



NTNU – Trondheim
Norwegian University of
Science and Technology

Topology optimization of jacket support structures with genetic algorithm

Lucia Barcena Pasamontes
Fernando Gomez Torres

Coastal and Marine Civil Engineering

Submission date: Januar 2014

Supervisor: Michael Muskulus, BAT

Co-supervisor: Daniel Zwick, BAT

Norwegian University of Science and Technology
Department of Civil and Transport Engineering

Acknowledgements

This master thesis was undertaken at the Department of Civil and Transport Engineering (NTNU) with the Offshore Wind Turbine Technology group and marks the end of our studies in Civil Engineering.

We would like to express our sincere gratitude to our supervisor Prof. Michael Muskulus for his acceptance in his working group and his guidance and support through this study. And also to Daniel Zwick and Sebastian Schafhirt for their continuous advice during the realization of this project and for teaching us the short cuts and trick of programming.

And finally we would like to thank our families for the unconditional support over the years of our studies and specially for having made possible for us to be the last semester at NTNU.

Abstract

This thesis is a study of the optimization of the support structure for offshore wind turbines with a genetic algorithm. The optimization of these structures plays an important role in the plan for reduction of the costs proposed for the coming years in the field of offshore wind energy.

The support structure proposed to optimize in this study is based on the model designed for the OC4 project. This model consists of 4 legs and 4 levels of X-braces forming a lattice. The optimization proposed in this thesis is based on the minimization of the weigh and the damaget of the structure establishing as design variables the geometry (diameters and thickness of the beams) and topology (location nodes). Material properties will not change.

In order to perform this task, a genetic algorithm has been implemented in Matlab (Version R2013a). The variables are converted into binary code in order to combine two different designs. The pairs of designs selected to be combined are chosen randomly within the population and after the combination, or crossed over, two new designs are generated. After that, the new designs are evaluated based on a fitness function (weight or damage) and selected again, based on the fitness value. This is an iterative process and finishes when an optimal design is found. Mutation and immigration operators are used in order to increase the variety in the search space.

Every iteration, analysis of each design is performed with a complete wind turbine simulation for a loadcase in the time-domain. Data obtained from the simulation are used for stress calculation and checked fatigue and ultimate limits.

These results demonstrate that automatic optimization of wind turbine support structures is feasible with a genetic algorithm, even for complex structures such us the one analyzed in this thesis.

Resumen

Esta tesis es un estudio de la optimización de la estructura de soporte para aerogeneradores marinos. La optimización de este tipo de estructuras juega un papel importante en el plan de reducción de costes planteado para los próximos años en el campo de la energía eólica marina.

La estructura propuesta a optimizar en este estudio se basa en el modelo planteado por Rambøll A/S del proyecto CO4. Dicho modelo consta de 4 pilas (legs) cosidas a cuatro niveles por barras (braces) formando una celosía. La optimización propuesta en esta tesis se basa en la minimización del peso de la estructura estableciendo como variables de diseño la geometría (diámetros y grosores de las barras) y la topología (localización de los nodos). Las propiedades del material se mantienen constantes.

Para llevar a cabo esta tarea, se ha construido un algoritmo genético en Matlab (Versión R2013a) que permita generar nuevos diseños en una búsqueda continua por encontrar los más óptimos. Para ello se codifican las variables en formato binario con el objetivo de poder combinar dos diseños diferentes. La selección de los diseños que serán combinados dos a dos se hace de manera aleatoria. Tras la combinación se generan dos nuevos diseños que son evaluados en base a una función (peso o daño) y seleccionados de nuevo dependiendo del valor de esa función, entrando así en un proceso iterativo que finaliza con la consecución de un diseño óptimo. Los operadores mutación e inmigración han sido añadidos al proceso con el objetivo de ampliar la variabilidad dentro del espacio de búsqueda.

A cada iteración los nuevos individuos son implementados en un simulador para analizar su comportamiento estructural bajo unas determinadas condiciones de carga y durante un determinado espacio de tiempo. Con los datos arrojados por el simulador, se calculan las tensiones en los puntos críticos de la estructura y se comprueba si cumple ELS y ELF (estado límite último y estado límite de fatiga).

Los resultados demuestran que la optimización automática de estructuras es factible con el método del algoritmo genético, incluso para estructuras complejas como la considerada en este estudio.

