INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Structural Optimization For Flutter Instability Problems

Pedro André de Carvalho Pastilha

Dissertação para obtenção do Grau de Mestre em

Engenharia Aeroespacial

Júri

Presidente:	Professor Afzal Suleman (co-orientador)
Orientador:	Professor Miguel António Lopes de Matos Neves
Vogais:	Professora Alexandra Gonçalves Aguiar Gomes

Outubro 2007

We are at the very beginning of time for the human race. It is not unreasonable that we grapple with problems. But there are tens of thousands of years in the future. Our responsibility is to do what we can, learn what we can, improve the solutions, and pass them on.

Richard Feynman

Acknowledgements

I would like to thank my supervisor, Professor Miguel Matos Neves, for his supervision, support and thoroughness in every aspect relating the present work. I treasure his dedication and patience with the difficulties I faced during the development of this project.

I am thankful to my co-supervisor, Professor Afzal Suleman, for his motivating influence, critical views, suggestions and guidance lines throughout the development of this thesis.

I would like to leave a thank you note to Olavo Silva from UFSC (Brazil) for his help, support and accurate suggestions during the development of the optimization process and Filipe Mendonça for kindly sharing his knowledge and experience resulting from his own work.

I am grateful to Professor Krister Svanberg from the Royal Institute of Technology, Stockholm, for providing the $MATLAB^{\circledR}$ version of the MMA algorithm.

I could never express my full gratitude to my family, especially my parents, for the unconditional support through all my life, for always having faith in me and for the constant wise words of advice.

Last but not least, I would like to thank all my friends, directly or indirectly involved with the present thesis. Without you life would not be worth living!

This research was partially supported by FCT (Fundação para a Ciência e Tecnologia) through the project POCTI/EME/44728/2002 (MMN), FEDER and IDMEC-IST.

Abstract

The present work deals with the finite element analysis and optimization of structures for flutter instability originated by the action of non-conservative forces. Three models are considered for this purpose. The first case considers a cantilevered circular beam subjected to a partially non-conservative end-load, such as the thrust generated by a single propeller rocket. The second considers a thick plate subjected to the same type of end load as the circular beam. Finally the third model focuses on the problem of a flat panel subjected to the non-conservative load generated by a low supersonic flow.

The aim of the three presented models is first to present a comprehensive development and formulation of the governing equations for the structural behavior for each model, use numerical methods to implement the solution of these equations and the development of optimization processes with the goal of achieving minimal volume structures while maintaining structural stability within a large interval of excitation frequencies. As a first optimization method, optimal designs are obtained by varying the column's cross-sectional area. In a later stage, plate element formulation is used to obtain different optimized structures for equivalent loading conditions. Considering this plate element formulation, optimized structures are also obtained for the case of the plate subjected to a supersonic flow.

Results for the stability analysis and optimized results are presented for each model. These results are discussed with respect to the stability characteristics, analyzing the mechanisms that characterize flutter for each considered model, comparing also the results from the different models and, whenever possible, comparisons with bibliography results are presented. The optimal designs are also discussed with respect to volume reduction and the influence of the optimization process on the structural stability. Considerations are made with respect to the numerical methods used, limitations and advantages of the considered models and future developments for the presented work.

Keywords: Dynamic Stability, Flutter, Follower Loads, Panel Flutter, Structural Optimization, Beck's Column, Finite Element Analysis.

Resumo

No presente trabalho apresenta-se uma metodologia de análise e optimização de estruturas com respeito à instabilidade dinâmica do tipo *flutter*, originada pela acção de forcas com componente não conservativa. Três modelos são apresentados e discutidos: O primeiro considera o caso de uma coluna de secção circular encastrada numa extremidade e com um foguete propulsor na extremidade livre; O segundo caso refere-se ao mesmo exemplo, mas considerando uma placa no lugar da coluna; Finalmente, o terceiro modelo considera o fenómeno de *panel flutter* que ocorre em painéis sujeitos a escoamentos supersónicos baixos.

O objectivo destes três modelos é, numa primeira instância, o desenvolvimento e formulação das equações que descrevem o comportamento estrutural de cada modelo, posteriormente aplicar métodos numéricos para implementar algoritmos para resolver estas equações e, finalmente, o desenvolvimento de processos de optimização com o propósito de obter estruturas com volume mínimo garantindo os mesmos parâmetros de estabilidade das estruturas originais. Como um primeiro método, os resultados optimizados são obtidos variando o diâmetro das colunas consideradas. A seguir, introduz-se um modelo de placa, para obter diferentes resultados de optimização variando a espessura da placa, para condições equivalentes às do modelo de coluna. Usando a mesma formulação de placa, obtém-se resultados de optimização para um painel simplesmente apoiado sujeito a um escoamento supersónico incidente.

São apresentados resultados para a análise de estabilidade de cada modelo considerado, sendo discutidos em termos das características de estabilidade, analisando os mecanismos que caracterizam o *flutter* para cada modelo, comparando também os resultados dos diferentes modelos entre si e, sempre que possível, comparando os resultados obtidos com resultados existentes na bibliografia. Os resultados dos processos de optimização são também discutidos, considerando as reduções de volume obtidas e a influência do processo de optimização na estabilidade das estruturas finais. São ainda apresentadas considerações sobre os métodos numéricos usados, vantagens e limitações dos modelos considerados e sua implementação e sugestões de futuros desenvolvimentos.

Palavras-Chave: Estabilidade Dinâmica, Flutter, Forças Seguidoras, Panel Flutter, Optimização estrutural, Coluna de Beck, Elementos Finitos.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In general terms, flutter is a dynamic instability of a body subjected to non-conservative forces wherein its oscillations increase without bounds. Hence, this is a serious concern when dealing with the design and project of any structure subjected to these forces. In aeronautical terms, flutter is an aeroelastic self-excited vibration with a sustained or divergent amplitude, which occurs when a structure is placed in a flow of sufficiently high velocity.

The current work presents some formulation for analysis and optimization of simple structural models subjected to the mentioned non-conservative forces. The three discussed models are:

- Stability of Beams subjected to non-conservative follower loads.
- Stability of Plates subjected to non-conservative follower loads.
- Stability of Plates under supersonic flows.

The main goal in the analysis of these models is the understanding and characterization of the underlying mechanisms that originate flutter instability, as well as learning to differentiate it from other instability methods in both cause and effect. The behavior of structures subjected to non-conservative forces is frequently of difficult interpretation, since these forces require the use of complex functionals in the problem formulation and, therefore, the intuitive understanding of the physical results becomes difficult, mostly because the indirect effects may very well be the determining ones when obtaining the structural response. In spite of this, the present report intends to contribute for a clear and objective definition and classification of the instability modes to which the studied structures are subjected.

Having developed the analysis methods that allow the structural stability characterization for each model, the next goal defined for the current thesis is achieving optimized structural designs, while preserving the main stability characteristics. Due to the non-conservative nature of the load conditions, the optimization problems require the study of the sensitivities of the parameters directly involved in the optimal design (such as the critical instability loads, the excitation frequencies and the structural volume) with respect to the design variables (such as the diameter of a beam, or the thickness of a plate). Efficient sensitivity studies are required for the primary analysis algorithms, which leads to the necessity of effective computational procedures and algorithms for large eigenvalue problems. Therefore, the importance of the numerical solution methods is taken into account and the development of the solution algorithms is described with considerable detail.

Aeroelasticity can be defined as the science which studies the mutual interaction between aerodynamic, elastic and inertial forces, and the influence of this interaction in aircraft design (as defined by Bisplinghoff et al. (1996)). Flutter is one of the main concerns within the aeroelasticity domain and it has influenced the evolution of aircraft design since very early in the development of aircraft structures. Since these structures must necessarily be light, they can present considerable deformations under load. These deformations change the aerodynamic load distribution in the structure, which in turn changes the deformations and this cyclic process can lead to flutter.

Historically, the first know developments on aeroelasticity and flutter go back to the design of windmills in The Netherlands, where several structural issues regarding the blades of the windmills were solved empirically by moving the front spars of the blades from the approximate mid-chord to the quarter chord position (as investigated by Drees (1978)). Another historical milestone on the study of aeroelasticity and flutter instability is, oddly enough, related to Civil Engineering: The instability and collapse of the Tacoma Narrows bridge in 1940. The collapse of the bridge was originated by heaving and torsional vibration of the bridge deck, as can be seen in figure 1.1. The collapse of the bridge originated a series of new studies and considerations on the influence of flutter in Civil Engineering structures (a further discussion on the subject is presented by Miyata (2003)).

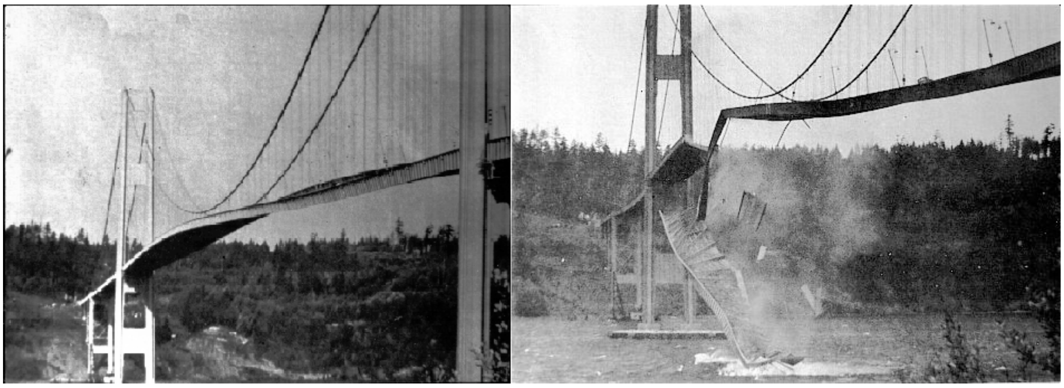


Figure 1.1: Tacoma Narrows bridge under flutter instability (*left*) which ultimately lead to structural collapse (*right*) (courtesy of Bashford and Thompson, 1940).

The first reports of flutter problems relating to aircraft structures go back to a few days before the historical flight of the Wright brothers when, professor Samuel P. Langley failed his second attempt to launch a powered flight machine, because of structural collapse, which was posteriorly considered to happen for flutter instability reasons.

The first documented flutter study was accomplished by F. W. Lanchester during World War I (Lanchester, 1916). This article studied the cause of the violent antisymmetric oscillations originated in the tail of a Handley Page 0/400 biplane bomber. The two major conclusions obtained in this work were that the oscillations did not result from the resonance induced by vibratory sources, but were self-excited and that the increase of the torsional stiffness of the elevators could eliminate the problem.

With the evolution of aircraft engineering and designs, the speeds increased and the developments in aeroelasticity and flutter instability became more and more extensive. Nowadays, it is possible to see large investments by both small and large companies in the aeronautics business, both in numerical analysis methods (as as can be seen in figure 1.2) and experimental wind-tunnel testing (as shown in figure 1.3). The same concern is reflected in the engineering research developed at universities.

A detailed description of aeroelasticity and flutter evolution throughout the 20^{th} century, with comprehensive reference data, can be followed through the work presented by Garrick and Reed (1981) and Dugundji (2003).

As for the present and future of aeroelasticity, Livne (2003) states that, although the main principles in the area are well established since the 1950s and 1960s, it is still a dynamic and challenging area. Continuous developments in computational fluid mechanics, numerical struc-

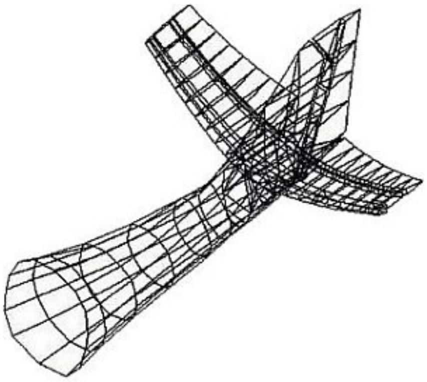


Figure 1.2: finite element model for the flutter stability analysis of the tail of the Lancair 360 airplane (courtesy of Aircraft Flutter, Inc., 2006).

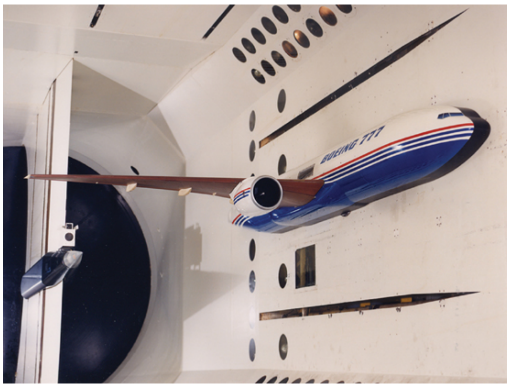


Figure 1.3: Boeing 777 flutter model for wind-tunnel testing (courtesy of NASA, 2003).

tural analysis methods and optimization processes allow for the development of new and more accurate developments in the area. In his work, Livne also presents a comprehensive bibliography relating to various aspects of modern aeroelasticity.

Although generally associated with aircraft design, flutter instability problems are also quite common in many other engineering applications. Any structure subjected to non-conservative loads that change with the displacements of the structure is prone to have flutter instability. Flutter instability problems can even be found in several other areas more or less related to engineering problems such as medicine, electronics or even music.

One of such problems is the flutter instability of spinning disks due to the aerodynamic pressure, as presented by Hansen et al. (1999). These problems include the study of disk saws and computer disk memory storage, since these devices tend to use thinner disks at higher rotation speeds. The main limitation in the design of these structures becomes the aerodynamically induced vibrations. Other related developments in the area are presented by Renshaw et al. (1994) and D'Angelo and Mote (1993).

Amongst other flutter instability problems are the instability of columns subjected to non-conservative end-loads, also designed as follower forces. As will be further discussed in the present work, these load conditions can be originated by rocket and jet engines, or the dry friction in automotive disk brake systems, as can be followed in the review by Kinkaid et al. (2003). A survey on different examples of beams subjected to follower loads is presented by Langthjem and Sugiyama (2000b) and a more detailed description for these problems is presented on the following chapter. Another structural model which is susceptible to flutter instability, closely related to the previous problem, is that of fluid conveying pipes. Païdoussis and Li (1993) present a survey on the problem.

Spacecraft dynamics is another engineering area in which flutter instability problems can occur. When large space structures need to change orbits or perform any position changes, they require thrusters. These thrusters are commonly connected to truss-like and plate-like structures which, due to the non-conservative nature of the forces originated by the thrusters, are quite susceptible to flutter instability (Kim and Park, 1998). The problem of plates subjected to these follower forces will also be more thoroughly discussed in the following chapter.

The current work sets upon the problem of obtaining optimized designs of columns subjected to non-conservative loads, following mainly as the work developed by Langthjem and Sugiyama (1999, 2000a,b). They developed numerical analysis methods and experimental verification, as shown in figure 1.4. Three basic models are presented. The first is the case of a cantilevered circular beam subjected to a partially non-conservative end-load. The second considers a thick plate subjected to the same type of end load as the circular beam. Finally the third model focuses on the problem of a flat panel subjected to the non-conservative load generated by a supersonic flow.

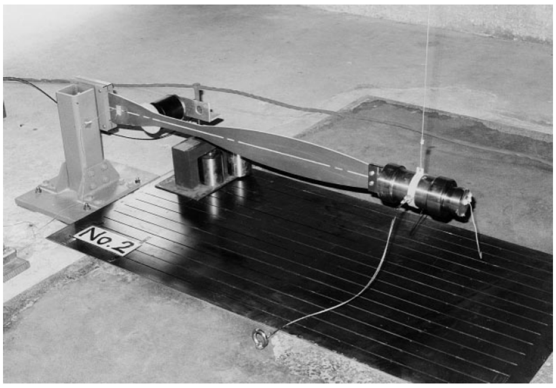


Figure 1.4: Experimental set for the verification of the work developed by Sugiyama et al. (2000).

The stability analysis for the presented models is developed using the finite element method, using the Euler-Bernoulli beam theory for the beam model and the Mindlin plate theory for the plate and panel models. These numerical solution schemes where selected based upon the characteristics of each model and the existing models in the bibliography. The optimization process was developed with a gradient based algorithm using the Method of the Moving Asymptotes (MMA) developed by Svanberg (1987). This optimization process was selected given that the complexity of the mechanisms that rule flutter instability problems render the conventional nonlinear optimization methods (such as $MATLAB^{®}$'s *fmincon* algorithm) inviable for the presented models.

The main purpose of the thesis is first to present a comprehensive development and formulation of the mentioned models, describe the considered numerical implementation for the analysis and optimization schemes and posteriorly discussing the obtained results. Another aim of the current work involves the development of a bridge between the initial considered model and other examples, with the purpose of developing a formulation that can be adapted and used in the future for a wider area of applications.

The present thesis is composed of four main sections: The first section deals with the theoretical formulation of the three considered models, presenting specific bibliographic reviews for each model. This includes the major bibliographical references, the context of the problem, the mathematical formulations, the considered assumptions, approximations and limitations for each model, and the dimensionless formulation of the model equations.

The second section focuses on the numerical methods used for the solution of the presented model equations. Finite element formulation is described including numerical integration considerations and the assembly of element equations. The solution schemes used for the eigenvalue problems are discussed and the computational implementation is also presented.

The third section presents the optimization problem to be considered and describes the iterative optimization process, including the sensitivity analysis formulation and computational implementation.

The fourth section presents the obtained results for the three models. The verification of the codes is presented and, whenever possible comparison with reference results is given and discussed. The presented results include the stability analysis for each model and the posterior optimization results.

Finally, the main conclusions and further developments regarding the obtained results, implementation methods and validity of the considered models are presented.

Chapter 2

Theoretical Models for Dynamic Stability Analysis

This chapter presents theoretical models which describe the behavior of structures subjected to both conservative and non-conservative loads. Three models are considered and their ruling equations, basic assumptions, principles, advantages and limitations are presented and discussed. The dimensionless formulation for each considered model is also developed in order to simplify the numerical implementation which will be focused on the next chapter. These models follow the work developed by other researchers and detailed formulations of the models are presented developing in greater detail some of the relevant aspects of the formulation.

2.1 Dynamic Stability of Elastic Columns Subjected to Non-Conservative Loads

2.1.1 Introduction

The first considered model deals with the behavior of cantilevered columns subjected to conservative compressive loads and tangential non-conservative follower forces. Columns are solid structures with one of the characteristic dimensions considerably larger than the other two.

The basic principle of a follower force can be widely observed in flexible missiles and spacecraft structures with end rocket thrusters. These structures are subjected to conservative forces, such as engine weight and non conservative forces which depend on the structural displacements such as the engine thrust or aerodynamic loads. A single propelled rocket structure can then be considered as a free-free beam subjected to a follower force which can be further simplified as a cantilevered beam subjected to a follower force at the free end. This simplification can be achieved by considering that the origin of a the fixed spatial coordinate system is located at the tip of the structure, as represented in figure 2.1 (discussed by Langthjem et al. (1999)).

The mentioned simplifications allow for simpler and faster analysis and optimization algorithms. In addition, the simplified structural response shows basically the same stability characteristics as a full model and the results have been confirmed experimentally (by Sugiyama et al. (2000)). A general discussion on the legitimacy of follower forces with respect to physical applications has been developed over the last years. This discussion can be found through many bibliographical references such as the articles published by Sugiyama et al. (2002); Langthjem et al. (1999) and more recently Elishakoff has published a comprehensive review on the subject (Elishakoff, 2005).

Although the present model does not take into account the influence of damping in the structural stability of the columns and ultimately the optimal designs, several studies are available

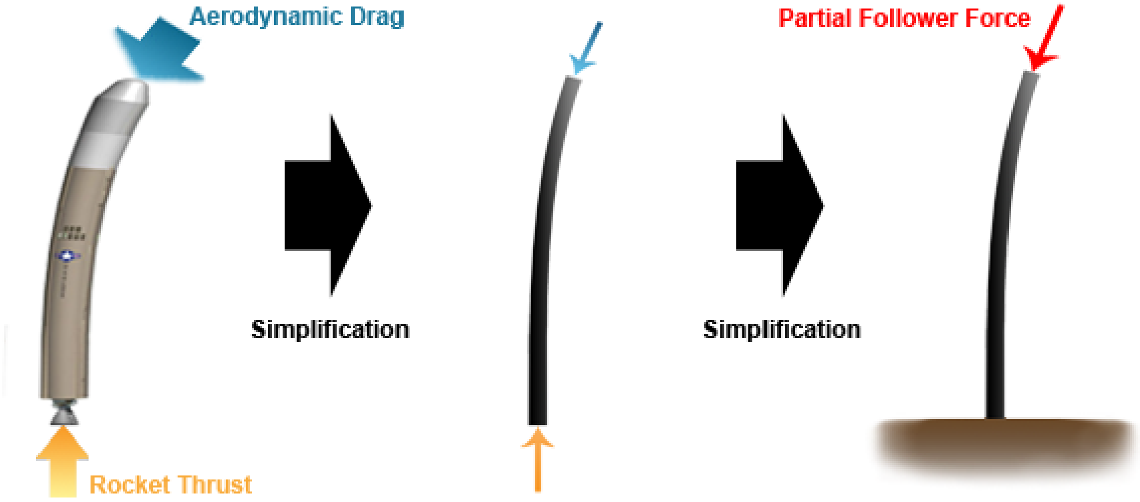


Figure 2.1: Assumed simplifications to establish the considered model.

in the area. Langthjem published a study on the influence of damping in problems of dynamic stability optimization (Langthjem, 1994). Another reference study is given by Langthjem and Sugiyama (2000c) where a follow up for the work presented by Langthjem and Sugiyama (2000a) is given, in which damping effects are taken into account.

On the influence of the torsional modes in the dynamic response of a beam the work from Butler and Banerjee (1996) can be followed. This work deals with the optimal design of beams with coupled bending-torsion loads and constraints in the frequency domain. A quite recent study on the subject is presented by Luo et al. (2006), where a multi objective programming scheme is presented for the conceptual design of an aerodynamic missile structure using topological optimization approaches in which the optimization objectives are set by the eigenfrequencies and critical loads.

The presented model follows the Euler-Bernoulli beam theory for an isotropic material. In this theory, it is assumed that plane cross-sections perpendicular to the axis of the beam remain plane and perpendicular to the axis after deformation (Reddy, 1992). From these basic assumptions, it is possible to verify that this theory produces accurate results only for small displacement problems and does not consider the shear deformation of the beam. Taking into account that the intended optimization process can originate large variations on the cross-sections of the beam along the length, this simplification may produce less accurate results. Furthermore the model does not consider rotatory inertia and, therefore no torsional modes are considered in the analysis. In spite of the shortcomings of the considered model, it stands as a simple, accurate and didactic analysis for the influence of non-conservative loads on the dynamic stability of a simple beam element and the corresponding optimal design.

2.1.2 Dynamic Stability Analysis

The mathematical model that describes small amplitude vibrations on a column can be obtained by the Euler-Bernoulli beam theory as follows:

$$\frac{\partial^2}{\partial x^2}\left(EI\frac{\partial^2 w}{\partial x^2}\right)+m\frac{\partial^2 w}{\partial t^2}+p\frac{\partial^2 w}{\partial x^2}+\frac{\partial^2}{\partial x^2}\left(E^*I\frac{\partial^3 w}{\partial x^2 \partial t}\right)+C\frac{\partial w}{\partial t}=0, \tag{2.1}$$

where E is the elasticity modulus, $I = I(x)$ is the area moment of inertia, E^* is the coefficient of dynamic visco-elastic resistance, C is the coefficient of external viscous damping, $m = m(x)$

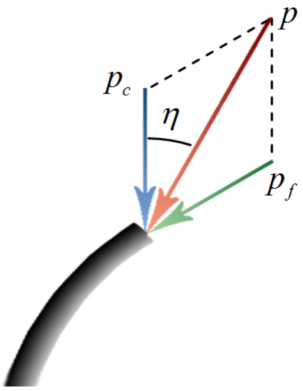


Figure 2.2: Graphical representation of the parameter η.

is the mass per unit length, p is the load applied at the free end of the column and $w = w(x,t)$ is the transverse displacement at the instant t and at the position x. For the present work, the effects of damping are neglected and as a consequence, equation 2.1 reduces to,

$$\frac{\partial^2}{\partial x^2}\left(EI\frac{\partial^2 w}{\partial x^2}\right) + m\frac{\partial^2 w}{\partial t^2} + p\frac{\partial^2 w}{\partial x^2} = 0. \tag{2.2}$$

In order to completely describe the problem, four boundary conditions are required. The presented model is cantilevered at $x = 0$ and free at $x = L$. At the free end, the column is subjected to a conservative compressive force of magnitude p_c and a non-conservative follower force of magnitude p_f as shown in figure 2.2. The resulting load applied to the column has a magnitude of $p = p_c + p_f$ and a direction given by the angle $\eta(\partial w/\partial x)_{x=L}$, where η gives the relation between the two load conditions according to the formula,

$$\eta = \frac{p_f}{p_c + p_f} = \frac{p_f}{p}. \tag{2.3}$$

From this definition, it is possible to note that $\eta = 0$ corresponds to a pure conservative load and therefore the Euler Buckling problem and $\eta = 1$ corresponds to a pure follower load and thus Beck's flutter problem. With these definitions in mind, the four boundary conditions can be given by,

$$\begin{gathered} w(x=0,t) = 0, \quad \frac{\partial w(x=0,t)}{\partial x} = 0, \quad EI\frac{\partial^2 w(x=L,t)}{\partial x^2} = 0, \\ \frac{\partial}{\partial x}\left(EI\frac{\partial^2 w(x=L,t)}{\partial x^2}\right) + p(1-\eta)\frac{\partial w(x=L,t)}{\partial x} = 0. \end{gathered} \tag{2.4}$$

Having defined the general equations which describe the behavior of the column, it is now possible to formulate the corresponding boundary value problem by considering that the displacements are given in the form,

$$w(x,t) = \tilde{w}(x)\exp(\lambda t), \quad \lambda = \alpha + i\omega \tag{2.5}$$

and scaling the problem variables into a dimensionless form as

$$\begin{gathered} \bar{x} = \frac{x}{L}, \quad \bar{w} = \frac{\tilde{w}}{L}, \quad \bar{p} = \frac{pL^2}{EI_0}, \quad \bar{m}(\bar{x}) = \frac{m(x)}{m_0}, \\ \bar{I}(\bar{x}) = \frac{I(x)}{I_0}, \quad \bar{t} = \frac{t}{L^2}\sqrt{\frac{EI_0}{m_0}}, \quad \bar{\lambda} = \frac{t}{\bar{t}}\lambda, \end{gathered} \tag{2.6}$$

where the index '0' corresponds to a uniform column. Replacing these dimensionless quantities in equations 2.2 and 2.4, we have,

$$
\begin{gathered}
\frac{\partial^2}{\partial \bar{x}^2}\left(\bar{I}\frac{\partial^2}{\partial \bar{x}^2}\left(\bar{w}\exp\left(\bar{\lambda}\bar{t}\right)\right)\right)+\bar{m}\frac{\partial^2}{\partial \bar{t}^2}\left(\bar{w}\exp(\bar{\lambda}\bar{t})\right)+\bar{p}\frac{\partial^2}{\partial \bar{x}^2}\left(\bar{w}\exp\left(\bar{\lambda}\bar{t}\right)\right)=0,\\
\bar{w}(\bar{x}=0,\bar{t})=0, \quad \frac{\partial \bar{w}(\bar{x}=0,\bar{t})}{\partial \bar{x}}=0, \quad \bar{I}\frac{\partial^2 \bar{w}(\bar{x}=1,\bar{t})}{\partial \bar{x}^2}=0,\\
\frac{\partial}{\partial \bar{x}}\left(\bar{I}\frac{\partial^2 \bar{w}(\bar{x}=1,\bar{t})}{\partial \bar{x}^2}\right)+\bar{p}(1-\eta)\frac{\partial \bar{w}(\bar{x}=1,t)}{\partial \bar{x}}=0.
\end{gathered}
\tag{2.7}
$$

The dimensionless boundary value problem can then be obtained by deriving the previous equation with respect to the dimensionless time resulting in,

$$
\begin{gathered}
\Lambda(\bar{w})=(\bar{I}\bar{w}'')''+\lambda^2\bar{m}\bar{w}+\bar{p}\bar{w}''=0,\\
\bar{w}(0)=0, \quad \bar{w}'(0)=0,\\
\bar{I}\bar{w}''(1)=0, \quad (\bar{I}\bar{w}'')'(1)+\bar{p}(1-\eta)\bar{w}'(1)=0,
\end{gathered}
\tag{2.8}
$$

in which Λ is the differential operator, and the prime is used to denote differentiation with respect to the dimensionless coordinate $\bar{x}$. As can be verified from equation 2.8, the presented dimensionless formulation does not depend on material properties. Considering that the column has circular cross sections with radius $r(x)$ and constant density it follows that,

$$
\bar{I}(\bar{x})=\frac{I(x)}{I_0}=\frac{\pi r(x)^4/4}{\pi r_0^4/4}=\left(\frac{r^2}{r_0^2}\right)^2=\left(\frac{\pi r^2\rho}{\pi r_0^2\rho}\right)^2=\left(\frac{m(x)}{m_0}\right)^2=\left(\bar{m}(\bar{x})\right)^2. \tag{2.9}
$$

This property will prove to be quite useful for the optimization formulation described in chapter 3. Having defined the dimensionless problem at this point, the over bars will no longer be used for notation simplicity.

Combining the equations presented in 2.8 and integrating the resulting equation in the dimensionless length of the column, the following functional can be obtained,

$$
L(w)=\int_0^1\left[Iw''+\lambda^2 w+pw'\right]dx+\eta p w'(1) \tag{2.10}
$$

which is stationary with respect to variations δw, thus satisfying the kinematic boundary conditions $w(0)=0$, $w'(0)=0$. This functional stands as the basis for any numerical discretization method, originating a discretized equation of motion in the form,

$$
\mathbf{Lw}=\left[\mathbf{K}+\lambda^2\mathbf{M}+p\left(\mathbf{G_c}+\eta\mathbf{G_f}\right)\right]\mathbf{w}=0, \tag{2.11}
$$

where $\mathbf{K}$ is the stiffness matrix, $\mathbf{M}$ is the mass matrix, $\mathbf{G_c}$ and $\mathbf{G_f}$ are the load matrices corresponding to the conservative and non-conservative load components, respectively. The discretization method used for the determination of these matrices is described in chapter 2.

