
```

C.5 Computation stiffness element

```
1   function [ke,Int]=knonlinear(ele,xx,yy,D,aa,bb,h,ue,nely,gamma_e, x_e)
2       ke = zeros(8,8);
3       M = zeros(4,4);
4       ele_stress = zeros(3,1);
5       Int = zeros(8,1);
6       gauss_point = 1/sqrt(3);
7       Id2 = eye(2);
8
9       % point1
10      s = -gauss_point;
11      t = -gauss_point;
12      [B,~,ele_strain,G] = shape(s,t,ele,xx,yy,aa,bb,ue,nely,gamma_e);
13      ele_stress = D*ele_strain;
14      M = [ele_stress(1)*Id2 ele_stress(3)*Id2
15           ele_stress(3)*Id2 ele_stress(2)*Id2];
16      ke1 = B'*D*B + gamma_e * G'*M*G;
17      Int1 = B'*ele_stress;
18
19      % point2
20      s = gauss_point;
21      t = -gauss_point;
22      [B,~,ele_strain,G] = shape(s,t,ele,xx,yy,aa,bb,ue,nely,gamma_e);
23      ele_stress = D*ele_strain;
24      M = [ele_stress(1)*Id2 ele_stress(3)*Id2
25           ele_stress(3)*Id2 ele_stress(2)*Id2];
26      ke2 = B'*D*B + gamma_e * G'*M*G;
27      Int2 = B'*ele_stress;
28
```

```
    % point3
    s = gauss_point;
    t = gauss_point;
    [B,~,ele_strain,G] = shape(s,t,ele,xx,yy,aa,bb,ue,nely,gamma_e);
    ele_stress = D*ele_strain;
    M = [ele_stress(1)*Id2 ele_stress(3)*Id2
         ele_stress(3)*Id2 ele_stress(2)*Id2];
    ke3 = B'*D*B + gamma_e * G'*M*G;
    Int3 = B'*ele_stress;

    % point4
    s = -gauss_point;
    t = gauss_point;
    [B,detJ,ele_strain,G] = shape(s,t,ele,xx,yy,aa,bb,ue,nely,gamma_e);
    ele_stress = D*ele_strain;
    M = [ele_stress(1)*Id2 ele_stress(3)*Id2
         ele_stress(3)*Id2 ele_stress(2)*Id2];
    ke4 = B'*D*B + gamma_e * G'*M*G;
    Int4 = B'*ele_stress;

    % Integration
    ke  = h*detJ*(ke1+ke2+ke3+ke4);
    Int = h*detJ*(Int1+Int2+Int3+Int4);
end
```

C.6 Computation stiffness matrix

```
%% element stiffness matrix and internal nodal force
function [ke,fint] = CalKFint(ele,edofMat,xx,yy,aa,bb,h,U,nely,nu,gamma_e, x_e)
    edof = edofMat(ele,:);% element dofs
    ue = U(edof);
    [D] = C(nu);% unit elastic matrix
    [ke, fint] = knonlinear(ele,xx,yy,D,aa,bb,h,ue,nely,gamma_e, x_e);
end
```

C.7 Shape function

```
%% shape function (Iso-parametric Q-4 element)
function [B,detJ,ele_strain,G,B0]=shape(s,t,ele,xx,yy,aa,bb,ue,nely,gamma_e)
     B0 = zeros(3,8);
     Bl = zeros(3,8);
     B = zeros(3,8);
     G = zeros(4,8);
     A = zeros(3,4);
     ele_strain = zeros(3,1);
     Theta = zeros(4,1);
    %% Interpolation function
    % N=[(1-s)*(1-t),(1+s)*(1-t),(1+s)*(1+t),(1-s)*(1+t)]/4;
    dNds = [t-1,1-t,1+t,-1-t]/4;
    dNdt = [s-1,-1-s,1+s,1-s]/4;
    %% element nodes coordinates
    xxyy = zeros(4,2);
    %%%%%%%%%%%%%%%%%%%%%%%%%%
    node = ele+ceil(ele/nely)-1;
    %%%%%%%%%%%%%%%%%%%%%%%%%%
    xxyy(4,1) = xx(node);
    xxyy(3,1) = xxyy(4,1)+aa;
    xxyy(2,1) = xxyy(3,1);
    xxyy(1,1) = xxyy(4,1);
    xxyy(4,2) = yy(node);
    xxyy(3,2) = xxyy(4,2);
```

