

# Optimization of stiffened panels using a combination of FEM and a predictor-corrector interior point method

M. Deklerck

Master of Science Thesis



# **Optimization of stiffened panels using a combination of FEM and a predictor-corrector interior point method**

MASTER OF SCIENCE THESIS

For the degree of Master of Science in Aerospace Engineering at Delft  
University of Technology

M. Deklerck

February 16, 2016

Faculty of Aerospace Engineering · Delft University of Technology





Copyright © Aerospace Structures & Computational Mechanics  
All rights reserved.

DELFT UNIVERSITY OF TECHNOLOGY  
DEPARTMENT OF  
AEROSPACE STRUCTURES & COMPUTATIONAL MECHANICS

The undersigned hereby certify that they have read and recommend to the Faculty  
of Aerospace Engineering for acceptance a thesis entitled

OPTIMIZATION OF STIFFENED PANELS USING A COMBINATION OF FEM AND A  
PREDICTOR-CORRECTOR INTERIOR POINT METHOD

by

M. DEKLERCK

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Dated: February 16, 2016

Committee chairman:

\_\_\_\_\_  
dr. M. M. Abdallah, MSc

Committee members:

\_\_\_\_\_  
dr. C. Kassapoglou, MSc

\_\_\_\_\_  
dr. ir. O. K. Bergsma



---

# Table of Contents

|                                                         |            |
|---------------------------------------------------------|------------|
| <b>List of Figures</b>                                  | <b>iii</b> |
| <b>List of Tables</b>                                   | <b>v</b>   |
| <b>Summary</b>                                          | <b>vii</b> |
| <b>Acknowledgements</b>                                 | <b>ix</b>  |
| <b>1 Introduction</b>                                   | <b>1</b>   |
| <b>2 Literature Study</b>                               | <b>3</b>   |
| 2.1 Stiffened panels . . . . .                          | 3          |
| 2.2 Structural Optimization . . . . .                   | 6          |
| 2.2.1 Transformation to unconstrained problem . . . . . | 7          |
| 2.2.2 Principle of duality . . . . .                    | 9          |
| 2.2.3 Approximations . . . . .                          | 11         |
| 2.2.4 Optimization methods . . . . .                    | 12         |
| 2.2.5 Starting point . . . . .                          | 21         |
| <b>3 Model definition</b>                               | <b>25</b>  |
| 3.1 Basic model . . . . .                               | 25         |
| 3.2 Model translation to NASTRAN . . . . .              | 27         |
| 3.2.1 Basic information for NASTRAN bdf setup . . . . . | 28         |
| 3.2.2 Required cards . . . . .                          | 28         |
| 3.2.3 Defining stiffeners . . . . .                     | 30         |
| 3.3 Mesh convergence study . . . . .                    | 33         |
| 3.4 Model Verification . . . . .                        | 35         |

|          |                                                     |           |
|----------|-----------------------------------------------------|-----------|
| <b>4</b> | <b>Model optimization</b>                           | <b>37</b> |
| 4.1      | Problem . . . . .                                   | 37        |
| 4.2      | Process . . . . .                                   | 38        |
| 4.2.1    | Sensitivity adjustment . . . . .                    | 39        |
| 4.2.2    | Approximation of the primal problem . . . . .       | 41        |
| 4.2.3    | Predictor-Corrector interior point method . . . . . | 42        |
| 4.2.4    | Convergence . . . . .                               | 46        |
| 4.3      | Test Cases . . . . .                                | 47        |
| 4.4      | Results . . . . .                                   | 49        |
| 4.4.1    | Basic Optimization . . . . .                        | 49        |
| 4.4.2    | Different cross-sections . . . . .                  | 50        |
| 4.4.3    | Multiple properties . . . . .                       | 51        |
| 4.4.4    | Different amount of stiffeners . . . . .            | 54        |
| <b>5</b> | <b>Conclusion</b>                                   | <b>57</b> |
| <b>6</b> | <b>Recommendations</b>                              | <b>59</b> |
|          | <b>Bibliography</b>                                 | <b>61</b> |
| <b>A</b> | <b>BDF setup</b>                                    | <b>65</b> |



---

## List of Figures

|     |                                                                                                                     |    |
|-----|---------------------------------------------------------------------------------------------------------------------|----|
| 2.1 | Typical cross-sections for stiffeners . . . . .                                                                     | 4  |
| 2.2 | Loads on a wing of an aircraft during flight [8] . . . . .                                                          | 4  |
| 2.3 | Simplistic wingbox configuration [9] . . . . .                                                                      | 4  |
| 2.4 | Buckling failure modes for stiffened panels [10] . . . . .                                                          | 5  |
| 2.5 | Panel Buckling . . . . .                                                                                            | 7  |
| 2.6 | Transformation methods from problem (f) with constraints (g) to unconstrained problem ( $\tilde{f}$ )[15] . . . . . | 8  |
| 2.7 | Linear and reciprocal approximation methods . . . . .                                                               | 12 |
| 3.1 | Test model for NASTRAN analysis . . . . .                                                                           | 26 |
| 3.2 | Conversion from 3D model to 2D FEM . . . . .                                                                        | 27 |
| 3.3 | Conversion from 3D model to 2D FEM . . . . .                                                                        | 27 |
| 3.4 | I and Z cross sections with relating dimensions for NASTRAN cars [47] . . . . .                                     | 31 |
| 3.5 | Tested stiffener cross sections and their design variables . . . . .                                                | 33 |
| 3.6 | 2D FEM model . . . . .                                                                                              | 34 |
| 3.7 | Nearly pure buckling modes for verification purposes . . . . .                                                      | 35 |
| 4.1 | Optimization process . . . . .                                                                                      | 40 |
| 4.2 | Convergence of objective . . . . .                                                                                  | 49 |
| 4.3 | Evolution of the objective during optimization of case 4 . . . . .                                                  | 53 |

|     |                                                                                                    |    |
|-----|----------------------------------------------------------------------------------------------------|----|
| 4.4 | Evolution of the buckling constraint during optimization of case 4 . . . . .                       | 53 |
| 4.5 | Critical buckling modes for case 4 for different steps in the optimization process                 | 55 |
| 4.6 | Evolution of the objective during optimization for different amounts of I-<br>stiffeners . . . . . | 56 |
| 4.7 | Evolution of the objective during optimization for different amounts of Z-<br>stiffeners . . . . . | 56 |
| A.1 | BDF analysis set-up . . . . .                                                                      | 66 |

---

# List of Tables

|      |                                                                                     |    |
|------|-------------------------------------------------------------------------------------|----|
| 3.1  | Model Properties . . . . .                                                          | 26 |
| 3.2  | Mesh convergence for $n_{bs}$ . . . . .                                             | 34 |
| 3.3  | Mesh convergence for $n_y$ . . . . .                                                | 34 |
| 3.4  | Comparison between analytical and numerical results for buckling . . . . .          | 36 |
| 4.1  | Design rules . . . . .                                                              | 38 |
| 4.2  | Minimum and maximum dimensions for different stiffener types . . . . .              | 39 |
| 4.3  | Available and required design variables for assorted stiffener types . . . . .      | 41 |
| 4.4  | Different test case settings for optimization . . . . .                             | 48 |
| 4.5  | Initial values for the parameters . . . . .                                         | 48 |
| 4.6  | Objective values per iteration . . . . .                                            | 49 |
| 4.7  | Constraint values per iteration . . . . .                                           | 50 |
| 4.8  | Parametric History . . . . .                                                        | 50 |
| 4.9  | Comparison for optimization objective with different cross sections . . . . .       | 51 |
| 4.10 | Start values for design variables . . . . .                                         | 51 |
| 4.11 | Constraint values for test case 3 . . . . .                                         | 51 |
| 4.12 | Comparison for optimization objective with different amount of properties . . . . . | 52 |
| 4.13 | Parameters per property after optimization . . . . .                                | 52 |
| 4.14 | Constraint values for test case 1 and 2 after optimization . . . . .                | 52 |

|                                                                                  |    |
|----------------------------------------------------------------------------------|----|
| 4.15 Optimization data for cases with different amount of I-stiffeners . . . . . | 54 |
| 4.16 Optimization data for cases with different amount of Z-stiffeners . . . . . | 56 |

---

# Summary

Structural optimization, first introduced by Schmidt in 1960 [1], is a rapid growing factor in the development of new aerospace structures. This growth is established by the increase in numerical modelling techniques, cheaper computer power, the increasing cost of production and competition between companies. The combination of both structural optimization and finite element software allowed for the rise of new and more efficient optimization methods provided that the software can perform sensitivity analysis.

Many programs used in industry today such as BOSS Quattro [2], PASCO [3] and VICONOPT [4] restrict themselves to basic optimization methods. The goal now is to develop an optimizer for stiffened panels, using a combination of FEM and a more advanced optimization method.

Interior point methods have been proven to be more efficient than primal-dual methods for solving sub-problems[5]. Therefore Mehrotra's predictor-corrector interior point method is used in the version of Zillober [7]. To reach convergence convex approximations are required. The conservative approximation from Fleury's ConLin [6] provides the basis of many other more advance approximation methods. Therefore this method is chosen to form the initial optimizer.

A 2D The FEM model is established using shell and bar elements for the panel and stiffeners respectively. This allows for easy adjustment of the geometry without the need to change the model itself. The bar element properties are defined by the PBAR card rather than the PBARL card in NASTRAN. This avoids the input of fixed NASTRAN specified cross sections with limited design freedom.

The sensitivities with respect to stiffener properties are extracted from NASTRAN. These are then converted to the required sensitivities using analytical equations. With all the necessary information available, the inner loop of the optimization process is initiated. Approximations of the constraints, objective and sensitivities are produced. Based on the approximations, the predictor step establishes a maximum step size, which is then adjusted by the corrector step to a more feasible one. This is done iteratively until the duality gap is below a specified limit. Finally a new outer iteration can start if no convergence is reached.

Three goals were achieved by analysing of 11 test cases. First the optimizer shows that it can handle different property sets for the stiffeners within the same panel. Secondly, the optimization works for different cross sections. Finally, when performed for similar panels with a different amount of stiffeners, an optimal number is found. The optimization is performed for minimum weight while limited by stress, buckling and design constraints. The results indicate that for 8 out of 11 cases convergence is reached within 12 cycles. Due oscillatory behaviour two other cases converged relatively slow and one did not converge at all. This happens due to the incapability of the optimizer to consider new buckling modes establishing with the adjustment of the parameters.

In the end however all three statements were proven outside of the three oscillating cases. For the model that was defined, the optimal amount of stiffeners is 7. Additionally I-beam stiffener provided the most consistent performance with respect to convergence. Finally for this case, although optimizing for different stiffener properties per panel lead to a small reduction in weight, it is not worth the computational effort.

So it can be concluded that the optimizer works. On top of this the restrictions on the cross-section defined by NASTRAN were eliminated by extracting a different set of sensitivities and adjusting them using analytical equations. This leads to an optimizer, which can perform size and shape optimization by use of NASTRAN analyses, analytical transformations and an interior point optimization method.

---

# Acknowledgements

In my thesis I developed an optimizer for uni-axially compressed stiffened panels. The original idea was to use topology optimization to determine the optimal stiffener shape for a specific load case. Then size and shape optimization can be used to adjust the stiffener based on the magnitude of the loads. This however was infeasible within the time period set for a thesis. So a feasible base for this idea was developed instead.

First of all I would like to express my gratitude to my supervisor Mr. Abdallah for guiding me to a feasible goal for the thesis and providing the support to achieve it. This enabled me to restrict myself and produce a decent result for my thesis instead of going overboard and having to deliver a program that is not finished. On top of this, it can still serve as a starting point for a new research, which can still result in the completion of my original idea.

I would also like to thank D. Peeters for taking over the supervision of my thesis while Mr. Abdallah was out due to illness; Z. Hong to guide me through some of the important elements of the interior point method; And E. Ferede for his support on the coupling between MATLAB and NASTRAN.

Additionally I would like to thank my family for giving me this opportunity and supporting me financially and mentally throughout all these years so that I could reach this point. The same goes for my friends who have always been there for me during some of the rough patches and enabled me to keep my spirit up. On top of this I would like to thank Annelies for pushing me to my limits and for being my refuge when things got difficult.

Finally I would like to thank the defence committee for their honest opinions and for the critical reflection of my work.

---

# Chapter 1

---

## Introduction

Structural optimization, first introduced by Schmidt in 1960 [1], is a rapid growing factor in the development of new structures. This growth is established by the increase in numerical modelling techniques, cheaper computer power, the increasing cost of production and competition between companies. The combination of both structural optimization and finite element software allowed for the rise of new and more efficient optimization methods. These methods are based on the fact that recent FE software is capable of providing a sensitivity analysis of the structure.

One of the most important structures in distinct field of engineering is a stiffened panel. In aerospace, the load bearing part of the wing consists primarily out of stiffened panels as they allow for low weight and high resistance to compressive and bending loads. Therefore these are generally optimized using packages such as BOSS Quattro [2], PASCO [3] and VICONOPT [4]. These however often restrict themselves to basic optimization methods.

The goal of this thesis is to develop an optimizer for stiffened panels by combining FE software with an interior point method coded in MATLAB.

The FE software and optimization method selected are NASTRAN and Mehrotra's predictor corrector interior point method as described by Zillober [7] respectively. The optimization method requires convex approximations in order to reach convergence. Therefore the Con-Lin conservative approximation scheme is used [6]. For the FE analyses, the problem is reduced from 3D to 2D for faster analyses and optimization. It simplifies the model and enables adjustment of stiffener dimensions by adjusting the applied properties. Stiffeners can be defined in NASTRAN by indicating its cross section geometry and dimensions. This limits



the types of cross sections and their design freedom to those in the NASTRAN database. A free choice of design variables is preferred therefore, a solution for these restrictions is established.

Mehrotra's predictor corrector method was selected together with ConLin's approximation method after a thorough literature study. The most important parts of this study are recapitulated in Chapter 2 clarifying the choice of that specific combination. Once the method is established, a model is defined for optimization. Chapter 3 provides a description of the setup and a reasoning behind some important choices with respect to property and element definition. Finally Chapter 4 describes the optimization process, which is applied to several test cases and discusses the results. The conclusion and recommendations for the thesis can be found in Chapter 5 and 6 respectively.

---

## Chapter 2

---

# Literature Study

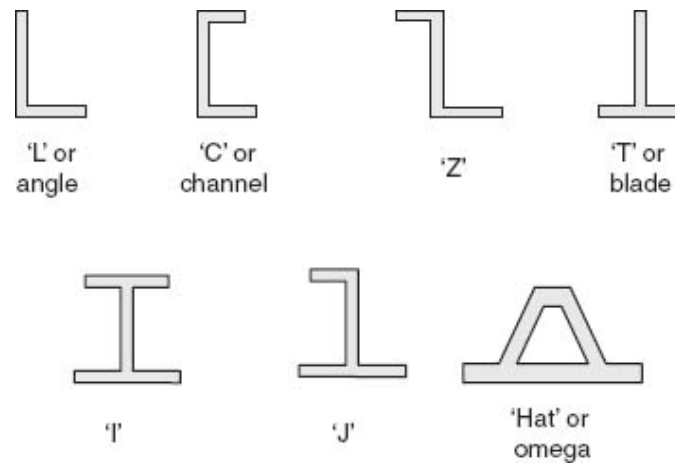
Optimizing a stiffened panel requires some general information about stiffened panels and optimization. Section 2.1 handles the typical stiffeners, loads and failure modes of stiffened panels, while Section 2.2 discusses the general optimisation process and some typical methods used for structures.

### 2.1 Stiffened panels

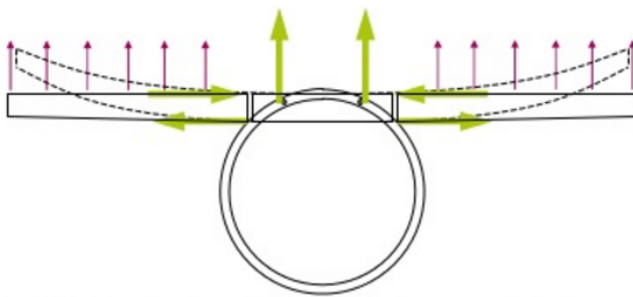
Stiffened panels are one of the more common structural parts used in distinct fields of engineering such as aerospace. The stiffeners attached to the panels are required to increase the panel's resistance against buckling and general stresses accomplishing a high strength to weight ratio. In practice, companies mostly use a distinct set of stiffener shapes of which most are depicted in Figure 2.1.

The loads on an aircraft wing during flight are represented by Figure 2.2. These loads clearly show that the load bearing parts of the wing need to sustain either compressive or tensional forces. The wingbox, which is the load bearing component of a wing needs to accomplish this feature. For that reason it consists of two stiffened panels on the top and bottom of the wing, two spars on the sides and ribs across the length of the wing as shown in Figure 2.3. Stiffened panels thus form an important part of the load bearing capabilities of a wing.

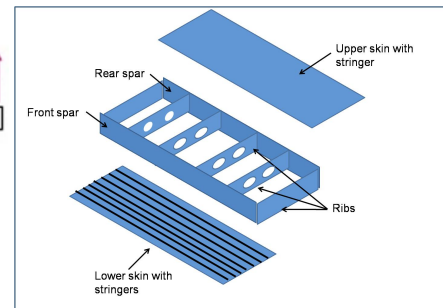
Compressive loads are the origin for more failure modes than tensional loads. Therefore, Stiffened panels in compressions form the basis for this section and research in general. Since



**Figure 2.1:** Typical cross-sections for stiffeners



**Figure 2.2:** Loads on a wing of an aircraft during flight [8]

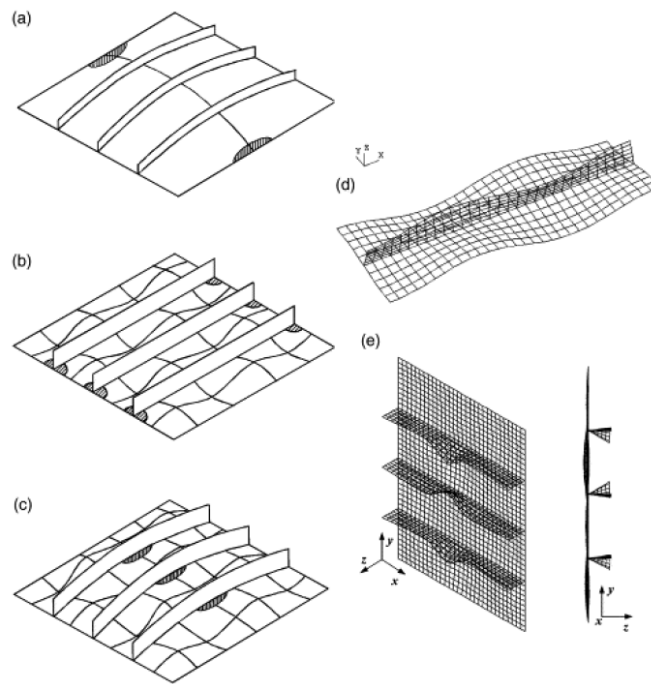


**Figure 2.3:** Simplistic wingbox configuration [9]

the wing is not supposed to fail there are several failure modes that have to be accounted for. These can consist of failure of the stiffener, panel or a combination of both. Figure 2.4 provides a visual representation of those failure modes while the list below provides a general description [10].

- **Stress failure** Failure of the panel induced by a stress exceeding the stress limit, which is usually the yield or ultimate stress depending if plastic deformation is allowed.
- **Buckling failure modes Figure 2.4**
  - a *Global buckling*: Buckling of the panel and stiffeners in a combined mode. This buckling mode appears more frequently when the stiffness of the panel is relatively larger than that of the stiffeners.
  - b *Panel buckling*: Local buckling of the part of the panel between stiffeners and/or boundary. This part is also called a sub-panel.