It should be noted that from equation 4.23 it is also possible to determine the structural natural frequencies (by setting $p=0$), and the Euler buckling loads (by setting $\lambda=0$ and $\eta=0$). This allows for a basic validation of the developed algorithms, by comparison with commercial finite element software such as $ANSYS^{®}$ as will be shown.

2.2 Dynamic Stability of Plates Subjected to Non-Conservative Loads

2.2.1 Introduction

This section focuses on the dynamic equations that describe the behavior of a plate subjected to non-conservative forces. On a first case, the analysis of a cantilevered plate subjected to the combined action of distributed conservative and non-conservative loads at the free end, as represented in figure 2.3, is presented and discussed.

The purpose of the considered model is to reproduce the previously presented beam model results and to allow for the development of thickness optimization procedures which can allow for improved optimal results. This is an original approach to the optimization problem of a structure subjected to a non-conservative load and no published studies have been found on the subject.

In spite of the lack of published studies on optimization methods for plates subjected to non conservative loads, several studies concerning the dynamic stability analysis for these problems can be found. The work developed by Zuo and Schreyer (1996) studies the flutter and divergence instability of beams and plates, establishing a parallelism between the two models and compares the stability characteristics for both models and the nature of the non-conservative forces for both cases. Kim and Park (1998) studied the dynamic stability of rectangular plates subjected to intermediate follower forces. According to their work these forces can occur in space structures under the action of control thrusters as previously mentioned. Kim and Kim (2000) present a study on the dynamic stability of plates under a follower force. In their study, several finite element models are presented and the influence of the selected finite elements and structural plate formulation on the stability solution is verified.

The lateral-torsional flutter instability modes is also studies for the plate model presented by Hodges (2001). In this article, the behavior of a cantilever plate subjected to a lateral follower force at the tip is investigated. The presented model can be considered as a simplified study of a high aspect ratio wing with a loading originated by a jet engine. Potapov (2004) developed a study on the stability of elastic and viscoelastic plates in gas flows. The developed model takes into account the shear strains for the structure. The plate is subjected to a load in the plane of the plate (a follower force), originated by a supersonic flow. The stability solutions are obtained using semi-analytical methods. Recently Jayaraman and Struthers (2005) presented a work on divergence and flutter instability of elastic specially orthotropic plates subjected to follower forces. The presented work studies the influence of the non-conservative load component, the plate's aspect ratio, the boundary conditions and the material orthotropy on the stability conditions of the plate, specifically on the instability mode and the critical loads. Another study is presented by Kumar et al. (2005), where the dynamic instability of doubly curved composite panels subjected to partially distributed follower forces is analyzed using the first order shear deformation theory.

As mentioned, the beam model was developed from the Euler-Bernoulli beam theory which does not consider shear deformations and rotary inertia. For the plate model, the first order shear deformable Mindlin theory is used to determine the structural behavior of the plate. This theory allows for the study of both thick plates and the rapid thickness variations that can result from the optimization process will now have to take into account the shear stresses through the plate. This model can also consider torsional loading, but for the verification of the previous beam model the torsional vibration modes should not influence significantly the basic response of the structure.

The second considered case deals with the response of flat plates subjected to the non-conservative surface load generated by a supersonic flow. The instability that results from this

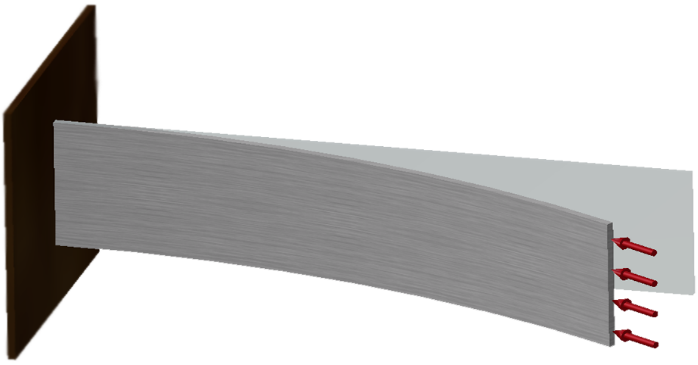


Figure 2.3: Deformed and undeformed shape of a cantilevered plate with a distributed end load.

load condition is known as panel flutter and it is nowadays a large concern when designing high speed aircraft. An early review on the aeroelastic stability of plates and shells is given by Dowell (1970). Van Keuren and Eastep (1977), presented an optimization method for the design of minimum weight square panels with dynamic constraints, using Galerkin's method for the numerical stability analysis. The study of panel flutter for composite panels is presented and discussed by Suleman and Venkayya (1994, 1996). An optimization study considering the application of piezoelectric actuators is presented by Suleman and Gonçalves (1997).

A study of the effects of hysteretic and aerodynamic damping on supersonic panel flutter has recently been developed by Koo and Hwang (2004). In this study it is possible to verify that structural damping has an important role when the aerodynamic damping is low, but does not affect the stability of the panel when aerodynamic damping effects are more significative. Odaka and Furuya (2005) presented structural optimization methods for a plate wing under a supersonic flow. This study shows the importance of the frequency separation constraints for the convergence of the optimization process and the robustness of the optimal designs.

For the present work, the structural response of the plate is determined by the shear deformable Mindlin plate theory (analogously to the plate model subjected to a non-conservative end-load) and the aerodynamic loading is calculated through the first order Piston theory (as presented by Odaka and Furuya (2005)). This theory has proven to generate good approximations for the dynamic pressure for low supersonic regimes.

2.2.2 Dynamic Stability Analysis of a Cantilevered Plate Subjected to an End Load

In order to describe the deformations of a plate the First Order Shear Deformation Theory, also known as the Mindlin plate theory, is considered. This theory is set upon a displacement field given by (Reddy, 1992),

$$u_1 = u + z\varphi_x, \quad u_2 = v + z\varphi_y, \quad u_3 = w, \tag{2.12}$$

where (u, v, w) are the displacements of a point with coordinates $(x, y, 0)$ and φ_x and φ_y are the rotations of the transverse normal about the y and $-x$ axes, respectively. For a linear theory based on infinitesimal strains and orthotropic materials it is possible to demonstrate that the in-plane displacements (u, v) are uncoupled from the transverse deflection. The in-plane displacements are governed by plane elasticity equations and are not considered in the present work therefore only the equations governing the bending deflections $(w, \varphi_x, \varphi_y)$ are developed. For the considered model, represented in figure 2.4, a distributed load P is applied to the side of the plate opposite do the cantilevered face. This load has a conservative in-plane component N_c and a tangential component P_f, where the relative magnitude of each force is determined once again by the parameter η which is defined analogously to the the beam model as

$$\eta = \frac{P_f}{N_c + P_f} = \frac{P_f}{P}. \tag{2.13}$$

Following the assumptions stated by Mindlin, the strain energy for an isotropic shear deformable plate is given by,

$$\begin{aligned} U \;&=\; \frac{1}{2}\int\int D\left[\left(\frac{\partial\varphi_x}{\partial x}\right)^2 + 2\nu\frac{\partial\varphi_x}{\partial x}\frac{\partial\varphi_y}{\partial y} + \left(\frac{\partial\varphi_y}{\partial y}\right)^2 + \frac{1-\nu}{2}\left(\frac{\partial\varphi_x}{\partial y} + \frac{\partial\varphi_y}{\partial x}\right)^2\right] + \\ &\quad + A\left[\left(\varphi_x + \frac{\partial w}{\partial x}\right)^2 + \left(\varphi_y + \frac{\partial w}{\partial y}\right)^2\right] dxdy, \end{aligned} \tag{2.14}$$

where D is the bending stiffness of the isotropic plate and A is the extensional stiffness, determined from

$$D = \frac{Eh^3}{12\left(1-\nu^2\right)}, \quad A = \left(\frac{5}{6}\right) Gh, \tag{2.15}$$

where G is the shear modulus and ν is the Poisson's ratio. Likewise, the kinetic energy of the plate is,

$$T = \frac{1}{2}\int\int I_m\left[\left(\frac{\partial\varphi_x}{\partial t}\right)^2 + \left(\frac{\partial\varphi_x}{\partial t}\right)^2\right] + m\left(\frac{\partial w}{\partial t}\right) dxdy, \tag{2.16}$$

where I_m corresponds to the mass moment of inertia and m is the mass per unit area of the plate. The potential energy corresponding to the conservative component of P is given by

$$V = \frac{1}{2}\int\int P\left(\varphi_x\right)^2 dxdy, \tag{2.17}$$

and the virtual work done by the nonconservative load can be obtained from,

$$\delta W_f = \int\int \eta P\frac{\partial w}{\partial x}\bar{\delta}(x-a)\delta w dxdy, \tag{2.18}$$

where $\bar{\delta}$ is the Dirac delta function. The equations of motion can now be derived from the previous equations using the generalized form of Hamilton's principle

$$\delta\int_{\bar{t}_1}^{\bar{t}_2}\left[T - U + V + W_f\right] d\bar{t}. \tag{2.19}$$

Assuming harmonic motion, the global displacement vector can be written as,

$$\mathbf{u} = \tilde{u}\exp(\lambda t), \quad \lambda = \alpha + i\omega, \tag{2.20}$$

Introducing the following dimensionless variables

$$\begin{aligned} &\bar{x} = \frac{x}{L_x}, \quad \bar{y} = \frac{y}{L_x}, \quad \bar{w} = \frac{\tilde{w}}{L_x}, \quad \bar{P} = \frac{PL_x^2}{D_0}, \quad \bar{A} = \frac{Ah_0}{D}, \\ &\bar{m}(\bar{x},\bar{y}) = \frac{m(x,y)}{m_0}, \quad \bar{I}_m(\bar{x},\bar{y}) = \frac{I_m(x,y)}{m_0 h_0^2}, \quad \bar{t} = \frac{t}{L^2}\sqrt{\frac{D_0}{m_0}}, \quad \bar{\lambda} = \frac{t}{\bar{t}}\lambda, \end{aligned} \tag{2.21}$$

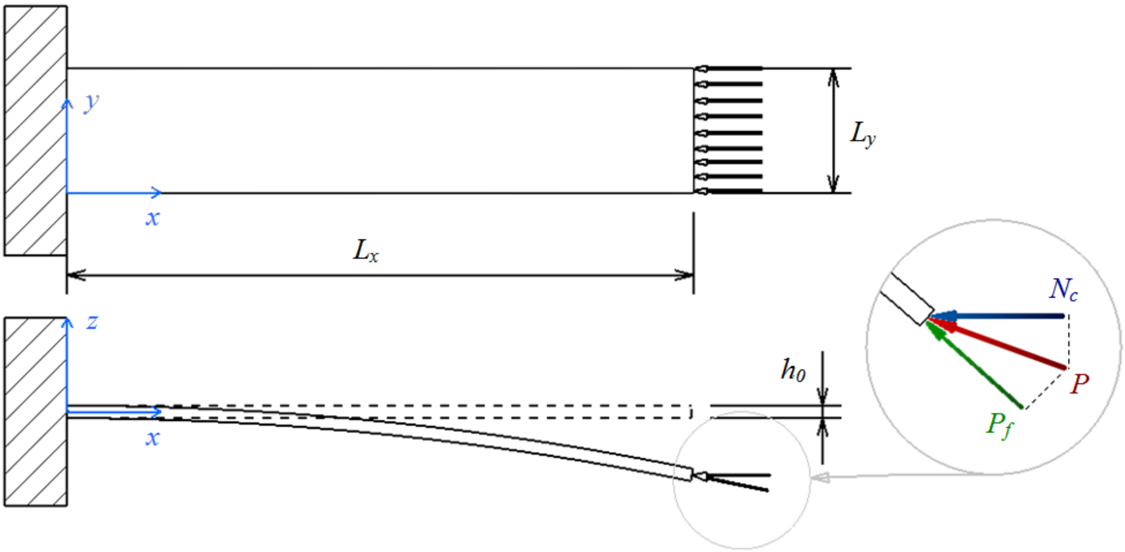


Figure 2.4: Geometric dimensions and load properties for the presented model.

where the index '0' corresponds to a plate with uniform thickness h_0. As for the beam model, the overbars denoting the dimensionless quantities will no longer be used for notation simplicity. By replacing these dimensionless quantities and the displacement vector defined by 2.20 into equation 2.19 the following functional is obtained,

$$\mathbf{L}\mathbf{u} = \left[\mathbf{K} - \lambda^2\mathbf{M} - P\left(\mathbf{G_c} - \eta\mathbf{G_f}\right)\right]\mathbf{u} = 0, \tag{2.22}$$

where $\mathbf{K}$, $\mathbf{M}$, $\mathbf{G_c}$ and $\mathbf{G_f}$ are the discretized numerical matrices as described in the previous section. It is possible to note that the discretized functional is identical to the one obtained from the beam formulation.

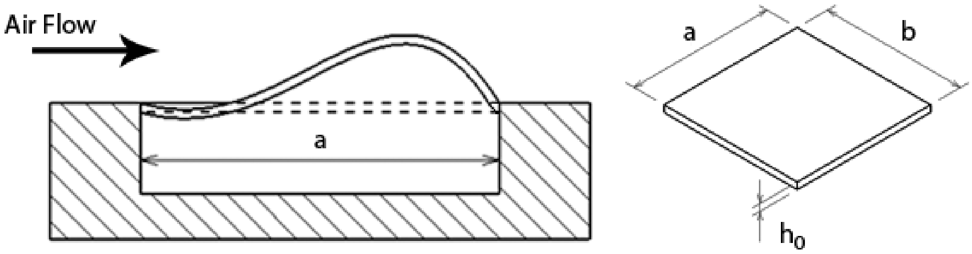


Figure 2.5: Simply supported plate under a Supersonic Flow.

2.2.3 Dynamic Stability Analysis of a Plate Under Supersonic Flow

When a thin plate or shell-like component is subjected to critical loading conditions generated by high speed passing airflow, self-excited dynamic instability may occur. This instability is known as Panel Flutter and is a main concern when designing high speed aircraft and missile surface skins. This dynamic instability normally occurs at supersonic speeds ($M > 1$) and depends upon various factors from the airflow Mach number and incidence angle, panel geometry and boundary conditions, in-plane stresses amongst other characteristics. For the present analysis model, a simply supported square plate is considered with length a, width b and initial thickness h_0. The upper surface of the plate is subjected to a supersonic flow at a speed of U_∞, density ρ_∞ and at a Mach M_∞. The lower surface of the panel contains air at rest. The aerodynamic load applied to the structure can be determined through the first order piston theory.

Piston theory is an inviscid unsteady aerodynamic theory that is used quite often in supersonic and hypersonic aeroelasticity. With it, it is possible to determine a relationship between the local pressure on any point of the considered surface and the component of fluid velocity normal to the moving surface. The derivation of this expression utilizes the isentropic simple wave expression for the pressure on the surface of a moving piston, resulting in the following pressure difference distribution (Suleman and Venkayya, 1994),

$$\Delta p = \frac{\rho_\infty U_\infty^2}{\sqrt{M_\infty^2 - 1}} \left(\frac{\partial w}{\partial x} + \frac{M_\infty^2 - 2}{M_\infty^2 - 1} \frac{1}{U_\infty} \frac{\partial w}{\partial t} \right). \tag{2.23}$$

For the present model the terms corresponding to aerodynamic damping are neglected and equation 2.24 is reduced to (as presented by Odaka and Furuya (2005)),

$$\Delta p = \frac{\rho_\infty U_\infty^2}{\sqrt{M_\infty^2 - 1}} \left(\frac{\partial w}{\partial x} \right), \tag{2.24}$$

where a zero flow deflection is assumed for the panel's leading and trailing edges. The work done by the surface pressure resulting from the aerodynamic flow can be obtained from the principle of virtual work as,

$$\delta W_{pf} = \int\int \frac{\rho_\infty U_\infty^2}{\sqrt{M_\infty^2 - 1}} \left(\frac{\partial w}{\partial x} \right) \delta w dx dy. \tag{2.25}$$

By replacing the term corresponding to the virtual work developed by the non-conservative end-load in equation 2.19 with the work given by equation 2.25 and following the same procedure as before and the resulting numerical discretization will assume the form,

$$\mathbf{Lu} = \left[\mathbf{K} - \lambda^2 \mathbf{M} - N\mathbf{G} - Q\mathbf{A} \right] \mathbf{u} = 0, \tag{2.26}$$

where **K**, **M** are the same as previously defined, **G** is the loading matrix due to the in-plane force N, **A** is the aerodynamic loading matrix and Q is a dimensionless parameter related to the dynamic pressure applied to the plate according to,

$$\bar{Q} = \frac{\rho_\infty U_\infty^2}{\sqrt{M_\infty^2 - 1}} \frac{a^3}{D}. \tag{2.27}$$

As can be verified from the presented formulation the problem originated by submitting a flat panel to a supersonic flow presents an almost identical discretized form as the previously presented models of structural elements subjected to non-conservative end loads. This fact comes to show, that the underlying mechanisms that originate flutter instability are described by the same basic mathematical principles.

Chapter 3

Numerical Methods for Dynamic Stability Analysis

This chapter describes the numerical methods adopted in this work for the discretization of the theoretical equations presented in Chapter 2. The Finite Element Method is selected for the numerical formulation of these equations taking into account the requirements for the optimization process. Two models are presented for the beam and plate formulations, including the selection of shape functions, element matrix construction, connectivity relations and global matrix construction. Numerical integration methods and reduced integration are also focused for the formulation of the plate finite element model.

3.1 Beam Element Formulation

3.1.1 Discretization of the domain

In order to obtain a numerical discretization for the problem of a column with length L subjected to an end-load p, as described by equation 2.11, the domain of the structure is divided into a set of N_e line elements, each element having two end nodes, as shown in figure 3.1. The variational form of these equations requires that the interpolation functions of any element are continuous with nonzero derivatives up to the second order (Reddy, 1992). Taking into account that there is a total of four boundary conditions, a third order polynomial must be used for the representation of w as,

$$w(x) = c_1 + c_2 x + c_3 x^2 + c_4 x^3. \tag{3.1}$$

This equation automatically meets the continuity conditions by ensuring the existence of a nonzero second derivative of w in the element. The determination of the constants is obtained from the boundary conditions at each node, which can be given in terms of the primary nodal variables as,

$$u_1^e = w(x_e), \quad u_2^e = \left(\frac{\partial w}{\partial x}\right)_{x=x_e}, \quad u_3^e = w(x_{e+1}), \quad u_4^e = \left(\frac{\partial w}{\partial x}\right)_{x=x_{e+1}}. \tag{3.2}$$

From these two equations it is possible to express the constants c_1, c_2, c_3 and c_4 as functions of u_1^e, u_2^e, u_3^e and u_4^e resulting in,

$$w^e(x) = u_1^e\phi_1^e + u_2^e\phi_2^e + u_3^e\phi_3^e + u_4^e\phi_4^e = \sum_{i=1}^{4} u_i^e\phi_i^e, \tag{3.3}$$

where ϕ_i^e are cubic interpolation functions derived by interpolating w and its derivative at both element nodes. These functions are known as the Hermite cubic interpolation functions and can

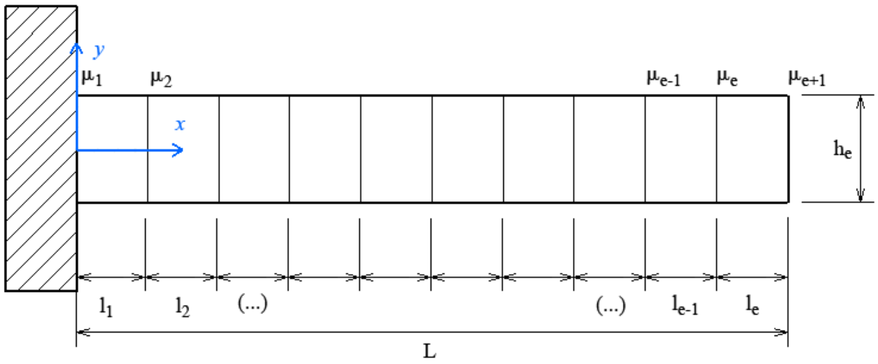


Figure 3.1: Numerical Discretization of the cantilevered column.

be expressed in terms of the element's local coordinate ξ as,

$$\phi_1^e = 1 - 3\left(\xi\right)^2 + 2\left(\xi\right)^3, \qquad \phi_2^e = -\xi\left(1-\xi\right)^2 l_e,$$
$$\phi_3^e = 3\left(\xi\right)^2 - 2\left(\xi\right)^3, \qquad \phi_4^e = -\xi\left[\left(\xi\right)^2 - \xi\right] l_e. \tag{3.4}$$

The finite element matrices from equation 2.11 can now be obtained by replacing w by the finite element interpolation 3.3. Considering that each element has a linear diameter evolution according to (Langthjem and Sugiyama (2000a)),

$$h_e = \left(1-\xi\right)\mu_e + \xi\mu_{e+1} \tag{3.5}$$

where μ_e is the equivalent diameter at node e. From these definitions it is possible to calculate the dimensionless stiffness matrix for a generic element e from,

$$K_{ij}^e = \int_{\xi_e}^{\xi_{e+1}} h_e^2 \frac{d^2\phi_i^e}{d\xi^2}\frac{d^2\phi_j^e}{d\xi^2} d\xi, \tag{3.6}$$

where $\xi_e = x_e/l_e$ and $\xi_{e+1} = x_{e+1}/l_e$. The dimensionless mass matrix is obtained following the same procedure,

$$M_{ij}^e = \int_{\xi_e}^{\xi_{e+1}} h_e \phi_i^e \phi_j^e d\xi. \tag{3.7}$$

The dimensionless conservative load matrix is obtained through,

$$G_{c_{ij}}^e = \int_{\xi_e}^{\xi_{e+1}} \frac{d\phi_i^e}{d\xi}\frac{d\phi_j^e}{d\xi} d\xi. \tag{3.8}$$

These matrices are all symmetric and positive definite. The dimensionless non-conservative load matrix is non-symmetric and is given by,

$$G_{f_{ij}}^e = \begin{cases} L, & \text{if } e = N_e,\ i = 3 \text{ and } j = 4 \\ 0, & \text{otherwise} \end{cases} \tag{3.9}$$

This definition is for a global matrix, defined for the complete discretized structure. The remaining equations define the numerical discretization of the matrices for a generic element e. In the next section the assembly of these matrices is presented in order to obtain the global matrices of the complete structure.

3.1.2 Assembly of Element Equations and Boundary Conditions

In the previous section the element equations are derived for a generic element which can be located anywhere in the discretized domain. In order to solve the proposed finite element analysis problem, the matrices for each element must be calculated for their corresponding position and connected with the adjacent elements, while ensuring the interelement continuity of the primary variables and (deflection and slope) and continuity of the secondary variables (shear force and bending moment) at the nodes which are common between the elements.

Consider an element e_1 calculated between global nodes n_1 and n_2 and an adjacent element e_2 calculated between global nodes n_2 and n_3. The connectivity matrix for these two elements is,

$$\mathbf{B} = \begin{bmatrix} n_1 & n_2 \\ n_2 & n_3 \end{bmatrix} \tag{3.10}$$

This matrix is used for the computational implementation process. Since there are two primary degrees of freedom per node, each node n corresponds to two degrees of freedom in the finite element matrices. The repeated nodes in matrix 3.11 indicate that the corresponding degrees of freedom in each element will add up. For a beam discretized with ten elements the connectivity matrix would take the form,

$$\mathbf{B} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \end{bmatrix} \tag{3.11}$$

At this point, the geometric boundary conditions for the specific analysis problem must be specified. For the present problem the column is clamped at $x = 0$ and the corresponding boundary conditions defined by 2.8 are imposed by considering that the discretized degrees of freedom at the first node (corresponding to element 1, which is located between $x = 0$ and $x = l_1$) are considered to be zero,

$$u_1^1 = w(x=0) = 0, \quad u_2^1 = \left(\frac{dw}{dx}\right)_{x=0} = 0. \tag{3.12}$$

These boundary conditions are implemented numerically by removing from the analysis algorithm the lines and columns associated with the presented degree's of freedom from the element matrices. This completes the definition of the finite element formulation for the beam model.

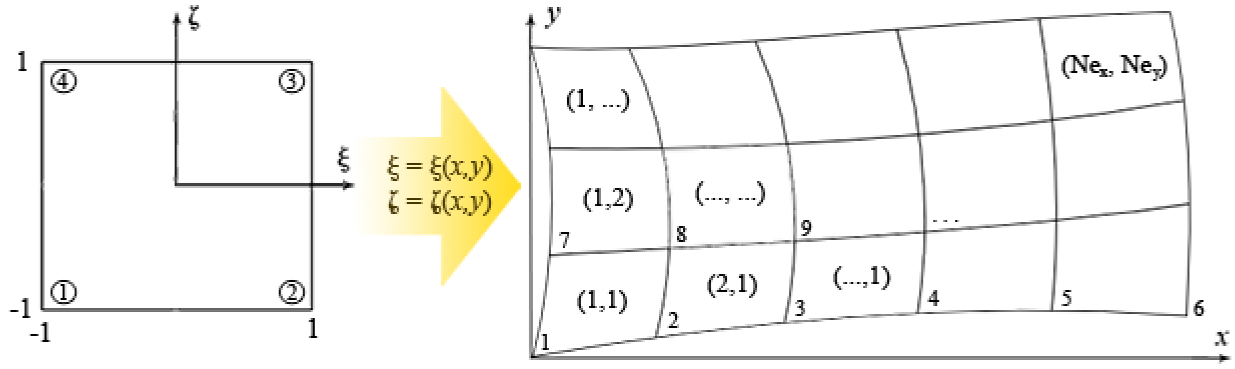


Figure 3.2: Generation of the plate mesh from a global element.

3.2 Plate Element Formulation

3.2.1 Discretization of the domain

The numerical formulation for the analysis of plates subjected to non-conservative loads presented theoretically by equations 2.22 and 2.26 will now be described. From the presented energy analysis of the plate it is possible to note that the primary variables for the Mindlin plate theory are the displacement,w and the rotations in the x and y directions, φ_x and φ_y. The secondary variables are the normal and twisting moments and the transverse stress. By inspecting the theoretical formulation it becomes clear that the primary variables are only differentiated once with respect to x and y. Therefore, it is possible to assume the finite element interpolation of w, φ_x and φ_y as,

$$w = \sum_{i=1}^{n} w_i \psi_i, \quad \varphi_x = \sum_{i=1}^{n} s_i^x \psi_i, \quad \varphi_y = \sum_{i=1}^{n} s_i^y, \psi_i, \tag{3.13}$$

where ψ are Lagrange interpolation functions. For the present analysis a four node linear element is considered and, consequently, the Lagrange interpolation functions are linear and given by,

$$\psi_i = \frac{1}{4}(1 + \xi\xi_i)(1 + \zeta\zeta_i), \qquad i = 1, \ldots, 4, \tag{3.14}$$

where (ξ_i, ζ_i) are the coordinates for the node i of a rectangular master element as shown in fig. 3.2. By choosing interpolation functions with the same degree for both the displacement and rotations, the use of reduced integration will be required for the evaluation of the stiffness coefficients associated with the transverse shear strains in order to avoid shear locking problems.

As for the beam model discretization, a linear thickness evolution is considered for each element according to,

$$h_e = \sum_{i=1}^{4} \frac{1}{4}(1 + \xi\xi_i)(1 + \zeta\zeta_i)\mu_i, \tag{3.15}$$

where μ_i is the equivalent thickness at node i. taking into account equations 3.13 and 3.15 into equations 2.22 and 2.26 it is possible to calculate the stiffness matrix as, in global coordinates,

$$\mathbf{K} = \begin{bmatrix} [K^{11}] & [K^{12}] & [K^{13}] \\ & [K^{22}] & [K^{23}] \\ sym. & & [K^{33}] \end{bmatrix}, \tag{3.16}$$

where,

$$
\begin{aligned}
K_{ij}^{11} &= \int\int h_e r\left(\frac{d\psi_i}{dx}\frac{d\psi_j}{dx}+\frac{d\psi_i}{dy}\frac{d\psi_j}{dy}\right)dxdy,\\
K_{ij}^{12} &= \int\int h_e r\left(\frac{d\psi_i}{dx}\psi_j\right)dxdy,\\
K_{ij}^{13} &= \int\int h_e r\left(\frac{d\psi_i}{dy}\psi_j\right)dxdy,\\
K_{ij}^{22} &= \int\int h_e^3\left(\frac{d\psi_i}{dx}\frac{d\psi_j}{dx}+\frac{1-\nu}{2}\frac{d\psi_i}{dy}\frac{d\psi_j}{dy}\right)+h_e r\left(\psi_i\psi_j\right)dxdy,\\
K_{ij}^{23} &= \int\int h_e^3\left(\nu\frac{d\psi_i}{dx}\frac{d\psi_j}{dy}+\frac{1-\nu}{2}\frac{d\psi_i}{dy}\frac{d\psi_j}{dx}\right)dxdy, \qquad (3.17)\\
K_{ij}^{22} &= \int\int h_e^3\left(\frac{1-\nu}{2}\frac{d\psi_i}{dx}\frac{d\psi_j}{dx}+\frac{d\psi_i}{dy}\frac{d\psi_j}{dy}\right)+h_e r\left(\psi_i\psi_j\right)dxdy,
\end{aligned}
$$

here, $r = 6k(1-\nu)h_0$ and $k = 5/6$ which denotes the shear correction coefficient, introduced to take into account the discrepancy between the constant state of shear stress considered in the Mindlin theory and the parabolic variation of the actual shear stress along the thickness of the plate. Another approach to avoid the use of this coefficient would be the use of higher order finite elements, as discussed by Moita et al. (2005). Following the same principles, the mass matrix can be determined from,

$$
\mathbf{M}=\begin{bmatrix} [M^{11}] & [0] & [0] \\ & [M^{22}] & [0] \\ sym. & & [M^{33}] \end{bmatrix}, \qquad (3.18)
$$

with,

$$
\begin{aligned}
M_{ij}^{11} &= \int\int h_e\psi_i\psi_j dxdy,\\
M_{ij}^{22}=M_{ij}^{33} &= \frac{1}{12}\left(\frac{1}{h_0}\right)^2\int\int h_e^3\psi_i\psi_j dxdy, \qquad (3.19)
\end{aligned}
$$

The conservative loading matrix, $\mathbf{G_c}$, defined for equations 2.22 is the same as the loading matrix due to in-plane stresses $\mathbf{G}$ for the panel flutter problem (equation 2.26) and is calculated through,

$$
\mathbf{G}=\mathbf{G_c}=\begin{bmatrix} [G^{11}] & [0] & [0] \\ & [0] & [0] \\ sym. & & [0] \end{bmatrix}, \qquad (3.20)
$$

with,

$$
G_{ij}^{11}=\int\int\frac{d\psi_i}{dx}\frac{d\psi_j}{dx}dxdy, \qquad (3.21)
$$

The calculus of the non-conservative loading matrix $\mathbf{G_f}$ is obtained by replacing equation 3.21 with,

$$
G_{f_{ij}}^{11}=\begin{cases}\int\int\frac{d\psi_i}{dx}\psi_j dxdy, & \text{if } x=L_x\\ 0, & \text{otherwise}\end{cases} \qquad (3.22)
$$

The aerodynamic loading matrix $\mathbf{A}$ can also be obtained by replacing 3.21 with,

$$
A_{ij}^{11}=\int\int\frac{d\psi_i}{dx}\psi_j dxdy, \qquad \text{for } 0\le x\le L_x \text{ and } 0\le y\le L_y. \qquad (3.23)
$$

From the above expressions it is possible to verify that matrices $\mathbf{G}$, $\mathbf{G_c}$, $\mathbf{G_f}$ and $\mathbf{A}$ are not positive definite and therefore the stability analysis of these equations requires special solution algorithms.