INDEX

1	<i>Introduction.....</i>	15
1.1	Background	15
1.1.1	Structural optimization	15
1.1.2	Genetic algorithm theory.....	17
1.1.3	Advantages of genetic algorithm.....	26
1.2	Objective of study	27
1.3	Environmental conditions and properties of the jacket structure	28
2	<i>Genetic algorithm proposed.....</i>	33
2.1	Representation encoded.....	33
2.2	Development	34
2.2.1	Initialization.....	34
2.2.2	Perform time domain simulation	35
2.2.3	Fitness evaluation.....	35
2.2.4	Genetic operators.....	37
2.2.5	Flow chart of the genetic algorithm	39
2.3	Convergence of proposed algorithm.....	40
3	<i>Results.....</i>	43
3.1	Objective function based on the weight	45
3.1.1	Geometry optimization.....	45
3.1.2	Topology and geometry optimization.....	56
3.2	Objective function based on the damage	68
3.2.1	Geometry optimization.....	68
3.2.2	Topology and geometry optimization.....	80
3.3	Different objective function for each step	92
3.3.1	Geometry optimization.....	92
3.3.2	Topology and geometry optimization.....	104
4	<i>Conclusion</i>	117
5	<i>References.....</i>	121
6	<i>Appendix A. Fatigue Calculations</i>	127
7	<i>Appendix B. Optimization algorithm m-files. Main script</i>	139

LIST OF FIGURES

<i>Figure 1-1. Flow chart of a simple genetic algorithm</i>	22
<i>Figure 1-2. Detail of pile head (left) and transition part (right) [49]</i>	29
<i>Figure 1-3. Front view of the support jacket structure [49].</i>	29
<i>Figure 1-4. Member properties of UpWind Jacket Model [49].</i>	30
<i>Figure 2-1. Example of crossover and mutation operators. For simplicity only two cuts (grey vertical lines) are shown and one mutation (red circles), and only 25 bits of the chromosome are shown</i>	38
<i>Figure 2-2. Flow chart of proposed genetic algorithm</i>	40
<i>Figure 2-3. The challenges of binary thresholds</i>	41
<i>Figure 3-1. Weight of the structure [kg], when optimizing the weight (224 generations and 9 individuals)</i>	45
<i>Figure 3-2. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight (224 generations and 9 individuals)</i>	46
<i>Figure 3-3. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight (224 generations and 9 individuals)</i>	47
<i>Figure 3-4. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight (224 generations and 9 individuals)</i>	48
<i>Figure 3-5. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight (224 generations and 9 individuals)</i>	49
<i>Figure 3-6. Improvement of the fitness through the generations when optimizing the weight (224 generations and 9 individuals)</i>	50
<i>Figure 3-7. Number of designs of bay 1 that change from one generation to the next one when optimizing the weight (224 generations and 9 individuals)</i>	51
<i>Figure 3-8. Number of designs of bay 2 that change from one generation to the next one when optimizing the weight (224 generations and 9 individuals)</i>	51
<i>Figure 3-9. Number of designs of bay 3 that change from one generation to the next one when optimizing the weight (224 generations and 9 individuals)</i>	52
<i>Figure 3-10. Number of designs of bay 4 that change from one generation to the next one when optimizing the weight (224 generations and 9 individuals)</i>	52
<i>Figure 3-11. Gamma constriction for the legs when optimizing the weight (224 generations and 9 individuals)</i>	53
<i>Figure 3-12. Gamma constriction for the braces when optimizing the weight (224 generations and 9 individuals)</i>	54
<i>Figure 3-13. Beta constrain for legs and braces when optimizing the weight (224 generations and 9 individuals)</i>	54
<i>Figure 3-14. Tau constrain for legs and braces when optimizing the weight (224 generations and 9 individuals)</i>	55
<i>Figure 3-15. Weight of the structure [kg], when optimizing the weight (50 generations and 10 individuals)</i>	56
<i>Figure 3-16. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight (50 generations and 10 individuals)</i>	57
<i>Figure 3-17. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight (50 generations and 10 individuals)</i>	58
<i>Figure 3-18. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight (50 generations and 10 individuals)</i>	59
<i>Figure 3-19. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight (50 generations and 10 individuals)</i>	60

Figure 3-20. Location of the nodes of the 10 best designs when optimizing the weight (50 generations and 10 individuals)	61
Figure 3-21. Improvement of the fitness through the generations when optimizing the weight (50 generations and 10 individuals)	61
Figure 3-22. Number of designs of bay 1 that change from one generation to the next one when optimizing the weight (50 generations and 10 individuals)	62
Figure 3-23. Number of designs of bay 2 that change from one generation to the next one when optimizing the weight (50 generations and 10 individuals)	62
Figure 3-24. Number of designs of bay 3 that change from one generation to the next one when optimizing the weight (50 generations and 10 individuals)	63
Figure 3-25. Number of designs of bay 4 that change from one generation to the next one when optimizing the weight (50 generations and 10 individuals)	63
Figure 3-26. Gamma constriction for the legs when optimizing the weight (50 generations and 10 individuals)	64
Figure 3-27. Gamma constriction for the braces when optimizing the weight (50 generations and 10 individuals)	65
Figure 3-28. Beta constrain for legs and braces when optimizing the weight (50 generations and 10 individuals)	65
Figure 3-29. Tau constrain for legs and braces when optimizing the weight (50 generations and 10 individuals)	66
Figure 3-30. Weight of the structure [kg], when optimizing the damage (200 generations and 9 individuals)	68
Figure 3-31. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the damage (200 generations and 9 individuals)	69
Figure 3-32. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the damage (200 generations and 9 individuals)	70
Figure 3-33. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the damage (200 generations and 9 individuals)	71
Figure 3-34. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the damage (200 generations and 9 individuals)	72
Figure 3-35. Improvement of the fitness through the generations when optimizing the damage (200 generations and 9 individuals)	73
Figure 3-36. Number of designs of bay 1 that change from one generation to the next one when optimizing the damage (200 generations and 9 individuals)	74
Figure 3-37. Number of designs of bay 2 that change from one generation to the next one when optimizing the damage (200 generations and 9 individuals)	74
Figure 3-38. Number of designs of bay 3 that change from one generation to the next one when optimizing the damage (200 generations and 9 individuals)	75
Figure 3-39. Number of designs of bay 4 that change from one generation to the next one when optimizing the damage (200 generations and 9 individuals)	75
Figure 3-40. Gamma constriction for the legs when optimizing the damage (200 generations and 9 individuals)	76
Figure 3-41. Gamma constriction for the braces when optimizing the damage (200 generations and 9 individuals)	77
Figure 3-42. Beta constrain for legs and braces when optimizing the damage (200 generations and 9 individuals)	78
Figure 3-43. Tau constrain for legs and braces when optimizing the damage (200 generations and 9 individuals)	79
Figure 3-44. Weight of the structure [kg], when optimizing the damage (50 generations and 10 individuals)	80