- c *Beam-Column buckling*: Column buckling of the stiffeners. Can result in global buckling or just happen in combination with another panel buckling mode. The buckling mode is enforced by failure of a combination of the stiffener and the effective panel width.
- d *Local buckling*: Stiffener induced failure mode by local buckling of the web.
- e *Flexural-torsional buckling*: Buckling mode similar to the local buckling of the stiffener web. However when 'tripping' occurs the plate loses its effective stiffness, which results in a global buckling mode.



**Figure 2.4:** Buckling failure modes for stiffened panels [10]

The finite element model presented in Chapter 3 can not determine failure modes c-e due to the reduction from 3D to 2D. Therefore, only global and panel buckling are discussed in more detail.

The global buckling load of a stiffened panel under uni-axial compression can be determined by handling it as a single beam. In case of a panel that is simply supported at the load bearing sides, the global buckling load can be determined by Equation (2.1). Where  $E$  is the young's modulus,  $I$  is the area moment of inertia,  $K$  is a factor necessary to account for different boundary conditions and  $(1 - \nu^2)L^2$  is the equivalent length [11]. The moment of inertia in this case is determined by the stiffeners and the effective panel width [12]. The latter is defined by Equation (2.2), where  $T$  is the panel thickness,  $E$  is young's modulus and  $\sigma_y$  is the

yield stress. The effective width represents the part of the skin which aides in the increase of the resistance against buckling of the stiffeners.

$$P_{cr} = \frac{\pi^2 EI}{K(1 - \nu^2)L^2} \quad (2.1)$$

$$W_{eff} = T \sqrt{\frac{E}{\sigma_y}} \quad (2.2)$$

Equation (2.3)[11] defines the buckling of a normal simply supported panel. However this same equation can be used for the sub-panels where the stiffeners define elastic boundary conditions.  $D$  is the flexural stiffness as given by Equation (2.4). The variables  $a, b$  and  $t$  are the length, width and thickness of the panel while  $m$  and  $n$  represent the half-waves in  $y$  and  $x$  direction respectively as depicted in Figure 2.5a. Note that for simplicity the boundary conditions enforced by the stiffeners are frequently defined as simply supported. However a slight deviation in the results should be expected under those assumptions. For different boundary conditions, coefficients for adjustment can be determined by use of Figure 2.5b [11].

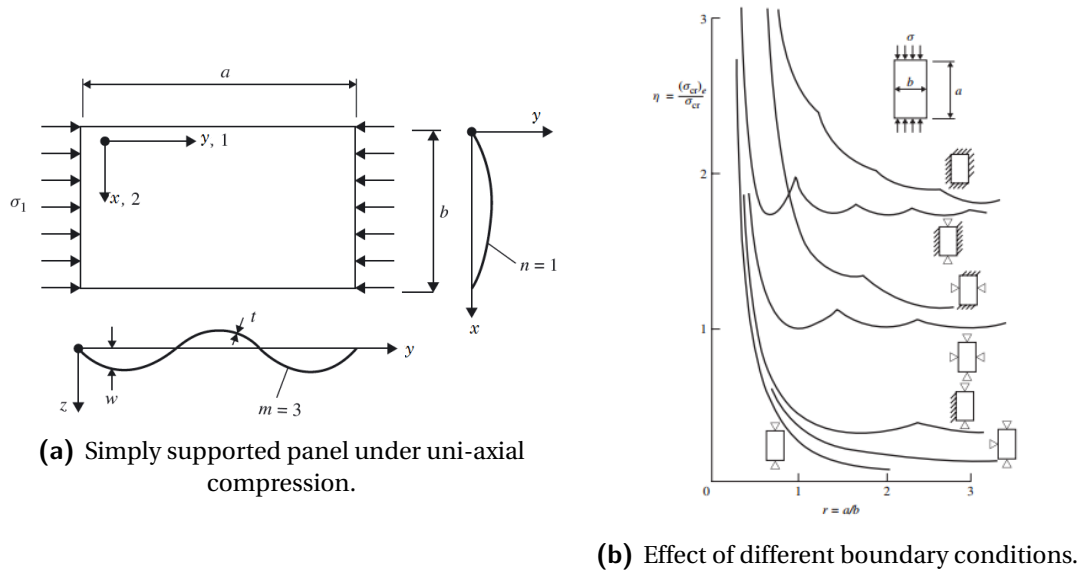
$$\sigma_{cr} = \frac{D\pi^2}{t} \left[ \left( \frac{m}{a} \right)^2 + \left( \frac{n}{b} \right)^2 \right] \left( \frac{a}{m} \right)^2 \quad (2.3)$$

$$D = \frac{Et^3}{12(1 - \nu^2)} \quad (2.4)$$

## 2.2 Structural Optimization

The general statement of an optimization problem consists in minimizing an objective function  $f_0(X)$  subjected to behaviour constraints  $g_j(X)$  insuring the feasibility of the structural design. This problem expressed by Equation 2.5 has to be solved in order to reach an optimal design.

$$\begin{aligned} & \min_X f_0(\mathbf{x}) \\ & s.t. : g_j(\mathbf{x}) \leq 0 \quad j = 1, \dots, m \\ & x_i^{min} \leq x_i \leq x_i^{max} \quad i = 1, \dots, n \end{aligned} \quad (2.5)$$



**Figure 2.5:** Panel Buckling

These kind of problems are generally solved in two steps. First an outer loop in which the analysis of the model is performed and secondly an inner loop where the optimization process is performed based on the information of the analysis. The inner loop transforms the problem into an unconstrained one, which is then optimised by use of dual variables, approximated sub-problems and a certain optimisation method.

Section 2.2.1 handles different methods used to transform a constraint problem to an unconstrained problem. The principle of duality is explained in Section 2.2.2. Some of the most basic approximation methods are explained in Section 2.2.3. Finally, the optimisation methods are described in Section 2.2.4. Once all the methods are explained, a small trade-off is performed to determine which one is most fit for implementation in Section 2.2.5.

### 2.2.1 Transformation to unconstrained problem

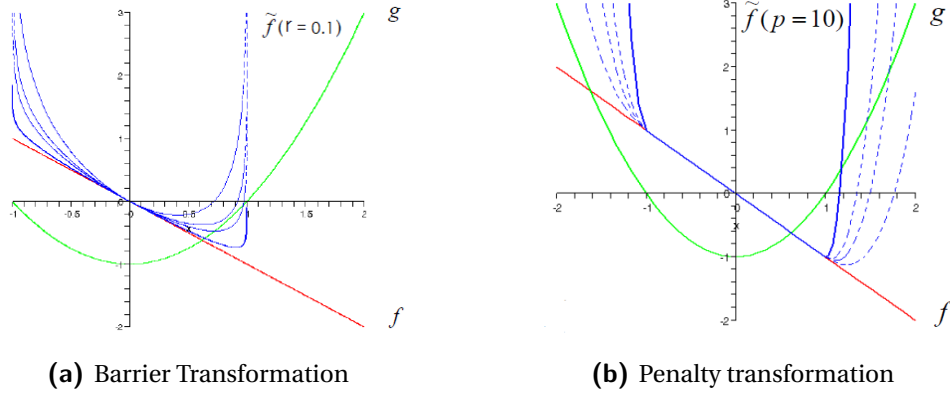
Sequential penalty transformation or exact penalty transformations are the two main methods for transforming a constrained to an unconstrained problem [14]. The group of sequential penalty transformation consists of 2 major classes, the barrier function methods and the penalty function methods. These are also called the interior-point penalty function methods and exterior-point penalty function methods respectively.

In practice, barrier functions are preferred as they will always lead to a feasible design even if convergence is not reached. Equation (2.6) indicates the general expression for a transformation using a barrier function. Where the barrier function  $B(\mathbf{x})$  is positively defined in the

interior of the constraint set and goes to infinity as  $\mathbf{x}$  approaches the boundary. Typical barrier functions are logarithmic as shown in Equation (2.7). Note that barrier functions are not applicable to equality constraints. An example of the barrier function method can be seen in Figure 2.6a.

$$T(\mathbf{x}, r) \triangleq f(\mathbf{x}) + rB(\mathbf{x}), \quad r > 0 \quad (2.6)$$

$$B(\mathbf{x}) \triangleq - \sum_{j=1}^m \ln[-g_j(\mathbf{x})] \quad (2.7)$$



**Figure 2.6:** Transformation methods from problem (f) with constraints (g) to unconstrained problem ( $\tilde{f}$ )[15]

Penalty function methods are generally used for solving equality constraints. Equation (2.8) shows the general transformation using the penalty function method. The corresponding penalty functions are given by Equations (2.9) and (2.10) for inequality and equality constraints respectively. The inequality penalty function is more generally named the quadratic loss function. An example of the penalty function method for inequality constraints can be seen in Figure 2.6b.

$$T(\mathbf{x}, r) \triangleq f(\mathbf{x}) + r^{-1}P(\mathbf{x}), \quad r > 0 \quad (2.8)$$

$$P(\mathbf{x}) \triangleq \sum_{j=1}^m [\max(0, g_j(\mathbf{x}))]^2 \quad (2.9)$$

$$P(\mathbf{x}) \triangleq \sum_{j=1}^m [h_j(\mathbf{x})]^2 \quad (2.10)$$

To avoid ill-conditioning of the penalty transformation methods, an augmented Lagrangian is implemented. This consists of a lagrangian function where the stationary point  $\mathbf{x}_*$  is kept consistent and thus only the Hessian of the Lagrangian projected on the tangent subspace is adjusted. In order to achieve this the function should comply with the following properties at  $\mathbf{x}_*$ :

- $\mathbf{x}_*$  is a stationary point
- The curvature in the tangent subspace is positive ( $\mathbf{y}^T \mathbf{H} \mathbf{y} > 0 \forall \mathbf{y}$ )
- the curvature in the normal subspace is 0 ( $\nabla f = \mathbf{0}$ )

### 2.2.2 Principle of duality

Duality is a very important concept in optimization [16]. It exploits the separable form of each approximate sub-problem to construct a sequence of explicit dual functions. The most general principle of duality is explained below. Assume the problem as seen in Equation 2.11.

$$\begin{aligned} \min_{\mathbf{x}} f_0(\mathbf{x}) &= \mathbf{c}^T \mathbf{x} \\ \text{s.t. : } g_j(\mathbf{x}) : \mathbf{a}_j \mathbf{x} &\geq \mathbf{b} \quad j = 1, \dots, m \\ x_i &\geq 0 \quad i = 1, \dots, n \end{aligned} \quad (2.11)$$

Now the constraints are replaced by maximum conditions, which leads to a new primal problem such as shown in Equation (2.12). Where  $\Pi$  and  $\mathbf{s}$  are dual variables, in this kind of optimisation also known as Lagrangian multipliers.

$$\tilde{f} = \mathbf{c}^T \mathbf{x} + \max_{\Pi \geq 0} (-\Pi^T (\mathbf{A}\mathbf{x} - \mathbf{b})) + \max_{\mathbf{s} \geq 0} (-\mathbf{s}^T \mathbf{x}) \quad (2.12)$$

By replacing them with maximum conditions the following relations specified by Equations (2.13) and (2.14) show that the constraints have to be uphold to reach a minimum.

$$\max_{\Pi \geq 0} (-\Pi^T (\mathbf{A}\mathbf{x} - \mathbf{b})) = \begin{cases} 0 & \text{if } \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ -\infty & \text{if else} \end{cases} \quad (2.13)$$

$$\max_{\mathbf{s} \geq 0} (-\mathbf{s}^T \mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \geq 0 \\ -\infty & \text{if else} \end{cases} \quad (2.14)$$

Knowing that  $\mathbf{x}$  and  $\mathbf{c}$  are not a function of  $\mathbf{s}$  and  $\Pi$ , it can be taken into the maximum criterion resulting in the optimization problem specified in Equation (2.15).

$$\min_{\mathbf{x}} \max_{\mathbf{s}, \Pi \geq 0} ((\mathbf{c}^T \mathbf{x} - \Pi^T (\mathbf{A}\mathbf{x} - \mathbf{b}) - \mathbf{s}^T \mathbf{x})) \quad (2.15)$$



Assuming super duality allows switching between the minimum and maximum criteria while transposing the components inside them resulting in the optimisation shown in Equation (2.16).

$$\max_{\mathbf{s} \geq, \Pi^T} \min_{\mathbf{x}} ((\mathbf{c} - \mathbf{A}^T \Pi - \mathbf{s}) \mathbf{x}^T + \mathbf{b}^T \Pi) \quad (2.16)$$

This equation using the reversed method from before, it can again be written as a problem as given by Equation 2.17. Note that the number of variables and constraints have changed.

$$\begin{aligned} \max_{\Pi} f_0(\Pi) &= \mathbf{b}^T \Pi \\ \text{s.t. : } g_j(\Pi) : \mathbf{a}_j \Pi &\geq \mathbf{c} \quad j = 1, \dots, n \\ s_i &\geq 0 \quad i = 1, \dots, m \end{aligned} \quad (2.17)$$

From this example it is clear that dual functions have the advantage of being solvable within the dual space for which the dimensionality is lower leading to higher computationally efficiency. Based on the theory of the duality, it is known that solving problems in the space of primal variables  $\mathbf{x}$  is equivalent to maximizing a function that depends on the Lagrangian multipliers  $\lambda_j$  as proven above. The general expression for the Lagrangian in numerical procedures is defined by Equation (2.18) [14].

$$L(x, \lambda) = f(x) - \lambda^T h(x) \quad (2.18)$$

Note that analytically, the Lagrangian multipliers  $\lambda_j$  are determined based on the necessary condition for a minimum. This condition states that to achieve a minimum, the gradient of the objective must be a linear combination of the gradients of the constraints, at this minimizing point. The actual Lagrangian multipliers are those who satisfy Equation (2.19). The ones that do not satisfy the stationary conditions are to be considered Lagrangian multiplier estimates.

$$\nabla f(x_+) + \lambda^T \nabla h(x_+) = \mathbf{0}^T. \quad (2.19)$$

Finally the solution to the primal problem can be given by

$$\max_{\underline{x}_i \leq x_i \leq \bar{x}_i} (\min_{\lambda} L(x, \lambda)) \quad (2.20)$$

### 2.2.3 Approximations

Approximation concepts for structures were introduced by Schmidt in 1976 [17]. They were developed since, up to that point, many optimization approaches lead to an excess of one or more of the issues mentioned below.

1. Too many independent design variables
2. Too many behaviour constraints are considered throughout the entire process.
3. Too many structural analyses

In 1985 Braibant and Fleury [18] stated that the goal for an approximation concepts approach is to replace the primary optimization by a sequence of explicit sub-problems by use of:

- Coordinate use of design variable linking
- Temporary constraint deletion
- Construction of high-quality explicit approximations for retained constraints.

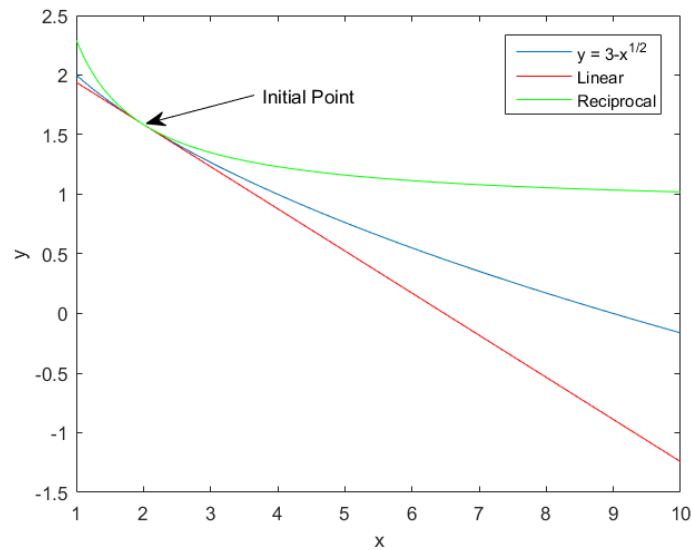
The two most basic approximations are described below. [19]

**Linear approximation** A linear approximation is simply the basic Taylor series expansion as given by Equation (2.21). It uses the information of an initial point in combination with its gradient to determine a linear approximation of the function such that a new point can be estimated based on the step length.

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^n (x_i - x_i^0) \frac{\delta f(x^0)}{\delta x_i} \quad (2.21)$$

**Reciprocal approximation** The reciprocal approximation can be seen as an adjusted form of the general Taylor series expansion. However instead of expanding the function with respect to  $x$  it is expanded to  $1/x$ . This expansion leads to Equation (2.22).

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^n (x_i - x_i^0) \frac{x_i^0}{x_i} \frac{\delta f(x^0)}{\delta x_i} \quad (2.22)$$



**Figure 2.7:** Linear and reciprocal approximation methods

Both approximations are presented in Figure 2.7. Here you can see that if the function  $y$  would represent a constraint, where above is feasible and below is infeasible, the reciprocal approximation would lead to a feasible point while the linear approximation would lead to an infeasible point. However if the gradient would be positive, the reciprocal approximation would deviate from the constraint faster than the linear approximation.

## 2.2.4 Optimization methods

One of the initial optimisation methods that was applied in structural optimization programs is the ConLin method introduced by Fleury in 1989 [6]. This method formed the basis of the well known method of moving asymptotes developed by Svanberg [13]. The globally convergent version of the method of moving asymptotes eventually resulted in a new class of methods called sequential convex programming. Additionally, another class under the name of Sequential Quadratic Programming was developed in the 1970's. All these different methods are addressed in this section.

### ConLin method

ConLin [6] is a convex linearisation algorithm developed by C.Fleury in 1989 and was one of the pioneering optimization algorithms for structural applications. ConLin is an extension on the approximation concepts approach. It performs a linearisation process with respect to

mixed variables, either direct or reciprocal for each function independently. The optimizer itself will select the appropriate approximation scheme based on the signs of the derivatives at each successive iteration point. Furthermore, the algorithm has the inherent tendency to steadily improve the feasible designs. Constraint relaxation is performed on violated constraints to cope with infeasible starting points.

The linearisation scheme used in the ConLin algorithm is presented in Equation (2.23) and is normalized to the unity of the current point  $x^0$  in Equation (2.24). Note that within this equation  $\sum_+$  and  $\sum_-$  indicate the summation of the terms where  $f_i$  is positive and negative respectively. Therefore this approximation carries out a direct linearisation over the positive set and a reciprocal linearisation over the negative set resulting in a conservative approximation.

$$f(x) = f(x^0) + \sum_+ f_i^0 (x_i - x_i^0) - \sum_- (x_i^0)^2 f_i^0 \left( \frac{1}{x_i} - \frac{1}{x_i^0} \right) \quad (2.23)$$

$$f(x') = f(x^0) + \sum_+ f_i' (x_i' - 1) - \sum_- f_i' \left( \frac{1}{x_i'} - 1 \right) \quad (2.24)$$

Eventually applying the linearisation technique to each function provides the following explicit sub-problem presented in Equation 2.25. Note that the superscript ' has been dropped to create a simpler representation.

$$\begin{aligned} \min \quad & \sum_+ f_{i0} x_i - \sum_- \frac{f_{i0}}{x_i} - \bar{f}_0 \\ \text{s.t.} \quad & \sum_+ g_{ij} x_i - \sum_- \frac{g_{ij}}{x_i} - \bar{g}_j \quad (j = 1, \dots, m) \\ & \underline{x}_i \leq x_i \leq \bar{x}_i \quad (i = 1, \dots, n) \end{aligned} \quad (2.25)$$

This problem can eventually be solved using a dual method approach.