3.2.2 Numerical Integration

The numerical integration of the element matrices is performed using the Gauss quadrature rule, which can only be applied when the integral is expressed over a square region defined between $-1 < (\xi, \zeta) < 1$. As a consequence of this, coordinate transformations are required to obtain the finite element matrices for the structural discretization from the master element represented in fig. 3.2. As such, in order to calculate the integrals defined for the finite element equations, the following coordinate transformation must be applied (Reddy, 1992),

$$\begin{aligned} \frac{\partial \psi_i^e}{\partial \xi} &= \frac{\partial \psi_i^e}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial \psi_i^e}{\partial y}\frac{\partial y}{\partial \xi}, \\ \frac{\partial \psi_i^e}{\partial \zeta} &= \frac{\partial \psi_i^e}{\partial x}\frac{\partial x}{\partial \zeta} + \frac{\partial \psi_i^e}{\partial y}\frac{\partial y}{\partial \zeta}. \end{aligned} \tag{3.24}$$

Considering the presented linear interpolation functions and a four node element, the Jacobian matrix for this transformation is given by,

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^4 x_i \frac{\partial \psi_i}{\partial \xi} & \sum_{i=1}^4 y_i \frac{\partial \psi_i}{\partial \xi} \\ \sum_{i=1}^4 x_i \frac{\partial \psi_i}{\partial \zeta} & \sum_{i=1}^4 y_i \frac{\partial \psi_i}{\partial \zeta} \end{bmatrix}. \tag{3.25}$$

This matrix must be invertible in order to determine the solution of 3.24, which requires that $\det(\mathbf{J}) > 0$. This condition is satisfied if the the transformation functions $\xi = \xi(x, y)$ and $\zeta = \zeta(x, y)$ are continuous, differentiable and invertible. This condition must be evaluated for every element transformation.

Having defined the equations in the master element domain it is now possible to apply the Gauss quadrature rule to integrate the element matrices. According to this, the integral of any given function $F(\xi\zeta)$ is obtained as,

$$\int_{-1}^{1}\int_{-1}^{1} F(\xi, \zeta) d\xi d\zeta = \sum_{I=1}^{M}\sum_{J=1}^{N} F(\xi_I, \zeta_J) W_I W_J, \tag{3.26}$$

where M and N denote the number of quadrature points in the ξ and ζ directions, (ξ_I, ζ_J) denote the Gauss points and W_I and W_J denote the corresponding Gauss weights. For the used linear Lagrange interpolation functions the two point rule is used, and the corresponding values are present in the following table:

	Points (ξ_I, ζ_J)	**Weights** (W_I, W_J)
One point formula:	$(0, 0)$	$(2, 2)$
Two point formula:	$\left(\pm\frac{\sqrt{3}}{3}, \pm\frac{\sqrt{3}}{3}\right)$	(1,1)

Table 3.1: Gauss points and corresponding weights.

The one point formula is used for the reduced integration required to avoid the previously mentioned shear locking problems. When analyzing a thin plate, the shear strains are negligible and consequently the element stiffness matrix becomes stiff which will originate erroneous results.

3.2.3 Assembly of Element Equations and Boundary Conditions

Having determined the element matrices for the discretized domain, it is now necessary to assemble these matrices in order to obtain the global matrices for the complete geometry. Consider a plate with length L_x and height L_y, divided into N_{e_x} elements in the x direction and N_{e_y} elements in the y direction. For any given element (e_x, e_y) the global nodal coordinates are given by (following the numbering in fig 3.2),

$$\begin{pmatrix} N_{e_x}(e_y - 1) + e_x, \\ N_{e_x}(e_y - 1) + e_x + 1, \\ N_{e_x}e_y + e_x + 2, \\ N_{e_x}e_y + e_x + 1 \end{pmatrix} \tag{3.27}$$

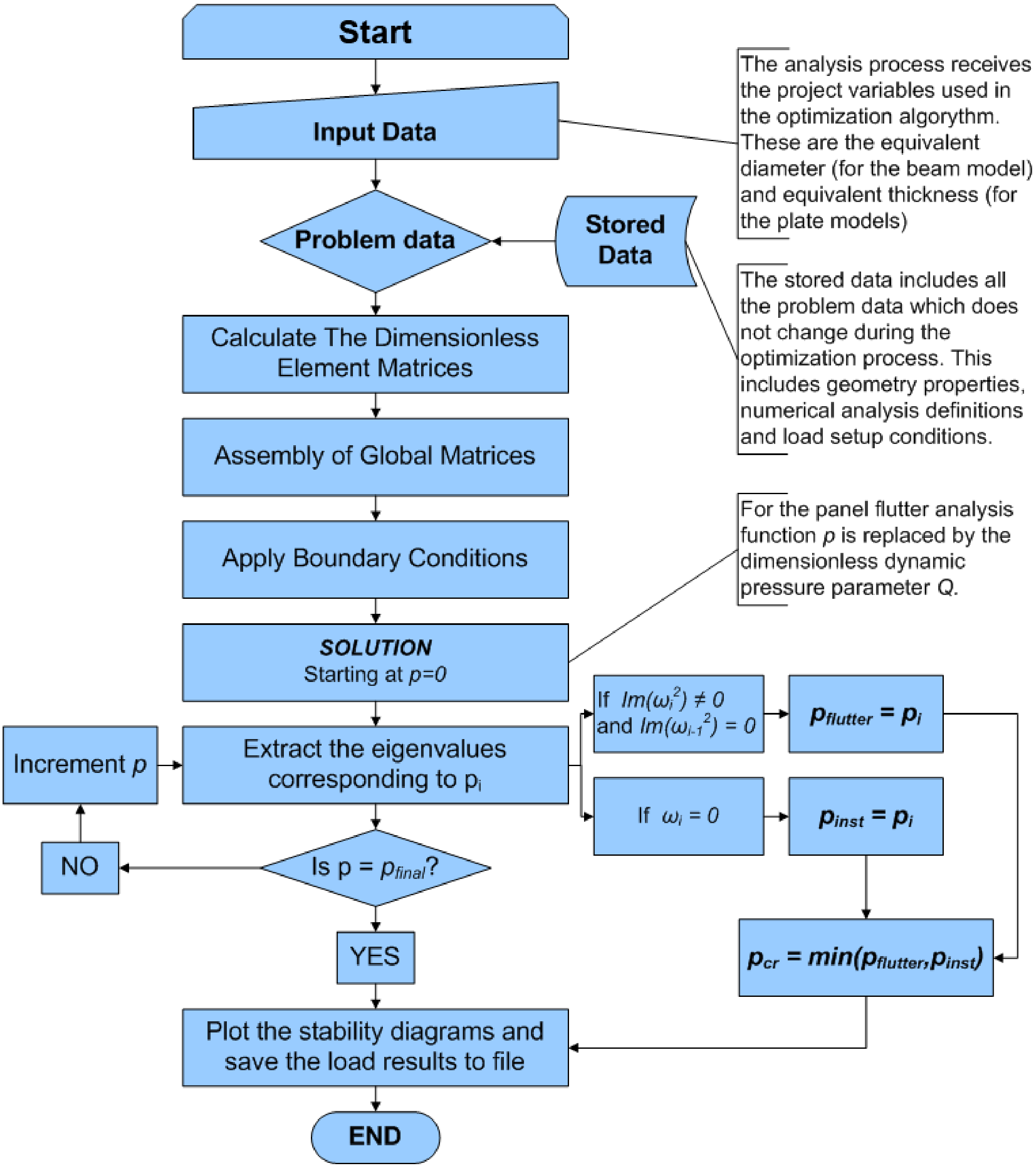
this relation is used to build the connectivity matrix for the construction of the global element matrices. The presented numbering was defined for simplicity of the connectivity relations. A shortcoming of the selected numbering method becomes evident for a rectangular plate with $N_{e_x} > N_{e_y}$ which is the case for the plate model with a non-conservative end load. In this case, the resulting global matrices will have a larger bandwidth when compared to alternative numbering sequences. This makes the solution of the finite element equations slower and for a large number of elements the solution time can be considerably affected.

For the present work, two geometric boundary conditions must be considered. First, when concerning the plate model with a non-conservative end-load, the plate side at $x = 0$ is considered to be clamped. This can be accomplished by considering that the displacements w and rotations φ_x and φ_y are zero at all the nodes located at $x = 0$. Secondly, when considering the panel flutter model, the plate is considered to be simply supported at all the edges. These conditions are obtained by considering the displacements zero at all the nodes located at the free ends and $\varphi_x = 0$ for $x = 0$, $x = a$ and $\varphi_y = 0$ for $y = 0$, $y = b$.

3.3 Stability Analysis Implementation

This section presents the methods used to obtain the solution of the boundary value problems presented by equation 2.11 for the beam model, equation 2.22 for the plate subjected to a non-conservative end load model and finally equation 2.26 for the panel flutter problem. The solution algorithm for these problems is presented in the diagram of figure 3.3. As can be seen from the presented diagram, the solution analysis scheme is performed by considering various load points between 0 and p_{final} and, for each of these load points, obtaining the corresponding eigenvalues that result from the solution of the presented eigenvalue problem. The value of p_{final} is chosen as to include the stability domain in study and the number of load points to consider is chosen as to produce accurate solutions.

The solution scheme adopted for the the eigenvalue problem uses the $MATLAB^{\circledR}$ routine `eigs` which, in turn uses *eigs* $ARPACK$ routines (presented by Lehoucq et al. (1998)) capable of solving large scale Hermitian, non-Hermitian, standard or generalized eigenvalue problems. This software is designed to compute any k eigenvalues selected by a previously defined characteristic such as the largest real part or largest magnitude. These algorithms use a variation of the Arnoldi method named the Implicitly Restarted Arnoldi Method, which is a technique for approximating the eigenvalues and corresponding eigenvectors for a general square matrix. When the matrix is symmetric, the algorithm uses a variation of the Lanczos method called the Implicitly Restarted Lanczos Method. These variations may be viewed as a synthesis of the general methods with the Implicitly Shifted QR technique which is suitable for large scale problems.



Start
Input Data
The analysis process receives the project variables used in the optimization algorythm. These are the equivalent diameter (for the beam model) and equivalent thickness (for the plate models)
Problem data
Stored Data
The stored data includes all the problem data which does not change during the optimization process. This includes geometry properties, numerical analysis definitions and load setup conditions.
Calculate The Dimensionless Element Matrices
Assembly of Global Matrices
Apply Boundary Conditions
For the panel flutter analysis function *p* is replaced by the dimensionless dynamic pressure parameter Q.
SOLUTION
Starting at *p=0*
Increment *p*
Extract the eigenvalues corresponding to p_i
If $Im(\omega_i^2) \neq 0$ and $Im(\omega_{i-1}^2) = 0$
$p_{flutter} = p_i$
If $\omega_i = 0$
$p_{inst} = p_i$
NO
Is p = p_{final}?
YES
$p_{cr} = min(p_{flutter}, p_{inst})$
Plot the stability diagrams and save the load results to file
END

Figure 3.3: Fluxogram of the stability analysis algorithm, implemented in $MATLAB^{®}$.

Chapter 4

Optimization Formulation

This chapter presents the formulation for the problem of determining minimal weight designs while maintaining stability boundaries in the structural load and frequency domains. The optimization algorithm is also described as well as the gradient analysis implementation.

The optimal designs are obtained considering a linear variation of diameter or thickness along the structure and reducing the optimization problem to the decision of the amount of material to be applied at each discretized node, as presented by Langthjem and Sugiyama (2000a).

The optimization problems presented in this work consider the optimal design of a beam subjected to a non-conservative end load, the optimal topology design of a plate subjected to the same end load and, finally, a plate subjected to a distributed aerodynamic loading.

4.1 The Optimization Problem

The purpose of the optimization method for the proposed models is to determine an optimal design which can allow the same stability boundaries as the original model. This problem can be presented by,

$$\underset{\mu}{\text{Minimize}} \quad V \tag{4.1}$$

subject to,

1. *Critical Load:* $p_{cr} \geq p_{cr}^0$

2. *Frequency Curves:* $\lambda_{n+1} - \lambda_n \geq c,$ for all $p \leq p_{cr}$

3. *Ensure Flutter Instability:* $\lambda_1 \geq 0,$ for all $p \leq p_{cr}$

4. *Design Parameters:* $\mu_j^{min} \leq \mu_i \leq \mu_j^{max},$ for $j = 1, 2, \ldots, N_e,$

where V is the dimensionless volume, p_{cr} is the dimensionless critical load, λ_n is the n^{th} dimensionless frequency and μ_i is the project variable at node i (which can be the equivalent dimensionless diameter or thickness, depending on the considered model).

The first constraint of the presented optimization problem insures that the structural critical load can increase during the optimization sequence, but can never be lower than the original critical load, obtained for the uniform structure. Constraint 2 is present to ensure that the coalescence of frequency curves will only occur for load values higher than the critical load and, by introducing a frequency separation constant c, gives robustness to the optimization process. By keeping frequencies apart for a specified distance, the possibility of a large variation of the critical load with small design changes is avoided (as discussed by Langthjem and Sugiyama (1999); Odaka and Furuya (2005)). The third constraint is used when the structural instability mode turns into flutter and ensures that the instability mode afterwards remains to be flutter. This constraint is implemented to avoid convergence problems in the optimization process due to the cyclic change of instability modes (as discussed by Langthjem and Sugiyama (2000a)). For the formulation of the panel flutter model, the load p is replaced by the dynamic pressure parameter Q, and constraint 3 is not considered. The design parameters are limited as described by constraint 4. These limits can be imposed with different values for each node in order to allow the introduction of areas in which large material changes are not desired. They can also be changed during the optimization process in order to control the evolution of the optimal designs.

4.2 The Method of Moving Asymptotes

The solution for the optimization problem is obtained with the Method of Moving Asymptotes (developed by Svanberg (1987)). The Matlab version of the MMA algorithm used in the present work, requires for the previous optimization problem to be written in the following form,

$$\underset{\mu}{\text{Minimize}} \quad f_0(\mu) = V \tag{4.2}$$

subject to,

$$\begin{array}{lll} f_1(\mu) = p_{cr}^0 - p_{cr} \leq 0 & & \\ f_i(\mu) = \lambda_n - \lambda_{n+1} + c \leq 0, & \text{for} & i = 2, \ldots, b \\ f_i(\mu) = -\lambda_1 \leq 0, & \text{for} & i = b, \ldots, m \\ \mu_j^{min} \leq \mu_j \leq \mu_j^{max}, & \text{for} & j = 1, \ldots, n \end{array} \tag{4.3}$$

where b and m depend upon the number of natural frequencies taken into account for the analysis and the number of discretized load points in the interval $[0; p_c r]$ that are considered. The value of n corresponds to the number of project variables and thus corresponds to the total number of nodes in the finite element model.

The solution scheme adopted in the Method of Moving Asymptotes is then given as follows (as presented by Svanberg (1987)): At each iteration, the current iteration point ($\mu^{(k)}$) is given. Then, an approximating explicit subproblem is generated. In this subproblem, the functions $f_i(\mu)$ are replaced by approximating convex functions $\tilde{f}_i^{(k)}(\mu)$. These approximations are essentially based on the gradient information for the current point, but they also take implicitly into account the information from previous iteration points. The solution for the subproblem is obtained and the new unique optimal solution becomes the next iteration point ($\mu^{(k+1)}$). After this, a new subproblem is originated until the solution converges. The subproblem for the optimization formulation, stated by equations 4.2 and 4.3, takes the form,

$$\underset{\mu,}{\text{Minimize}} \quad \tilde{f}_0^{(k)}(\mu) \tag{4.4}$$

subject to,

$$\begin{array}{lll} \tilde{f}_i^{(k)}(\mu) \leq 0, & \text{for} & i = 1, \ldots, m \\ \alpha_j^{(k)} \leq \mu_i \leq \beta_j^{(k)}, & \text{for} & j = 1, \ldots, n, \end{array} \tag{4.5}$$

where the approximation functions $\tilde{f}_i^{(k)}(\mu)$ are calculated from,

$$\tilde{f}_i^{(k)}(\mu) = \sum_{j=1}^{n} \left(\frac{p_{ij}^{(k)}}{u_j^{(k)} - \mu_j} + \frac{q_{ij}^{(k)}}{\mu_j - l_j^{(k)}} \right) + r_i^{(k)}, \quad i = 0, 1, \ldots, m, \tag{4.6}$$

where,

$$p_{ij}^{(k)} = \left(u_j^{(k)} - \mu_j^{(k)}\right)^2 \left[\left(\frac{\partial f_i}{\partial x_j}(\mu^{(k)}) \right)^+ + \kappa_{ij}^{(k)} \right], \tag{4.7}$$

$$q_{ij}^{(k)} = \left(\mu_j^{(k)} - l_j^{(k)}\right)^2 \left[\left(\frac{\partial f_i}{\partial x_j}(\mu^{(k)}) \right)^- + \kappa_{ij}^{(k)} \right], \tag{4.8}$$

$$r_i^{(k)} = f_i(\mu^{(k)}) - \sum_{j=1}^{n} \left(\frac{p_{ij}^{(k)}}{u_j^{(k)} - \mu_j^{(k)}} + \frac{q_{ij}^{(k)}}{\mu_j^{(k)} - l_j^{(k)}} \right), \tag{4.9}$$

$$\alpha_j^{(k)} = \max\left\{\mu_j^{min}, \quad 0.9 l_j^{(k)} + 0.1 \mu_j^{(k)}\right\} \tag{4.10}$$

$$\beta_j^{(k)} = \min\left\{\mu_j^{max}, \quad 0.9 u_j^{(k)} + 0.1 \mu_j^{(k)}\right\}, \tag{4.11}$$

and where,

$$\begin{aligned}\left(\frac{\partial f_i}{\partial x_j}(\mu^{(k)})\right)^+ &= \max\left\{0, \frac{\partial f_i}{\partial x_j}(\mu^{(k)})\right\} && (4.12)\\ \left(\frac{\partial f_i}{\partial x_j}(\mu^{(k)})\right)^- &= \max\left\{0, -\frac{\partial f_i}{\partial x_j}(\mu^{(k)})\right\}. && (4.13)\end{aligned}$$

The lower asymptotes, $l_j^{(k)}$ and upper asymptotes $u_j^{(k)}$ are calculated for the first two iterations from,

$$\begin{aligned}l_j^{(k)} &= \mu_j^{(k)} - 0.5\left(\mu_j^{max} - \mu_j^{min}\right), && (4.14)\\ u_j^{(k)} &= \mu_j^{(k)} + 0.5\left(\mu_j^{max} - \mu_j^{min}\right). && (4.15)\end{aligned}$$

For $k \geq 3$ the asymptotes take are calculated through,

$$\begin{aligned}l_j^{(k)} &= \mu_j^{(k)} - \gamma_j^{(k)}\left(\mu_j^{(k-1)} - l_j^{(k-1)}\right), && (4.16)\\ u_j^{(k)} &= \mu_j^{(k)} + \gamma_j^{(k)}\left(u_j^{(k-1)} - \mu_j^{(k-1)}\right), && (4.17)\end{aligned}$$

in which $\gamma_j^{(k)}$ are a set of parameters that can be adjusted to influence the evolution of the optimization process. The default values are,

$$\gamma_j^{(k)} = \begin{cases} 0.7, & \text{if} \quad \left(\mu_j^{(k)} - \mu_j^{(k-1)}\right)\left(\mu_j^{(k-1)} - \mu_j^{(k-2)}\right) < 0 \\ 1.2, & \text{if} \quad \left(\mu_j^{(k)} - \mu_j^{(k-1)}\right)\left(\mu_j^{(k-1)} - \mu_j^{(k-2)}\right) > 0 \\ 1, & \text{if} \quad \left(\mu_j^{(k)} - \mu_j^{(k-1)}\right)\left(\mu_j^{(k-1)} - \mu_j^{(k-2)}\right) = 0 \end{cases} \qquad (4.18)$$

and the default values for $\kappa_{ij}^{(k)}$ are,

$$\kappa_{ij}^{(k)} = 10^{-3}\left|\frac{\partial f_i}{\partial x_j}(\mu^{(k)})\right| + \frac{10^{-6}}{u_j^{(k)} - l_j^{(k)}}, \qquad \text{for} \quad i = 0, 1, \ldots, m \quad \text{and} \quad j = 1, \ldots, n. \qquad (4.19)$$

This results that the approximation functions $\tilde{f}_i^{(k)}$ are strictly convex, which in turn implies that there is always a unique optimal solution of the MMA subproblem. It is possible to verify that $\tilde{f}_i^{(k)}$ are always first order approximations of the original functions f_i at each iteration point. From these equations it is then possible to obtain the project variables μ_j which define the new optimized design.

4.3 Sensitivity Analysis

As shown, the optimization procedure previously presented requires the knowledge of the partial derivatives of the design constraints and objective function with respect to the project variables. The determination of these equations requires the sensitivity analysis of the dynamic models presented in chapters 2 and 3. The sensitivity analysis described in this section follows the work presented by Pedersen (1994, 2003). Considering equations 2.11 for the beam model, 2.22 for the plate model and 2.26 for the panel flutter model, we have the following functional,

$$[L]\{w\} = 0, \tag{4.20}$$

where the matrix $[L]$ depends on the complex eigenvalue λ with α as a stability measure and ω as the frequency values according to the definition presented. In addition, $\mathbf{L}$ depends also on the applied load, p (Q in the panel flutter model), and on the design variables defined as μ. For the present sensitivity analysis, the adjoint problem for equation 4.25 must also be considered. This Hermitian Adjoint can be obtained parting from the functional Λ, defined in equation 2.8, as follows,

$$\int_0^1 v\Lambda(w)dx = \int_0^1 w\Lambda^*(v)dx, \tag{4.21}$$

this equation if satisfied when,

$$\begin{aligned} \Lambda^*(v) = (Iv'')'' - \lambda^2 mv + pv'' = 0, \\ v(0) = 0, \quad v'(0) = 0, \\ Iv''(1) + p\eta v(1) = 0, \quad (Iv'')'(1) + pv'(1) = 0. \end{aligned} \tag{4.22}$$

From this equation and as previously presented in the analytical formulation the following functional can be obtained,

$$L(v) = \int_0^1 \left[Iv'' - \lambda^2 v + pv'\right] dx + \eta pv(1) \tag{4.23}$$

which is also stationary with respect to variations δv, thus satisfying the kinematic boundary conditions, $v(0) = 0$ and $v'(0) = 0$. For any numerical discretization method, this functional originates the following equation,

$$\mathbf{L}^T\mathbf{v} = \left[\mathbf{K} - \lambda^2\mathbf{M} + p\left(\mathbf{G_c} + \eta\mathbf{G_f}\right)\right]^T \mathbf{v} = 0, \tag{4.24}$$

and this expression, when written in the same form as 4.25, becomes

$$\{v\}^T[L] = 0, \tag{4.25}$$

The use of the adjoint problem is needed because, as previously discussed, matrix $[L]$ is non-symmetric and therefore the presented problem is not self-adjoint and $\{w\} \neq \{v\}$. This means that both the right eigenvector w and the left eigenvector v need to be considered for the sensitivity calculation of any eigenvalue (which can be the frequency values or the critical load, as will be shown). For simplicity in the present formulation, the derivation of the sensitivity analysis equations will be performed in terms of a specific mutual energy. The term specific means that the energy may be presented per unit area or volume (depending on the considered model) and the term mutual indicates that two different eigenvalues are used to determine the energy. The resulting functional is given by,

$$\Pi = \{v\}^T[L]\{w\} = 0 \tag{4.26}$$

and is called the specific mutual energy. This functional is obtained from 4.25 and 4.24. considering any variation of Π it results that,

$$\delta\Pi = 0 \Rightarrow \{\delta v\}^T[L]\{w\} + \{v\}^T[\delta L]\{w\} + \{v\}^T[L]\{\delta w\} = 0 \tag{4.27}$$

It is possible to verify that the variations of the eigenvectors $\{v\}$ and $\{v\}$ disappear, which justifies the introduction of the adjoint eigenvector and the variations of Π instead of 4.25. Considering the sensitivity analysis for $[\delta L] \neq 0$, equation 4.27 can be written as variations of the independent parameters δp, $\delta\lambda$ and $\delta\mu$ as the following functionals

$$\begin{aligned}\mathcal{A} &= \{v\}^T\frac{\partial[L]}{\partial p}\{w\},\\ \mathcal{B} &= \{v\}^T\frac{\partial[L]}{\partial\lambda}\{w\},\\ \mathcal{C} &= \{v\}^T\frac{\partial[L]}{\partial\mu}\{w\}.\end{aligned} \tag{4.28}$$

From this, equation 4.27 can be re-written as,

$$\mathcal{A}(\delta\alpha + i\delta\omega) + \mathcal{B}\delta p + \mathcal{C}\delta\mu = 0. \tag{4.29}$$

This equation allows for the direct deduction of the partial derivatives required for the optimization problem.

The sensitivity of the flutter load, p_{cr} with respect to design changes can be obtained from equation 4.29 by considering that the instability is initiating and it is valid to assume that $\alpha = 0$ and $\delta\alpha = 0$. For $\mathcal{A} \neq 0$ and $i\delta\omega$ purely imaginary it is possible to write (as presented by Pedersen (2003)),

$$\frac{\partial p_{cr}}{\partial\mu} = \frac{-Re\left(\frac{\mathcal{C}}{\mathcal{A}}\right)}{Re\left(\frac{\mathcal{B}}{\mathcal{A}}\right)}. \tag{4.30}$$

One can note that when $\omega = \delta\omega = 0$ we have a divergence load. Considering the previously presented finite element formulation for a beam or plate subjected to a partial non-conservative load, equation 4.30 can be written as (reference Langthjem and Sugiyama (2000a))

$$\frac{\partial p_{cr}}{\partial\mu_j} = \frac{\mathbf{v}^T\left(\partial\mathbf{L}/\partial\mu_j\right)\mathbf{w}}{\mathbf{v}^T\left(\mathbf{G_c} - \eta\mathbf{G_f}\right)\mathbf{w}} \tag{4.31}$$

Analogously, the sensitivity of the critical dynamic pressure with respect to the design variables is given by (following Odaka and Furuya (2005)),

$$\frac{\partial Q_{cr}}{\partial\mu_j} = \frac{\mathbf{v}^T\left(\partial\mathbf{L}/\partial\mu_j\right)\mathbf{w}}{\mathbf{v}^T\mathbf{A}\mathbf{w}} \tag{4.32}$$

which is used for the panel flutter analysis. Following the same line of thought, the sensitivity of the frequency with respect to design changes can be obtained from 4.29 by considering that the instability criterion is $\alpha = 0$ and $Re(\mathcal{B}/\mathcal{A}) < 0$. Considering once again that the instability is initiating, we have that δp is real and $\delta a = 0$ resulting in (from Pedersen (2003)),

$$\frac{\partial\lambda}{\partial\mu} = \frac{-Im\left(\frac{\mathcal{C}}{\mathcal{B}}\right)}{Re\left(\frac{\mathcal{A}}{\mathcal{B}}\right)} \tag{4.33}$$

The necessary condition that $\mathcal{B} \neq 0$ follows from the fact that $Re(\mathcal{B}/\mathcal{A}) < 0$. The finite element form of this equation is (from Langthjem and Sugiyama (2000a)),

$$\frac{\partial\lambda}{\partial\mu_j} = \frac{\mathbf{v}^T\left(\partial\mathbf{L}/\partial\mu_j\right)\mathbf{w}}{\mathbf{v}^T 2\lambda\mathbf{M}\mathbf{w}} \tag{4.34}$$

and $\lambda = \omega$ for all $p < p_{cr}$.

