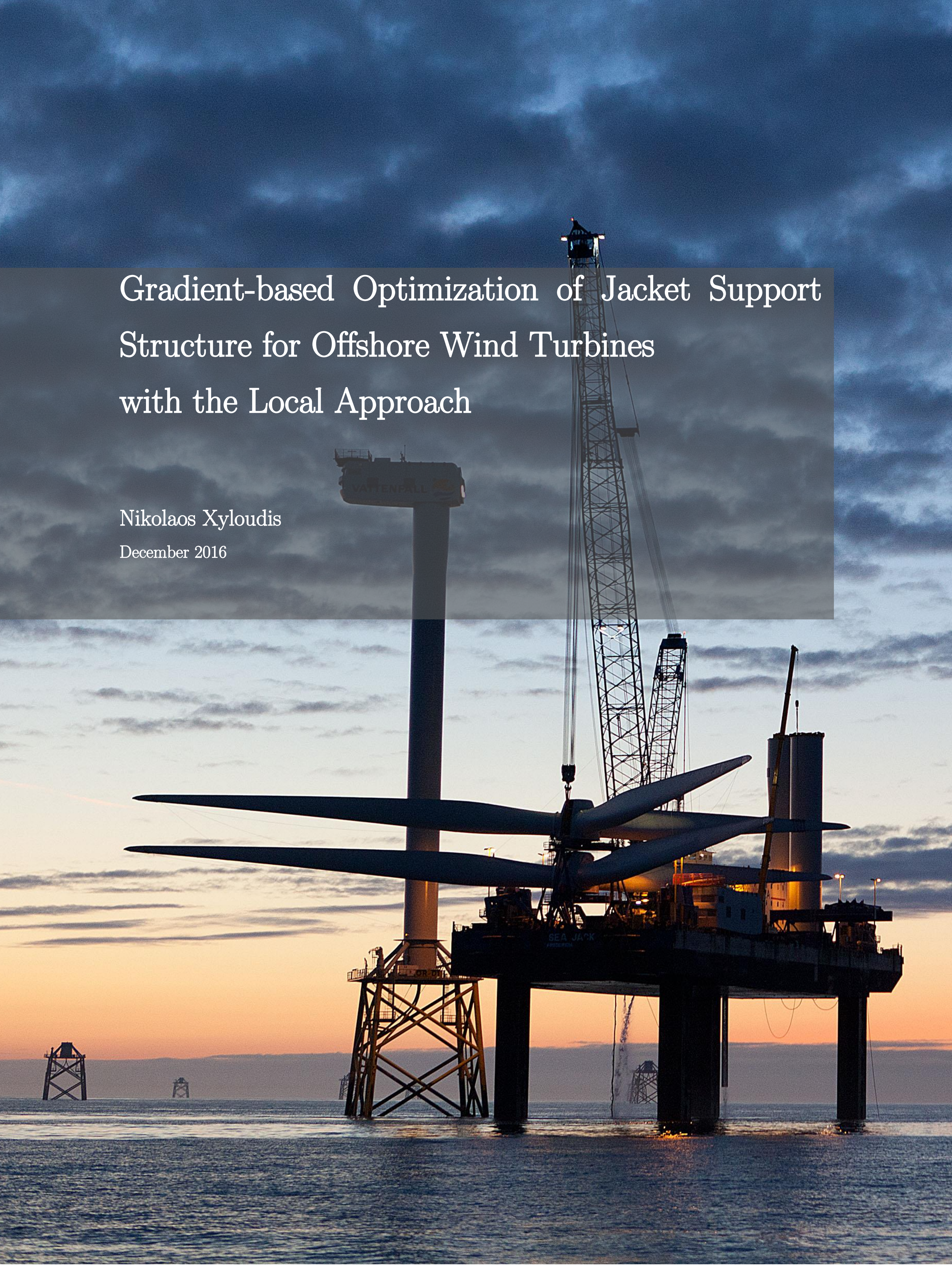
NTNU
Norwegian University of
Science and Technology

Gradient-based Optimization of Jacket Support Structure for Offshore Wind Turbines with the Local Approach

Nikolaos Xyloudis

Wind Energy
Submission date: January 2017
Supervisor: Torgeir Moan, IMT
Co-supervisor: Yilmaz Salman, Keppel Verolme BV
Michael Muskulus, NTNU
Andrei Metrikine, TU Delft

Norwegian University of Science and Technology
Department of Marine Technology



Gradient-based Optimization of Jacket Support Structure for Offshore Wind Turbines with the Local Approach

Nikolaos Xyloudis

December 2016

EWEM European Wind Energy Master
Erasmus Mundus MSc programme

TU Delft

NTNU

Keppel Verolme

Gradient-based Optimization of Jacket Support Structure for Offshore Wind Turbines

with the

Local Approach

By

Nikolaos Xyloudis

to be defended publicly on Wednesday December 21, 2016 at 10.30 AM.
at the Delft University of Technology.

Committee:	Prof. dr. Torgeir Moan	NTNU	(Supervisor)
	Prof. dr. Michael Muskulus	NTNU	(Co-supervisor)
	Prof. dr. Andrei Metrikine	TU Delft	(Co-supervisor)
	Ir. Yilmaz Salman	Keppel Verolme	(Co-supervisor)

.

EWEM European Wind Energy Master
Erasmus Mundus MSc programme

TU Delft NTNU Keppel Verolme

TU Delft Delft University of Technology

NTNU Norwegian University of Science and Technology

Keppel Verolme

Cover photo: Ormonde offshore wind farm [1]

Summary

A support structure system contributes significantly to the total capital cost in an offshore wind project. It is an area which attracts numerous studies in academia and industry, due to the potential for cost reduction. Structural optimization is not a usual practice in offshore wind turbine support structure design. The reason for this is that the design process involves a large number of iterations since the optimization problem is highly constrained and non-convex. There is a need for an optimization methodology which is effective and efficient in designing the offshore wind turbine support structures. This becomes even more important when complex structures are considered that have many design variables, e.g. space frame structures.

In order to reduce the computational cost here it is assumed that when changing dimensions of a structural member, sectional response forces and moments remain constant. This assumption is made for both the actual changed member and for all other members of the structure. Additionally, this is assumed to be valid for a simultaneous change of all members all over the structure. It is the so-called local optimization approach. The local optimization approach has been successfully applied for the sizing of a full-height lattice tower and a jacket support structure, but under the use of a very simplified algorithm for the local optimization of structural members [2].

The objective of this project is the development of a gradient based optimization algorithm for the global optimization of the jacket support structure of the offshore wind turbine and the implementation of the algorithm using the principle of local approach.

The UpWind reference jacket used within the OC4-project with the generic NREL 5MW turbine atop was used for this study. The development of the gradient based algorithm necessitates the study of changes in global performance when modifying structural members locally and a method to receive these sensitivities in structural performance elementwise.

Thereafter, the algorithms were used with the local optimization approach. The implementation was carried out using already existing customized *Matlab* scripts for the automated load analysis performed with the finite element solver *FEDEM WindPower*.

Keywords: structural optimization; gradient-based algorithm; local approach; sensitivity analysis; offshore wind; support structures.

Acknowledgements

The period I have spent working on this project has been both challenging and rewarding. As a civil engineer and an offshore engineer candidate I am very interested in offshore wind support structures but I had no experience with optimization techniques before this project.

The learning curve has therefore been steep. I am pleased with the outcome of the project, but at the same time I realized that there is so much more I want to look into the field of optimization of offshore wind support structures. As usual with this kind of projects, you wish you had the knowledge you have today when you started the work, because then you would have done so much better.

I would like to give my sincere thanks to my advisors; prof. dr. Torgeir Moan, prof. dr. Michael Muskulus, dr. Eliz-Mari Lourens, prof. dr. Andrei Metrikine, to PhD-researcher Sebastian Schafhirt and to research engineer Yilmaz from Keppel Verolme, for their guidance and the kind cooperation we had.

Delft, December 2016,
Nikolaos Xyloudis

Table of contents

List of figures

List of tables

List of definitions

Constraint function Criteria that the system has to satisfy for each feasible design.

Design sensitivity analysis Analysis concerned with the relationship between design variables and the structural response determined by the laws of mechanics. It provides a quantitative estimate of desirable design change.

Design parameterization Process of defining the geometric parameters that characterize the structural model and selecting a subset of the geometric parameters as design variables.

Design variables System parameters that can be changed during the design process (e.g. the geometry).

Dred Scaling factor for defining the dimensions of legs and braces of the support structure.

Hessian The matrix of second partial derivatives of a function with respect to its variables.

Hot spot A point in structure where a fatigue crack may initiate due to the combined effect of structural stress fluctuation and the weld geometry or a similar notch.

Kbrace Brace stub at a K-joint (or Y-joint) of a jacket support structure.

Kleg Chord can at a K-joint (or Y-joint) of a jacket support structure.

Objective function Function to be minimized/maximized during the optimization process. It is also called *Cost function*.

Performance measure The result of structural analysis. It can be quite general in engineering fields and can include weight, stiffness, fatigue life, safety etc. In any case, performance measures are presumed to be measurable quantities.

Xbrace Brace stub at an X-joint of a jacket support structure.

Nomenclature

Latin symbols

∇	Differential operator	[–]
A	Cross sectional area	[m^2]
D	Outer diameter of a tubular element	[m]
d	Direction vector during a gradient-based optimization iteration	[–]
E	Modulus of Elasticity (Young' s Modulus)	[N/m^2]
F	Load vector	[N]
f	Objective function	[–]
g_j	Inequality constraint j	[–]
H	Hessian matrix	[–]
H_s	Significant wave height	[m]
h_k	Equality constraint k	[–]
I	Area moment of inertia	[m^4]
J	Total number of inequality constraint	[–]
K	Stiffness matrix	[N/m]
L	Length of a tubular element	[m]
L_c	Length of cut for welding	[m]
L_w	Length of a weld between tubular members	[m]
N	Total number of elements in the structure	[–]
Q	Total number of equality constraints	[–]
T	Wall thickness of a tubular element	[m]
T_p	Peak spectral period	[s]
t	Time	[s]
V	Total number of design variables	[–]
V_{hub}	Mean wind speed at hub height	[m/s]
W	Section modulus	[m^3]
X	Vector of design variables	[–]
X_i	Design variable i	[–]
z	Vertical coordinate measured positively upwards from MSL	[m]

Greek symbols

α	Step size used by an optimization algorithm	[–]
β	Ratio D_{brace}/D_{chord} in a tubular joint	[–]
γ	Peak enhancement parameter for JONSWAP spectrum or stress concentration factor parameter equal to D/2T	[–]
δ	Relative change in cost function used as optimization criterion	[euro]
ϑ	Topology angle between brace and chord at a tubular joint	[rad]
λ	Lagrange multiplier vector	[–]
σ_{yield}	Yield stress	[N/m^2]
σ_{HSS}	Hot spot stress	[N/m^2]
ψ	Performance measure	[–]

Subscripts

i	General index numbering	[–]
j	Number indicating inequality constraint	[–]
k	Number indicating iteration	[–]
n	Number indicating member element	[–]
q	Number indicating equality constraint	[–]

Abbreviations

DDM	Direct Differentiation Method	[–]
DLC	Design Load Case	[–]
DSA	Design Sensitivity Analysis	[–]
FLS	Fatigue Limit State	[–]
HSS	Hot Spot Stress	[N/m^2]
KKT	Karush Kuhn Tucker optimality conditions	[–]
MSL	Mean Sea Level	[m]
rpm	revolutions per minute	[–]
SQP	Sequential Quadratic Programming	[s]
TI	Turbulence Intensity (ratio of std. to mean wind speed)	[%]
ULS	Ultimate Limit State	[–]

1 INTRODUCTION

1.1 Motivation - Framework

Foundations (support structures) are one of the critical components of offshore wind farms, not only because of the overall stability of the offshore wind turbine (OWT) structure but also because of the financial viability of the project.

Offshore wind turbines are exposed to complex irregular and highly fluctuating environmental conditions. The analysis is therefore commonly performed in the time domain, resulting in a time consuming and computationally demanding optimization process.

In order to reduce the computational cost, an optimization approach using the principle of decomposition can be utilized. Here, it is assumed that when changing dimensions of a structural member, sectional response forces and moments remain constant. This assumption is made for both the actual changed member and for all other members of the structure. Additionally, this is assumed to be valid for a simultaneous change of all members all over the structure. It is the so-called *local optimization approach* and allows for an individual (local) optimization of each member of the structure. The local optimization approach has been successfully applied for the sizing of a full-height lattice tower and a jacket support structure, but under the use of a very simplified optimization algorithm for the local optimization of structural members.

1.2 Purpose of the project work

The objective of this project is the development of a gradient based optimization framework for the local optimization (sizing) of structural members of an offshore wind turbine with jacket support structure and the implementation of the framework using the principle of decomposition. The OWT structure used in this study consists of the UpWind reference jacket used within the OC4-project [3] which supports the NREL 5MW turbine [4].

1.3 Study approach

The development of the gradient based algorithm for the local optimization necessitates the study of changes in global performance when modifying structural members locally and a method to evaluate these sensitivities in structural performance elementwise.

Thereafter, the algorithms were used with the local optimization approach. The implementation was carried out using already existing customized *Matlab* scripts for the automated load analysis with the finite element solver *FEDEM WindPower*.

1.4 Software used

This section presents the software that has been used in this thesis. Both commercial programs and scripts from a previous project have been used.

- FEDEM Windpower version R7.1.2, a commercial simulation tool for dynamic analysis of complete wind turbine systems, FEDEM Technology AS. FEDEM Windpower is a version of Fedem specialized in simulating onshore and offshore wind turbines, with integrated soil description and control system.
- Matlab version R2016a, general purpose mathematical modelling program, MathWorks Inc.
- Python 3.4.0, open source programming language. Python is only used as the interface between Matlab and FEDEM.

1.5 Structure of this report

Following this introductory first chapter, in chapter 2, fundamental theory and background are given on all the aspects considering the current work (optimization methods, offshore environment, wind turbine support structures). This chapter is the result of a literature study that preceded the writing of this report. In chapter 3 the problem to be solved is described. All the measures are expressed in mathematical terms and values and expressions are given. In chapter 4 the developed optimization framework is described including a sensitivity analysis. In chapter 5 the results of this study are presented and in chapter 6 the conclusions are made along with recommendations for future work.

2 BACKGROUND

In this chapter the results of a thorough literature study and brief theoretical background is given. Structural optimization, gradient-based optimization techniques, sensitivity analysis and support structures for offshore wind farms are at the core of this chapter and establish a theoretical basis for the methods to be applied in the following chapters.

2.1 Design optimization

Design optimization refers to the process of generating improved designs. The purpose of many structural design problems is to find the best design among many possible candidates-solutions. For this reason, the design engineer has to specify the best possible design as well as the best possible candidates. A possible candidate must exist within a feasible design region to satisfy problem constraints. Every design in the feasible region is an acceptable (*feasible*) design, even if it is not the best one (*optimal*). The best design is usually the one that minimizes (or maximizes) the objective function of the design problem. Thus, the goal of the design optimization problem is to find a design that minimizes the objective function among all feasible designs [5].

We must first identify some *objective,* a quantitative measure of the performance of the system under study. An objective function is characterized as linear, when it is expressed as a linear combination of the design variables and as non-linear otherwise. This objective could be profit (or cost), time, potential energy, natural frequency, mass, volume, fatigue life or any quantity or combination of quantities that can be represented by a single number.

The objective depends on certain characteristics (or parameters) of the system which are subdivided into so-called *preassigned parameters* and *design variables* (or *unknowns*). Each one of the design variables has its own domain. Having in mind that mechanical structures have a physical substance and they are not abstract theoretical concepts, both the lower and the upper bounds must obtain logical values with physical meaning. The number of the design variables denotes the dimension of the solution space of the corresponding optimization problem. Our goal is to find values of the variables that optimize the objective. Any set of values of the design variables defines a design of the structure and may be represented as a point in the so-called *design space*. Often the variables are restricted, or constrained, in some way. For instance, the interest rate on a loan cannot be negative.

Constraint functions are the criteria that the system has to satisfy for each feasible design. Among all design ranges, those that satisfy the constraint functions are candidates for the optimum design [5]. The constraints can be either equality or inequality expressions. As for the objective function, a constraint is called linear, when it is expressed as a linear combination of the design variables, and non-linear in any other case. The constraints divide the design space into two domains, the *feasible* domain where the constraints are satisfied, and the *infeasible* domain where at least one of the constraints is violated. In most practical problems, the optimal design is found on the boundary between the feasible and infeasible domains.

The constraint functions together with the objective function are called also *performance measures*. In the special case where the objective function and all of the constraints (both equality and inequality) are linear, the optimization problem is characterized as *linear*. In any other case, it is characterized as *non-linear*.

The process of identifying objective, variables and constraints for a given problem is known as *modeling*. Construction of an appropriate model is the first step - sometimes the most important step - in the optimization process. If the model is too simplistic, it will not give useful insights into the practical problem. If it is too complex, it may be too difficult to solve.

Once the model has been formulated, computer *analysis* (computational tools) or sometimes experimental tools are used to judge the quality of our designs. We use finite element analysis to calculate stresses, deflections, vibration frequencies, etc. of a structure. When we perform an analysis, we create a mathematical idealization of some physical system in order to obtain estimates of certain response quantities. The goal is to obtain an accurate prediction (within the limits of the analysis model) of the responses that can be expected from the real structure.

An *optimization algorithm* is used afterwards to find a solution, usually with the help of a computer. There is no universal optimization algorithm but rather a collection of algorithms, each of which is tailored to a particular type of optimization problem. The responsibility of choosing the algorithm that is appropriate for a specific application often falls on the user. This choice is an important one, as it may determine whether the problem is solved rapidly or slowly and, indeed, whether the solution is found at all.

After an optimization algorithm has been applied to the model, we must be able to recognize whether it has succeeded in its task of finding a solution. In many cases, there are elegant mathematical expressions known as *optimality conditions* for checking that the current set of variables is indeed the solution of the problem. If the optimality conditions are not satisfied, they may give useful information on how the current estimate of the solution can be improved. The model may be improved by applying techniques such as *sensitivity analysis*, which reveals the sensitivity of the solution to changes in the model and data [6]. For an effective use of design optimization, a well posed problem goes along with the following basic principles:

- The design goals are clear. An explicit definition of the objective function permits the design engineer to specify the structural performance from which the sensitivity information can be computed.
- Constraints are meaningful and well-conditioned. Design variables must be able to influence the objective function.
- Design variables are chosen carefully.

2.2 Structural optimization

The focus of this thesis is on structural optimization which is usually a nonlinear problem. Nonlinear problems arise in various fields of real life and there is a wide variety of approaches for solving the resulting nonlinear optimization models. Structural optimization began in the 1960s and a general structural optimization framework can be described by the figure below.

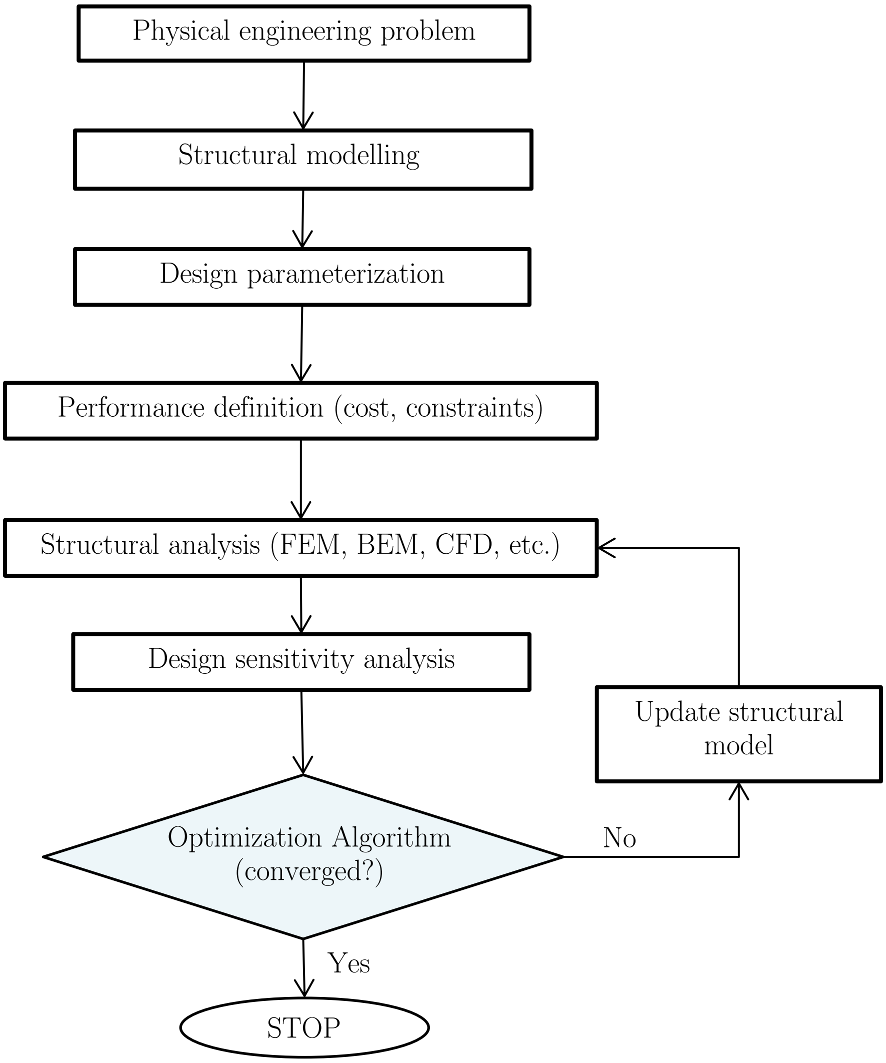

Physical engineering problem
Structural modelling
Design parameterization
Performance definition (cost, constraints)
Structural analysis (FEM, BEM, CFD, etc.)
Design sensitivity analysis
Update structural model
Optimization Algorithm (converged?)
No
Yes
STOP

Figure 2.1: Structural optimization process [5]

Structural optimization problems may be categorized in three large groups with respect to the design variables considered:

- *Size optimization* - Include those methodologies that vary the sizes (e.g. diameters or cross-sections) of already defined structural members in order for the objective function to be minimized and no constraint to be violated. This implies that before such an optimization can be done, the structure to be analyzed must be defined.

- *Topology optimization* - Include those procedures which result in the formation of holes inside the design domain by appropriately moving inner nodes; in this way, redundant material is removed and the objective function is minimized while no constraint is violated.

- *Shape optimization* - Include those methods according to which the border nodes are appropriately moved so that again the objective function is minimized and all of the constraints are fulfilled.

In the most general case, the aforementioned optimization problems are not uncoupled. Methodologies that simultaneously deal with all three problems are called *layout optimization methods*.

2.3 Optimization problems

An important step in the optimization process is classifying the optimization model, since there is no general algorithm to solve any problem but methods for solving optimization problems are effective in particular types of problem. The problems to be optimized can be classified in numerous categories according to different criteria. The table below gives a comprehensive overview of optimization problems.

Table 2.1: Overview of optimization problems

Classification Criterion	Problem Characterization
Mathematical form of equations involved	Linear Nonlinear
Number of objective functions	Single Multi-objective
Nature of design variables	Deterministic Stochastic
Permissible value of design variables	Continuous Discrete (Integer, Mixed-integer etc.)
Presence of constraints	Constrained Unconstrained
Convexity	Convex Non-convex
Number of involved disciplines	Single discipline Multidisciplinary design optimization (MDO)
Availability of algebraic model (availability of accurate problem data)	Deterministic Uncertainty (Stochastic Prog., Robust Opt.)

The optimization of a jacket support structure encountered in this work, is a nonlinear, single-objective, single-discipline, deterministic, continuous and constrained problem.

2.3.1 Constrained optimization

Most engineering problems have constraints that must be satisfied during the design optimization process (*constrained optimization*). Constraints act as bounds in the design space which is the region that defines all of the possible solutions to the problem. They directly or indirectly impose limits on the ranges of variation of the design variables and define hyper-surfaces that encircle the set of acceptable designs. The mathematical expression of the design problem is called the basic optimization problem statement and for the constraint optimization, it can be written as:

$$
\begin{array}{llll}
\text{find:} & X = \left[x_1, x_2, \ldots, x_V\right],\ x_i \in \Re & & \text{Design variable vector} \\
\text{to minimize:} & f(X) & & \text{Objective function} \\
\text{subject to:} & g_j(X) \leq 0, & \text{j=1,…, J} & \text{Inequality constraints} \\
 & h_l(X) = 0, & \text{q=1,…, Q} & \text{Equality constraints} \\
 & x_i^L \leq x_i \leq x_i^U, & \text{i=1,…, V} & \text{Side constraints}
\end{array}
\qquad (2.1)
$$

where *V* is the number of design variables and *J, Q* are the numbers of inequality and equality constraints respectively. The objective function is the scalar quantity to be minimized. It is a function of the set of design variables. Although the problem is stated as a minimization task, we can easily maximize a function by minimizing its negative. Equality, inequality and side constraints are possible and these types of constraints are handled separately.

Inequality constraints are placed on the design variables to limit the region of search, for example, tubes whose wall thicknesses are less than one-tenth of the outer radius. The inequality constraints are expressed in a less than or equal to zero form by convention. We have satisfied a constraint if the constraint value is negative. We have violated the constraint if its value is positive.

Equality constraints, if present, must be satisfied exactly at the optimal design. Note that if there are as many equality constraints as design variables, a unique solution exists (as long as the equalities are linearly independent). This solution can be found using standard algebraic methods. A finite element analysis belongs to this category of problem. When the number and type of constraints do not enable a direct, unique solution, the job becomes complex and numerical optimizers must be used. We need a systematic method of searching for an optimal design.

Another classification of constraints can be made with respect to their nature. We can identify two types: *geometrical constraints* and *behavioral constraints*.

Geometrical (or side) constraints are restrictions imposed explicitly on the design variables due to considerations such as manufacturing limitations, physical practicability, aesthetics, etc. Constraints of this kind are typically inequality constraints that specify lower or upper bounds on the design variables, but may also be equality constraints e.g. linkage constraints that prescribe given proportions between a group of design variables.

Behavioral constraints are generally nonlinear and implicit in terms of the design variables and they may be subdivided in two types. The first type consists of equality constraints such as state and compatibility equations governing the structural response associated with the loading condition(s) under consideration. The second type of behavioral constraints comprises inequality constraints that specify restrictions on those quantities that characterize the response of the structure. These constraints may impose bounds on local (or point-wise) quantities like stresses and deflections, or on global (or integral) quantities such as compliance, natural vibration frequencies, etc.

2.3.2 Convexity in optimization

Convexity is a mathematical term, defining that something is curved outward. If a straight line between two points in a domain can never be drawn such that the line crosses the domain boundary, the mentioned domain is a convex one. Similarly, if the domain S ≥ f is a convex domain, f is a convex function.

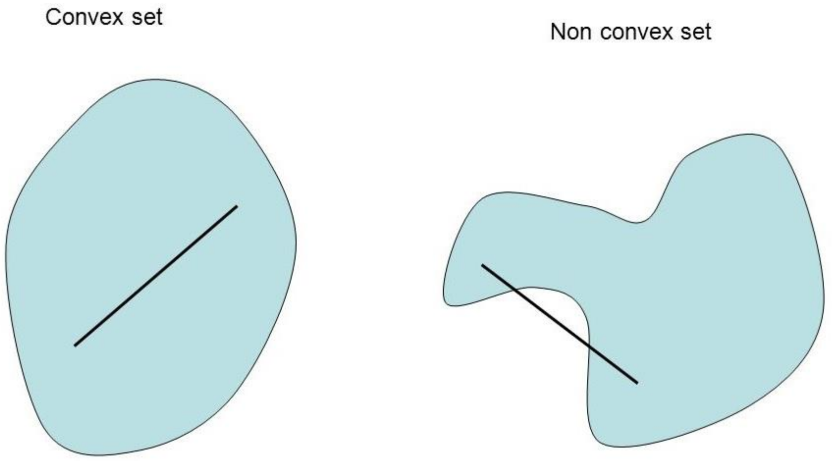


Figure 2.2: Convex and non-convex (concave) sets in 2D

A convex function is one whose graph slopes everywhere toward its minimum value, whereas a nonconvex function (bottom) may have many basins or local minima.

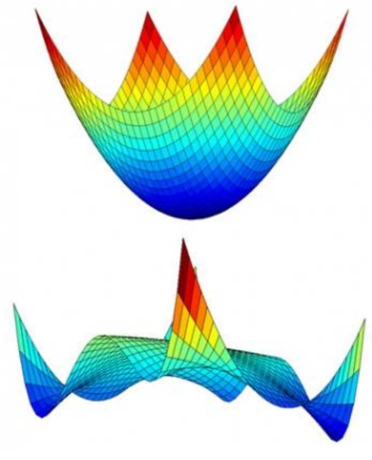


Figure 2.3: Convex (top) and non-convex (bottom) functions of two variables

In continuous optimization (variables are continuous sets and functions are also continuous), a convex problem is the special case where a convex function is optimized over a convex domain. An optimization problem can be characterized as *convex* according to the following:

1. f is convex &		1. f is convex &
2. every g is convex &	or	2. feasible region is a convex set
3. every h is linear		

Convexity is a strong assumption. The advantage of a convex problem is that any local minimum will also be a global minimum. On the other hand, if either the objective function or the domain are non-convex, they introduce possibilities for local minima.

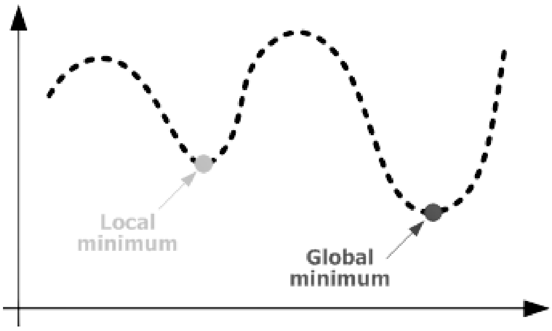


Figure 2.4: Definition of global and local minima

The methods discussed in the following sections are local optimization methods, which can be defined as methods that adapt the solution locally and that are guaranteed to converge to a local minimum only, when they have to deal with non-convex functions. However, extra techniques exist that can help these methods and lead them to global optimum solutions even for non-convex problems.

2.4 Optimization methods

For an engineering optimization problem, once the formulation is established, the problem can be regarded as a mathematical problem. Numerous techniques are available to solve an optimization problem, all of which are classified as numerical optimization methods and are in principle mathematical tools. Numerical methods do not take into consideration inherent characteristics of the problem at hand. Therefore, they deal with a structural optimization problem in the same way as any other optimization problem.

Usually, the optimization method is an iterative one for finding the minimizer of an optimization problem. The basic idea is that for a given design (initial point) X_0 that defines the structural model, structural analysis provides the values of the objective and constraint functions for the algorithm. Then the algorithm generates an iterative sequence X^k by means of some iterative rule, such that when X^k is a finite sequence, the last point is the optimal solution of the model problem; when X^k is infinite, it has a limit point which is the optimal solution of the model problem. When a given convergence rule is satisfied, the iteration will be terminated. Let X^k be the k-th iteration, d^k the k-th direction and a^k the k-th step length. Then we have:

$$X^{k+1} = X^k + a^k d^k \text{ with } f\left(X^{k+1}\right) < f\left(X^k\right) \tag{2.2}$$

Note here that in a general case problem with many variables, a point in the design space is defined by a vector of coordinates and therefore X^k, d^k and a^k in this case are vectors.

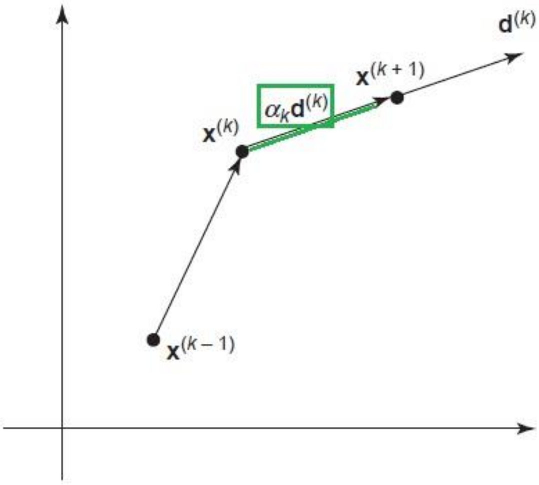


Figure 2.5: Conceptual diagram for iterative steps of an optimization method

Optimization methods may be categorized in two large groups, the stochastic methods and the deterministic methods (can be either derivative-free or gradient-based). The *stochastic* techniques are derivative-free, at least at their basic version, they are not stuck in local optima and are good in solving very complex problems. However stochastic techniques need to perform a thorough exploration and exploitation of the feasible region (a lot of iterations needed), which in turn results to a cumulated high computational cost. In addition, the solution they find is not the best possible but a solution very close to it, an "approximate" optimum. Genetic algorithms and other heuristic techniques belong to this category. As an example, the simplest algorithm of this type is *random search*. Random algorithms compare the current iterate x with a randomly generated candidate. The current iterate is updated only if the candidate is a better point (in terms of the objective function). As can be understood, the number of function evaluations for this type of method becomes prohibitive.

On the other hand, the *deterministic* optimization procedures may use either objective function values only (*zeroth order* or *non-gradient methods*), or gradients (*first order methods*) or even Hessian information (*second order methods*). The basic difference between gradient and non-gradient methods is the nature of the pattern search. In the former methods, the implementation of gradient and/or Hessian information demands the calculation of derivatives, which has a significant computational cost, may cause numerical instabilities and may result in locating local rather than global minima.

Obviously, each technique has its own advantages and disadvantages. Generally, stochastic methods for direct search outperform in terms of time the deterministic ones, when the number of the design variables is small. On the contrary, as the dimension of the design space increases so does the relative efficiency of the deterministic procedures with respect to the stochastic ones [7].

2.4.1 Gradient-based techniques

This thesis will focus on the gradient based methods. For most practical design tasks, we are usually dealing with a vector of design variables (multivariable optimization). The resultant vector of partial derivatives is called the *gradient.* Numerical optimization algorithms that rely on gradient (or sensitivity) information are termed *gradient based* (or *derivative based*).

The distinction between most optimization algorithms is in how the search direction is chosen and how the step size determined. The different gradient-based algorithms that exist, differ mostly in and are usually named, after the logic used to determine the search direction. There are two fundamental strategies for moving from the current point X^k to a new point X^{k+1}. These are the *line search* strategy and the *trust region* strategy.

Line search starts by fixing the direction and then identifying an appropriate distance. In trust region, first a maximum distance is chosen and then a direction and step determined that attain the best improvement possible subject to this distance constraint.

The computational method in line search strategy, in order to find a solution to the general problem in Eq. (2.1) is:

1. First, for a given point in the design space, we find a direction ***d*** that it is both descending and feasible, i.e. it can reduce the objective function *f* while correcting for any constraint violations. The following must hold for ***d*** to be a feasible direction:

$$\nabla f(X)^T d < 0 \tag{2.3}$$

$$\nabla g_j(X)^T d < 0 \tag{2.4}$$

$$\nabla h_q(X)^T d = 0 \tag{2.5}$$

   Equation (2.3) implies that we are descending, and Eq. (2.4) and (2.5) imply that we are increasing feasibility (by moving in the direction tangential to the active set for the inequality constraints and parallel for the equality constraints). Note that Eq. (2.4) must hold only for the active (that are binding at the considered point) inequality constraints.

2. Determine the step size of movement ***a*** in the direction of ***d*.** The determination of the step size is done in many algorithms by running a (sub)algorithm for minimizing the one-dimensional function $\phi(a) = f(X + ad)$. This is called line minimization (or line search [8]), i.e., f is minimized over the line $x + ad$. Alternative options for the step size selection can be found in [5, 8, 9]. We can also choose a constant step size or a diminishing step size.

We continue to repeat this procedure until we cannot reduce the objective function any further i.e. until we can make no further improvement in our objective without violating any of the constraints.

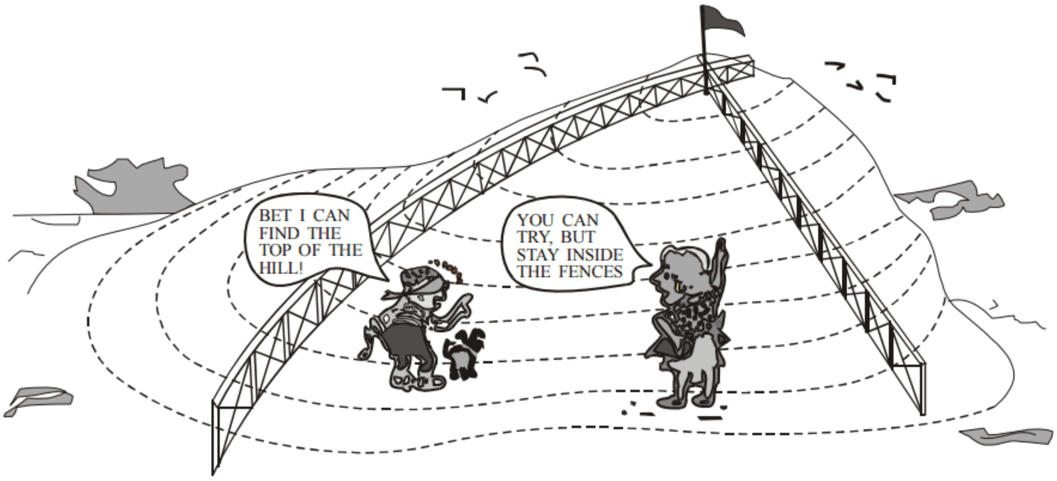

BET I CAN FIND THE TOP OF THE HILL!
YOU CAN TRY, BUT STAY INSIDE THE FENCES

Figure 2.6: Gradient based method in a real life problem [9]

This two-step process can be explained using the example of a blindfolded boy on a hill (Figure 2.6) used by [9]. The boy has to reach the highest point on the hill (the objective function), while staying inside the fences (the constraints). The design variables are the x and y coordinates of the boy. In an analogy to gradient-based optimization, the blindfolded boy can reach the top of the hill by taking a single step in the x direction and another step in the y direction. Based on the information gained from these two steps, he can estimate a direction that would take him uphill. The boy can then start walking in this direction until no more progress is made, which may include reaching a fence. At this point the boy can again take two small steps to determine a new direction that will take him uphill, while staying inside the fences and continue the process until he reaches the top of the hill. Some methods move mostly along the constraint boundaries, some mostly on the inside (interior point algorithms).

The gradient-based methods for constrained optimization are divided in two main categories, the primal and the transformation methods. The *primal methods* attempt to incorporate information about the constraints directly into the optimization problem. They start with a candidate in the feasible set and search only the feasible set for an optimal solution. The main characteristic of these algorithms is that they find new candidates that simultaneously decrease the objective function at each step, while remaining feasible.

Transformation methods (or *Penalty* and *Barrier* methods) use methods for unconstrained optimization to solve the constrained problem. The basic concept here is to convert the original constrained optimization problem to a sequence of unconstrained problems through the use of penalties (penalty functions) for constraints. The constraint functions are transformed to a penalty function and added to the objective function. Then any of the many existing unconstrained optimization methods can be exploited.

In Figure 2.7 on the next page, an effort has been done to present an overview of the optimization methods existing and found in literature [5, 6, 9, 10, 8, 11, 12, 13]. The performance of an optimization algorithm critically depends on the characteristics of the design problem and the types of objective and constraint functions. The most established gradient based methods for the general constrained optimization problem with nonlinear functions [5], are:

Constrained Steepest Descent Method (CSD) - Design sensitivity of a penalized cost function is used in order to combine effects from original objective function (unconstrained) and the violated constraint functions. The method converges to a local minimum point starting from an arbitrary point, feasible or infeasible. However, the starting point can affect the performance of the algorithm [14].