### Method of moving asymptotes

The method of moving asymptotes (MMA) is a generalised version of Fleury's ConLin [6] and was initially established by Svanberg [13] in 1987. For this method the functions  $f_i^k$  are defined by Equation (2.26). Here  $L_j^k$  and  $U_j^k$  are the lower and upper asymptotes respectively so that  $L_j^k < x_j^k < U_j^k$ .

$$f_i^k = r_i^k + \sum_{j=1}^n \left( \frac{q_{ij}^k}{U_j^k - x_j} + \frac{p_{ij}^k}{x_j - L_j^k} \right) \quad (2.26)$$

where

$$p_{ij}^k = \begin{cases} (U_j^k - x_j)^2 \delta f_i / \delta x_j & \text{if } \delta f_i / \delta x_j > 0 \\ 0 & \text{if } \delta f_i / \delta x_j \leq 0 \end{cases} \quad (2.27)$$

$$q_{ij}^k = \begin{cases} 0 & \text{if } \delta f_i / \delta x_j \geq 0 \\ -(x_j - L_j^k)^2 \delta f_i / \delta x_j & \text{if } \delta f_i / \delta x_j < 0 \end{cases} \quad (2.28)$$

When filling in the coefficients  $p_{ij}^k$  and  $q_{ij}^k$  and writing them in a similar way as the ConLin method described by Equation (2.23) leads to Equation (2.29). From this equation it is clear that now both terms for positive and negative gradient summations possess a linear and reciprocal part. This is why the MMA is said to be a generalised version of ConLin

$$f_i^k = r_i^k \sum_{+} \frac{(U_j^k - x_j)^2}{x_j - L_j^k} \frac{\delta f_i}{\delta x_j} - \sum_{-} \frac{(x_j - L_j^k)^2}{U_j^k - x_j} \frac{\delta f_i}{\delta x_j} \quad (2.29)$$

The second derivatives of  $f_i^k$  are given by Equation (2.30).

$$\frac{\delta^2 f_i^k}{\delta x_j^2} = \frac{2p_{ij}^k}{(U_j^k - x_j)^3} + \frac{2q_{ij}^k}{(x_j - L_j^k)^3} \quad (2.30)$$

Some general rules for changing the variables  $L_j^k$  and  $U_j^k$  are:

- for an oscillating process, then move the asymptotes closer to the current iteration point as an attempt to stabilise the process.
- For a monotone but slow process, the asymptotes need to be relaxed.

To solve the sub-problem posed by the MMA it is thrown into a dual statement for which the lagrangian is given by Equation (2.31)

$$l_j(x, y) = \frac{p_{0j} + \lambda^T \mathbf{p}_j}{U_j - x_j} + \frac{q_{0j} + \lambda^T \mathbf{q}_j}{x_j - L_j} \quad (2.31)$$

When in the first iteration, the starting point was badly chosen, artificial variables are proposed by Svanberg [13] such that the sub-problem is still solvable. In that case, the sub-problem becomes:

$$\sum_{j=1}^n \left( \frac{p_{0j}}{U_j - x_j} + \frac{q_{0j}}{x_j - L_j} \right) + \sum_{i=1}^m (d_i z_i + d_i z_i^2) + r_0 \quad (2.32)$$

In 1993, Zillober [20] showed that an MMA together with a line search subjected to an augmented lagrangian provides a globally convergent version of the MMA. Where the augmented lagrangian is shown in Equation (2.33) [21]. This method is known as Sequential Convex Programming (SCP) and is further discussed in the next section.

$$\Phi_r \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = f(\mathbf{x}) + \sum_{j=1}^m \begin{cases} u_j h_j(\mathbf{x}) + \frac{r}{2} h_j^2(\mathbf{x}) & \text{if } -\frac{u_j}{r} \leq h_j(\mathbf{x}) \\ -\frac{u_j^2}{2r} & \text{otherwise} \end{cases} \quad (2.33)$$

In 2002, Svanberg who originally developed the method of moving asymptotes, also proposed a globally convergent version of the MMA (GCMMA) [22]. This method was based on conservative convex separable approximations (CCSA). The main difference between the original MMA and the GCMMA is that the introduction of CCSA imposed inner iterations to be made. Within these inner iterations, the curvature of the approximating functions is updated until they become conservative. This conservatism takes away the need for any linesearch and provides global convergence to the MMA itself. These two versions of Svanberg were implemented in MATLAB in 2007 [23].

The GCMMA of Svanberg [22] uses the approximating functions described by Equation (2.34) with the coefficients  $p_{ij}^{(k,l)}$ ,  $q_{ij}^{(k,l)}$  and  $r_i^{(k,l)}$  determined by Equations (2.35) to (2.37)

$$f_i^{(k,l)}(x) = \sum_{j=1}^n \left( \frac{p_{ij}^{(k,l)}}{U_j^k - x_j} + \frac{q_{ij}^{(k,l)}}{x_j - L_j^k} \right) + r_i^{(k,l)} \quad (2.34)$$

$$p_{ij}^{(k,l)} = (\sigma_j^k)^2 \max \left\{ 0, \frac{\delta f_i}{\delta x_j}(x^k) \right\} + \frac{p_i^{(k,l)} \sigma_j^k}{4} \quad (2.35)$$

$$q_{ij}^{(k,l)} = (\sigma_j^k)^2 \max \left\{ 0, -\frac{\delta f_i}{\delta x_j}(x^k) \right\} + \frac{p_i^{(k,l)} \sigma_j^k}{4} \quad (2.36)$$

$$r_i^{(k,l)} = f_i(x^k) - \sum_{j=1}^n \frac{p_{ij}^{(k,l)} + q_{ij}^{(k,l)}}{\sigma_j^k} \quad (2.37)$$

### Sequential convex programming

Sequential Convex Programming (SCP) also finds its origin in the CONLIN algorithm of Fleury [6]. As mentioned in Section 2.2.4 Zillober [24] established a SCP by adding a line search to the MMA algorithm. The same thing can be done using a thrust region method. Over the years SCP's have been proven to surpass most of the other optimization methods with respect to reliability and efficiency. Therefore, a lot of development with respect to this method has been made. In this section only the most state of the art methods are described. The main difference between all these methods is the way in which, the sub-problems are defined.

**SCP using interior point method** SCIP is a subroutine based on MMA and SCP using interior point method instead of the dual approach to solve the subproblem [25]. The interior point method used by Zillober is the one from Mehrotra [26], which was first implemented by Lustig et al. [5]. This method is called the predictor-corrector interior point method, which have been proven to be the most efficient for linear programming. [5]

The Method described by Mehrotra [26] is a power series variant of the primal-dual algorithm without considering explicit bounds [27]. Instead of using Newton's method, the logarithmic barrier Lagrangian, presented in Equation (2.38), is used to derive the first order Karush-Kuhn-Tucker conditions given in Equation (2.39).

$$\begin{aligned}
 & L_\mu(x, y, c, r, s, t, d_r, d_s, d_t) \\
 & = f(\tilde{x}) - \mu \sum_{j=1}^m \ln r_j - \mu \sum_{i=1}^m \ln s_i - \mu \sum_{i=1}^m \ln t_i \\
 & + y^T (g(\tilde{x}) + c) + d_r^T (-c + r) + d_s^T (\underline{x}' - x + s) + d_t^T (x - \overline{x}' + t)
 \end{aligned} \tag{2.38}$$

$$\left. \begin{aligned}
 \nabla_x : \quad & \nabla f(\tilde{x}) + Jy - d_s + d_t = 0 \\
 \nabla_y : \quad & g(\tilde{x}) + c = 0 \\
 \nabla_c : \quad & y - d_r = 0 \\
 \nabla_r : \quad & d_r - \mu R^{-1} e = 0 \\
 \nabla_s : \quad & d_s - \mu S^{-1} e = 0 \\
 \nabla_t : \quad & d_t - \mu T^{-1} e = 0 \\
 \nabla_{d_r} : \quad & -c + r = 0 \\
 \nabla_{d_s} : \quad & \underline{x}' - x + s = 0 \\
 \nabla_{d_t} : \quad & x - \overline{x}' + t = 0
 \end{aligned} \right\} = b_\mu := \begin{pmatrix} b_x \\ b_y \\ b_c \\ b_{\mu,r} \\ b_{\mu,s} \\ b_{\mu,t} \\ b_{d_r} \\ b_{d_s} \\ b_{d_t} \end{pmatrix} \tag{2.39}$$

Where  $\tilde{g}$  represents the inequality constraints, which are turned into equality constraints by the slack variable  $c$ .  $y, d_r, d_s, d_t$  are the dual variable vectors corresponding to the different sets of equality constraints.  $s$  and  $t$  are slack variables put on the inequality constraints specifying the boundaries of domain of  $x$ . Finally,  $r$  is a slack variable coupled to  $c$  as it is used to enable  $c$  to become 0 if  $r$  is not used, which is possible, the dual variable  $y$  and  $c$  are to be positive. Note that the barrier terms are consisting of the slack variables. This is preferred since

the computational cost for barrier function depending on the slack variable is way lower than a barrier function dependent on a complicated constraint.

Also, R, S, and T are the diagonal matrices consisting of the elements of r, s and t respectively while J is defined as  $\nabla g(\tilde{x})^T$

Newton's method is applied to obtain the following set up Equation (2.40).

$$\begin{Bmatrix} \nabla_{xx}L & J & & & & & -I & I \\ J^T & & I & & & & & \\ & & & I & & & -I & \\ & & & & D_r & & R & \\ & & & & & D_s & & S \\ & & & & & & D_t & T \\ & & -I & I & & & & \\ -I & & & & I & & & \\ I & & & & & I & & \end{Bmatrix} \begin{Bmatrix} \Delta x \\ \Delta y \\ \Delta c \\ \Delta r \\ \Delta s \\ \Delta t \\ \Delta d_r \\ \Delta d_s \\ \Delta d_t \end{Bmatrix} = b_\mu \quad (2.40)$$

Where,  $\nabla_{xx}L = \nabla^2 f(\tilde{x}) + \frac{d}{dx}Jy$ . The second term on the righthandside of this equation is given by Equation (2.41).

$$\frac{d}{dx}Jy = \sum_{j=1}^m \frac{\delta^2 g_j(\tilde{x})}{\delta x_k \delta x_i} y_j \quad (2.41)$$

Know that for one particular component of the gradient of a constraint, Equation (2.42) is used for computational efficiency.

$$\frac{\delta \tilde{g}_j}{\delta x_i} \Big|_x = \begin{cases} \frac{\delta g_j}{\delta x_i} \Big|_x \frac{(U_i^k - x_i^k)^2}{U_i^k - x_i^k} & \text{if } \frac{\delta g_j}{\delta x_i} \Big|_x \geq 0 \\ \frac{\delta g_j}{\delta x_i} \Big|_x \frac{(x_i^k - L_i^k)^2}{x_i^k - L_i^k} & \text{if } \frac{\delta g_j}{\delta x_i} \Big|_x < 0 \end{cases} \quad (2.42)$$

For the predictor step, the system is solved without the terms including the homotopy parameter  $\mu$  in  $b_\mu$ . However, the predictor step is dependent on the definiteness and size of the system [7].

In corrector step the  $\mu$  terms are added again due to there part in the original gradient of the Lagrangian. In addition to this, the terms of Equation (2.43) are added to the right hand side of Equation (2.40), where the underlined values are those predicted by the predictor step. These terms represent the non-linear parts of the equations.



$$\begin{aligned}
b_x &: -\left(\sum_{j=1}^m \frac{\delta^2 \bar{g}_j}{\delta x_k^2} \Delta x_k \Delta y_j\right)_{k=1,\dots,n}, \\
b_{\mu,r} &: -\underline{\Delta r \Delta d_r}, \\
b_{\mu,s} &: -\underline{\Delta s \Delta d_s}, \\
b_{\mu,t} &: -\underline{\Delta t \Delta d_t},
\end{aligned} \tag{2.43}$$

**SCP using two-point approximations** This method uses the information of the two previous points rather than just one point as used in all previously mentioned methods. This was first established by Fadel in 1990 [28] who proposed a two-point exponential approximation (TPEA). His work combined the ideas of Haftka [29] and Prasad [30] who suggested the use of previous points and exponential approximations respectively. Fadel's approximation is presented by Equation (2.44) where  $p_i$  is determined to match the derivatives at the previous point and is given by Equation (2.45).

$$f(\mathbf{X}) = f(\mathbf{X}_0) + \sum_i [(x_i/x_{0i})^{p_i} - 1] \left(\frac{x_{0i}}{p_i}\right) \frac{\delta f(\mathbf{X}_0)}{\delta x_i} \tag{2.44}$$

$$p_i = 1 + \log \left( \left[ \frac{\delta f(X_1)/\delta x_i}{\delta f(X_0)/\delta x_i} \right] \right) / \log \left( \frac{x_{1i}}{x_{0i}} \right) \tag{2.45}$$

$p_i$  is limited to -1 or +1. This limitation was removed by Wang and Grandhi [31] which lead to the TPEA-change method.

Further work of Wang and Grandhi is the two point adaptive non-linear approximation (TANA) method using adaptive intervening variables [31]. The TANA series exist of 3 versions TANA-1, TANA-2 and TANA-3 [32] [33]. These methods are presented by Equations (2.46) to (2.48) respectively.

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}_1) + \sum_{i=1}^n \frac{\delta f(\mathbf{x}_1)}{\delta x_i} \frac{x_{i,1}^{1-p_i}}{p_i} (x_i^{p_i} - x_{i,1}^{p_i}) + \varepsilon_1 \tag{2.46}$$

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}_2) + \sum_{i=1}^n \frac{\delta f(\mathbf{x}_2)}{\delta x_i} \frac{x_{i,2}^{1-p_i}}{p_i} (x_i^{p_i} - x_{i,2}^{p_i}) + \frac{1}{2} \varepsilon_2 \sum_{i=1}^n (x_i^{p_i} - x_{i,2}^{p_i})^2 \tag{2.47}$$

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}_2) + \sum_{i=1}^n \frac{\delta f(\mathbf{x}_2)}{\delta x_i} \frac{x_{i,2}^{1-p_i}}{p_i} (x_i^{p_i} - x_{i,2}^{p_i}) + \frac{1}{2} \varepsilon_3(\mathbf{x}) \sum_{i=1}^n (x_i^{p_i} - x_{i,2}^{p_i})^2 \tag{2.48}$$

Where, for TANA-1,  $\varepsilon_1$  is a constant; For TANA-2 the Hessian's diagonal elements only have the value  $\varepsilon_2$  and for TANA-3 the variable  $\varepsilon_3$  is defined by Equation (2.49).

$$\varepsilon_3(\mathbf{x}) = \frac{H}{\sum_{i=1}^n (x_i^{p_i} - x_{i,1}^{p_i})^2 + \sum_{i=1}^n (x_i^{p_i} - x_{i,2}^{p_i})^2} \tag{2.49}$$

In 2001 Kim et al. [34] established the two-point diagonal quadratic approximation (TDQA) method. This method uses the intervening variables presented in Equation (2.50). Where  $c_i$  is the shifting level and  $p_i$  is determined by Equation (2.51)

$$y_i = (x_i + c_i)^{p_i} \quad (2.50)$$

$$p_i = 1 + \ln \left[ \frac{\delta f(\mathbf{x}_1)}{\delta x_i} / \frac{\delta f(\mathbf{x}_2)}{\delta x_i} \right] / \ln \left[ \frac{x_{i,1} + c_i}{x_{i,2} + c_i} \right] \quad (2.51)$$

Eventually the approximation as defined by Kim et al. [34] is shown in Equation (2.52).  $G_i$  is the  $i$ th component of the diagonal Hessian matrix and is computed using Equation (2.53).

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}_2) + \sum_{i=1}^n \frac{\delta f(\mathbf{x}_2)}{\delta y_i} (y_i - y_{i,2}) + \frac{1}{2} \eta \sum_{i=1}^n G_i (y_i - y_{i,2})^2 \quad (2.52)$$

$$G_i = \frac{1}{2(y_{i,1} - y_{i,2})} \left( \frac{\delta f(\mathbf{x}_1)}{\delta y_i} - \frac{\delta f(\mathbf{x}_2)}{\delta y_i} \right) \quad (2.53)$$

Kim and Choi elaborated on this method by establishing an enhanced TDQA (eTDQA) method in 2008 [35]. The approximation used within this method is specified as Equation (2.55). Here  $p_i$  is again determined by equation Equation (2.51) and  $G_i$  by Equation (2.54). The correction coefficient  $\eta_e$  is computed using Equation (2.57) and  $H_i$  is defined by Equation (2.56) to avoid the denominator of the quadratic correction term to be 0.

$$G_i = \begin{cases} \frac{1}{(y_{i,1} - y_{i,2})} \left( \frac{\delta f(\mathbf{x}_1)}{\delta y_i} - \frac{\delta f(\mathbf{x}_2)}{\delta y_i} \right) & \text{if } [\delta f(\mathbf{x}_1)/\delta x_i] \cdot [\delta f(\mathbf{x}_2)/\delta x_i] \leq 0 \\ 0 & \text{otherwise} \end{cases} \quad (2.54)$$

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}_2) + \sum_{i=1}^n \frac{\delta f(\mathbf{x}_2)}{\delta y_i} (y_i - y_{i,2}) + \frac{1}{2} \sum_{i=1}^n G_i (y_i - y_{i,2})^2 \dots \\ + \frac{1}{2 \sum_{i=1}^n H_i (y_i - y_{i,1})^2 + \sum_{i=1}^n H_i (y_i - y_{i,2})^2} \eta_e \sum_{i=1}^n H_i (y_i - y_{i,2})^2 \quad (2.55)$$

$$H_i = \begin{cases} G_i & \text{if } [\delta f(\mathbf{x}_1)/\delta x_i] \cdot [\delta f(\mathbf{x}_2)/\delta x_i] \leq 0 \\ 1 & \text{otherwise} \end{cases} \quad (2.56)$$

$$\eta_e = 2 \left[ f(\mathbf{x}_1) - f(\mathbf{x}_2) - \sum_{i=1}^n \frac{\delta f(\mathbf{x}_2)}{\delta y_i} (y_{i,1} - y_{i,2}) - \frac{1}{2} \sum_{i=1}^n G_i (y_{i,1} - y_{i,2})^2 \right] \quad (2.57)$$

Knowing all this, the derivatives of the objective function with respect to  $x$  can be derived by Equation (2.58).

$$\frac{\delta \tilde{f}(\mathbf{x})}{\delta x_i} = \left[ \eta_e H_i \left[ \frac{(y_i - y_{i,2}) \sum_{j=1}^n H_j (y_i - y_{i,1})^2 - (y_i - y_{i,1}) \sum_{j=1}^n H_j (y_i - y_{i,2})^2}{\sum_{j=1}^n H_j (y_i - y_{i,1})^2 + \sum_{j=1}^n H_j (y_i - y_{i,2})^2} \right] \dots \right. \\ \left. + \frac{\delta f(\mathbf{x}_2)}{\delta y_i} + G_i (y_i - y_{i,2}) \right] p_i (x_i + c_i)^{p_i-1} \quad (2.58)$$

The eTDQA method developed by Kim[35] was implemented into the SCP environment by Park et al. in 2014 [36]. To solve this in the dual environment of Falk [37], strict convexity of the approximate objective function and constraints is necessary. This was enforced by adjusting the diagonal Hessian terms by dividing it into two parts as indicated by Equations (2.59) to (2.61)

$$g_{i,j} = P_1 + P_2 \quad (2.59)$$

$$P_1 = \frac{p_i - 1}{x_{i,2} + c_i} \left( \frac{\delta f(\mathbf{x}_2)}{\delta x_i} \right) \quad (2.60)$$

$$P_2 = \max \left( \varepsilon, \left[ \frac{\eta_e H_i}{\sum_{l=1}^n H_l (y_{l,2} - y_{l,1})^2} + G_i \right] (p_i (x_{i,2} + c_i)^{p_i-1})^2 \right) \quad (2.61)$$

Since the Hessian has to be positive definite, and  $\varepsilon$  is a small positive value, certain conditions, specified by Equation (2.62) apply for  $P_1$ .

$$If \begin{cases} \frac{\delta f}{\delta x_i} < 0 \Rightarrow & p_i < 1, \text{ then } P_1 \geq 0 \\ \frac{\delta f}{\delta x_i} \geq 0 \Rightarrow & p_i > 1 \end{cases} \quad (2.62)$$

More recent, in 2012 Groenwold [38] established an SQP type method from the SCP class using approximated approximations [39].