In the presented optimization problem, the objective function is the dimensionless structural volume, V, which for the beam model is calculated through a linear combination of the element lengths and nodal diameters as,

$$V = \mathbf{a}^T \mu, \quad \mathbf{a} = [l_1, l_1 + l_2, l_2 + l_3, \ldots, l_{N_e - 1} + l_{N_e}, l_{N_e}] \tag{4.35}$$

where l are the element lengths and μ the project variables. For the plate model, the dimensionless volume is a linear combination of element lengths and heights with the nodal thickness, and is given by,

$$V = \sum_{j=1}^{N_{ey}+1} \sum_{j=1}^{N_{ex}+1} \mu_k b_i c_j, \qquad k = i + (j-1)(N_{e_x} + 1) \tag{4.36}$$

with,

$$\begin{aligned} \mathbf{b} &= \left[l_{x1}, l_{x1} + l_{x2}, l_{x2} + l_{x3}, \ldots, l_{xN_{e_x}-1} + l_{xN_{e_x}}, l_{xN_{ex}} \right] \\ \mathbf{c} &= \left[l_{y1}, l_{y1} + l_{y2}, l_{y2} + l_{y3}, \ldots, l_{yN_{e_y}-1} + l_{yN_{e_y}}, l_{yN_{e_y}} \right] \end{aligned}$$

where l_x is the element lengths in the x direction and l_y the lengths in the y direction. It is now possible to write the derivatives of the volume with respect to changes in the design variables as,

$$\frac{\partial V}{\partial \mu_j} = a_j \tag{4.37}$$

for the beam model and,

$$\frac{\partial V}{\partial \mu_k} = b_i c_j, \qquad k = i + (j-1)(N_{e_x} + 1) \tag{4.38}$$

for the plate models. It should be noted that these derivatives are constant and do not depend on the design variables.

4.4 Computational Implementation

The implementation of the described optimization procedure is represented in figure 4.1. In the formulation of the optimization problem, the frequency constraints are applied in the load range $0 \leftrightarrow p_{cr}$. In order to apply this constraint, this range is divided into a set of equally spaced points, and the constraints are calculated at each of these points. The number of frequencies that are considered in the optimization process must also be defined before the iterative process begins. The selection of the number of frequencies must take into account the structural stability and the number of instability modes to be considered. After these parameters are defined, a first stability analysis is performed in order to obtain the relevant data for the initial structure, which will be the reference data for the optimization process. The structural volume for the uniform structure is set to 1. (according to the dimensionless formulation for equations 4.35 and 4.36).

As the optimization cycle starts, sensitivities are calculated for the critical load and the selected frequencies at each point, and the data is sent to the MMA optimization algorithm which generates a new set of design variables. A new stability analysis is performed for the new design and the volume is calculated. The new volume value is compared with the previous iteration and the relative volume variation is calculated. If this variation is under a predetermined value (the default is set to 0.01 %), the algorithm considers that the optimization process has converged, presents the final values and ends the function. The cycle will also end if the optimization process does not converge before the function reaches the maximum allowed iterations, *maxiter* (the default value of *maxiter* is set to 150).

At each iteration the limits imposed to the project variables by μ_j^{min} and μ_j^{max} are recalculated according to a predetermined slack parameter as well as the structural volume from,

$$\begin{aligned} \mu_j^{min} &= \max\left[(1-\epsilon)\mu_j^{(k+1)}, \mu_{min}\right] \\ \mu_j^{max} &= \min\left[(1+\epsilon)\mu_j^{(k+1)}, \mu_{max}\right], \end{aligned} \tag{4.39}$$

where μ_{min} is the lowest allowable value for any project variable at any iteration, μ_{max} is the highest allowed value and ϵ is the slack parameter, which is reduced when the structural volume goes under predetermined values. These values are user controlled and must be defined according to the expected evolution of the optimization process and experience with the optimization algorithm. The purpose of these limits is to ensure that the optimal designs have a smooth evolution along the optimization process, especially when the volume is greatly reduced, since the structure becomes much more sensitive to the structural changes. A more detailed discussion on the importance and effects of the use of moving limits can be followed in the work from Neves et al. (2000).

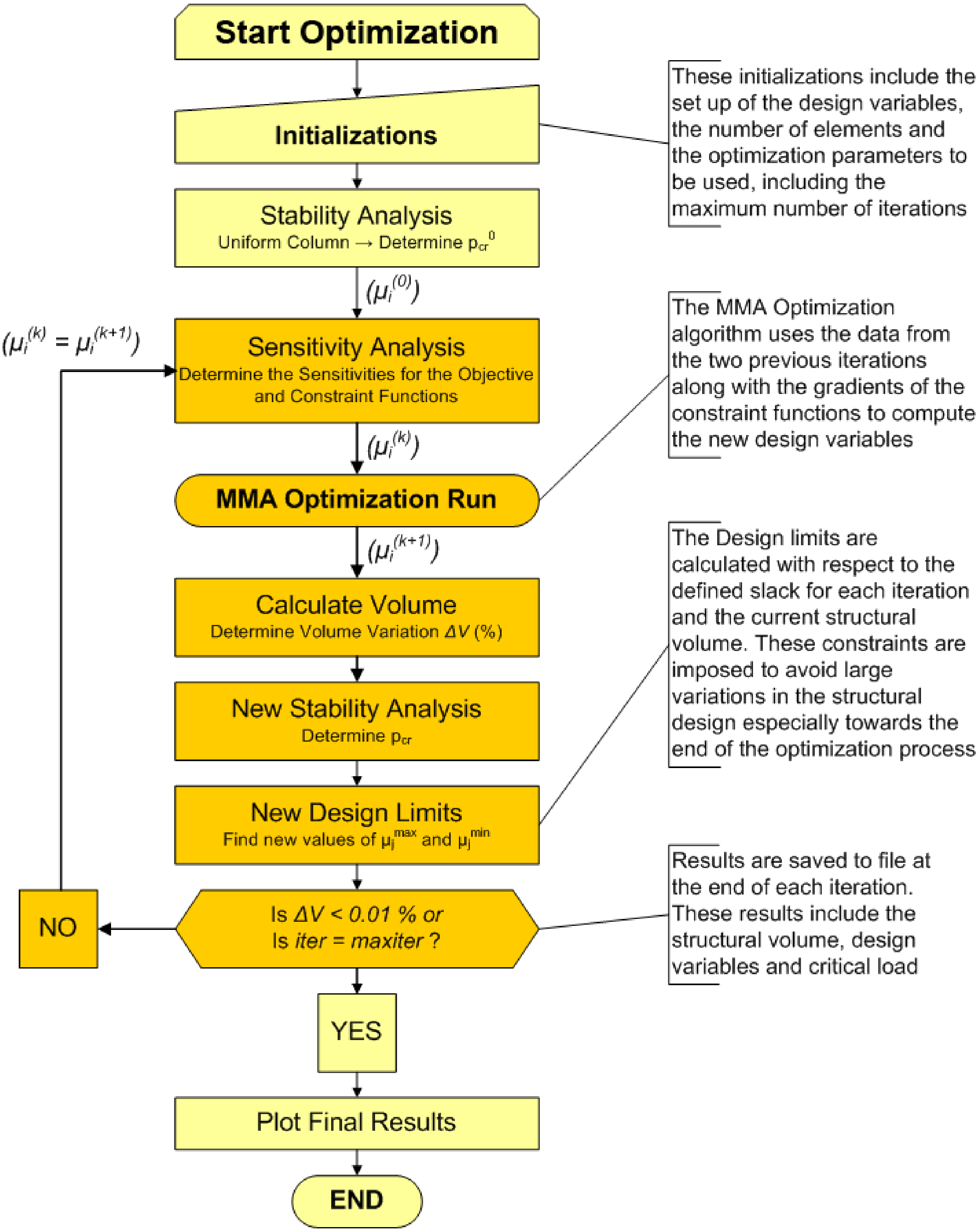

Start Optimization
Initializations
Stability Analysis
Uniform Column → Determine p_{cr}^0
$(\mu_i^{(0)})$
$(\mu_i^{(k)} = \mu_i^{(k+1)})$
Sensitivity Analysis
Determine the Sensitivities for the Objective and Constraint Functions
$(\mu_i^{(k)})$
MMA Optimization Run
$(\mu_i^{(k+1)})$
Calculate Volume
Determine Volume Variation ΔV (%)
New Stability Analysis
Determine p_{cr}
New Design Limits
Find new values of μ_j^{max} and μ_j^{min}
Is $\Delta V < 0.01$ % or Is *iter = maxiter* ?
NO
YES
Plot Final Results
END
These initializations include the set up of the design variables, the number of elements and the optimization parameters to be used, including the maximum number of iterations
The MMA Optimization algorithm uses the data from the two previous iterations along with the gradients of the constraint functions to compute the new design variables
The Design limits are calculated with respect to the defined slack for each iteration and the current structural volume. These constraints are imposed to avoid large variations in the structural design especially towards the end of the optimization process
Results are saved to file at the end of each iteration. These results include the structural volume, design variables and critical load

Figure 4.1: Fluxogram representing the optimization process, implemented in *MATLAB*

Chapter 5

Results

The results obtained for the described flutter analysis of the presented models and the optimized designs that result from the implementation of the presented optimization procedure are discussed in this chapter. For each model, results are compared and verified with available bibliographical results and commercial finite element software. The convergence of the finite element methods is also considered as well as the influence of the number of elements considered for the discretization on the optimized results.

5.1 Column Subjected to a Partial Non-Conservative End Load

5.1.1 Code Verification

In order to proceed with the stability analysis of the presented problem of a column subjected to an end load with a non-conservative component, it is first necessary to verify if the developed analysis algorithm is producing valid results. In order to do this, the obtained results are compared with the ones obtained from the commercial finite element analysis software $ANSYS^{®}$.

As mentioned previously, when solving the problem posed by equation 2.11 if the frequency λ is purposely set to zero, the resulting critical load, p_{cr} will correspond to the analogous Euler load of the column. Likewise, if the load p is set to zero, the resulting frequencies will correspond to the structural natural frequencies. These results are compared with the results obtained solving the analogous problem with $ANSYS^{®}$, and are summarized on table 5.1. These results where obtained for a uniform aluminium alloy column with a Young's modulus of $72GPa$, density of $2800Kg/m^3$, Poisson ratio of $\nu = 0.2857$, a cross sectional area of $0.0025m^2$, second moment of area of $5.2e - 7m^4$ and a length of $1m$. The model was discretized with 10 finite elements in both cases. It is possible to verify that the results are almost identical for the considered numerical precision with the advantage that the analysis run in the $MATLAB^{®}$ developed code was performed slightly faster than the $ANSYS^{®}$ run.

	First Frequency		**Critical Load**	
$MATLAB^{®}$ **code**	40.958	Hz	92527.62	N
$ANSYS^{®}$	40.938	Hz	92528.00	N
Relative Deviation	0.05	%	0.00	%

Table 5.1: Results verification for the beam model.

For the Euler-Bernoulli column problems, the number of elements used in the finite element formulation has a significant influence only on the number and precision of eigenvalues (and

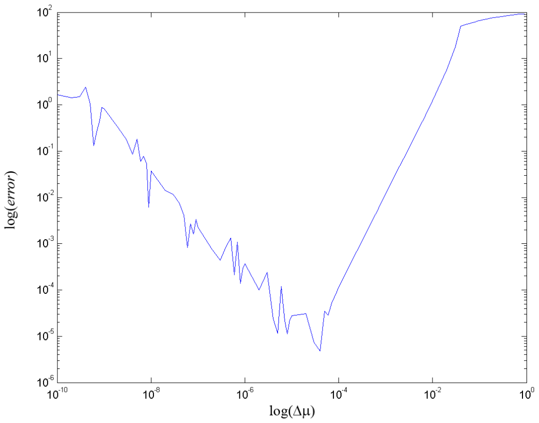


Figure 5.1: Relative error vs. $\Delta\mu$ for the sensitivity analysis of the critical load for the 11^{th} design variable and a load parameter of $\eta = 1$.

eigenvectors) that can be extracted both for the frequency values and the critical loads. The general empiric rule applied to define the number of finite elements is that in order to analyze n frequency modes, the number of elements for the finite element discretization should be $2n$ or higher. For the analysis considered in this section, the first 5 eigenvalues and eigenvectors were considered the most relevant for both the stability analysis and the optimization process. Therefore, 10 finite elements were selected both for convergence purposes and in order to obtain an identical discretization as the one used by Langthjem and Sugiyama (2000a).

The sensitivity analysis for the frequency and critical load constraints, calculated from equations 4.31, 4.32 and 4.33 were validated by comparison with central finite differences. According to this numerical method, the previous derivatives are calculated through,

$$\begin{aligned}
\frac{\partial p_{cr}}{\partial \mu_j} &= \frac{p_{cr}(\mu_j + \Delta\mu) - p_{cr}(\mu_j - \Delta\mu)}{2\Delta\mu} \\
\frac{\partial Q_{cr}}{\partial \mu_j} &= \frac{Q_{cr}(\mu_j + \Delta\mu) - Q_{cr}(\mu_j - \Delta\mu)}{2\Delta\mu} \\
\frac{\partial \lambda}{\partial \mu_j} &= \frac{\lambda(\mu_j + \Delta\mu) - \lambda(\mu_j - \Delta\mu)}{2\Delta\mu}
\end{aligned} \tag{5.1}$$

The results obtained from the analytical equations were compared with the numerical results calculated from 5.1 for various distances $\Delta\mu$ and the relative error was determined for each design variable. For the current analysis, the minimum errors obtained were always under a magnitude of 10^{-5} %. As an example, figure 5.1 shows the typical results of this comparative study. The presented diagram shows the evolution of the relative error with the variation of $\Delta\mu$ for the sensitivity calculus of the critical load for the 11^{th} design variable and a load parameter of $\eta = 1.0$.

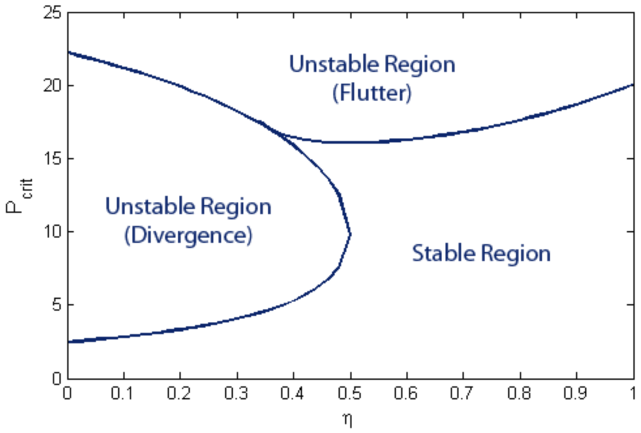


Figure 5.2: Stability diagram of the uniform column.

5.1.2 Stability Results

Having verified the implemented finite element code, the solution of the full problem presented by 2.11 can now be studied. As previously described, the numerical analysis of the problem is performed by extracting the frequencies corresponding to a discretized set of load values between 0 and p_{final}. Evaluating these frequency values, it is possible to identify the loads for which the structure loses rigidity and the corresponding instability mode. For a given load parameter μ, when a frequency reaches zero, the instability mode is divergence. Flutter occurs when two frequency values coalesce and become complex, resulting also in the loss of structural rigidity.

Figure 5.2 represents the dimensionless stability regions for a uniform column for various load combinations, from a pure conservative load ($\eta = 0$, an Euler buckling problem) to a pure non-conservative load ($\eta = 1$, Beck's column). As it can be seen from the diagram, the main instability mode changes from divergence to flutter at $\eta = 0.5$. Although flutter instability can be obtained for lower values of η (flutter instability modes for the uniform column appear for a load component of approximately $\eta = 0.35$), it is at $\eta = 0.5$ that the lowest instability load becomes flutter instead of divergence and consequently becoming the main instability mode. From this diagram it is also possible to notice that as divergence turns into flutter, the stability margin also increases significantly. This is mainly because damping effects are not considered in the present analysis.

As mentioned earlier, flutter occurs when frequency coupling is verified at any given load condition. Thus, one of the most important analysis aspects to consider when studying this type of instability problems is the structural load-frequency response. Figure 5.4 shows the dimensionless load-frequency curves for the dimensionless uniform column for several load types. These results where obtained for a p_{final} value of 150 and 3000 load steps and the represented frequency values correspond only to the real part of the frequency values.

The Euler column (subjected to a conservative load, $\eta = 0.0$) is shown in figure 5.4a). The instability mode for this column is divergence and the first critical load occurs at $p_{cr} = 2.467$. Figure 5.4b) represents the structural response for a load with a 25% non conservative component. For this column, the instability mode is also divergence, and occurs at $p_{cr} = 3.651$. When the non-conservative load is 50% of the total load, as represented in figure 5.4c), the first instability mode is still divergence (at $p_{cr} = 9.870$), but flutter instability can now occur at $p_{cr} = 16.1$. As shown in figure 5.4d) flutter is the main instability mode for a 75% non-conservative load, and instability occurs at a load of $p_{cr} = 17.2$. Finally, when the load is completely non-conservative we have the problem described by Beck's column, and flutter occurs at $p_{cr} = 20.05$ as shown in

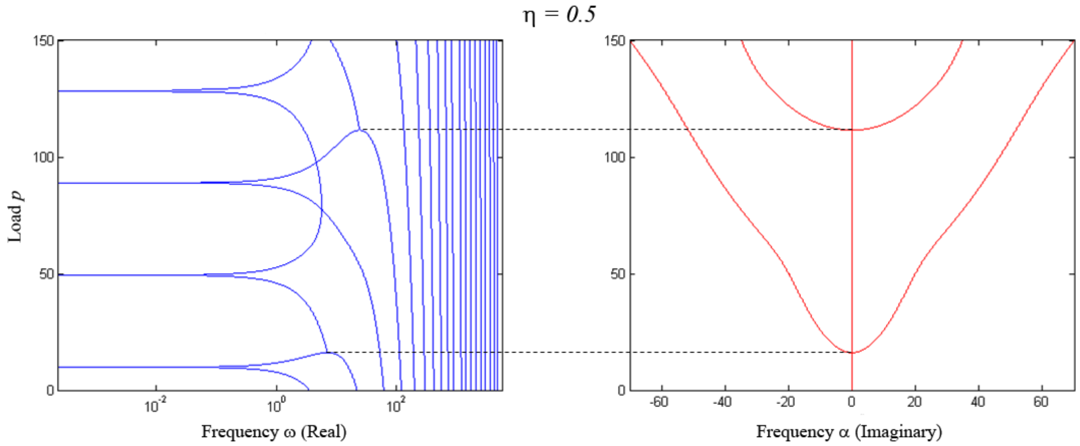


Figure 5.3: Load frequency curves for a load parameter of $\eta = 0.5$, representing the real and imaginary parts of the frequency values.

figure 5.4e). These results show good agreement with the work developed by other authors (see Langthjem and Sugiyama (2000a), for instance).

In order to best understand and identify the origin of flutter instability the frequency analysis must also take into account the complex domain. Considering that the displacements as given by 2.5, it is possible to verify that small amplitude vibrations will be stable at any given load p_i if $\omega_j > 0$ for all j and for all loads in the interval $[0; p_i]$. The critical load can be defined as divergence when, for a load p_d, we have $\omega_j = 0$ at $p = P_d$ and $\omega_j > 0$ and $\alpha_j = 0$ for all loads in the interval $[0; p_d[$. The critical load becomes flutter if, for any load p_f, $\omega_j > 0$ and $\alpha \neq 0$. The structural critical load can then be defined as the minimum of these two loads. Figure 5.3 shows the frequency load curves of a uniform column with a load parameter of $\eta = 0.5$, representing both the real part of the frequency and the imaginary part. This load condition was chosen because at this load condition the main instability mode changes from divergence ($\eta < 0.5$) to flutter ($\eta > 0.5$). As the image shows, the critical load for the structure is divergence and $p_d = 9.870$. Notwithstanding, at this load condition flutter modes are also present. From the represented imaginary part of the frequency it is possible to identify two flutter modes. At the flutter load, while the real part of the two adjacent frequencies coalesce, their imaginary parts become non-zero and originate the two symmetric branches presented in the diagram. The first load occurs at a load of $p_f = 16.05$.

The obtained results show a good agreement with the reported results by Langthjem and Sugiyama (2000a, 1999) as well as the results presented by Zuo and Schreyer (1996). The presented analysis of the load-frequency response and the determination of the critical loads for the uniform structure presents the initial steps of the optimization analysis whose results are presented in the following section.

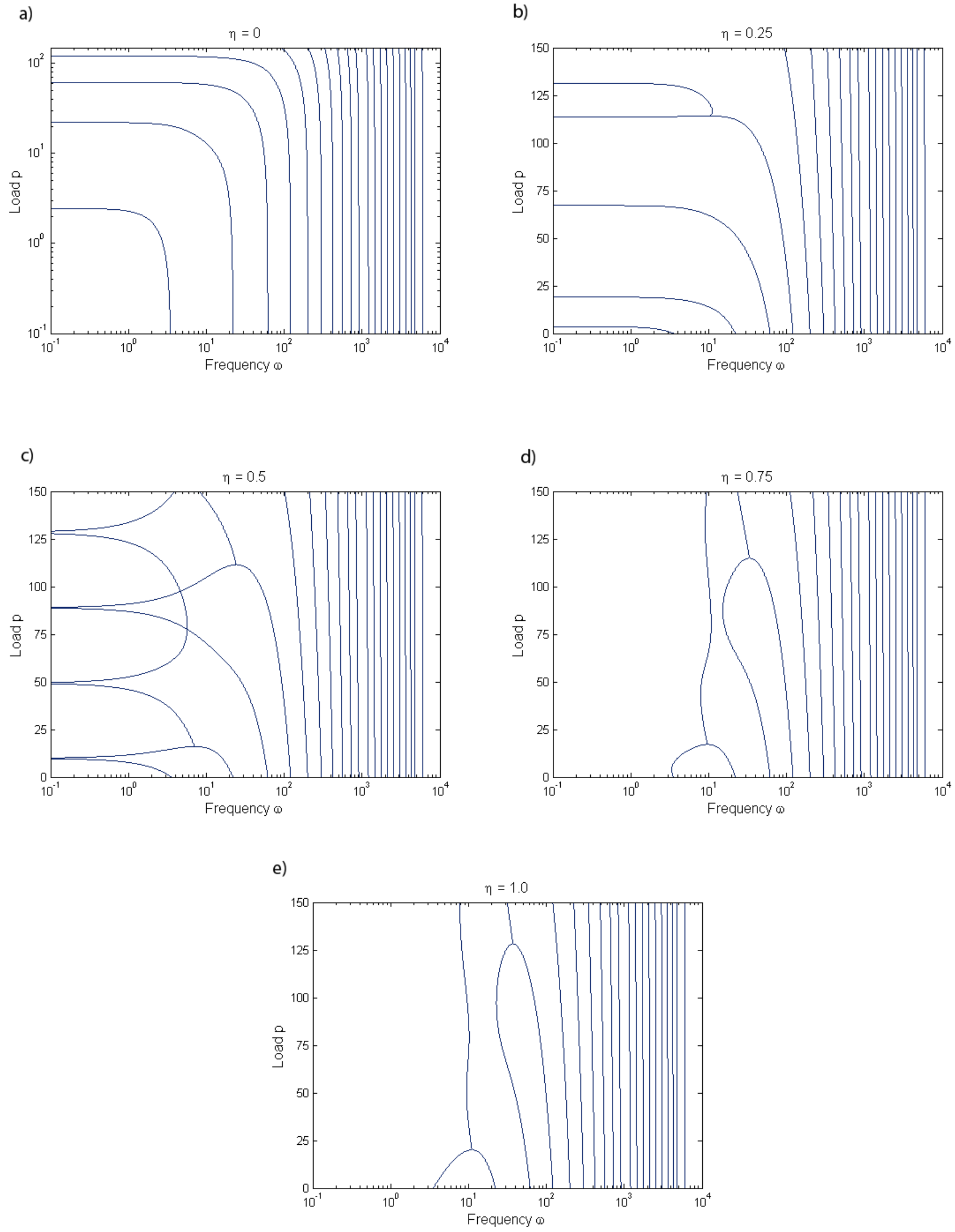


Figure 5.4: Load-Frequency curves for various load conditions, from a conservative force ($\eta = 0$) to a pure non-conservative load (Beck's column at $\eta = 1.0$). The presented results were obtained for a uniform **beam** with a dimensionless diameter of $\mu = 1.0$.

5.1.3 Optimization Results

This section presents the optimized structures obtained from the dimensionless uniform columns with stability characteristics as presented in the previous section. A summary of the optimization results obtained for the five load conditions considered in the previous section is presented in the following table: These results were obtained using 3000 load steps for the stability analysis and 20 load points between 0 and p_{cr} for the calculus of the frequency sensitivities. The maximum and minimum values allowed for the design variables were selected as $\mu_{max} = 10$ and $\mu_{min} = 10^{-8}$, respectively. The slack parameter presented in equation 4.39, ϵ, used to define the design limits at each iteration was selected according to,

Load Condition (η)	**0.00**	**0.25**	**0.50**	**0.75**	**1.00**
Optimized Volume	0.866	0.476	0.473	0.490	0.374
Volume Reduction (%)	13.4	52.4	52.7	50.9	62.6
Reference Results (Langthjem and Sugiyama, 2000a)	0.866	–	0.486	–	0.379
Number of iterations	8	56	23	82	105

Table 5.2: Optimization results for the beam model.

$$\epsilon = \begin{cases} 0.3, & \text{for} \quad 0.5 < V \leq 1 \\ 0.1, & \text{for} \quad 0 < V \leq 0.5 \end{cases}$$

these conditions where always used to define the new limits of the design variables at each iteration unless the new values exceeded the absolute limits imposed by μ_{min} and μ_{max}, in which case, the new design limits for the new iteration would be the respective absolute limit. The frequency separation constant was set to $c = 10$ for all load conditions except for $\eta = 0$, where $c = 0$. These frequency separation values where selected in accordance to the values used by Langthjem and Sugiyama (2000a).

The *MMA* tuning parameters, defined in equation 4.18 where also adjusted for the various optimization runs, in order to improve the optimized solutions. The adjustments made to these parameters were based on the experience gained along several optimization runs.

As the results from the table show, the optimization process allowed for considerably large volume reductions while respecting all the imposed constraints. Comparing these results with the work presented by Langthjem and Sugiyama (2000a) once more, it is possible to verify that, for identical load conditions, the optimized results are lower, or equal at the worst, to the ones presented by these authors.

Figure 5.5 shows the shape of the dimensionless optimized columns for the presented results. The stability diagrams for these columns are presented in figure 5.6. From these results it is possible to confirm that the stability boundaries are maintained for the optimized columns, and the critical loads are all equal or above the values obtained for the uniform columns. These diagrams were obtained from the following project variables, As expected, divergence is only present on the column subjected to a pure conservative load ($\eta = 0.0$, figure 5.6 *a)*), with flutter as the first instability mode for all the other columns (subjected to a non-conservative load component). As can also be noted from figure 5.6, the flutter loads are almost of the same magnitude. As the optimization process evolved for these columns it was possible to note that the convergence of the solutions was somewhat delayed because the main instability mode kept changing from the first mode (corresponding to coalescence of the first and second frequencies) to the second (corresponding to the third and fourth frequencies). In spite of the slow convergence the frequency separation constraints avoided some more serious convergence problems. For the load cases described on figures 5.6 *c)*, *d)* and *e)*, the critical flutter load

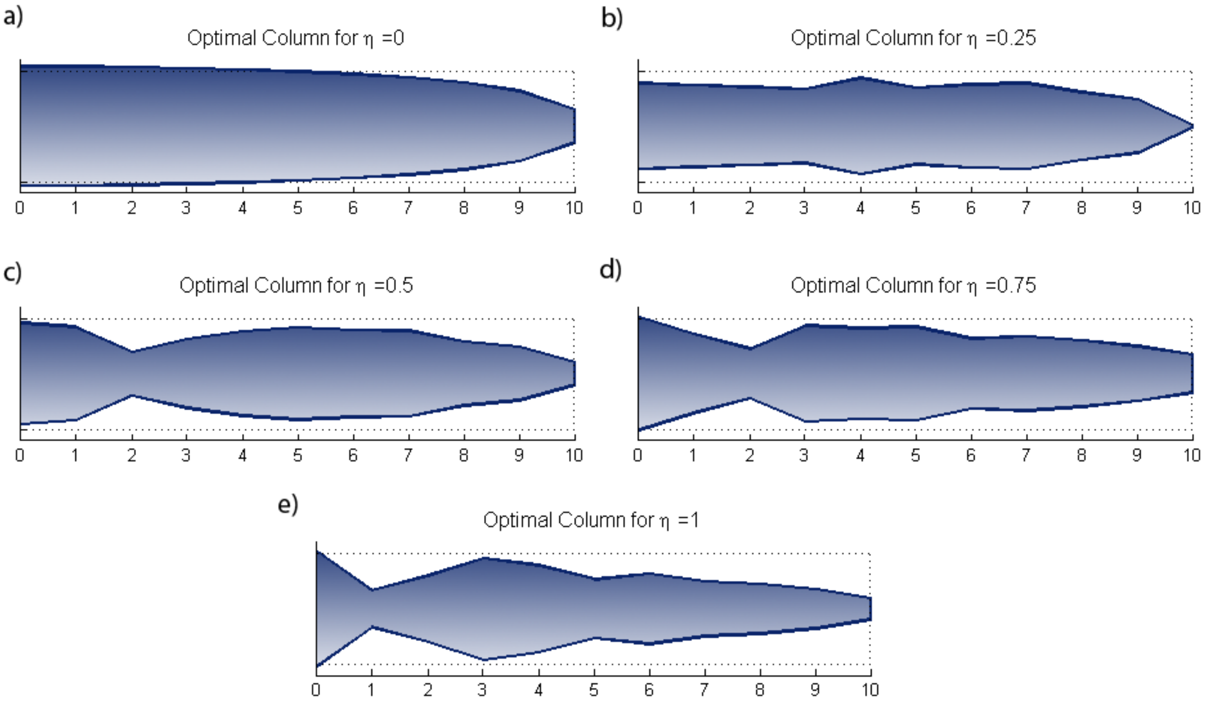


Figure 5.5: Optimized columns for various non-conservative load parameters. The dashed lines represent the uniform column.

occurs from the coalescence of the two first frequencies. Exception is made for the load condition presented in figure 5.6 *b)* where flutter occurs when the third and fourth frequencies coalesce.