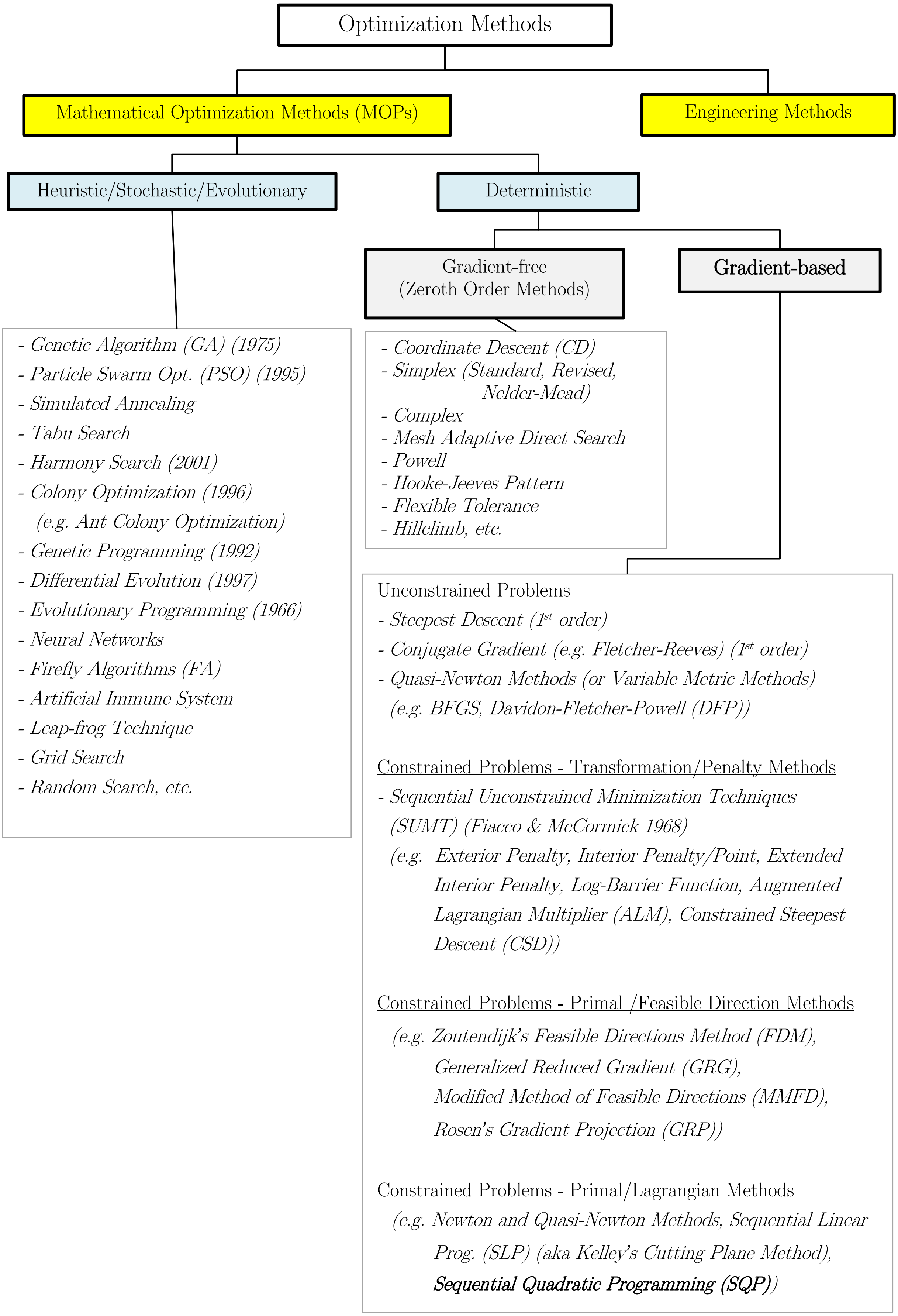

Optimization Methods
Mathematical Optimization Methods (MOPs)
Engineering Methods
Heuristic/Stochastic/Evolutionary
Deterministic
Gradient-free
(Zeroth Order Methods)
Gradient-based
- Genetic Algorithm (GA) (1975)
- Particle Swarm Opt. (PSO) (1995)
- Simulated Annealing
- Tabu Search
- Harmony Search (2001)
- Colony Optimization (1996)
(e.g. Ant Colony Optimization)
- Genetic Programming (1992)
- Differential Evolution (1997)
- Evolutionary Programming (1966)
- Neural Networks
- Firefly Algorithms (FA)
- Artificial Immune System
- Leap-frog Technique
- Grid Search
- Random Search, etc.
- Coordinate Descent (CD)
- Simplex (Standard, Revised,
Nelder-Mead)
- Complex
- Mesh Adaptive Direct Search
- Powell
- Hooke-Jeeves Pattern
- Flexible Tolerance
- Hillclimb, etc.
Unconstrained Problems
- Steepest Descent (1st order)
- Conjugate Gradient (e.g. Fletcher-Reeves) (1st order)
- Quasi-Newton Methods (or Variable Metric Methods)
(e.g. BFGS, Davidon-Fletcher-Powell (DFP))
Constrained Problems - Transformation/Penalty Methods
- Sequential Unconstrained Minimization Techniques
(SUMT) (Fiacco & McCormick 1968)
(e.g. Exterior Penalty, Interior Penalty/Point, Extended
Interior Penalty, Log-Barrier Function, Augmented
Lagrangian Multiplier (ALM), Constrained Steepest
Descent (CSD))
Constrained Problems - Primal /Feasible Direction Methods
(e.g. Zoutendijk's Feasible Directions Method (FDM),
Generalized Reduced Gradient (GRG),
Modified Method of Feasible Directions (MMFD),
Rosen's Gradient Projection (GRP))
Constrained Problems - Primal/Lagrangian Methods
(e.g. Newton and Quasi-Newton Methods, Sequential Linear
Prog. (SLP) (aka Kelley's Cutting Plane Method),
Sequential Quadratic Programming (SQP))

Figure 2.7: Review of optimization methods for multivariable functions

Zoutendijk's Feasible Directions Method (FDM) - A direction for search is determined by utilizing linear approximations (the gradients of objective and constraint functions) and then the actual function values are used to guide the search along this direction. It attempts to follow the constraint boundaries without being precisely tangent to them. This is achieved by the introduction of a push-off factor ϑ during the direction calculation. Two features of the method are attractive, the guaranteed convergence even for nonconvex problems and the generation of feasible points at each iteration. Disadvantage of the method is that it may be subject to jamming and have a slow rate of convergence as a result. The method is described extensively in [9], [15] and [16].

Generalized Reduced Gradient (GRG) -The GRG algorithm was first developed by Abadie and Carpenter (1969). A search direction is found such that any active constraints remain precisely active for a small move in this direction. This is achieved by adding one slack variable to each inequality constraint to give an addition to the total number of variables (initial + slack). The basic concept of GRG method is to recognize that for each equality constraint, we can define a dependent design variable, thereby reducing the total number of independent design variables. For this reason, the method can be a powerful design tool if there are only a few inequality constraints and several equality constraints. On the other hand, if the problem is highly nonlinear it may not converge [9].

Modified Method of Feasible Directions (MMFD) -This method modifies (by setting the push-off factors equal to zero) the FDM so that the method resembles the GRG method without using the slack variables of it [9].

Rosen's Gradient Projection Method (GRP) - This method is very similar to the CSD method and is based on a modification of the original method of the FDM. The modification provides a search direction that follows the active constraint bounds (rather than moving inside the feasible region) towards the optimum. The steepest descent direction (negative of the gradient of the objective function) is projected onto the working surface (the subset of the constraints that are currently active). However, this may not be a feasible direction since the working surface may be curved. To deal with curvature, one searches for a feasible descent direction along an embedded curve within the constraint surface. When the starting point is infeasible, a correction algorithm (correction steps) should follow the optimization method in order to obtain a feasible design during the iteration process. A disadvantage is that convergence of the algorithm is tedious to enforce but the GRP algorithm is extremely robust, has the property of quickly finding the feasible design space and is widely used in structural optimization [13].

Sequential Linear Programming (SLP) - It simplifies the general non-linear constrained optimization problem to an equivalent linear problem by creating linear approximations to the objective and constraint functions about the current design point. In particular, it employs the first order Taylor series expansion to iteratively develop and solve a new LP subprogram to solve the Karush-Kuhn-Tucker (KKT) (see following section 2.4.2) conditions associated with the NP problem. The method is very sensitive to move limits and may not always find a feasible solution. SLP is generally not robust. As a result, it is considered inferior to the MMFD and SQP algorithms.

Sequential Quadratic Programming (SQP) - This algorithm finds a search direction by solving an approximate problem based on a quadratic approximation of the objective function and linear approximations of the constraint functions. SQP methods are currently considered to represent the state of the-art gradient based approach to the solution of nonlinear constraint optimization problems [11]. An SQP method will be used in the optimization process proposed in the current thesis. More on the method and its performance can be found in chapter 4.3.

2.4.2 Optimality criteria

A critical part of the overall optimization process is determining when to stop the optimization process. In the numerical search algorithm, it is necessary to have some formal definition of an optimum. Any trial design can then be measured against this criterion to see if it is met and an optimum has been found. Optimality conditions are based on differential calculus and they are said to be of first order if they involve only first derivatives. They are of second order if second derivatives are also required. Since differential calculus gives only local information about the problem and when we do not include convexity assumptions, only relative minima can be characterized. If (2.2) is satisfied, it implies that the gradient vector $\nabla f(X^k)$ tends to zero and the sequence X^k converges to a stationary point. Some of the termination criteria that can be used are [9]:

- Maximum number of iterations
- Absolute change in the objective function
- Relative change in the objective function (this is the most natural stopping criterion for an unconstrained problem, $\left|\nabla f(X^k)\right| \leq \varepsilon$ where ε is a prescribed tolerance).
- The KKT conditions

The KKT conditions state that if we compute the gradients of the objective function and all the active constraint functions at a point, then the vector sum of them must be equal to zero given an appropriate choice of multiplying factors λ (called the Lagrange multipliers). This condition is described in Eq. (2.7c). Note that, constraints that are not active at the proposed point are not included in the vector summation. Lagrange multipliers allow us to maximize or minimize functions with the constraint that we only consider points on a certain surface. Factors λ are 'dummy' variables we introduce and we don't care about their value but help us define the X values at an optimum point. The KKT conditions provide the necessary conditions for a local optimum and can be summarized as:

$$X^* \text{ is feasible} \tag{2.7a}$$

$$\text{Constraints are satisfied: } g(X^*) \leq 0 \text{ and } h(X^*) = 0 \tag{2.7b}$$

$$\nabla L(X^*, \lambda) = 0 \;\Rightarrow\; \nabla f(X^*) + \sum_{j=1}^{J} \lambda_j \nabla g_j(X^*) + \sum_{q=1}^{Q} \lambda_{J+q} \nabla h_q(X^*) = 0 \tag{2.7c}$$

$$\lambda g(X^*) = 0 \tag{2.7d}$$

Where $L(X, \lambda) = f(X) + \sum_{j=1}^{J} \lambda_j g_j(X) + \sum_{q=1}^{Q} \lambda_{J+q} h_q(X)$ is the Lagrangian and λ are the Lagrange multipliers; $\lambda_j \geq 0$ and λ_{J+q} are unrestricted in sign. The proof and complete theory related to KKT can be found in [17].

The KKT conditions are useful for identifying a local optimum, but cannot indicate whether a global optimum has been found. However, if the design space is convex (see chapter 2.3.2), the KKT conditions define a global optimum. For a design space to be convex, the Hessian matrix of the objective function and all constraints must be positive definite for all possible combinations of the design variables. Positive definite H, means that the scalar $X^T H X$ is positive for any non-zero column vector X of real numbers.

The KKT conditions are useful even if there are no active constraints at the optimum. In this case, only the objective function gradient is considered and this is identically equal to zero; i.e., any finite move in any direction will not decrease the objective function.

2.5 Structural sensitivity analysis

Sensitivity analysis is a necessary tool in the design optimization process. By the term sensitivity analysis here, we are referring to gradient computations. Sensitivity analysis computes the gradients (sensitivities) of the performance measures (objective and constraint functions) with respect to the design variables. The design sensitivity information provides a quantitative estimate of the desirable design change. Based on the design sensitivity results, an engineer can decide on the direction and amount of design change (step size) needed to improve the objective function [5]. Design sensitivity analysis comes in between the structural analysis and the optimization algorithm, i.e. it uses the output from the structural analysis and provides the input for the optimization algorithm.

In optimization, we are interested in how a quantity (obj. or constraint function) changes as some other related quantities (design variables) change. The usual way to calculate or specify how the quantity changes (sensitivity) is with the use of derivatives.

In the case of functions with two or more variables we need to consider derivatives of the function with respect to all variables. This requires the use of partial derivatives arranged in the gradient vector. The gradient represents the slope of the tangent of the graph of the function.

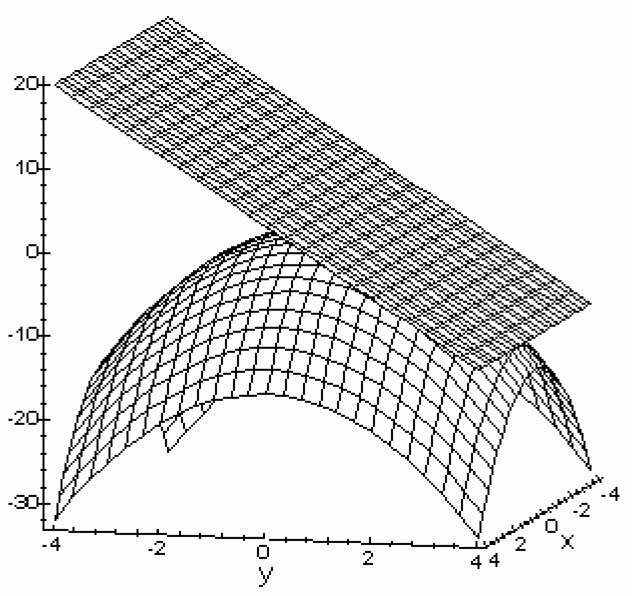


Figure 2.8: Illustration of gradient plane (flat area) on a two-variable surface at a given point

An easy way to envision the gradient of a two variable function *f(x,y)* is to imagine a X-Y grid on a flat surface and *f(x,y)* is the height of the surface of interest above the flat grid surface. The gradient is related to the slope of the surface at every point. The gradient is represented using the symbol "∇".

Apparently, sensitivity analysis presumes differentiability of the objective and constraint functions with respect to the design variables, at least around the current design point. If the function is explicitly dependent on the design variables, only algebraic handling is required to obtain the design sensitivity of such a function. Sensitivity analysis can be performed by:

Approximation approach (also called *Divided/Finite Difference Method* or *Numerical differentiation*) – It is using either *forward/backward finite difference* or *central finite* *difference* method. Advantages are that it is very easy to code and it can always be applied, meaning that the sensitivities are calculated the same way independently of the problem type considered. Disadvantages are the high computation costs (many function evaluations for large scale problems with many design variables) and the reduced accuracy (especially for highly nonlinear problems). Techniques have of course been developed to mitigate these two problems, but these techniques increase rapidly in programming complexity.

*Analytical Sensitivities (*also called *Algebraic/Symbolic differentiation)* – It is exact but an explicit representation is difficult to obtain. This technique is perfectly mechanical, giving access to the expression that defines the function of interest. The advantage of symbolic differentiation is that it enables fast and accurate calculations and, if the original function is simple enough, it can produce a legible symbolic description of the derivative, which can be very valuable for further thinking and easier computer calculation. However, drawback can be that careless symbolic differentiation can easily produce exponentially large symbolic expressions which take exponentially long to evaluate.

*Automatic Differentiation (*also called *Algorithmic/Computational differentiation)* – It is a set of techniques which apply the rules for differentiating sums, products, functions of functions and so on, to numerically evaluate the derivative of a function specified by a computer program. AD is distinct from – although clearly related to – software for symbolic differentiation which operates on mathematical formulae and produces corresponding formulae for derivatives with respect to chosen variables.

Derivatives of arbitrary order can be computed automatically and accurately to working precision. This technique can handle any derivative or gradient, of any function you can program, or of any program that computes a function, with machine accuracy and ideal asymptotic efficiency. In the context of optimization, automatic differentiation can provide derivatives of the objective function without requiring a user to do any calculus. The derivatives are evaluated as accurately as the function itself (i.e., subject only to possible rounding errors). Truncation errors do not arise as they do for derivatives estimated by finite differences. AD comes in two basic forms. The first form is called *forward mode*, which computes directional derivatives. The *reverse mode* on the other hand computes directional gradients, is much more efficient than the forward mode, but can be expensive in its memory requirements.

2.6 Design of offshore wind support structures

Several definitions of offshore wind support structure can be found among different literature references. In this thesis, the support structure will be defined as the whole part below the tower, excluding the structure under the soil. Offshore wind turbines can be mounted either on a bottom-fixed (or grounded to the sea bed) structure, or on a floating structure. Bottom-fixed structures are the dominating solution among wind farms today. Noticeable technical designs among the bottom-fixed solutions are: monopile, tripod, gravity foundation and jackets. Usually, a tubular tower is mounted on top of the respective foundation structures, as illustrated in Figure 2.9 and Figure 2.10.

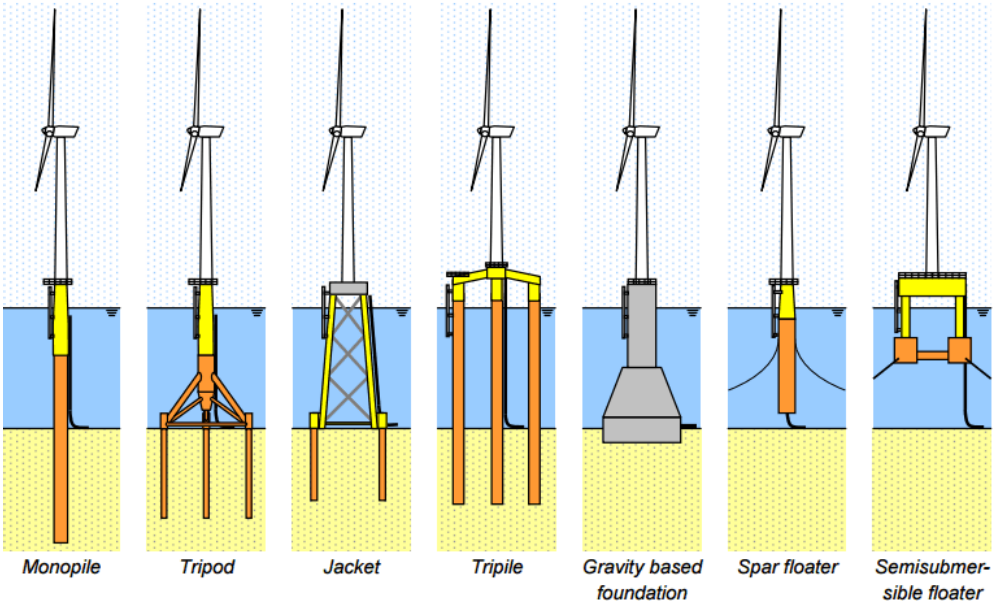


Figure 2.9: Existing offshore wind energy support structures concepts [18]

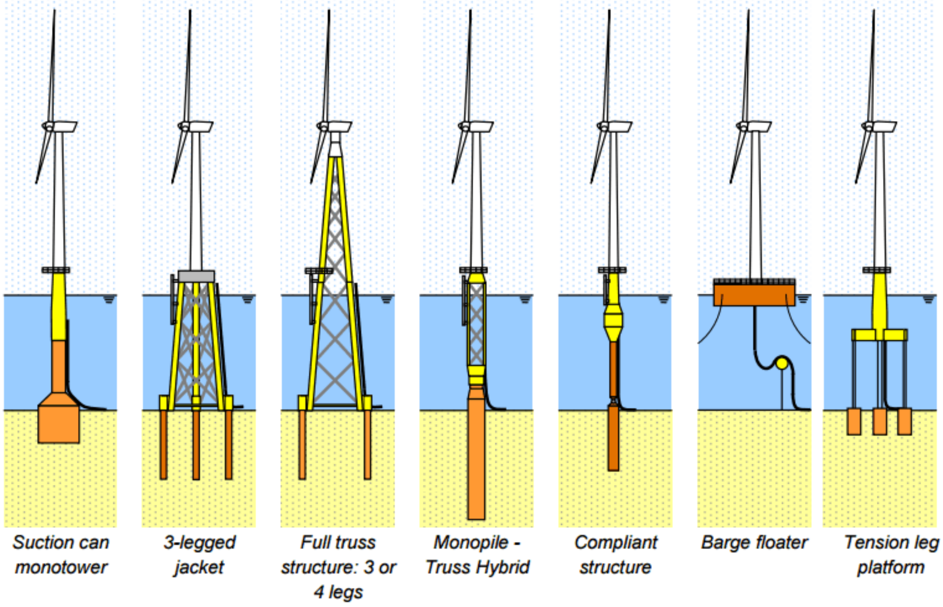


Figure 2.10: Alternative offshore wind energy support structures concepts [18]

Jackets were among the very first structures to be used in the offshore oil & gas industry. A jacket is a structure made up of three or four legs connected by slender braces. All the elements are tubular and they are joined by welding. Each of the joints has to be specially fabricated, taking a lot of time to complete the whole structure. A jacket-like structure will need some additional consideration. The large number of relatively slender members with joints connecting them can be excited locally, in additional to global vibrations of the tower, giving additional contributions to the fatigue loading. Usually the result is a structure with fatigue lifetime as the main dimensioning factor. As this thesis will focus on the optimization of a jacket support structure, in the table below we summarize the pros and cons of this concept.

Table 2.2: Jacket structures – advantages and disadvantages

Advantages	*Disadvantages*
Lightweight and stiff structure	Complexity of fabrication. Large number of joints required (may lead to high cost)
Better global load transmission compared to monopiles	Logistical issues due to large size
Economically viable on transitional/deep waters	Complex connection to transition pieces
No scour protection needed	
Structural redundancy	
Low soil dependency	
Good response to wave loads	

2.6.1 Loads in the offshore environment

There is much experience in designing offshore support structures within the oil and gas sector. The loading of oil and gas installations is often dominated by the huge selfweight. The structures are therefore less exposed to dynamic excitation. The loading of an offshore wind turbine is principally different. It is characteristic for an offshore wind turbine that the support structure is subjected to large moments at seabed and strong cyclic loading, originating from wind and wave loads on the structure.

One of the main aims of the support structure is to transfer all the loads from the wind turbine structure to the ground within the allowable deformations. The loads acting on the wind turbine and its support structure are ultimately transferred to the foundation and the resulting loading is primarily governed by a large moment at seabed level while the horizontal and vertical loads are comparably small.

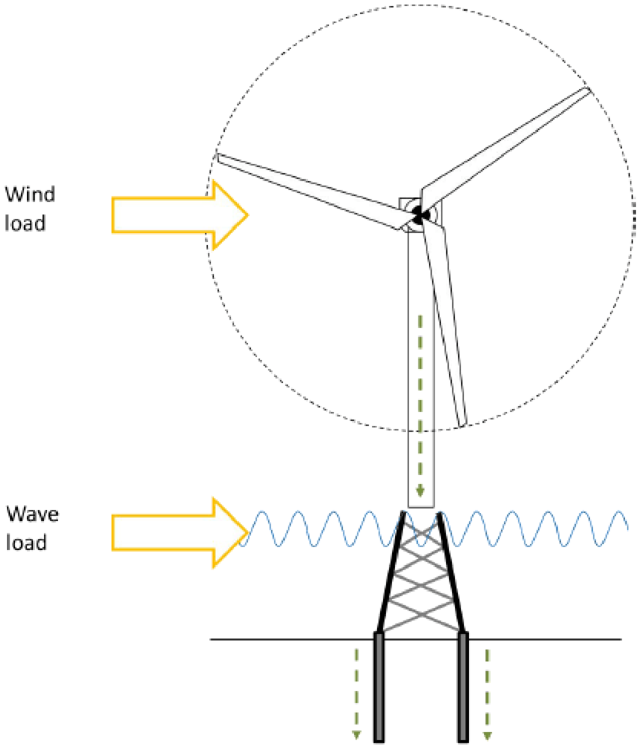


Figure 2.11: Loads acting on the support structure [19]

The loads can be classified into two types, static and dynamic (environmental) loads:

- Static or dead loads because of the selfweight of the components of the offshore wind turbine and its substructure (tower + support structure) and because of marine growth.

- The hydrodynamic load caused by the waves and current crashing against the substructure. The waves produce a dynamic load and its magnitude depends on the wave height and wave period, while current is considered as static load and its magnitude depends on the current speed (assumed to be constant).

- The aerodynamic load caused by the wind acting at the hub level (top of the tower) from the rotating blades and on the tower. The magnitude of the dynamic component depends on the turbulence intensity which by turn is dependent on the height and the roughness of the terrain. Rougher terrain and lower altitude give higher turbulence intensities. Considering the wind load, there are two frequencies that are critical to the wind turbine design. The first one corresponds to the excitation caused by mass and aerodynamic imbalances in the rotor and the blades. This load has a frequency equal to the rotational frequency of the rotor (corresponding to a full revolution and referred to as 1P loading). The second one corresponds to the excitation caused by the rotation of the blades. Vibrations are caused on the tower by blade shadowing effects (referred to as 2P/3P loading in the literature). The blades of the wind turbine passing in front of the tower cause a shadowing effect and produce a loss of wind load on the tower. This impulse-like excitation has a frequency equal to three times (3P) the rotational frequency of the turbine for three bladed wind turbines and two times (2P) the rotational frequency of the turbine for a two-bladed turbine.

Significant vibrations of the support structure of an offshore wind turbine can lead to lower power production, increased fatigue damage and even collapsing. For this reason, the frequency of the loads exciting the offshore wind turbine should be avoided to coincide with the natural frequencies of the support structure (especially with its fundamental structural frequency, the 1st nat. freq.). Extreme waves typically occur in the range 0.07-0.14 Hz, the energy rich wind turbulence lies below 0.1 Hz and the rotational speed for a modern wind turbine is typically in the range 10-20 revolutions per minute resulting to a 1P excitation frequency in the range 0.17-0.33 Hz and a 3P frequency between 0.5-1 Hz.

Today, offshore wind turbines are designed with the 1st natural frequency, f_1, in the range between 1P and 3P; in the wind industry sector this is referred to as a soft-stiff structure. However, it is possible to design a soft-soft structure with the 1st natural frequency below 1P, or a stiff-stiff structure with the 1st natural frequency above 3P. The choice of frequency range for f_1 sets criteria for the stiffness of the foundation (support structure); in general, less steel is required for a soft structure. Also, a softer structure requires a smaller diameter of the body which reduces the hydrodynamic loads. However, issues of fatigue or ultimate capacity may become dominant. On the other hand, a structure lying in stiff-stiff region is generally more economically expensive as it requires more material to be used.

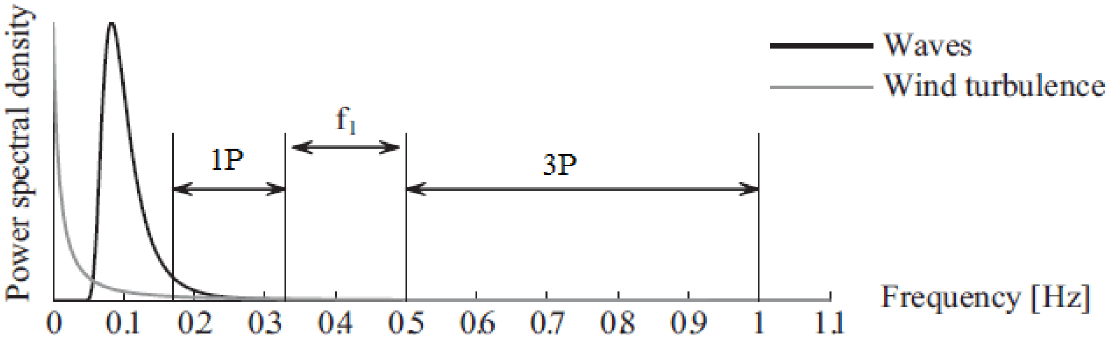


Figure 2.12: Typical excitation ranges of offshore wind turbines

2.6.2 Design considerations

Although design guidelines are available for offshore oil and gas installation foundations, their direct extrapolation/ interpolation to offshore wind turbine foundation design is not always possible. There are two reasons: (a) The foundations of these structures are moment resisting, that is, large overturning moments at the foundation which are disproportionally higher than the vertical load; (b) The structure is dynamically sensitive and therefore fatigue is a design driver. The design of offshore wind turbine support structures needs to consider the following six characteristic issues [20]:

- *Nonlinearities* – Wind turbines exhibit non-linear effects and time-history dependence (wake development, dynamic wind inflow, nonlinear wave loads, structural nonlinearities e.g., foundation when using the general p-y method for piled foundations. Because of these effects, current standards prescribe that wind turbine analysis is performed in the time-domain (IEC, 2005; IEC, 2009; DNV, 2013).

- *Complex environment* – Wind turbines are subject to complex, irregular and highly fluctuating environmental conditions. The wind is turbulent (gusts also appear) and the wave loads and currents, depending on the site conditions, consist of both viscous and inertial effects. Additional effects due to scour, marine growth and sea ice might further complicate the loading. The irregular nature of the environment makes it necessary to simulate relatively long time-intervals in order to obtain sufficiently accurate results. Current standards for offshore wind turbines recommend at least 60 min of simulation time per load case, i.e., 6x10 min or 1x60 min in IEC (2009) to ensure sampling enough variability from the stationary loading process. In offshore engineering, it has been common practice for many years to consider simulations of at least 3 hours.

- *Fatigue as design driver* – Wind turbine structures are exposed to quasi-periodic excitations due to oscillatory wind and wave actions Wind turbine structures are therefore prone to failure from fatigue damage. Fatigue is a failure mechanism that causes structures to fail after repeated loading, even though the loading itself is well below the ultimate strength of the structure. The failure is caused by microscopic cracks that form and grow for each loading cycle. For a lattice structure, the fatigue cracking will almost always happen in the joints first. This is due to the higher stress concentrations in these areas [21]. A jacket-like structure needs some additional consideration. Consisting of a large number of relatively slender members with joints connecting them, these members can be excited locally, in additional to global vibrations of the tower, giving additional contributions to the fatigue loading. The result of all these is a structure with the fatigue lifetime as the main dimensioning factor.

- *Specialized analysis software* – For jackets and other more complex offshore support structures general finite element capabilities are needed.

- *Tightly coupled systems* – Wind turbines consist of many different parts and subsystems. From an optimization point of view, the most computationally efficient manner would be to optimize each part separately in a modular approach. This is simply not possible if, at the same time, high accuracy is sought. Optimizing individual parts separately will, in general, lead to suboptimal or even potentially infeasible solutions.

- *Many design variables and constraints* – Multi-member support structure concepts (e.g., jacket structures) are characterized by many parameters and design variables. Even if the model is symmetric (such that, e.g., each bay in a jacket has the same type of legs or braces, with differences only between distinct bays), parameters typically number into the hundreds. This is normally not an issue for modern optimization algorithms that can deal with millions of parameters, but when the analysis is highly involved and time-consuming, this becomes a limiting factor. Moreover, since structural optimization problems are highly constrained and nonconvex, the objective function is usually multi-modal, i.e., exhibiting many local minima. It is therefore not straightforward to find an optimal solution.

2.6.3 Optimization of wind support structures

The criterion for choosing the optimum design among many possible design solutions must be a quantity representative and characteristic for any design. The total structural cost can be and it is a suitable such quantity since it includes the material, the level of difficulty to implement the considered design and the time needed to build the structure.

However, the estimation of the total structural cost is an extremely complex process with an erratic result as it fluctuates according to the contemporary values of several economic parameters (steel price, cost of transportation, location of manufacturing, etc.). For this reason, in most practical engineering optimization problems, the weight (directly linked to mass of the structure) is used as the objective function-criterion for the design instead of the total cost. Despite the fact that mass (or volume) is a representative quantity of the design, we must be aware that the minimization of it does not always lead to a more efficient design with respect to the total cost. This is more obvious in structures using standardized (i.e. with certain discrete dimensions) structural members like steel jacket frames.

In real-life engineering problems, minimum weight (or mass or volume) does not necessarily mean minimum cost and the reason for this is that the estimation of the structural cost takes into account various parameters, not explicitly obvious. Maybe the most important of these parameters are the lack of *commonality*, the cost of the *welding* and the quantity of the remaining material (*scrap*) [7].

The term "commonality" describes how similar or different the profiles of a structure are. For instance, a truss consists of a number of bar elements, each one of which, theoretically speaking, may have a different profile. However, while this may be good from the viewpoint of the total structural weight, it is not good from the viewpoint of the total structural cost, because it is almost always cheaper to buy larger amounts of the same profile than to purchase smaller amounts of different profiles. Therefore, it is of interest to increase the commonality of a structure, so that fewer different profiles can be ordered.

The cost of the welding is another parameter that may significantly affect the design of a structure. For instance, in order to minimize the weight of a closed cross-section, as is the case of a box girder, it is possible to suggest a design with longitudinal reinforcements. However, these reinforcements must be welded to the cross-section, thus the structural cost increases due to the welding, which is expensive. Instead, it would be possible to increase the thickness of the cross-section plates such that no longitudinal reinforcement is required. In the latter case, while the structural weight is higher, the structural cost may be lower, because there is a break-even-point where the increase of the structural cost due to purchasing plates is less than the increase of the structural cost due to welding. As a result, it is possible to adopt a whole new design trend which, although heavier than the one proposed by weight minimization procedures, is preferable in terms of structural cost.

Another parameter that determines the structural cost is the scrap material; that is the material that remains after the construction is completed. Having zero scrap is the ideal case and for this to occur the optimization procedure to be applied must be case-oriented; in this way, the design characteristics and particularities of the structure under examination can be exploited as much as possible. According to the ideal case of 'zero-scrap design', the structure consists of an integer number of commercially available structural members, such as sheet plates or beams.

The typical optimization problem for a wind support structure is to minimize the structural cost by varying parameters that define its geometry, i.e., diameters and thicknesses of structural elements. Alternatively, objectives for wind support structure optimization can be to maximize the stiffness of a structure, to maximize the structural eigenfrequencies or to maximize the stiffness/mass ratio. The objective function quantifies the degree by which the structure succeeds in fulfilling these objectives by a single numerical value (e.g., the total structural cost or the total weight of the structure). This value depends in a fixed and predetermined way on the geometric parameters that describe the structure. Additional equations either describe fixed relationships between these parameters and other variables of the problem (equality or inequality constraints) or describe side constraints that restrict the range of parameters or solution variables within certain bounds. For example, diameters of steel members have both a minimum and maximum size due to manufacturing constraints.

The rotor of a wind turbine is subject to aero-elastic effects that result in nonlinear, time-history dependent forces acting on the supporting structure. Characteristics such us the *1p* and *3p* frequencies are critical in wind turbine support structure design.

3 PROBLEM FORMULATION

3.1 Problem description

The objective of this thesis is to find an efficient way to perform a sizing (i.e. size optimization) on the structural members of a jacket support structure for an offshore wind turbine. During the size optimization, the cross sectional properties (outer diameter and thickness) of all members are modified while no elements are added or removed. This implies that the topology of the design is kept the same as for the reference jacket, throughout the optimization process.

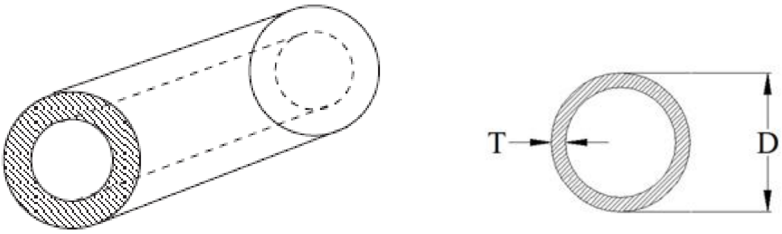


Figure 3.1: Sizing optimization variables

The design of an offshore wind turbine support structure is a non-linear dynamic response constrained structural optimization problem. Initially, an integrated model of the structure (wind turbine + tower + support structure) was built in FEDEM Windpower version R7.1.2 software. Then a dynamic analysis was performed using the same software tool. The rotor is subject to aerodynamic loads (i.e. aerodynamic damping and excitation forces) while the submerged parts are subjected to hydrodynamic forces (i.e. hydrodynamic damping and excitation forces). The whole system is modelled as being rigidly clamped at the seabed. During the analysis, the finite element method is employed to solve the overall system response. Subsequently, internal forces or stresses are recovered from the support structure. Note here, that the member forces and hence stresses, are dependent on the design of the structure, i.e. on the member cross sectional areas.

The dynamic analysis is followed by an iterative optimization procedure that was developed in Matlab. Generally, the objective function f to be minimized is the cost of the support structure. The cost of the structure was chosen here instead of the weight in order to make a more accurate analysis during the optimization process and to capture possible relations between the variables that decrease the cost further which normally cannot be considered when weight of the structure is the objective (e.g. the cost of welding maybe smaller for a member with smaller thickness but with same cross sectional area as another one. In that case the cost objective function will lead the design in the choice of the first member whereas the optimization with weight objective function could choose the second one). The constraints considered in the optimization process were based on limit states functions prescribed by the design standards used within the offshore and wind industries [22, 23]. They can be classified into:

- *Sizing constraints* which define the lower and upper bounds of the design variables as well as the geometrical relationships among the variables, respectively.

- *Extreme load constraints* which are based on the ultimate limit state (ULS) analysis performed on tubular members.

- *Fatigue load constraints* which is based on the fatigue limit state (FLS) analysis performed on tubular joints to confirm that a minimum survivability of 20 years of service life is attained under the design load case.

3.2 System model

3.2.1 Support structure

The structural model used for the application of the local optimization is based on the OC4 reference jacket [3] originally designed by Ramboll AS for the Upwind project [24]. The jacket support structure used here is located at the UpWind deep reference site in 50m water depth [24]. The four-legged jacket was a symmetrical design comprised of four levels (bays) of X-braces, a bottom mud brace at each side connecting the legs and four central piles with a penetration depth of 45 m. Piles are grouted to the jacket legs.

The connection to the tubular tower was modeled as a rigid transition piece (TP) made out of concrete that is penetrated by the upper parts of the four jacket legs. The total height of the jacket from mudline including the TP and excluding the tower is 70.15 m.

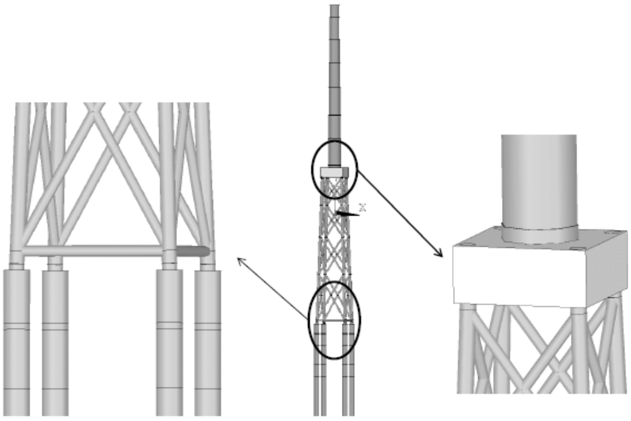


Figure 3.2: Jacket with tower and piles (middle), concrete TP (right) and pile heads in detail (left) [3].

The model has been used and verified in earlier studies [25]. The jacket was assumed to be rigidly clamped on the seabed.