### Sequential quadratic programming

Sequential quadratic programming (SQP) is a class of methods that came to existence in the 1970's. It is a general purpose method to solve smooth non-linear optimization problems. These should not be too large, well scaled and of the functions and gradients can be determined with a sufficient quality[40]. It was established from extensive comparative numerical tests[41]. Convergence of the method was proven by Schittkowski [42]. Equation (2.63)

shows the quadratic programming problem as specified by Schittkowski [40], where  $B_k$  is the approximation of the Hessian of the Lagrangian function.

$$\begin{aligned} \min_{d \in \mathbb{R}} \quad & \frac{1}{2} d^T B_k d + \nabla f(x_k)^T d \\ \text{s.t.} \quad & \nabla f_j(x_k)^T d + f_j(x_k) = 0, \quad j = 1, \dots, m_e \\ & \nabla f_j(x_k)^T d + f_j(x_k) \geq 0, \quad j = 1, \dots, m \end{aligned} \quad (2.63)$$

From this problem, Equation (2.64) can be derived for obtaining the next iterate where  $d_k$  is the optimal solution and  $u_k$  is the corresponding multiplier.

$$\begin{pmatrix} x_{k+1} \\ v_{k+1} \end{pmatrix} = \begin{pmatrix} x_k \\ v_k \end{pmatrix} + \alpha_k \begin{pmatrix} d_k \\ u_k - v_k \end{pmatrix} \quad (2.64)$$

$\alpha_k$  is any value between 0 and 1 that provides a suitable step length. To enforce global convergence, this step length should provide a sufficient decrease in a merit function  $\phi_r(\alpha)$  given by Equation (2.65). Within this equation  $\psi_r(x, v)$  is a penalty function. Implementing this penalty function leads to the augmented Lagrangian function presented by Equation (2.33). Do note that the Lagrangian given there is for inequality constraints of the type  $f_j(x) \leq 0$  where for this problem  $f_j(x) \geq 0$  is used.

$$\phi_r(\alpha) := \psi_r \left( \begin{pmatrix} x \\ v \end{pmatrix} = \begin{pmatrix} x \\ v \end{pmatrix} + \alpha \begin{pmatrix} d \\ u - v \end{pmatrix} \right) \quad (2.65)$$

### 2.2.5 Starting point

In order to have an idea of which optimization methods and approximation methods to use in which situation, they have to be tested and compared. This is usually done using numerical examples such as the popular 10-bar truss problem and many more. In this section, a comparison is presented between the methods described in Section 2.2.4. Initially, the methods themselves are compared. After which, the efficiency of the approximation methods is investigated. The efficiency of the sub-problem solvers, which are required to solve the approximated sub-problems, are then discussed. Finally a starting point is defined.

#### Efficiency of the optimization methods

Initially Svanberg proved that the Method of Moving Asymptotes (MMA) was superior to the traditional methods in 1987 [13]. This was proved based on three different numerical problems, the cantilever beam problem, a 8-bar truss problem and a 2-bar truss problem. Later

on, the globally convergent version of the MMA (SCP) proved to be even more efficient as it ensured that the objective function improved at each consecutive step [20]. To prove this, the typical 10-bar truss problem was used.

The paper of Zillober [24] provides proof that SCP is more reliable and has better convergence properties than SQP in the CUTE [43] test environment. The SQP algorithm presented was the routine NLPQL [44], which has been proven to be very robust. Although, as suggested by Etman et al. [38], it is possible to achieve a SQP type-method based on the SCP method by implementation of the approximate approximation concept of Groenwold [39]. This paper suggests that the established method shows promise for the use in optimization problems where the number of design variables and constraints are high, based on the multilevel cantilever beam problem.[45]

### Efficiency of the approximation methods

As mentioned above, the approximated approximations concept was introduced by Groenwold in 2010 [39], which lead to the establishment of an SQP-type method from the SCP method. In this paper however a large comparison between approximation methods was made. The approximations at hand are:

|            |                                                                                       |
|------------|---------------------------------------------------------------------------------------|
| L:         | linear approximation.                                                                 |
| R:         | reciprocal approximation.                                                             |
| E:         | exponential approximation.                                                            |
| C:         | conservative approximation.                                                           |
| MMA:       | approximations from the method of moving asymptotes.                                  |
| T2:R:      | a quadratic approximation to the reciprocal approximation.                            |
| T2:E:      | a quadratic approximation to the exponential approximation.                           |
| T2:C:      | a quadratic approximation to the conservative approximation.                          |
| T2:MMA:    | a quadratic approximation to the approximations from the method of moving asymptotes. |
| T2:TANA-3: | a quadratic approximation to the TANA-3 approximation.                                |

Using the 10-bar truss problem and the 5 variate cantilever beam problem as tests, it becomes clear that the quadratic approximations of the original approximations outperform the original approximations. There are some discrepancies on some test problems but in general the above statement is correct.

In 2008 Kim et al. already showed that their revised version of the two-point diagonal quadratic approximation (eTDQA) was more efficient than their original.[35][34] In 2014 Park et al. proposed a scheme to enforce convexity onto the eTDQA algorithm and with it ensure global convergence.[36] They renamed it to gc-T2:eTDQA and proved based on the same typical problems such as the vanderplaats 5 variate cantilever beam problem that it is indeed globally convergent and it even outperforms the original eTDQA.

### **Efficiency of the sub-problem solver**

One of the sub-problem solvers is based on the principle of duality that was discussed in Section 2.2.2. The primal-dual method, basically uses dual variables to compute the step size for the next primal iteration. The other sub-problem solver, which is slightly discussed in Section 2.2.4, is the predictor-corrector interior point method. In the paper of Lustig et al. [5] it has been proven that the predictor-corrector interior point method is generally more efficient than the primal-dual method. This was based on 86 test cases presented by the NETLIB test set [46].

### **Definition starting point**

Based on the above trade-off, the starting point for this research should be the gc-T2:eTDQA approximation method in combination with an interior point sub-problem solver.

However it should also be noted that generally most of the approximation methods used in structural optimization originate from Fleury's ConLin method [6]. Since the initial goal is to set up the optimization with a direct link to FEM analysis for sensitivities, a sturdy base is required. Therefore, the complexity of the initial set-up needs to be reduced although this will have its influences on efficiency.

Eventually, the approximation method from ConLin will be used in combination with Mehrotra's [26] predictor-corrector interior point method as presented by Zillober [7]. When a good base is formed and the validity of the concept is proven, it will pave the way to improvement.



---

## Chapter 3

---

# Model definition

The model is defined using MATLAB version 2015 to generate a bdf file, which is inserted into the NASTRAN solver version 2010. Therefore, first the initial 3D model and its transformation to 2D are described in Section 3.1.

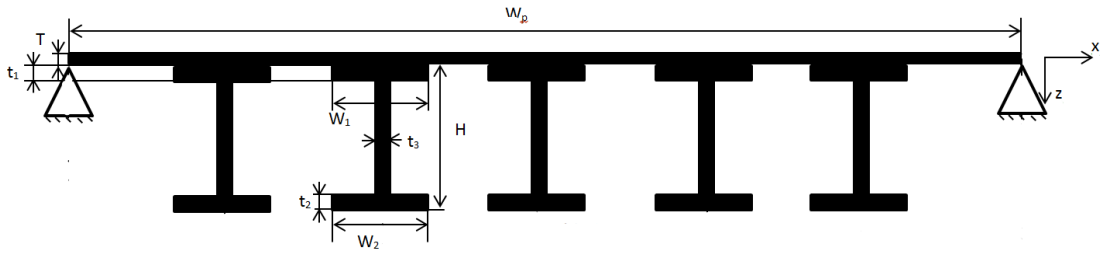
The 2D mode is then implemented into NASTRAN. This however does not come without any complications such as the definition of the NASTRAN cards and the element types. All of this is discussed in Section 3.2. Generating accurate results from a FEM analysis requires a refined mesh. Since computational effort does make a huge difference in optimization, a mesh convergence study is performed. Section 3.3 presents the results of this study, such that a trade-off between accuracy and sparsity of the mesh can be performed. Finally, in Section 3.4 the basic model is verified with respect to analytical results.

### 3.1 Basic model

The initial setup model is shown in Figure 3.1, where the values for the dimensions and material properties have been defined as presented by Table 3.1. The model is a basic stiffened panel where the edges are restrained in x and z direction. The force as seen in the table is the total force, which is distributed by MPC's in the FEM setup. Note that this model is under-designed, as shown by analysis later. This to ensure that the global and panel buckling modes will occur for verification purposes.

Now that the basic 3D model is defined, it has to be converted to a finite element model. For





**Figure 3.1:** Test model for NASTRAN analysis

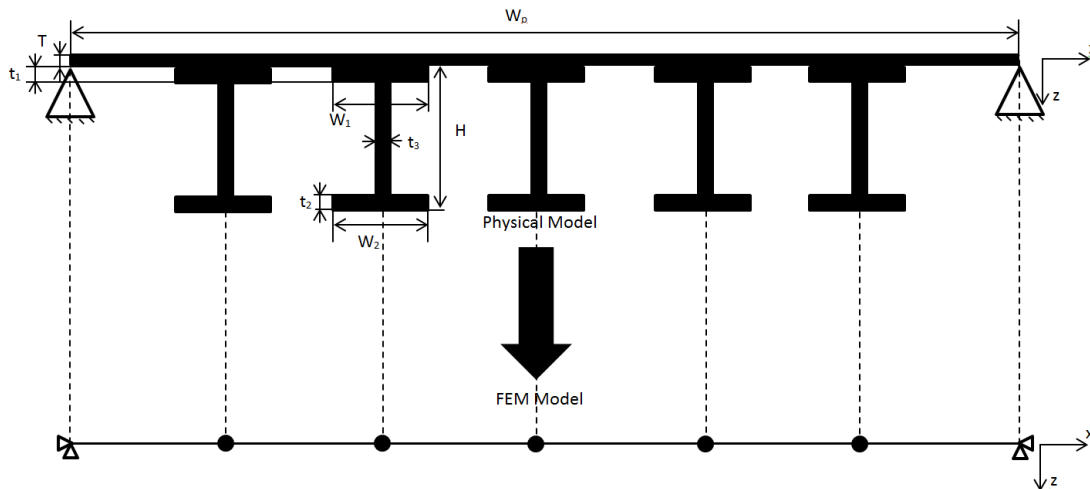
**Table 3.1:** Model Properties

| Dimensions            |      |      | Properties                                           |       |       |
|-----------------------|------|------|------------------------------------------------------|-------|-------|
| <u>Panel:</u>         |      |      | <u>Material:</u>                                     |       |       |
| $W_p$                 | 1000 | [mm] | E                                                    | 71700 | [MPa] |
| $L_p$                 | 1000 | [mm] | Nu                                                   | 0.33  | [-]   |
| T                     | 3    | [mm] | $\sigma_y$                                           | 503   | [Mpa] |
| <u>Stiffeners(5):</u> |      |      | <u>Load:</u>                                         |       |       |
| H                     | 26   | [mm] | $F_{y_{tot}}$                                        | 160   | [kN]  |
| $W_2$                 | 8    | [mm] | <u>Constraints:</u><br>All Edges      xz translation |       |       |
| $W_1$                 | 8    | [mm] |                                                      |       |       |
| $t_3$                 | 2    | [mm] |                                                      |       |       |
| $t_2$                 | 2    | [mm] |                                                      |       |       |
| $t_1$                 | 2    | [mm] |                                                      |       |       |

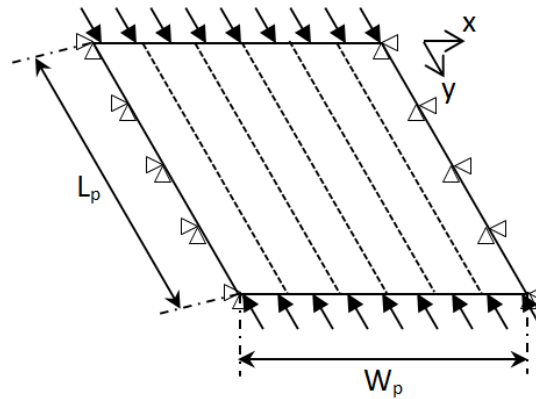
simplifications the model is therefore transformed from 3D to 2D as presented in Figure 3.2. The final unmeshed 2D model is shown in Figure 3.3, where the dotted lines represent the stiffeners.

Now the panel itself can be represented by shell elements and the stiffeners by bar elements. Doing this has major advantages for the optimization to come. Because for a 2D model, the properties of the stiffener can be adjusted without having to adjust the model itself. Meaning that only the properties cards have to be adjusted.

The goal is to use NASTRAN for analysis, which means a bdf file is required to represent the model. This can either be done by directly extracting a file from a NASTRAN GUI such as PATRAN or by defining it manually. Since the bdf file should be adjustable for the optimization, it is written by setting up a data structure in MATLAB, which is easily converted to a bdf file.



**Figure 3.2:** Conversion from 3D model to 2D FEM



**Figure 3.3:** Conversion from 3D model to 2D FEM

## 3.2 Model translation to NASTRAN

The model as shown in Figure 3.1 needs to be converted to a finite element model and used for optimization. The basic setup of the bdf for the optimization is discussed in Section 3.2.1. As mentioned before, shell and bar elements are used to define the panel and stiffeners respectively. Except for that, nodes, properties, constraints and loads are defined. Furthermore, since the eventual goal is to use this set-up for optimization, design variables and responses are defined. The NASTRAN input cards for this are listed in Section 3.2.2. Additionally, a choice between the two different property cards for bar elements is made in Section 3.2.3.

### 3.2.1 Basic information for NASTRAN bdf setup

A basic bdf file consists of two major parts, the case control section and the bulk data section. The first specifies the solver, type of analyses, active load cases and the requested output. The bulk data entries define the element types and properties, loads and constraints, optimization specific information, etc.

NASTRAN cards are used as input for all bulk data. The format for these cards consist of 10 columns of which 9 are used for card entries and one for continuations symbols. Each column consists 8 digits [47]. Some entries require integers, some characters and some real values. If an integer has to be specified it is critical no decimals are noted e.g. 1.0 is not allowed. A function was written, outside the scope of this thesis to write cards within the correct format. This was partially rewritten with respect the data structure used to set up the stiffened panel problem. Additional cards had to be implemented, such as PBARL, PBAR, CBAR, MDLPRM, MPC1 and MPCADD, since these were not yet accounted for within the provided script [47].

The model is analysed using SOL200, which specifies an optimization solution. Since the optimization process itself is programmed into MATLAB only the sensitivities are required. Therefore the command DSAPRT is added within the case control section to only compute sensitivities. Sub-cases are defined for static and buckling analysis such that the required responses can be computed. A full overview of the case control section is presented in Appendix A.

Once all sub-cases are defined, including active load cases and constraints, the bulk data entries are loaded. Finally as the entire BDF is defined it is send to NASTRAN using the following code:

```
!C:\Users\MDeklerck\MSC.Software\MD_Nastran\20101\bin\mdnastran  
SOL_200_PanelOptim.bdf SCR=YES old=no
```

Note that the requested output format has been altered for the extraction of the correct sensitivities and responses to MATLAB. This is possible by inclusion of alteration files within the bdf.

### 3.2.2 Required cards

The required cards for the bulk data setup are presented below. These includes a small description of its function within the bdf file.

- Model

GRID: Defines the nodes required for element creation

CQUAD4: Defines the shell elements representing the panel

CBAR: Defines bar elements representing the stiffener and determines the actual position of the stiffener and its orientation

MDLPRM: Defines the offset definition of the bar elements by addition of "OFFDEF".

- Properties

MAT1: Defines the isotropic material to be used for the panel and stiffener

PBARL: Defines the stiffener based on stiffener type and dimensions as input

PBAR: Defines the stiffener based on property input such as area, moment of inertia,...

PSHELL: Defines the properties of the shell elements, which represent the panel

- Load case

SPC1: Defines a single point constraint on a node

SPCADD: Creates a set of SPC containing all SPCs for constraint selection

MPC: Defines an explicit multi-point constraint linking degrees of freedom of a dependent node to those of an independent node

MPCADD: Creates a set of MPC containing all MPCs for constraint selection

FORCE: Defines a point force on a grid point

LOAD: Creates a set of forces, moments,... containing all loads on the model for load selection

EIGRL: Defines the amount of roots requested for the eigenvalue analysis and is required to perform buckling analysis

- Optimization

DESVAR: Defines the design variables for optimization

DVPREL1: Defines the relation between the design variables and property inputs

DRESP1: Defines the responses for the optimization

DCONSTR: Defines the upper and lower boundaries on the design variables

Some additional remarks for these cards are necessary to fully understand the NASTRAN set-up. The bar elements themselves for instance are defined in plane with the shell elements. However, in order to represent the physical model, an offset needs to be determined such that the beam is located beneath the panel instead of centred in the middle of it. The value of this offset is defined in the CBAR card.

Except for the value also the method in which NASTRAN should interpret is critical. MDLPRM is used to define exactly this. Now the default of this card specifies a fixed offset, which presents problems when initializing the buckling analyses as this requires the differential stiffness matrix. Therefore the offset is defined using "MDLPRM OFFDEF LROFF", which specifies large rotation offsets and enables the computation of the differential stiffness matrix.

On top of this is the issue of defining which property card to use for the bar elements. This problem will be discussed in more detail in Section 3.2.3. More information with respect to the definition of the design variables and responses is available in Chapter 4.

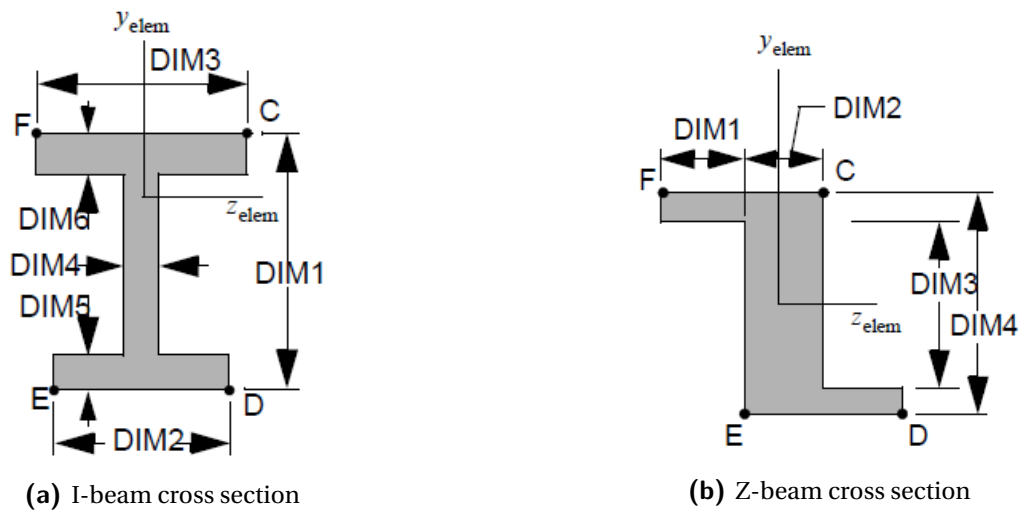
### 3.2.3 Defining stiffeners

There are several ways to represent a stiffener by use of a one-dimensional element. NASTRAN uses either CBEAM or CBAR. These elements require designated properties. The properties in turn can again be set up in several ways, which are PBEAM, PBEAML, PBAR and PBARL. As seen before, only bar elements are considered. Although beam elements present more possibilities, the model at hand is relatively simple and thus has no need for more elaborate types of elements. Now that this has been established, one question remains. Which type of bar element would present the most interesting opportunities?