Another important fact that remains to be considered is the stability of the optimized columns for load conditions other than the one used in the optimization sequence. Figure 5.7, shows the stability diagrams for each optimized column for the various load conditions. An interesting fact to notice is the jumps that occur in the critical flutter load near the corresponding optimized load conditions. These jumps result from the optimization process and are positioned to ensure that the coalescence of frequency curves occurs at a load above or equal to the critical load of the uniform column. The transition from divergence to flutter also changes significantly for the optimized columns. The optimized design for a pure conservative load, shows this transition at a load parameter of $\eta = 0.27$. In the case concerning the column optimized for a load with a 25% non-conservative component, the transition occurs at $\eta = 0.24$. The column optimized for a 50% non-conservative load changes from divergence to flutter at $\eta = 0.47$. The column optimized for a 75% non-conservative load shows the instability transition at $\eta = 0.42$. Finally, the column optimized for a pure tangential (non-conservative) load has the transition from divergence to flutter at $\eta = 0.26$.

Comparing the stability diagrams for the optimized designs from figure 5.7 with the stability diagram for the uniform column, it is possible to assert that the behavior of the optimized columns gets worse when the load parameter differs from the project value, as was to be expected. In order to obtain an improved stability for all load conditions, a new optimization formulation is required in order to control the different load parameters during the optimization process.

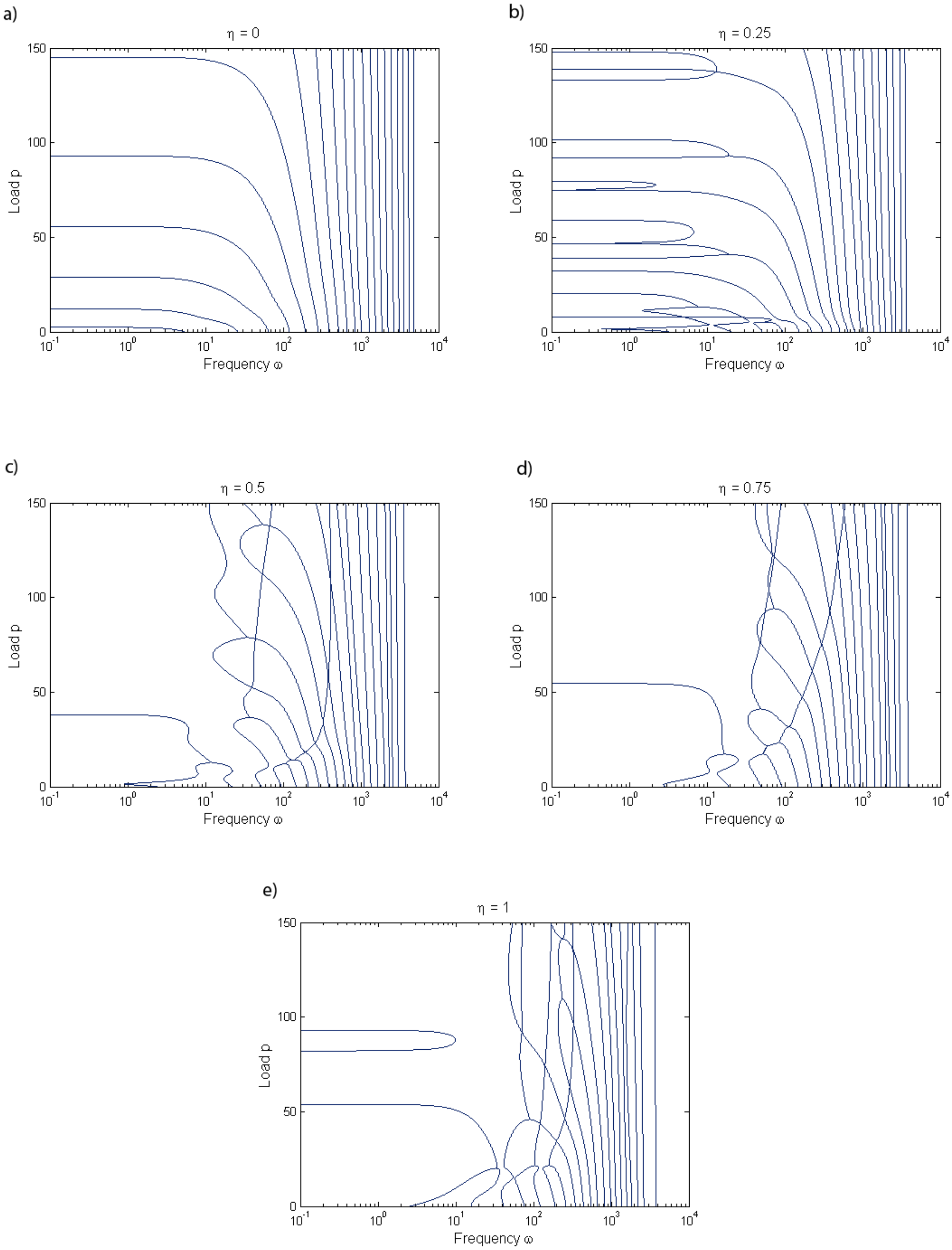


Figure 5.6: Load-Frequency curves for the optimized columns.

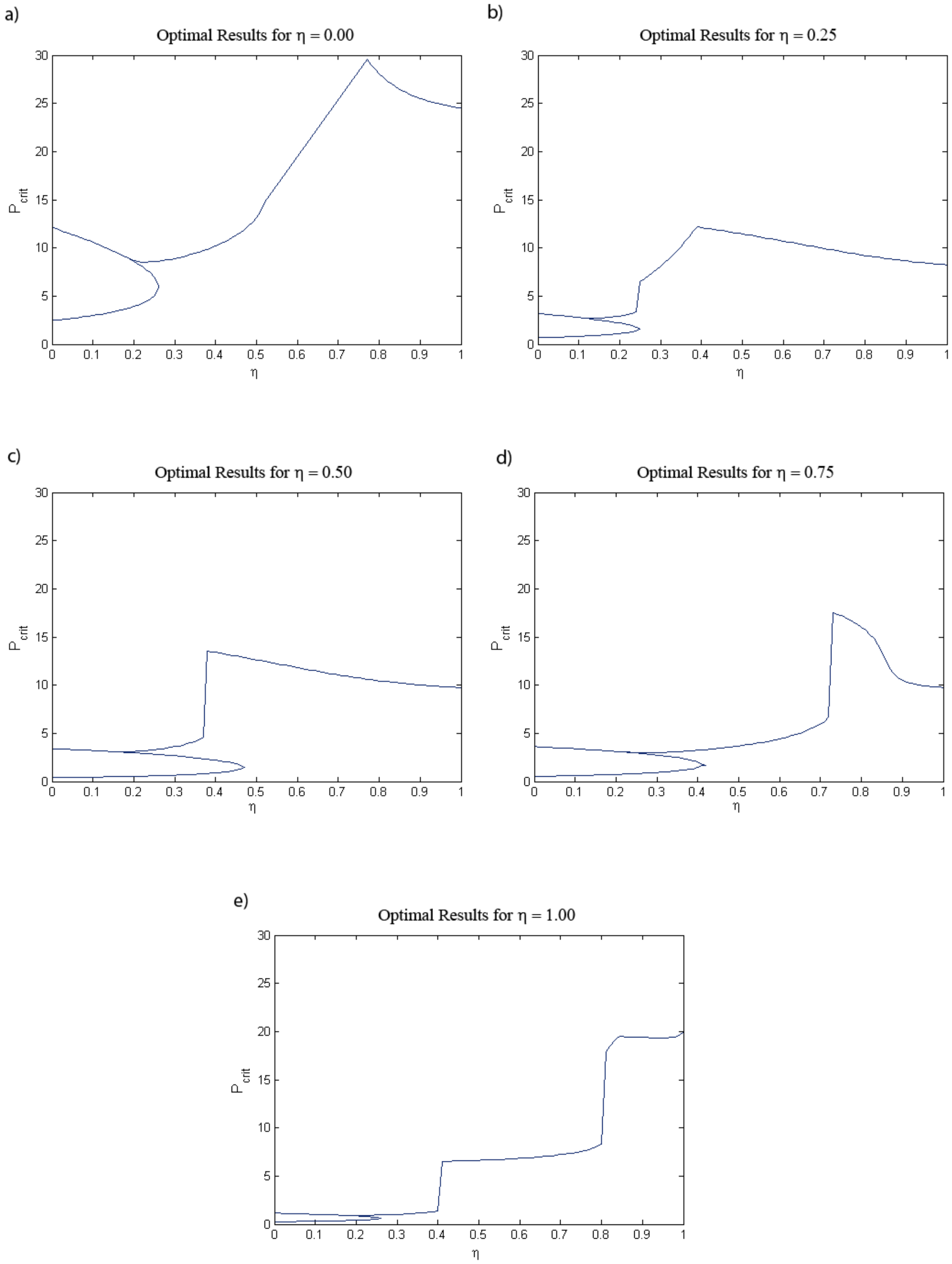


Figure 5.7: Stability diagrams of the optimized columns.

5.2 Plate Subjected to a Partial Non-Conservative End Load

5.2.1 Code Verification

As for the previous presented results, the algorithm developed for the Mindlin plate model must be verified in order to ensure that the presented stability analysis is accurate. As for the beam model, this verification was performed by comparing the natural frequency and critical load values obtained from the implemented $MATLAB^{\circledR}$ code with the results from $ANSYS^{\circledR}$. For verification purposes, a plate with $1m \times 0.25m$ and a thickness of $0.01m$ was considered, built in an aluminium alloy with a Young's modulus of $72GPa$, density of $2800Kg/m^3$ and Poisson ratio of $\nu = 0.2857$. The obtained results are presented in table 5.4:

		$MATLAB^{\circledR}$		$ANSYS^{\circledR}$		**Deviation**	
1st Frequency	(10x5) mesh	8.3217	Hz	8.3219	Hz	0.00	%
	(20x10) mesh	8.3066	Hz	8.3117	Hz	0.06	%
	(100x50) mesh	8.3013	Hz	8.3070	Hz	0.07	%
	(200x100) mesh	8.3011	Hz	8.3068	Hz	0.07	%
5th Frequency	(10x5) mesh	207.40	Hz	196.92	Hz	5.32	%
	(20x10) mesh	206.50	Hz	196.34	Hz	5.17	%
	(100x50) mesh	205.59	Hz	196.06	Hz	4.86	%
	(200x100) mesh	205.86	Hz	196.04	Hz	4.86	%
Critical Load	(10x5) mesh	15138.7	N	15110.0	N	0.19	%
	(20x10) mesh	15073.0	N	15078.0	N	0.03	%
	(100x50) mesh	15051.5	N	15065.0	N	0.09	%
	(200x100) mesh	15050.8	N	15064.0	N	0.09	%

Table 5.3: Results verification for the plate and panel flutter models.

The presented results show a good agreement between the developed and implemented numerical model and $ANSYS^{\circledR}$ for all the presented meshes. The fifth natural frequency is presented because it is the highest frequency mode considered for the flutter analysis and optimization algorithm. As can be seen, the relative error for this frequency mode is somewhat high for all the meshes, which can be justified by the nature of the Mindlin theoretical formulation and implementation. Nonetheless, the obtained value can be considered well within reason for the present study.

As for the convergence study it is possible to verify that, as the number of elements increase, the frequency and critical load values do not change considerably and when compared to the $ANSYS^{\circledR}$ results one can note that the relative error can slightly increase. This occurs because as the number of elements increase, the numerical error also increases because the size of the elements becomes too small which originates badly scaled matrices. On the other hand, the computational time required for the analysis is greatly affected by the number of used elements, going from about 10 seconds for a 10×5 mesh to around 30 minutes for a 200×100 mesh, on a standard single processor computer. Therefore, the presented results where all obtained with a 10×5 element mesh. In spite of generating good results, the relatively coarse mesh has a problem when dealing with the optimization process, because the coarse mesh may not be able to detect local vibration modes that can originate in areas of the plate where a large volume reduction occurs. This problem will be further discussed when presenting the optimization results.

As previously shown for the beam model, the sensitivities of the constraints and objective function with respect to the project variables had to be verified and, once again, the verification was performed by comparison with the finite difference method. The obtained results have shown a good agreement with minimum relative errors ranging from 10^{-4} to 10^{-6} %. Figure 5.8 presents

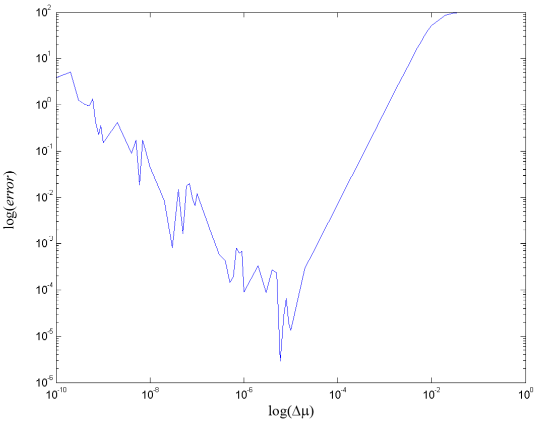


Figure 5.8: Relative error vs. $\Delta\mu$ of the sensitivity calculus of the critical load for the 1^{st} design variable and a load parameter of $\eta = 1$.

a plot with the sensitivities of the critical load of the first project variable for a η of 1 and stands as a representative plot of the general sensitivities of both the critical load and the frequencies obtained for all the project variables at different load conditions.

In addition to the presented verification methods, the stability results obtained with the plate model can also be validated by comparison with the results obtained for the beam model, presented in the previous section. This verification is possible because both models were put in similar dimensionless forms (as can be confirmed by the theoretical formulation leading to equations 2.11 for the beam model and 2.22 for the plate model) and, whenever possible, identical properties where defined for both models. A discussion and comparison of the obtained results is presented in the following section.

5.2.2 Stability Results

This section presents the results obtained for a plate subjected to a non-conservative end load as described theoretically in section 2.2 and formulated numerically in section 5.2.2. As mentioned in these sections, the presented models follow the shear deformable Mindlin theory and where obtained by a finite element discretization with 10 × 5 elements. These results were obtained for a plate with ratio between the thickness and the plate characteristic length of 0.01.

The stability diagram presented in figure 5.9 shows the evolution of the critical divergence and flutter loads with different possible load conditions ranging from ($\eta = 0$ to $\eta = 1$), as was done before. The results obtained for the plate are presented along with the stability results obtained for the beam model (presented in 5.2). Upon comparison, it is possible to verify that the results obtained for the plate model are very similar to the ones obtained for the beam model, showing basically the same stability characteristics for the different load conditions. Both critical load conditions have lower values throughout the different loadings for the plate model. This could be expected mainly because of the chosen Mindlin finite element formulation, but also because this model takes into account the shear stresses, which are neglected in the Euler-Bernoulli beam formulation.

Figure 5.10 shows the load frequency curves for the plate model. The considered non-conservative load components where the same as the ones previously considered for the beam model and the obtained results are quite similar to the ones presented for this model. The main difference that can be seen between the two plots is the presence of torsional frequency modes. As could be expected, for a thin plate under a uniform compressive load, the torsional instability loads are significantly higher than the longitudinal instability loads and, as a consequence of this, they are not present for the scale of the presented load-frequency curves. From these diagrams it is possible to confirm once again that the obtained instability loads for each value of η are lower than the ones previously obtained for the beam model, in agreement with the stability diagram from figure 5.9. The following table presents the instability loads obtained from the plate model as well as the same loads obtained for the beam model:

Load Condition	**Plate Model**		**Beam Model**		**Variation**		
$\eta = 0.00$ (fig. 5.10 a)	2.275	N	2.467	N	7.78	%	(divergence)
$\eta = 0.25$ (fig. 5.10 b)	3.363	N	3.651	N	7.89	%	(divergence)
$\eta = 0.50$ (fig. 5.10 c)	8.052	N	9.870	N	18.42	%	(divergence)
$\eta = 0.75$ (fig. 5.10 d)	16.26	N	17.2	N	5.47	%	(flutter)
$\eta = 1.00$ (fig. 5.10 e)	19.2	N	20.05	N	4.24	%	(flutter)

Table 5.4: Instability loads for the plate and beam models.

As the table shows, all the instability loads obtained from the plate model are lower than the previously obtained from the beam model as verified from simple graphical inspection. The largest difference in critical loads occurs for $\eta = 0.5$. This can be justified by the fact that this is the critical turning point between main instability modes (divergence and flutter), and thus is the most susceptible to abrupt variations in the instability load.

Although presenting similar results, there are still some main differences that need to be taken into account between the beam and plate model. The first fact that must be considered, is that the beam model has circular cross-sections and, because of this, the instability mode can occur in any direction. Since the plate model has rectangular cross sections with different area moments of inertia, the main instability mode for a uniform plate will always occur in the same direction, (the direction with the lowest area moment of inertia). This fact can be considerably useful in situations when it is desired that the instability mode should occur in a

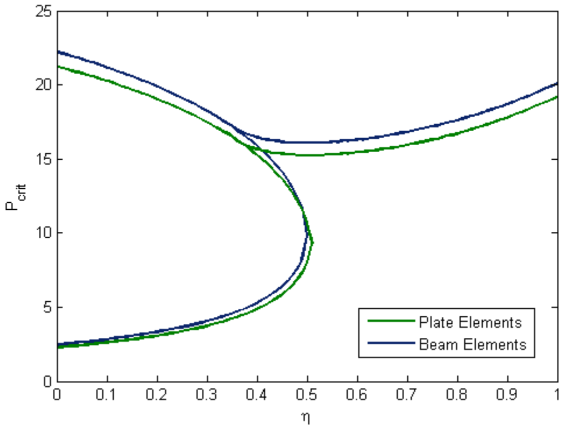


Figure 5.9: Stability diagram of the uniform plate, presenting also the results from the beam model.

specific direction. One of these situations can be the experimental verification of the presented models (details on experimental sets for columns subjected to non-conservative end loads are presented and discussed by Sugiyama et al. (2000)).

Another important fact to take into account is the presence of torsional modes, which for the uniform plate model present relatively high instability modes for the uniform plate as discussed, but may have influence in the optimization process.

Also to be taken into account is the fact that the implemented Mindlin plate model considers shear stresses along the plate while the beam model does not take into account the stresses in the columns. This fact may explain the differences in the critical loads obtained for the two models. Another fact that may influence these results is the occurrence of shear locking problems due to the nature of the implemented numerical methods as discussed in chapter . As for the beam model, the presented stability results set the basis for the optimization process that follows.

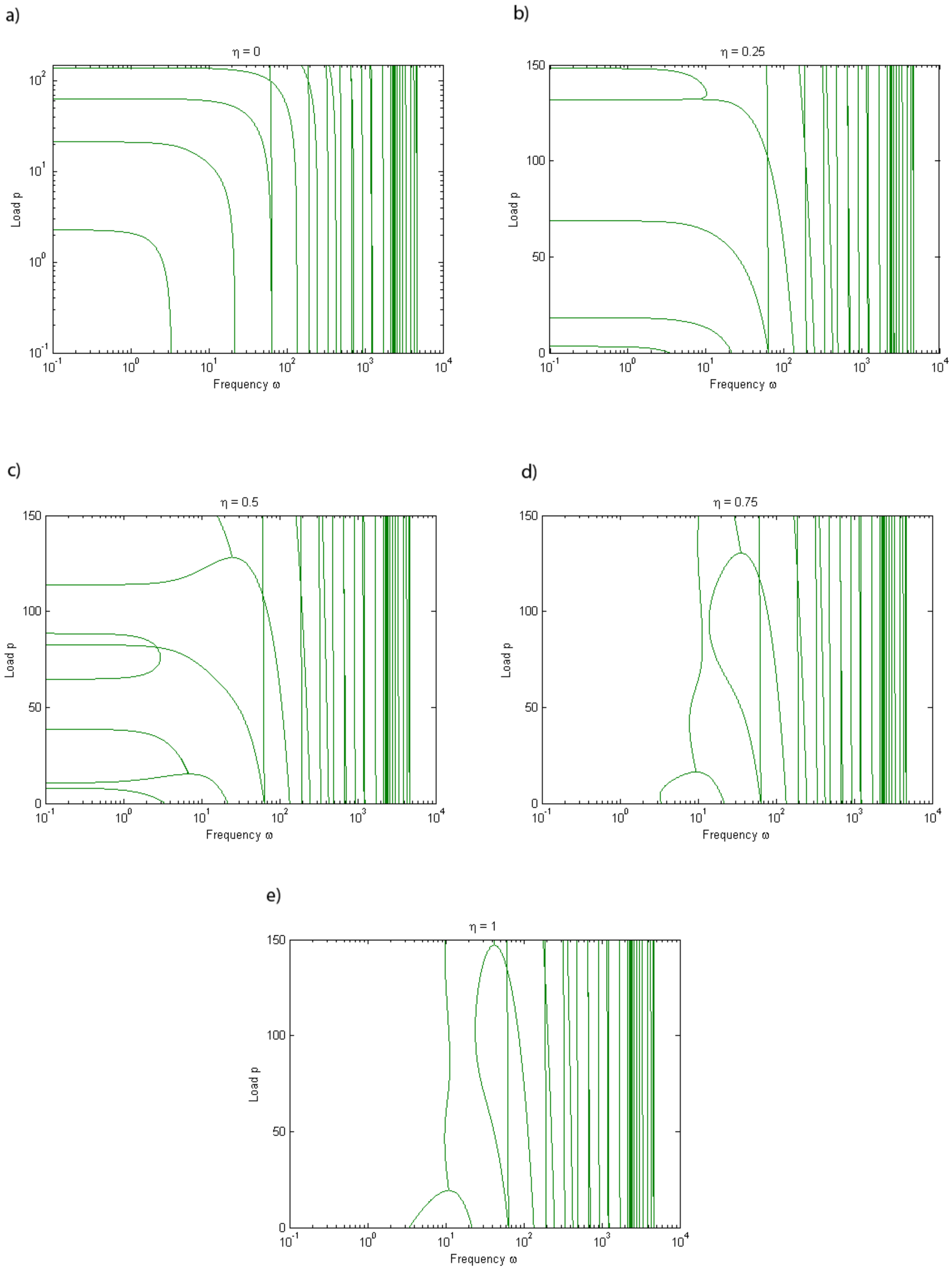


Figure 5.10: Load-Frequency curves for various load conditions, from a conservative force ($\eta = 0$) to a pure non-conservative load (Beck's column at $\eta = 1.0$). The presented results were obtained for a uniform **plate** with a dimensionless thickness of $\mu = 1.0$.

5.2.3 Optimization Results

The optimization results presented in this section where obtained starting from the uniform dimensionless plate with the stability characteristics presented in the previous section. As for the beam model, a summary of the results is presented in the next table. Since the plate model was developed so as to have similar properties as the beam model, the optimized volumes obtained from the beam analysis are also presented,

Load Condition (η)	**0.00**	**0.25**	**0.50**	**0.75**	**1.00**
Optimized Volume	0.759	0.547	0.776	0.509	0.577
Volume Reduction (%)	24.1	45.3	22.4	49.1	42.3
Beam Model Results	0.866	0.476	0.473	0.490	0.374
Number of iterations	42	32	44	46	33

Table 5.5: Optimization results for the cantilevered plate model.

In order to obtain the present results, several parameters had to be considered, as was the case for the beam model. For each load condition, the design variable limits where set as $\mu_{max} = 1.5$ and $\mu_{min} = 10^{-8}$. The slack parameter function (the same as defined for the beam model optimization) was set to,

$$\epsilon = \begin{cases} 0.3, & \text{for} \quad 0.75 < V \leq 1 \\ 0.1, & \text{for} \quad 0 < V \leq 0.75 \end{cases}$$

The choice of the slack parameter is different from the beam model, because it was verified that the optimization process for the plate elements had more difficulties in converging to lower volume values. The implied stability analysis algorithm used in the optimization process used 500 load steps and, once again, 20 load points between 0 and p_{cr}. The *MMA* tuning parameters where again upon experience, as for the previous analysis.

These results also allowed for considerable volume reductions, although the obtained volume values where higher than the presented values for the beam model. This was somewhat unexpected, but can possibly be justified by a series of reasons. First, the plate model takes into account the shear stresses on the structure and, as discussed by Langthjem and Sugiyama (2000a), this forces the thickness evolution along the plate to be smoother and, consequently inducing another limitation on the volume reduction. Another possible reason is the relatively small precision of the performed numerical calculus, both by the reduced number of elements used as well as the reduced number of load steps.

Figure 5.11 shows the evolution of the dimensionless thickness parameter. The darker tone indicated areas where material reinforcements are required, while lighter areas indicate that there is a reduced need of material in these areas. Although these material reinforcements and reductions where determined in terms of a general dimensionless thickness, they can give a qualitative idea of how the results would be in case the use of materials with different densities was considered, assuming as a simplification, a linear variation of the Young's Modulus with the density. The choice of a dimensionless thickness parameter was defined for computational simplicity reasons and to have a higher coherence with the dimensionless form used in the beam model. This fact is relevant when considering that the construction of the presented optimized plates with variable thickness is difficult and expensive, while topology optimization methods, using different materials for the design of optimal plates may prove to be more efficient. As an example, a recent study on the application of an advanced topology optimization method is presented by Gomes and Suleman (2006).

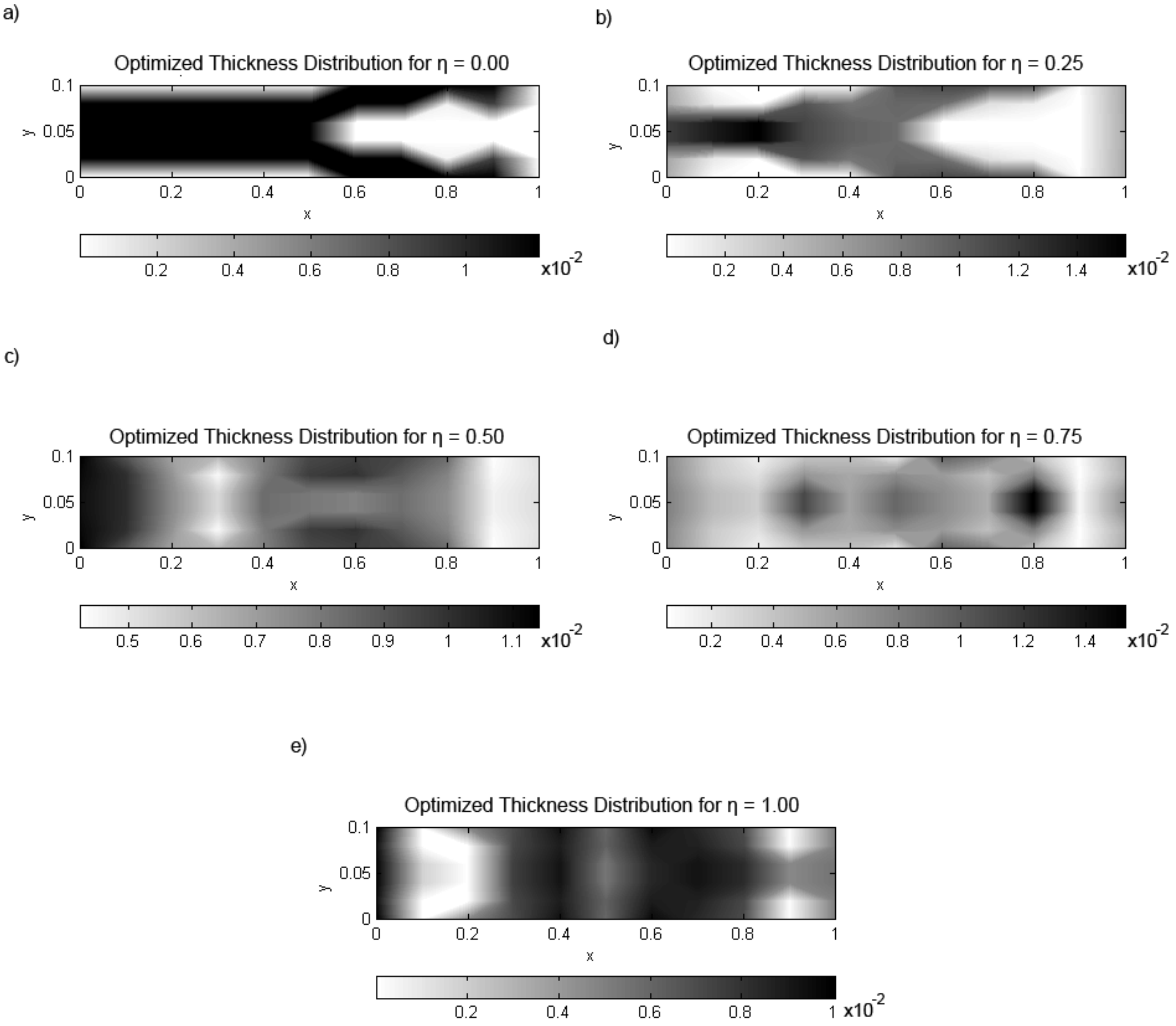


Figure 5.11: Optimized plate designs, showing the dimensionless thickness distribution for various non-conservative load parameters.

The load frequency curves obtained for the optimized columns is presented in figure 5.12. As was the case for the beam model, it is possible to confirm that the optimized columns maintain the imposed stability constraints, and the critical load is not lower than the value obtained for a uniform plate. Once again, the only case where instability occurs through divergence is for the pure conservative load ($\eta = 0.0$).

The convergence plots for each load condition are presented in figure 5.13, where the evolution of the dimensionless volume (objective function) is presented according to the number of iterations. These results show that there was a smooth convergence for all the load conditions except for $\eta = 0..75$. This fact can be justified by the different settings of the MMA tuning parameters used, which allowed for the optimization algorithm to do more abrupt changes on the design and consequently harm the convergence process. Nonetheless, even for this case there were significant volume reductions.