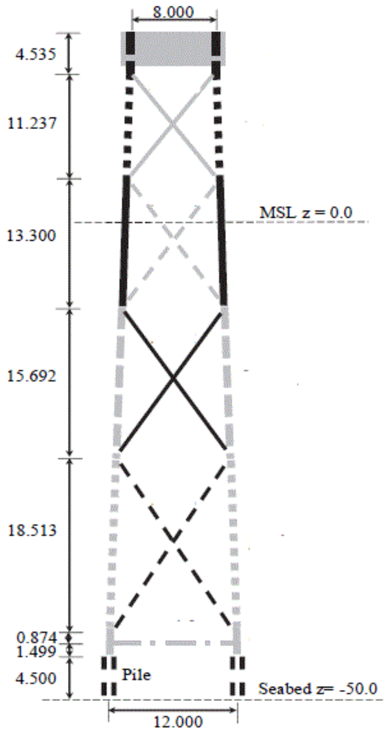


Figure 3.3: The OC4 jacket substructure [26]

3.2.2 Wind turbine

On top of the jacket a conical tower of the horizontal axis three-bladed, NREL 5MW baseline turbine defined by [4] is placed. The conical tower has a total length of 68m leading to a realistic hub height over the mean sea level (MSL) of 90.55 m.

Table 3.1: Gross properties for the NREL 5MW Baseline Wind Turbine

Turbine parameter	*Value*	*Unit*
Rating	5	[MW]
Rotor orientation	Upwind	
Configuration	3 Blades	
Control	Variable speed, Collective Pitch	
Drivetrain	High Speed, Multi-Stage Gearbox	
Rotor diameter	126	[m]
Hub diameter	3	[m]
Hub height	90.55	[m]
Cut-in/Rated/Cut-out wind speed	3/11.4/25	[m/s]
Lower/Nominal/Upper rotor speed	6.9/12.1/12.1	[rpm]
Rated tip speed	80	[m/s]
Length/Width/Height of nacelle	19/6/7	[m]
Rotor mass	110.000	[kg]
Nacelle mass	240.000	[kg]
Tower mass	347.460	[kg]

3.2.3 Material properties

The material assumed to be used for the components of this case study is structural steel. The steel properties are as follows for the whole structural model:

Table 3.2: Material parameters used in the definition of the tubular members

Symbol	*Notation*	*Value*	*Unit*
ρ_s	Material density	7850	[kg/m^3]
E_s	Young' s modulus (Elastic modulus)	2.1*10^{11}	[N/m^2]
v_s	Poisson' s ratio	0.29	[−]
σ_{yield}	Yield tensile strength	250*10^6	[N/m^2]

3.2.4 Cost factors

The costs factors for the raw materials (steel) and for the labor used in the current study are given in the following table:

Table 3.3: Cost factors used in the definition of the optimization criterion

Symbol	*Notation*	*Value*	*Unit*
k_m	Material cost factor	0.60	[€/kg]
k_f	Fabrication cost factor	0.57	[€/min]

For the material factor, [27] provides a value of 0.536 euro/kg as of May 2015, [28] gives 0.27 euro/kg in August 2016, [29] provides weekly updated values and for August 18th the steel price was 0.312 euro/kg. On the other hand, [30] indicates values between 0.5 and 1.5 euro/kg. Considering the above, we can say that at the moment of writing of this study the steel price is relatively low (lower than average) and trying to be close to current trends, here a value of 0.6 euro/kg for the raw steel was used.

Considering the fabrication cost factor, the latest available data from Eurostat were used [31]. According to these, the total labour costs for the Netherlands for the year 2015, were 34.1 euro/hr = 0.57euro/min. This value was used here and covers wages and salaries as well as nonwage costs (employer's social contributions plus taxes less subsidies).

3.2.5 Environmental loads

Wind and hydrodynamic loads were considered in the simulations. The wind conditions were modeled as three-dimensional turbulent wind fields according to the Von Kaimal spectral model. The wave conditions were modeled as wheeler stretched irregular waves according to the JONSWAP wave spectrum. The load cases considered were:

Table 3.4: Load cases considered during the analysis [32]

DLC	*Description*	*Type*	V_{hub} *[m/s]*	*TI* *[%]*	*Hs* *[m]*	*Tp* *[m/s]*	γ *[–]*
1.2	Power production	FLS	9	15.6	1.395	5.705	1.0
	Power production	FLS	14	14.2	1.910	6.070	1.0
	Power production	FLS	19	13.5	2.615	6.850	1.0
5.1	Emergency shut down	ULS	14	14.2	1.910	6.070	1.0
6.1	Parked under extreme conditions	ULS	42	12.0	9.400	10.870	3.3

where V_{hub} is the wind speed at the hub height, TI is the turbulence intensity of the wind field, H_s and T_p are the significant wave height and the peak spectral period of the waves respectively and γ is the peak enhancement parameter for JONSWAP spectrum.

3.2.6 Analysis procedures and tools

The structural model was analyzed in a time domain simulation with the use of the FEDEM Windpower software. This is a finite element method analysis tool that allows for a dynamic analysis in the time-domain. Structural responses (axial force and in-plane and out-of-plane bending moment) to wind and wave loads were calculated at each time step and for each member. Simulation runs with a total length of 300 s were performed. The first 90sec were not included in further analysis in order to exclude transients and avoid noise in the calculations later. Therefore, the actual simulation length was 210 s.

The time series were then converted to nominal stresses using the members' cross sectional areas and moment of inertias. Resulting stress time series were further processed by the superposition of stresses in tubular joints, including the impact of stress concentration factors (SCF). Then the calculated stress time series were investigated by FLS, ULS and buckling analysis. Subsequently, a local optimization of the structural members was performed with an in-house Matlab code. Each sizing step is based on time series results from the last time domain finite element simulation that was performed. When the design is completely optimized for a given load history, new simulations are initiated with the optimized design. This produces a new load history, which is used to optimize the design again.

3.3 Design variables

In the structural analysis of truss structures the cross sectional area is required as a parameter of the problem. This parameter is defined by the diameter and the thickness of the considered structural member. Therefore, the design variables selected for this study were the *outer diameters* and the *wall thicknesses* of the tubular jacket members. They were distinct for various bays and member types, either legs or braces. In addition, the bottom-most sections of the legs were the pile members that protruded out from the soil. These parts were not included in the optimization study. Due to symmetry, the four side planes are identical and at each side there is also symmetry along the middle vertical line. Since beam elements of legs and braces can be described by the cross section at their ends i.e. at joints, the design variables are the outer diameters and the thicknesses of each of three cross section types. The design variables are independent from bay to bay. The first type is the chord at the K-joints, called *K-leg* here. Ten variables (5xD & 5xT) belong to this category and represent the legs of the jacket at five different levels, at the bottom (x1), between bays (x3) and at the top (x1). The cross sections of braces close to the legs (*K-braces*) belong to the second type of variables and are considered independent from bay to bay and from upper and lower areas within a bay. Therefore, there are 16 variables (8xD & 8xT) belonging to this category. At the last type belong the cross sections of the braces at the X-joint (*X-braces*) side. Since all the braces connected at an X-joint assumed to have the same dimensions, there are 8 variables (one D and one T per bay/per X-joint) of this type. Equation (3.2) shows how the design variable vector is constructed for our problem and at Figure 3.4 it is given how these variables are arranged over the jacket structure.

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ . \\ x_{34} \end{bmatrix} = \begin{bmatrix} D_1 \\ T_1 \\ D_2 \\ . \\ . \\ . \\ T_{17} \end{bmatrix} \begin{array}{l} \left.\begin{array}{c} \\ \\ \end{array}\right\} 8\ \ elements\ \ K-braces\ \ (lower+upper\ \ per\ \ bay) \\ \left.\begin{array}{c} \\ \\ \\ \end{array}\right\} 4\ \ elements\ \ X-braces\ \ (one\ \ per\ \ bay) \\ \left.\begin{array}{c} \\ \\ \end{array}\right\} 5\ \ elements\ \ K-legs\ \ (bottom+between\ \ bays+top) \end{array} \tag{3.2}$$

Table 3.5: Design variables

Category	*Cross section type*	*Design variables*
1	Leg at K-joint	10
2	Brace at K-joint	16
3	Brace at X-joint	8
	Total	34

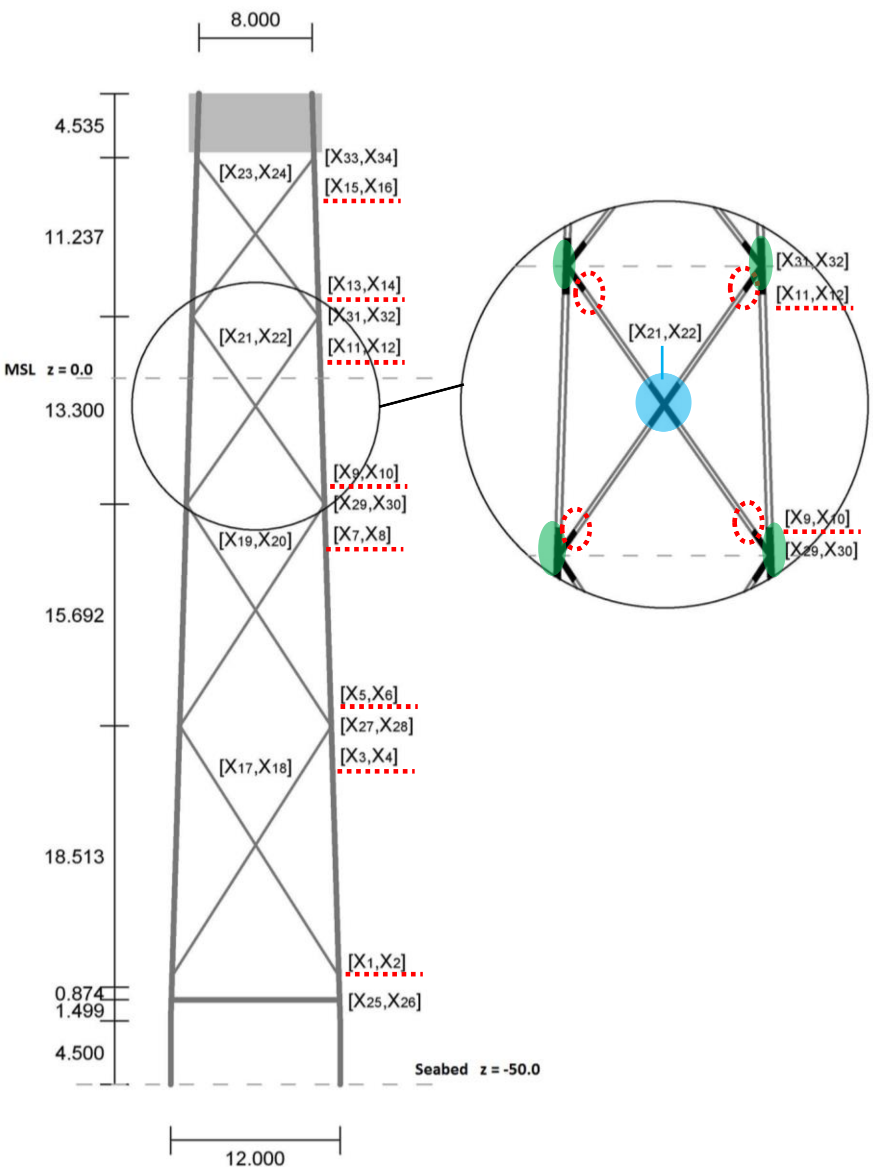


Figure 3.4: Design variables and their locations

In the configuration of the structural model in this study, the leg and brace dimensions (middle sections between two joints) are dependent on the variable cross sections and are defined by the use of a reduction factor *Dred,* as follows:

Legs: $X_{\text{leg}} = \text{Dred}{*}X_{\text{K-leg,,}}$

where K-leg is the attached cross section from the upper bay, i.e. the leg dimensions at bay 2 are dependent on the dimensions of the K-leg section between bays 2 and 3.

Braces: $X_{\text{brace}} = \text{Dred}{*}X_{\text{X-brace,,}}$

where X-brace is the attached cross section at the X-joint side, i.e. the brace dimensions at bay 2 are dependent on the X-brace dimensions of bay 2.

Jacket models for two different values of the reduction factor were tested, Dred=0.6 and Dred=1.0 (for more details see the results chapter, 5).

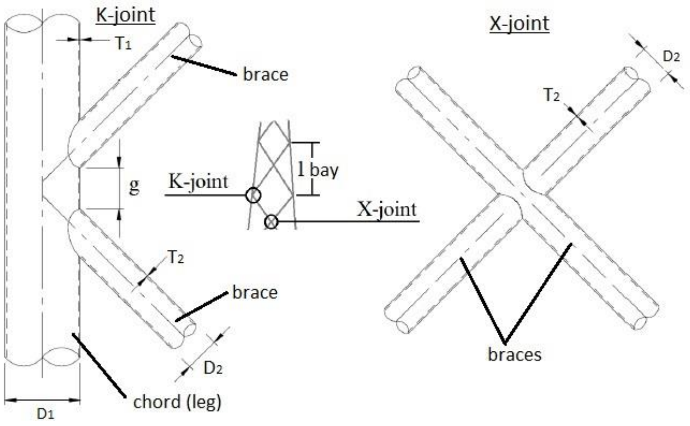


Figure 3.5: A K-joint and an X-joint

3.4 Optimization criterion

The criterion must be a scalar function whose numerical value can be obtained once a design is specified; that is, it must be a function of the design variable vector x. Such a criterion is called an objective function for the optimum design problem and when it is to be minimized it is usually called a cost function. A valid objective function must be influenced directly or indirectly by the variables of the design problem.

The objective function to be minimized in this study is the cost of the support structure [€]. The cost of a structure is the sum of the material, fabrication, transportation, erection and maintenance costs. In this thesis, only the material and the fabrication costs are considered and all the other costs are ignored. The cost function can be expressed as

$$f(x) = K(x) = K_m(x) + K_f(x) \tag{3.1}$$

where K_m and K_f are the total material and fabrication costs respectively.

3.4.1 Material cost

It is assumed that k_m is constant for different size hollow sections. Since steel has a certain price per kilogram, the material cost can be calculated simply by multiplying the mass of the structure by the current material cost factor. The support structure is modelled using simple, circular beam elements. The volume of material of a single tubular member is simply the beam area A, multiplied by the total length of the member. The total material cost for the jacket structure can be expressed as

$$K_m = k_m \rho \sum_{n=1}^{N} A_n L_n \tag{3.2}$$

where k_m is a material cost factor [euro/kg], ρ is the material density [kg/m^3], N is the number of elements in the structure, A_n is the cross sectional area [m^2] and L_n the length [m] of the member n.

3.4.2 Fabrication costs

In the calculation of the fabrication cost the common approach is to divide the manufacturing process into several production stages and to consider the cost of each stage. The fabrication cost elements then are the welding, cutting, preparation, assembly, tacking, painting, etc. Times of the fabrication phases will be used as the basis to estimate the fabrication cost. The fabrication times depend on the technological level of the country and the manufacturer (technical level of a machine workshop, the level of labor cost and the volume of production). After the necessary time is computed for a fabrication work element, it is multiplied by a specific factor which represents the development level differences [33]. The fabrication cost function can be expressed as

$$K_f = k_f \sum_{n=1}^{N} \sum_{i=1}^{6} t_{n,i} \tag{3.3}$$

where k_f is a fabrication cost factor [euro/min] and t_i the times needed for the different fabrication stages. The fabrication times considered in this work are the times for forming shells, hand cutting of ends, assembly, welding, deslagging, painting, respectively. The expressions for the fabrication times are given in Table 3.6 according to [34].

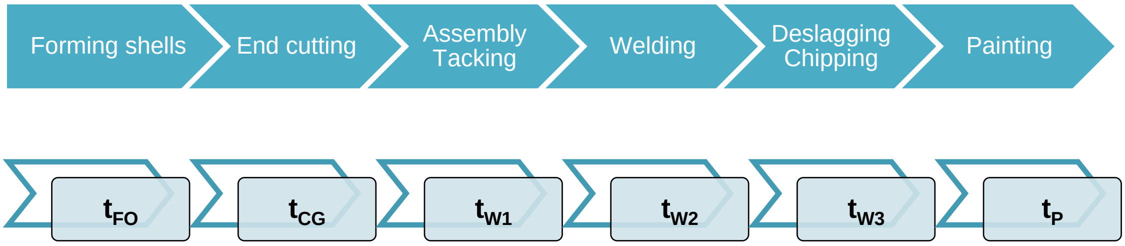


Table 3.6: Fabrication times[1]

Activity	*Expression*	
Forming plates into shell elements	$t_1 = t_{FO} = \Theta_{df} e^{\mu}$	(3.4)
Hand cutting and machine grinding of strut ends (tubular structures)	$t_2 = t_{CG} = \Theta_{dc} \left(4.54 + 0.4229T^2\right) L_c$	(3.5)
Preparation, assembly, tacking	$t_3 = t_{W1} = C_1 \Theta_{dw} \sqrt{k\rho AL}$	(3.6)
Real welding time	$t_4 = t_{W2} = C_2 a_w^n L_w$	(3.7)
Changing electrode, deslagging, chipping	$t_5 = t_{W3} = 0.3t_{W2}$	(3.8)
Painting	$t_6 = t_P = \Theta_{dp} \left(a_{gc} + a_{tc}\right) \pi A_s$	(3.9)

where Θ_i are difficulty factors for each specific task determined by the type of the structure subject to the task (e.g. tubular member, jacket structure etc.), μ is an expression related to the geometry of the shell members, T is the thickness of the cross sectional area, L, L_c, L_w are the member length, the cut length of the ends of a member, the welding length, respectively, C_i are welding technology factors, k is the number of elements to be assembled, A, A_s are the cross sectional area and the area to be painted respectively, α_w is the welding size (width), n is a constant related to the type of the welding connection and α_{gc}, α_{tc} are parameters related to painting technology. Note that the dimensions (diameters, thicknesses, lengths) in all the expressions are in [mm] and the times in [min]. Analytic expressions as well as values of the parameters are given in Appendix A.

All fabrication times are functions of the design variables (D_n, T_n) of each tubular member n. Considering the above, the objective function in the Equation (3.1) can be written:

$$f = \sum_{n=1}^{N} \left[k_m \rho A_n L_n + k_f \left(t_{FO,n} + t_{CG,n} + t_{W1,n} + t_{W2,n} + t_{W3n} + t_{P,n} \right) \right] \tag{3.10}$$

More specifically, Eq. (3.10) with respect to the variables X is expanded as:

$$\begin{aligned} f(X) = {} & k_m \sum_{n=1}^{N} \rho A_n(X) L_n + k_f \sum_{n=1}^{N} \left[\Theta_{df} e^{\mu(X)} + \Theta_{dc} \left(4.54 + 0.4229T_n^2\right) L_c(X) + \right. \\ & \left. + C_1 \Theta_{dw} \sqrt{k\rho A_n(X) L_n} + 1.3C_2 a_w^n L_w(X) + \Theta_{dp} \left(a_{gc} + a_{tc}\right) \pi A_s(X) \right] \end{aligned} \tag{3.11}$$

[1] NOTE: Times calculated in [min] and dimensions (D, T, L) are given in [mm]. However, there is an exception in Equation (3.7) where the thickness is in [mm] but the cut length must be given in [m].

For the present study, the fabrication parameters in Table 3.7 were used. These parameters depend on the type of the subject structure (e.g. jacket), on the type of the single members (e.g. tubular), on the welding technology used and other geometrical factors. In Appendix A, more details on the welding technologies and on fabrication times in general can be found.

Table 3.7: Values of fabrication parameters used in the present work [30] [33]

Symbol	*Value*	*Explanation*	*Symbol*	*Value*	*Explanation*
Θ_{df}	3	Tubular member	C_2	$0.1346*10^{-3}$	SAW technology
Θ_{dc}	3	Tubular member	n	2	V butt weld
C_1	1	Welding technology factor	Θ_{dp}	3	Complicated structure (jacket)
Θ_{dw}	3	2 or 3 for tubular members	a_{gc}	$3.0*10^{-6}$	const.
k	2	Number of elements to be assembled	a_{tc}	$4.15*10^{-6}$	const.

3.5 Formulation of constraints

3.5.1 Sizing constraints

Sizing constraints define the lower and upper bounds of the continuous design variables as well as the (linear) relationships between them. For the bounds, NORSOK N-004 proposes T ≤ 6mm. Also, due to rolling difficulties during the manufacturing process the thickness of the elements cannot exceed certain limits (>80mm). In addition, despite the recent records of cylindrical structures (e.g. monopiles) constructed with diameters larger than 8m, an optimized design should try to adopt members already commercially available. Such tubular members (CHS profiles, steel piles, etc.) have maximum diameters of the order of 2m. Considering the above and for benchmarking reasons with previous studies (e.g. [2]), the following bounds were set to the design variables D_{min}=0.4m, D_{max}=1.6m, T_{min}=0.020m, and T_{max}=0.060m.

$$\begin{gathered} D_{\min} \leq D \leq D_{\max} \\ T_{\min} \leq T \leq T_{\max} \end{gathered} \tag{3.12}$$

Standards for the strength of tubular members typically have limitations with respect to the D/t ratio. ISO19902 proposes $D/t \leq 120$, NORSOK N-004 proposes $D/t \leq 120$ (ULS) as well and $20 \leq D/t \leq 120$ (FLS), DNV-RP-C203 proposes $16 \leq D/t \leq 64$ (FLS). Here we consider all the above limitations by imposing the following constraints:

$$20 \leq D/T \leq 64 \Rightarrow \begin{cases} g_1 = 20T - D \leq 0 \\ g_2 = D - 64T \leq 0 \end{cases} \tag{3.13}$$

The intuitive geometric requirement for the outer diameter needed to be larger or equal to two times the thickness ($2T - D \leq 0$), is already covered by the imposed constraints in Eq. (3.13).

Again according to DNV [22] standard for the fatigue calculations, the equations used for calculating the fatigue damage at tubular joints are valid only within some ranges for the parameters β , τ , α :

$$\beta = \frac{D_{brace}}{D_{chord}} \quad , \quad \tau = \frac{T_{brace}}{T_{chord}} \quad , \quad \alpha = \frac{2L_{chord}}{D_{chord}} \tag{3.14}$$

where L_{chord} is the length of the chord at a tubular joint, D_{brace}, D_{chord} are the diameters of the brace and the chord respectively and T_{brace}, T_{chord} are the thicknesses of the brace and the chord respectively. The above equations are limited and impose some additional linear constraints to the variables of our problem as:

$$0.2 \leq \beta \leq 1 \Rightarrow \begin{cases} g_3 = -D_{chord} + D_{brace} \leq 0 \\ g_4 = 0.2D_{chord} - D_{brace} \leq 0 \end{cases} \tag{3.15}$$

$$0.2 \leq \tau \leq 1 \Rightarrow \begin{cases} g_5 = -T_{chord} + T_{brace} \leq 0 \\ g_6 = 0.2T_{chord} - T_{brace} \leq 0 \end{cases} \tag{3.16}$$

$$4 \leq \alpha \leq 40 \Rightarrow \begin{cases} g_7 = -20D + L_{chord} \leq 0 \\ g_8 = 2D - L_{chord} \leq 0 \end{cases} \tag{3.17}$$

Furthermore, as the configuration of the structure allows different cross sections at the braces and at the connection of the braces to the legs (i.e. stubs), there must be set another limitation. After a lot of testing, it was found out that the stubs should have at least equal dimensions or larger than the braces attached to them at any step during the optimization process. If stubs become smaller, then the local phenomena take place which lead to high bending moments (M_y, M_z) at these locations. As a result, during optimization the dimensions of both stubs and braces will have increased dimensions but is it is still possible that high bending moments will be present (and therefore constraints may not be satisfied) since the important feature in this case is the ratio of the dimensions between the stubs and the braces attached to them. It was found after tests that this ratio(X_{stub}/X_{brace}) should not become lower than 0.8. In this case, convergence of the optimization process may not be achieved or if it is achieved it will produce inefficient designs with extraordinary large dimensions. For the above reasons the following geometrical constraints are more than necessary in order to avoid the above described phenomenon.

$$0.8 \le D_{stub} / D_{brace} \Rightarrow g_9 = 0.8D_{brace} - D_{stub} \le 0 \tag{3.18}$$

$$0.8 \le T_{stub} / T_{brace} \Rightarrow g_{10} = 0.8T_{brace} - T_{stub} \le 0 \tag{3.19}$$

3.5.2 Extreme load constraints

Ultimate limit state (ULS) analysis is performed on the offshore structure to evaluate if the tubular members and joints satisfy structural strength and stability requirements, i.e. yield check and buckling assessment. Under the complicated environmental conditions, the offshore tubular members are potentially subject to any combination of axial tension, axial compression, bending, shear and hydrostatic loads. The standards like [23] distinguish the members which are free from hydrostatic pressure, for instance the beams situated above the mean sea level (MSL) or flooded internally and submerged members which experience hydrostatic pressure externally. Herein we will consider only a reduced set of checks/constraints which corresponds to members free of hydrostatic pressure and consists of the following two constraints:

1) *Yielding*. The maximum absolute stresses from combined action of axial force and bending moments derived after the time simulation analysis are restricted to the yield strength of structural steel (=250MPa).

$$\frac{\sigma_{Sd}}{\sigma_{Rd}} \le 1 \Rightarrow \frac{N_t}{N_{t,Rd}} + \frac{M_y}{M_{Rd}} + \frac{M_z}{M_{Rd}} \le 1 \Rightarrow \frac{N_t}{A\sigma_{yield}} + \frac{M_y + M_z}{W\sigma_{yield}} - 1 \le 0 \tag{3.20a}$$

or

$$g_{11}(X) = \frac{4\mathrm{N}_t}{\pi\left(DT - T^2\right)\sigma_{yield}} + \frac{32D\left(M_y + M_z\right)}{\pi\left(D^4 - (D-2T)^4\right)\sigma_{yield}} - 1 \tag{3.20b}$$

Where σ_{Sd} , σ_{Rd} , σ_{yield} are the design stress [MPa], the stress design resistance [MPa] and the yield strength of steel [MPa] respectively, A is the cross sectional area, $W = 2I / D$ is the elastic section modulus [m³], I is the area moment of inertia [m⁴], N_t, M_y, M_z are the design tensile force [N], in-plane design bending moment [Nm] and out of-plane design bending moment [Nm] respectively; $N_{t,Rd}$, M_{Rd} are design resistances for axial tension [N] and bending moment [Nm] respectively.

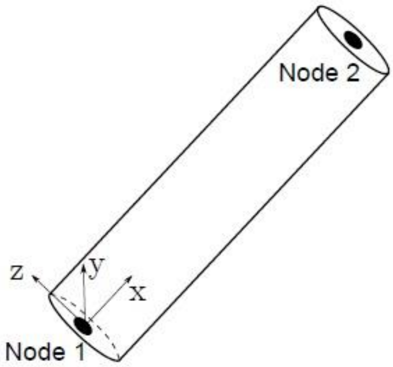


Figure 3.6: Local coordinate system for a tubular element of the jacket structure

2) *Column buckling (Euler).* The loading of long and slender members in lattice structures might, due to insufficient stability, result in a buckling failure of the structure. Therefore, a buckling constraint was imposed for each structural design to avoid optimized members having critical dimensions in terms of buckling. The buckling safety check was performed on a general basis using the formula derived by Euler [35].

$$\frac{N_c}{N_E} \le 1 \tag{3.21a}$$

or

$$g_{12}(X) = \frac{N_c}{\frac{n\pi^2 EI}{L^2}} - 1 = \frac{64L^2 N_c}{n\pi^3 E\left(D^4 - (D-2T)^4\right)} - 1 \tag{3.21b}$$

where N_c is the design compressive force [N] and N_E is the Euler buckling (either member y- or z- axes) [N].

Since the extreme load constraints are time-dependent, they are required to be satisfied at all time steps. The *worst case approach* was implemented in this study. This method identifies the maximum violated limit state values in time as design constraints, while gradients are calculated for the constraints at those active time grid points.

3.5.3 Fatigue load constraint

Fatigue is a failure mechanism that causes structures to fail after repeated loading, even though the loading itself is well below the ultimate strength of the structure. The failure is caused by microscopic cracks that form and grow for each loading cycle. For a lattice structure, the fatigue cracking will almost always happen in the joints first. This is due to the higher stress concentrations in these areas, which is the most important parameter for prediction on crack initiation.

It is important to use an adequate cumulative damage theory when determining the fatigue damage in variable amplitude loading (like the environmental loading on an offshore wind support structure). The damage is defined as a fraction of the life of the structure. Typically, fatigue limit state (FLS) analysis is carried out to ensure a minimum survivability of 20 years of service life.

To predict the fatigue life, the damage fractions are summed using an accumulation rule. *Palmgren-Miner' s linear damage rule* is applied in this study. This rule does not take sequential effects and interaction of events into account, even though it can potentially have a large influence on the fatigue life of the structure. However, this is how the current recommended offshore practice [22] addresses fatigue. The material data for fatigue is given by *S-N curves* (or *Wohler diagrams*). An S-N curve represents the number of cycles to fatigue failure in high-cycle regime as a function of the stress range.

FLS analysis is carried out on every tubular joint. Evaluating the fatigue in the cross section in each end of each element will thus approximate the fatigue in the welds where failure is expected to occur. Since the loading conditions are multiaxial, it is very difficult to predict where the highest accumulated damage will occur. For this reason, the damage must be evaluated at many local points for each stress cycle. According to the DNV [22], eight hot spot stresses (HSS) around the circumference of the intersection (joint) of two tubular members, are determined from the superposition of nominal stresses that are pre-multiplied by stress concentration factors (SCF). The SCFs are calculated using empirical formulae, based on the joint configuration.

$$\begin{aligned}
\sigma_{1,5} &= SCF_{AC}\sigma_x \pm SCF_{MIP}\sigma_{my} \\
\sigma_{2,6} &= \frac{1}{2}\left(SCF_{AC} + SCF_{AS}\right)\sigma_x \pm \frac{1}{2}\sqrt{2}SCF_{MIP}\sigma_{my} \mp \frac{1}{2}\sqrt{2}SCF_{MOP}\sigma_{mz} \\
\sigma_{3,7} &= SCF_{AS}\sigma_x \mp SCF_{MOP}\sigma_{mz} \\
\sigma_{4,8} &= \frac{1}{2}\left(SCF_{AC} + SCF_{AS}\right)\sigma_x \mp \frac{1}{2}\sqrt{2}SCF_{MIP}\sigma_{my} \mp \frac{1}{2}\sqrt{2}SCF_{MOP}\sigma_{mz}
\end{aligned} \tag{3.22}$$

where $\sigma_{1\text{-}8}$ are the HSS; σ_x, σ_{my}, σ_{mz} are the maximum nominal stresses due to axial load and in-plane and out-of-plane moment respectively; SCF_{AC}, SCF_{AS}, SCF_{MIP}, SCF_{MOP} are SCFs for axial load at the crown, for axial load at the saddle, for in-plane moment and for out-of-plane moment respectively.

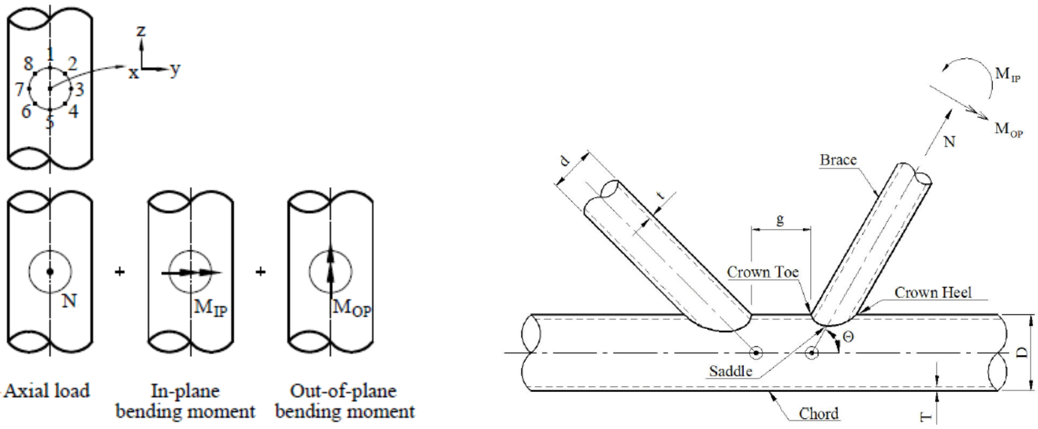


Figure 3.7: Superposition of stresses and geometrical definitions for tubular joints [22]

After which, the *rainflow counting* is performed to determine the effective number of stress cycles and the corresponding stress ranges. Rainflow counting is a technique for counting and analyzing cycles resulting from time domain simulations such that the results can be used for lifetime prediction. The technique utilizes the successive extremes of the loading sequence and is well suited for situations where the amplitude of the loading is varying.

Taking into account the thickness effect, the total local (i.e. for each hot spot location) equivalent (accumulated) damage is estimated using *Palmgren-Miner's rule* as:

$$D = \sum_{i=1}^{n_{block}} \frac{n_i}{N_i} = \frac{1}{\bar{a}\eta} \sum_{i=1}^{n_{block}} n_i \left(\Delta\sigma_{i,HSS} \cdot \left(\frac{T}{T_{ref}} \right)^k \right)^m \leq D_{lifetime} \tag{3.23}$$

where $D_{lifetime} = 1.0$ (when fatigue failure is expected to occur), $\bar{a}$ = intercept of S-N curve with log N axis; m = negative inverse slope of the S-N curve; n_{block} = total number of stress blocks; $\Delta\sigma_{i,HSS}$ = constant stress range block of HSS [MPa]; N_i = predicted number of cycles to failure at constant stress range $\Delta\sigma_{i,HSS}$; n_i = number of stress cycles in stress block *i*; η = usage factor; T , T_{ref} and k are member thickness [m], reference thickness [m] and thickness exponent, respectively.

The unity check of fatigue constraint can now be written as:

$$g_{13} = \frac{1}{\bar{a}\eta} \sum_{i=1}^{n_{block}} n_i \left(\Delta\sigma_{i,HSS} \cdot \left(\frac{T}{T_{ref}} \right)^k \right)^m - 1 \leq 0 \tag{3.24}$$

For practical fatigue design, welded joints are divided into several classes, each with a corresponding design S-N curve. All tubular joints are assumed to be class 'T' and the corresponding parameter values [22] are given in the Table 3.8.

Table 3.8: S-N curve parameters for tubular joints (class 'T')

Parameter		*Value*				*Unit*
$\log(\bar{a})$	(submerged member)	11.764	(for $N \leq 10^6$)	15.606	(for $N > 10^6$)	[–]
	(member in air)	12.164	(for $N \leq 10^7$)	15.606	(for $N > 10^7$)	[–]
m	(submerged member)	3.0	(for $N \leq 10^6$)	5.0	(for $N > 10^6$)	[–]
	(member in air)	3.0	(for $N \leq 10^7$)	5.0	(for $N > 10^7$)	[–]
η			0.500			[–]
T_{ref}			0.032			[m]
k			0.250			[–]

Consider a member at a tubular joint, with thickness 0.04m, located under water surface (e.g. at a jacket offshore) and after a rainflow counting for a HSS location around the connection, we find N$_1$ =5*10^2 corresponding to the stress range (block) $\Delta\sigma_1$ =1*10^5 MPa. Then the fatigue damage contribution from this stress block will be:

$$D_1 = \frac{1}{10^{11.764} \cdot 0.5} \cdot \left(\frac{0.040}{0.032} \right)^{0.25 \cdot 3} \cdot 5 \cdot 10^2 \left(10^5\right)^3 = 0.203555 \tag{3.25}$$

The same calculation will be repeated for every stress block and then all the damages will be summed up to give the total fatigue damage for this HSS location.

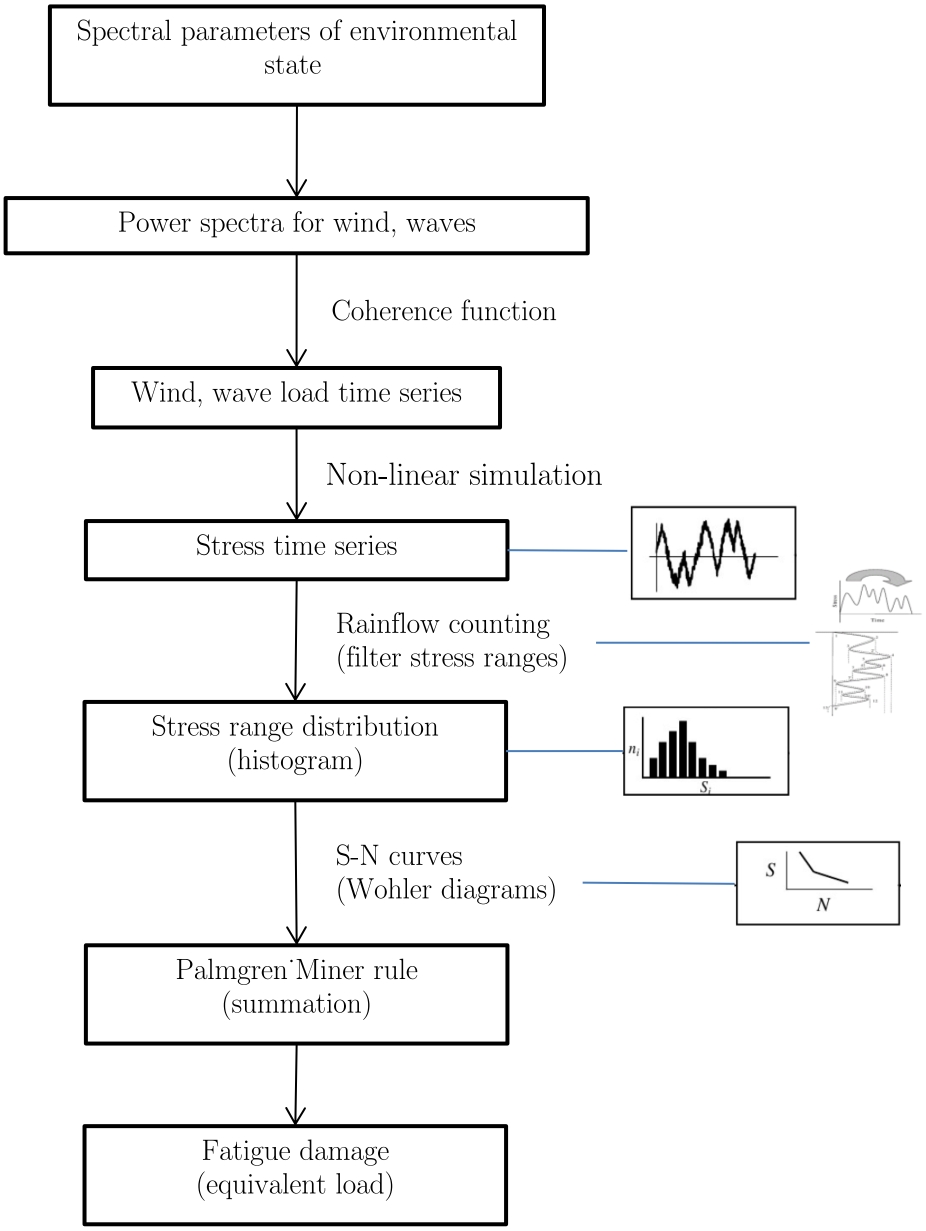

Spectral parameters of environmental state
Power spectra for wind, waves
Coherence function
Wind, wave load time series
Non-linear simulation
Stress time series
Rainflow counting (filter stress ranges)
Stress range distribution (histogram)
n_i
S_i
S-N curves (Wohler diagrams)
S
N
Palmgren˙Miner rule (summation)
Fatigue damage (equivalent load)

Figure 3.8: Fatigue damage calculation in the time domain

3.5.4 Eigenfrequency

In the current study the eigenfrequency is not considered as a constraint during the optimization process. The main purpose of the natural frequency constraint is to avoid high fatigue damage on the support structure and at the same time avoid significant movement of the structure that leads to lower power production. The latter one is not a highly probable problem for our study since jacket support structures are quite stiff. In addition, since the fatigue constraint is already included in the optimization and the fatigue is mostly the design driver for the design of offshore wind support structures [2, 20], the elimination of the natural frequency constraint is considered acceptable and it doesn't affect the performance of the optimization framework proposed here. Despite the eigenfrequency of the structure is not considered as a constraint in this study, for verification of the previous explained theory, the eigenfrequency of all the optimized designs will be checked if it falls within the accepted region. For this reason, we present below the eigenfrequency limitations for the structure used in the current case study.