**PBARL** The PBARL card in NASTRAN defines the properties of an element based on a cross section type and its dimensions. Within this thesis only two cross-sections are used, which are the basic I and Z-cross sections. These cross-sections including dimensions as required by NASTRAN [47] are presented in Figure 3.4.

This type of property definition is limited to the amount of cross-sections pre-described in NASTRAN and thus no free design can be defined by this property card. In addition to this there are only a limited number of free dimensions in some of the prescribed cases such as for the Z-stiffener. On the upside it does not produce the same amount of round-off errors as the PBAR property card.

**PBAR** The PBAR property card defines the geometrical properties of a certain element solely based on direct input. In other words the area, moments of inertia and rotational stiffness need to be hand in manually. To do this several equations are required, which are presented below. The variables  $cgz_i$  and  $cgx_i$  represent the location of the centre of gravity (CG) for the sub-parts of the stiffener as defined in Figure 3.5 with respect to the reference axes. Note that the middle of the web and the top of the stiffener are considered the x-axis and z-axis respectively in these calculations.  $A_i$  represents the area of the  $i^{th}$  sub-part,  $I$  represents the moment of inertia where the subscript defines the axis around which it is defined and  $J$  is



**Figure 3.4:** I and Z cross sections with relating dimensions for NASTRAN cars [47]

the rotational stiffness of the cross section.

**I-stiffener**

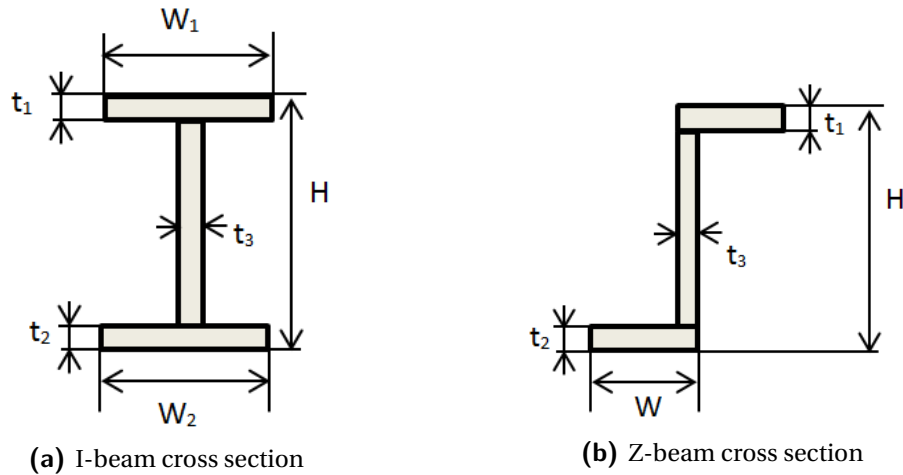
$$\begin{aligned}
A_1 &= W_1 t_1 \\
A_2 &= W_2 t_2 \\
A_3 &= (H - t_1 - t_2) t_3 \\
A &= \sum_i A_i \\
CGz &= \frac{\sum_i c g z_i A_i}{A} \\
I_{zz_1} &= \frac{W_1 t_1^3}{12} \\
I_{zz_2} &= \frac{W_2 t_2^3}{12} \\
I_{zz_3} &= \frac{t_3 (H - t_1 - t_2)^3}{12} \\
I_1 &= \sum_i I_{zz_i} + \sum_i (c g z_i - CGz)^2 A_i \\
I_{xx_1} &= \frac{t_1 W_1^3}{12} \\
I_{xx_2} &= \frac{t_2 W_2^3}{12} \\
I_{xx_3} &= \frac{(H - t_1 - t_2) t_3^3}{12} \\
I_2 &= \sum_i I_{xx_i} \\
J &= \frac{W_1 t_1^3 + W_2 t_2^3 + (H - \frac{t_1}{2} - \frac{t_2}{2}) t_3^3}{3}
\end{aligned} \tag{3.1}$$

**Z-stiffener**

$$\begin{aligned}
A_1 &= W t_1 \\
A_2 &= W t_2 \\
A_3 &= (H - t_1 - t_2) * t_3 \\
A &= \sum_i A_i \\
CGz &= \frac{\sum_i c g z_i A_i}{A} \\
CGx &= \frac{\sum_i c g x_i A_i}{A} \\
I_{zz_1} &= \frac{W_1 t_1^3}{12} \\
I_{zz_2} &= \frac{W_2 t_2^3}{12} \\
I_{zz_3} &= \frac{t_3 (H - t_1 - t_2)^3}{12} \\
I_1 &= \sum_i I_{zz_i} + \sum_i (c g z_i - CGz)^2 A_i \\
I_{xx_1} &= \frac{t_1 W^3}{12} \\
I_{xx_2} &= \frac{t_2 W^3}{12} \\
I_{xx_3} &= \frac{(H - t_1 - t_2) t_3^3}{12} \\
I_2 &= \sum_i I_{xx_i} + \sum_i (c g x_i - CGx)^2 A_i \\
J &= \frac{W t_1^3 + W t_2^3 + (H - t_1 - t_2) t_3^3}{3}
\end{aligned} \tag{3.2}$$

Since the dimensions can be chosen, the Z-stiffener is made slightly more controllable. The dimensions are linked as shown in Figure 3.5. This directly indicates one of the major advantages of this property card. It is independent on the shape of the cross section and the amount of different dimensions necessary to define it. This paves the way to great opportunities for optimization since it breaks with the bounds of the standard cross-section types and/or definitions in FEM packages. However the user himself does need to put in the effort to compute these properties. Additionally, analytical expressions for transforming the sensitivities from extracted to required design variables will be a necessity.

**Final definition of bar element** By exploring all possibilities it can be stated that a PBAR property definition should be used in order to freely design the stiffeners. This also gives



**Figure 3.5:** Tested stiffener cross sections and their design variables

room to test a new principle for optimization. By extracting the sensitivities with respect to the geometrical properties from FEM, they can be adjusted analytically to the desired design variables. Since the properties have already been computed, those same equations can be used to transform them to the required design variables.

The offset as requested by CBAR is the distance between the middle of the panel and the CG of the stiffener. This because all properties are determined with the CG as reference point.

### 3.3 Mesh convergence study

FEM results are very dependent on the mesh size. Therefore, mesh refinement is an important part of this process as it will lead to the accuracy needed for a decent optimization. The mesh density is defined based on the amount of nodes along the x ( $n_x$ ) and y axis ( $n_y$ ).  $n_x$  is determined based on the amount of nodes required for the sub-panels ( $n_{bs}$ ).

Buckling analysis is more mesh dependent than the static analysis for this load case. Therefore, the eventual mesh size is decided based on its convergence with respect to the eigenvalues. To find the optimal solution, the amount of nodes is changed according to a certain step size and the convergence is checked. Note that for both variables, only one is adjusted and the other remains the same to find the influence of the amount of nodes in a single direction. Once the value converged sufficiently, both converged values are used to create the final mesh density.

The variable of  $n_{bs}$  is chosen to check convergence instead of total nodes along the x-axis. This to ensure a steady accuracy for the buckling in between the stiffeners. The mesh densities



checked and their eigenvalues for different sets of  $n_{bs}$  and  $n_y$  is shown in Tables 3.2 and 3.3 respectively. The reduction is defined as the change in eigenvalue with respect to that of the mesh density of the row above.

The final selected mesh is an  $n_{bs}$  is 9 and  $n_y$  is 11. The selection of  $n_{bs}$  is most likely an exaggeration however, for low amounts of stiffeners, this higher value will become critical for buckling modes where more than one half wave appears in between stiffeners.

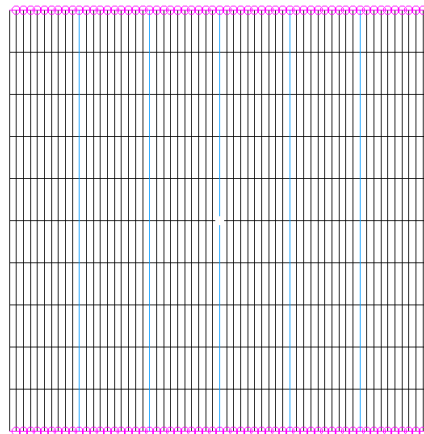
**Table 3.2:** Mesh convergence for  $n_{bs}$

| $n_{bs}$ | $n_y$ | Mode 1   | Mode 2   | Mode 3   | Reduction |
|----------|-------|----------|----------|----------|-----------|
| 3        | 9     | 0,518572 | 0,534381 | 0,640502 |           |
| 5        | 9     | 0,516455 | 0,530392 | 0,639374 | 0,41%     |
| 7        | 9     | 0,51576  | 0,529079 | 0,639004 | 0,13%     |
| 9        | 9     | 0,515456 | 0,528501 | 0,638844 | 0,06%     |
| 11       | 9     | 0,515302 | 0,528201 | 0,638762 | 0,03%     |

**Table 3.3:** Mesh convergence for  $n_y$

| $n_{bs}$ | $n_y$ | Mode 1   | Mode 2   | Mode 3   | Reduction |
|----------|-------|----------|----------|----------|-----------|
| 9        | 3     | 0,501631 | 0,513677 | 0,617174 |           |
| 9        | 5     | 0,504892 | 0,525101 | 0,616405 | -0,65%    |
| 9        | 7     | 0,511834 | 0,526779 | 0,632356 | -1,36%    |
| 9        | 9     | 0,515456 | 0,528501 | 0,638844 | -0,70%    |
| 9        | 11    | 0,517391 | 0,529601 | 0,642025 | -0,37%    |

The final FEM model is shown in Figure 3.6, where black and blue represent the shell and bar elements respectively. The pink edges are the applied MPC's.



**Figure 3.6:** 2D FEM model

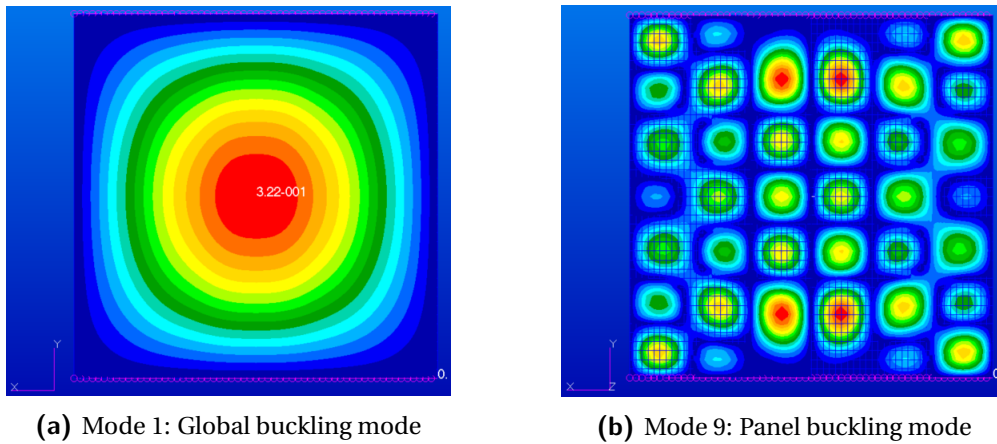
### 3.4 Model Verification

Now that the model has been defined in 2D, the stiffener's properties by property card PBAR and the mesh density is set to 9 nodes in between stiffeners and 11 across the length. All that remains is to verify the results. To do this, the model as described in Section 3.1 is used.

The model is verified solely based on buckling responses. If this is achieved, it is assumed that the stiffness matrix of the panel is correct. In that case, it can be assumed that also the stress is determined with sufficient accuracy. The validity of this assumption comes from the fact that the load case is relatively simple and thus the relation between stiffness and stress is considered linear. To minimize errors due to mesh density, it is increased ( $n_y=41$ ) for verification purposes.

However since the equations as provided in Section 2.1 are for a simply supported panel rather than a panel where the edges are constrained in x and z direction. The boundary conditions are therefore altered to those for simply supported panels.

From the initial analysis of the panel, it shows that most of the buckling modes are combined modes. To simplify things, two modes were selected for, which the buckling mode can be assumed either global buckling or pure panel buckling. These modes are shown in Figure 3.7. The two chosen buckling modes are the first and ninth. Although the ninth is not pure panel buckling, notice the difference in amount of half waves across the sub-panels, the buckling of a single sub-panel is relatively pure.



**Figure 3.7:** Nearly pure buckling modes for verification purposes

Now in Section 2.1 global buckling of a panel is defined by Equation (2.1). However this is under the assumption that it can be treated as column buckling. In other words, the edges along the direction of the force are free.

Since this is not the case an effective width ( $W_{eff}$ ) needs to be determined. This determines

the width of the panel that actually carries part of the load when global buckling initiates. The effective width is determined by Equation (2.2). Note that this is an equation for the effective width accounted for per stiffener so the total width of load carrying part is multiplied with the amount of stiffeners.

The load carrying part of the panel is now added to the stiffener and the whole thing is handled as if it is a column. The moment of inertia of the full structure is thus presented by Equation (3.3). Here the summation includes the effective panel sections and the CG is determined solely on the effective parts. The buckling of the sub-panel is determined by Equation (2.3) based on the middle part where the number of half waves, as presented by Figure 3.7b, are  $m=5$  and  $n=1$ .

$$I = \sum_i I_i + \sum_i (cg_i - CG)^2 A_i \quad (3.3)$$

The numerical and analytical results are presented in Table 3.4. It can be seen that there is a slight difference, less than 7%, between the two.

A deviation of 3,49% is seen for the global buckling. This value can partially be devoted to the round-off errors of the stiffener properties or offset input. Another influential part of course is that it is a 2D model. This includes that the geometrical properties are defined by a property card rather than the elements themselves.

For panel buckling the biggest deviation, of 6,58%, between analytical and numerical results is found. One factor for this is most likely the mesh density. Although it was increased for verification purposes, an even finer mesh should lead to an even more accurate result. Another factor is the fact that mode 9, which was selected for this purpose, closely represents pure panel buckling. In other words there is already a basic deviation to be expected since it is still a mixed panel buckling mode. On top of this, simply supported boundaries are assumed, while the stiffeners actually represent elastic boundaries instead. This should slightly increase the eigenvalue.

So in general, although there is a difference between the analytical and numerical results, there are plenty of influential factors that need to be considered before dismissing the model. After careful consideration of those factors, it can be stated that the accuracy of the model is sufficient and that it can be used for optimization purposes.

**Table 3.4:** Comparison between analytical and numerical results for buckling

|        | Buckling type | Analytical | Numerical | Difference |
|--------|---------------|------------|-----------|------------|
| Mode 1 | Global        | 0,532      | 0,513     | 3,49%      |
| Mode 9 | Panel         | 1,597      | 1,702     | 6,58%      |

---

## Chapter 4

---

# Model optimization

This section handles the optimization of the model. Several test cases will be presented to prove the following points.

- Show basic optimization capabilities on the model defined in Chapter 3
- Show its capability to handle different cross sections
- Show its capability to handle different property sets for stiffeners on the same panel
- Optimize model with respect to different amount of stiffeners finding the actual optimal panel

In order to do so, the basic optimization problem is presented in Section 4.1. After which, the process of the optimization is described in Section 4.2. Once the entire process is known, the test cases are represented in Section 4.3. Finally the results are discussed in Section 4.4.

### 4.1 Problem

In optimization the goal is to reach the best possible design within a set of constraints while minimizing a certain objective. In this case the optimization is performed for stress, buckling and design constraints while minimizing weight. Equation 4.1 represents the problem where

$\sigma$  is vector containing the axial stress in the bar elements and normal stress in the shell elements;  $\lambda$  is a vector containing 20 eigenvalues; **Dc** represents the design constraints for the stiffeners and  $x_i^{min}$  and  $x_i^{max}$  the minimum and maximum boundaries of the design variables (DVs) respectively. The objective is the volume, which is the area  $A(\mathbf{x})$  times the length of the panel  $L_p$ . Note that the area is a function of the DVs. The DVs are the stiffener dimensions and the panel thickness. The areas of the different stiffener cross-sections are determined using the equations provided in Section 3.2.3. The respective dimensions for each stiffener type are indicated in Figure 3.5.

$$\begin{aligned} \min_{\mathbf{x}} A(\mathbf{x})L_p \\ s.t. : \sigma \leq \sigma_y \end{aligned} \quad (4.1)$$

$$1/\lambda \leq 1 \quad (4.2)$$

$$\begin{aligned} \mathbf{Dc} \leq 1 \\ x_i^{min} \leq x_i \leq x_i^{max} \quad i = 1, \dots, n \end{aligned} \quad (4.3)$$

The design constraints are implemented such that no excessive ratio's between the dimensions of the stiffeners can occur. The implemented design rules are extracted from the paper of COLSON et al. [48] and are described in Table 4.1. This results in a total of 7 constraints where,  $T$  is the panel's thickness and the other dimensions are those as defined for the I-stiffener as given in Figure 3.5. Note that the same ratio's and boundaries apply for the Z-stiffener only there  $W_2$  should become  $W$ .

**Table 4.1:** Design rules

| Design rule                               | Constraint values                          |
|-------------------------------------------|--------------------------------------------|
| Attached flange ratio (AFR)               | $3 \leq \frac{26.8 + t_3}{t_1} \leq 20$    |
| Web ratio (WR)                            | $3 \leq \frac{H - t_1 - t_2}{t_3} \leq 20$ |
| Free flange ratio (FFR)                   | $3 \leq \frac{W_2}{t_2} \leq 10$           |
| Attached flange vs. skin thickness (AFSR) | $1.3 \leq \frac{t_1}{T}$                   |

The minimum and maximum values for the DVs are shown in Table 4.2.

## 4.2 Process

The entire optimization process is depicted in Figure 4.1. After the initial model has been established as explained in Chapter 3 the optimization is initiated. This section will serve as a

**Table 4.2:** Minimum and maximum dimensions for different stiffener types

| I-Stiff [mm] |   |    |      | Z-Stiff [mm] |   |    |      |
|--------------|---|----|------|--------------|---|----|------|
| 25           | ≤ | H  | ≤ 55 | 25           | ≤ | H  | ≤ 55 |
| 8            | ≤ | W2 | ≤ 26 | 8            | ≤ | W  | ≤ 26 |
| 8            | ≤ | W1 | ≤ 4  | 2            | ≤ | t3 | ≤ 4  |
| 2            | ≤ | t3 | ≤ 4  | 1,6          | ≤ | t2 | ≤ 4  |
| 1,6          | ≤ | t2 | ≤ 4  | 1,6          | ≤ | t1 | ≤ 4  |
| 1,6          | ≤ | t1 | ≤    |              |   |    |      |

guide through the process. First the analysis is performed and the sensitivities and responses are extracted. The first step is to adjust the sensitivities obtained from the analysis to those in function of the DVs, which is explained in Section 4.2.1. Now that the required data is at hand, the inner loop of the optimization can start. Initially the primal problem is approximated using the CONLIN approximation, which is discussed more into detail in Section 4.2.2. Once the approximation is defined it serves as starting point for Mehrotra's [26] predictor-corrector interior point method as implemented by Zillober [7]. Section 4.2.3 discusses the implementation of this method. Finally if the inner loop is finished, a new model has to be defined for re-analysis and the convergence criteria have to be checked as discussed in Section 4.2.4.