```
25      xxyy(2,2) = xxyy(3,2)+bb;
26      xxyy(1,2) = xxyy(2,2);
27      %% Jacobi matrix
28      % J=zeros(2,2);
29      J11 = dNds*xxyy(:,1);
30      J12 = dNds*xxyy(:,2);
31      J21 = dNdt*xxyy(:,1);
32      J22 = dNdt*xxyy(:,2);
33      % J=[J11 J12
34      %    J21 J22];
35      detJ = J11*J22-J12*J21;
36      % inv(J)=[J22 -J12
37      %         -J21 J11]/detJ;
38      % [dNdx;dNdy]=inv(J)*[dNds;dNdt];
39
40      dNdx = (J22*dNds-J12*dNdt)/detJ;
41      dNdy = (J11*dNdt-J21*dNds)/detJ;
42
43      B0=[dNdx(1)    0       dNdx(2)    0       dNdx(3)    0       dNdx(4)    0
44             0     dNdy(1)     0      dNdy(2)     0      dNdy(3)     0      dNdy(4)
45          dNdy(1) dNdx(1)   dNdy(2) dNdx(2)   dNdy(3) dNdx(3)   dNdy(4) dNdx(4)];
46
47      G=[dNdx(1)    0       dNdx(2)    0       dNdx(3)    0       dNdx(4)    0
48           0     dNdx(1)      0     dNdx(2)     0      dNdx(3)     0      dNdx(4)
49         dNdy(1)    0       dNdy(2)    0       dNdy(3)    0       dNdy(4)    0
50           0     dNdy(1)     0      dNdy(2)     0      dNdy(3)     0      dNdy(4)];
51
52      Theta = G*ue;
53      Thetax = [Theta(1);Theta(2)];
54      Thetay = [Theta(3);Theta(4)];
55      A = [Thetax' 0 0;0 0 Thetay';Thetay' Thetax'];
56      Bl = A*G;
57
58      B =  Bl .* gamma_e + B0; %(B0+Bl)* gamma_e - gamma_e * B0 + B0;
59
60      ele_strain_0 = B0*ue;
61      ele_strain = ele_strain_0 + 0.5 * Bl * ue * gamma_e;
62
63  end
```

Bibliography

[1] N.-H. Kim. *Introduction to Nonlinear Finite Element Analysis*. Springer Science, 2015.

[2] J.P. Noël and G. Kerschen. Nonlinear system identification in structural dynamics: 10 more years of progress. *Mechanical Systems and Signal Processing*, 83:2 – 35, 2017.

[3] J.P. Noël and G. Kerschen. Nonlinear vibrations of nonlinear of aerospace structures. University of Liege, 2018.

[4] G. Habib and G. Kerschen. A principle of similarity for nonlinear vibration absorbers. *Physica D: Nonlinear Phenomena*, 332:1 – 8, 2016.

[5] G. Habib, T. Detroux, R. Viguié, and G. Kerschen. Nonlinear generalization of den hartog's equal-peak method. *Mechanical Systems and Signal Processing*, 52-53:17 – 28, 2015.

[6] O. Sigmund B. S. Lazarov, K. E. Meyer, and J. Alexandersen. Experimental validation of additively manufactured optimized shapes for passive cooling. *Applied Energy*, 226:330 – 339, 2018.

[7] N. Vasios. Nonlinear analysis of structure. Harvard University, 2015.

[8] G.H. Yoon. Maximizing the fundamental eigenfrequency of geometrically nonlinear structures by topology optimization based on element connectivity parameterization. *Computers and Structures*, 88(1):120 – 133, 2010.

[9] J.P. Den Hartog. *Mechanical Vibrations*. McGraw-Hill, 1934.

[10] G. Kerschen, K. Worden, A. F. Vakakis, and J.C. Golinval. Past, present and future of nonlinear system identification in structural dynamics. *Mechanical Systems and Signal Processing*, 20(3):505 – 592, 2006.

[11] G.R. Tomlinson K. Worden. *Nonlinearity in structural dynamics*. Institute of Physics Publishing, 2001.

[12] J.P. Noël and G. Kerschen. Frequency-domain subspace identification for nonlinear mechanical systems. *Mechanical Systems and Signal Processing*, 40(2):701 – 717, 2013.

[13] O. Nishihara T. Asami. Closed-form solutions to the exact optimizations of dynamic vibration absorbers (minimizations of the maximum amplitude magnification factors). *Journal of Vibration and Acoustics*, 125:398–405, 2003.

[14] G. Duffing. *Erzwungene Schwingungen bei Veränderlicher Eigenfrequenz*. F. Vieweg u. Sohn, Braunschweig, 1918.

[15] T. Norton. *Topology Optimization Key to Additive Manufacturing*, 2013. `http://on3dprinting.com/tag/tony-norton/`.