The rotor frequency range (1P) for the NREL 5MW lies between 0.115 Hz and 0.202 Hz and the corresponding blade passing frequency (3P) lies in the range 0.345 Hz – 0.600 Hz. The total system's (jacket + transition piece + tower + wind turbine) natural frequency is intended to be within the soft-stiff range in between the 1P and 3P frequency ranges. A safety margin of 10% on the maximum and minimum rotor speed is adopted (DNV code suggests that the first natural frequency should not be within 10% of the 1P and 3P ranges), which means that the allowable frequency is between 0.222 Hz and 0.311 Hz. In the figure below also typical frequency distributions for wind and wave loading are presented together with the 1P and 3P ranges.

$$0.222Hz \leq f_1 \leq 0.311Hz \tag{3.26}$$

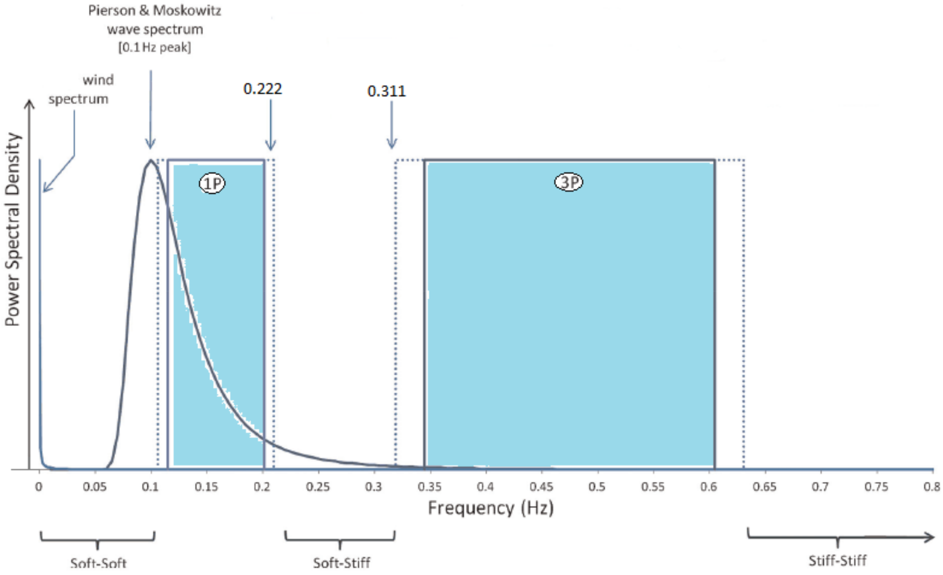


Figure 3.9: Allowable natural frequency range for the support structure [24]

3.6 Overview of optimization problem

The whole optimization problem can be summarized in the table below. For the rest of this study, the nonlinear constraints can also be referred as ULS, FLS or BUC instead of g_{11}, g_{12} and g_{13} respectively.

Table 3.9: Overview of the optimization problem

	Total number	*Description*
Objective function	1	Total cost of the structure, Eq. (3.11)
Design variables	34	K-brace (16), (two sections per bay) X-brace (8), (one section per bay) K-legs (10), (five sections, bottom(1) + between bays(3) +top(1))
Equality constraints	0	No equality constraints included in this problem
Linear inequality const.	84	g_1 (x9), (at K-legs(5) and X-braces(4)) g_2 (x9), (at K-legs(5) and X-braces(4)) g_3 (x8), (at brace to chord at K-joints only) g_4 (x8), (at brace to chord at K-joints only) g_5 (x8), (at brace to chord at K-joints only) g_6 (x8), (at brace to chord at K-joints only) g_7 (x9), (at K-legs(5) and X-braces(4)) g_8 (x9), (at K-legs(5) and X-braces(4)) g_9 (x8), (at each K-brace to X-brace connection) g_{10} (x8), (at each K-brace to X-brace connection)
Nonlinear inequality const.	46	g_{11} (x17), (yielding constraint at every cross section of tubular joints) g_{12} (x12), (Euler buckling constraint at every brace and leg, 3 values per bay, legs are symmetric) g_{13} (x17), (fatigue constraint at every cross section of tubular joints)

4 OPTIMIZATION PROCESS

The most accurate analysis method for wind turbines is the time-domain simulation. The main disadvantage of such an approach with the current simulation times of less than real-time, is that it is extremely time consuming. Especially for more complex support structures, such as jackets or lattice towers, the number of load cases that can be used is normally strongly limited.

The principal complicating factor in structural optimization is that the response quantities of interest are usually implicit functions of the design variables. In any real structural optimization task, each of the data points in the design space can only be determined based on the results of a complete finite element analysis. Also, since a numerical optimizer usually needs a number of these function evaluations throughout the search process, the costs associated with this analysis can quickly become enormous. These factors combined with hundreds of design variables and thousands of constraints make it obvious that we do not have the luxury of invoking a full finite element analysis each time the optimizer proposes an incremental design change. Therefore, we are forced to consider methods for efficiently coupling the structural analysis routines with numerical optimizers.

So, the challenge is how to reduce the problem enough so that only the most relevant information is considered during the optimization process. Some basic concepts used in structural optimization are:

Design Variable Linking: A small set of well-chosen independent variables can be used instead of hundred possible options.

Constraint Deletion. Only a few constraints that guiding the design are considered and all the rest are temporarily disregarded. This leads to a reduced set of constraints and greatly simplifies the optimization process.

Formal Approximations. A parametric analysis (*Design Sensitivity Analysis*) is carried out (e.g. using Taylor series expansions of the objective and constraint functions) in order to determine how the constraints vary as the design is modified.

In an effort to reduce the computational cost for detailed analyses, an efficient optimization approach using the *principle of decomposition* described by Freeman and Newell (1971) is used here. The method is based on the idea of a decomposition of the structure into weakly coupled substructures. Thereby, each member of the complete structure is optimized locally and simultaneously, assuming that:

1. The loads are not changed when changing the member's geometry. This means that given a load history, the optimal geometry of all members can be determined analytically, without the need for further simulations.

2. The loads are not changed when changing the geometry of other members in the structure. This means that all structural members can be sized at the same time, independently of each other.

The above constitute the so-called *local optimization approach.* This approach has been applied for the sizing of a full-height lattice tower in an iterative process earlier [25] [36], and it demonstrated weak coupling between individual members in a lattice tower; each member could be optimized individually. The advantages of this approach are:

- Each section can be optimized locally-internally without the need of running a time domain simulation analysis at every iteration step.
- Few simulations are needed so less computational cost for the optimization process.

4.1 Optimization framework

The structural model was analyzed in a time domain simulation and time series of the responses (forces) were converted to nominal stresses by use of cross sectional areas and moment of inertias. Resulting stress time series were further processed by the superposition of stresses in tubular joints, including the impact of stress concentration factors (SCF). SCF for eight hot spots around the circumference of each member intersection were calculated. These, together with the stress time series were used to determine the hot spot stress (HSS) at the same locations. Then the HSS time series were investigated by fatigue (FLS) analysis, while the stress time series were used for yielding (ULS) and buckling (BUC) analysis.

Subsequently, an optimization of the jacket structure was performed with an in-house code using Matlab scripts as will be described later in this chapter. The code was written as an extension of a program developed by Daniel Zwick and Sebastian Schafhirt during their PhD-studies on the optimization of lattice support structures for offshore wind [2]. Each sizing step is based on time series results from the last time domain finite element simulation that was performed. The assumptions of the local optimization approach used here make it possible to perform the sizing internally without the need to run the external finite element solver for each newly established design. The local sizing is a faster process and requires only small computational resources compared to the analysis with the finite element solver.

When the design is completely optimized for a given load history, new simulations are initiated with the optimized design. This produces a new load history, which is used to optimize the design again. This continues until the design has converged. The design is assumed to have been converged if all the constraints are satisfied and at the same time the relative cost difference between two FEDEM simulations is less than 2%. It is therefore assumed that a potential of 2% reduction of the cost is left.

The flowchart at the next page gives an overview of the optimization approach followed in this study.

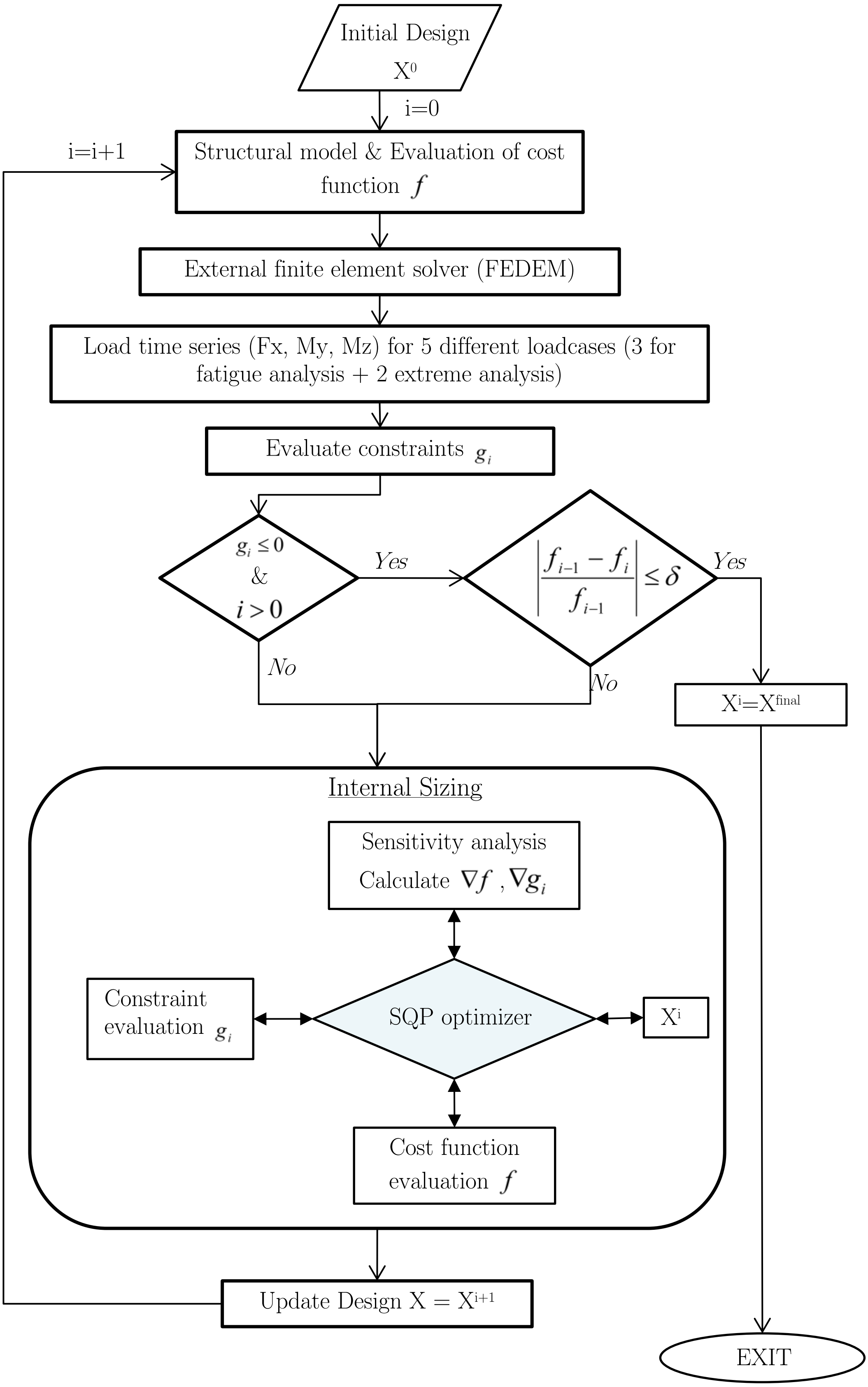

Initial Design
X^0
i=0
i=i+1
Structural model & Evaluation of cost function f
External finite element solver (FEDEM)
Load time series (Fx, My, Mz) for 5 different loadcases (3 for fatigue analysis + 2 extreme analysis)
Evaluate constraints g_i
$g_i \leq 0$ & $i > 0$
Yes
$\left|\frac{f_{i-1} - f_i}{f_{i-1}}\right| \leq \delta$
Yes
No
No
$X^i = X^{final}$
Internal Sizing
Sensitivity analysis
Calculate ∇f, ∇g_i
Constraint evaluation g_i
SQP optimizer
X^i
Cost function evaluation f
Update Design $X = X^{i+1}$
EXIT

Figure 4.1: Optimization flowchart

4.2 Design sensitivity analysis (DSA)

The design sensitivities of the objective function and the constraint functions are defined by the gradient of the function at the considered point (value) at each iteration. Here, the DSA is performed analytically to ensure accurate and fast gradient evaluations. The direct differentiation method was used to derive the sensitivities. The functions were differentiated with respect to the design variables.

The sensitivities here are arranged in gradient vectors which are column vectors consisting of the partial derivatives of the functions (objective and constraints) with respect to the design variables.

The cost and the constraints are functions of the variables (*D, T*) and the area and the second moment of inertia (*A, I*) which in turn are explicit functions of the variables as well. As seen on Figure 4.2 the relationships for these two quantities are nonlinear. Furthermore, a change in either area or second moment of inertia will have a direct influence on the cost and the stresses, since they are an explicit function of the two. From Figure 4.2 it can be seen that the diameter has a larger influence than the thickness for most of the designs. In addition, we notice that as the diameter increases the influence from the thickness increases as well.

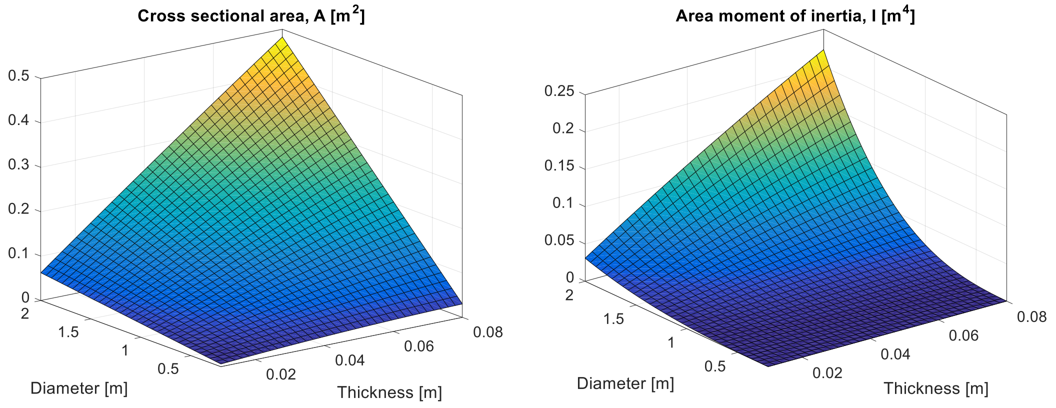


Figure 4.2: Parametric study of the design variables for the area, and the second moment of inertia.

Below, the sensitivities of some essential quantities used to derive the analytic sensitivities objective and constraint functions are provided.

$$\nabla A = \begin{bmatrix} \dfrac{\partial A}{\partial D} \\ \dfrac{\partial A}{\partial T} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial\left(\pi(DT - T^2)\right)}{\partial D} \\ \dfrac{\partial\left(\pi(DT - T^2)\right)}{\partial T} \end{bmatrix} = \begin{bmatrix} \pi T \\ \pi(D - 2T) \end{bmatrix} \tag{4.1}$$

$$\nabla\left(\frac{1}{A}\right)=\begin{bmatrix}\dfrac{\partial(1/A)}{\partial D}\\ \dfrac{\partial(1/A)}{\partial T}\end{bmatrix}=\begin{bmatrix}-\dfrac{1}{\pi(DT-T^2)(D-T)}\\ \dfrac{2T-D}{\pi(DT-T^2)T(D-T)}\end{bmatrix}=\begin{bmatrix}\dfrac{1}{A}\cdot\left(-\dfrac{1}{D-T}\right)\\ \dfrac{1}{A}\cdot\dfrac{2T-D}{T(D-T)}\end{bmatrix} \tag{4.2}$$

$$\nabla\left(\frac{1}{W}\right)=\nabla\left(\frac{D}{2I}\right)=\begin{bmatrix}\dfrac{\partial(D/2I)}{\partial D}\\ \dfrac{\partial(D/2I)}{\partial T}\end{bmatrix}=\begin{bmatrix}\dfrac{-8D^3+12D^2T-8T^3}{\pi T\left(D^3-3D^2T+4DT^2-2T^3\right)^2}\\ \dfrac{-256D(D-2T)^3}{\pi\left(D^4-(D-2T)^4\right)^2}\end{bmatrix} \tag{4.3}$$

4.2.1 Cost function sensitivity

As the cost function defined in eq. (3.10) is an explicit expression of the given design variables, its sensitivity is easily determined as:

$$\nabla f=\begin{bmatrix}\dfrac{\partial f}{\partial D}\\ \dfrac{\partial f}{\partial T}\end{bmatrix}=\sum_{n=1}^{N}\begin{bmatrix}\dfrac{\partial\left(k_m\rho AL+k_f\left(t_{FO}+t_{CG}+t_{W1}+t_{W2}+t_{W3}+t_P\right)\right)}{\partial D}\\ \dfrac{\partial\left(k_m\rho AL+k_f\left(t_{FO}+t_{CG}+t_{W1}+t_{W2}+t_{W3}+t_P\right)\right)}{\partial T}\end{bmatrix} \tag{4.4a}$$

Below analytic expressions for each component of Eq. (4.4a) are given:

$$\begin{bmatrix}\dfrac{\partial(k_m\rho LA(X))}{\partial D}\\ \dfrac{\partial(k_m\rho LA(X))}{\partial T}\end{bmatrix}=\begin{bmatrix}k_m\rho L\pi T\\ k_m\rho L\pi(D-2T)\end{bmatrix} \tag{4.4b}$$

$$\begin{bmatrix}\dfrac{\partial(t_{FO})}{\partial D}\\ \dfrac{\partial(t_{FO})}{\partial T}\end{bmatrix}=\begin{bmatrix}\dfrac{\partial\left(\Theta e^{\mu(X)}\right)}{\partial D}\\ \dfrac{\partial\left(\Theta e^{\mu(X)}\right)}{\partial T}\end{bmatrix}=\begin{bmatrix}0.00475\Theta e^{\mu}D^{-0.5}\\ 2.265\Theta e^{\mu}T^{-1.5}\end{bmatrix} \tag{4.4c}$$

$$\begin{bmatrix}\dfrac{\partial(t_{CG})}{\partial D}\\ \dfrac{\partial(t_{CG})}{\partial T}\end{bmatrix}=\begin{bmatrix}\dfrac{\partial\left(\Theta_{DC}\left(4.54+0.4229T^2\right)L_c(X)\right)}{\partial D}\\ \dfrac{\partial\left(\Theta_{DC}\left(4.54+0.4229T^2\right)L_c(X)\right)}{\partial T}\end{bmatrix}=\begin{bmatrix}\left(4.54+0.4229T^2\right)2\pi\Theta_{DC}/1000\sin\theta\\ 0.8458DT2\pi\Theta_{DC}/1000\sin\theta\end{bmatrix} \tag{4.4d}$$

$$\begin{bmatrix} \dfrac{\partial(t_{W1})}{\partial D} \\ \dfrac{\partial(t_{W1})}{\partial T} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial\left(C_1\Theta_{dw}\sqrt{k\rho A(X)L}\right)}{\partial D} \\ \dfrac{\partial\left(C_1\Theta_{dw}\sqrt{k\rho A(X)L}\right)}{\partial T} \end{bmatrix} = \begin{bmatrix} C_1\Theta_{dw}\dfrac{k\rho L\pi T}{2\sqrt{k\rho L\pi\left(DT-T^2\right)}} \\ C_1\Theta_{dw}\dfrac{k\rho L\pi(D-2T)}{2\sqrt{k\rho L\pi\left(DT-T^2\right)}} \end{bmatrix} \tag{4.4e}$$

$$\begin{bmatrix} \dfrac{\partial(t_{W2}+t_{W3})}{\partial D} \\ \dfrac{\partial(t_{W2}+t_{W3})}{\partial T} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial\left(1.3C_2a_w^nL_w(X)\right)}{\partial D} \\ \dfrac{\partial\left(1.3C_2a_w^nL_w(X)\right)}{\partial T} \end{bmatrix} = \begin{bmatrix} 2.6\pi C_2a_w^n/\sin\theta \\ 0 \end{bmatrix} \tag{4.4f}$$

$$\begin{bmatrix} \dfrac{\partial(t_P)}{\partial D} \\ \dfrac{\partial(t_P)}{\partial T} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial\left(\Theta_{dp}\left(a_{gc}+a_{tc}\right)A_S(X)\right)}{\partial D} \\ \dfrac{\partial\left(\Theta_{dp}\left(a_{gc}+a_{tc}\right)A_S(X)\right)}{\partial T} \end{bmatrix} = \begin{bmatrix} \Theta_{dp}\left(a_{gc}+a_{tc}\right)\pi L \\ 0 \end{bmatrix} \tag{4.4g}$$

Combining Eq. (4.4) we derive an explicit expression for the gradient vector of the cost function of our optimization problem:

$$\nabla f = \begin{bmatrix} \dfrac{\partial f}{\partial D} \\ \dfrac{\partial f}{\partial T} \end{bmatrix} = \sum_{n=1}^{N} \left[\begin{array}{l} k_m\rho L\pi T \quad + k_f\left(\ 0.00475\Theta e^{\mu}D^{-0.5} + \right. \\ k_m\rho L\pi(D-2T) + \ k_f\left(\ 2.265\Theta e^{\mu}T^{-1.5} \quad + \right. \end{array} \right.$$

$$\begin{array}{ll} +\left(4.54+0.4229T^2\right)\pi\Theta_{DC}/500\sin\theta + & C_1\Theta_{dw}k\rho L\pi T/2\sqrt{k\rho L\pi\left(DT-T^2\right)} \quad + \\ +0.8458DT\pi\Theta_{DC}/500\sin\theta & +C_1\Theta_{dw}k\rho L\pi\left(D-2T\right)/2\sqrt{k\rho L\pi\left(DT-T^2\right)}+ \end{array} \tag{4.5}$$

$$\left. \begin{array}{ll} +2\pi C_2a_w^n/\sin\theta & +\Theta_{dp}\left(a_{gc}+a_{tc}\right)\pi L) \\ +0 & +0) \end{array} \right]$$

In Figure 4.3 it can be seen that the expression of the cost is nonlinear with respect to the variables. We can also notice that both the diameter and the thickness have similar influence.

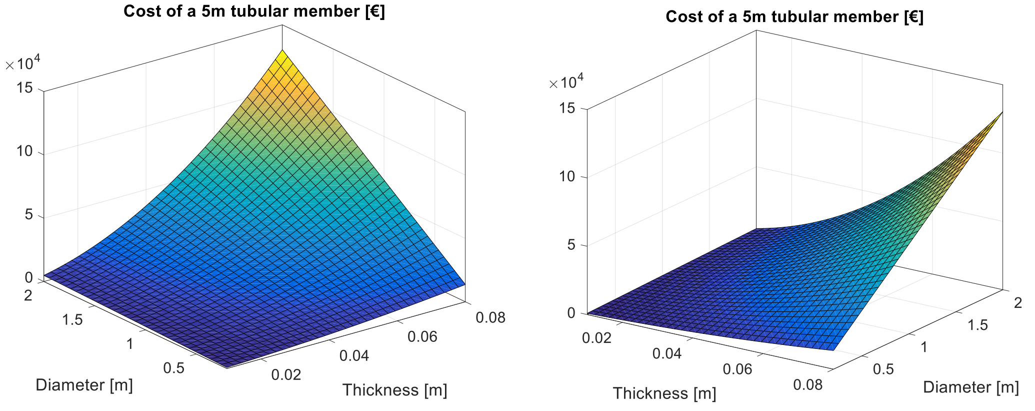


Figure 4.3: Parametric study of the design variables for the cost function of a single member

4.2.2 Sizing constraint sensitivities

The sizing constraints' sensitivities are shown below. It's important to mention that these sensitivity values are applied only to the members that the constraints are applied on. For all the rest of the members, the sensitivities are equal to zero.

$$\nabla g_1 = \begin{bmatrix} \dfrac{\partial g_1}{\partial D} \\ \dfrac{\partial g_1}{\partial T} \end{bmatrix} = \begin{bmatrix} -1 \\ 20 \end{bmatrix}, \qquad \nabla g_2 = \begin{bmatrix} \dfrac{\partial g_2}{\partial D} \\ \dfrac{\partial g_2}{\partial T} \end{bmatrix} = \begin{bmatrix} 1 \\ -64 \end{bmatrix} \tag{4.6}$$

$$\nabla g_3 = \begin{bmatrix} \dfrac{\partial g_3}{\partial D} \\ \dfrac{\partial g_3}{\partial T} \end{bmatrix} = \begin{bmatrix} -1 \;\; or \;\; 1 \\ 0 \end{bmatrix}, \qquad \nabla g_4 = \begin{bmatrix} \dfrac{\partial g_4}{\partial D} \\ \dfrac{\partial g_4}{\partial T} \end{bmatrix} = \begin{bmatrix} 0.2 \;\; or \;\; -1 \\ 0 \end{bmatrix} \tag{4.7}$$

$$\nabla g_5 = \begin{bmatrix} \dfrac{\partial g_5}{\partial D} \\ \dfrac{\partial g_5}{\partial T} \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \;\; or \;\; 1 \end{bmatrix}, \qquad \nabla g_6 = \begin{bmatrix} \dfrac{\partial g_6}{\partial D} \\ \dfrac{\partial g_6}{\partial T} \end{bmatrix} = \begin{bmatrix} 0 \\ 0.2 \;\; or \;\; -1 \end{bmatrix} \tag{4.8}$$

$$\nabla g_7 = \begin{bmatrix} \dfrac{\partial g_7}{\partial D} \\ \dfrac{\partial g_7}{\partial T} \end{bmatrix} = \begin{bmatrix} -20 \\ 0 \end{bmatrix}, \qquad \nabla g_8 = \begin{bmatrix} \dfrac{\partial g_8}{\partial D} \\ \dfrac{\partial g_8}{\partial T} \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \tag{4.9}$$

$$\nabla g_9 = \begin{bmatrix} \dfrac{\partial g_9}{\partial D} \\ \dfrac{\partial g_9}{\partial T} \end{bmatrix} = \begin{bmatrix} 0.8 \;\; or \;\; -1 \\ 0 \end{bmatrix}, \qquad \nabla g_{10} = \begin{bmatrix} \dfrac{\partial g_{10}}{\partial D} \\ \dfrac{\partial g_{10}}{\partial T} \end{bmatrix} = \begin{bmatrix} 0 \\ 0.8 \;\; or \;\; -1 \end{bmatrix} \tag{4.10}$$

In Figure 4.4 the linear sizing constraints g_1, g_2 and the limit surface (="0") are plotted in the design space for a single tubular member. We can notice the constant slope as expected from the linearity and from the equations (4.6) and (4.7).

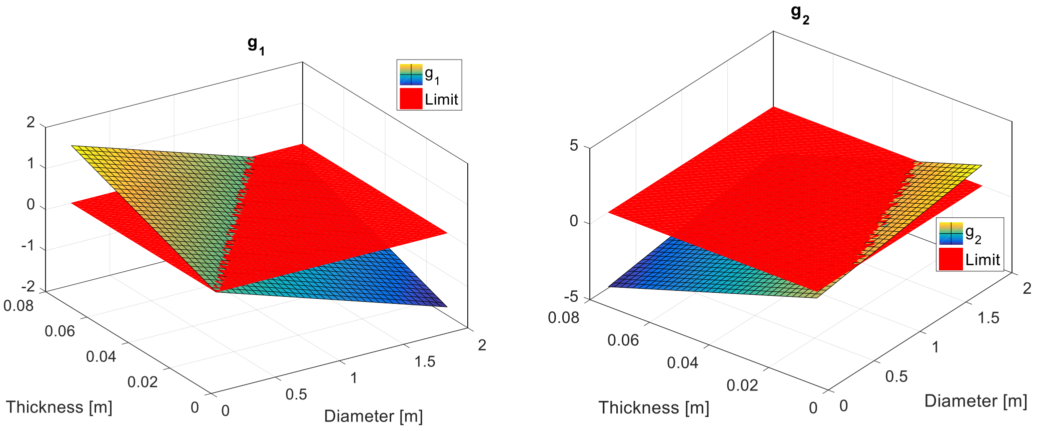


Figure 4.4: Parametric study of the design variables for the sizing constraints g_1 and g_2

In Figure 4.5 the feasible area (acceptable design combinations of *D, T*) with respect to the sizing constraints only, is presented for the whole design space (all possible combinations of *D, T*). This plot corresponds to a single tubular member of the jacket structure.

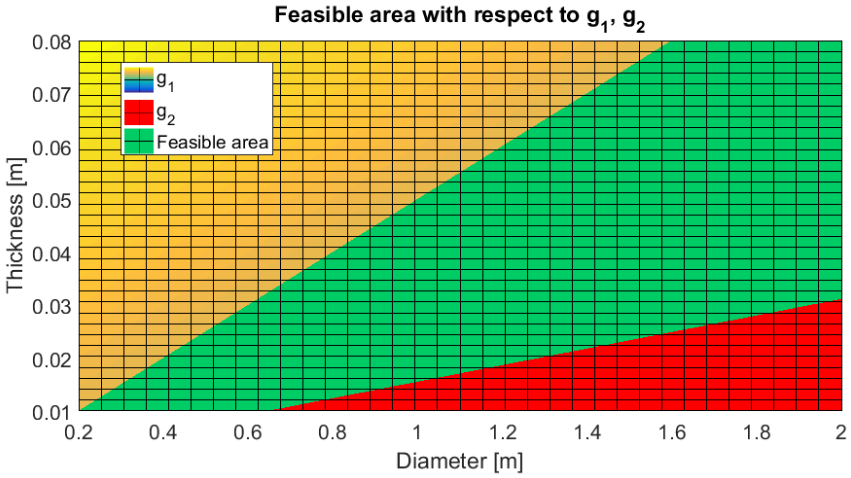


Figure 4.5: The feasible area (green colour) considering only the sizing constraints

4.2.3 ULS sensitivities

Yielding constraint:

$$\nabla g_{11} = \begin{bmatrix} \dfrac{\partial g_{11}}{\partial D} \\ \dfrac{\partial g_{11}}{\partial T} \end{bmatrix} = \begin{bmatrix} \dfrac{-4\mathrm{N}_t}{\pi\sigma_{yield}T(D-T)^2} + \dfrac{(M_y+M_z)(-8D^3+12D^2T-8T^3)}{\pi\sigma_{yield}T(D^3-3D^2T+4DT^2-2T^3)^2} \\ \dfrac{4\mathrm{N}_t(2T-D)}{\pi\sigma_{yield}T^2(D-T)^2} + \dfrac{(M_y+M_z)(-256D(D-2T)^3)}{\pi\sigma_{yield}(D^4-(D-2T)^4)^2} \end{bmatrix} \tag{4.11}$$

Column buckling (Euler) constraint:

$$\nabla g_{12} = \begin{bmatrix} \dfrac{\partial g_{12}}{\partial D} \\ \dfrac{\partial g_{12}}{\partial T} \end{bmatrix} = \begin{bmatrix} \dfrac{64L^2N_c}{n\pi^3E}\cdot\dfrac{-4D^3+4(D-2T)^3}{(D^4-(D-2T)^4)^2} \\ \dfrac{64L^2N_c}{n\pi^3E}\cdot\dfrac{-8(D-2T)^3}{(D^4-(D-2T)^4)^2} \end{bmatrix} \tag{4.12}$$

4.2.4 FLS sensitivities

The differentiation of eq. (3.24) against X gives eq. (4.13).

$$\nabla g_{13} = \begin{bmatrix} \dfrac{\partial g_{13}}{\partial D} \\ \dfrac{\partial g_{13}}{\partial T} \end{bmatrix} = \begin{bmatrix} \sum_{\iota=1}^{l} \dfrac{1}{\bar{a}\eta} \dfrac{1}{T_{ref}^{mk}} \dfrac{dT^{mk}}{dD} n_i \left(\Delta\sigma_{i,HSS}\right)^m + \sum_{\iota=1}^{l} \dfrac{1}{\bar{a}\eta} \left(\dfrac{T}{T_{ref}}\right)^{mk} n_i \dfrac{d\left(\Delta\sigma_{i,HSS}\right)^m}{dD} \\ \sum_{\iota=1}^{l} \dfrac{1}{\bar{a}\eta} \dfrac{1}{T_{ref}^{mk}} \dfrac{dT^{mk}}{dT} n_i \left(\Delta\sigma_{i,HSS}\right)^m + \sum_{\iota=1}^{l} \dfrac{1}{\bar{a}\eta} \left(\dfrac{T}{T_{ref}}\right)^{mk} n_i \dfrac{d\left(\Delta\sigma_{i,HSS}\right)^m}{dT} \end{bmatrix} \tag{4.13}$$

Among the terms, the derivatives of the thickness are calculated:

$$\frac{dT^{mk}}{dD} = 0 \tag{4.14a}$$

$$\frac{dT^{mk}}{dT} = mkT^{mk-1} \tag{4.14b}$$

Furthermore, the derivative $d(\Delta\sigma_{i,HSS})^m/dX$ needs some attention. Since we follow the local approach, the forces at the members remain constant during the optimization but the stresses are changing due to the change in the cross-sectional areas. Given this, and considering that generally $\sigma = F/A$ we can use the following expressions

$$\frac{d\sigma}{dD} = \frac{d\left(\dfrac{F}{A}\right)}{dD} = F\frac{d\left(\dfrac{1}{A}\right)}{dD} = F\frac{1}{A}\cdot\left(-\frac{1}{D-T}\right) = -\frac{1}{D-T}\cdot\sigma \tag{4.15a}$$

$$\frac{d\sigma}{d\mathrm{T}} = \frac{d\left(\dfrac{F}{A}\right)}{d\mathrm{T}} = F\frac{d\left(\dfrac{1}{A}\right)}{d\mathrm{T}} = F\frac{1}{A}\cdot\frac{2T-D}{T(D-T)} = \frac{2T-D}{T(D-T)}\cdot\sigma \tag{4.15b}$$

$$\frac{d\Delta\sigma}{dD} = \frac{d\sigma_1}{dD} - \frac{d\sigma_2}{dD} = -\frac{1}{D-T}\cdot\sigma_1 - \frac{1}{D-T}\cdot\sigma_2 = -\frac{1}{D-T}\cdot\Delta\sigma \tag{4.16a}$$

$$\frac{d\Delta\sigma}{dT} = \frac{d\sigma_1}{dT} - \frac{d\sigma_2}{dT} = \frac{2T-D}{T(D-T)}\cdot\sigma_1 + \frac{2T-D}{T(D-T)}\cdot\sigma_2 = \frac{2T-D}{T(D-T)}\cdot\Delta\sigma \tag{4.16b}$$

Then we can calculate the derivatives as:

$$\frac{d\left(\Delta\sigma_{i,HSS}\right)^m}{dD} = m\left(\Delta\sigma_{i,HSS}\right)^{m-1}\cdot\frac{d\Delta\sigma_{i,HSS}}{dD} = -\frac{1}{D-T}m\left(\Delta\sigma_{i,HSS}\right)^m \tag{4.17a}$$

$$\frac{d\left(\Delta\sigma_{i,HSS}\right)^m}{dT} = m\left(\Delta\sigma_{i,HSS}\right)^{m-1}\cdot\frac{d\Delta\sigma_{i,HSS}}{dT} = \frac{2T-D}{T\left(D-T\right)}m\left(\Delta\sigma_{i,HSS}\right)^m \tag{4.17b}$$

Finally, the FLS constraint sensitivity is given by the following expression:

$$\nabla g_{13} = \begin{bmatrix} \dfrac{\partial g_{11}}{\partial D} \\ \dfrac{\partial g_{11}}{\partial T} \end{bmatrix} = \begin{bmatrix} 0 + \\ \displaystyle\sum_{\iota=1}^{l}\frac{mk}{\bar{a}\,\eta}\frac{T^{mk-1}}{T_{ref}^{mk}}n_i\left(\Delta\sigma_{i,HSS}\right)^m + \\ \displaystyle -\frac{1}{D-T}\cdot\sum_{\iota=1}^{l}\frac{m}{\bar{a}\,\eta}\left(\frac{T}{T_{ref}}\right)^{mk} n_i\left(\Delta\sigma_{i,HSS}\right)^m \\ \displaystyle \frac{2T-D}{T\left(D-T\right)}\cdot\sum_{\iota=1}^{l}\frac{m}{\bar{a}\,\eta}\left(\frac{T}{T_{ref}}\right)^{mk} n_i\left(\Delta\sigma_{i,HSS}\right)^m \end{bmatrix} \tag{4.18}$$

At this point it is important to mention that due to the local approach, there are two other assumptions made here in order to calculate the fatigue damage sensitivities. The first one is that during the internal sizing steps, the stress concentration factors are kept constant. If this assumption is not made, then the change in dimensions of one member, affects the loads in another and the independent sizing of the members is not possible.

The second assumption we make here is that the slope parameters m (= 3 or 5) used in the S-N curve during the rainflow counting is also assumed to be constant during the internal sizing step. If this was not true, then the application of a gradients based algorithm becomes very tedious as the first derivative of the fatigue constraint, wouldn't be a continuous function. which is one of the requirements for many gradient based algorithms (especially those ones that are using second order information of the problem functions). Of course, the above two assumptions are valid only for small changes in the design variables during the local (internal) sizing but are essential in the local approach.

The following figures (Figure 4.6 & Figure 4.7) are showing qualitatively how the fatigue damage is varying with respect to the design variables during the internal sizing steps, under the aforementioned assumptions. Loads and SCFs on the members are kept constant.

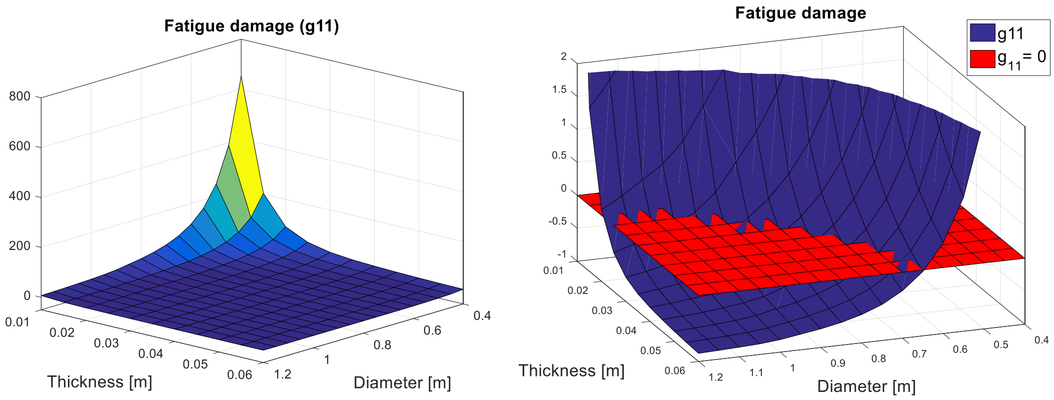


Figure 4.6: Parametric study of the design variables for the fatigue damage at a tubular joint

The Figure 4.6 shows how the fatigue damage of a certain location is changing according to the dimensions of one of the members at a tubular joint. More specifically, the maximum fatigue damage out of the 8 hot spot locations at the brace side of K-joint is plotted against the brace dimensions of this location. The next figure is showing the feasible for the dimensions of the previous considered K-brace member, when keeping the SCFs constant.