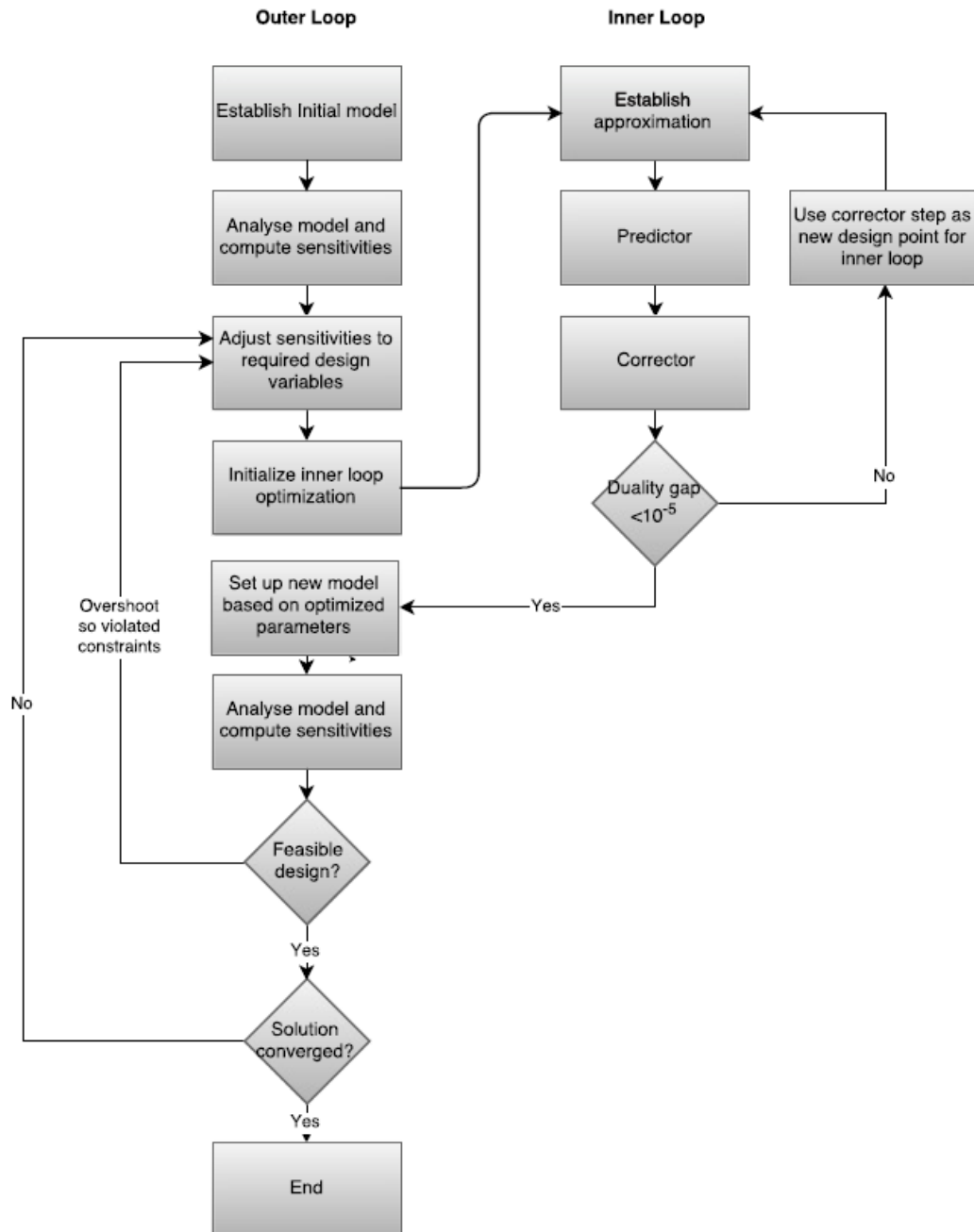
### 4.2.1 Sensitivity adjustment

Analysing the model in NASTRAN provides the basic information such as responses (stress and buckling) and the sensitivities. However, the latter can be determined in two different ways.

1. Extract from NASTRAN with respect to DVs.
2. Extract from NASTRAN with respect to geometrical properties.

Statement (1) can be accomplished by defining a linear approximation between the geometrical properties and the DVs to serve as coefficient input for the PREL card [47]. Statement (2) has the great advantage that the extracted sensitivities can be transformed to the required ones using analytical equations. This should provide a higher accuracy due to lack of round-off errors and fixes the amount of DVs requested from NASTRAN per property.

A transformation of sensitivities is shown in Equation (4.4). Here,  $y_j$  and  $x_i$  represent the available and requested DVs respectively and  $f$  represents the response vector.



**Figure 4.1:** Optimization process

$$\frac{\delta f}{\delta x_i} = \sum_j \frac{\delta f}{\delta y_j} \frac{\delta y_j}{\delta x_i} \quad (4.4)$$

Since the derivatives  $\delta f / \delta y_j$  are extracted from NASTRAN only the derivatives of the available DVs with respect to the required ones are necessary. In MATLAB this is entered in a symbolic manner such that the values can just be filled into the respective equations for each outer iteration minimizing computation times.

Table 4.3 shows an overview of the DVs available from analysis and those requested for the different stiffeners. Here  $A$  is the cross-sectional area,  $I_1, I_2$  and  $I_{12}$  are the moments of inertia,  $J$  is the rotational stiffness and  $Off_z$  and  $Off_x$  are the offsets of the element in  $z$  and  $x$  direction respectively.

**Table 4.3:** Available and required design variables for assorted stiffener types

| Cross-section | DV from analysis                       | DV required                  |
|---------------|----------------------------------------|------------------------------|
| I             | $A, I_1, I_2, I_{12}, J, Off_z$        | $H, W_2, W_1, t_3, t_2, t_1$ |
| Z             | $A, I_1, I_2, I_{12}, J, Off_z, Off_x$ | $H, W, t_3, t_2, t_1$        |

The equations presenting the relation between the DV from analysis and the required ones are those mentioned in Section 3.2.3 and 3.2. So eventually, the sensitivities of the responses to the requested DVs can be computed using Equation (4.5). Note that  $Off_x$  for I-stiffeners inherently zero due to symmetry.

$$\frac{\delta f}{\delta x_i} = \frac{\delta f}{\delta A} \frac{\delta A}{\delta x_i} + \frac{\delta f}{\delta I_1} \frac{\delta I_1}{\delta x_i} + \frac{\delta f}{\delta I_2} \frac{\delta I_2}{\delta x_i} + \frac{\delta f}{\delta I_{12}} \frac{\delta I_{12}}{\delta x_i} + \frac{\delta f}{\delta J} \frac{\delta J}{\delta x_i} + \frac{\delta f}{\delta Off_z} \frac{\delta Off_z}{\delta x_i} + \frac{\delta f}{\delta Off_x} \frac{\delta Off_x}{\delta x_i} \quad (4.5)$$

#### 4.2.2 Approximation of the primal problem

The approximation used for this optimization process is the same as for the CONLIN optimizer [6]. It is a combination of linear and reciprocal approximations coupled to the sign of the gradient. Equation (4.6) is used to create the approximate sub-problems. Here  $r$  is the approximated function consisting of all constraints and objective. It is important to approximate everything in the same manner such that all approximated sub-problems are convex. Note that  $x_i^0$  and  $x_i^k$  are the  $i^{th}$  DV from the initial and  $k^{th}$  iteration.

$$\mathbf{r} = \mathbf{c} + \sum a_{ij} (x_i^k - x_i^0) + \sum b_{ij} \left( \frac{1}{x_i^k} - \frac{1}{x_i^0} \right) \quad (4.6)$$



Where

$$\mathbf{r} = \begin{bmatrix} \tilde{\sigma} \\ \tilde{\lambda} \\ \mathbf{\tilde{D}c} \\ o\tilde{b}j \end{bmatrix}$$

$$a_{ij} = \begin{cases} \frac{\delta f_j}{\delta x_i} & \text{if } \frac{\delta f_j}{\delta x_i} > 0 \\ 0 & \text{if } \frac{\delta f_j}{\delta x_i} \leq 0 \end{cases}$$

$$b_{ij} = \begin{cases} 0 & \text{if } \frac{\delta f_j}{\delta x_i} > 0 \\ -(x_i)^2 \frac{\delta f_j}{\delta x_i} & \text{if } \frac{\delta f_j}{\delta x_i} \leq 0 \end{cases}$$

Note that the resulting equation is exactly the same as Equation (2.23) however now  $\mathbf{a}$  and  $\mathbf{b}$  have the same size, which makes the implementation into MATLAB much easier. Also it should be mentioned that the coefficients  $\mathbf{a}$ ,  $\mathbf{b}$  and  $\mathbf{c}$  are determined solely based on the information available from the outer loop ( $0^{th}$  step). Therefore the approximation is adjusted throughout the inner loop as a function of the initial values and the step size.

On top of this, the coefficients are normalised such that all quantities are in the same order of magnitude. The stress is normalised to  $\sigma_y$  and the buckling modes by adjusting  $\lambda$  to  $1/\lambda$ . The design constraints are normalised to their upper bound as  $\mathbf{Dc}/U$  and to their lower bound as  $L/\mathbf{Dc}$ , where  $U$  and  $L$  represent the values for upper and lower bound respectively. So in general, the constraints are normalised such that  $\mathbf{g} \leq 1$ . To this extent design constraints are defined in inverse relation to their lower bound to define them in the same manner. After normalisation they are defined as required where  $\mathbf{g} \leq 0$ .

#### 4.2.3 Predictor-Corrector interior point method

Before the implementation of the method it is described in a more elaborate matter. In other words, an extension to the paragraph of SCPIP [25] [7] in Section 2.2.4 is required. Here the equations as used within the script are defined with respect to the variables used. To initialise this procedure, the first thing required is the Lagrangian, which is specified by Equation (4.7). Where the variables are defined below.

- x Design variables
- s Slack variables of the constraints
- r Slack variables of design variables to the upper bound

- t Slack variables of the design variables to the lower bound
- $d_s$  Dual variable of the constraints
- $d_r$  Dual variable of design variables to the upper bound
- $d_t$  Dual variable of the design variables to the lower bound
- z Normalized objective =  $f_0$
- f(x) Normalised constraints
- $\mu$  Homotopy parameter

$$L(x, y, r, s, t, d_r, d_s, d_t) = f_0 + d_s^T (f(x) - z \cdot e + s) + d_r^T (x - x^{max} + r) + d_t (x_{min} - x + t) - \mu \sum \ln s_i - \mu \sum \ln r_i - \mu \sum \ln t_i \quad (4.7)$$

$e$  is a vector containing zeros for constraints and a one for the objective. In this way, both constraints and objective can be defined in one vector while  $e$  is used to filter out the necessary equations. To reach an optimal design, a Karush-Kuhn-Tucker point needs to be found. This is where the gradient of the Lagrangian becomes zero, leading to the equations presented in 4.8. The capital letters define diagonal matrices where the diagonal the entries correspond to the related vector.

$$\begin{aligned} \nabla_{f_0} : \quad & 1 - d_s^T e = 0 \\ \nabla_x : \quad & \nabla f(x) d_s + d_r - d_t = 0 \\ \nabla_r : \quad & D_r r - \mu e = 0 \\ \nabla_s : \quad & D_s s - \mu e = 0 \\ \nabla_t : \quad & D_t t - \mu e = 0 \\ \nabla_{d_r} : \quad & x - x^{max} + r = 0 \\ \nabla_{d_s} : \quad & f(x) - z + s = 0 \\ \nabla_{d_t} : \quad & x^{min} - x + r = 0 \end{aligned} \quad (4.8)$$

These equations are expanded using Newton's method, which is shown in Equation (4.9)[49]. The variable  $x$  is unrelated to the problem and  $f(x)$  represents any function of  $x$  and  $f'(x)$  is the derivative of  $f(x)$ . Applying this to the equations in 4.8 results in Equations 4.10 to 4.17. These equations are used to determine the step size of the variables. Where  $H(x)$  is the hessian matrix.

$$\Delta x = - \frac{f(x)}{f'(x)} \quad (4.9)$$

$$-\Delta d_s = e^T d_s - 1 \quad (4.10)$$

$$\nabla f(x) \Delta d_s + H(x) d_s \Delta x + \Delta d_r - \Delta d_t = -\nabla f(x) d_s - d_r + d_t \quad (4.11)$$

$$-\Delta x + \Delta t = x - x^{min} - t \quad (4.12)$$

$$\Delta x + \Delta r = x^{max} - x - r \quad (4.13)$$

$$\nabla f^T(x) \Delta x - \Delta z e + \Delta s = z e - f(x) - s \quad (4.14)$$

$$s \Delta d_s + d_s \Delta s = \mu - d_s s \quad (4.15)$$

$$r \Delta d_r + d_r \Delta r = \mu - d_r r \quad (4.16)$$

$$t \Delta d_t + d_t \Delta t = \mu - d_t t \quad (4.17)$$

To simplify the equations above, a variable is assigned to each equation's left hand side and listed below in Equation (4.18).

$$\begin{pmatrix} f_z \\ f_x \\ f_{d_t} \\ f_{d_r} \\ f_{d_s} \\ f_s \\ f_r \\ f_t \end{pmatrix} = \begin{pmatrix} e^T d_s - 1 \\ -\nabla f(x) d_s - d_r + d_t \\ x - x^{min} - t \\ x^{max} - x - r \\ z e - f(x) - s \\ \mu - d_s s \\ \mu - d_r r \\ \mu - d_t t \end{pmatrix} \quad (4.18)$$

Relating the variables to each other results in a reduction of the system. The final system is defined by Equation (4.19). Where  $K_y$  and  $sys_{f_{d_s}}$  are defined by Equations 4.20 and 4.21 respectively. This system can be solved to define the step size for the DVs.

$$\begin{bmatrix} 0 & -e^T \\ -e^T & K_y \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta d_s \end{bmatrix} = \begin{bmatrix} f_z \\ sys_{f_{d_s}} \end{bmatrix} \quad (4.19)$$

$$K_y = -(\nabla f^T(x) diag^{-1}(H(x) d_s + \frac{d_t}{t} + \frac{d_r}{r}) x + diag(\frac{s}{d_s})) \quad (4.20)$$

$$sys_{f_{d_s}} = f_{d_s} - diag^{-1}(d_s) f_s - \nabla f^T(x) diag^{-1}(H(x) d_s + \frac{d_t}{t} + \frac{d_r}{r}) (f_x + diag^{-1}(t)(f_t - d_t f_{d_t}) - diag^{-1}(r)(f_r - d_r f_{d_r})) \quad (4.21)$$

In order to solve the system, initial values for the dual variables need to be determined, which is done using Equation 4.22

$$\begin{aligned} d_s &= \frac{\mu_0}{s} \\ d_r &= \frac{\mu_0}{r} \\ d_t &= \frac{\mu_0}{t} \end{aligned} \quad (4.22)$$

Where  $\mu_0$  is defined by Equation (4.23). Here  $\dim_x$  and  $\dim_{d_s}$  are vectorial lengths for the  $x$  and  $d_s$  variables respectively.

$$\mu_0 = \frac{1}{\dim_{d_s} + 2\dim_x} \quad (4.23)$$

**Predictor step** In the predictor step, an initial guess for the step size of the DVs is made. After setting the homotopy parameter  $\mu$  to zero the system of Equation (4.19) is solved. This produces values for the increments of the design, slack and dual variables. This information is used to determine the value of  $\mu$  for the corrector step. To do so, first the step size for the primal and dual parameters should be computed, which is done in Equations 4.24 and 4.25 respectively.

$$\delta_{prim} = 0.95 \min\{\max\{\frac{-t_i}{\Delta t_i}\}, \max\{\frac{-s_i}{\Delta s_i}\}, \max\{\frac{-r_i}{\Delta r_i}\}, 0.95\} \quad (4.24)$$

$$\delta_{dual} = 0.95 \min\{\max\{\frac{-d_{t_i}}{\Delta d_{t_i}}\}, \max\{\frac{-d_{s_i}}{\Delta d_{s_i}}\}, \max\{\frac{-d_{r_i}}{\Delta d_{r_i}}\}, 0.95\} \quad (4.25)$$

Now the duality gap has to be computed for previous and updated solution, as provided by Equation (4.26) and 4.27.

$$d_{gap_{old}} = s^T d_s + r^T d_r + t^T d_t \quad (4.26)$$

$$d_{gap} = (s + \delta_{prim}\Delta s)^T (d_s + \delta_{dual}\Delta d_s) + (r + \delta_{prim}\Delta r)^T (d_r + \delta_{dual}\Delta d_r) + (t + \delta_{prim}\Delta t)^T (d_t + \delta_{dual}\Delta d_t) \quad (4.27)$$

Finally the homotopy parameter  $\mu$  can be determined by use of Equation (4.28). This parameter enables the use of constraint relaxation in the optimization process.

$$\mu_{run} = \min\{\max\{(d_{gap}/d_{gap_{old}})^2, 0.1\}, 1\} \mu_{run-1} \quad (4.28)$$

As you can see,  $\mu$  is restricted such that it will always be smaller or equal to the one in the previous iteration. Additionally, a minimum reduction of  $\mu$  of 90% is introduced. This is done to avoid  $\mu$  going to 0 in just a few iterations.

**Corrector step** In the corrector step Equation (4.19) is solved once more with the implementation of  $\mu$  as computed in the predictor step. Additionally some additional backward substitution is required to add the non-linear terms of the gradient to the corrector step. Equation (4.29) represents the terms that need to be added to the right hand side of the respective equations as shown in Equation (4.18). Note that the  $\Delta$  values in this equation are those computed by the predictor step.

$$\begin{aligned} f_x &: -H\Delta x\Delta d_s \\ f_s &: -\Delta s\Delta d_s \\ f_r &: -\Delta r\Delta d_r \\ f_t &: -\Delta t\Delta d_t \end{aligned} \quad (4.29)$$

Once this is done, again the primal and dual steps are determined using Equations 4.24 and 4.25 respectively. Now the solution is updated as shown below.

$$\begin{aligned}
 x^{k+1} &= x^k + \delta_{prim} \Delta x \\
 s^{k+1} &= s^k + \delta_{prim} \Delta s \\
 r^{k+1} &= r^k + \delta_{prim} \Delta r \\
 t^{k+1} &= t^k + \delta_{prim} \Delta t \\
 d_s^{k+1} &= d_s^k + \delta_{dual} \Delta d_s \\
 d_r^{k+1} &= d_r^k + \delta_{dual} \Delta d_r \\
 d_t^{k+1} &= d_t^k + \delta_{dual} \Delta d_t
 \end{aligned} \tag{4.30}$$

This entire process is repeated until a duality gap smaller than  $10^{-5}$  is reached.

#### 4.2.4 Convergence

Once the inner loop is converged and a new analysis is performed, a convergence check becomes imminent. To reach convergence two statements must hold.

- The new design must be feasible, meaning it must satisfy all constraints
- The new objective value must be lower than the minimum of the old feasible objective values

If these statements hold, the convergence requirements are checked using Equations 4.31 and 4.32.

$$CC_{constr} \geq 1 - \max(\mathbf{g}) \tag{4.31}$$

$$CC_{obj} \geq 1 - \frac{obj_{new}}{obj_{old}} \tag{4.32}$$

Note that  $\mathbf{g}$  represents the normalized constraint values. Therefore, due to the statement that the new design has to be feasible, this value should be slightly smaller than 1. This is similar for  $CC_{obj}$ . Now one can decide to either go for both or just one of those requirements to be fulfilled in order to achieve convergence. However it should be noted that the  $CC_{constr}$  criterion not necessarily results in the best feasible design. But due to the statement that the objective value should have decreased, the final design will be more optimal than the initial one.

Eventually it is chosen to only converge with respect to  $CC_{obj}$  for which the value is set to 0.005. Although this value is very small, the goal of this thesis is to show the optimization process, for which convergence is a critical aspect. Therefore the value is set low to have more iterations and thus more test data.