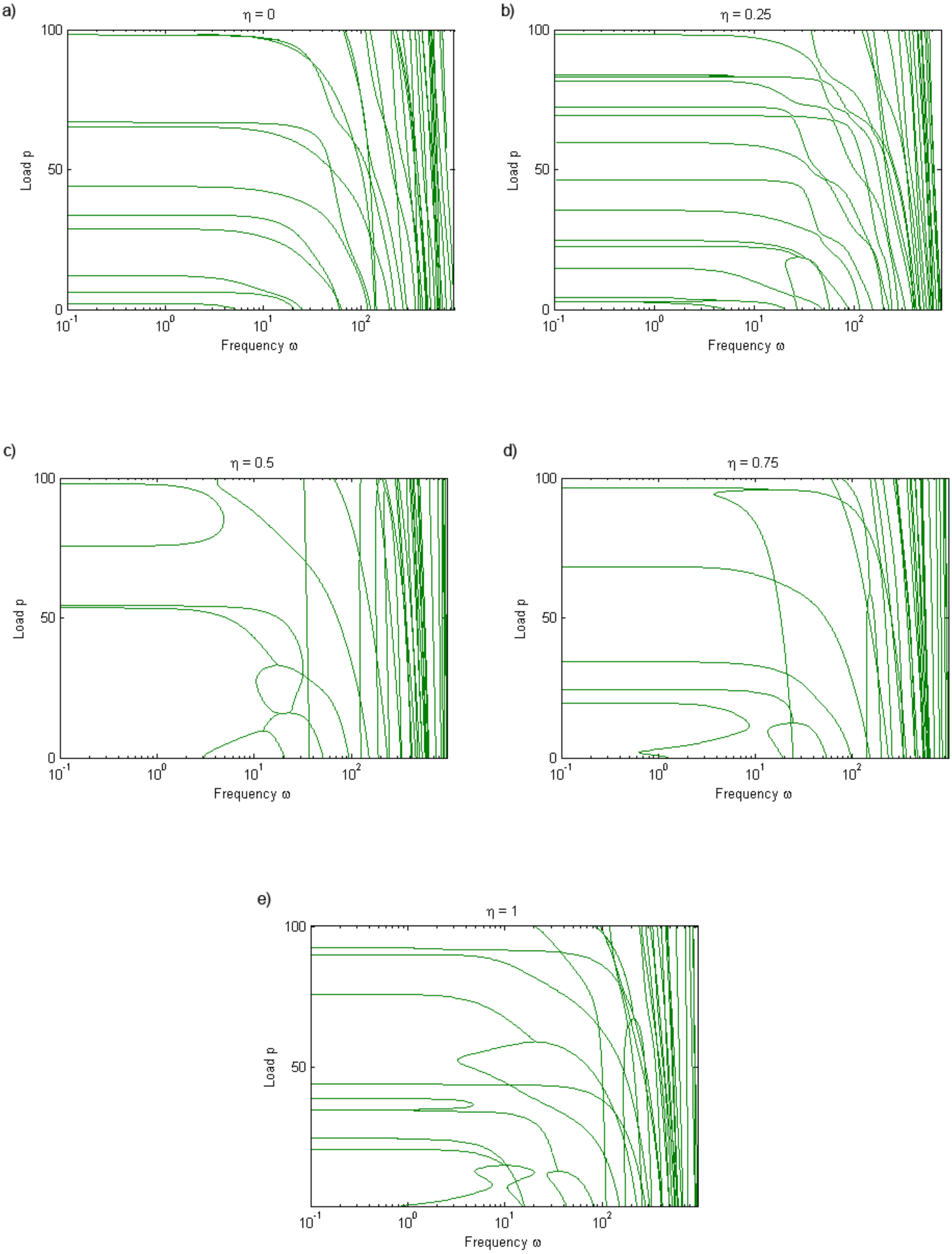


Figure 5.12: Load frequency curves for the different optimized plates.

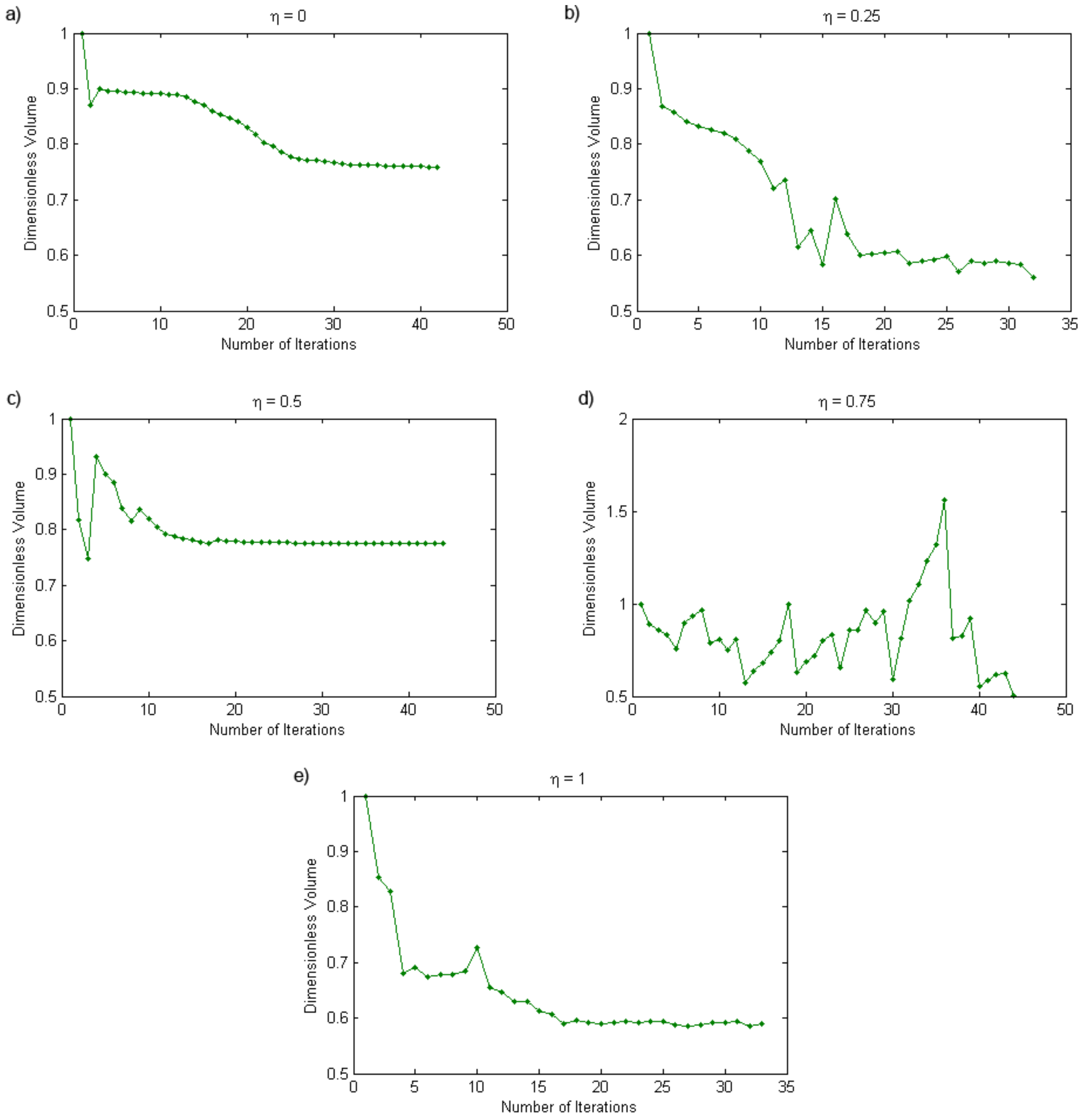


Figure 5.13: Volume variation along the optimization process.

5.3 Simply Supported Panel Under a Supersonic Flow

5.3.1 Code Verification

The finite element model used for the panel analysis now presented is based upon the same algorithm used for the previous model in which the only modifications where the geometrical dimensions and the boundary and loading conditions. As the previous analysis algorithm showed a good agreement with the finite element software, it is considered that the code is also approved for the present model.

The comparison study used to confirm the correct analytical calculations of the sensitivities of the constraint and objective functions by using finite difference methods was also implemented for the present model. The main difference between this model and the previous plate model lies in the loading conditions, which correspond to one of the optimization constraints. Given this fact some attention was given to the verification of the sensitivities of this constraint. The results have shown that, while the sensitivities for the frequencies showed relative errors with the same magnitude of the previous models $(10^{-4}to10^{-5}\%)$ as expected, the sensitivities of the critical dynamic pressure have shown an higher precision when compared to any of the results obtained for the present or previous models, with relative errors around $10^{-7}\%$. Once again the presented results show that the analytical expressions used to determine the sensitivities of the objective and constraint functions are correct.

5.3.2 Stability Analysis

The stability analysis was implemented for a simply supported square panel, with 0.3 × 0.3 meters and a thickness of $0.01m$. These values are only relevant for the relative thickness with respect to the characteristic length of the plate, since all the results are presented in a dimensionless form.

The stability analysis was developed with three different meshes to verify the convergence of the solutions. Meshes with 10 × 10, 20 × 20 and 50 × 50 elements were considered, from where it was possible to conclude that for a 10 × 10 element mesh, the critical dynamic pressure was 379, for the 20 × 20 element mesh, the critical dynamic pressure was 378 and for a 50 × 50 element mesh, the critical dynamic pressure obtained was 378. It is the possible to assume that the 10 × 10 element mesh has converged.

The instability mode originated by a low supersonic flow on the considered square panel is

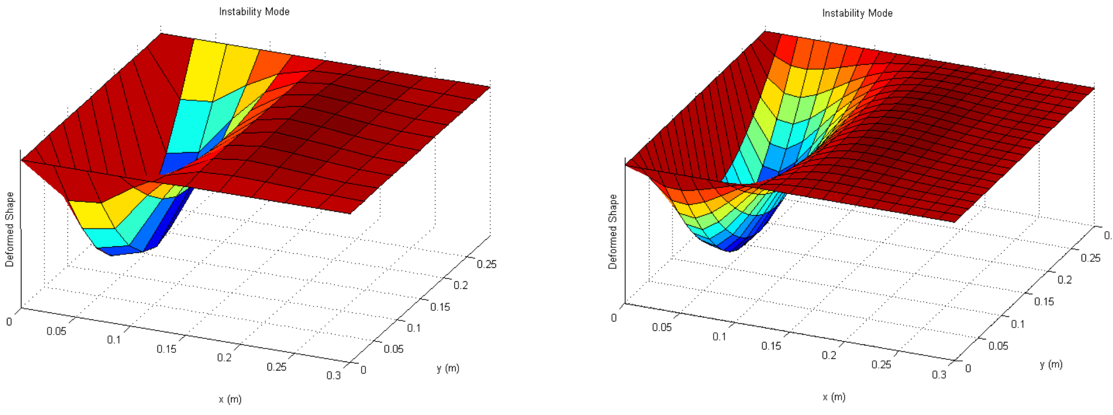


Figure 5.14: Instability mode for the panel under a low supersonic flow. The results on the left were obtained for a 10 × 10 mesh and on the right with a 20 × 20 mesh.

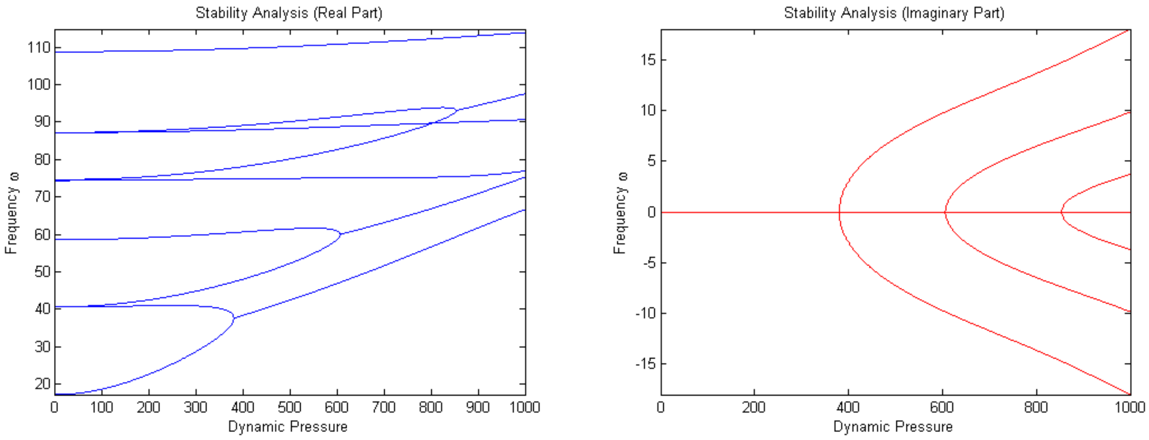


Figure 5.15: Dimensionless dynamic pressure as function of the frequency. Real (left plot) and Imaginary (right plot) representation.

presented in figure 5.14, for two different meshes. From a qualitative inspection of the deformed shapes it is also possible to confirm that the smaller mesh can produce accurate results.

The dimensionless dynamic pressure as a function of the frequency is presented in the two plots from figure 5.15, representing both the real and imaginary evolution of the critical dynamical pressure for the panel. These results where obtained with a 10 × 10 mesh, using 1000 load steps. Comparing these results with the results presented in figure 5.3, it is possible to recognize some similarities in the load frequency curves, especially in the complex domain, which reflect directly on the flutter mechanisms that originate such a structural response. When, for a certain dynamic pressure, flutter instability is reached, the load values enter the complex domain in the same way that when flutter occurred in the beam model, the frequency values presented two complex branches different from zero. These similarities help to reinforce the mentioned fact that the simple case of a beam subjected to a non-conservative end load has equivalent characteristics of more complex structural and loading models, while maintaining the simplicity required for a more straightforward comprehension of the underlying mechanisms of flutter instability problems. The comprehension of these mechanisms is essential for the development of optimization procedures as the ones described in the present work.

5.3.3 Optimization Results

The optimization process developed for the minimization of the structural weight of a panel subjected to a low supersonic flow, follows the same basic considerations of the two previously presented models and some aspects relating the optimization of structures subjected to non-conservative loads were once again confirmed.

One of these aspects is the importance of the frequency separation constraints defined in the optimization problem formulation 4.3. In order to illustrate the importance of this constraint, a first optimization procedure was considered where the only considered constraint was the dimensionless dynamic pressure, as defined by the original optimization problem. Figure 5.16 shows the evolution of the volume with the optimization iterations. These results were obtained for a 10 × 10 element mesh, with the protect variables limited between $\mu_{max} = 10$ and $\mu_{min} = 10^{-8}$. As it is possible to see in the plot, the optimization process does not converge and is constantly in a repetitive cycle. The reason why this occurs can be understood by analyzing the load frequency curves of one of these cycles.

Figure 5.17 shows the six first iterations of this optimization sequence. With the evolution of the iterations it is possible to see that the higher flutter loads become lower, until they become the dominant flutter load. When this occurs there is a large drop on the instability load and the optimization algorithm jumps back to a previous stable design and starts over again, but always with the same solution in mind which is never reached. This change in lower and higher flutter modes is the responsible for the presented lack of convergence of the optimization sequence.

By introducing some geometric constraints and project variable limitations, the problem still remains. Figure 5.18 shows the volume evolution of an optimization sequence where the project variables were limited between $\mu_{max} = 1.5$ and $\mu_{min} = 0.5$ and the slack parameter, ϵ, was set to 0.1 in all cases. Although limited on the design variables, the optimization process still presented convergence problems, having not converged to any viable solution.

The need for frequency constraints in order to obtain a convergent optimization sequence is discussed with some level of detail by Odaka and Furuya (2005). They refer to the fact that when considering frequency separation in the optimization process we have a robust optimization scheme, because by ensuring this frequency separation, the structure is optimized taking into account not only the main flutter instability mode, but also to higher modes. This robustness comes as a consequence of avoiding the mentioned convergence problems.

Considering now the frequency constraints in the optimization process it is possible to obtain

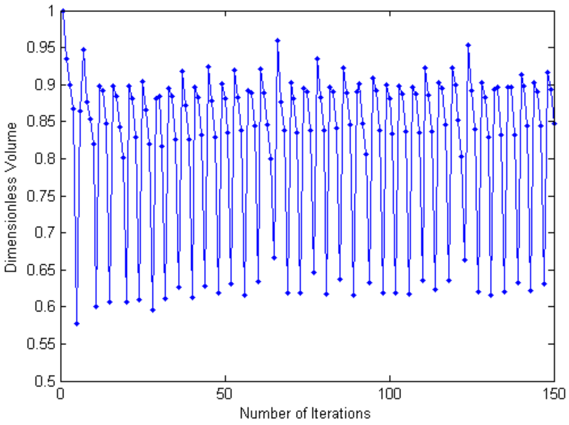


Figure 5.16: Volume evolution with the number of iterations with no constraints on the frequencies or the project variables.

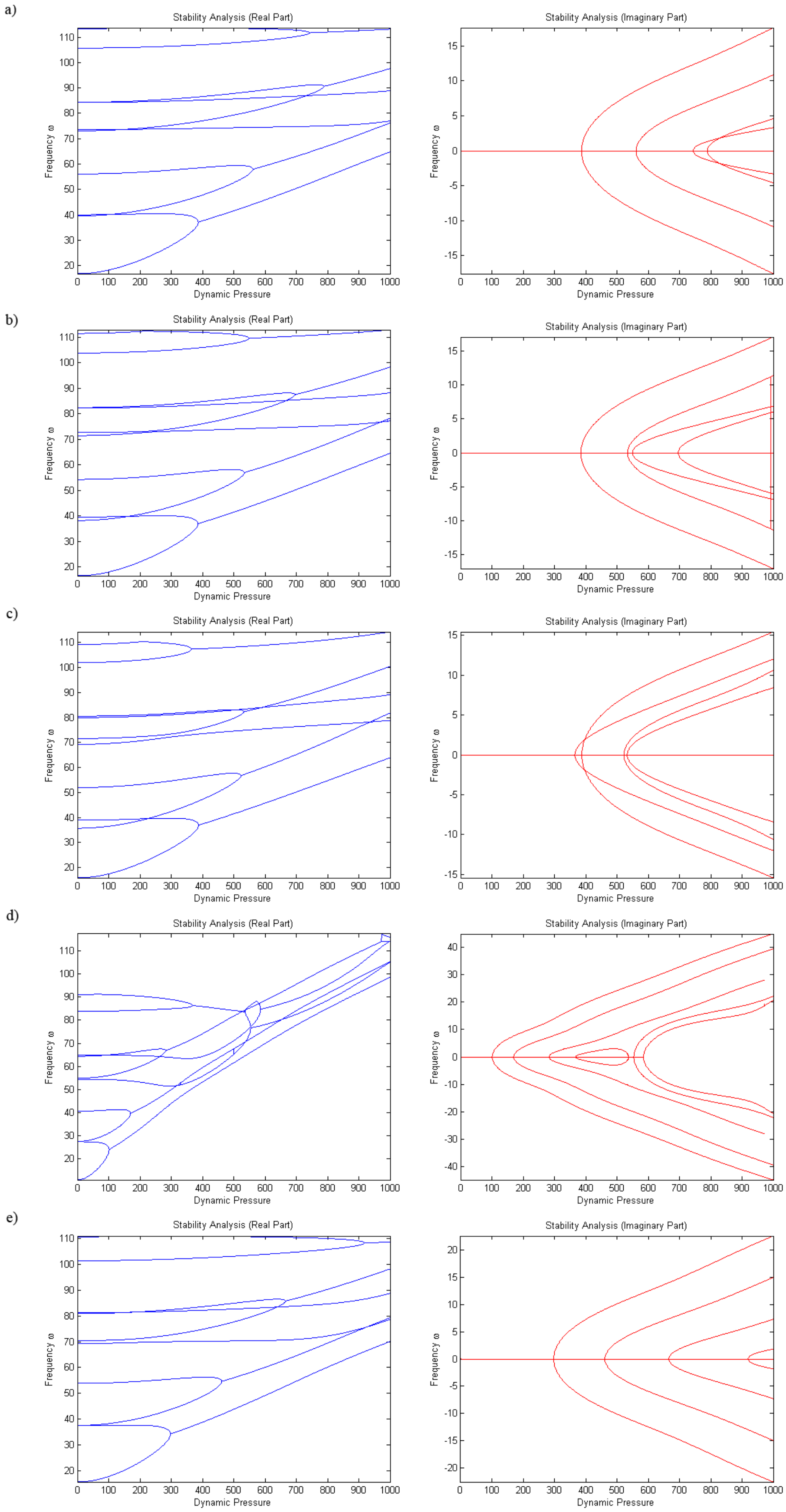


Figure 5.17: First five iterations of the optimization process with no frequency constraints.

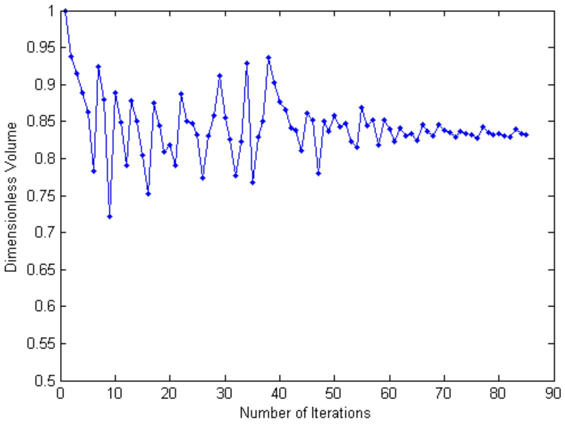


Figure 5.18: Volume evolution with the number of iterations with no constraints on the frequencies and limited design variables.

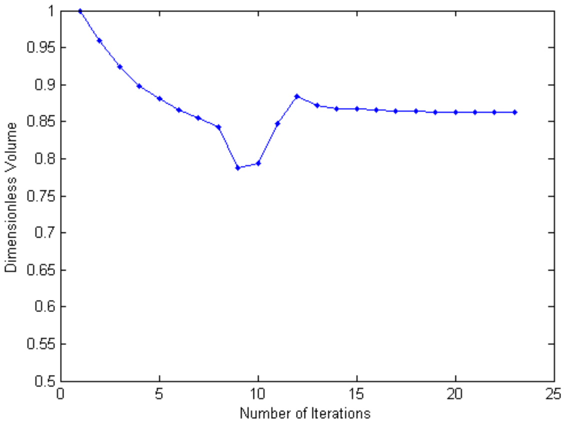


Figure 5.19: Volume evolution with the number of iterations obtained with a frequency separation constant of $c = 10$ and constraints on the project variables.

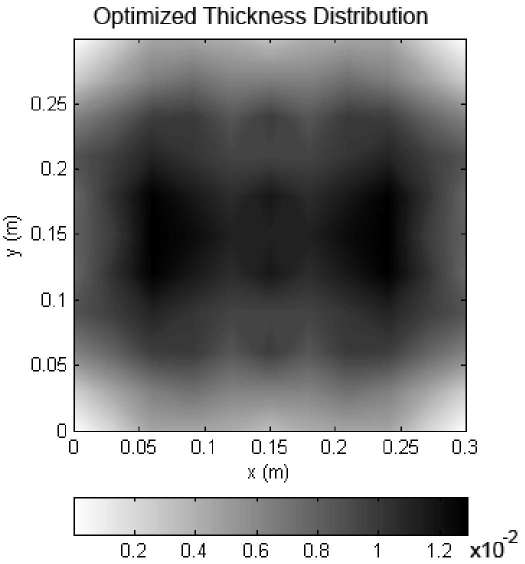


Figure 5.20: Dimensionless thickness distribution on the panel. Results obtained with a frequency separation constant of $c = 10$ and constraints on the project variables.

a good convergence of the results in a relatively reduced number of iterations. Setting the frequency separation constant as $c = 10$, it is possible to obtain a volume reduction of 13.7%, with the thickness distribution represented in figure 5.20. These results were obtained with the project variables limited between $\mu_{max} = 1.5$ and $\mu_{min} = 0.5$ and a slack parameter set as $\epsilon = 0.1$. Figure 5.19 shows the volume evolution of the solution. As it is possible to note there is a relatively smooth evolution until the convergence of the solution.

Chapter 6

Conclusions and Further Remarks

The present thesis addressed the problem of obtaining optimal designs for structures subjected to non-conservative loads. As a first stage of the work, the objective was to review the state of the art in these specific problems, implement and reproduce the results obtained in reference works on the area. On a second stage the project evolved into the extension of the problem to a plate model with thickness optimization, which was not found in the literature as far as the author's knowledge. As a final stage the main objective was the generalization of the developed plate algorithm for the panel flutter problem. The three models were implemented, tested and results were obtained.

From the first model, using beam formulation, it is possible to verify one of the basic characteristics of structural optimization problems: When a structure is optimized for a certain load condition, it's behavior when subjected to loading conditions different from the project load will be affected. A possible solution for this fact is to consider a multi purpose optimization, where constraints different constraints are simultaneously defined for different load conditions.

The implemented beam model was tested for several different load conditions, reproducing results available in the bibliography for both the stability analysis and optimization methods.

Another fact that was noted was the influence of the shape of the cross-sectional areas on the beam and plate models subjected to non-conservative end loads. As shown, for equivalent dimensionless formulations leading to equivalent dimensionless cross-sections, the critical instability loads are similar for both models. In spite of this, the structural stability of both models presents some important differences. In the beam model, with circular cross-sections, the instability mode can occur in any radial direction. For the plate model, with rectangular cross sections and one dimension much lower than the other, the main instability modes will occur mainly in the direction with the lowest area moment of inertia. This plate model also presents torsional modes that although significantly higher than the other instability modes, have shown to influence the optimization process and, consequently the obtained results.

It is also important to keep in mind that the obtained optimization results for the plate and panel flutter models are preliminary and where obtained while writing this thesis. Since the optimization process depends upon a large number of factors that must be adjusted according to the developers experience and sensitivity to the problem at hand, it is likely that further work will produce better results. Initially, the plate model with a non-conservative end load was implemented with the objective of improving the results obtained from the beam model. The author believes that this objective can still be achieved.

The main advantage of the present developed work was surely the comprehension of the mechanisms relating flutter instability, as well as the experience and knowledge acquired in optimization methods and algorithms, numerical analysis and the finite element method. Another important fact that results from this thesis is the original approach that was performed to the problem, considering the plate model for the problem of a column subjected to an non-conservative end-load, which lead to different optimization results.

One of the main advantages of starting with a simple model such as a rocket with an end tip non-conservative load, and posteriorly moving to more complex models, is that it allows for a gradual familiarization with the concepts that involve flutter and aeroelasticity. In addition, the implemented software is considered of interest for academic and industrial applications.

As further work to be developed, it is possible to point out the verification of the developed models with any available reference examples and data. The development of the analysis algorithms using different theoretical principles, such as considering the beam formulation by the Timoshenko beam and a Kirchhoff plate formulation. These verifications would ensure the robustness and accuracy of the developed models.

The development of an experimental validation model would also be of extreme value for the presented work. From this could result the development of a new finite element model or the application of different load and boundary conditions to the present models.

Another important development for the current thesis would be to study to a higher degree the influence of the user defined parameters for the optimization process in the general optimized results. These parameters could include a study on the influence of the frequency separation constant c, the global limits for the project variables μ_{min} and μ_{max}, the slack parameter definition, ϵ, the influence of the initial design on the optimization process and the MMA optimization parameters (defined in equation 4.18).

The development and implementation of different optimization methods and procedures would also be a valuable addition to the presented work. For the presented optimization procedures were set on size optimization, they can present a basic first approach to topology optimization problems or the structural optimization taking into account different load conditions, as previously mentioned.

Finally, the present work was intended from the very beginning to be a starting point for the development of a range of optimization methods for different flutter problems. For this the theoretical formulations existing in the bibliography are here presented in higher detail, as well as the numerical implementation of these equations, and the complete analysis process. From this, the starting point is set for the development and analysis of more complex structural models with different approaches to the optimization problem and better optimized results.

References

R. L. Bisplinghoff, H. Ashley, and R. L. Halfman. *Aeroelasticity.* Dover Publications, 1996.

R. Butler and J. R. Banerjee. Optimum design of bending-torsion beams with frequency or aeroelastic constraints. *Computers and Structures*, 60(5):715–724, 1996.

C. D'Angelo and C. D. Mote. Aerodynamically excited vibration and flutter of a thin disk rotating at supercritical speed. *Journal of Sound and Vibration*, 168(1):15–30, 1993.

E. H. Dowell. Panel flutter: A review of the aeroelastic stability of plates and shells. *AAIA Journal*, 8(3):385–399, 1970.

J. M. Drees. Blade twist, droop snoot, and forward spars. *Wind Technology Journal*, 1:10–16, 1978.

J. Dugundji. Personal perspective of aeroelasticity during the years 1953-1993. *Journal of Aircraft*, 40(5):809–812, 2003.

I Elishakoff. Controversy associated with the so-called 'Follower Forces': Critical overview. *Transactions of the ASME*, 58:117–142, 2005.

I. E. Garrick and W. H. (III) Reed. Historical development on aircraft flutter. *Journal of Aircraft*, 11(897-912), 1981.

A. A. Gomes and A. Suleman. Application of spectral level set methodologies in topology optimization. *Structural Multidisciplinary Optimization*, (31):430–443, 2006.

M. H. Hansen, A. Raman, and C. D. (Jr.) Mote. Estimation of non-conservative aerodynamic pressure leading to flutter of spinning disks. Technical Report 605, Danish Center For Applied Mathematics and Mechanics, 1999.

D. H. Hodges. Lateral-torsional flutter of a deep cantilever loaded by a lateral follower force at the tip. *Journal of Sound and Vibration*, 247(1):175–183, 2001.

G. Jayaraman and A. Struthers. Divergence and flutter instability of elastic specially orthotropic plates subject to follower forces. *Journal of Sound and Vibration*, 281:357–373, 2005.

J. H. Kim and J. H. Park. On the dynamic stability of rectangular plates subjected to intermediate follower forces. *Journal of Sound and Vibration*, 209(5):882–888, 1998.

J. S. Kim and H. S. Kim. A study on the dynamic stability of plates under a follower force. *Computers and Structures*, 74:351–363, 2000.