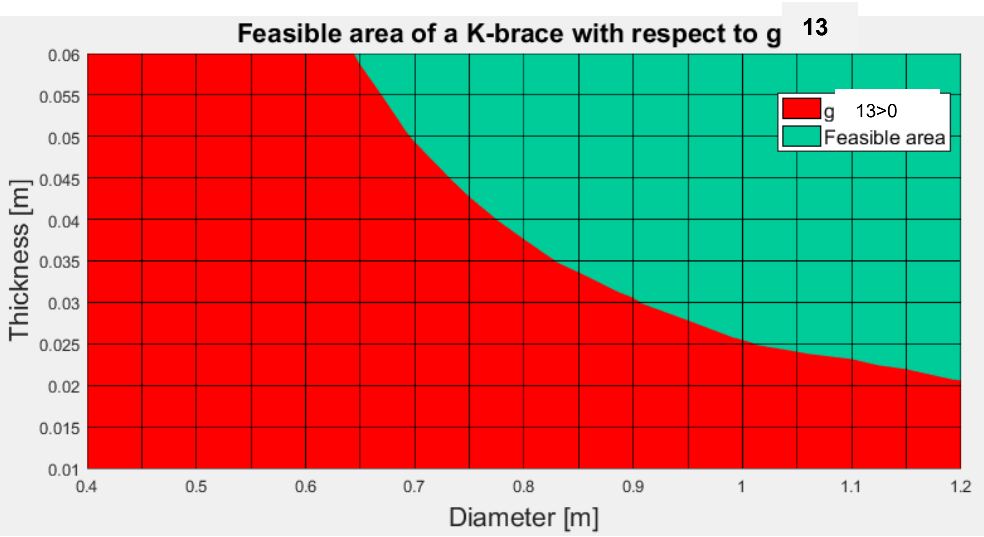


Figure 4.7: The feasible area (green colour) considering only the fatigue constraint (g_{13})

4.3 Optimization algorithm

The internal sizing step of the jacket structure was performed by the use of a gradient-based technique. In order to choose a proper optimization algorithm, we need to take into consideration the characteristics of the optimization problem to be solved. The following table provides all this information.

Table 4.1: Characterization of the optimization model for the jacket support structure

Aspect	*Description*
Type	Size optimization
Availability of derivatives	Yes (gradient-based)
Number of objective functions	One (single objective)
Linearity of objective function	Nonlinear
Presence of constraints	Yes (constrained)
Type of constraints	Inequality
Linearity of constraint functions	Nonlinear
Type of variables	Continuous real numbers
Continuity of functions	Continuous with continuous 1st order derivatives

Considering the previous table, a Sequential Quadratic Programming (SQP) method was selected. The SQP is a very powerful gradient-based method embedded in Matlab and is called through the function *fmincon*. The method has no other limitations except that it is designed to work on problems where the objective and constraint functions are both continuous and have continuous first derivatives. Some of the characteristics of the method are described in this chapter. Detailed information on the method and its application is provided in [37], which is available online. An overview of SQP methods can be found in [38].

The inputs for the SQP algorithm in order to be able to provide optimized solutions, are:

1. *Cost function* – An expression for the objective function to be minimized.

2. X^0 – An initial point/design specified as a vector which contains values of the design variables. The number of elements in X is the number of variables.

3. *Linear constraints* – The constraints are given separately depending on their type (inequality or equality).

4. *Non-linear constraints* – The nonlinear constraints (equalities or inequalities are given separately in a function that is called by the optimizer.

5. *Gradient information* – *A*nalytic gradients derived from a sensitivity analysis can be provided in order to increase the efficiency and accuracy of the method.

Given an initial design X^k and the corresponding values for the cost function, the constraints and their gradients, the SQP method used here works in four steps at each major iteration $\boldsymbol{k}$ as explained below:

1. Calculate the Hessian Matrix. At each major iteration an approximation of the Hessian (second order derivative) of the Lagrangian function (see chapter 2.4.2), $\boldsymbol{H}$, is calculated using the *Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula*. A summarized algorithm of BFGS can be found in [9, 8], while detailed derivation of the method is given by [6] and by [39]. The Hessian combined with the gradients are then used at the next step.

Lagrangian: $$L\left(X^{\kappa},\lambda^{k}\right)=f\left(X^{k}\right)+\sum_{j=1}^{J}\lambda_{j}^{k}g_{j}\left(X^{k}\right)+\sum_{q=1}^{Q}\lambda_{q}^{k}h_{q}\left(X^{k}\right) \qquad (4.19)$$

Hessian: $$H^{k}=\nabla_{xx}^{2}L\left(X^{k},\lambda^{k}\right) \qquad (4.20)$$

2. Create a quadratic approximation (QP subproblem) subject to linearized constraints as it can be seen in eq. (4.18). The solution of the QP problem provides the direction of improvement $\boldsymbol{d}$ (search direction). The QP problem of finding the minimum of a quadratic is linear, since you just have to take the gradient, a linear function, and set it to zero. The method used in MATLAB is an active set strategy (also known as a projection method) similar to that of Gill et al., described in [40], [41] and modified for Quadratic Programming (QP) problems. The BFGS method used in the previous step, guarantees that the Hessian is positive definite, meaning that the scalar $\boldsymbol{d^{T}Hd}$ is always positive and as a result the QP subproblem is convex and its solution is unique [5].

$$\begin{array}{ll} \text{find:} & d \\ \text{to minimize:} & Q(d)=\frac{1}{2}d^{T}H^{k}d+\nabla f\left(X^{k}\right)^{T}d \\ \text{subject to:} & \nabla g_{j}\left(X^{k}\right)^{T}d+g_{j}(X^{k})\leq 0,\ \text{j=1,..., J} \\ & \nabla h_{q}\left(X^{k}\right)^{T}d+h_{q}(X^{k})=0,\ \text{q=1,..., Q} \end{array} \qquad (4.21)$$

3. Create a step size calculation subproblem. This subproblem is unconstrained since a merit function (also called penalty function) that combines the objective function with the active constraints at the considered point is constructed according to [42] and [43]. The step length parameter $\boldsymbol{\alpha}$ is determined in order to minimize this penalty function which results in objective function and constraint violations minimization, at the same time.

Merit function:
$$P\left(X^{\kappa}\right)= f\left(X^{k}\right)+\sum_{j=1}^{J} r_j \cdot \max\left[0, g_j\left(X^{k}\right)\right]+\sum_{q=1}^{Q} r_q h_q\left(X^{k}\right) \qquad (4.22)$$

4. Form the new design as: $X^{k+1} = X^{k} + a \cdot d$

For more details on SQP formulation and how it operates can be found in [44], but two important features of the algorithm are described in the following two paragraphs.

The SQP algorithm takes every iterative step in the region constrained by bounds. Bounds are not strict meaning that a step can be exactly on a boundary. Characteristic of the method is that the constraints are satisfied at every iteration. However, during the iterations, the SQP algorithm can attempt to take a step that fails. In this case, the algorithm attempts again by trying a smaller step. When the problem is infeasible, the method attempts to minimize the maximum constraint value.

When constraints are not satisfied, the solution of the general quadratic subproblem defined in Eq. (4.21) for finding the direction vector at each iteration step follows two approaches: The first one is that SQP algorithm combines the objective and constraint functions into a merit function. The algorithm attempts to minimize the merit function subject to relaxed constraints. The second one is that if an attempted step causes the constraint violation to grow, the SQP algorithm attempts to obtain feasibility using a second-order approximation to the constraints. Both approaches can lead to a feasible solution but they have some defects in terms of efficiency of the method. The first one slows the solution of the subproblem because it creates more variables than the original subproblem and therefore increases the size of the problem. The latter one also slows the solution since it requires more evaluations of the nonlinear constraint functions.

4.3.1 Termination criteria

The formulation of the optimization problem and the assumption of the local approach requires termination criteria in two levels, in an outer (global) and in an internal (local) level.

The local convergence criteria limit the internal sizing process and are set within the *fmincon* function that applies the SQP method. Due to the local approach and the assumption that forces are not changing during the local sizing, the target tolerance for the constraint violation was set to -0.2. This way, the locally optimized design has a redundancy towards the constraint limits that will be needed for the time domain analysis with the optimized dimensions that follows. The local convergence criteria chosen for this study were:

Termination tolerance on the constraint violation:	TolCon $\leq$ 5*10^{-2}	(4.23)
Termination tolerance on the function value:	TolFun $\leq$ 2*10^{-2}	(4.24)
Termination tolerance on X:	TolX $\leq$ 5*10^{-3}	(4.25)
Maximum number of iterations:	MaxIt $\leq$ 15	(4.26)

The first tolerance (*TolCon*) is an upper bound and operates differently from the other tolerances. The tolerance must be satisfied at the optimum point but if it is not satisfied the solver attempts to continue. The next two tolerance are lower bounds. This means that iterations end when changes in the values of function or step size (change in variable size, TolX) are less than the tolerance values. This changes can be either absolute (when the magnitude of the considered measure is less than one) or relative (when the magnitude of the measure is larger than one). The algorithm stops when:

$$\max\left\{abs\left(\frac{f(x_{k-1})-f(x_k)}{f(x_{k-1})}\right), abs\left(f(x_{k-1})-f(x_k)\right)\right\} \leq TolFun \tag{4.27}$$

$$\max\left\{abs\left(\frac{x_k-x_{k-1}}{x_{k-1}}\right), abs\left(x_k-x_{k-1}\right)\right\} \leq TolX \tag{4.28}$$

The maximum number of iterations is an absolute upper bound and if exceeded the algorithm ends.

The global convergence criterion was defined as follows. The whole iterative process will be terminated after the "k^{th}" iteration when all the constraints are satisfied and at the same time the following convergence criterion is satisfied:

$$\left| \frac{f(x_{k-1}) - f(x_k)}{f(x_{k-1})} \right| < \delta \tag{4.29}$$

For our study this criterion was set to ***δ=2 %*** and implies that the variation of the total cost of the jacket structure does not exceed the ***2%*** between two consecutive global optimization steps (evaluated after each simulation analysis with FEDEM software is completed).

4.4 Novelty and distinctive features of the method

Three key features of the current thesis are of great importance and have a scientific interest. The first one is the objective function which in the current work takes into consideration both the material and the manufacturing costs of a jacket support structure. Realistic cost factors for both costs are taken into account considering up to date market and labor costs. Since the aim of an optimization process is to minimize the cost of structure, the result here is that a more accurate measure is used compared to weight that is typically used in structural optimization procedures. In this way, the cost is calculated directly in terms of currency and the savings (or loss) as a result of the optimization process is realistic and accurate.

The second interesting feature of the thesis is the way the fatigue constraint (g_{13}) sensitivity is calculated. The gradient of this constraint is calculated analytically, but under the assumption of the local approach, an analytic expression is derived much easier by scaling (see Eq. (4.18)) the stress ranges $\Delta\sigma_{i,HSS}$ calculated through the rainflow counting.

The last characteristic of this study is the optimization framework itself. It is a gradient-based iterative process, taking into account the objective function and all the design variables and constraints of the system to calculate the new design at each iteration (global optimization) but at the same time assumes that the sectional responses (forces) of the members calculated in the last FEM analysis, remain the same during the sizing of the members until a new FEM analysis is performed (local approach). It is a kind of *global-local* size optimization, if we are allowed to use this term. The result is an efficient process, meaning accurate (analytical sensitivities can be calculated) and fast (only one FEM analysis for each iteration step, where all the members are optimized).

5 RESULTS

The results presented in this chapter are divided in two different design configurations for the jacket, one with the reduction factor (Dred) set equal to 0.6 and one with Dred=1.0. The reason for this is that the first approach (Dred=0.6) is presented for benchmarking reasons with earlier studies [2]. The second category is fabrication-wise easier to build and closer to the common practice of jacket construction while still allowing the final designs to distinguish between members of different bays. Each subsection of this chapter will include figures of both of this two designs approaches.

In addition, the results are based on a number of selected optimization runs, whereas each optimization run consists of three up to seven evaluation (full time domain analysis). For the different optimization runs, the design variables for the initial designs were randomly chosen within the constraints. The analysis was focused on the optimization progress, with special concern on the sizing process, both in terms of accuracy and cost, as a central functionality of the local optimization approach. As an important characteristic in structural design, the resulting eigenfrequency of the structure was investigated for all the designs. For presentation reasons and clarity, the results presented in this chapter have been scaled as the threshold is set to 1 instead of 0 (as it is the formulation in chapter 3).

5.1 Progress

The major challenge in this study is to prove if the main idea of changing all members at all locations at the same time, combined with the local approach assumptions, leads to useful results. Figure 5.1 and Figure 5.2 give an overview over the progress of all central results in the optimization and the performance of the optimization process is presented. Results are given for a number of data samples, having their starting point randomly chosen. For these several initial designs the method proposed achieves convergence to a similar final design. Initial designs start with constraint values that may or may not exceed the limits (=1.0). The horizontal axis in this plots gives the generation step where odd number of generation means a full time domain evaluation and an even number represents an internal sizing step with the local approach.

While the structural cost is converging for an increasing number of generations, maximum results for the non-linear constraints of all cross-section categories are forced to get under the threshold of 1.0. Overall the simulation sizing process shows very good progress with fast convergence. A final design is found after maximum 11 generations when Dred=0.6 and maximum 13 generations when Dred=1.0. This means that at maximum only six (seven in case of Dred=1.0) full time domain analysis are needed to achieve an optimized design. In average, 5.1 and 5.2 full time domain evaluations are needed when Dred=0.6 and Dred=1.0 respectively. Significant constraint reduction is achieved after the first two internal sizing processes (each internal process consists of several internal steps). The cost as an objective function may be higher or lower at the final design compared to the initial one. This is the idea of an optimization process as the initial design may have a low cost but not fulfilling all the constraints or vice versa.

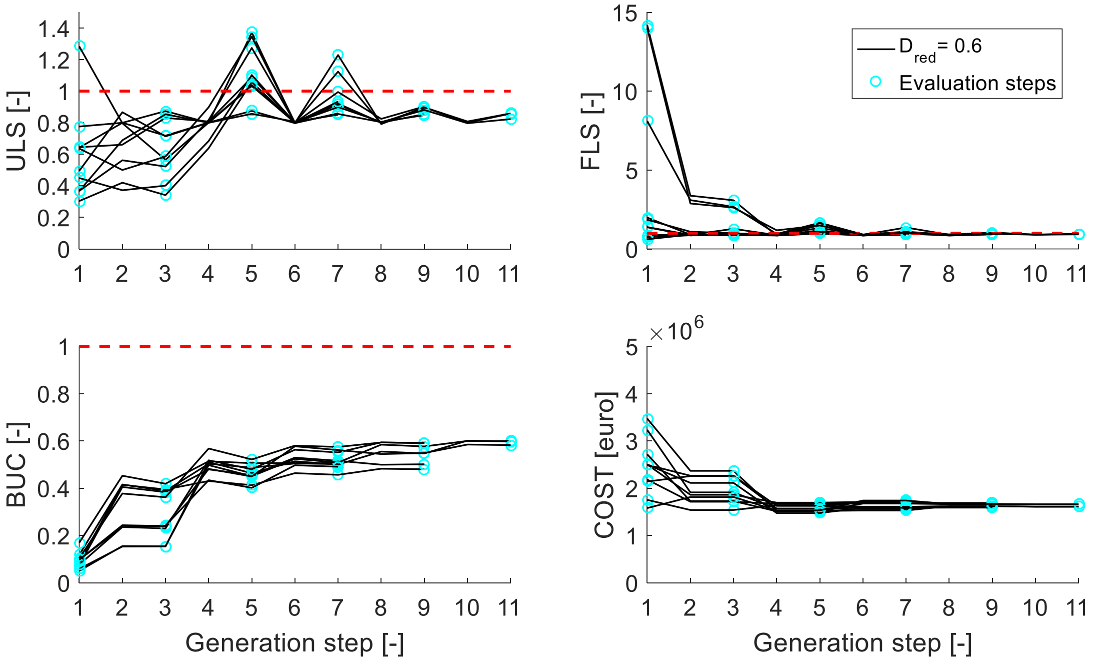


Figure 5.1: Optimization progress for ULS, FLS, BUC and total cost. Data are presented for ten different, independently started optimization runs (D_{red}=0.6).

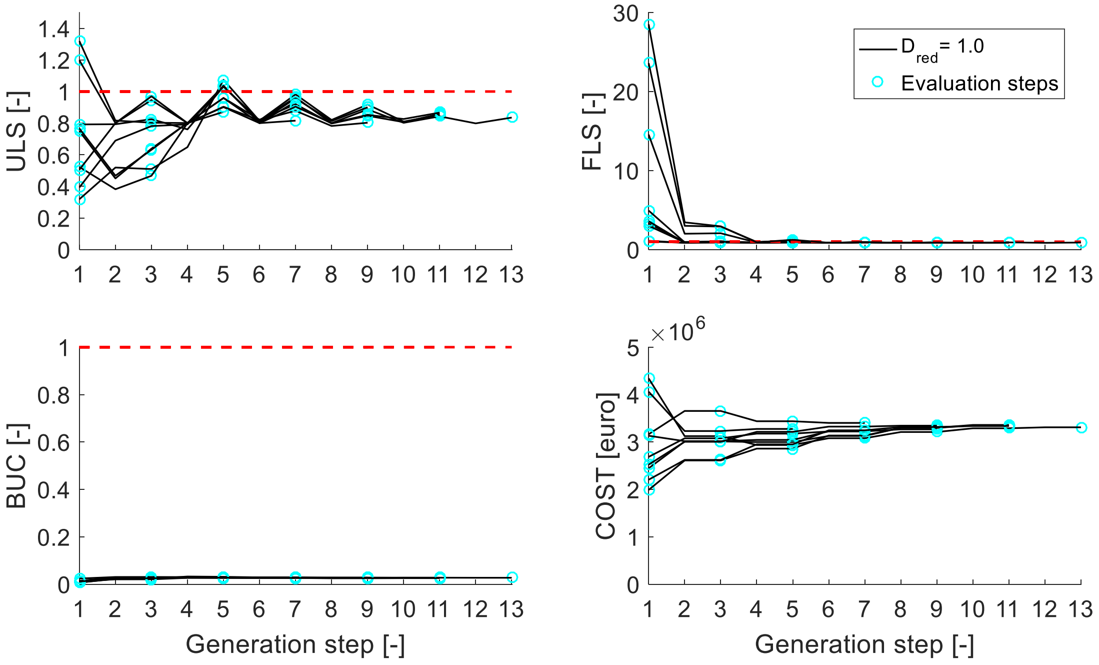


Figure 5.2: Optimization progress for ULS, FLS, BUC and total cost. Data are presented for ten different, independently started optimization runs (D_{red}=1.0).

The cost shows largest changes in the first two evaluation steps (generations steps 1 to 3 and 3 to 5), and converge after several generations. Initial designs start out with fatigue damage or ultimate stress constraint above and below the threshold of 1.0 as indicated by the first evaluation. The buckling constraint values stay significantly below the threshold of 1.0 especially for the configuration with Dred=1.0. A significant fatigue damage reduction is achieved already after the first evaluation.

The optimization progress also shows that the internal sizing process can cope well with the challenge of a large number of design variables and constraints chosen here. While the initial designs significantly differ in ULS, FLS performance and cost, the resulting cost of the structures varies only between 1.61 and 1.69 million euro for the case of Dred=0.6 and between 3.25 and 3.39 million euro for the case of Dred=1.0. The average structure cost obtained is 1.65*10^6 and 3.31*10^6 euro respectively for the two configurations. It is good to remind here, that these values achieved with respect to the termination criterion of the optimization process that was set to 2% for the cost function. If this threshold is even lower, then the variation of the final costs will be even less.

5.2 Accuracy

The success of the strategy of the local approach is directly dependent on the accuracy which can be achieved during the internal sizing process. The deviation of predicted (while sizing) versus achieved (time-domain simulation) results is normalized to internal sizing results, e.g., a positive deviation indicates that the result found by simulation is higher than what was predicted by the internal sizing process. In other words, a negative deviation means that the internal sizing overestimated the ULS and FLS constraint performance. In figures Figure 5.3 - Figure 5.8, the accuracy of the method is presented separately for each of the three cross sectional types (Klegs, Kbraces, Xbraces) and for the two design configurations (Dred=0.6 & 1.0). Results for the ULS and FLS constraints are presented only, as these are found to be the guiding constraints of the design process. Furthermore, a quadratic approximation for each constraint (ULS, FLS) is plotted on each figure along with the scatter points.

The overall impression confirms the expectation, that a large number of scatter points is found around the origin (0,0), where small deviations come along with small cross sectional changes. Looking at all accuracy figures, the following statements can be made: first, the results seem to have the same trend regardless of the jacket configuration, i.e. the accuracy of the local approach is independent from the Dred factor; secondly, as expected, generally the larger the cross-sectional area change the larger the deviation is. However, about 80% of the samples of each cross section type (Klegs, Kbraces, Xbraces) is located inside the ± 20% deviation region; thirdly, for the Klegs, ULS deviations are in general smaller compared to the FLS; fourthly, for the Xbraces the FLS deviations are smaller than the ULS. FLS deviations higher than ±20% for small (±10%) cross sectional changes can be explained by the difference in the SCF values during the internal sizing steps. This type of deviation is more intense between the first evaluation steps. This analysis provides useful information for the adjustment of the optimization parameters and leaded also to an ULS and a FLS target values of 0.80 (when the threshold is 1.0) for the internal sizing process.

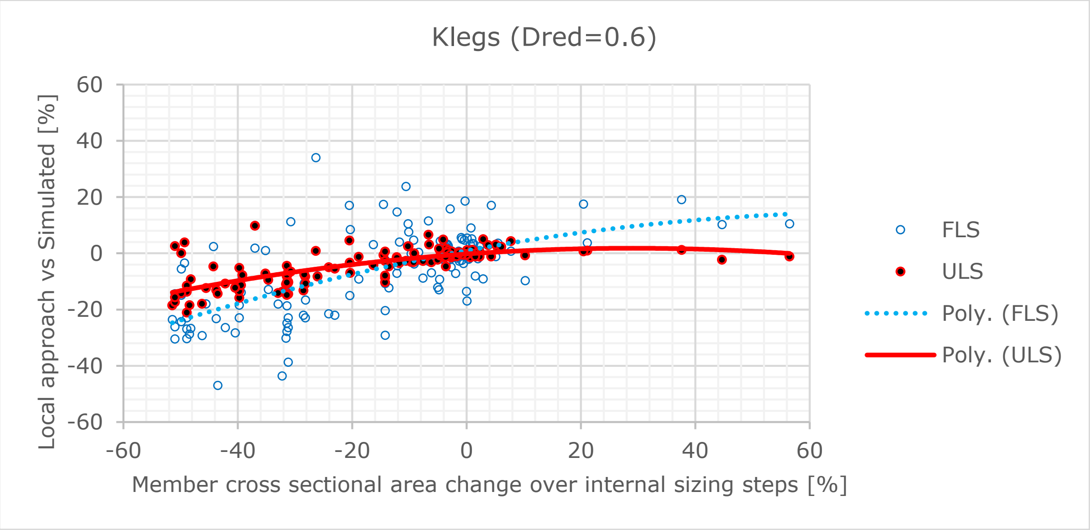


Figure 5.3: Accuracy of internal sizing step for the Kleg cross sectional area change in between two time domain evaluations (Dred=0.6).

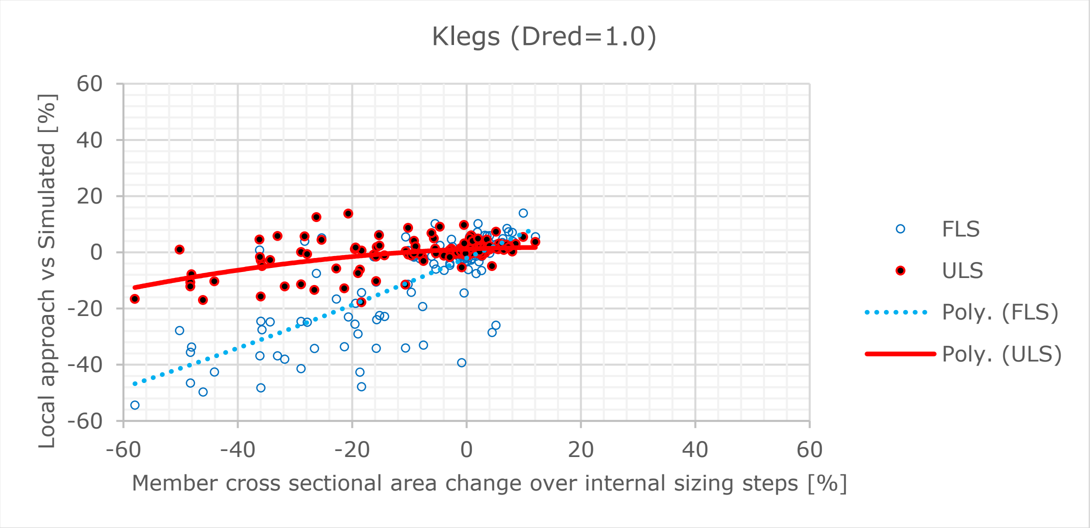


Figure 5.4: Accuracy of internal sizing step for the Kleg cross sectional area change in between two time domain evaluations (Dred=1.0).

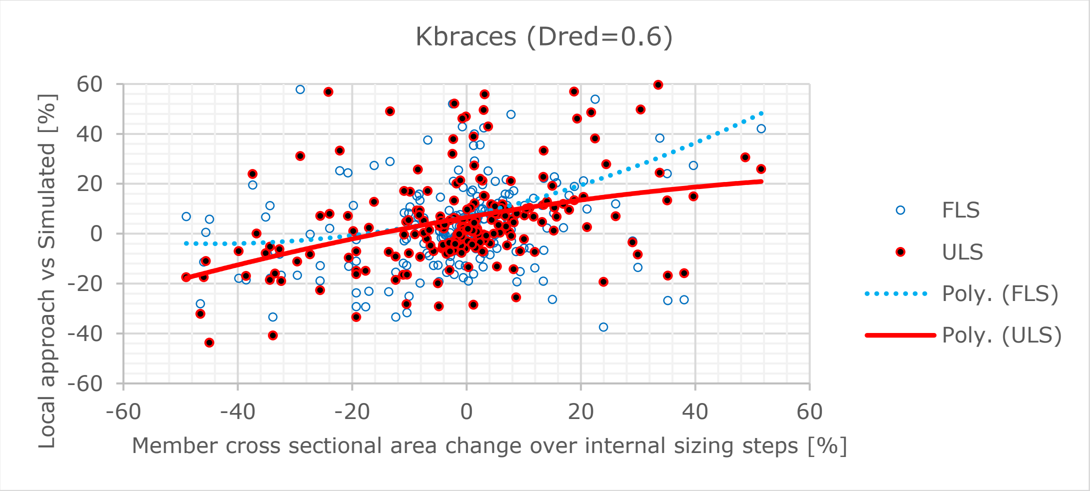


Figure 5.5: Accuracy of internal sizing step for the Kbrace cross sectional area change in between two time domain evaluations (Dred=0.6).

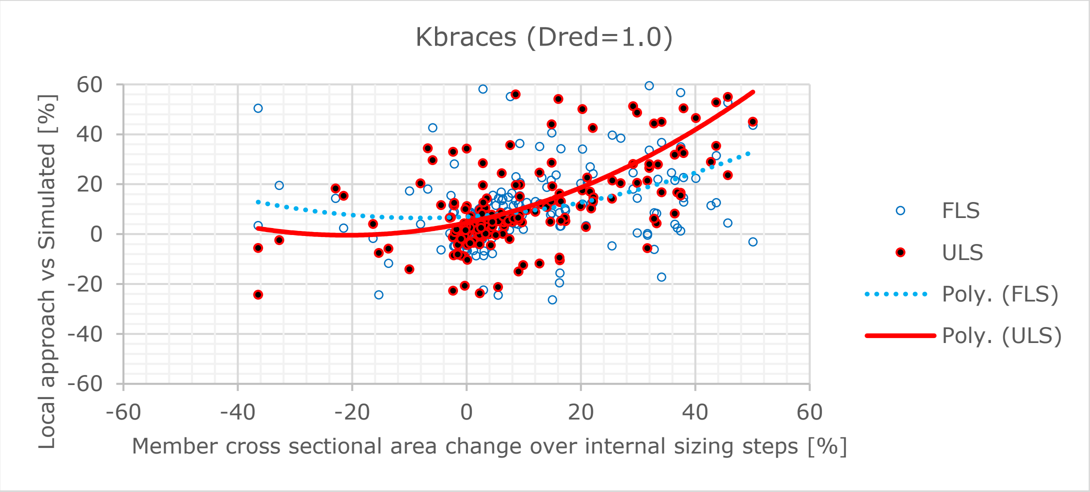


Figure 5.6: Accuracy of internal sizing step for the Kbrace cross sectional area change in between two time domain evaluations (Dred=1.0).

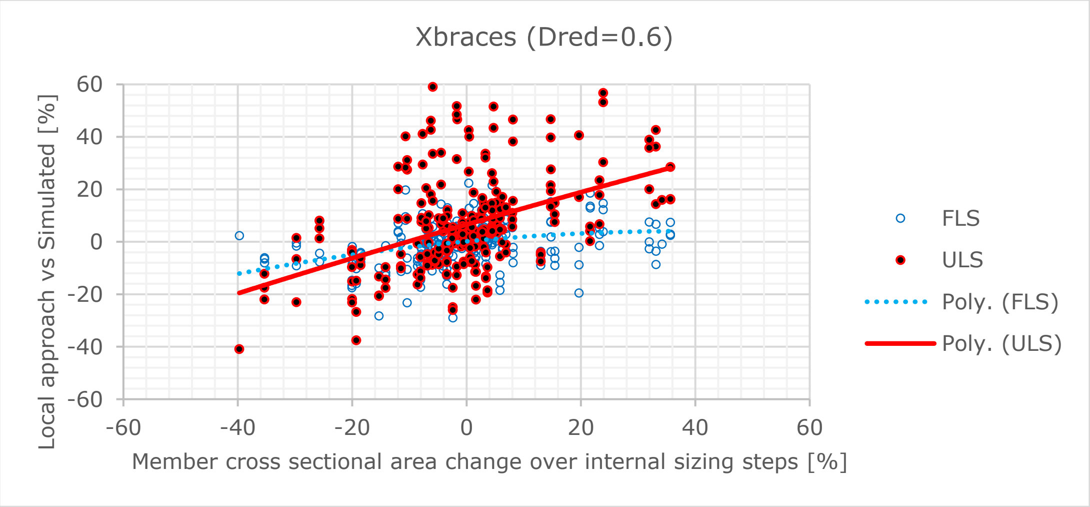


Figure 5.7: Accuracy of internal sizing step for the Xbrace cross sectional area change in between two time domain evaluations (Dred=0.6).

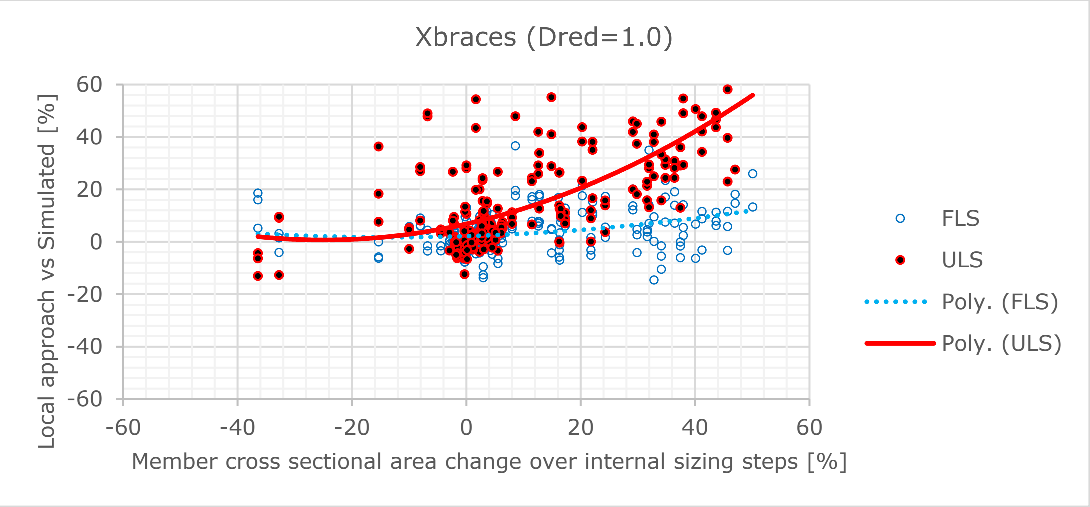


Figure 5.8: Accuracy of internal sizing step for the Xbrace cross sectional area change in between two time domain evaluations (Dred=1.0).

Talking about the accuracy of the method, as mentioned earlier the optimization framework achieves convergence around the same values of for the final dimensions of the jacket and therefore for the objective function (see figures 5.1, 5.2). For the case with Dred=0.6, the final designs have a cost between 1.61 and 1.69 million euro. In terms of structural weight, the final designs weigh between 270 and 281 tons. For the case with Dred=1.0, final costs vary between 3.25 and 3.39 million euro, while structural weight varies between 695 and 713 tons. Considering all the above, we can conclude that the proposed optimization framework is an accurate and robust process.

5.3 Efficiency

The cost change achieved during the sizing process is developed over a number of several sizing processes. Structural dimensions (D,T) are changed stepwise towards the design constraints, while step quantities depend on the values of all the constraints and the analytically calculated gradients at the considered point. Generally, it was noticed that FLS and sometime the ULS were the driving design factors (i.e. these two constraints were violated first before others). Designs with an initial design closer to the optimum need fewer iterations to converge. The largest cost savings are achieved within the first internal sizing process (up to $1*10^6$ euro), which requires in average 6.64 sizing steps.

For judging the efficiency of the method, we should investigate the time needed to achieve optimum designs. In average (considering both cases for Dred), after the first evaluation (initial design), the optimization process requires eight generation steps. As an internal sizing is always followed by an evaluation, these generations consist of four sizing processes and four time domain evaluations. One single finite element solver process was estimated to take about 23.6 minutes in average. This is the execution time of the external solver (FEDEM Windpower) for a 300-second load case (dynamic analysis of the entire offshore wind turbine under aerodynamic and hydrodynamic loading is performed). The external solver evaluation contained 5 design load cases which simulated in parallel. In contrast, the internal sizing for all load cases was performed in series. Each sizing process consists of 5.91 sizing steps in average. At the same time, a single sizing step including the evaluation of all the constraints (ULS, FLS, etc) was calculated to take 9.6 minutes. The duration for the sizing step is mainly driven by the reading speed of the hard disc, since several input files are required for the ULS, FLS and BUC analysis out of which the fatigue analysis and the performance of rainflow counting is the most time consuming of all. The comparison of a similar optimization approach using local assumptions proposed in [2] is listed in the following table.

Table 5.1: Comparison of computation time needed for an optimization with local assumptions and approximate sensitivity analysis versus the analytical sensitivity analysis approach.

Approach	*Evaluation time*	*Sizing time*	*Total time*
Approximate sensitivity analysis	4.1*23.6 = 96.8 min	8.2*5.5 = 45.1 min	141.9 min
Analytical sensitivities	4.0*23.6 = 94.4 min	4*5.9*9.6 = 226.6 min	321.0 min

The local approach with the analytical gradients is more time consuming and this happens for two reasons. Firstly, as expected, the time needed at each sizing step is almost double for the analytical case and this is due to the analytical calculation of the gradients, especially those of the nonlinear-constraints (ULS, FLS, BUC). The second reason is the number of the internal sizing steps needed in average (8.2 versus 23.6). This difference is highly affected by the values of the termination criteria used for the internal sizing process. Loosening these criteria but still ensuring that all the constraints are satisfied, the internal steps needed can be reduced from 23.6 to 12 (4*3) or even less as it was simulated in some tests. However, the exact and optimum selection of the termination criteria can be the objective of another study. Furthermore, it is good to keep in mind that time efficiency should always be regarded along with the final (optimum) designs obtained and the corresponding cost function savings. For the aforementioned case, the approximate analysis achieved in average a final design of 285 tons structural weight while the analytical approach achieved an average of 275 tons (both configurations are for Dred=0.6).

At this point we should keep in mind two important factors affecting the whole process. Firstly, we should consider the capabilities of the machine which performs the process. All the times presented above correspond to runs that were performed on a powerful university server.

Just for comparison reasons, if we use a personal laptop with the following specifications:

- Processor: Intel(R) Core(TM) i7-4700MQ CPU @ 2,40 GHz,
- RAM: 6.00 GB,
- System type: 64-bit operating system, x64-based processor.

then the 300-second load case would require 130 minutes in average to be executed instead of just 23 minutes with the university server. The second factor that affects the efficiency of the method is the termination criteria used for both the internal sizing process and the whole optimization framework (see section 4.3.1). If these parameters are tightened or loosened, then this also affects significantly the time required to achieve optimum designs.

5.4 Optimum designs

The obtained final designs, as a distribution of the dimensional properties (D, T) over the tower height, are presented in figures Figure 5.9 and Figure 5.10. In these figures the mean values are presented by the plotted curves and the data samples of all the final designs are plotted as well. As an overall picture from these figures, we can see that the final dimensions are larger for the configuration of the jacket with Dred=1.0. The main reason for this is the higher hydrodynamic load that the structure suffers due to the increase of the diameter at the middle sections of the legs and the braces. This explains also why these larger dimensions appear at the members below or close the mean sea level (MSL at z=0m) but the values at the top bay (bay 4) present minor differences between the two configurations. The influence of wave loads in the splash zone is visible for the Klegs and the Kbraces (K-joint between bays 3 and 4 at z=4.3m) in the case of Dred=0.6. In contrast in the case of Dred=1.0, this is obvious only for the Kbraces. As a result, the diameter and thickness at these locations show an increase compared to the neighboring bays. Another interesting feature of the structure is that the diameter and thickness of all Klegs and Xbraces show the same behaviour over the tower height, with larger decreasing dimensions as we move towards the bottom or the top of the jacket.

Looking into the figures more detailed, Klegs and Xbraces have their largest dimensions in bay 2. This applies for both configurations for the different Dred factors. Kbraces have their largest values around the intersection of bays 3 and 4 for Dred=0.6 while their largest dimensions appear at bay 2 when Dred=1.0. However, this latter remark, is a result of the Kbrace dimensions trying to fulfill the geometrical constraints (in this case they should have dimensions larger or equal to the braces attached to them which by turn are equal to Xbraces for the case of Dred=1.0) and not to satisfy the nonlinear constraints (ULS, FLS, BUC). This is confirmed by Figure 5.11 at the right, where we can see that in these locations the maximum constraint (nonlinear) values are well below the target value of 1.0. Moreover, we can notice small increases in diameter and thickness of Kbraces from the lower to upper part of the K-joints. Only the Kbraces at upper bay 2 (for Dred=1.0) and the Kbraces at the highest K-joint (for the Dred=0.6 case) do not follow this trend.