## 4.3 Test Cases

As mentioned in the introduction of this chapter, there are several points that should be achieved by this thesis. In order to prove that this is the case, certain test cases are defined. These cases are defined under the same loads and boundary conditions as specified in Chapter 3. The changing variables for these cases are shown below.

$N_s$  The amount of stiffeners on the panel. This variable is adjusted to show the convergence with respect to the amount of stiffeners such that also the ideal number can be determined.

$N_p$  The amount of possible properties for different stiffeners in one panel. These are generally selected such that the panel is still forced to be symmetrical. However this will prove that the optimization process can handle multiple sets of properties.

*Type* The type of cross-section. Although only one cross-section is handle per test case, it will show the optimizer's possibilities to handle different cross-sections.

The different test cases that were optimized are shown in Table 4.4. In the table, the amount of stiffeners  $N_s$ , the amount of different stiffener properties  $N_p$  and the type of stiffener are specified. On top of this, the column Prop/Stiff determines which property is assigned to which stiffener knowing that the stiffeners are also placed on the panel in that order starting from the left side while looking in the direction of the force.

For all the test cases, the values for the DVs have starting values defined by Table 4.5. Note however that the \* for test case three means that the starting values for the DVs were adjusted to reach an optimal solution, and will be discussed in more detail in Section 4.4.

The convergence of the test cases is solely defined on its decrease in objective function. So if the difference from one feasible point to the next is less than 0.5%, then the optimization has converged. This margin is relatively strict however the goal is to show the convergence therefore such strict criteria has to be implemented.

**Table 4.4:** Different test case settings for optimization

| Test Case | Nstiff | Nprop | Prop/Stiff        | type |
|-----------|--------|-------|-------------------|------|
| 1         | 5      | 1     | [1 1 1 1 1]       | I    |
| 2         | 5      | 3     | [1 2 3 2 1]       | I    |
| 3*        | 5      | 1     | [1 1 1 1 1]       | Z    |
| 4         | 5      | 3     | [1 2 3 2 1]       | Z    |
| 5         | 4      | 1     | [1 1 1 1]         | I    |
| 6         | 6      | 1     | [1 1 1 1 1 1]     | I    |
| 7         | 6      | 1     | [1 1 1 1 1 1]     | Z    |
| 8         | 7      | 1     | [1 1 1 1 1 1 1]   | I    |
| 9         | 7      | 1     | [1 1 1 1 1 1 1]   | Z    |
| 10        | 8      | 1     | [1 1 1 1 1 1 1 1] | I    |
| 11        | 8      | 1     | [1 1 1 1 1 1 1 1] | Z    |

**Table 4.5:** Initial values for the parameters

| Panel   |    |    |         |    |    |
|---------|----|----|---------|----|----|
| T       |    | 3  |         | mm |    |
| I-stiff |    |    | Z-stiff |    |    |
| H       | 55 | mm | H       | 55 | mm |
| W2      | 26 | mm | W       | 26 | mm |
| W1      | 26 | mm | t3      | 4  | mm |
| t3      | 4  | mm | t2      | 4  | mm |
| t2      | 4  | mm | t1      | 4  | mm |
| t1      | 4  | mm |         |    |    |

## 4.4 Results

In this section, the results of the test cases are displayed and discussed. They are related to the statements mentioned in the introduction of this chapter. The general convergent properties of the program are shown in Section 4.4.1 based on the initial model defined in Chapter 3. Next Z-stiffeners are implemented into the initial model instead of I-stiffeners and optimized the results of which are shown in Section 4.4.2. Eventually, the programs capability of handling a different amount of property sets within one optimization is shown in Section 4.4.3. Finally the optimal amount of stiffeners is deduced in Section 4.4.4. Note that the area is defined to describe the objective rather than the volume since the length of the panel is constant. Furthermore, all test cases showed to be buckling critical therefore only the first eigenvalue is shown as constraint. However it is important to state that although only the first eigenvalue is shown, the first 20 are taken into account.

The results presented here are partially restricted to the optimal solution. For the complete set of results and the MATLAB script used to generate them please refer to this link : <http://we.tl/RJYMOpNHZ9>.

Additionally, for the abbreviations used for the design constraints please refer to Table 4.1. Finally note that the \* in test case 3 specifies the use of a different starting point.

### 4.4.1 Basic Optimization

To show the basic optimization capabilities of the program, it is applied to the model discussed in Chapter 3, which is evidently test case 1. Figure 4.2 shows the convergence of the objective along the optimization and the respective values are presented by Table 4.6.

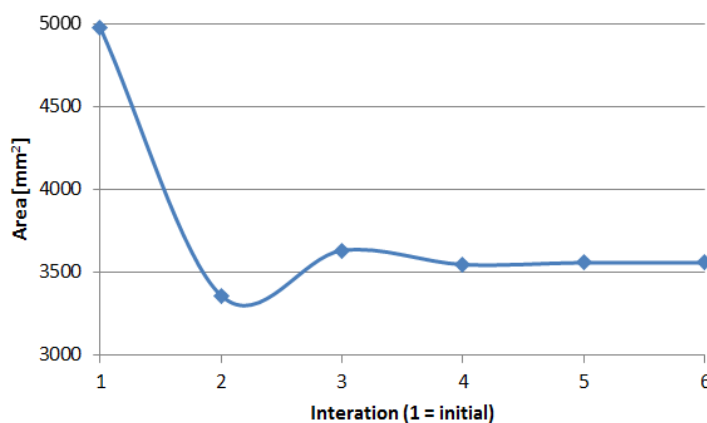


Figure 4.2: Convergence of objective

Table 4.6: Objective values per iteration

| Obj       | Area [mm <sup>2</sup> ] |
|-----------|-------------------------|
| Initial   | 4980,000                |
| run 1     | 3362,343                |
| run 2     | 3634,173                |
| run 3     | 3549,019                |
| run 4     | 3560,915                |
| run 5     | 3559,354                |
| Reduction | 28,53%                  |



These results clearly indicate a steady convergence towards an optimal solution, which has 28.53% less area than the initial model. The values for the constraints across the optimization are shown in Table 4.7. Here it can be seen that although some infeasible designs were achieved in the process, the method does converge to a feasible one. The reason for this is most likely the use of the approximations.

Approximations are always most accurate in the neighbourhood of the initial point. A large step size as seen in run 1 for example leads to a less accurate approximation. By slowly getting closer to the optimal result, the step size computed in the inner loop decreases leading to a more accurate approximation. Eventually this leads to new feasible designs.

**Table 4.7:** Constraint values per iteration

|         | Mode 1 | AFR[-] | WR[-]  | FFR[-] | AFSR[-] |
|---------|--------|--------|--------|--------|---------|
| Initial | 1,577  | 7,700  | 11,750 | 6,500  | 1,333   |
| run 1   | 0,595  | 7,922  | 13,836 | 6,164  | 1,310   |
| run 2   | 1,042  | 7,970  | 19,554 | 6,155  | 1,300   |
| run 3   | 0,981  | 7,992  | 19,979 | 5,904  | 1,300   |
| run 4   | 1,000  | 8,023  | 19,989 | 5,660  | 1,300   |
| run 5   | 1,000  | 8,031  | 19,999 | 5,409  | 1,300   |

For completeness of this section, the values of the parameters for each consecutive step are provided in Table 4.8. Looking at the values shows that the parameters behave as expected. The optimizer has the tendency of decreasing the thickness and levelling out the other parameters with respect to it. The height and the width of the free flange remain the highest resulting in higher buckling resistance.

**Table 4.8:** Parametric History

|         | T[mm] | H[mm]  | W2[mm] | W1[mm] | t3[mm] | t2[mm] | t1[mm] |
|---------|-------|--------|--------|--------|--------|--------|--------|
| Initial | 3,000 | 55,000 | 26,000 | 26,000 | 4,000  | 4,000  | 4,000  |
| run 1   | 2,776 | 33,592 | 14,083 | 8,189  | 2,000  | 2,285  | 3,635  |
| run 2   | 2,791 | 47,932 | 17,889 | 8,000  | 2,117  | 2,906  | 3,628  |
| run 3   | 2,775 | 47,000 | 15,933 | 8,000  | 2,037  | 2,699  | 3,608  |
| run 4   | 2,769 | 47,995 | 15,557 | 8,000  | 2,083  | 2,749  | 3,600  |
| run 5   | 2,768 | 48,284 | 15,024 | 8,000  | 2,095  | 2,777  | 3,598  |

#### 4.4.2 Different cross-sections

To show the programs capability of handling different cross sections, Z-stiffeners were implemented instead of I-stiffeners into the initial model. This should not be any problems

since the properties entered in NASTRAN are unrelated to the cross section type. However the amount of requested sensitivities and the equations for the conversion are different. This should prove the ease of implementation of this concept.

Table 4.9 shows the results of the optimization with Z-stiffeners (test case 3) with that of the I-stiffeners (test case 1). The final optimal weight is less for Z-stiffeners than for I-stiffeners. It must be stated that a different starting point had to be chosen in order to reach a feasible design after optimization. The starting values for the DVs are shown in Table 4.10. If the original starting point is used, the optimization would get stuck in an endless oscillatory sequence. This happens due to an overshoot in the first iteration resulting an extreme infeasible design, which can no longer return to a feasible one. A similar phenomenon is seen when implementing test case 4, which will be discussed in Section 4.4.3.

**Table 4.9:** Comparison for optimization objective with different cross sections

| Test Case | Cross-Section | Nr.it | Initial | Optimal | reduction |
|-----------|---------------|-------|---------|---------|-----------|
| 1         | I             | 5     | 4980    | 3559    | 28,53%    |
| 3*        | Z             | 12    | 4515    | 3556    | 21,24%    |

To reach an optimal design the constraints must hold. Prove of this is shown in Table 4.11. Therefore it can be concluded that the optimizer is capable of addressing different cross-sections while the FEM input and output request remain the same. This proves that random cross-sections can be utilised within this optimization provided that the analytical equations are available to transform the received sensitivities into the required ones.

**Table 4.10:** Start values for design variables

| DV | Value [mm] |
|----|------------|
| T  | 3          |
| H  | 55         |
| W  | 8          |
| t3 | 3          |
| t2 | 3          |
| t1 | 3          |

**Table 4.11:** Constraint values for test case 3

| Constr | Value [-] |
|--------|-----------|
| Buck   | 1,000     |
| AFR    | 8,061     |
| WR     | 19,937    |
| FFR    | 3,000     |
| AFSR   | 1,300     |

### 4.4.3 Multiple properties

There are two test cases where multiple properties are addressed, case 2 and 4. Where they define multiple I and Z-beam properties respectively. The bar properties are assigned symmetrically on the panel in the order as defined in Table 4.4.

The optimal objective value for case 2 with respect to the initial case is found in Table 4.12. The reason that test case 4 is not included in this table is because it did not converge. More explanation on this follows. From Table 4.12 it can be seen that the total reduction is higher when different properties are assigned to the stiffeners. This is only logical as it enables the optimizer to adjust the stiffeners more freely and thus a more optimal design can be found. Off course this also has its influence on the computational time.

The optimization for the different properties takes 138 outer iterations while for one property only 5. For a increase in reduction of only 0.39%, it does not seem reasonable to optimize for different properties. However when the load cases become more complex this reduction should increase drastically, which would make it very attractive.

**Table 4.12:** Comparison for optimization objective with different amount of properties

| Test Case | #it | Initial | Optimal | reduction |
|-----------|-----|---------|---------|-----------|
| 1         | 5   | 4980    | 3559    | 28,53%    |
| 2         | 138 | 4980    | 3540    | 28,92%    |

Table 4.13 shows an overview of the different properties after optimization of test case 2 and 1. Clearly the middle stiffener with property 3 is less significant than the others.

**Table 4.13:** Parameters per property after optimization

| Test Case | Property | T     | H      | W2     | W1    | t3    | t2    | t1    |
|-----------|----------|-------|--------|--------|-------|-------|-------|-------|
| 1         | 1        | 2,768 | 48,284 | 15,024 | 8,000 | 2,095 | 2,777 | 3,598 |
| 2         | 1        | 2,791 | 47,583 | 10,666 | 8,000 | 2,022 | 3,544 | 3,629 |
| ⋮         | 2        | ⋮     | 48,835 | 11,104 | 8,000 | 2,080 | 3,686 | 3,629 |
| ⋮         | 3        | ⋮     | 46,692 | 9,640  | 8,000 | 2,000 | 3,070 | 3,629 |

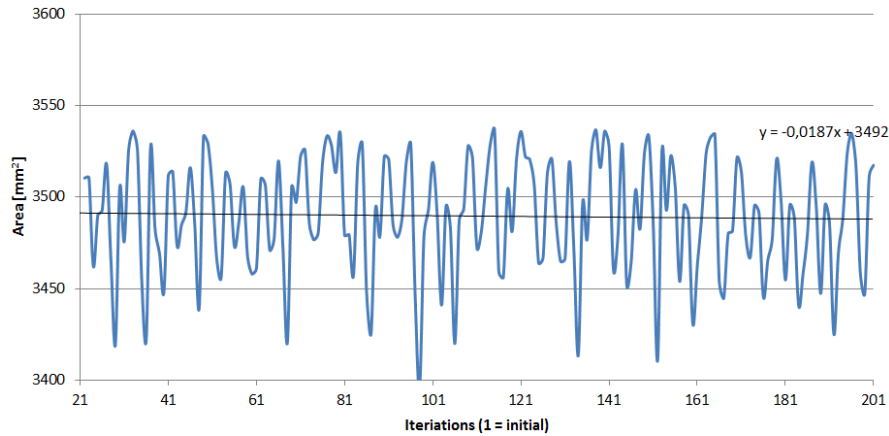
Table 4.14 shows once more that all constraints are satisfied and thus the design is feasible.

**Table 4.14:** Constraint values for test case 1 and 2 after optimization

| Test Case | Property | Mode 1 | AFR     | WR      | FFR     | AFSR    |
|-----------|----------|--------|---------|---------|---------|---------|
| 1         | 1        | 1,000  | 8,03    | 20,00   | 5,41    | 1,30    |
| 2         | 1        | 1,001  | 7,9424  | 19,9895 | 3,00985 | 1,30004 |
| ⋮         | 2        | ⋮      | 7,95847 | 19,9623 | 3,01224 | 1,30004 |
| ⋮         | 3        | ⋮      | 7,93645 | 19,9967 | 3,13981 | 1,30004 |

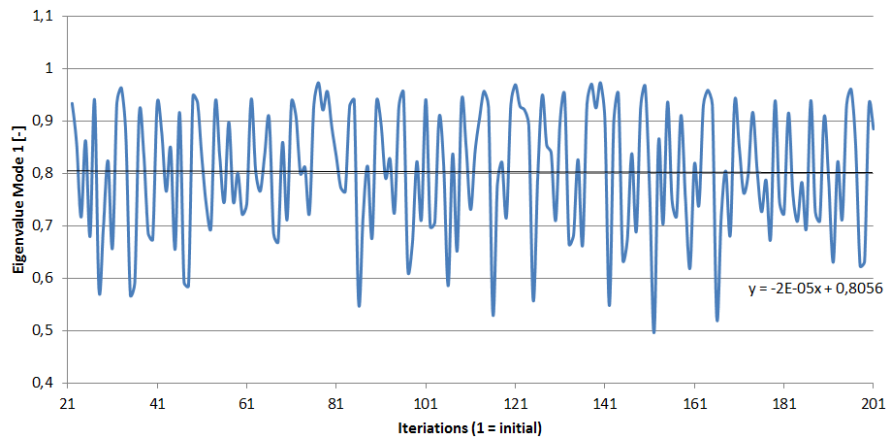
For test case 4 the solution did not converge. Figure 4.3 shows the change in objective throughout the optimization process. It was halted at 200 iterations. A trendline was added to the

graph and its equation is shown within the figure. The graph only starts at run 21 such that the oscillatory part is better represented. This indicates that although the process is oscillating continuously, the objective is slowly minimizing.



**Figure 4.3:** Evolution of the objective during optimization of case 4

The same visualisation is prepared for the critical buckling mode in Figure 4.4. Here can be seen that a similar trend as for the objective exists for the critical buckling mode. Since the eigenvalue is already beneath its constraint value after the first iteration, this value should not keep decreasing. However the rate at which it does is nearly 0 so it can be assumed to be a stable oscillation.



**Figure 4.4:** Evolution of the buckling constraint during optimization of case 4

The reason for the oscillatory behaviour is simply its design freedom. During the inner loop only the 20 modes determined by the analysis can be taken into account. Therefore, it cannot account for possible new modes due to adjustment of the parameters.

The critical buckling modes for runs 20 to 29 are presented in Figure 4.5. Note that most

modes that are not symmetrical have a symmetric counterpart with a slightly higher eigenvalue due to the slight shift in CG along the x-axis of the Z-stiffeners with respect to the elements. Taking for example run 21 shows that the stiffeners with property 1 and 2 are underdesigned. During the optimization these are thus adjusted to be more stiff while property 3 is adjusted to be counter the increase for 1 and 2 since the objective function should not increase. This results in an underdesigned stiffener in the middle with property 3. Therefore, the new critical buckling mode originates from that stiffener in the successive iteration. This continues such that eventually a pattern forms as seen in Figure 4.5.

So for this section it can be concluded that for this specific load case, optimizing with respect to a set of properties rather than one is not very beneficial. On top of this it may cause serious oscillatory behaviour and lead to non-convergence.

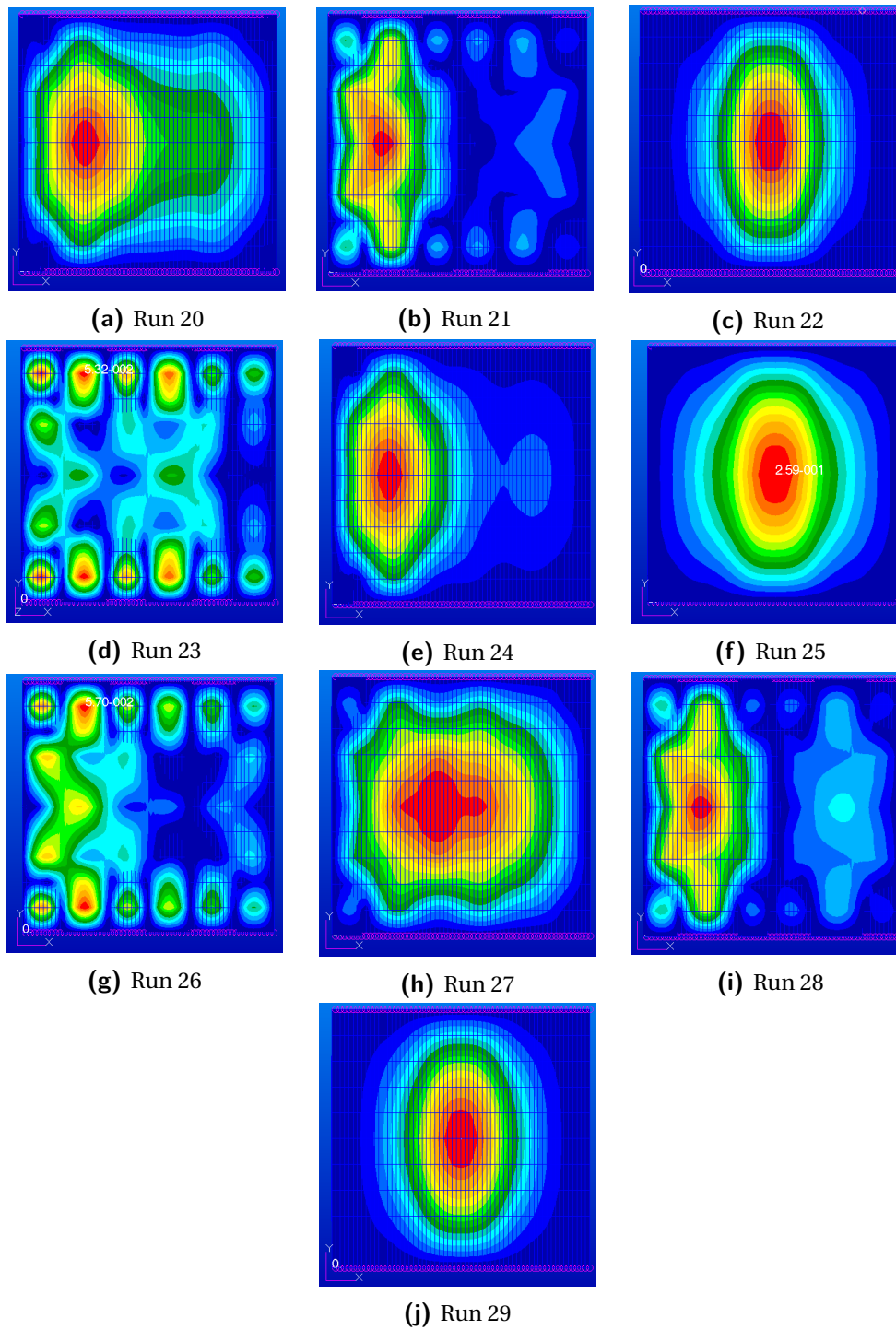
#### 4.4.4 Different amount of stiffeners

Finally the program is run for different cases with a variable number of stiffeners. This is done for both I-stiffeners and Z-stiffeners. Figure 4.6 and 4.7 show the convergence for different amounts of I and Z-stiffeners respectively. These indicate that all cases nicely converge. Note however that for Figure 4.7, case 9 is not included because it takes 182 iterations to converge as can be seen in Table 4.16. The slow convergence is devoted to similar oscillatory behaviour as test case 3.