N. M. Kinkaid, O. M. O'Reilly, and P. Papadopoulos. Review - automotive disk brake squeal. *Journal of Sound and Vibration*, (267):105–166, 2003.

N. Koo, K and W. S. Hwang. Effects of hysteretic and aerodynamic damping on supersonic panel flutter of composite plates. *Journal of Sound and Vibration*, 273:569–583, 2004.

L. R. Kumar, P. K. Datta, and D. L. Prabhakara. Dynamic instability characteristics of laminated composite doubly curved panels subjected to partially distributed follower edge loading. *International Journal of Solids and Structures*, 42:2243–2264, 2005.

F.W. Lanchester. Torsional vibrations of the tail of an aeroplane. *R&M*, 276(1), 1916.

M. A. Langthjem. On the influence of damping in a problem of dynamic stability optimization. *Structural Optimization*, 7:227–236, 1994.

M. A. Langthjem and Y. Sugiyama. Optimum design of cantilevered columns under the combined action of conservative and nonconservative loads, part 1: The undamped case. *Computers and Structures*, 74:385–398, 2000a.

M. A. Langthjem and Y. Sugiyama. Optimum design of Beck's column with a constraint on the static buckling load. *Structural Optimization*, 18:228–235, 1999.

M. A. Langthjem and Y. Sugiyama. Dynamic stability of columns subjected to Follower Loads: A survey. *Journal of Sound and Vibration*, 238(5):809–851, 2000b.

M. A. Langthjem and Y. Sugiyama. Optimum design of cantilevered columns under the combined action of conservative and nonconservative loads, part 1: The undamped case. *Computers and Structures*, 74:399–408, 2000c.

M. A. Langthjem, Y. Sugiyama, and B. J. Ryu. Letters to the editor: Realistic Follower Forces. *Journal of Sound and Vibration*, 225(4):779–782, 1999.

R. B. Lehoucq, D. C. Sorensen, and C Yang. *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods.* SIAM Publications, 1998. ISBN 0-07-113210-4.

E. Livne. Future of airplane aeroelasticity. *Journal of Aircraft*, 40(6):1066–1092, 2003.

Z. Luo, J. Yang, and L. Chen. A new procedure for aerodynamic missile designs using topological optimization approach of continuum structures. *Aerospace Science and Technology*, 10:364–373, 2006.

T. Miyata. Historical view of long-span bridge aerodynamics. *Journal of Wind Engineering and Industrial Aerodynamics*, 91:1393–1410, 2003.

J. M. S. Moita, C. M. M. Soares, and Soares C. A. M. Active control of forced vibrations in adaptive structures using a higher order model. *Composite Structures*, 71(3 - 4):349–355, 2005.

M. M. Neves, Rodrigues H., and Guedes J. M. Optimal design of periodic linear elastic microstructures. *Computers and Structures*, 76(1-3):421–429, 2000.

Y. Odaka and H. Furuya. Robust structural optimization of plate wing corresponding to bifurcation in higher mode flutter. *Structural Multidisciplinary Optimization*, 30:437–446, 2005.

M. P. Païdoussis and G. X. Li. Pipes conveying fluid - a model dynamical problem. *Journal of Fluids and Structures*, 7:137–204, 1993.

P. Pedersen. *Optimal Designs - Structures and Materials - Problems and Tools.* draft, May 2003.

P. Pedersen. *Notes On Non-Conservative Dynamic Systems - Analysis, Sensitivity Analysis, Optimization.* Technical University of Denmark, 1994.

V. D. Potapov. Stability of elastic and viscoelastic plates in a gas flow taking into account shear strains. *Journal of Sound and Vibration*, 276:615–626, 2004.

J. N. Reddy. *Introduction to the finite element method.* McGraw-Hill Publishing Co., 1992. ISBN 0-07-113210-4.

A. A. Renshaw, C. D'Angelo, and C. D. (Jr.) Mote. Aerodynamically excited vibration of a rotating disk. *Journal of Sound and Vibration*, 177(5):577–590, 1994.

Y. Sugiyama, K. Katayama, K. Kiriyama, and B. J. Ryu. Experimental verification of dynamic stability of vertical cantilevered columns subjected to a sub-tangential force. *Journal of Sound and Vibration*, 236(2):193–207, 2000.

Y. Sugiyama, S. U. Ryu, and M. A. Langtjem. Beck's column as the ugly duckling. *Journal of Sound and Vibration*, 254(2):407–410, 2002.

A. Suleman and M. A. Gonçalves. Optimization issues in application of piezoelectric actuators in panel flutter control. *SPIE 4th Annual Symposium on Smart Structures and Materials*, 1997.

A. Suleman and V. B. Venkayya. Formulation of a composite panel with piezoelectric layers for application to the panel flutter problem. *IUTAM Symposium on The Active Control of Vibration*, 1994.

A. Suleman and V. B. Venkayya. Flutter control of an adaptive laminated composite panel with piezoelectric layers for application to the panel flutter problem. *AAIA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1996.

K. Svanberg. The Method of Moving Asymptotes - a new method for structural optimization. *International Journal for Numerical Methods in Engineering*, 24(2):359–373, 1987.

G. M. Van Keuren and F. E. Eastep. Use of galerkin's method for minimum weight panels with dynamic constraints. *Journal of Spacecraft and Rockets*, 14(7):414–418, 1977.

K. H. Zuo and H. L. Schreyer. Flutter and divergence instability of nonconservative beams and plates. *Int. J. Solids Structures*, 33(9):1355–1367, 1996.

Appendix A

Code Listings for the Plate Optimization Sequence

In the following pages, the listings for the algorithms used in a optimization sequence for the plate model subjected to a non-conservative end-load are presented. The codes developed for the remaining models follow the same basic structure.

A.1 Cantilevered Plate Subjected to an End Load

A.1.1 Function *flutter_ calc.m*

```
%-------------------------------------------------------------------------------
% STABILITY ANALYSIS OF A BIDIMENSIONAL PLATE (MINDLIN THEORY)
%
% Based on the adapted version by M.M. Neves of the 99 line topology
% optimization code by Ole Sigmund
%
% March 2007 - Pedro Pastilha
%-------------------------------------------------------------------------------
function [P_cr, p, w, res, set_freq] = flutter_calc(nelx, nely, ue, eta, steps, res)

tic

fprintf(1,'\n--> FLUTTER STABILITY ANALYSIS\n');

eta   = 0.00;
steps = 1000;
res   = 20;
nelx = 10;
nely = 5;
% ue(1:66) = 1;           % Project Variables


%-----------------------------------------------------------------------
%   Physical Properties:

Lx    = 1;              % Rod Lenght (m)
Ly    = 0.1;           % Rod Height (m)
h     = 0.01;           % Initial thickness

poi = 0.3;


%-----------------------------------------------------------------------
```

```
%   Finite Element data:

nodes = (nelx+1)*(nely+1);       % Number of mesh nodes
lex = Lx/nelx;
ley = Ly/nely;
x(1:1:(nelx+1)) = 0:lex:Lx;
y(1:1:(nely+1)) = 0:ley:Ly;

xn(1:1:(nelx+1)) = 0;
xn(nelx+1) = 1;


%--------------------------------------------------------------------------
%   Buckling Load Direction:

N_x  = 1;
N_y  = 0;
N_xy = 0;


%--------------------------------------------------------------------------
%   Dimensionless Problem Formulation:

L = Lx;

x  = x/L;       % Dimensionless Length
y  = y/L;       % Dimensionless Heigth
ar = L/h;       % Length/Thickness Aspect Ratio


%-------------------------------------------------------------------------
% FINITE ELEMENT MATRIX CONSTRUCTION
%-------------------------------------------------------------------------

% Definition of global matrices:

K   = sparse(3*nodes, 3*nodes);
M   = sparse(3*nodes, 3*nodes);
G_C = sparse(3*nodes, 3*nodes);
G_N = sparse(3*nodes, 3*nodes);

for ely =1:nely
    for elx =1:nelx

        ee = [ ue((nelx+1)*(ely-1)+elx),...
               ue((nelx+1)*(ely-1)+elx+1),...
               ue((nelx+1)*(ely-1)+nelx+elx+2),...
               ue((nelx+1)*(ely-1)+nelx+elx+1)];

        % Determination of Element Matrices:

        [Ke Me Ge Qe] = element_matrices_adim(elx, ely, x, y, ee, ar, poi, ...
 ..., N_x, N_y, N_xy);

        % Connectivity Relations:

        n  =  [1; 5; 9];
        n1 = [n; n+1; n+2; n+3];
        n2 = 3*((nelx+1)*(ely-1)+elx)-2;

        edof = [n2; n2+1; n2+2;                                   % node 1
                n2+3; n2+4; n2+5;                                 % node 2
```

```
              n2+3*nelx+6; n2+3*nelx+7; n2+3*nelx+8;                % node 3
              n2+3*(nelx-2)+9; n2+3*(nelx-2)+10; n2+3*(nelx-2)+11]; % node 4

       % Global Stiffness Matrix:

       K(edof,edof) = K(edof,edof) + Ke(n1,n1);

       % Global Mass Matrix:

       M(edof,edof) = M(edof,edof) + Me(n1,n1);

       % Global Conservative Stability Matrix:

       G_C(edof,edof) = G_C(edof,edof) + Ge(n1,n1);

       % Global Non Conservative Stability Matrix:

       G_N(edof,edof) = G_N(edof,edof) + Qe(n1,n1)*xn(elx+1)*(2*nely);

    end
end


% ---------------------------------------------------------------------------
% GLOBAL SOLUTION
% ---------------------------------------------------------------------------

fprintf(1,'\n   Solving');

% Displacement constraints:

fixeddofs =[(1:3*(nelx+1):3*nodes),(2:3*(nelx+1):3*nodes),...
..., (3:3*(nelx+1):3*nodes)]; %  Clamped for y=0
alldofs   = 1:3*nodes;
freedofs  = setdiff(alldofs,fixeddofs);


%---------------------------------------------------------------------------
%   Solution: (S-w^2*M-p*(Q_C-eta*Q_N) = 0)

% For given values of p, we have the following:

p_final     = 100;
n_modes     = 30;
solution    = zeros(steps,n_modes);
p(1:steps)  = 0;

% Set a progress meter:

progress = round(steps/10);


%---------------------------------------------------------------------------
%   Calculus of the critical divergence load:

[mode_p,P_d] = eig(full(K(freedofs,freedofs)),full(G_C(freedofs,freedofs)-...
     ...eta*G_N(freedofs,freedofs)));
P_d  = diag(P_d);
k = 1;
for i=1:size(P_d,1)
    if real(P_d(i)) > 0
        temp(k) = P_d(i);
        k = k+1;
    end
end
temp = sort(temp);
P_d = round(temp(1:n_modes)*10)/10;
P_f = round(P_d(1)*10)/10;


%-------------------------------------------------------------------------
%   Calculus of the critical flutter load and corresponding frequencies:

match = 0;
for i=0:1:steps

    p(i+1) = i*(p_final/steps);

    opts.disp = 0;
    [mode,lambda] = eigs(K(freedofs,freedofs)-p(i+1)*(G_C(freedofs,freedofs)-...
 ...eta*G_N(freedofs,freedofs)),M(freedofs,freedofs),n_modes,'sm',opts);

    %   Sorting of the frequencies (by the real part):

    lambda = diag(lambda);
    temp   = real(lambda);
    [temp, order] = sort(temp);
    temp(1:n_modes) = lambda(order(1:n_modes));
    lambda(1:n_modes) = temp(1:n_modes);

    solution(i+1,1:n_modes) = sqrt(lambda(1:n_modes));

    %   Determine when flutter occurs:

    for k = 1:1:(n_modes-1)
        eval = abs(imag(lambda(k) + lambda(k+1)));
        if imag(lambda(k)) ~= 0 && eval == 0 && match == 0
            P_f = p(i);
            match = 1;
            set_freq = k;
        end
    end

    if i == progress
        fprintf(1,'.');
        progress = progress + round(steps/10);
    end
end

%   The critical load is given by:

if abs(imag(P_d(1))) == 0 && real(P_d(1)) > 0
    P_cr = min(real(P_d(1)),real(P_f));
    set_freq = 1;
else
    P_cr = real(P_f);
end

fprintf(1,'\n\n   The first critical load for the given column is %2.4f\n',P_cr);


%   Definition of the optimization constraint points (w and p vectors):

dim = ceil((P_cr/p_final)*steps);
```

```
p = p(1:1:dim);
w = solution(1:1:dim,:);
p_new(1:res) = 0;
w_new = zeros(res,n_modes);

if res > dim
    res = dim;
    p_new = p;
    w_new = w;
else
    for  i=2:1:(res-1)
        p_new(i) = p(ceil(dim*((i-1)/(res-1))));
        w_new(i,:) = w(ceil(dim*((i-1)/(res-1))),:);
    end
    p_new(1) = p(1);
    p_new(res) = p(dim);
    w_new(1,:) = w(1,:);
    w_new(res,:) = w(dim,:);
end
p = p_new;
w = w_new;


% --------------------------------------------------------------------------
% POST PROCESSING:
% --------------------------------------------------------------------------

figure(1)
semilogx(solution,0:(p_final/steps):p_final,'b');
axis tight
title(['\eta = ', num2str(eta)]);
xlabel('Frequency \omega');
ylabel('Load p');


%   Finish the time counter:

time = toc;

fprintf(1,'\n(%2.4f seconds)\n\n',time);

end
```

A.1.2 Function *element_ matrices_ adim.m*

```
%-------------------------------------------------------------------------------------------
%       FINITE ELEMENT MATRIX CALCULATION FOR THE
%                 FIRST ORDER PLATE THEORY
%
% March 2007 - Pedro Pastilha
%-------------------------------------------------------------------------------------------
function [Ke Me Ge Qe] = element_matrices_adim(elx, ely, x, y, ee, ar, poi, N_x, N_y, N_xy)

%-------------------------------------------------------------------------------------------
%   INPUT VARIABLES (for testing purposes only):

% % FUNCTION INPUTS
%
% elx = 1;
% ely = 1;
%
% rho  = 7800;           % Material density (Kg/m^3)
% E   = 12*rho^2;        % Young's modulus (Pa)
% poi = 0.3;
% h   = 0.005;             % Thickness (m)
%
% N_x  = 1;
% N_y  = 1;
% N_xy = 1;
%
% % NON FUNCTION INPUTS:
%
% nelx = 4;
% nely = 4;
% Lx = 2;
% Ly = 2;
%
% lex  = Lx/nelx;
% ley  = Ly/nely;
% x(1:1:(nelx+1)) = (0:lex:Lx);
% y(1:1:(nely+1)) = (0:ley:Ly);
% l    = Lx;


%------------------------------------------------------------------------------------------
%   LOCAL VARIABLES:

mat_output = 0;         % 0/1 - Hide/Show Element Matrices

xe  = [x(elx), x(elx+1), x(elx+1), x(elx)];
ye  = [y(ely), y(ely), y(ely+1), y(ely+1)];

%------------------------------------------------------------------------------------------
%   MATERIAL PROPERTIES:

% Dimensionless Plate rigidities (Isotropic):

k = 5/6;            % Shear correction coefficient

r = (6*k*(1-poi));              % ISOTROPIC MATERIAL!!!!

D_11 = 1;
D_22 = 1;
D_12 = poi;
D_66 = (1-poi)/2;
A_44 = r*ar;
A_55 = r*ar;

% Dimensionless Moments of Inertia:

I_0 = 1;
I_2 = (1/12)*(1/ar)^2;


%------------------------------------------------------------------------------------------
%   CONSTRUCTION OF THE ELEMENT MATRICES (NUMERICAL):

GP = [0, -sqrt(1/3), sqrt(1/3)];   % Gauss integration points
GW = [2, 1, 1];                    % Gauss integraion weights

K_11 = zeros(4);
K_12 = zeros(4);
K_13 = zeros(4);
K_22 = zeros(4);
```

```
K_23 = zeros(4);
K_33 = zeros(4);
M_11 = zeros(4);
M_22 = zeros(4);
G    = zeros(4);
Q    = zeros(4);


for i=1:1:4
    for j=1:1:4

        K_11s = 0;
        K_12s = 0;
        K_13s = 0;
        K_22s = 0;
        K_33s = 0;

        K_22n = 0;
        K_23n = 0;
        K_33n = 0;
        M_11n = 0;
        M_22n = 0;
        Gn    = 0;
        Qn    = 0;

        %   Numerical integration for Gauss Quadrature Rule (1x1 Points)
        %       -> Shear Contribution for the Stiffness Matrix

        eta = GP(1);
        ksi = GP(1);

        % Calculate the shape functions at the selected nodes:

        [me psi psi_ksi psi_eta J J_inv Jacobian] = shape_functions(eta,ksi,xe,ye,ee);

        K_11s = K_11s + (me*A_55*(J_inv(1,1)*psi_ksi(i) + ...
 J_inv(1,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + J_inv(1,2)*psi_eta(j)) + ...
                         me*A_44*(J_inv(2,1)*psi_ksi(i) + ...
 J_inv(2,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
 J_inv(2,2)*psi_eta(j)))*Jacobian*GW(1)*GW(1);

        K_12s = K_12s + (me*A_55*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*psi(j))*Jacobian*GW(1)*GW(1);

        K_13s = K_13s + (me*A_44*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*psi(j))*Jacobian*GW(1)*GW(1);

        K_22s = K_22s + (me*A_55*psi(i)*psi(j))*Jacobian*GW(1)*GW(1);

        K_33s = K_33s + (me*A_44*psi(i)*psi(j))*Jacobian*GW(1)*GW(1);

        %   Numerical integration for Gauss Quadrature Rule (2x2 Points)
        %       -> Bending and Membrane Contribution for the Stiffness Matrix
        %       -> Mass Matrix
        %       -> Stability Matrix

        for n=2:1:3

            eta = GP(n);

%             % Calculate the shape functions at the selected nodes:
%
```

```
%             [me psi psi_ksi psi_eta J J_inv Jacobian] = shape_functions(eta,1,xe,ye,ee);
%
%             % Element Non Conservative Matrix Coefficients
%
%             Qn = Qn + ((J_inv(1,1)*psi_ksi(i) + J_inv(1,2)*psi_eta(i))*psi(j))*Jacobian*GW(n);

            for m=2:1:3

               ksi = GP(m);

               % Calculate the shape functions at the selected nodes:

               [me psi psi_ksi psi_eta J J_inv Jacobian] = shape_functions(eta,ksi,xe,ye,ee);

               % Element Stiffness Matrix Coefficients:

               K_22n = K_22n + (me^3*D_11*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + ...
J_inv(1,2)*psi_eta(j)) + ...
                                me^3*D_66*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
J_inv(2,2)*psi_eta(j)))*Jacobian*GW(n)*GW(m);

               K_23n = K_23n + (me^3*D_12*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
J_inv(2,2)*psi_eta(j)) + ...
                                me^3*D_66*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + ...
J_inv(1,2)*psi_eta(j)))*Jacobian*GW(n)*GW(m);

               K_33n = K_33n + (me^3*D_66*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + J_inv(1,2)*psi_eta(j)) + ...
                                me^3*D_22*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
J_inv(2,2)*psi_eta(j)))*Jacobian*GW(n)*GW(m);

               % Element Mass Matrix Coefficients:

               M_11n = M_11n + me*I_0*psi(i)*psi(j)*Jacobian*GW(n)*GW(m);

               M_22n = M_22n + me^3*I_2*psi(i)*psi(j)*Jacobian*GW(n)*GW(m);

               % Element Stability Matrix Coefficients:

               Gn = Gn + (N_x*(J_inv(1,1)*psi_ksi(i) + J_inv(1,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + ...
 J_inv(1,2)*psi_eta(j)) + N_y*(J_inv(2,1)*psi_ksi(i) + ...
 J_inv(2,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + J_inv(2,2)*psi_eta(j)) +...
 N_xy*((J_inv(1,1)*psi_ksi(i) + J_inv(1,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
 J_inv(2,2)*psi_eta(j)) + (J_inv(2,1)*psi_ksi(i) + ...
 J_inv(2,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + ...
 J_inv(1,2)*psi_eta(j))))*Jacobian*GW(n)*GW(m);

               % Element Non Conservative Matrix Coefficients

               Qn = Qn + psi(i)*(J_inv(1,1)*psi_ksi(j) + J_inv(1,2)*psi_eta(j))*Jacobian*GW(n)*GW(m);

            end
        end

        % Matrix Entries:

        K_11(i,j) = K_11s;
        K_12(i,j) = K_12s;
        K_13(i,j) = K_13s;
        K_22(i,j) = K_22n + K_22s;
        K_23(i,j) = K_23n;
        K_33(i,j) = K_33n + K_33s;
        M_11(i,j) = M_11n;
        M_22(i,j) = M_22n;
        G(i,j)    = Gn;
        Q(i,j)    = Qn;
    end
end

null_mat = zeros(4);

% Element Stiffness Matrix:

Ke_upper = [ K_11,     K_12,     K_13;
             null_mat, K_22,     K_23;
             null_mat, null_mat, K_33 ];

Ke_upper = triu(Ke_upper);
Ke_lower = transpose(Ke_upper) - diag(diag(Ke_upper));
Ke = Ke_upper + Ke_lower;

% Element Mass Matrix:

Me_upper = [M_11,     null_mat, null_mat;
            null_mat, M_22,     null_mat;
            null_mat, null_mat, M_22   ];

Me_upper = triu(Me_upper);
Me_lower = transpose(Me_upper) - diag(diag(Me_upper));
Me = Me_upper + Me_lower;

% Element Stability Matrix:

Ge = [G,        null_mat, null_mat;
      null_mat, null_mat, null_mat;
      null_mat, null_mat, null_mat];

% Element Non-Conservative Matrix:

Qe = [Q,        null_mat, null_mat;
      null_mat, null_mat, null_mat;
      null_mat, null_mat, null_mat];

%-------------------------------------------------------------------------------------------
%   Matrix Printout:

if mat_output == 1

fprintf(1,'\nElement Stiffness Matrix (symmetric):\n');

pretty(Ke);

fprintf(1,'\nElement Mass Matrix (symmetric):\n');

pretty(Me);

fprintf(1,'\nElement Stability Matrix (symmetric):\n');

pretty(Ge);
```

```
end

end

%   Lagrange Interpolation Functions:
%
%   (Rectangular Element With 4 Nodes)
%
%                ^ eta
%        4       |       3
%      ^  +------|------+
%      |  |      |      |     ksi
%   2b |  |      +------+--->
%      |  |             |
%      v  +-------------+
%         1              2
%         <------------->
%
%   psi_i = (1/4)*(1 + ksi*ksi_i)*(1 + eta*eta_i)
%   (i = node 1,2,...)

function [me psi psi_ksi psi_eta J J_inv Jacobian] = shape_functions(eta, ksi, xe, ye, ee)

% Thickness distribution along the element:

me = (1/4)*(1-ksi)*(1-eta)*ee(1) + (1/4)*(1 + ksi)*(1 - eta)*ee(2) + ...
     (1/4)*(1 + ksi)*(1 + eta)*ee(3) +  (1/4)*(1 - ksi)*(1 + eta)*ee(4);

% Vector definition with the interpolation function at each node:

psi = [(1/4)*(1 - ksi)*(1 - eta) ;
       (1/4)*(1 + ksi)*(1 - eta) ;
       (1/4)*(1 + ksi)*(1 + eta) ;
       (1/4)*(1 - ksi)*(1 + eta)];

% Derivatives of the interpolation function at each node:

psi_ksi = [-(1/4)*(1 - eta) ;
            (1/4)*(1 - eta) ;
            (1/4)*(1 + eta) ;
           -(1/4)*(1 + eta)];
psi_eta = [-(1/4)*(1 - ksi) ;
           -(1/4)*(1 + ksi) ;
            (1/4)*(1 + ksi) ;
            (1/4)*(1 - ksi)];

% Compute the Jacobian Matrix:

J = transpose([xe; ye]*[psi_ksi, psi_eta]);     % Jacobian Matrix

Jacobian = det(J);                              % Jacobian

if Jacobian <= 0
    errordlg('Oh boy, oh boy! The determinant has become zero...RUN FOR YOUR LIVES!!!!')
    error('My existance has lost all meaning...The determinant has become zero');
end


J_inv = inv(J);                                 % Inverse of the Jacobian Matrix

end
```

A.1.3 Function *sensitivity.m*

```
%--------------------------------------------------------------------------------------------
% SENSITIVITY ANALISYS FOR FLUTTER OPTIMIZATION
%
% optimization of flutter instability using flutter_calc.m to determine the
% constraints and volume.m as the objective function
%
% September 2006 - Pedro Pastilha
%--------------------------------------------------------------------------------------------

function [d_p d_w] = sensitivity(nelx, nely, ue, eta, p, P_cr, res, set_freq)

fprintf(1,'\n--> SENSITIVITY ANALYSIS\n');

tic

% eta   = 0.0;
% res   = 5;
% nelx = 10;
% nely = 4;
% p = [ 0    0.4000    1.0000    1.6000    2.2000];
% set_freq = 1;
% P_cr = 2.2547;
% ue(1:(nelx+1)*(nely+1)) = 1;           % Project Variables

%--------------------------------------------------------------------------------------------
%   Physical Properties:

Lx   = 1;             % Rod Lenght (m)
Ly   = 0.1;           % Rod Height (m)
h    = 0.01;          % Initial Thickness (m)

poi = 0.3;

nfreqs = 10;         % Number of frequencies to consider


%--------------------------------------------------------------------------------------------
%   Finite Element data:

nodes = (nelx+1)*(nely+1);       % Number of mesh nodes
lex = Lx/nelx;
ley = Ly/nely;
x(1:1:(nelx+1)) = 0:lex:Lx;
y(1:1:(nely+1)) = 0:ley:Ly;

xn(1:1:(nelx+1)) = 0;
xn(nelx+1) = 1;


%--------------------------------------------------------------------------------------------
%   Buckling Load Direction:

N_x  = 1;
N_y  = 0;
N_xy = 0;


%--------------------------------------------------------------------------------------------
%   Dimensionless Problem Formulation:
```

```
L = Lx;

x  = x/L;       % Dimensionless Length
y  = y/L;       % Dimensionless Heigth
ar = L/h;       % Length/Thickness Aspect Ratio


%------------------------------------------------------------------------------------------
% SENSITIVITY ANALYSIS:
%------------------------------------------------------------------------------------------

% Array preallocation (for speed):

d_w = zeros(res,nodes,nfreqs);
d_p(1:nodes) = 0;

fprintf(1,'\n   Calculating');

progress = round(nodes/10);

for e=1:1:nodes

%------------------------------------------------------------------------------------------
    % FINITE ELEMENT MATRIX CONSTRUCTION

    % Definition of global matrices:

    K   = sparse(3*nodes, 3*nodes);
    M   = sparse(3*nodes, 3*nodes);
    G_C = sparse(3*nodes, 3*nodes);
    G_N = sparse(3*nodes, 3*nodes);

    for ely =1:nely
        for elx =1:nelx

            ee = [ ue((nelx+1)*(ely-1)+elx),...
                   ue((nelx+1)*(ely-1)+elx+1),...
                   ue((nelx+1)*(ely-1)+nelx+elx+2),...
                   ue((nelx+1)*(ely-1)+nelx+elx+1)];

            % Determination of Element Matrices:

            [Ke Me Ge Qe] = element_matrices_adim(elx, ely, x, y, ee, ar, poi, N_x, N_y, N_xy);

            % Connectivity Relations:

            n  =  [1; 5; 9];
            n1 = [n; n+1; n+2; n+3];
            n2 = 3*((nelx+1)*(ely-1)+elx)-2;

            edof = [n2; n2+1; n2+2;                                      % node 1
                    n2+3; n2+4; n2+5;                                    % node 2
                    n2+3*nelx+6; n2+3*nelx+7; n2+3*nelx+8;               % node 3
                    n2+3*(nelx-2)+9; n2+3*(nelx-2)+10; n2+3*(nelx-2)+11]; % node 4

             % Global Stiffness Matrix:

             K(edof,edof) = K(edof,edof) + Ke(n1,n1);

             % Global Mass Matrix:

             M(edof,edof) = M(edof,edof) + Me(n1,n1);
             % Global Conservative Stability Matrix:

             G_C(edof,edof) = G_C(edof,edof) + Ge(n1,n1);

             % Global Non Conservative Stability Matrix:

             G_N(edof,edof) = G_N(edof,edof) + Qe(n1,n1)*xn(elx+1)*(2*nely);

        end
    end


%------------------------------------------------------------------------------------------
    % DERIVATIVE FINITE ELEMENT MATRIX CONSTRUCTION

    % Definition of global matrices:

    d_K   = sparse(3*nodes, 3*nodes);
    d_M   = sparse(3*nodes, 3*nodes);
    d_G_C = sparse(3*nodes, 3*nodes);
    d_G_N = sparse(3*nodes, 3*nodes);

    for ely =1:nely
        for elx =1:nelx

            ee = [ ue((nelx+1)*(ely-1)+elx),...
                   ue((nelx+1)*(ely-1)+elx+1),...
                   ue((nelx+1)*(ely-1)+nelx+elx+2),...
                   ue((nelx+1)*(ely-1)+nelx+elx+1)];

           % Determination of the Derivative Element Matrices:

           if     ((nelx+1)*(ely-1)+elx) == e
                lnode = 1;
                [d_Ke d_Me d_Ge d_Qe] = element_matrices_diff(elx, ely, lnode, x, y,...
 ee, ar, poi, N_x, N_y, N_xy);

            elseif ((nelx+1)*(ely-1)+elx+1) == e
                lnode = 2;
                [d_Ke d_Me d_Ge d_Qe] = element_matrices_diff(elx, ely, lnode, x, y,...
 ee, ar, poi, N_x, N_y, N_xy);

            elseif ((nelx+1)*(ely-1)+nelx+elx+2) == e
                lnode = 3;
                [d_Ke d_Me d_Ge d_Qe] = element_matrices_diff(elx, ely, lnode, x, y,...
 ee, ar, poi, N_x, N_y, N_xy);

            elseif ((nelx+1)*(ely-1)+nelx+elx+1) == e
                lnode = 4;
                [d_Ke d_Me d_Ge d_Qe] = element_matrices_diff(elx, ely, lnode, x, y,
 ee, ar, poi, N_x, N_y, N_xy);

           else
                d_Ke = zeros(12);
                d_Me = zeros(12);
                d_Ge = zeros(12);
                d_Qe = zeros(12);
           end

            % Connectivity Relations:
```

```
            n  =  [1; 5; 9];
            n1 = [n; n+1; n+2; n+3];
            n2 = 3*((nelx+1)*(ely-1)+elx)-2;

            edof = [n2; n2+1; n2+2;                                      % node 1
                    n2+3; n2+4; n2+5;                                    % node 2
                    n2+3*nelx+6; n2+3*nelx+7; n2+3*nelx+8;               % node 3
                    n2+3*(nelx-2)+9; n2+3*(nelx-2)+10; n2+3*(nelx-2)+11]; % node 4

             % Global Stiffness Matrix:

             d_K(edof,edof) = d_K(edof,edof) + d_Ke(n1,n1);

             % Global Mass Matrix:

             d_M(edof,edof) = d_M(edof,edof) + d_Me(n1,n1);

             % Global Conservative Stability Matrix:

             d_G_C(edof,edof) = d_G_C(edof,edof) + d_Ge(n1,n1);

             % Global Non Conservative Stability Matrix:

             d_G_N(edof,edof) = d_G_N(edof,edof) + d_Qe(n1,n1)*xn(elx+1)*(2*nely);

        end
    end


%------------------------------------------------------------------------------------------
    %   CURRENT VARIABLE SENSITIVITY CALCULATION:

    % Displacement constraints:

    fixeddofs = [(1:3*(nelx+1):3*nodes),(2:3*(nelx+1):3*nodes),...
(3:3*(nelx+1):3*nodes)]; %  Clamped for y=0
    alldofs   = 1:3*nodes;
    freedofs  = setdiff(alldofs,fixeddofs);
    n_modes   = 30;


%------------------------------------------------------------------------------------------
    %   Sensitivities of the frequencies

    for i=1:1:res


        %   Stability solution to determine the critical load and right eigenvector:

        opts.disp = 0;
        [mode,lambda] = eigs((K(freedofs,freedofs)-p(i)*(G_C(freedofs,freedofs)-...
 eta*G_N(freedofs,freedofs))),(M(freedofs,freedofs)),n_modes,'sm',opts);

        lambda = diag(lambda);
        temp   = real(lambda);
        [temp, order] = sort(temp);
        temp(1:n_modes) = lambda(order(1:n_modes));
        w = sqrt(temp(1:n_modes));

        d = mode(:,order(1:n_modes));

        %   Solution of the adjoint problem to determine the left eigenvector:
```

```
        opts.disp = 0;
        [mode_adj,lambda_adj] = eigs((K(freedofs,freedofs)'-p(i)*(G_C(freedofs,freedofs)'-...
eta*G_N(freedofs,freedofs)')),(M(freedofs,freedofs)'),n_modes,'sm',opts);

        lambda_adj = diag(lambda_adj);
        temp   = real(lambda_adj);
        [temp, order_adj] = sort(temp);

        b = mode_adj(:,order_adj(1:n_modes));

        % Calculate the sensitivity for each natural frequency:

        for f=1:1:nfreqs

            wf = w(f);
            df = d(:,f);
            bf = b(:,f);

            if wf < 10^-5
                d_w(i,e,f) = d_w(i-1,e,f);
            else
                d_w(i,e,f) = double((transpose(bf)*(d_K(freedofs,freedofs)-...
 wf^2*d_M(freedofs,freedofs))*df)/(transpose(bf)*2*wf*M(freedofs,freedofs)*df));
            end

        end

    end

%-------------------------------------------------------------------------------------------
    %   Sensitivities of the Critical Load:

    %   Stability solution to determine the critical frequency and right eigenvector:

    opts.disp = 0;
    [mode,lambda] = eigs((K(freedofs,freedofs)-P_cr*(G_C(freedofs,freedofs)-...
eta*G_N(freedofs,freedofs))),(M(freedofs,freedofs)),n_modes,'sm',opts);

    lambda = diag(lambda);
    temp   = real(lambda);
    [temp, order] = sort(temp);
    temp(1:n_modes) = lambda(order(1:n_modes));
    w = sqrt(temp(set_freq));

    d = mode(:,order(set_freq));

    %   Solution of the adjoint problem to determine the left eigenvector:

    [mode_adj,lambda_adj] = eigs((K(freedofs,freedofs)'-P_cr*(G_C(freedofs,freedofs)'-...
eta*G_N(freedofs,freedofs)')),(M(freedofs,freedofs)'),n_modes,'sm',opts);

    lambda_adj = diag(lambda_adj);
    temp   = real(lambda_adj);
    [temp, order_adj] = sort(temp);

    b = mode_adj(:,order_adj(set_freq));

    d_p(e) = double((transpose(b)*(d_K(freedofs,freedofs)-w^2*d_M(freedofs,freedofs))*d)...
/(transpose(b)*full((G_C(freedofs,freedofs)-eta*G_N(freedofs,freedofs)))*d));

    %   Check progress meter:
```

```
    if e == progress
        fprintf(1,'.');
        progress = progress + round(nodes/10);
    end
end

time = toc;

fprintf(1,'\n(%2.4f seconds)\n\n',time);

end
```