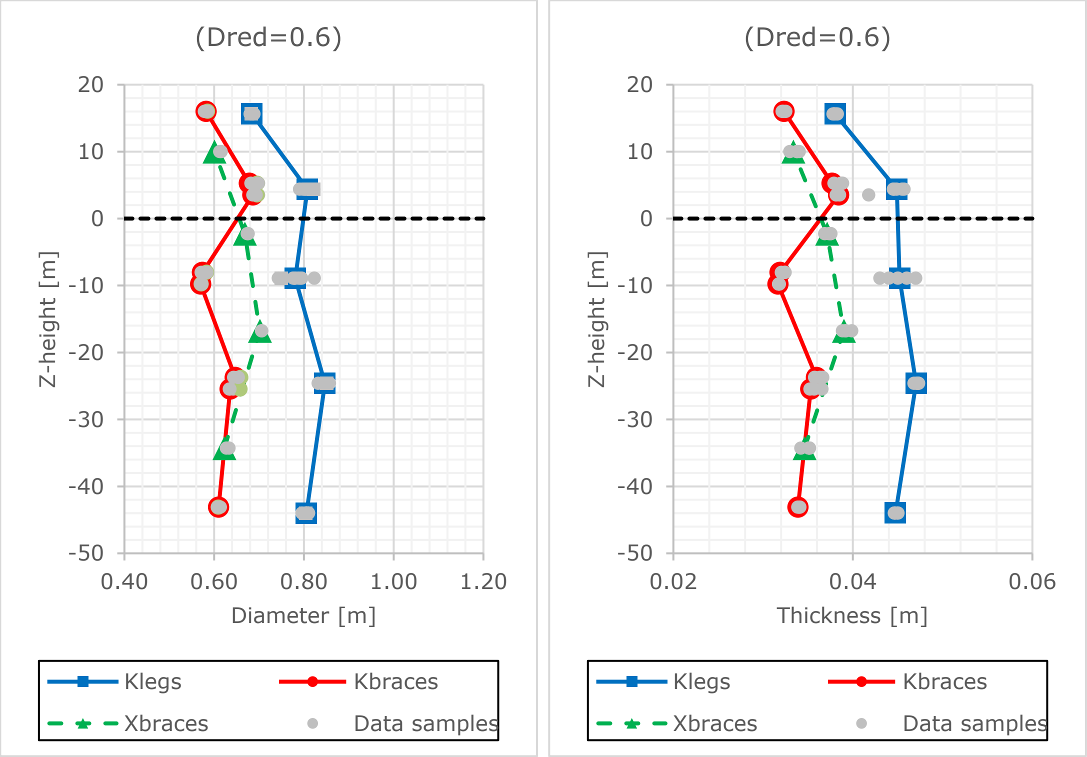


Figure 5.9: Mean dimensions of the final (optimum) designs for all the members over the tower height. Data samples of all optimized designs are provided as well (Dred=0.6).

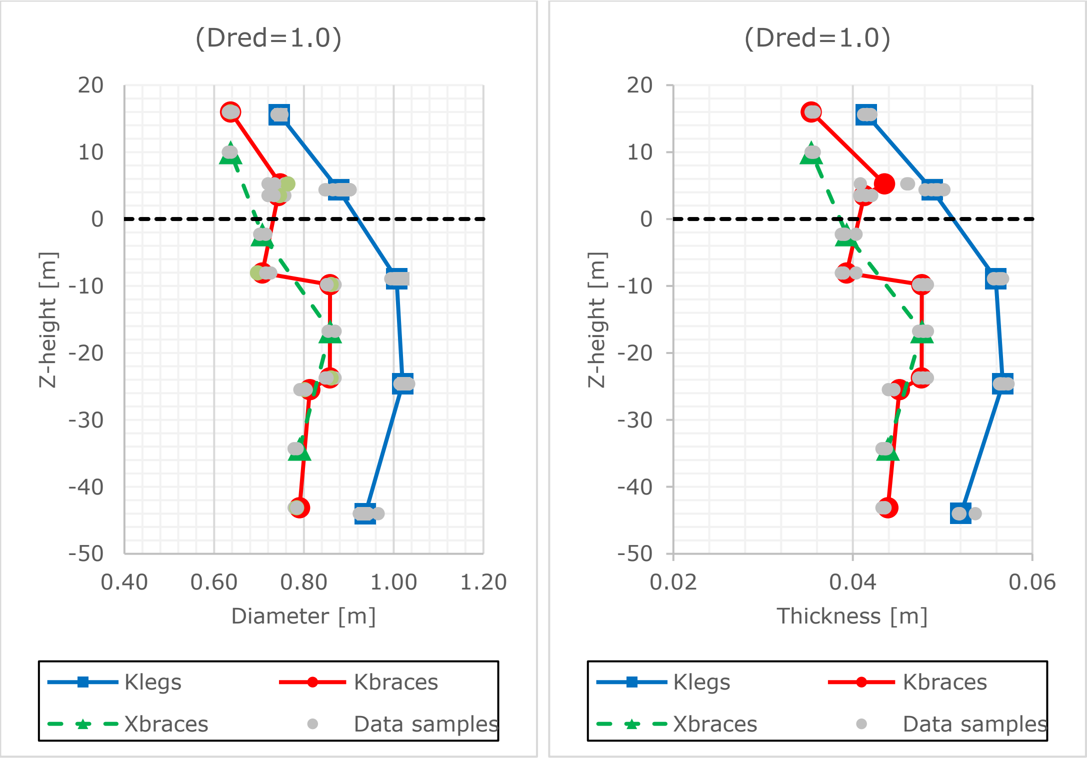


Figure 5.10: Mean dimensions of the final (optimum) designs for all the members over the tower height. Data samples of all optimized designs are provided as well (Dred=1.0).

Results for the maximum nonlinear constraint values at the final designs, as an indication of efficient member utilization (in terms of max nonlinear constraint values achieved), are given in Figure 5.11. Maximum utilization can be achieved if all the members have their nonlinear constraint values as close to the threshold (=1) as possible. Ideally all optimized members would have "max CON" values between 0.9-1.0. However, this is not possible as the geometrical constraints should also be satisfied at the optimum designs. This guides some members to have conservative dimensions with respect to the ULS, FLS and BUC constraints and so being away from the maximum utilization region.

Especially, Xbraces show very good utilization at every location and for both jacket configurations. At the same time Klegs show good utilization for the lower bays but as we move higher, the final dimensions are very conservative with respect to the nonlinear constraints. Finally, Kbraces show good utilization at bay 2 and at K-joint between bays 3 and 4 for the case of Dred=0.6, while for Dred=1.0, good utilization is achieved at the same K-joint and at upper bay 1.

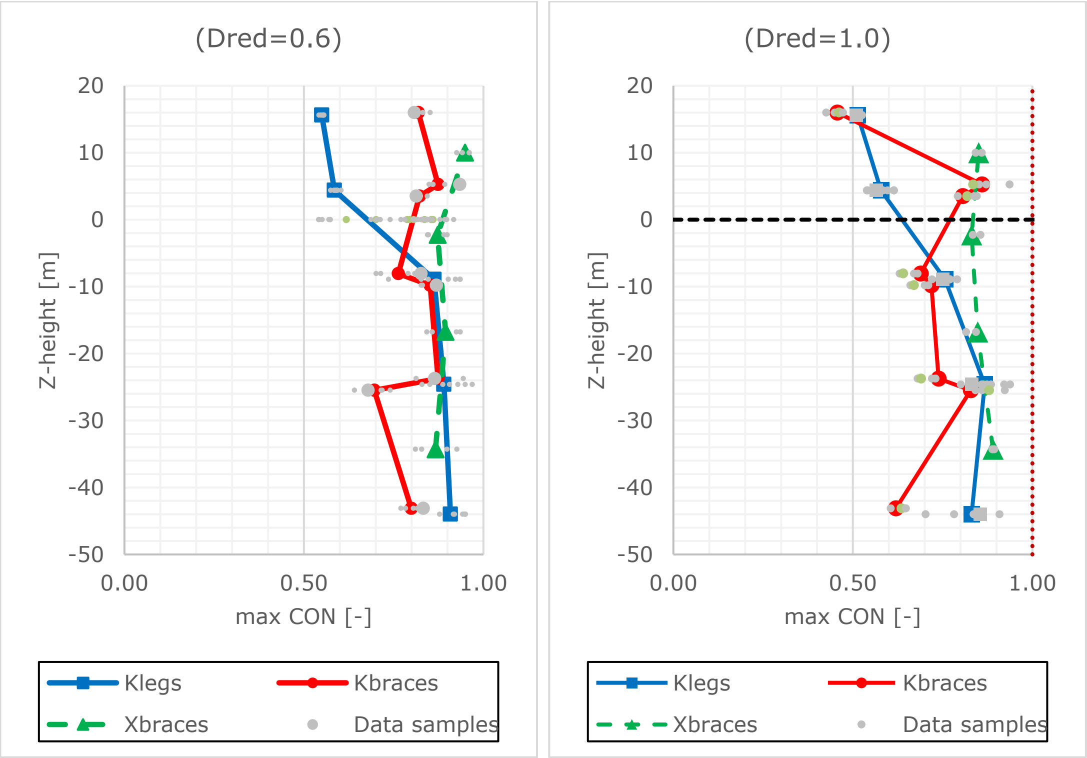


Figure 5.11: Mean values of the maximum nonlinear constraints of the final (optimum) designs for all the members over the tower height. Data samples of all optimized designs are provided as well.

5.5 A typical run

In this section, the performance of a selected run with Dred=0.6, is presented. The optimization progress is shown in Figure 5.12. The total number of iteration steps needed to find the optimum design was 30 (including the iteration '1' which is just the evaluation of the initial design) and convergence is achieved after only 5 full time domain analysis (including the first evaluation). The initial design is not a valid design (i.e. at least one of the constraints is exceeds the threshold value of 1). The full-time domain analyses are indicated by the vertical dotted lines within the figure.

A reduction in cost of 9.88*10^5 euro is achieved during the first internal sizing (iteration step 2-6). At this point it's good to mention that during the internal sizing process, the target value for the nonlinear constraint was set to 0.8 instead of the threshold 1.0 (see section 4.3.1). The purpose of this target is to guide the design towards the limit without exceeding it after an internal sizing step which is based on local assumptions. Applying this target gives redundancy to the optimized design in order to compensate for the possible deviation from the result of the following full time evaluation.

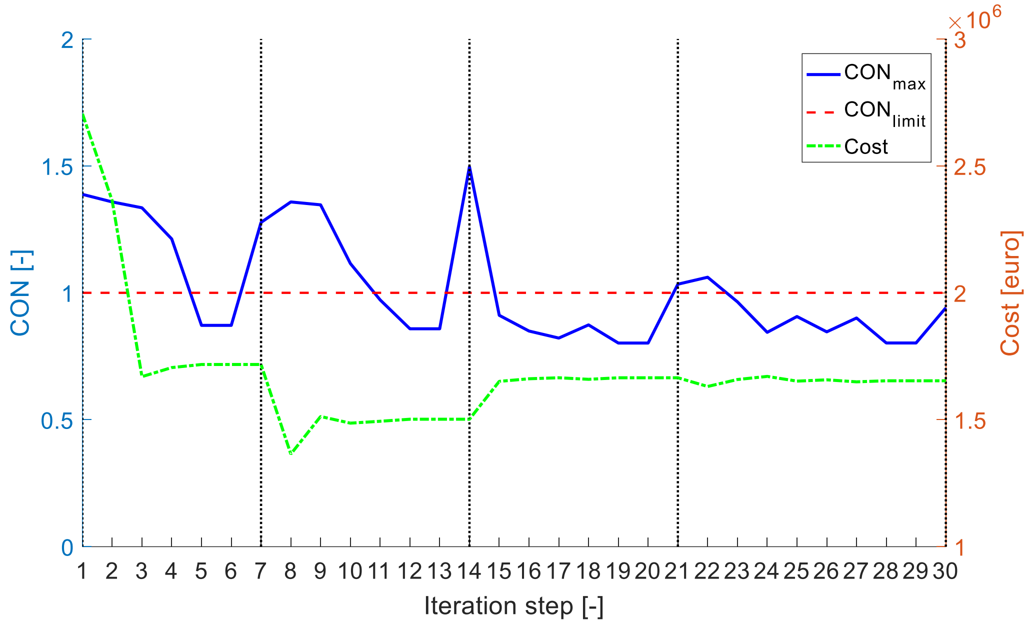


Figure 5.12: Maximal constraint value (CON_{max}) and cost of structure (Cost) during optimization progress of selected run. The vertical dotted lines stand for an iteration step with evaluation (full time domain analysis) performed.

The first two internal sizing processes (iteration steps 2-6 and 8-13), do not manage to predict very accurately the constraint performance and the control runs using the external finite element solver (iterations 7, 14) do not confirm the predicted values. This is the result of high cost function changes (implying also high dimension changes of the members) during these internal sizing runs. This is of course an expected behaviour due to the local approach assumptions, but despite this, these two first sizings manage to approach very well the final optimum design. This is obvious in the subsequent iterations where costs changes are lower and convergence while all constraint are fulfilled, is achieved only after two more internal sizings. It can finely be seen that the deviation of predicted (while sizing) and calculated (while evaluation) maximum constraint values (CON$_{\text{max}}$), decreases for smaller changes in cost function. This matter was already explained in more detail in section 5.2. and the following table illustrates this relationship for the selected run.

Table 5.2: Cost change and constraint deviation over the sizing process

	1st sizing	*2nd sizing*	*3rd sizing*	*4th sizing*
Cost change [%]	36.5	12.6	10.9	0.7
CON$_{\text{max}}$ deviation [%]	46.5	74.3	28.9	17.2

In the following figures the geometry dimensions as long as structural cost, structural weight and constraint values at each member of the selected run are presented for the initial (iteration 1) and the final (iteration 30) steps of the optimizations progress for the selected run.

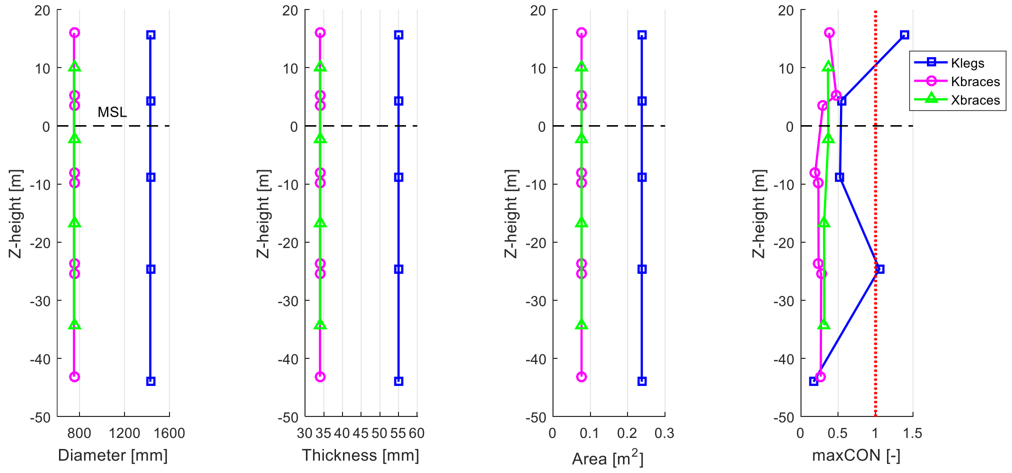


Figure 5.13: Initial (iteration step 1) design and maximum constraint values of the selected run.

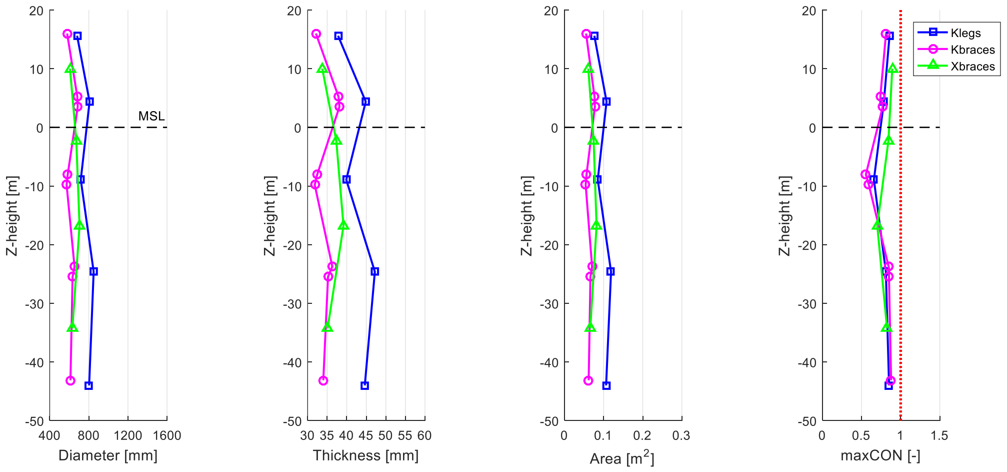


Figure 5.14: Final (iteration step 30) design and maximum constraint values of the selected run.

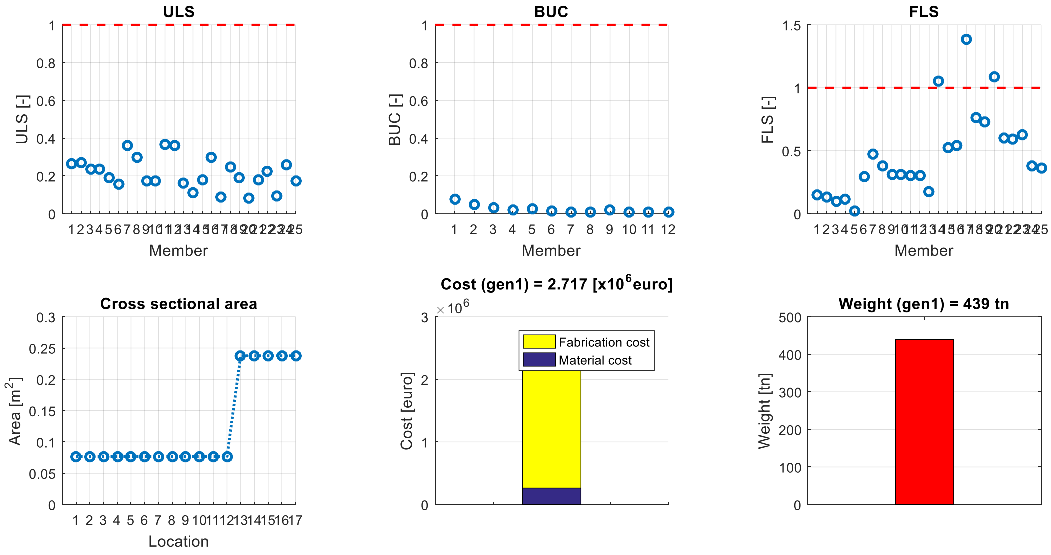


Figure 5.15: Initial (iteration step 1) constraint values, areas, cost and structural weight of the selected run.

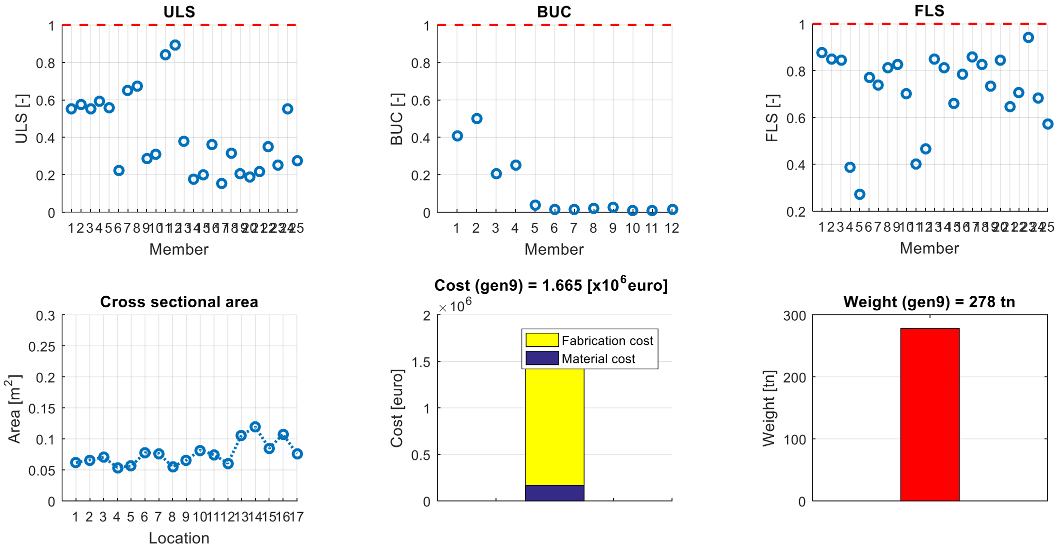


Figure 5.16: Final (iteration step 30) constraint values, areas, cost and structural weight of the selected run.

In the following three figures, the performance of the third internal sizing process (iteration steps 14-20) is presented. The dimensions (diameter, thickness and cross sectional area) of all the variables are presented at each internal sizing step. The blue lines correspond to the values at the beginning of the sizing (step 14) and the red lines correspond to the optimized designs at the end of the sizing (step 20). Furthermore, the maximum constraint values (also corresponding to maximum violation as the threshold is the 0) over the internal sizing steps is also illustrated at the top right subfigure (internal iterations 1-7 correspond to iteration steps 14-20 of the global iteration process). At this point it is good to mention again, that all the constraints as formulated in chapter 3, have a threshold value of zero as it is also obvious from figures Figure 5.17 and Figure 5.18. At Figure 5.12, the values for the constraint have been scaled up (add +1) just for presentation reasons and to match with the previously presented results.

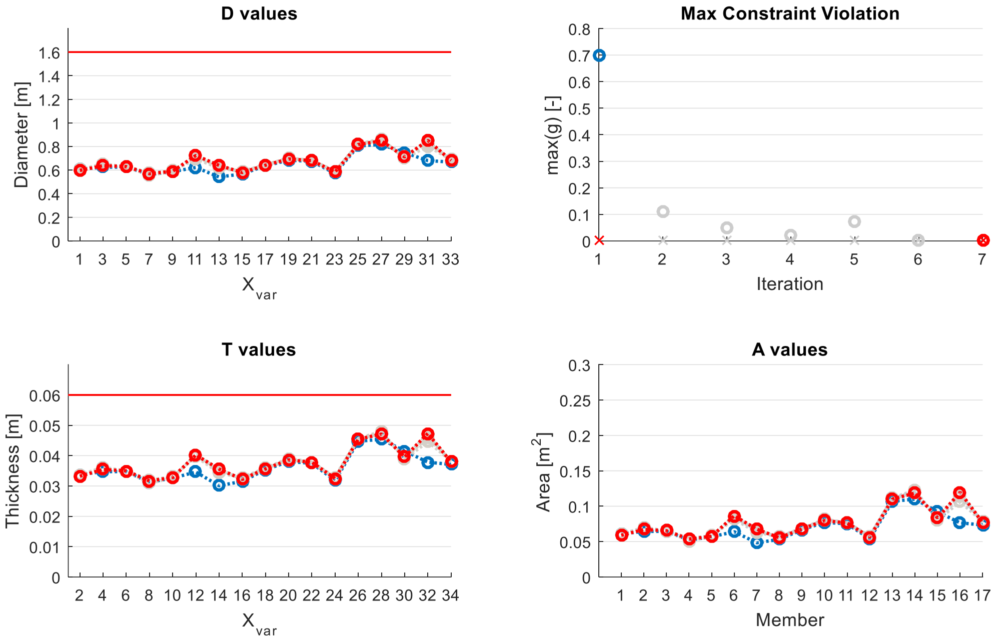


Figure 5.17: Dimensions of all members and maximum constraint values over the internal sizing steps

In addition, constraint values include the target value of the internal sizing process (equal to -0.2, see section 4.3.1). This means that a constraint value of 0.15 in figures Figure 5.17 and Figure 5.18 corresponds to a real constraint value of -0.05 calculated as:

$$\text{RealValue} = \text{PlotValue} - (\text{Real threshold} - \text{target}) = 0.15 - (0 - (- 0.2)) = -0.05$$

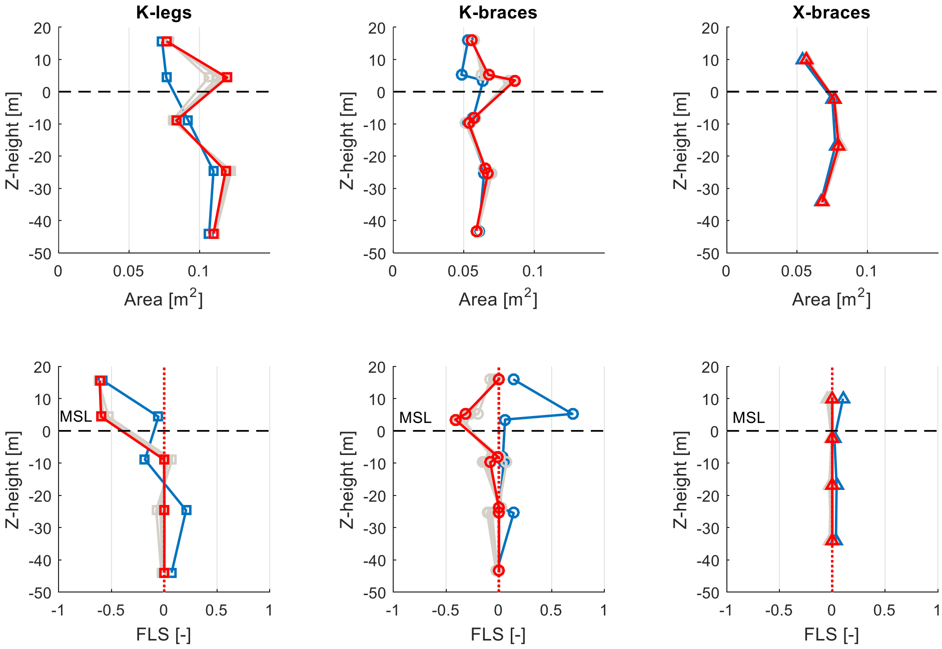


Figure 5.18: Final design (areas) of all members and the corresponding FLS values of the selected run.

Looking at Figure 5.18, we can see that the largest differences between initial and final dimensions occur for the Kbraces around the K-joint between bays 3 and 4. The constraint violations are higher at these locations and this explains the increase in their dimensions. Specifically, the FLS value is high for upper Kbrace at this location and ULS value (not shown here) is high for the lower Kbrace. At the end of the sizing process, the areas of these two members have increased by 30% and 45% respectively. At the same time, high increase of the Kleg dimensions at the same location is observed, but this is due to the geometrical constraints that must be satisfied. Constraint values are not exceeding the threshold for the Kleg at this location but the increase of dimensions in the attached Kbraces, forces the Kleg dimensions to increase as well as the diameter and thickness of the Klegs should be at least equal to the attached Kbraces (implied by constraint $g_3 - g_6$). At the same time the Xbraces present minor changes in their dimensions as they present small constraint violation only at the top bay. In general, members where the constraint values are very close or below the threshold of zero at the first step of the internal sizing, indicate very good prediction of the response from the previous internal sizing process (iteration steps 7-13) and therefore small deviation between those values and the full time evaluation that followed. In general, during all the runs tested, it was noticed that the Xbrace members exhibited lower deviations compared to the Klegs and Kbraces, with the Kbraces providing the higher values. Figure 5.19 illustrates the changes in dimensions of all the members during the third internal sizing (iteration steps 14-20) of the selected run. In this figure, except the detailed values for all dimensions, it is important to notice that diameters and the thicknesses follow the same trend when changing (i.e. a 30% increase in the diameter of a member at the end of the sizing process, is followed by a similar increase in the thickness of the same member).

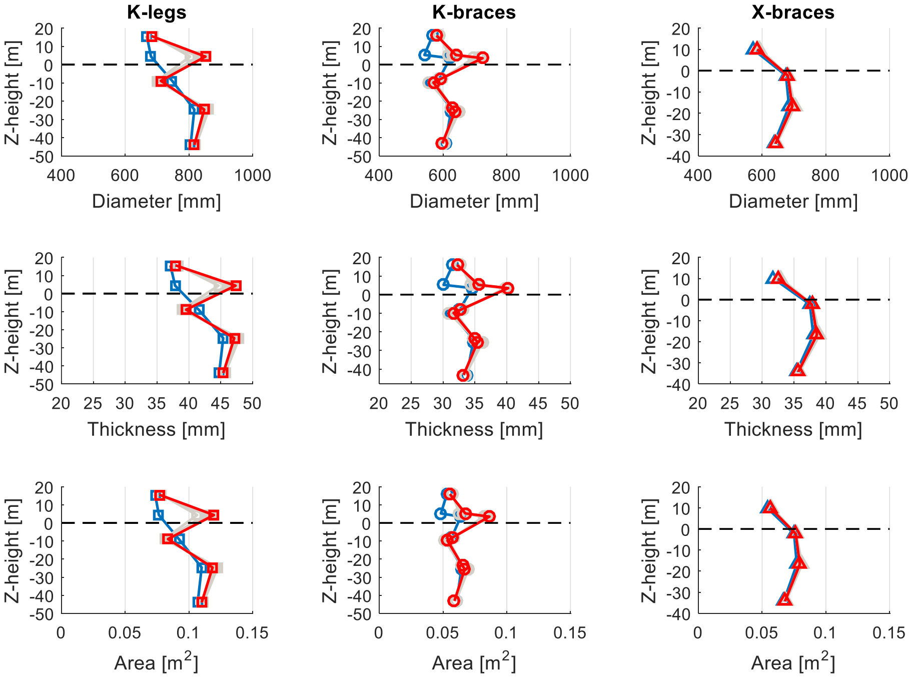


Figure 5.19: Dimensions (diameter, thickness, cross sectional area) of all members over a sizing process of the selected run.

5.6 The constraints g_9 and g_{10}

At this point it is of high importance to talk about the linear constraints g_9 and g_{10} (see section 3.5.1). These constraints were not initially included in the optimization formulation but their presence is crucial for the success of the optimization for the configuration of the jacket used in the current work as we will show below. As discussed in chapter 3, the selection of the design variables for this study, is based on the dimensions of the cross sections around the joints of the jacket. More specifically, the configuration with variables the Kleg, Kbrace and Xbrace sections, allows for three different cross sections between a K-joint (or Y-joint) and an X-joint; namely the Kbrace, the Xbrace and the brace between them which depends on the Xbrace dimensions through the Dred factor. According to this, the Kbrace and the Xbrace sections attached to the ends of a brace, are totally independent from each other, allowing the Kbraces to have smaller or larger dimensions from the brace attached to it (as the brace dimensions are defined from the Xbrace dimensions).

However, as it was detected during several tests conducted for the analysis of the proposed method, when the Kbrace dimensions become less than the braces attached to them, high bending moments start to develop at both ends (Kbrace and Xbrace sections) of the braces which lead to very high stresses and high FLS and ULS values. This high values as a response during the full time domain analysis, is a structural behaviour that the optimization process not only cannot "predict" but also cannot overcome. The reason for this is that when such bending moments start to develop, the internal sizing increases the dimensions of the corresponding members (Kbraces and Xbraces) in order to deal with the high constraint violations. However, at the next iteration the problem remains as what actually causes the problematic behaviour is the ratio between the Kbrace and Xbrace dimensions and not the dimensions themselves. As a result, if at any moment, a design requires the brace to have larger dimensions of the Kbraces attached to it, then the optimization guides the design towards very large dimensions after many iterations which at the end may still not fulfill the constraints.

This behaviour was detected when the reduction design factor was set to 1.0. For the configuration with Dred=0.6 (meaning that the brace has dimensions 40% less than the Xbrace attached to it), the Kbrace could have dimensions up to 40% less than the corresponding Xbrace. In fact, such case was never detected as Kbraces could require to have dimensions of up to 30% lower than the Xbraces. However, when Dred=1.0 (meaning that brace is equal to Xbrace), the aforementioned behaviour was encountered, as even when Kbrace had dimensions just 10% less than the Xbrace, the bending moments were becoming very important.

The solution to the described problem was found to be the implementation of two new linear constraints, the g_9 and g_{10}. These constraints do not allow the ratio of a Kbrace dimension to the attached brace dimension to become less than one. Figure 5.20, illustrates the performance of a run with Dred=1.0 and with the g_9 and g_{10} implemented but allowing the Kbraces to have 20% lower dimensions than the braces attached to them. In the figure, the Fedem evaluations are represented by the odd number of iteration steps and are indicated with black dotted vertical lines while the internal sizing process are represented by the even number of iteration steps (each internal sizing process consists of several internal iterations not exhibited here for clarity reasons).

In this case the optimization finally converges, but it is important to pay attention at two points. Firstly, the number of full time domain analyses needed to achieve the final design is eight (including the first evaluation at the beginning). This number is higher than the average number of evaluations needed for a similar design (Dred=1.0), which is 5.2 evaluations, but it is also higher than the maximum Fedem evaluations ever needed for an optimization with Dred=1, which is 7. The second interesting thing of this figure is that the maximum constraint value remains almost constant at the three Fedem evaluations at steps 5, 7 and 9, while the cost of the structure (and therefore the dimensions of the members) is constantly increasing during steps 6 and 8.

To conclude, when allowing Kbrace to have dimensions lower than the attached braces, convergence is delayed or in many cases not achieved at all. Therefore, the proposed geometry constraints g_9 and g_{10} are strongly recommended to be implemented when we have to optimize jacket designs which allow for different cross sections along a member (in this case a brace).

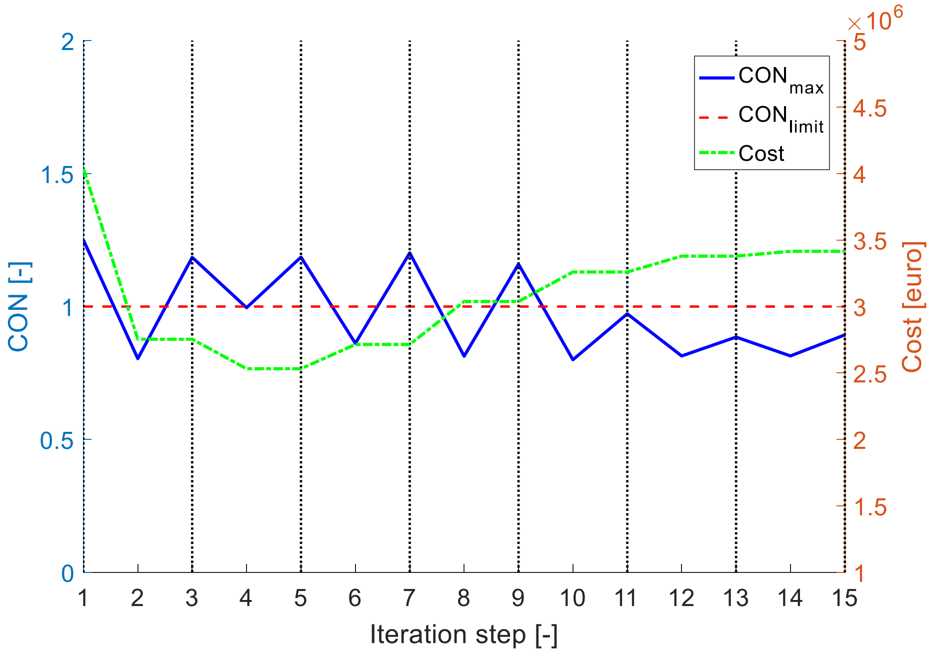


Figure 5.20: Overview of selected run with Dred=1.0 and constraints g_9 and g_{10} allowing for Kbrace dimensions 10% lower than the attached braces.

5.7 Natural frequency

Despite not included as a constraint in the optimization process, the resulting eigenfrequency of the structure was investigated for various all the runs performed, as an important characteristic in structural design. As structural design is changing over iterations, the eigenfrequencies of the support structure are changing as well. The common intention of a support structure design for wind turbines requires that the 1st global eigenfrequency of the whole structure (support structure + wind turbine) should lie between the potential resonance ranges of the rotational frequency of the rotor (1P region) and the tower passage frequency of the blades (3P region) as mentioned in section 3.5.4. For the case of the NREL 5 MW wind turbine used in this study, the 1P and 3P regions are 0.115 – 0.202 Hz and 0.345 – 0.600 Hz respectively. The following tables provides the average values and the ranges of the 1st and 3rd eigenfrequency values of the jacket, measured for all the tested designs.

Table 5.3: Eigenfrequencies of tested designs

	1st Eigfreq.	*Mean 1st Eigfreq.*	*3rd Eigfreq.*	*Mean 3rd Eigfreq.*
Dred = 0.6	0.255-0.300	0.277	0.553-0.621	0.612
Dred = 1.0	0.311-0.343	0.331	0.616-0.628	0.625

As it can be seen from the results above, no eigenfrequency for any design falls into the 1P region, and actually they are not even close to this area. However, several designs have their 3rd eigenfrequency inside the 3P region or very close to the higher limit of the 3P region for the configuration of the jacket with Dred=0.6. At the same time, when Dred=1.0, the first eigenfrequency may be very close to the 3P region. Despite no resonance phenomena were observed during the tests, these results indicate that when optimizing jacket support structures, we should be aware of possible resonance incidents (see also section 3.5.4).

6 CONCLUSION

6.1 Discussion

The gradient-based optimization with the local approach presented here, has its main objective in reducing computation time compared to an optimization approach with a full finite element analysis for each iteration step. Despite the local assumptions, the predicted results during the internal sizing process were in most cases found to not deviate more than 25% from the full time domain analysis that follows each such process. The optimization process makes use of this information, by setting the threshold value with an offset to the target value. As constraint results are predicted and not accurately calculated in the sizing process, the offset increases the probability of achieving acceptable (below the threshold) results after the full time analysis. By this, large structural changes can be obtained with reasonable accuracy without the need for a costly time domain analysis at each step.

The achieved accuracy of the sizing process should also be considered with respect to the simulation error caused by the variability of input loading and the length of the time series simulated. In the present study, time series with length of 210s were used for the time domain simulations in order to prove the functionality of the method. Use of longer time series may improve further the accuracy of the method.

The level of optimization of a design can be evaluated by its used capacity (ability to fulfill the nonlinear constraints, ULS, FLS and BUC) under different load conditions. The approach here is aiming for a cost reduction of the structure (so smaller dimensions) and at the same time the fulfillment of all the nonlinear constraints just below their threshold values. However, full utilization is not possible for all members as the geometry constraints that should also be fulfilled, prevent the optimization of proceeding towards a full utilization at some locations.

Despite the behavioral constraints (like ULS, FLS, BUC) are more important for evaluating the performance of the optimization, we should not forget the geometrical constraints. They should be carefully selected and as discussed in section 5.6, they can have a severe impact on the success or not of the optimization algorithm.

Moreover, the selection of the variable values during the internal sizing process was based on the assumption that increased cross sectional areas reduce the nonlinear constraint values of the corresponding member. This same idea was also used to calculate analytically the sensitivities of the fatigue damage constraints.

The resulting final designs with varying diameter and thickness values over the tower height prove the adaptation of the structure to the variant load conditions from bottom to top of the structure. Comparing the shape of diameter and thickness curves over the tower height (Figure 5.9 and Figure 5.10) with results obtained by Schafhirt et al. [2], a good match can be seen for corresponding tower heights.

6.2 Conclusion

This work was initiated by a literature study on the topics of structural optimization, gradient based methods, sensitivity analysis and local optimization approach.

Subsequently, the process of finding a good objective function, along with all the corresponding parameters, was started. An equation considering both material and fabrication costs for the support structure was chosen at the end as a realistic optimization measure of a jacket support structure.

Next, an optimization framework was developed using the local optimization approach, as it is described in chapter 4. The principle of decomposition was used in order to implement these algorithms. The optimization framework was applied to a case study of the OC4 offshore jacket substructure model.

Finally, a substantial amount of work was performed to implement the optimization algorithm and all relevant constraints, and incorporate this into the existing pre- and post-processing. This phase did of course also include quite a lot of debugging.

The objective of the thesis is met. A gradient-based procedure for the optimization of a jacket support structure was developed. This method uses an analytical sensitivity analysis with the local approach in order to achieve global sizing optimization of the jacket. Optimized designs are produced by applying the method to several jacket configurations. In addition, the objective function for this study, was selected to be the structural cost. A cost function including material and fabrication costs was constructed in order to optimize in terms that are closer to the desired goals (i.e. cost savings) when designing support structures. The use of this function also proved to work well.

The success of the applied optimization approach can be seen in three areas. Firstly, the assumptions made by the use of the local approach lead to reasonable results. Despite these assumptions, the accuracy of the method is proved in chapter 5. Similar final designs are found under the same conditions. Secondly, the analytical gradients calculated, managed to guide the internal sizing process towards the optimized designs successfully. Thirdly, the efficiency of the method was also investigated. Convergence was achieved after a few full time evaluations and from several initial points (designs). The method needs just 5 full time domain analyses in average, in order to find optimum designs. This process is for sure more efficient than trying to optimize by running a full time domain analysis at every step. However as presented in Table 5.1, the process is more time consuming than a method that uses a simplified algorithm for the optimization instead of the analytical gradient approach followed here.