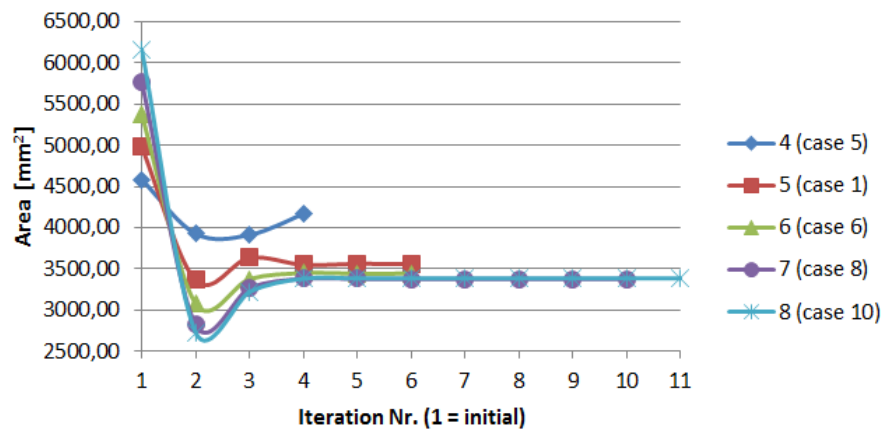
The data for the optimized cases can be found in Table 4.15 and 4.16. Again for all panels, buckling formed the critical constraint and therefore the stress constraints are omitted. From the data its clear that test case 8 and 9 propose the most optimal result for I and Z-stiffeners respectively. These test cases both refer to 7 stiffeners as the optimal amount. So it can be concluded 7 stiffeners are required to reach the optimal result for this specific load case.

**Table 4.15:** Optimization data for cases with different amount of I-stiffeners

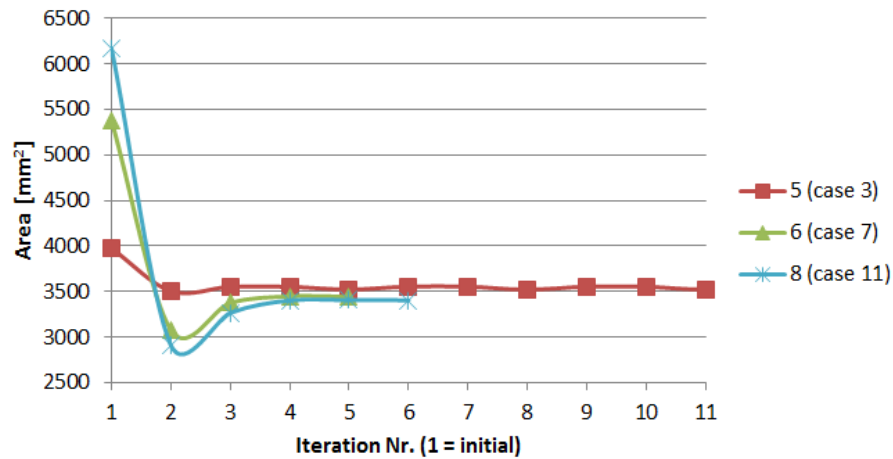
| Test Case | #it | Objective |         |           | Constraints |      |       |      |      |
|-----------|-----|-----------|---------|-----------|-------------|------|-------|------|------|
|           |     | Initial   | Optimal | reduction | Buck        | AFR  | WR    | FFR  | AFSR |
| 1         | 5   | 4980      | 3559    | 28,53%    | 1,000       | 8,03 | 20,00 | 5,41 | 1,30 |
| 5         | 3   | 4584      | 4170    | 9,04%     | 1,021       | 7,45 | 13,66 | 6,50 | 1,30 |
| 6         | 5   | 5376      | 3443    | 35,96%    | 1,001       | 8,97 | 20,00 | 5,39 | 1,30 |
| 8         | 9   | 5772      | 3375    | 41,53%    | 1,001       | 9,66 | 20,00 | 4,37 | 1,30 |
| 10        | 10  | 6168      | 3384    | 45,13%    | 1,001       | 9,93 | 20,00 | 4,39 | 1,30 |



**Figure 4.5:** Critical buckling modes for case 4 for different steps in the optimization process



**Figure 4.6:** Evolution of the objective during optimization for different amounts of I-stiffeners



**Figure 4.7:** Evolution of the objective during optimization for different amounts of Z-stiffeners

**Table 4.16:** Optimization data for cases with different amount of Z-stiffeners

| Test Case | #it | Objective |         |           | Constraints |       |       |      |      |
|-----------|-----|-----------|---------|-----------|-------------|-------|-------|------|------|
|           |     | Initial   | Optimal | reduction | Buck        | AFR   | WR    | FFR  | AFSR |
| 3         | 12  | 4515      | 3556    | 21,24%    | 1,000       | 8,06  | 19,94 | 3,00 | 1,30 |
| 7         | 4   | 5376      | 3442    | 35,98%    | 1,001       | 8,92  | 20,00 | 3,01 | 1,30 |
| 9         | 182 | 5772      | 3386    | 41,33%    | 1,000       | 9,74  | 20,00 | 3,00 | 1,30 |
| 11        | 5   | 6168      | 3405    | 44,80%    | 1,001       | 10,04 | 20,00 | 3,00 | 1,30 |

---

## Chapter 5

---

# Conclusion

The goal of this thesis was to develop an optimizer for shape and size optimization of stiffened panels by combining FE software with an interior point method coded in MATLAB.

This is achieved using Mehrotra's predictor-corrector interior point method as implemented by Zillober [7]. In combination with Fleury's ConLin approximation scheme it results in an easy implementable optimizer.

The input for the optimizer is defined by NASTRAN based on a 2D model of a stiffened panel. A 2D representation enables the adjustment of bar properties and shell thickness without having to change the mesh, which is very attractive for optimization purposes. The panel and stiffeners are defined by shell and bar elements respectively. The bar elements need to be defined using the PBAR property card. These directly implement the geometrical properties instead of stiffener cross sections and dimensions into the bar element. Therefore a free cross section design is made possible. Note that this has the drawback of having to calculate the properties manually and the production of additional round-off errors.

The sensitivity's are extracted from NASTRAN. To achieve the highest accuracy, they are requested with respect to the geometrical properties rather than the design variables. This allows for the transformation of the extracted sensitivities to the required ones by use of analytical equations. Provided these equations are available, this method allows for a fixed amount of extracted information, which can be adjusted to any number of design variables.

The optimizer is capable of handling problems with different properties for the stiffeners. It is also capable of handling different cross sectional shapes and can be used to determine the



optimal amount stiffeners. These statements are supported by the results. Steady convergence for most cases is achieved however some showed oscillatory behaviour. From investigation it can be concluded that this behaviour originates from the optimizer's incapability to anticipate buckling modes, outside of those set as constraints, within inner loop.

In the end the goal of developing the optimizer was achieved while additional restrictions due to the FEM package were circumvented.

---

## Chapter 6

---

# Recommendations

For further research on this approach, some small recommendations are in order. These should help to establish a more robust program capable of solving different load cases and achieve more stable convergence.

Momentarily the program uses MATLAB to define the data structure for the bdf. However a method could be developed where MATLAB can analyse the bdf and extract its data structure from it. This would provide the user with the opportunity to create his model in a FEM GUI environment rather than having to learn the setup method of the data structure. All the same properties can still be implemented and the equations for sensitivity adjustment and geometrical properties could be provided through an additional input. This method would also allow for easier implementation of new load cases. On the other hand it would restrict the automatic FEM model generation for a variable number of stiffeners.

Another issue noticed in the program until now is the oscillatory behaviour. In the optimization scheme, constraint relaxation is implemented in the inner loop such that the approximated solution can steadily converge towards the constraints. If a similar approach would be implemented for the outer loops, the inner loop would be less likely to adjust the design variables in such manner that a new and maybe even more critical buckling mode is reached. Eventually this would provide possibly slower but more stable convergence.

Finally as mentioned, one of the most basic approximations methods is implemented within the optimizer. This could be adjust such that more elaborate approximation schemes can be used. Doing this will allow the program to more accurately approximate the functions in the inner loop. More accurate approximations should also lead to a more robust program.



---

# Bibliography

- [1] L. A. Schmidt. Structural Design by Systematic Synthesis. In *Proceedings of the Second ASCE Conference on Electronic Computation*, pages 105–132, New York, 1960. ASCE.
- [2] Y. Radovicic and a. Remouchamps. BOSS QUATTRO: An open system for parametric design. *Structural and Multidisciplinary Optimization*, 23(2):140–152, 2002.
- [3] Malvin S. Anderson, W. Jefferson Stroud, Barbara J. Durling, and Katherine W. Hennessy. *PASCO: Structural panel analysis and sizing code, user's manual*. 1980.
- [4] F. W. Williams, D. Kennedy, and P. W. L. Williams. Current Experience of the Use of the Stiffened Panel Design Software Viconopt in Optimum Design of Aircraft. *Engineering Optimization*, 29(1-4):39–48, 1997.
- [5] Irvin J. Lustig, Roy E. Marsten, and David F. Shanno. On Implementing Mehrotra's Predictor-Corrector Interior Point Method for Linear Programming. *SIAM Journal on Optimization Journal on Optimization*, 2(3):435–449, 1992.
- [6] C. Fleury. CONLIN: An efficient dual optimizer based on convex approximation concepts. *Structural Optimization*, 1:81–89, 1989.
- [7] Christian Zillober. A combined convex approximation—interior point approach for large scale nonlinear programming. *Optimization and Engineering*, 2:51–73, 2001.
- [8] René Alderliesten. Reader Materials and Structures, 2010.
- [9] Sahil Kapoor. No Title, 2013.
- [10] D.G. Stamatelos, G. N. Labeas, and K. I. Tserpes. Analytical calculation of local buckling and post-buckling behavior of isotropic and orthotropic stiffened panels. *Thin-Walled Structures*, 49(3):422–430, 2011.

- [11] David W. A. Rees. *Mechanics of Optimal Structural Design*, chapter Appendix B, pages 525–535. John Wiley & Sons, Ltd, Chichester, UK, oct 2009.
- [12] Sjoerd van der Veen and Daniel Coatta. STIFFENED PANELS IN COMPRESSION: REDIRECTING LOADS TOWARD HIGH-STRENGTH STIFFENERS. In *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*, number April, pages 1–11, 2004.
- [13] K Svanberg. The method of moving asymptotes: a new method for structural optimization, 1987.
- [14] Panos Y. Papalambros and Douglass J. Wilde. *Principles of Optimal Design: Modeling and Computation*. 2000.
- [15] Matthijs Langelaar and Fred Van Keulen. Engineering Optimization - Lecture Slides.
- [16] Derren Wesley Wood. *Dual Sequential Approximation Methods in Structural Optimisation*. PhD thesis, 2012.
- [17] L. A. Schmidt. Some Approximation Concepts for Structural Synthesis. Technical report, 1976.
- [18] V. Braibant and C. Fleury. An approximation-concepts approach to shape optimal design. *Computer Methods in Applied Mechanics and Engineering*, 53(2):119–148, 1985.
- [19] Andy Keane and Prasanth Nair. *Computational Approaches for Aerospace Design: The Pursuit of Excellence*. 2005.
- [20] C. Zillober. A globally convergent version of the method of moving asymptotes. *Structural Optimization*, 6:166–174, 1993.
- [21] K. Shittkowski. The nonlinear programming method of Wilson, Itan and Powell with an augmented Lagrangian type line search function. *Numerische Matimatik*, 38:83–114, 1981.
- [22] K Svanberg. A class of globally convergent optimization methods based on conservative convex seperable approximations. *SIAM Journal on Optimization*, 12(2):555–573, 2002.
- [23] Krister Svanberg. MMA and GCMMA – two methods for nonlinear optimization. 1:1–15, 2007.
- [24] Christian Zillober. Global convergence of a nonlinear programming method using convex approximations. *Numerical Algorithms*, 27:265–289, 2001.
- [25] C. Zillober. SCIP - An efficient software tool for the solution of structural optimization problems. *Structural and Multidisciplinary Optimization*, 24:362–371, 2002.
- [26] Sanjay Mehrotra. On the Implementation of a Primal-Dual Interior Point Method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.

- [27] Irvin J. Lustig, Roy E. Marsten, and David F. Shanno. Computational experience with a primal-dual interior point method for linear programming. *Linear Algebra and its Applications*, 152:191–222, 1991.
- [28] Gm Fadel, Mf Riley, and Jm Barthelemy. Two point exponential approximation method for structural optimization. *Structural Optimization*, 124:117–124, 1990.
- [29] Raphael T. Haftka, Joel a. Nachlas, Layne T. Watson, Thomas Rizzo, and Rajendra Desai. Two-point constraint approximation in structural optimization. *Computer Methods in Applied Mechanics and Engineering*, 60(3):289–301, 1987.
- [30] B. Prasad. Explicit constraint approximation forms in structural optimization. *Computer Methods in Applied Mechanics and Engineering*, 40:1–26, 1983.
- [31] Liping Wang and R.V. Grandhi. Efficient safety index calculation for structural reliability analysis. *Computers & Structures*, 52(1):103–111, 1994.
- [32] Liping Wang and Ramana V Grandhi. Improved two-point function approximations for design optimization. *AIAA Journal*, 33(9):1720–1727, sep 1995.
- [33] Suqiang Xu and Ramana V. Grandhi. Effective two-point function approximation for design optimization. *AIAA Journal*, 36(12):2269–2275, 1998.
- [34] M S. Kim, J R. Kim, Y. Jeon J., and D H. Choi. Efficient mechanical system optimization using two-point diagonal quadratic approximation in the nonlinear intervening variable Space. *KSME Inter. J.*, 15(9):1257–1267, 2001.
- [35] Jong Rip Kim and Dong Hoon Choi. Enhanced two-point diagonal quadratic approximation methods for design optimization. *Computer Methods in Applied Mechanics and Engineering*, 197(6-8):846–856, 2008.
- [36] Seonho Park, Seung Hyun Jeong, Gil Ho Yoon, Albert a. Groenwold, and Dong Hoon Choi. A globally convergent sequential convex programming using an enhanced two-point diagonal quadratic approximation for structural optimization. *Structural and Multidisciplinary Optimization*, 0:739–753, 2014.
- [37] Je Falk. Lagrange multipliers and nonlinear programming. *Journal of Mathematical Analysis and Applications*, 19:141–159, 1967.
- [38] L. F P Etman, Albert a. Groenwold, and J. E. Rooda. First-order sequential convex programming using approximate diagonal QP subproblems. *Structural and Multidisciplinary Optimization*, 45:479–488, 2012.
- [39] Albert a. Groenwold, L. F P Etman, and Derren W. Wood. Approximated approximations for SAO. *Structural and Multidisciplinary Optimization*, 41(1):39–56, 2010.
- [40] Klaus Schittkowski. Sequential Quadratic Programming Methods with Distributed and Non-Monotone Line Search. *1st Weimar Optimization and Stochastic Days*, pages 1–21, 2004.

- [41] Klaus Schittkowski. The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function - Part 2: An efficient implementation with linear least squares subproblems. *Numerische Mathematik*, 38(1):115–127, 1982.
- [42] Klaus Schittkowski. On the convergence of a sequential quadratic programming method with an augmented lagrangian line search function. *Mathematische Operationsforschung und Statistik. Series Optimization*, 14(2):197–216, 1983.
- [43] I. Bongartz, a. R. Conn, Nick Gould, and Ph. L. Toint. CUTE: constrained and unconstrained testing environment. *ACM Transactions on Mathematical Software*, 21(1):123–160, 1995.
- [44] Klaus Schittkowski. NLPQL: A Fortran subroutine for solving constrained nonlinear programming problems. *Mathematical Programming*, 5(2):485–500, 1986.
- [45] G.N. Vanderplaats. Very large scale continuous and discrete variable optimization, 2004.
- [46] D.M. Gay. Electronic mail distribution of linear programming test problems, 1985.
- [47] MSCSoftware. *MD / MSC Nastran 2010 Quick Reference Guide*. 2010.
- [48] Benoît Colson, Michaël Bruyneel, Stéphane Grihon, Caroline Raick, and Alain Remouchamps. Optimization methods for advanced design of aircraft panels: A comparison. *Optimization and Engineering*, 11(4):583–596, 2010.
- [49] a. Galántai. The theory of Newton's method. *Journal of Computational and Applied Mathematics*, 124(1-2):25–44, 2000.

---

# Appendix A

---

## BDF setup

Figure A.1 shows the basic set-up for the bdf files used within this thesis. Note that after this initialisation part of the bdf, the bulk data entries as specified by Section 3.2.2 are included. In this initialisation part, some similar commands are seen. These however do not represent the NASTRAN input cards but merely reference which sets of SPC, MPC, LOADS, etc. have to be used within the analysis of the subcase. Furthermore, STRAIN, STRESS, FORCE, etc. refer to the output request. More information can be found in the Quick Reference Guide of NASTRAN [47].



```

SOL 200
INCLUDE 'alter_respl.alt'
INCLUDE 'alter_correct_sens.alt'
CEND
$<----->
ECHO = NONE
DSAPRT (NOPRINT,EXPORT,START=1,END=SENS)=ALL
DISP   = ALL
SUBCASE 1
ANALYSIS = STATICS
SUBTITLE = static load
SPC      = 1
MPC      = 1
LOAD     = 1
DESSUB   = 100
STRAIN (SORT1,REAL,VONMISES,STRCUR,CENTER)=ALL
STRESS (SORT1,REAL,VONMISES,CENTER)=ALL
FORCE (SORT1,REAL)=ALL
SPCFORCES=ALL
GPFORCE=ALL
SUBCASE 2
ANALYSIS = BUCK
SUBTITLE = buckling analysis
STATSUB BUCKLING = 1
SPC      = 1
MPC      = 1
DESSUB   = 200
METHOD   = 1
$<----->
BEGIN BULK
PARAM          POST    -1
PARAM  PRTMAXIM  YES
PARAM   AUTOSPC   YES
PARAM    GRDPNT    0
PARAM     OMID   YES
PARAM    DSNOKD  0.0

```

**Figure A.1:** BDF analysis set-up