A.1.4 Function *element_ matrices_ diff.m*

```
%-------------------------------------------------------------------------------------------
%       FINITE ELEMENT MATRIX CALCULATION FOR THE
%                 FIRST ORDER PLATE THEORY
%
% March 2007 - Pedro Pastilha
%-------------------------------------------------------------------------------------------
function [d_Ke d_Me d_Ge d_Qe] = element_matrices_diff(elx, ely, lnode, x, y, ee, ar,...
 poi, N_x, N_y, N_xy)


%-------------------------------------------------------------------------------------------
%   LOCAL VARIABLES:

mat_output = 0;         % 0/1 - Hide/Show Element Matrices

xe  = [x(elx), x(elx+1), x(elx+1), x(elx)];
ye  = [y(ely), y(ely), y(ely+1), y(ely+1)];


%-------------------------------------------------------------------------------------------
%   MATERIAL PROPERTIES:

% Dimensionless Plate rigidities (Isotropic):

k = 5/6;            % Shear correction coefficient

r = (6*k*(1-poi));              % ISOTROPIC MATERIAL!!!!

D_11 = 1;
D_22 = 1;
D_12 = poi;
D_66 = (1-poi)/2;
A_44 = r*ar;
A_55 = r*ar;

% Dimensionless Moments of Inertia:

I_0 = 1;
I_2 = (1/12)*(1/ar)^2;

%-------------------------------------------------------------------------------------------
%   CONSTRUCTION OF THE ELEMENT MATRICES (NUMERICAL):

GP = [0, -sqrt(1/3), sqrt(1/3)];   % Gauss integration points
GW = [2, 1, 1];                    % Gauss integraion weights

K_11 = zeros(4);
```

```
K_12 = zeros(4);
K_13 = zeros(4);
K_22 = zeros(4);
K_23 = zeros(4);
K_33 = zeros(4);
M_11 = zeros(4);
M_22 = zeros(4);
G    = zeros(4);
Q    = zeros(4);


for i=1:1:4
    for j=1:1:4

        K_11s = 0;
        K_12s = 0;
        K_13s = 0;
        K_22s = 0;
        K_33s = 0;

        K_22n = 0;
        K_23n = 0;
        K_33n = 0;
        M_11n = 0;
        M_22n = 0;
        Gn    = 0;
        Qn    = 0;

        %   Numerical integration for Gauss Quadrature Rule (1x1 Points)
        %       -> Shear Contribution for the Stiffness Matrix

        eta = GP(1);
        ksi = GP(1);

        % Calculate the shape functions at the selected nodes:

        [me d_me d_me3 psi psi_ksi psi_eta J J_inv Jacobian] = shape_functions(eta,ksi, xe, ye, ee);

        K_11s = K_11s + (d_me(lnode)*A_55*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + J_inv(1,2)*psi_eta(j)) +...
                        d_me(lnode)*A_44*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
J_inv(2,2)*psi_eta(j)))*Jacobian*GW(1)*GW(1);

        K_12s = K_12s + (d_me(lnode)*A_55*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*psi(j))*Jacobian*GW(1)*GW(1);

        K_13s = K_13s + (d_me(lnode)*A_44*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*psi(j))*Jacobian*GW(1)*GW(1);

        K_22s = K_22s + (d_me(lnode)*A_55*psi(i)*psi(j))*Jacobian*GW(1)*GW(1);

        K_33s = K_33s + (d_me(lnode)*A_44*psi(i)*psi(j))*Jacobian*GW(1)*GW(1);

        %   Numerical integration for Gauss Quadrature Rule (2x2 Points)
        %       -> Bending and Membrane Contribution for the Stiffness Matrix
        %       -> Mass Matrix
        %       -> Stability Matrix

        for n=2:1:3

            eta = GP(n);
```

```
%             % Calculate the shape functions at the selected nodes:
%
%             [me d_me d_me3 psi psi_ksi psi_eta J J_inv Jacobian] = shape_functions(eta, 1, xe, ye, ee);
%
%             % Element Non Conservative Matrix Coefficients
%
%             Qn = Qn + ((J_inv(1,1)*psi_ksi(i) + J_inv(1,2)*psi_eta(i))*psi(j))*Jacobian*GW(n);

            for m=2:1:3

               ksi = GP(m);

               % Calculate the shape functions at the selected nodes:

               [me d_me d_me3 psi psi_ksi psi_eta J J_inv Jacobian] = shape_functions(eta,ksi, xe, ye, ee);

               % Element Stiffness Matrix Coefficients:

               K_22n = K_22n + (d_me3(lnode)*D_11*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + ...
J_inv(1,2)*psi_eta(j)) +...
                                d_me3(lnode)*D_66*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
J_inv(2,2)*psi_eta(j)))*Jacobian*GW(n)*GW(m);

               K_23n = K_23n + (d_me3(lnode)*D_12*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
J_inv(2,2)*psi_eta(j)) +...
                                d_me3(lnode)*D_66*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + ...
J_inv(1,2)*psi_eta(j)))*Jacobian*GW(n)*GW(m);

               K_33n = K_33n + (d_me3(lnode)*D_66*(J_inv(1,1)*psi_ksi(i) + ...
J_inv(1,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + ...
J_inv(1,2)*psi_eta(j)) +...
                                d_me3(lnode)*D_22*(J_inv(2,1)*psi_ksi(i) + ...
J_inv(2,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + ...
J_inv(2,2)*psi_eta(j)))*Jacobian*GW(n)*GW(m);

               % Element Mass Matrix Coefficients:

               M_11n = M_11n + d_me(lnode)*I_0*psi(i)*psi(j)*Jacobian*GW(n)*GW(m);

               M_22n = M_22n + d_me3(lnode)*I_2*psi(i)*psi(j)*Jacobian*GW(n)*GW(m);

               % Element Stability Matrix Coefficients:

               Gn = Gn + (N_x*(J_inv(1,1)*psi_ksi(i) +
  J_inv(1,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + J_inv(1,2)*psi_eta(j)) + ...
                          N_y*(J_inv(2,1)*psi_ksi(i) +
  J_inv(2,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + J_inv(2,2)*psi_eta(j)) + ...
                          N_xy*((J_inv(1,1)*psi_ksi(i) +
  J_inv(1,2)*psi_eta(i))*(J_inv(2,1)*psi_ksi(j) + J_inv(2,2)*psi_eta(j)) + ...
                          (J_inv(2,1)*psi_ksi(i) + J_inv(2,2)*psi_eta(i))*(J_inv(1,1)*psi_ksi(j) + ...
  J_inv(1,2)*psi_eta(j))))*Jacobian*GW(n)*GW(m);

               % Element Non Conservative Matrix Coefficients

               Qn = Qn + psi(i)*(J_inv(1,1)*psi_ksi(j) + J_inv(1,2)*psi_eta(j))*Jacobian*GW(n)*GW(m);

            end
```

```
        end

        % Matrix Entries:

        K_11(i,j) = K_11s;
        K_12(i,j) = K_12s;
        K_13(i,j) = K_13s;
        K_22(i,j) = K_22n + K_22s;
        K_23(i,j) = K_23n;
        K_33(i,j) = K_33n + K_33s;
        M_11(i,j) = M_11n;
        M_22(i,j) = M_22n;
        G(i,j)    = Gn;
        Q(i,j)    = Qn;
    end
end

null_mat = zeros(4);

% Element Stiffness Matrix:

Ke_upper = [ K_11,     K_12,     K_13;
             null_mat, K_22,     K_23;
             null_mat, null_mat, K_33 ];

Ke_upper = triu(Ke_upper);
Ke_lower = transpose(Ke_upper) - diag(diag(Ke_upper));
d_Ke = Ke_upper + Ke_lower;

% Element Mass Matrix:

Me_upper = [M_11,     null_mat, null_mat;
            null_mat, M_22,     null_mat;
            null_mat, null_mat, M_22   ];

Me_upper = triu(Me_upper);
Me_lower = transpose(Me_upper) - diag(diag(Me_upper));
d_Me = Me_upper + Me_lower;

% Element Stability Matrix:

d_Ge = [G,        null_mat, null_mat;
        null_mat, null_mat, null_mat;
        null_mat, null_mat, null_mat];

% Element Non-Conservative Matrix:

d_Qe = [Q,        null_mat, null_mat;
        null_mat, null_mat, null_mat;
        null_mat, null_mat, null_mat];

%-------------------------------------------------------------------------------------------
%   Matrix Printout:

if mat_output == 1

fprintf(1,'\nElement Stiffness Matrix (symmetric):\n');

pretty(Ke);

fprintf(1,'\nElement Mass Matrix (symmetric):\n');
```

```
pretty(Me);

fprintf(1,'\nElement Stability Matrix (symmetric):\n');

pretty(Ge);

end

end

%------------------------------------------------------------------------------------------
%   Lagrange Interpolation Functions:
%
%   (Rectangular Element With 4 Nodes)
%
%                ^ eta
%        4       |       3
%      ^  +------|------+
%      |  |      |      |     ksi
%   2b |  |      +------+--->
%      |  |             |
%      v  +-------------+
%         1              2
%         <------------->
%
%   psi_i = (1/4)*(1 + ksi*ksi_i)*(1 + eta*eta_i)
%   (i = node 1,2,...)
%------------------------------------------------------------------------------------------

function [me d_me d_me3 psi psi_ksi psi_eta J J_inv Jacobian] = shape_functions(eta, ksi, xe, ye, ee)

% Thickness distribution along the element (and respective derivatives):

me = (1/4)*(1-ksi)*(1-eta)*ee(1) + (1/4)*(1 + ksi)*(1 - eta)*ee(2) +...
 (1/4)*(1 + ksi)*(1 + eta)*ee(3) +  (1/4)*(1 - ksi)*(1 + eta)*ee(4);

% First derivative of (me):

d_me  = [ (1/4-1/4*ksi)*(1-eta),...
          (1/4+1/4*ksi)*(1-eta),...
          (1/4+1/4*ksi)*(1+eta),...
          (1/4-1/4*ksi)*(1+eta) ];

% First derivative of (me)^3:

d_me3 = [ 3*((1/4-1/4*ksi)*(1-eta)*ee(1)+(1/4+1/4*ksi)*(1-eta)*ee(2)+...
(1/4+1/4*ksi)*(1+eta)*ee(3)+(1/4-1/4*ksi)*(1+eta)*ee(4))^2*(1/4-1/4*ksi)*(1-eta),...
          3*((1/4-1/4*ksi)*(1-eta)*ee(1)+(1/4+1/4*ksi)*(1-eta)*ee(2)+...
(1/4+1/4*ksi)*(1+eta)*ee(3)+(1/4-1/4*ksi)*(1+eta)*ee(4))^2*(1/4+1/4*ksi)*(1-eta),...
          3*((1/4-1/4*ksi)*(1-eta)*ee(1)+(1/4+1/4*ksi)*(1-eta)*ee(2)+...
(1/4+1/4*ksi)*(1+eta)*ee(3)+(1/4-1/4*ksi)*(1+eta)*ee(4))^2*(1/4+1/4*ksi)*(1+eta),...
          3*((1/4-1/4*ksi)*(1-eta)*ee(1)+(1/4+1/4*ksi)*(1-eta)*ee(2)+...
(1/4+1/4*ksi)*(1+eta)*ee(3)+(1/4-1/4*ksi)*(1+eta)*ee(4))^2*(1/4-1/4*ksi)*(1+eta) ];

% Vector definition with the interpolation function at each node:

psi = [(1/4)*(1 - ksi)*(1 - eta) ;
       (1/4)*(1 + ksi)*(1 - eta) ;
       (1/4)*(1 + ksi)*(1 + eta) ;
       (1/4)*(1 - ksi)*(1 + eta)];

% Derivatives of the interpolation function at each node:
```

```
psi_ksi = [-(1/4)*(1 - eta) ;
            (1/4)*(1 - eta) ;
            (1/4)*(1 + eta) ;
           -(1/4)*(1 + eta)];

psi_eta = [-(1/4)*(1 - ksi) ;
           -(1/4)*(1 + ksi) ;
            (1/4)*(1 + ksi) ;
            (1/4)*(1 - ksi)];

% Compute the Jacobian Matrix:

J = transpose([xe; ye]*[psi_ksi, psi_eta]);     % Jacobian Matrix

Jacobian = det(J);                              % Jacobian

if Jacobian <= 0
    errordlg('Oh boy, oh boy! The determinant has become zero...RUN FOR YOUR LIVES!!!!')
    error('My existance has lost all meaning...The determinant has become zero');
end


J_inv = inv(J);                                 % Inverse of the Jacobian Matrix

end
```

A.1.5 Function *volume.m*

```
%------------------------------------------------------------------------------------------
% VOLUME CALCULATION FOR THE PLATE MODEL
%
% May 2007 - Pedro Pastilha
%------------------------------------------------------------------------------------------
function [V d_V] = volume(nelx,nely,ue)

%------------------------------------------------------------------------------------------
%   Physical Properties:

Lx   = 1;             % Rod Lenght (m)
Ly   = 1;          % Rod Height (m)


% % %!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
% fprintf(1,'Local values are set!!!!');
% nelx = 10;
% nely = 5;
% ue(1:(nelx+1)*(nely+1)) = 1;

%------------------------------------------------------------------------------------------
%   Finite Element data:

nodes = (nelx+1)*(nely+1);       % Number of mesh nodes
lex(1:1:(nelx+1)) = Lx/nelx;
ley(1:1:(nelx+1)) = Ly/nely;


%------------------------------------------------------------------------------------------
% Linear combination of displacements in the x direction:

a(1:nelx+1) = 0;
a(1) = (1/2)*lex(1);
```

```
for i=2:1:nelx
    a(i) = (1/2)*(lex(i-1)+lex(i));
end
a(nelx+1) = a(1);


%------------------------------------------------------------------------------------------
% Linear combination of displacements in the y direction:

b(1:nely+1) = 0;
b(1) = (1/2)*ley(1);
for i=2:1:nely
    b(i) = (1/2)*(ley(i-1)+ley(i));
end
b(nely+1) = b(1);


%------------------------------------------------------------------------------------------
% VOLUME CALCULATION:

V = 0;
d_V(1:nodes) = 0;
for j=1:1:(nely+1)
    for i=1:1:(nelx+1)
        k   = i + (j-1)*(nelx+1);
        V   = V + ue(k)*a(i)*b(j);
        d_V(k) = a(i)*b(j);
    end
end

end
```

A.1.6 Function *optimize.m*

```
%------------------------------------------------------------------------------------------
% OPTIMIZATION ANALYSIS USING MMA (Method of Moving Asymptodes)
%
% Optimization of flutter instab50.550.5ility using flutter_calc.m to determine the
% constraints and volume.m as the objective function
%
% May 2006 - Pedro Pastilha
%------------------------------------------------------------------------------------------
function optimize(nelx,nely,eta)

clear all
clc

fprintf(1,'\n\n-----------------------------------------------------\n');
fprintf(1,'\n FLUTTER OPTIMIZATION ANALYSIS (PLATE MODEL)\n');
fprintf(1,'\n-----------------------------------------------------\n\n');

%   Start time counter:

tic

%   Problem data:

res   = 5;                    % Number of constraint points to consider
steps = 500;                  % Number of load steps to calculate
slack = 0.5;                  % Maximum slack for the project variables in each iteration

eta  = 1.0;                  % Nonconservativeness parameter !!!!!
```

```
nelx = 10;                    % Number of elements in the x direction
nely = 5;                     % Number of elements in the y direction

nodes = (nelx+1)*(nely+1);    % Number of mesh nodes
Lx   = 1;                     % Rod Lenght (m)
Ly   = 0.1;                   % Rod Height (m)
lex = Lx/nelx;
ley = Ly/nely;
x(1:1:(nelx+1)) = 0:lex:Lx;
y(1:1:(nely+1)) = 0:ley:Ly;


%------------------------------------------------------------------------------------------
%   OPTIMIZATION PROCESS:
%------------------------------------------------------------------------------------------

ue_min = 10^-8;    % Lower Limit for the project variable
ue_max = 10;       % Upper Limit for the project variable

%   Uniform Column:

ue(1:nodes) = 1;
[V, d_V] = volume(nelx,nely,ue);
[P_cr_0, p, w, res] = flutter_calc(nelx, nely, ue, eta, steps, res);


%------------------------------------------------------------------------------------------
%   Open Results File:

filename = ['Optimization_Results_eta_', num2str(eta),'_(',num2str(nelx),'x',num2str(nely),')','.txt'];
file = fopen(filename,'w');
fprintf(file,'\n\n----------------------------------------------------\n');
fprintf(file,'OPTIMIZATION RESULTS FOR AN ETA OF %1.2f', eta);
fprintf(file,'\n----------------------------------------------------\n\n');


%------------------------------------------------------------------------------------------
%   MMA Process:

%   First Iteration Data:

n       = nodes;
xval    = ue';
xmin    = (1-slack)*xval;
xmax    = (1+slack)*xval;
% xmin    = ue_min';
% xmax    = ue_max';
xold1   = xval;            % First iteration! Error if this doesn't exist
xold2   = xval;            %
low     = xmin;
upp     = xmax;
a0      = 1;

iter    = 1;
maxiter = 150;

%   MMA Cycle:

while iter < maxiter

    %   Fixed Region (fixed limits for the project variables):
```

```
    xmin((nelx+1):(nelx+1):nodes) = 0.5;
    xmax((nelx+1):(nelx+1):nodes) = 1.5;
    xmin(1:(nelx+1):nodes) = 0.5;
    xmax(1:(nelx+1):nodes) = 1.5;


    fprintf(file,'\n-----------------------------------------------------\n Iteration %d\n',iter);
    fprintf(1,'\n-----------------------------------------------------\n Iteration %d\n',iter);

    [P_cr, p, w, res, set_freq] = flutter_calc(nelx, nely, xval, eta, steps, res);
    [d_p d_w] = sensitivity(nelx, nely, xval, eta, p, P_cr, res, set_freq);

    figname = ['Load_vs_Freq',num2str(iter-1),'.png'];

    fprintf(1,'\n--> MMA OPTIMIZATION SEQUENCE\n');

    %   Start time counter:

    tic

    m       = 1 + 10*res;

    f0val   = V;
    df0dx   = d_V';
    df0dx2  = 0*df0dx;

%-------------------------------------------------------------------------------------------

    c = 0;      % Even frequnecy distance
    w = real(w);

    constr(1)                = P_cr_0 - P_cr;           % P_cr >= P_cr_0
    constr(2:res+1)          = -w(:,1);                 % W1 >= 0
    constr(res+2:2*res+1)    = (w(:,1) - w(:,2));       % W2 - W1 >= 0
    constr(2*res+2:3*res+1)  = (w(:,2) - w(:,3)) + c;
    constr(3*res+2:4*res+1)  = (w(:,3) - w(:,4));
    constr(4*res+2:5*res+1)  = (w(:,4) - w(:,5)) + c;
    constr(5*res+2:6*res+1)  = (w(:,5) - w(:,6));
    constr(6*res+2:7*res+1)  = (w(:,6) - w(:,7)) + c;
    constr(7*res+2:8*res+1)  = (w(:,7) - w(:,8));
    constr(8*res+2:9*res+1)  = (w(:,8) - w(:,9)) + c;
    constr(9*res+2:10*res+1) = (w(:,9) - w(:,10));
    fval                     = constr';

    dfdx(1,:)                = - d_p(:);
    dfdx(2:res+1,:)          = -d_w(1:res,:,1);
    dfdx(res+2:2*res+1,:)    = d_w(1:res,:,1) - d_w(1:res,:,2);
    dfdx(2*res+2:3*res+1,:)  = d_w(1:res,:,2) - d_w(1:res,:,3);
    dfdx(3*res+2:4*res+1,:)  = d_w(1:res,:,3) - d_w(1:res,:,4);
    dfdx(4*res+2:5*res+1,:)  = d_w(1:res,:,4) - d_w(1:res,:,5);
    dfdx(5*res+2:6*res+1,:)  = d_w(1:res,:,5) - d_w(1:res,:,6);
    dfdx(6*res+2:7*res+1,:)  = d_w(1:res,:,6) - d_w(1:res,:,7);
    dfdx(7*res+2:8*res+1,:)  = d_w(1:res,:,7) - d_w(1:res,:,8);
    dfdx(8*res+2:9*res+1,:)  = d_w(1:res,:,8) - d_w(1:res,:,9);
    dfdx(9*res+2:10*res+1,:) = d_w(1:res,:,9) - d_w(1:res,:,10);
    dfdx2                    = 0*dfdx;

%-------------------------------------------------------------------------------------------
%     %   eta = 0.0 ONLY!!!
%
%     c = 0;      % Even frequnecy distance
%     w = real(w);
%     m       = 1;
%
%     constr(1)                = P_cr_0 - P_cr;           % P_cr >= P_cr_0
%     fval                     = constr';
%
%     dfdx(1,:)                = - d_p(:);
%     dfdx2                    = 0*dfdx;


    a     = zeros(m,1);
    c     = 1000*ones(m,1);
    d     = zeros(m,1);

    [xmma,ymma,zmma,lam,xsi,eta_mma,mu,zet,s,low,upp] = ...
    mmasub(m,n,iter,xval,xmin,xmax,xold1,xold2,f0val,df0dx,df0dx2,fval,dfdx,dfdx2,low,upp,a0,a,c,d);

    xold2 = xold1;
    xold1 = xval;
    xval  = xmma;

    ue = xval';

    xmin(1:nodes) = (1-slack)*xval(1:nodes);
    xmax(1:nodes) = (1+slack)*xval(1:nodes);

    for e=1:1:nodes
        if xmin(e) < ue_min
            xmin(e) = ue_min;
        end
        if xmax(e) > ue_max
            xmax(e) = ue_max;
        end
    end

    V_old = V;
    [V d_V]  = volume(nelx,nely,ue);

    if V < 0.50
        slack = 0.1;
    end

    error = (abs(V_old-V)/V_old)*100;

    if error < 10^-3
        maxiter = iter;
    else
        iter = iter+1;
    end

    fprintf(1,'Iteration results:\n   Project Variables:\n');
    fprintf(file,'Project Variables:\n');
    for i=1:1:(nodes)
        fprintf(1,'   %1.5f',ue(i));
        fprintf(file,'   %1.5f',ue(i));
    end
    fprintf(1,'\n   Volume: %1.5f\n',V);
    fprintf(1,'   Variation:  %1.5f %% \n',error);
    fprintf(file,'\n   Volume: %1.5f\n',V);
    fprintf(file,'   Variation:  %1.5f %% \n',error);
    fprintf(file,'   Critical Load: %1.5f \n',P_cr);

    ue_iter(iter,:) = ue(:);
    V_iter(iter) = V;

    %   Finish time counter:

    toc

end

fclose(file);

%-------------------------------------------------------------------------------------
% POST PROCESSING:
%-------------------------------------------------------------------------------------

% Final results:

fprintf(1,'\n\n---------------------------------------\nFINAL RESULTS:\n   ...
Project Variables:\n');

for i=1:1:(nodes)
    fprintf(1,'      %1.5f',ue(i))
end
fprintf(1,'\nVolume:     %1.5f (%2.3f %% reduction)\n',V, (1-V)*100);
fprintf(1,'Iterations: %d \n', maxiter);
fprintf(1,'\n---------------------------------------\n\n');

% Generate a Mat File with the results at each iteration:

file_mat = ['Iteration_results_eta_',num2str(eta),'_(',num2str(nelx),'x',num2str(nely),')','.mat'];
save(file_mat, 'ue_iter', 'V_iter');

% Plot the stability results:

flutter_plot(nelx, nely, ue, eta, steps)

% Convert the thickness vector into a mesh matrix:

ue_grid = zeros((nely+1),(nelx+1));
for j=1:1:(nely+1)
    for i=1:1:(nelx+1)
        ue_grid(j,i) = ue(i + (j-1)*(nelx+1));
    end
end

% Generate a grayscale colormap going from light to dark:

color(:,:) = [0:0.01:1; 0:0.01:1; 0:0.01:1]';
color = 1-color;

figure(3)
contourf(x,y,ue_grid,100,'LineStyle','none')
title('Thickness Distribution');
view([0 90]);
xlabel('x (m)');
ylabel('y (m)');
zlabel('Thickness');
colormap(color)
colorbar('SouthOutside')
set(gca,'DataAspectRatio',[2 1 40]);

figure(4)
surf(x,y,ue_grid)
```

```
view([45 45]);
xlabel('x (m)');
ylabel('y (m)');
zlabel('Thickness');
colormap(color)
set(gca,'DataAspectRatio',[2 1 40]);

end


%------------------------------------------------------------------------------------------
%   "This is the End
%   Beautiful Friend
%   This is the End
%   My only Friend, the End"
%                   Jim Morrison, The Doors
```