The adjustment of several optimization parameters and constraints, such as the termination criteria used, can give further opportunities for the development of this approach and increase its efficiency. Finally, it is worth mentioning that the proposed optimization framework is not restricted to jacket support structures only, but it can be applied to other offshore wind turbine support structures (e.g. monopiles and tripods) which present a lower level of complexity and accuracy of the method may be increased.

6.3 Future work

Although this study has illustrated and proved some important aspects of the use of the gradient based optimization in conjunction with the local approach in structural design of offshore jacket support structures, there are still unanswered questions waiting for additional research. Here are some suggestions for future work:

1. Including more parameters as design variables in the optimization process. Number of legs, number of sections, variable section height, bottom leg distance and top leg distance are all variables with major impact on structural performance. It would be interesting to see if these variables can be included in the same optimization process, how that affects the performance of the optimization and if further minimization of the objective function is possible.

2. Account for more constraints during the optimization process. Natural frequency, combined stress actions and other constraints recommended in design standards for offshore structures, could enable a more accurate analysis.

3. Calculate and include the SCF sensitivities (gradients) in the optimization process. This will make the internal sizing process faster and more accurate. Deviations between internal sizing and solver evaluations will decrease.

4. Define the design driving constraints and use this information to make the optimization process more efficient.

5. Include extra costs in the cost function except the manufacturing and material costs. Transportation and installation costs account for a big portion of the total cost for an offshore support structure and therefore they should be considered especially when topology of the structure is subject to optimization.

6. Realistic optimization should also account for discrete variables, and possibilities for using the same dimensions for several members. A concern with doing optimization at a level of detail this high is the increased complexity in production. Unless adjustments are done after the optimization, there will be a large number of members where almost all have different dimensions. This will complicate the logistics related to production. Some economy of scale is of course possible for large wind farms, given that many of the turbines are optimized for the same site conditions. But with more efficient optimization techniques it is possible to optimize each individual structure, at least if there are differences in water depth, etc. This could again result in every single structure being unique, with its own dimensions on every member.

7. During this study, all optimization has been done based on a few load cases. Real world development does, however, require testing with hundreds of load cases, if design standard recommendations are to be followed. Further work can therefore investigate how the optimization procedure can coexist and interact with this large number of load cases, resulting in a design that is in guaranteed to comply with all requirements.

REFERENCES

[1] B. B. P. Vattenfall, 24 March 2011. [Online]. Available: https://corporate.vattenfall.com/press-and-media/image-bank/.

[2] S. Schafhirt, D. Zwick and M. Muskulus, "Two-stage Local Optimization of Lattice Type Support Structures for Offshore Wind Turbines," *Ocean Engineering,* 2015.

[3] F. Vorpahl, W. Popko and D. Kaufer, "Description of a Basic Model of the "Upwind Reference Jacket" for Code Comparison in the OC4 Project under IEA Wind Annex 30," Franhofer IWES, Bremerhaven, Germany, 2013.

[4] J. Jonkman, S. Butterfield, W. Musial and G. Scott, "Definition of a 5 MW Reference Wind Turbine for Offshore System Development," Tech. Rep. NREL/TP-500-38060, National Renewable Energy Laboratory, U.S., 2009.

[5] K. K. Choi and N.-H. Kim, Structural Sensitivity Analysis and Optimization 1: Linear Systems, Springer, 2005a.

[6] J. Nocedal and S. J. Wright, Numerical Optimization Second Edition, Springer, 2006.

[7] D. T. Venetsanos, *Development of methods for Topology and Shape Optimization of Mechanical Structures, PhD Thesis,* Athens, 2010.

[8] J. S. Arora, Introduction to Optimum Design Second Edition, San Diego, California, USA: Elsevier Inc., 2004.

[9] G. N. Vanderplaats, Multidiscipline Design Optimization, Colorado Springs: Vanderplaats Research and Development Inc., 2007.

[10] R. T. Haftka and Z. Gurdal, Elements of Structural Optimization (Third revised and expanded edition), Dordrecht, The Netherlands: Kluwer Academic Publishers, 1992.

[11] J. A. Snyman, Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and new Gradient Based Algorithms, Boston: Springer Science & Buisiness Media, Inc, 2005.

[12] G. N. Vanderplaats, "Structural Optimization for Statics, Dynamics and Beyond," *Journal of the Brazilian Society of Mechanical Sciences and Engineering, vol. 28, no.3,* pp. 316-322, 2006.

[13] G. Venter, "Review of Optimization Techniques," *Encyclopedia of Aerospace Engineering,* 2010.

[14] J. S. Arora, Introduction to Optimum Design (3rd Edition), Elsevier Inc., 2012.

[15] A. Ravindran, K. M. Ragsdell and G. v. Reklaitis, Engineering Optimization Methods and Applications (2nd ed.), Hoboken, New Jersey: John Wiley & Sons, 2006.

[16] G. Zoutendijk, Methods of Feasible Directions, Amsterdam: Elsevier, 1960.

[17] J. Nocedal and S. J. Wright, Numerical Optimization (2nd edition), Springer, 1999.

[18] T. Fischer, "UpWind WPA Executive Summary," Project UpWind , 2011.

[19] O. Y. Minambres, "Assessment of Current Offshore Wind Support Structures Concepts – Challenges and Technological Requirements by 2020," 2012.

[20] M. Muskulus and S. Schafhirt, "Design Optimization of Wind Support Structures - A Review," *Journal of Ocean and Wind Energy (ISSN 2310-3604),* 2014.

[21] H. Molde, "Simulation-Based Optimization of LAttice Support Structures for Offshore Wind Energy," Norwegian University of Science and Technology, Department of Civil and transport Engineering, Trondheim, 2012.

[22] DNV, Fatigue Design of Offshore Steel Structures, Recommended Practice DNV-RP-C203, Norway: Det Norske Veritas AS, 2012.

[23] NORSOK, NORSOK Standard N-004: Design of Steel Structures, Rev 3, Lysaker, Norway: Standards Norway, 2013.

[24] T. Fischer, W. de Vries and B. Schmidt, "Upwind WP4 Design Basis," Project UpWind, 2010.

[25] D. Zwick, M. Muskulus and G. Moe, "Iterative Optimization Approach for the Design of Full-Height Lattice Towers for Offshore Wind Turbines," *Energy procedia, Vol 24,* pp. 297-304, January 2012.

[26] K.-H. Chew, T. Kang, N. E.Y.K. and M. Muskulus, "Optimization of Offshore Wind Turbine Support Structures Using an Analytical Gradient-Based Method," in *12th Deep Sea Offshore Wind R&D Conference, EERA Deep Wind'2015*, 2015.

[27] L. MEPS, May 2015. [Online]. Available: http://www.meps.co.uk/NordicPrice.html.

[28] 18 August 2016. [Online]. Available: https://www.quandl.com/collections/markets/industrial-metals.

[29] 18 August 2016. [Online]. Available: http://www.tradingeconomics.com/commodity/steel.

[30] J. Farkas and K. Jarmai, Design and Optimization of Metal Structures, Chichester, UK: Horwood Publishing Limited, 2008.

[31] 2015. [Online]. Available: http://ec.europa.eu/eurostat/web/labour-market/labour-costs/database.

[32] IEC-61400-1, International Standard, Wind Turbines - Part 1: Design Requirements, Geneva: International Electrotechnical commission (IEC), 2005.

[33] J. Farkas and K. Jarmai, Analysis and Optimum Design of Metal Structures, Rotterdam: A.A. Balkema, 1997.

[34] J. Farkas and K. Jarmai, Optimum Design of Steel Structures, Berlin, Heidelberg: Springer-Verlag, 2013.

[35] G. Domokos and P. Holmes, "Euler's Problem, Euler's Method and the standard Map; or the Discrete charm of Buckling," *Journal of Nonlinear Science 3,* pp. 109-151, 1993.

[36] K.-H. Chew, M. Muskulus, Y. E. Ng and K. Tai, "Structural Optimization and Parametric Study of Offshore Wind Turbine Jacket Substructure," in *Twenty-third International Offshore and Polar Engineering*, Anchorage, Alaska, 2013.

[37] MathWorks, Optimization Toolbox User's Guide, Natick, MA, USA: The MathWorks inc, 2015.

[38] R. Fletcher, Practical Methods of Optimization (Second Edition), John Wiley & Sons Ltd, 1987.

[39] P. E. Gill, M. Walter and M. H. Wright, Practical Optimization, San Diego CA: Academic Press ,Inc., 1981.

[40] P. E. Gill, M. W, M. A. Saunders and M. H. Wright, "Procedures for Optimization Problems with a Mixture of Bounds and General Linear Constraints," *ACM Trans. Math. Software Vol. 10,* pp. 282-298, 1984.

[41] P. E. Gill, W. Murray and M. H. Wright, Numerical Linear Algebra and Optimization Vol. 1, Addison Wesley, 1991.

[42] S. P. Han, "A Globally Convergent Method for Nonlinear Programming," *J. Optimization Theory and Applications, Vol 22,* p. 297, 1977.

[43] M. J. D. Powell, A Fast Algorithm for Nonlinearly Constrained Optimization Calculations Numerical Analysis, vol. 630, Lecture Notes in Mathematics, Springer Verlag, 1978.

[44] R. Fletcher, "The Sequential Quadratic Programming Method," 2007.

[45] J. V. D. Tempel, "Design of Support structures for Offshore Wind Turbines," DUWIND, Delft, 2006.

[46] Siemens PLM Software, "Design Sensitivity and Optimization User's Guide for NX Nastran," 2014. [Online]. Available: https://docs.plm.automation.siemens.com/tdoc/nxnastran/10/help/#uid:index. [Accessed 14 November 2015].

[47] K. Sandal, "Improved Design of the Support Structure for the NOWITECH 10MW Reference Turbine," Norwegian University of Science and Technology, Trondheim, 2014.

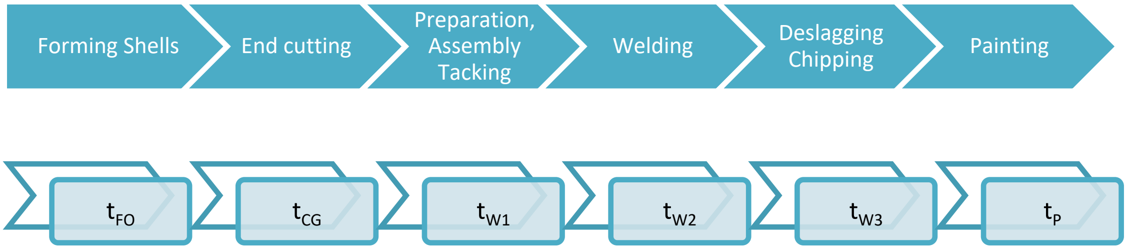
[48] K.-H. Chew, K. Tai, E. Ng and M. Muskulus, "Optimization of offshore wind turbine support structures using analytical gradient-based method," ResearchGate, Trondheim, Norway, March 2015.

[49] T. Coleman, M. Branch and G. A, Optimization toolbox for Use with MATLAB: User's Guide, 7th Ed., The MathWorks, Inc., 2014.

[50] G.-J. Park, Analytic Methods for Design Practice, London: Springer, 2007.

APPENDIX A – Fabrication times

Fabrication times for manufacturing of a truss steel structure considered in the present work [30]:



1. Forming plates into shell segments

$$t_{FO} = \Theta e^{\mu} \qquad [\text{min}]$$

Where:

$$\Theta = 3 \qquad [-]$$

$$\mu = 6.86 - 4.53\mathrm{T}^{-0.5} + 0.0095\mathrm{D}^{0.5} \quad D \leq 3000mm \qquad [\text{min}]$$

2. Hand cutting and machine grinding of strut end for tubular structures

$$t_{CG} = \Theta_{dc}\left(4.54 + 0.4229T^{2}\right)L_{c} \qquad [\text{min}/\,mm]$$

CAUTION: $\mathrm{L_c}$ is in meters while T is in mm !!!

Where:

$$\Theta_{dc} = 3 \qquad [-]$$

θ is the angle between the two members connected [deg]

$L_c = \dfrac{2\pi}{\sin\theta}D$ is the cut length $[m]$

3. Preparation, assembly, tacking

$$t_{W1} = C_1 \Theta_{dw} \sqrt{k\rho AL}$$ [min]

Where:

$C_1 = 1$

$\Theta_{dw} = 2,3$ (for tubes [33]) [–]

$k = 1$ (number of elements to be assembled) [–]

4. Real welding time

$$t_{W2} = C_2 a_w^n L_w$$ [min]

Where:

C_2 is a constant depending on the welding technology $[\text{min}/mm^2]$

a_w is weld size $[mm]$

$L_w = \dfrac{2\pi}{\sin\theta} D$ is weld length $[mm]$

5. Changing electrode, deslagging and chipping

$$t_{W3} = 0.3 t_{W2}$$ [min]

6. Painting

$$t_P = \Theta_{dp} \left(a_{gc} + a_{tc}\right) A_S$$ [min]

Where:

$\Theta_{dp} = 1,2,3$ (=3 for complicated structures like truss) [–]

$a_{gc} = 3 * 10^{-6}$ $[\text{min}/mm^2]$

$a_{gc} = 4.15 * 10^{-6}$ $[\text{min}/mm^2]$

$A_S = \pi LD$ (surface to be painted) $[mm^2]$

Table A.1: Welding times t_{w2} (min/mm) in the function of weld size aw (mm) for longitudinal fillet welds, downhand position.

Welding technology	a_w [mm]	$10^3 t_{W2} = 10^3 C_2 a_w^2$
SMAW	0-15	0.7889 a_w^2
SMAW HR	0-15	0.5390 a_w^2
GMAW-C	0-15	0.3394 a_w^2
GMAW-M	0-15	0.3258 a_w^2
FCAW	0-15	0.2302 a_w^2
FCAW-MC	0-15	0.4520 a_w^2
SSFCAW (ISW)	0-15	0.2090 a_w^2
SAW	0-15	0.2349 a_w^2

Table A.2: Welding times t_{w2} (min/mm) in the function of weld size aw (mm) for longitudinal 1/2 V and V butt welds downhand position.

			æ V butt welds		V butt welds	
Welding technology	a_w [mm]		$10^3 t_{W2} = 10^3 C_2 a_w^2$		$10^3 t_{W2} = 10^3 C_2 a_w^2$	
SMAW	4-6	6-15	3.13 a_w	0.5214 a_w^2	2.7 a_w	0.45 a_w^2
SMAW HR	4-6	6-15	2.14 a_w	0.3567 a_w^2	1.8462 a_w	0.3077 a_w^2
GMAW-C	4-15		0.2245 a_w^2		0.1939 a_w^2	
GMAW-M	4-15		0.2157 a_w^2		0.1861 a_w^2	
FCAW	4-15		0.1520 a_w^2		0.1311 a_w^2	
FCAW-MC	4-15		0.2993 a_w^2		0.2582 a_w^2	
SSFCAW (ISW)	4-15		0.1384 a_w^2		0.1194 a_w^2	
SAW	4-15		0.1559 a_w^2		0.1346 a_w^2	

Table A.3: Welding times t_{w2} (min/mm) in the function of weld size aw (mm) for longitudinal K and X butt welds downhand position.

		K butt welds	X butt welds
Welding technology	a_w [mm]	$10^3 t_{W2} = 10^3 C_2 a_w^n$	$10^3 t_{W2} = 10^3 C_2 a_w^n$
SMAW	10-40	$0.3539\,a_w^{1.93}$	$0.3451\,a_w^{1.9}$
SMAW HR	10-40	$0.2419\,a_w^{1.93}$	$0.2363\,a_w^{1.9}$
GMAW-C	10-40	$0.1520\,a_w^{1.94}$	$0.1496\,a_w^{1.9}$
GMAW-M	10-40	$0.1462\,a_w^{1.94}$	$0.1433\,a_w^{1.9}$
FCAW	10-40	$0.1032\,a_w^{1.94}$	$0.1013\,a_w^{1.9}$
FCAW-MC	10-40	$0.2030\,a_w^{1.94}$	$0.1987\,a_w^{1.9}$
SSFCAW (ISW)	10-40	$0.0937\,a_w^{1.94}$	$0.0924\,a_w^{1.9}$
SAW	10-40	$0.1053\,a_w^{1.94}$	$0.1033\,a_w^{1.9}$

Welding Technologies:

SMAW = Shielded Metal Arc Welding

SMAW HR = Shielded Metal Arc Welding High Recovery

GMAW-C = Gas Metal Arc Welding with CO2

GMAW-M = Gas Metal Arc Welding with Mixed Gas

FCAW = Flux Cored Arc Welding

FCAW-MC = Metal Cored Arc Welding

SSFCAW (ISW) = Self Shielded Flux Cored Arc Welding

SAW = Submerged Arc Welding

GTAW = Gas Tungsten Arc Welding.

APPENDIX B – Parameter formulation

Parameters of a tubular member (D=diameter, T=thickness, L=length, E=Young's modulus).

$\sigma_{Sd} = \frac{N_{Sd}}{A} + \frac{M_{y,Sd}^{2} + M_{z,Sd}^{2}}{W}$	Design cross sectional stress.	[N/m^2]	(B.1.1)
$\sigma_{Rd} = \sigma_{yield}$	Design resistance for yield strength	[N/m^2]	(B.1.2)
$A = \pi\left(DT - T^{2}\right)$	Cross sectional area of hollow cylinder.	[m^2]	(B.1.3)
$I = \frac{\pi}{64}\left(D^{4} - \left(D - 2T\right)^{4}\right)$	Area moment of inertia.	[m^4]	(B.1.4)
$W = \frac{2}{D} I$	Elastic section modulus (a measure of the flexural strength of the beam).	[m^3]	(B.1.5)
$N_{Rd} = A\sigma_{yield}$	Design resistance for axial tension	[N]	(B.1.6)
$M_{Rd} = W\sigma_{yield}$	Design resistance for bending	[Nm]	(B.1.7)
$N_{E} = \frac{n \cdot \pi^{2} \cdot E \cdot I}{L^{2}}$	Euler buckling design resistance (n=4 was used here since fixed-fixed conditions assumed for all members).	[N]	(B.1.8)

APPENDIX C – Matlab scripts

In this appendix, the main scripts used for the optimization process are presented.

C1 – Internal sizing function

Below is presented the internal sizing step function that makes use of the embedded “fmincon” function from the Optimization Toolbox of Matlab. This function is the main script for internal sizing step. It sets all the parameters, lower bounds, upper bounds, tolerances, objective and constraint functions and at the end calls the optimization algorithm for optimization.

```
function  DIM =
INTERNAL_SIZING_main(FP,gen,JAC,DIM,COST,ULS,FLS,BUC,THE,LEN,SCF,tim
)

%******************************************************************
%
%  JACKET DESIGN FOR MINIMUM COST USING 'fmincon' FROM OPTIMIZATION
%  TOOLBOX (analytic expressions for gradients provided)
%
%******************************************************************
% written by Nikolaos Xyloudis, 09/2016


% SOLVE THE PROBLEM
%   Minimize:  cost_FUNCTION
% subject to:  calcLINEAR_CONSTRAINTS
%              calcNONLINEAR_CONSTRAINTS
%              Lower (Lb) & Upper (Ub) bounds on variables

%---INPUT-----------------------------------------------------
%   FP        - FEDEM and FATIGUE parameter
%   gen       - number of generation
%   fed       - counter of Fedem runs
%   JAC       - general jacket parameters
%   DIM       - bay specific jacket parameters
%   COST      - cost function value
%   ULS       - ultimate load constraint value
%   FLS       - fatigue damage cosntraint value
%   BUC       - buckling constraint value
%   THE       - structural angle theta for SCF calculation
%   LEN       - length of legs and braces
%   tim       - identification of time series file to be used

%---OUTPUT-----------------------------------------------------
%   DIM       - updated DIM matrix after the internal sizing process

% copy DIM and THE matrix to new generation
    DIM{gen,1} = DIM{gen-1,1};
    THE{gen,1} = THE{gen-1,1};
    LEN{gen,1} = LEN{gen-1,1};
```

```
%% (1) DEFINE INITIAL DESIGN XO (it can be scalar, vector or matrix)
%
%Even if you pass an initial point X0 as a matrix, solvers pass the
%current point X0 as a column vector to linear constraints.
%XO here will be a [M x N] = [1 x 34] where X0 = [D1 T1 D2 T2 D3...]
%
%      17   = number of locations we check in the structure
%      (5 K-stubs + 8 brace stubs(4 upper + 4 lower) + 4 X-stubs)
%      17x2(D,T) = 34 is the total number of design variables
%      X(1,1:16)    - Kbraces
%      X(1,17:24)   - Xbraces
%      X(1,25:34)   - Klegs

       X0      = DIMtoX(gen,DIM); % initial design [1 x 34]
       n_var   = numel(X0);       % total number of variables
       MaxViol = 7e-2;


%% (2) SET BOUNDS FOR VARIABLES
%      for X0 = [M x N] = MxN = Z variables, Lb and Ub must be
vectors
%      like Lb,Ub = [Z x 1].

       upper = [1.600; 0.060]; % D <= 1.6,  T <= 0.06 [m]
       lower = [0.400; 0.020]; % D >= 0.4,  T >= 0.02 [m]
%        Ub = repmat(upper, size(X0,2)/2, 1); % [34 x 1]
%        Lb = repmat(lower, size(X0,2)/2, 1); % [34 x 1]


%-----SET LIMITATION TO VARIABLE CHANGE (step size)-----------------
  factor_D = 60;     % allowable D change per internal sizing [%]
  factor_T = 60;     % allowable T change per internal sizing [%]

        for j=1:2:33
            Ub(j)   = min(upper(1),(1+factor_D/100)*X0(j));
            Ub(j+1) = min(upper(2),(1+factor_T/100)*X0(j+1));
            Lb(j)   = max(lower(1),(1-factor_D/100)*X0(j));
            Lb(j+1) = max(lower(2),(1-factor_T/100)*X0(j+1));
        end


%% (3) DEFINE CONSTRAINTS & OBJECTIVE FUNCTION
 [A, b]         = calcLINEAR_CONSTRAINTS(n_var, MaxViol);
 nlconstraints =
@(X)calcNONLINEAR_CONSTRAINTS(FP,gen,JAC,THE,XtoDIM(gen,X,DIM),ULS,F
LS,BUC,LEN,tim,SCF,upper,n_var);
 objective     =
@(X)calcCOST_JACKET(gen,JAC,XtoDIM(gen,X,DIM),COST);


%% (4) SET OPTIONS OF 'fmincon'
% 'outputSQP_FUNC'      is a user-defined function called at each
iteration and gives plots.
% 'ConstraintTolerance' is an upper bound and operates differently
from other tolerances. If it is not satisfied the solver attempts
to continue.
% 'StepTolerance'       is a lower bound. Iterations end when: |  xi
-   xi+1 | < StepTolerance
```

```
% 'FunctionTolerance'   is a lower bound. Iterations end when:
|f(xi) - f(xi+1)| < FunctionTolerance
% 'OptimalityTolerance' is a lower bound.

options = optimoptions ('fmincon', 'Algorithm'              ,'sqp',
'MaxIterations'      , 9       , 'SpecifyConstraintGradient', true,
'OptimalityTolerance', 1e-2    , 'SpecifyObjectiveGradient' , true,
'ConstraintTolerance', MaxViol , 'CheckGradients'           , false,
'FunctionTolerance'  , 3e-2    , 'Diagnostics'              ,'off' ,
'StepTolerance'      , 9e-3    , 'Display'                 ,'iter',
'OutputFcn'          , @outputSQP_FUNC ,...
'PlotFcn'            , {'optimplotfval','optimplotstepsize'});


%% (5) INVOKE 'fmincon' ('[]' indicate no linear equalities in the
problem)
    [X, ~, exitflag, output] = fmincon(objective,X0,A,b,[ ],[
],Lb,Ub,nlconstraints,options);

    disp(exitflag);
    disp(output);

%% (6) UPDATE 'DIM' MATRIX
    DIM = XtoDIM(gen,X,DIM);


%% (7) SAVE FIGURES & EXIT
    h = get(0,'children');
    for i=1:length(h)

savefig(h(i),sprintf('%sOutput_PLOTS/InternalSizingFigs_gen%d_fig%d.
fig',FP{2},gen,i));
    end
    close all
    clear h


end % of function
```

C2 – Total cost of jacket

The function below calculates the total cost of the jacket structure as well as the gradients of it with respect to the design variables.

```
function [total_cost, costgrad] = calcCOST_JACKET(gen,JAC,DIM,COST)

********************************************************************
%
%   Calculate cost of all designs
%   this is the objective function/criterion for the optimization
%
*********************************************************************
% written  by Daniel Zwick,      05/2014
% modified by Nikolaos Xyloudis, 05/2016


% CONTENT
% ------------------------------------------------------------------
% 1) Calculate cost (manufacturing + material)

% INPUT parameters
% ------------------------------------------------------------------
% gen       - number of generation
% JAC       - general jacket parameters
% DIM       - bay specific jacket parameters
% COST      - cost of the structure in [euros]
% THE       - angle between members. These angles are used for
%             calculating the welded length between members in order
              to find the fabric. cost. Here instead of calculating

% OUTPUT parameters
% ------------------------------------------------------------------
---
%    COST      - cost of jacket structure [euros]
%    costgrad  - gradient of the cost function
%    MAT_COST  - material cost
%    FAB_COST  - fabrication cost


%% Adding functions directory to Matlab search path
    addpath('JACKETscripts')


%% 1) Calculate cost of structure (material + fabrication)

    % calculate cost for all designs
        % definitions
            COST{gen,1}     = 0;
%             MAT_COST{gen,1} = 0;
%             FAB_COST{gen,1} = 0;
            i=0;
            j=0;
            k=0;
            l=0;
            m=0;
```

```
% calculate jacket parameters (nodes and beams)
     clear Jnod Jbea
     [Jnod, Jbea, ~, ~] = jacketMATLAB_v0(JAC{gen,1},DIM{gen,1});

% calculate cost of each beam and add them at the end
for n=1:length(Jbea(:,1))  %length(Jbea(:,1) = total number of beams
                if (all(Jbea(n,:)==0) == 0)

                    % extract length of beam
                        P1 = Jnod(Jbea(n,1),:);
                        P2 = Jnod(Jbea(n,2),:);
                        L  = pdist([P1; P2]);

% extract diameter and thickness
     if (Jbea(n,3) < 100) % K-stubs on legs (64 elements)
                            category = 1;
                            i=i+1;
                            el=i;
                            id = Jbea(n,3);
                            jd = 1;
                            D  = DIM{gen,1}{2,2}(id,jd);
                            T  = DIM{gen,1}{2,2}(id,jd+1);
theta=90; %stubs are parallel to legs (welded length is equal to D)

      elseif ((Jbea(n,3) > 100) && (Jbea(n,3) < 200))
            % K-stubs on braces (64 elements)
                            category = 2;
                            j=j+1;
                            el=j;
                            id = ceil((Jbea(n,3)-100)/2);
                            jd = 3 - 2 * mod((Jbea(n,3)-100),2);
                            D  = DIM{gen,1}{1,4}(id,jd);
                            T  = DIM{gen,1}{1,4}(id,jd+1);
theta=34.37; %mean value among all bays for angles between leg-brace
at K-joints

      elseif ((Jbea(n,3) > 200) && (Jbea(n,3) < 300))
            % X-stubs (64 elements)
                            category = 3;
                            k=k+1;
                            el=k;
                            id = ceil((Jbea(n,3)-200)/3);
% "ceil" rounds each element to the nearest integer greater than or
equal to that element
                            md = mod((Jbea(n,3)-200),3);
% "mod" returns the remainder after division
                            jd = 3*md^2 - 7*md + 5;
                            D  = DIM{gen,1}{1,5}(id,jd);
                            T  = DIM{gen,1}{1,5}(id,jd+1);
theta  = 63; % approximate value for angle between brace-brace at X-
joints
```

```
       elseif ((Jbea(n,3) > 300) && (Jbea(n,3) < 400))
            % legs between stubs (16 elements)
                            category = 4;
                            l=l+1;
                            el=l;
                            id = Jbea(n,3)-300;
                            jd = 1;
                            D  = DIM{gen,1}{1,2}(id,jd);
                            T  = DIM{gen,1}{1,2}(id,jd+1);
theta=90; % legs are parallel to stubs (welded length is equal to D)

       elseif ((Jbea(n,3) > 400) && (Jbea(n,3) < 500))
            % braces between stubs (64 elements)
                            category = 5;
                            m=m+1;
                            el=m;
                            id = ceil((Jbea(n,3)-400)/2);
                            jd = 3 - 2 * mod((Jbea(n,3)-400),2);
                            D  = DIM{gen,1}{1,3}(id,jd);
                            T  = DIM{gen,1}{1,3}(id,jd+1);
theta=90;%braces are parallel to stubs (welded length is equal to D)
       end
% cost + gradient of current beam member
[cost, cost_grad_DT,~,~] = calcCOST_MEMBER(D,T,L,theta);

% calculate cost by adding the cost of last beam member to the
previous calculated ones
COST{gen,1}     = COST{gen,1}      +  cost;

% calculate grads for each member and arrange them in a cell matrix
 cost_grad{category,id,jd  }(el) = cost_grad_DT(1);
 cost_grad{category,id,jd+1}(el) = cost_grad_DT(2);
     end
 end
total_cost = COST{gen,1};


%%  Calculate gradient of cost function
   if nargout > 1 % gradient required, must be a [numVar x 1] matrix

   % K-braces
    for i=1:4
    costgrad(i)     = sum(cost_grad{2,1,i}(:));
    costgrad(i+4)   = sum(cost_grad{2,2,i}(:));
    costgrad(i+8)   = sum(cost_grad{2,3,i}(:));
    costgrad(i+12)  = sum(cost_grad{2,4,i}(:));
    end

% X-braces
% bay1
costgrad(17) = sum(cost_grad{3,1,1}(:)) + sum(cost_grad{3,1,3}(:)) +
sum(cost_grad{3,1,5}(:));
costgrad(17) = costgrad(17) + sum(cost_grad{5,1,1}(:)) +
sum(cost_grad{5,1,3}(:));

costgrad(18) = sum(cost_grad{3,1,2}(:)) + sum(cost_grad{3,1,4}(:)) +
sum(cost_grad{3,1,6}(:));
costgrad(18) = costgrad(18) + sum(cost_grad{5,1,2}(:)) +
sum(cost_grad{5,1,4}(:));
```

```
% bay2
costgrad(19) = sum(cost_grad{3,2,1}(:)) + sum(cost_grad{3,2,3}(:)) +
sum(cost_grad{3,2,5}(:));
costgrad(19) = costgrad(19) + sum(cost_grad{5,2,1}(:)) +
sum(cost_grad{5,2,3}(:));

costgrad(20) = sum(cost_grad{3,2,2}(:)) + sum(cost_grad{3,2,4}(:)) +
sum(cost_grad{3,2,6}(:));
costgrad(20) = costgrad(20) + sum(cost_grad{5,2,2}(:)) +
sum(cost_grad{5,2,4}(:));

% bay3
costgrad(21) = sum(cost_grad{3,3,1}(:)) + sum(cost_grad{3,3,3}(:)) +
sum(cost_grad{3,3,5}(:));
costgrad(21) = costgrad(21) + sum(cost_grad{5,3,1}(:)) +
sum(cost_grad{5,3,3}(:));

costgrad(22) = sum(cost_grad{3,3,2}(:)) + sum(cost_grad{3,3,4}(:)) +
sum(cost_grad{3,3,6}(:));
costgrad(22) = costgrad(22) + sum(cost_grad{5,3,2}(:)) +
sum(cost_grad{5,3,4}(:));

% bay4
costgrad(23) = sum(cost_grad{3,4,1}(:)) + sum(cost_grad{3,4,3}(:)) +
sum(cost_grad{3,4,5}(:));
costgrad(23) = costgrad(23) + sum(cost_grad{5,4,1}(:)) +
sum(cost_grad{5,4,3}(:));

costgrad(24) = sum(cost_grad{3,4,2}(:)) + sum(cost_grad{3,4,4}(:)) +
sum(cost_grad{3,4,6}(:));
costgrad(24) = costgrad(24) + sum(cost_grad{5,4,2}(:)) +
sum(cost_grad{5,4,4}(:));


% K-legs

% bay1
costgrad(25) = sum(cost_grad{1,1,1}(:));
costgrad(26) = sum(cost_grad{1,1,2}(:));

% bay2
costgrad(27) = sum(cost_grad{1,2,1}(:)) + sum(cost_grad{4,1,1}(:));
costgrad(28) = sum(cost_grad{1,2,2}(:)) + sum(cost_grad{4,1,2}(:));

% bay3
costgrad(29) = sum(cost_grad{1,3,1}(:)) + sum(cost_grad{4,2,1}(:));
costgrad(30) = sum(cost_grad{1,3,2}(:)) + sum(cost_grad{4,2,2}(:));

% bay4
costgrad(31) = sum(cost_grad{1,4,1}(:)) + sum(cost_grad{4,3,1}(:));
costgrad(32) = sum(cost_grad{1,4,2}(:)) + sum(cost_grad{4,3,2}(:));

% bay5
costgrad(33) = sum(cost_grad{1,5,1}(:)) + sum(cost_grad{4,4,1}(:));
costgrad(34) = sum(cost_grad{1,5,2}(:)) + sum(cost_grad{4,4,2}(:));
    end
end   % of function
```

C3 – Cost function

The function below calculates the material and manufacturing costs of each member.

```
function [cost,cost_grad,material_cost,fabrication_cost] =
calcCOST_MEMBER(D,T,L,theta)
%*********************************************************************
%
%   COST FUNCTION (OBJECTIVE FUNCTION) + 1st ORDER GRADIENTS
%
%
*********************************************************************
% written by Nikolaos Xyloudis, 06/2016


% INPUT parameters
% --------------------------------------------------------------------
%        D               [m]     Diameter of member
%        T               [m]     Thickness of member
%        L               [m]     Member length
%    theta               [deg]   Angle between welded members
(0<theta<=90)
k_material    = 0.60; % [€/kg]  material cost for steel
                        (0.5<k_material<1.5)
k_fabrication = 0.57; % [€/min] labour cost (0< k_fabrication<1)


% OUTPUT parameters
% ---------------------------------------------------------------------
%   cost    - cost of structure calculated member by member and then
%             summed up (includes material+fabrication costs)
%
%  cost_grad  - gradient of the cost function with respct to the
                design variables


% BASIC parameters (every dim is in mm except the D at "t_CG")
% ---------------------------------------------------------------------
rho_steel = 7850*10^(-9);  % [kg/mm^3] material density of
structural steel
D = D*1000;                % [mm]
T = T*1000;                % [mm]
L = L*1000;                % [mm]
angle = deg2rad(theta);    % [rad]  Convert angle to rad from deg
A_steel   = pi*(D*T-T^2);  % [mm^2] Cross sectional area of member
V_steel   = A_steel*L ;    % [mm^3] Volume of member


%% MATERIAL cost
material_cost = k_material*rho_steel*V_steel ;  % [€] Cost of
material
```

```
%% FABRICATION cost

% (1) Forming plates into shell elements
theta_FO = 3; % (Farkas, 2008)
mu       = 6.8582513-(4.527217*T^(-0.5))+(0.009541996*D^0.5);
t_FO     = theta_FO*exp(mu); % [min]

t_FO_grad_D = theta_FO* 0.00475*(D^(-0.5))* exp(mu);
t_FO_grad_T = theta_FO* 2.265*  (T^(-1.5))* exp(mu);


% (2) Hand cutting and machine grinding  % !!! D is in [m] and T is
in [mm] only for this equation !!!
theta_DC = 3;                            % (Farkas, 2008)
L_cut    = (2*pi/sin(angle))*(D/1000);
t_CG     = theta_DC*(4.54+0.4229*T^2)*L_cut; % [min]

t_CG_grad_D = (2*pi/sin(angle))* theta_DC* (4.54+0.4229*T^2) /1000;
t_CG_grad_T = (2*pi/sin(angle))* theta_DC* 0.8458*D*T        /1000;



% (3) Welding times


% Time of preparation, assembly and tacking
C_1       = 1;
theta_DW  = 3; % can take values of 2 or 3 for tubular members
(Farkas, 1997)
k_members = 2; % number of elements to be welded
tw1 = C_1*theta_DW*(k_members*rho_steel*V_steel)^0.5;  % [min]

tw1_grad_D = C_1*theta_DW*((k_members*rho_steel*pi*L*T)/
(2*(k_members*rho_steel*V_steel)^0.5));

tw1_grad_T = C_1*theta_DW*((k_members*rho_steel*pi*L*(D-
2*T))/(2*(k_members*rho_steel*V_steel)^0.5));


% Real welding time
C_2   = 0.1346*10^(-3);          % constant depending on the welding
technology (0.1346 for "SAW")
a_w   = 10;                      % welding size (width) [mm]
nu    = 2;                       % depending on the welding tecnology
and the type of weld (K, X, V), (2 here for V-butt weld and "SAW"
technology)
L_weld = (2*pi/sin(angle))*D;    % weld length
tw2   = C_2*(a_w^nu)*L_weld;     % [min]

tw2_grad_D = C_2*(a_w^nu)*(2*pi/sin(angle));
tw2_grad_T = 0;
```

```
% Changing electrode, deslagging, chipping
tw3  = 0.3*tw2; % [min]
tw3_grad_D = 0.3*tw2_grad_D;
tw3_grad_T = 0.3*tw2_grad_T;


% Sum up welding costs
t_weld = tw1 + tw2 + tw3;
t_weld_grad_D = tw1_grad_D + tw2_grad_D + tw3_grad_D;
t_weld_grad_T = tw1_grad_T + tw2_grad_T + tw3_grad_T;


% (4) Painting
theta_DP = 3; % Can take values of 1,2,3. 3 is used for complicated
structures
alpha_gc = 3*10^(-6);     % [min/mm^2]
alpha_tc = 4.15*10^(-6);  % [min/mm^2]
A_paint  = pi*L*D;        % [mm^2] area to be painted
t_P      = theta_DP*(alpha_gc+alpha_tc)*A_paint; % [min]
t_P_grad_D = theta_DP*(alpha_gc+alpha_tc)*pi*L;
t_P_grad_T = 0;


fabrication_cost = k_fabrication*(t_FO + t_CG + t_weld + t_P) ;


%% TOTAL cost of the considered tubular member
cost = material_cost + fabrication_cost ; % [€]


%% COMPUTE GRADIENTS OF COST FUNCTION (explicit expressions provided
below)

% ATTENTION: need to multiply with '1000' since variables in
'fmincon' are in [m] and the cost is calculated in terms of [mm].
'fmincon' tries steps in 'm' but the sensitivity information
provided from the functions must be also in 'm'. Therefore the cost
sensitivity in terms of [m] needs to be calculated this way.
sens_factor = 1000;

material_cost_grad_D = sens_factor*k_material*rho_steel*L*pi*T;

material_cost_grad_T = sens_factor*k_material*rho_steel*L*pi*(D-
2*T);

fabrication_cost_grad_D = sens_factor*k_fabrication*(t_FO_grad_D +
t_CG_grad_D + t_weld_grad_D + t_P_grad_D);

fabrication_cost_grad_T = sens_factor*k_fabrication*(t_FO_grad_T +
t_CG_grad_T + t_weld_grad_T + t_P_grad_T);

cost_grad_D = material_cost_grad_D + fabrication_cost_grad_D;
cost_grad_T = material_cost_grad_T + fabrication_cost_grad_T;

% Gradient vector with respect to D, T (analytic expressions)
cost_grad = [cost_grad_D, cost_grad_T];

end   % of function
```