[16] S. Wasserman. *Why Marry 3D Printing with Topology Optimization?*, 2015. `https://www.engineering.com/DesignSoftware/DesignSoftwareArticles/ArticleID/11008/3D-Printing-Brings-Out-the-Full-Potential-of-Topology-Optimization.aspx`.

[17] M. P. Bendsøe. Optimal shape design as a material distribution problem. *Structural optimization*, 1(4):193–202, Dec 1989.

[18] O. Sigmund T. Buhl, C.B.W. Pedersen. Stiffness design of geometrically nonlinear structures using topology optimization. *Structural and Multidisciplinary Optimization*, 9(2):93–104, 2000.

[19] T. E. Bruns and D. A. Tortorelli. Topology optimization of non-linear elastic structures and compliant mechanisms. *Computer Methods in Applied Mechanics and Engineering*, 190(26):3443 – 3459, 2001.

[20] M. Schevenels B.S. Lazarov O. Sigmund E. Andreassen, A. Clausen. Efficient topology optimization in matlab using 88 lines of code. *Structural and Multidisciplinary Optimization*, 43(1):1–16, 2011.

[21] M.s Wallin, N. Ivarsson, and D. Tortorelli. Stiffness optimization of non-linear elastic structures. *Computer Methods in Applied Mechanics and Engineering*, 330:292 – 307, 2018.

[22] C. Livermore. *course materials for 6.777J / 2.372J Design and Fabrication of Microelectromechanical Devices*, 2007 (accessed July 24, 2018). MIT.

[23] K. Svanberg. The method of moving asymptotes - modelling apsects and solution schemes. Lecture Notes for the DCAMM Course: advanced Topics in Structural Optimization, June 1998.

[24] O. Sigmund. Morphology-based black and white filters for topology optimization. *Structural and Multidisciplinary Optimization*, 33(4):401–424, Apr 2007.

[25] Fengwen Wang, Boyan Stefanov Lazarov, and Ole Sigmund. On projection methods, convergence and robust formulations in topology optimization. *Structural and Multidisciplinary Optimization*, 43(6):767–784, Jun 2011.

[26] O. Sigmund J. Søndergaard Jensen F. Wang, B. S. Lazarov. Interpolation scheme for fictitious domain techniques and topology optimization of finite strain elastic problems. *Computer Methods in Applied Mechanics and Engineering*, 276:453 – 472, 2014.

[27] J. Iott, R. Haftka, and H.M.Adelman. Selecting step sizes in sensitivity analysis by finite differences. NASA, 1985.

[28] Pedersen Claus B. W., Buhl T., and Sigmund O. Topology synthesis of large-displacement compliant mechanisms. *International Journal for Numerical Methods in Engineering*, 50(12):2683–2705, 2001.

[29] Bruns T. E. and Tortorelli D. A. An element removal and reintroduction strategy for the topology optimization of structures and compliant mechanisms. *International Journal for Numerical Methods in Engineering*, 57(10):1413–1430, 2003.

[30] G. H. Yoon and Y.Y. Kim. Element connectivity parameterization for topology optimization of geometrically nonlinear structures. *International Journal of Solids and Structures*, 42(7):1983 – 2009, 2005.

[31] G.H. Yoon, Y.S. Joung, and Y.Y. Kim. Optimal layout design of three dimensional geometrically nonlinear structures using the element connectivity parameterization method. *International Journal for Numerical Methods in Engineering*, 69(6):1278–1304, 2 2007.

[32] G. H. Yoon, Y. Kim Yoon, M. Langelaar, and F. van Keulen. Theoretical aspects of the internal element connectivity parameterization approach for topology optimization. *International Journal for Numerical Methods in Engineering*, 76(6):775–797, 2008.

[33] J.S Jensen F. Wang, O. Sigmund. Design of material with prescribed nonlinear properties. *Journal of the Mechanics and Physics of Solid*, 69(2):156–174, 2014.