C4 –Linear constraints

This function calculates the linear constraints of the optimization problem. These constraints are due to the limitations for the validity of the SCF equations used to calculate the fatigue at tubular joints (‘Euthymiou’ equations for SCF formulas).

```
function  [A, b] = calcLINEAR_CONSTRAINTS(n_var,contol)
%
%  Linear constraints, take x in vector form, x(:). i.e. a linear
%  constraint of the form A*x <= b or Aeq*x = beq takes x as a
%  vector, not a matrix. Ensure that matrix A or Aeq has the same
%  number of columns as x0 has elements.
%  When there are K constraints on variables X with Z components,
%  you supply the [KxZ] matrix A and the [Kx1] vector b.
%
%  for X0=[M x N] and  4 constraints --> A=[4 x (MxN)] & b=[4x1]


%  FOR OC4 GEOMETRY:

% gamma = D/2*T     (8 < gamma < 32)
    %  9 * g1:   - D + 16*T  <=  0 ---> A1=[9 x 34] & b1=[9 x 1]
    %  9 * g2:     D - 64*T  <=  0 ---> A2=[9 x 34] & b2=[9 x 1]

% beta = d/D        (0.2 < beta < 1.0)
    %  8 * g3:   - D +    d  <=  0 ---> A3=[8 x 34] & b3=[8 x 1]
    %  8 * g4: 0.2*D -    d  <=  0 ---> A4=[8 x 34] & b4=[8 x 1]

% tau = t/T         (0.2 < tau < 1.0)
    %  8 * g5:   - T +    t  <=  0 ---> A5=[8 x 34] & b5=[8 x 1]
    %  8 * g6: 0.2*T -    t  <=  0 ---> A6=[8 x 34] & b6=[8 x 1]

% alpha = 2*L/D     (4 < alpha < 40)
    %  9 * g7:     -20*D     <= -L ---> A7=[9 x 34] & b7=[9 x 1]
    %  9 * g8:       2*D     <=  L ---> A8=[9 x 34] & b8=[9 x 1]

% ratio between Tbraces & Xbraces
   %8* g9 :  DXbrace - DKbrace  <= 0 ---> A9 =[8 x 34] & b9 =[8 x 1]
   %8* g10:  TXbrace - TKbrace  <= 0 ---> A10=[8 x 34] & b10=[8 x 1]

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

% Constraints g1 (x9), geometry (parameter 'gamma', FLS, NORSOK N-
004)
     n_g1 = 17;                              % number of g1 constraints
     A1 = zeros(n_g1, n_var);
     A1(1      :2*n_g1+1:end) = -1;          % D multipliers
     A1(n_g1+1:2*n_g1+1:end) = 16.1;           % T multipliers
     b1 = zeros(n_g1,1);                     % b1 = [9 x  1]
```

```
% Constraints g2 (x9), geometry (parameter 'gamma', FLS, DNV-RP-
C203)
     n_g2 = 17;                          % number of g2 constraints
     A2 = zeros(n_g2, n_var);
     A2(1     :2*n_g2+1:end) = 1;        % D multipliers
     A2(n_g2+1:2*n_g2+1:end) = -18;      % T multipliers
     b2 = zeros(n_g2,1);

%------------------------------------------------------------------
% Constraints g3 (x8), geometry (parameter 'beta', FLS, DNV-RP-C203)
     n_g3=8;      Dterm3=-0.9+contol;    dterm3=1;
     A3 = zeros(n_g3, n_var);
     A3(1,25)=Dterm3; A3(8,33)=Dterm3;% D_chord multipliers for 5 K-
chords
     for i=2:2:6
         A3(i:i+1, 25+i) = Dterm3;
     end
     for i=1:8                        % d_brace multipliers for 4
bays
         A3(i, 2*i-1) = dterm3;
     end
     b3 = zeros(n_g3,1);


% Constraints g4 (x8), geometry (parameter 'beta', FLS, DNV-RP-C203)
     n_g4=8;     Dterm4=0.2+contol;    dterm4=-1;

     A4 = zeros(n_g4, n_var);
     A4(1,25)=Dterm4; A4(8,33)=Dterm4;% D_chord multipliers for 5 K-
chords
     for i=2:2:6
         A4(i:i+1, 25+i) = Dterm4;
     end
     for i=1:8                        % d_brace multipliers for 4
bays
         A4(i, 2*i-1) = dterm4;
     end
     b4 = zeros(n_g4,1);

%-------------------------------------------------------------------
% Constraints g5 (x8), geometry (parameter 'tau', FLS, DNV-RP-C203)
     n_g5=8;      Tterm5=-0.9+contol;    tterm5=1;

     A5 = zeros(n_g5, n_var);
     A5(1,26)=Tterm5; A5(8,34)=Tterm5;% T_chord multipliers for 5 K-
chords
     for i=2:2:6
         A5(i:i+1, 26+i) = Tterm5;
     end
     for i=1:8                        % t_brace multipliers for 4
bays
         A5(i, 2*i) = tterm5;
     end
     b5 = zeros(n_g5,1);
```

```
% Constraints g6 (x8), geometry (parameter 'tau', FLS, DNV-RP-C203)

     n_g6=8;      Tterm6=0.2+contol;     tterm6=-1;

     A6 = zeros(n_g6, n_var);
     A6(1,26)=Tterm6; A6(8,34)=Tterm6;% T_chord multipliers for 5 K-
chords
     for i=2:2:6
         A6(i:i+1, 26+i) = Tterm6;
     end
     for i=1:8                        % t_brace multipliers for 4
bays
         A6(i, 2*i) = tterm6;
     end
     b6 = zeros(n_g6,1);

%------------------------------------------------------------------
% Constraints g7 (x9), geometry (parameter 'alpha', FLS, NORSOK N-
004)
     L    = 5;               % length of joints [m]
('stressfactors.m')
     n_g7 = 9;               % number of g7 constraints

     A7 = zeros(n_g7, n_var);
     A7(16*n_g7+1:2*n_g7+1:22*n_g7+4) = -20;  % D multipliers
Xbraces
     A7(24*n_g7+5:2*n_g7+1:end      ) = -20;  % D multipliers Klegs
     b7 = -L * ones(n_g7,1);        % b7 = [9 x  1]

     % Constraints g8 (x9), geometry (parameter 'alpha', FLS, DNV-
RP-C203)
     n_g8 = 9;               % number of g8 constraints

     A8 = zeros(n_g8, n_var);
     A8(16*n_g8+1:2*n_g8+1:end) = 2;        % D multipliers
     b8 = L * ones(n_g8,1);

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

 % Constraints g9 (x8), (D_Xbrace - D_Kbrace <= 0)
     n_g9=8;        DXbrace=1;      DKbrace=-1;

     A9 = zeros(n_g9, n_var);
     A9(1,17) = DXbrace;  A9(1,1 ) = DKbrace;
     A9(2,17) = DXbrace;  A9(2,3 ) = DKbrace;
     A9(3,19) = DXbrace;  A9(3,5 ) = DKbrace;
     A9(4,19) = DXbrace;  A9(4,7 ) = DKbrace;
     A9(5,21) = DXbrace;  A9(5,9 ) = DKbrace;
     A9(6,21) = DXbrace;  A9(6,11) = DKbrace;
     A9(7,23) = DXbrace;  A9(7,13) = DKbrace;
     A9(8,23) = DXbrace;  A9(8,15) = DKbrace;

     b9 = zeros(n_g9,1);
```

```
% Constraints g10 (x8),  (T_Xbrace - T_Kbrace <= 0)
     n_g10=8;         TXbrace=1;    T_Kbrace=-1;

     A10 = zeros(n_g9, n_var);
     A10(1,18) = TXbrace;  A10(1,2 ) = T_Kbrace;
     A10(2,18) = TXbrace;  A10(2,4 ) = T_Kbrace;
     A10(3,20) = TXbrace;  A10(3,6 ) = T_Kbrace;
     A10(4,20) = TXbrace;  A10(4,8 ) = T_Kbrace;
     A10(5,22) = TXbrace;  A10(5,10) = T_Kbrace;
     A10(6,22) = TXbrace;  A10(6,12) = T_Kbrace;
     A10(7,24) = TXbrace;  A10(7,14) = T_Kbrace;
     A10(8,24) = TXbrace;  A10(8,16) = T_Kbrace;

     b10 = zeros(n_g10,1);

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

   % create input matrices of linear ineq cosntraints for 'fmincon'
     A = [A1;A2;A3;A4;A5;A6;A7;A8;A9;A10];   % A = [84 x 34]
     b = [b1;b2;b3;b4;b5;b6;b7;b8;b9;b10];   % b = [84 x  1]


end % of function
```

C5 – Nonlinear constraints

The function below calculates all the nonlinear constraint values (g9, g10, g11) of the problem as well as their 1st order gradients by calling the three corresponding constraint functions for each type of constraint (yielding, buckling, fatigue).

```
function  [g, h, gg, gh] =
calcNONLINEAR_CONSTRAINTS(FP,gen,JAC,THE,DIM,ULS,FLS,BUC,LEN,tim,SCF
,upperLim,n_var)

% -----------------------------------------------------------------*
%  This function calculates the nonlinear constraints g11, g12, g13
%  (FLS, ULS, BUC)
% -----------------------------------------------------------------*
% written  by Nikolaos Xyloudis, 06/2016


% INPUT parameters
% ------------------------------------------------------------------
%  FP     - FEDEM and FATIGUE parameter
%  gen    - number of generation
%  fed    - counter of Fedem runs
%  JAC    - general jacket parameters
%  DIM    - bay specific jacket parameters
%  ULS    - normalized ultimate loads
%  FLS    - normalized fatigue damage
%  BUC    - buckling safety factor
%  THE    - structural angle theta for SCF calculation
%  LEN    - length of legs and braces
%  tim    - identification of time series file to be used


% OUTPUT parameters
% ------------------------------------------------------------------
%  g   - returns inequality constraints, (here it is a [1 x 46]
matrix)
%  h   - returns equality constraints,    (here it is a [] matrix)
%  gg  - returns gradients of inequalities; each column contains a
gradient
%  gh  - returns gradients of equalities; each column contains a
gradient


%% SET TARGET FOR OPTIMIZATION (+0.1 means that g<=-0.1 instead of
<=0)
%  Beacause of the LOCAL APPROACH !!!
%
%  Intention of the targets is to guide the design towards the limit
(<=0)
%  without exceeding it after an internal sizing step which is based
on
%  local assumptions. Apply this target in order to give redundancy
to the
%  optimized design for the FEDEM analysis to follow.
    TARGET         = 0.20;
```

```
%% ORGANIZE SIMULATION RUNS & CALCULATE STRESS SERIES

    Dam              = cell(length(FP{6}),1);  % Create 3 cells for
production load cases (FLS), length(FP{6}) = number of production
load cases (here is equal to 3)
    Dam_grad_D       = cell(length(FP{6}),1);
    Dam_grad_T       = cell(length(FP{6}),1);

    U          = cell(FP{4}-length(FP{6}),1); % Create 2 cells for
extreme load cases (ULS), FP{4} = total number of load cases (here
is equal to 5 = 3 (FLS) + 2 (ULS))
    U_grad_D   = cell(FP{4}-length(FP{6}),1);
    U_grad_T   = cell(FP{4}-length(FP{6}),1);

    B          = cell(FP{4},1);               % Create 5 cells for
all load cases (BUC)
    B_grad_D   = cell(FP{4},1);
    B_grad_T   = cell(FP{4},1);

 % calculate SCFs
   SCF =
stressfactors(DIM{gen,1},THE{gen,1},length(DIM{gen,1}{1,1}),1);


   for lc=1:FP{4}   % check all load cases, (FP{4}=5=number of load
cases)

    % stress calculation + gradients
     clear matfileTIM SIG  SIG_grad_D  SIG_grad_T...
                      HSS  HSS_grad_D  HSS_grad_T
     matfileTIM =
sprintf('%smatfiles/TIM_%03.0f_%02.0f.mat',FP{2},tim,lc);

     [SIG, SIG_grad_D, SIG_grad_T] =
stressseries(DIM{gen,1},FP,gen,lc,length(DIM{gen,1}{1,1}),JAC{gen,1}
(1,1),matfileTIM);
     [HSS, HSS_grad_D, HSS_grad_T] = sig2hss(SIG,SIG_grad_D,
SIG_grad_T,SCF,length(DIM{gen,1}{1,1}),JAC{gen,1}(1,1));


%% CALCULATE COSNTRAINTS OVER ALL LOAD CASES
    % Fatigue (Dam), Ultimate (U) and Buckling (B) calculation

    if (lc <= length(FP{6}))
     % length(FP{6})= number of power production load cases (here
=3)
     % Dam{b,t,l,e} - 192 values in total per FLS load case
     %                (max damage value in each of 192 joints)
     [Dam{lc,1},Dam_grad_D{lc,1},Dam_grad_T{lc,1}] =
hss2damage(HSS,FP,length(DIM{gen,1}{1,1}),JAC{gen,1}(1,1),lc,DIM{gen
,1});

    else
% U{b,t,l,e} = {1:4, 1:3, 1:4, 1:4} = 192 total values per ULS load
case
% 12 values per bay per side/plane of the 3D jacket
```

```
[U{lc-length(FP{6}),1}, U_grad_D{lc-length(FP{6}),1}, U_grad_T{lc-
length(FP{6}),1}] =  hss2ultimate(HSS,HSS_grad_D,
HSS_grad_T,length(DIM{gen,1}{1,1}),JAC{gen,1}(1,1));
    end

[B{lc,1},B_grad_D{lc,1}, B_grad_T{lc,1}] =
bucklingcheck(DIM{gen,1},LEN{gen,1},FP,gen,lc,length(DIM{gen,1}{1,1}
),JAC{gen,1}(1,1),matfileTIM);

end % of 'for' loop for load cases


%% CALCULATE MAX VALUES OF CONSTRAINTS ('WORST CASE APPROACH')

% g11 - calculate maximum ultimate loading over all extreme load
cases
%   ULS{gen,1} = [25 x 1] matrix (ULS>=1)
    [ULS{gen,1},ULS_grad_D{gen,1},ULS_grad_T{gen,1}] =
maxultimate(U,U_grad_D,U_grad_T,length(DIM{gen,1}{1,1}),JAC{gen,1}(1
,1));

   % Calculate cosntraint (g_ULS<=0)
    g_ULS        = ULS{gen,1} + TARGET;  % [25 x 1] matrix
    g_ULS_grad_D = ULS_grad_D{gen,1};    % [25 x 1] matrix
    g_ULS_grad_T = ULS_grad_T{gen,1};    % [25 x 1] matrix


%--------------------------------------------------------------------
% g12 - calculate max buckling value over al load cases
%   BUC{gen,1} = [4 x 3] matrix
    [BUC{gen,1},BUC_grad_D{gen,1},BUC_grad_T{gen,1}] =
totalbuckling(B,B_grad_D,B_grad_T,length(DIM{gen,1}{1,1}),FP);

%   Transform [4x3] matrix to [1x12] vector matrix
%   Each column represents a constraint:
%       columns 1-4  = legs (1 bay1, 2 bay2...)
%       columns 5-8  = lower braces (5 bay1, 6 bay2...)
%       columns 9-12 = upper braces (9 bay1, 12 bay2...)
    g_BUC         = reshape(BUC{gen,1},1,[]) + TARGET;   % [1 x 12]
    g_BUC_grad_D  = reshape(BUC_grad_D{gen,1},1,[]);     % [1 x 12]
    g_BUC_grad_T  = reshape(BUC_grad_T{gen,1},1,[]);     % [1 x 12]


%--------------------------------------------------------------------
 % g13 - calculate total fatigue damage over all power production
load cases
   % FLS{gen,1} = [25 x 1] matrix
    [FLS{gen,1},FLS_grad_D{gen,1},FLS_grad_T{gen,1}] =
totaldamage(Dam,Dam_grad_D,Dam_grad_T
,length(DIM{gen,1}{1,1}),JAC{gen,1}(1,1));
   %[     ~    ,      ~          ,FLS_grad_Tchord{gen,1}] =
totaldamage(Dam,Dam_grad_D,Dam_grad_Tchord,length(DIM{gen,1}{1,1}),J
AC{gen,1}(1,1));

   % Calculate cosntraint (g_FLS<=0)
    g_FLS         = FLS{gen,1} + TARGET;     % [25 x 1] matrix
    g_FLS_grad_D  = FLS_grad_D{gen,1};       % [25 x 1] matrix
    g_FLS_grad_T  = FLS_grad_T{gen,1};       % [25 x 1] matrix
```

```
%% NONLINEAR INEQUALITY CONSTRAINTS (g <=0)
%  g is a vector with each (column) element being a constraint
%  e.g. for 15 ineq. constraints --> g = [1 x 15]

% g11(ULS), 17 yielding constraints
    g11(1:8)     = g_ULS(6:13);     % K-stubs on braces (2 values per
bay)
    g11(9)       = g_ULS(14)  ;     % X-stubs on braces bay 1
    g11(10)      = g_ULS(17)  ;     % X-stubs on braces bay 2
    g11(11)      = g_ULS(20)  ;     % X-stubs on braces bay 3
    g11(12)      = g_ULS(23)  ;     % X-stubs on braces bay 4
    g11(13:17)   = g_ULS(1:5) ;     % K-stubs on legs

% g12(BUC), 12 buckling constraints
    g12(1:12)  = g_BUC;  % for 12 column (Euler) buckling const.,
BUC

% g13(FLS), 17 fatigue constraints
    g13(1:8)   = g_FLS(6:13);                          % K-stubs on
braces (2 values per bay)
    g13(9)     = g_FLS(14);   % X-stubs on braces bay 2
    g13(10)    = g_FLS(17);   % X-stubs on braces bay 2
    g13(11)    = g_FLS(20);   % X-stubs on braces bay 3
    g13(12)    = g_FLS(23);   % X-stubs on braces bay 4
    g13(13:17) = g_FLS(1:5);  % K-stubs on legs
%   g13(18:20) = g_FLS(26:28) ;  % K-stubs on legs (chord side upper
braces at bays 1,2,3)

%% ALL NONLINEAR CONSTRAINTS
    g = [g11, g12, g13]; % nonlinear inequality consrtaints
    h = [ ];            % nonlinear equality consrtaints (h=0)


%% 1st ORDER GRADIENTS OF INEQUALITY COSNTRAINTS
   if nargout > 2 % if gradients required

%  Each column of the gradient matrix 'gg' is associated with one
const.
%  This is the transpose of the form of Jacobians.
%                   gg = [K x L] where:
%                         K = number of variables,
%                         L = number of nonlinear inequality const.
%  In our problem:  gg = [34 x 46]
%                         34 - Variables
%                         46 - Cosntraints (17 ULS + 12 BUC + 17
FLS)


%% ULS constraints (x17), gg11 = [34x17] --> gg(:,1:17)
    gg11 = zeros(n_var, length(g11));

   % constraints on K-stubs on braces (x8)
     for i=1:2:15
     gg11(i:i+1,int8(i/2)) = [g_ULS_grad_D(int8(i/2)+5);
g_ULS_grad_T(int8(i/2)+5)];
    %gg11(1:16 ,1:8      )    g_ULS_grad_D(   6:13     )
g_ULS_grad_T( 6:13      )
     end
   % constraints on X-stubs on braces (x4)
     gg11(17:18,9)  = 
[extremum_AtoB_FUNC(g_ULS(14),g_ULS_grad_D(14));
extremum_AtoB_FUNC(g_ULS(14),g_ULS_grad_T(14))];
     gg11(19:20,10) = 
[extremum_AtoB_FUNC(g_ULS(17),g_ULS_grad_D(17));
extremum_AtoB_FUNC(g_ULS(17),g_ULS_grad_T(17))];
     gg11(21:22,11) = 
[extremum_AtoB_FUNC(g_ULS(20),g_ULS_grad_D(20));
extremum_AtoB_FUNC(g_ULS(20),g_ULS_grad_T(20))];
     gg11(23:24,12) = 
[extremum_AtoB_FUNC(g_ULS(23),g_ULS_grad_D(23));
extremum_AtoB_FUNC(g_ULS(23),g_ULS_grad_T(23))];
    %gg11(17:24,9:12)

   % constraints on K-stubs on legs (x5)
     for i=1:2:9
     gg11(24+i:24+i+1,int8(i/2)+12) = [g_ULS_grad_D(int8(i/2)); 
g_ULS_grad_T(int8(i/2))];
    %gg11(25:34     ,13:17        )    g_ULS_grad_D(   1:5   ) 
g_ULS_grad_T( 1:5     )
     end


%% BUC constraints (x12), gg12 = [34x12] --> gg(:,18:29)
    gg12 = zeros(n_var,length(g12)); % [34x12] matrix

  % constraints on K-stubs on braces (x0), gg12(1:16,1:12)=0
    % no buckling constraints related to K-stubs on braces

  % constraints on X-stubs on braces (x8), gg12(17:24,5:12)
     % a) constraints related to lower braces (x4), gg12(17:24,5:8)
     gg12(17:18,5)  = [g_BUC_grad_D(5); g_BUC_grad_T(5)];  % lower 
brace bay1
     gg12(19:20,6)  = [g_BUC_grad_D(6); g_BUC_grad_T(6)];  % lower 
brace bay2
     gg12(21:22,7)  = [g_BUC_grad_D(7); g_BUC_grad_T(7)];  % lower 
brace bay3
     gg12(23:24,8)  = [g_BUC_grad_D(8); g_BUC_grad_T(8)];  % lower 
brace bay4

     % b) constraints related to upper braces (x4), gg12(17:24,9:12)
     gg12(17:18,9)  = [g_BUC_grad_D(9);  g_BUC_grad_T(9)];  % upper 
brace bay1
     gg12(19:20,10) = [g_BUC_grad_D(10); g_BUC_grad_T(10)]; % upper 
brace bay2
     gg12(21:22,11) = [g_BUC_grad_D(11); g_BUC_grad_T(11)]; % upper 
brace bay3
     gg12(23:24,12) = [g_BUC_grad_D(12); g_BUC_grad_T(12)]; % upper 
brace bay4

  % constraints on K-stubs on legs (x4), gg12(25:34,1:4)
   % gg12(25:26,:)  = 0; ('0' because legs are related to K-stubs 
above)
     gg12(27:28,1)  = [g_BUC_grad_D(1); g_BUC_grad_T(1)];  % legs 
bay1
     gg12(29:30,2)  = [g_BUC_grad_D(2); g_BUC_grad_T(2)];  % legs 
bay2
```

```
     gg12(31:32,3)   = [g_BUC_grad_D(3); g_BUC_grad_T(3)];  % legs
bay3
     gg12(33:34,4)   = [g_BUC_grad_D(4); g_BUC_grad_T(4)];  % legs
bay4


%% FLS constraints (x17), gg13 = [34x17] --> gg(:,30:46)
    gg13 = zeros(n_var,length(g13));

  % constraints on K-stubs on braces (x8)
     gg13(1 :2 ,1) = [g_FLS_grad_D(6 ); g_FLS_grad_T(6)];
     gg13(3 :4 ,2) = [g_FLS_grad_D(7 ); g_FLS_grad_T(7)];
     gg13(5 :6 ,3) = [g_FLS_grad_D(8 ); g_FLS_grad_T(8)];
     gg13(7 :8 ,4) = [g_FLS_grad_D(9 ); g_FLS_grad_T(9)];
     gg13(9 :10,5) = [g_FLS_grad_D(10); g_FLS_grad_T(10)];
     gg13(11:12,6) = [g_FLS_grad_D(11); g_FLS_grad_T(11)];
     gg13(13:14,7) = [g_FLS_grad_D(12); g_FLS_grad_T(12)];
     gg13(15:16,8) = [g_FLS_grad_D(13); g_FLS_grad_T(13)];

  % constraints on X-stubs on braces (x4)
     gg13(17:18,9)  =
[extremum_AtoB_FUNC(g_FLS(14),g_FLS_grad_D(14));
extremum_AtoB_FUNC(g_FLS(14),g_FLS_grad_T(14))];
     gg13(19:20,10) =
[extremum_AtoB_FUNC(g_FLS(17),g_FLS_grad_D(17));
extremum_AtoB_FUNC(g_FLS(17),g_FLS_grad_T(17))];
     gg13(21:22,11) =
[extremum_AtoB_FUNC(g_FLS(20),g_FLS_grad_D(20));
extremum_AtoB_FUNC(g_FLS(20),g_FLS_grad_T(20))];
     gg13(23:24,12) =
[extremum_AtoB_FUNC(g_FLS(23),g_FLS_grad_D(23));
extremum_AtoB_FUNC(g_FLS(23),g_FLS_grad_T(23))];
    %gg13(17:24,9:12)

 % constraints on K-stubs on legs (x5)
     gg13(25:26,13) = [g_FLS_grad_D(1); g_FLS_grad_T(1)];
     gg13(27:28,14) = [g_FLS_grad_D(2); g_FLS_grad_T(2)];
     gg13(29:30,15) = [g_FLS_grad_D(3); g_FLS_grad_T(3)];
     gg13(31:32,16) = [g_FLS_grad_D(4); g_FLS_grad_T(4)];
     gg13(33:34,17) = [g_FLS_grad_D(5); g_FLS_grad_T(5)];
    %gg13(25:34,13:17)


%% ALL GRADIENTS of NONLINEAR CONSTRAINTS
    gg = [gg11, gg12, gg13]; % [34 x 46] matrix, nonlinear ineq.
consrtaints
    gh = [ ];                % empty    matrix, equality
consrtaints


%% PLOT FLS CONSTRAINTS
    plot_calcNONLINEAR_CONSTRAINTS(g13, gg13, gen, DIM, upperLim)


  end  % of 'if' statement

 end  % of function
```

C7 – Buckling constraint (g_{10})

The function below calculates the Euler buckling constraint (g_{10}) and its 1st order gradient.

```
function [BUC, BUC_grad_D,BUC_grad_T]=
bucklingcheck(DIMr,LENr,FP,gen,lc,bays,legs,matfileTIM)

%=================================================================
% FUNCTION - buckling check + sensitivities (1st order gradients)
%==================================================================
% written by Daniel Zwick, 05/2014
% modified by Nikolaos Xyloudis, 06/2016


% CONTENT
% -----------------------------------------------------------------
% 1) Calculate maximum or critical force for Euler buckling of
specific beam element
% 2) Compare critical force with actual force


% INPUT parameters
% -----------------------------------------------------------------
%    DIMr      - reduced dimension matrix [m]
                  (eg.DIMr{1,2}=DIM{1,1}{1,2})
%    LENr      - reduce beam length matrix [m]
%    FP        - FEDEM and FATIGUE parameters
%    gen       - number of generation
%    lc        - event identification
%    bays      - number of bays = 4
%    legs      - number of legs


% OUTPUT parameters
% -----------------------------------------------------------------
%  BUC      - buckling check (<=0) for each member (legs, Ubraces,
Lbraces)
%  BUC_grad_D  - 1st order gradient of BUC with respect to D
%  BUC_grad_D  - 1st order gradient of BUC with respect to T


%% 1) Calculate maximum or critical force for Euler buckling of
specific beam element

    % Euler parameter, 4 chosen for fixed-fixed support conditions
        n_euler = 4; % [-]

    % Young's modulus
        Emod = 2.1*10^11; % [N/m^2]

    % for all bays, legs, lower braces and upper braces
        I      = zeros(bays,3);
        buc_dD = zeros(bays,3);
        buc_dT = zeros(bays,3);
```

```
 for b=1:bays
            for t=1:3
                % extract diameter and thickness
                    clear D T

                    if (t == 1) % legs between stubs:
            D = DIMr{1,2}(b,1); % D of legs between stubs [4x1]
            T = DIMr{1,2}(b,2); % T of legs between stubs [4x1]

                    else % braces between stubs:
                        D = DIMr{1,3}(b,2*t-3);
                        T = DIMr{1,3}(b,2*t-2);
                    end

% calculate moment of inertia
I(b,t)  = pi * (D^4 - (D-2*T)^4) / 64;

% calculate quantities for gradient calculation - NIKOS' update
buc_dD(b,t) = (-4*D^3 + 4*(D - 2*T)^3)/(D^4 - (D - 2*T)^4)^2;
buc_dT(b,t) = (-8*(D - 2*T)^3)/(D^4 - (D - 2*T)^4)^2;
            end
        end

% critical force for Euler buckling
        Feul      = zeros(bays,3);
        Feul_grad = zeros(bays,3);
        for b=1:bays
            for t=1:3
Feul(b,t) = (n_euler*pi^2 * Emod * I(b,t)) / (LENr(b,t))^2;


% calculate quantities for gradient calculation - NIKOS' update
Feul_grad(b,t) = (64*LENr(b,t)^2)/(n_euler*pi^3*Emod);
            end
        end

%% 2) Compare critical force with actual force

% load time series data
load(matfileTIM);  % Loads 'TIM' from
C:/Users/Nikolaos/Desktop/Project_Scripts NEW 09-
08/simdata/run128/matfiles/TIM_001_01.mat

    % calculate actual force of all members
        for b=1:bays
            % definitions
                legind = 0;
                br1ind = 0;
                br2ind = 0;
            % calculation
                for t=1:3
                    for e=1:4
                        % legs, lower and upper part
                            if ((t <= 2) && (e == 2))
                                for l=1:legs
                                    legind = legind + 1;
TIMmax_tmp{1,1}(b,legind) = max(abs(TIM{b,t,l,e,1}));
% 1-axial force
                                end
```

```
     % braces, lower part
       elseif (((t == 1) && (e >= 3)) || ((t == 3) && (e <= 2)))
         for l=1:legs
         br1ind = br1ind + 1;
         TIMmax_tmp{2,1}(b,br1ind) = max(abs(TIM{b,t,l,e,1}));
         end
     % braces, upper part
         elseif (((t == 2) && (e >= 3)) || ((t == 3) && (e >= 3)))
                                for l=1:legs
                                    br2ind = br2ind + 1;
        TIMmax_tmp{3,1}(b,br2ind) = max(abs(TIM{b,t,l,e,1}));
                                end
                            end
                    end
                end
        end

    % identify maximum force for each element in each bay
        Fmax = zeros(bays,3);
        for b=1:bays
         for t=1:3 % '1' for legs, '2' lower brace, '3' upper brace
           Fmax(b,t) = max(TIMmax_tmp{t,1}(b,:)); % Fmax = [4x3]
            end
        end

    % compare forces
        for b=1:bays
            for t=1:3
                % warning message
                    if (Feul(b,t) <= Fmax(b,t))
%                         fprintf('WARNING! - Critical force of
Euler buckling exceeded in bay %d, element %d
(bucklingcheck.m)\n',b,t);
                    end
            end
        end


%------------------------------------------------------------------
% Calculate BUC constraints and their gradients BUC_grad.

%  (12 at total = 3 categories for each of 4 bays), (BUC<=1)
    BUC = Fmax./Feul; % [4x3] = [4(bays) x 3(legs,Lbraces,Ubraces)]

    BUC_grad_D = Fmax.*Feul_grad.*buc_dD;  % dBUC/dD, [4x3] matrix
    BUC_grad_T = Fmax.*Feul_grad.*buc_dT;  % dBUC/dT, [4x3] matrix

end   % of function
```

C8 – Fatigue damage calculation with rainflow counting

```
function [Damage, Damage_grad_D, Damage_grad_T] =
                    damage(TIMsignal,SIGsignal,FP,submerged,lc,D,T)
===================================================================
% FUNCTION - Damage calculation
%            + sensitivities (gradients)
===================================================================
% written by Daniel Zwick,        01/2014
% modified by Nikolaos Xyloudis, 09/2016


% INPUT
% TIMsignal - time series of original forces/moments, converted to
unscaled HSS
% SIGsignal - time series of SIGMA stress, converted to scaled HSS
% FP        - simulation timedata
% submerged - 1 or 2 for seawater or air
% lc        - load case identification
% T         - thickness of the pipe
  DFF = 1;       % (from 'OS-C101 Section 6' Fatigue Limit States)
  eta = 1/DFF;   % 'DFF' = 1,2,3 ---> 'eta' = 1/1,1/2,1/3

% OUTPUT
%    Damage          - fatigue damage
%    Damage_grad_D   - damage gadient with respect to D
%    Damage_grad_T   - damage gadient with respect to T


% 1) rainflow counting algorithm
% -----------------------------------------------------------------
    rf     = inrain_main(SIGsignal);

% 2) calculating damage by Palmgren-Miner
% -----------------------------------------------------------------
% Fatigue calculation based on DNV-RP-C203, "Fatigue Design of
  Offshore Steel Structures", October 2012
%    - Table 2-2, S-N curve in seawater with cathodic protection, T
%    - Table 2-1, S-N curve in air, T
    switch submerged
        case 1                                      % Table 2-2
            FL = 10^6;
            m1=3.0;         log_a1=11.764;     % N < 10^6 cycles
            m2=5.0;         log_a2=15.606;     % N > 10^6 cycles
        otherwise                                   % Table 2-1
            FL = 10^7;
            m1=3.0;         log_a1=12.164;     % N < 10^7 cycles
            m2=5.0;         log_a2=15.606;     % N > 10^7 cycles
    end

% counting number of cycles and stress ranges
        [N, C] = hist(rf(1,:),FP{5}(1,5));

% scale number of cylces to lifetime
        Ns = N * FP{5}(1,6) * FP{7}(1,lc);

% convert stress range to [MPa]
        C  = C * 10^-6;
```

```
% thickness effect (DNV-RP-C203, eq. 2.4.3, page 17)
       k    = 0.25;
       tref = 0.032;

        if   (T <= tref)
           F  = 1;
           Tr = tref;  T1 only in thick. effect for gradient calc

        else
           F  = (T/tref)^k;
           Tr = T;     T1 only in thick. effect for gradient calc
        End
     C_t = C * F;

% calculation of total damage by given time series
        Dbin        = zeros(FP{5}(1,5),1);
        Dbin_grad_D = zeros(FP{5}(1,5),1);
        Dbin_grad_T = zeros(FP{5}(1,5),1);


for j=1:FP{5}(1,5)
%       FP{5}(1,5)=100=number of bins for Palmgren-Miner

clear  log_Na    Ni_grad_D    Ni_grad_T


% calculating the maximum allowable number of cycles to failure
  (Na), at constant stress range for each stress range
                if (Ns(j) <= FL)
                    log_Na    = log_a1 - m1 * log10(C_t(j));

% Calculate fatigue gradients analytically (local assumption)
Ni_grad_D =  0  + (-1/(D-T))*(m1/(10^log_a1))*C_t(j)^m1;

Ni_grad_T = ((m1*k*Tr^(m1*k-1))/(tref^(m1*k)) + ((2*T-D)/(D*T-
            T^2))*(Tr/tref)^(m1*k))*(m1/(10^log_a1))*C(j)^m1;

               else
                    log_Na    = log_a2 - m2 * log10(C_t(j));

% Calculate fatigue gradients analytically (local assumption)
Ni_grad_D = 0 + (-1/(D-T)) * (m2/(10^log_a2)) * C_t(j)^m2;

Ni_grad_T = ((m2*k*Tr^(m2*k-1))/(tref^(m2*k)) + ((2*T-D)/(D*T-T^2))
* (Tr/tref)^(m2*k)) * (m2/(10^log_a2)) * C(j)^m2;
                end

% calculating fraction of actual/allowable number of cycles
                Dbin(j,1) = Ns(j) / (10^log_Na);
                Dbin_grad_D(j,1) = Ns(j) * Ni_grad_D;
                Dbin_grad_T(j,1) = Ns(j) * Ni_grad_T;
        end

% summing up damage
Damage        = sum(Dbin(:,1)) / eta;
Damage_grad_D = sum(Dbin_grad_D(:,1)) / eta;
Damage_grad_T = sum(Dbin_grad_T(:,1)) / eta;

end   % of function
```

C9 – Function that updates DIM matrix with variables

```
function [DIM] = XtoDIM(gen,X,DIM)
%   This function updates the DIM matrix with the optimized X values

%   1. Reads existing 'DIM' matrix
%   2. Creates 'DIMnew' by replacing the X values in 'DIM

Dred = 0.6;     % proportion of member's dimensions between
                % stubs/cans (=1 means no difference)

% bay height definition [m] - NOT CHANGED
%    DIM{gen,1}{1,1} = DIM{gen,1}{1,1};           % [4 x 1] matrix

    % K-stubs on braces: D-lower, T-lower, D-upper, T-upper [m]
    % X(1,1:16)
      DIM{gen,1}{1,4}(1,:) = X(1,1:4);
      DIM{gen,1}{1,4}(2,:) = X(1,5:8);
      DIM{gen,1}{1,4}(3,:) = X(1,9:12);
      DIM{gen,1}{1,4}(4,:) = X(1,13:16);           % [4 x 4] matrix


% Replace with X values for D-long, T-long        % [6 x 4] matrix
      DIM{gen,1}{1,5}(1,1:2) = X(1,17:18);
      DIM{gen,1}{1,5}(2,1:2) = X(1,19:20);
      DIM{gen,1}{1,5}(3,1:2) = X(1,21:22);
      DIM{gen,1}{1,5}(4,1:2) = X(1,23:24);

    % copy values to lower and upper X-brace
      DIM{gen,1}{1,5}(:,3) = DIM{gen,1}{1,5}(:,1);
      DIM{gen,1}{1,5}(:,5) = DIM{gen,1}{1,5}(:,1);
      DIM{gen,1}{1,5}(:,4) = DIM{gen,1}{1,5}(:,2);
      DIM{gen,1}{1,5}(:,6) = DIM{gen,1}{1,5}(:,2);


% horizontal offset of K-joints along main leg line [m] - NOT
CHANGED
%       DIM{gen,1}{2,1} = DIM{gen,1}{2,1};    % [5 x 1] matrix


% K-stubs on legs: D, T on all bay division lines [m]
  DIM{gen,1}{2,2} = [X(25:2:33)', X(26:2:34)']; % [5 x 2] matrix


%% COPY TO LEGS & BRACES FROM NEW VARIABLES

% legs between stubs: D, T [m] equal to K-stub element above
  DIM{gen,1}{1,2} = Dred*DIM{gen,1}{2,2}(2:5,:);  % [4 x 2] matrix

% braces between stubs: D-lower, T-lower, D-upper, T-upper [m]
% braces equal to X-stub of brace part
 DIM{gen,1}{1,3} = Dred*DIM{gen,1}{1,5}(:,3:6);  % [4 x 4] matrix

end % of function
```

C10 – Create variable matrix from DIM matrix

```
function [ X ] = DIMtoX(gen,DIM)

%  From the DIM matrix, creates the X matrix of the variables of the
%  optimization process


%% OUTPUT
% D,T of 17 members arranged in X = [1x34] = [D T D T D....] where:
% X(1,1:16)= K-stubs on braces (8 elements, lower and upper per bay)
% X(1,17:24)= X-stubs on braces (4 elements, one per bay)
% X(1,25:34)= K-stubs on legs (5 elements, one per bay + one on top)


%% CREATE VARIABLE MATRIX
% (1) K-stubs on braces: D-lower, T-lower, D-upper, T-upper [m]
         Kstub_brace = (DIM{gen,1}{1,4})';
         X(1,1:16) = Kstub_brace(:)';

% (2) X-stubs on braces: D-long, T-long [m]
        Xstub_brace = DIM{gen,1}{1,5}(:,1:2)';
        X(1,17:24)   = Xstub_brace(:)';

% (3) K-stubs on legs: D, T on all bay division lines [m]
        Kstub_legs = DIM{gen,1}{2,2}';
        X(1,25:34) = Kstub_legs(:)';

End % of function
```

C11 – Corresponding value of max A to B

```
function [ extreme ] = extremum_AtoB_FUNC(A,B)

% ----------------------------------------------------------------*
%  This function finds the max value of matrix A, and returns the
%  value at the corresponding location at matrix B.
%
%  1. A and B must have the same dimensions!!!
%
%  2. If A is the absolute value of a matrix, then the max(A) is
%  the 'extremest' value in A.
% ----------------------------------------------------------------*
% written  by Nikolaos Xyloudis, 06/2016


[~,l] = max(A(:));
[r,c] = ind2sub(size(A),l);

extreme = B(r,c);

end  % of function
```