Figure 3-45. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the damage (50 generations and 10 individuals) _____ 81

Figure 3-46. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the damage (50 generations and 10 individuals) _____ 82

Figure 3-47. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the damage (50 generations and 10 individuals) _____ 83

Figure 3-48. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the damage (50 generations and 10 individuals) _____ 84

Figure 3-49. Location of the nodes of the 10 best designs when optimizing the weight (50 generations and 10 individuals) _____ 85

Figure 3-50. Improvement of the fitness through the generations when optimizing the damage (50 generations and 10 individuals) _____ 85

Figure 3-51. Number of designs of bay 1 that change from one generation to the next one when optimizing the damage (50 generations and 10 individuals) _____ 86

Figure 3-52. Number of designs of bay 2 that change from one generation to the next one when optimizing the damage (50 generations and 10 individuals) _____ 86

Figure 3-53. Number of designs of bay 3 that change from one generation to the next one when optimizing the damage (50 generations and 10 individuals) _____ 87

Figure 3-54. Number of designs of bay 4 that change from one generation to the next one when optimizing the damage (50 generations and 10 individuals) _____ 87

Figure 3-55. Gamma constriction for the legs when optimizing the damage (50 generations and 10 individuals) _____ 88

Figure 3-56. Gamma constriction for the braces when optimizing the damage (50 generations and 10 individuals) _____ 89

Figure 3-57. Beta constrain for legs and braces when optimizing the damage (50 generations and 10 individuals) _____ 90

Figure 3-58. Tau constrain for legs and braces when optimizing the damage (50 generations and 10 individuals) _____ 91

Figure 3-59. Weight of the structure [kg], when optimizing the weight and damage (200 generations and 9 individuals) _____ 92

Figure 3-60. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight and damage (200 generations and 9 individuals) _____ 93

Figure 3-61. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight and damage (200 generations and 9 individuals) _____ 94

Figure 3-62. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight and damage (200 generations and 9 individuals) _____ 95

Figure 3-63. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight and damage (200 generations and 9 individuals) _____ 96

Figure 3-64. Improvement of the fitness through the generations when optimizing the weight and damage (200 generations and 9 individuals) _____ 97

Figure 3-65. Number of designs of bay 1 that change from one generation to the next one when optimizing the weight and damage (200 generations and 9 individuals) _____ 98

Figure 3-66. Number of designs of bay 2 that change from one generation to the next one when optimizing the weight and damage (200 generations and 9 individuals) _____ 98

Figure 3-67. Number of designs of bay 3 that change from one generation to the next one when optimizing the weight and damage (200 generations and 9 individuals) _____ 99

Figure 3-68. Number of designs of bay 4 that change from one generation to the next one when optimizing the weight and damage (200 generations and 9 individuals) _____ 99

Figure 3-69. Gamma constriction for the legs when optimizing the weight and damage (200 generations and 9 individuals) _____ 100

Figure 3-70. Gamma constriction for the braces when optimizing the weight and damage (200 generations and 9 individuals)	101
Figure 3-71. Beta constrain for legs and braces when optimizing the weight and damage (200 generations and 9 individuals)	102
Figure 3-72. Tau constrain for legs and braces when optimizing the weight and damage (200 generations and 9 individuals)	103
Figure 3-73. Weight of the structure [kg], when optimizing the weight and damage (50 generations and 10 individuals)	104
Figure 3-74. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight and damage (50 generations and 10 individuals)	105
Figure 3-75. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight and damage (50 generations and 10 individuals)	106
Figure 3-76. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight and damage (50 generations and 10 individuals)	107
Figure 3-77. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight and damage (50 generations and 10 individuals)	108
Figure 3-78. Location of the nodes of the 10 best designs when optimizing the weight and damage (50 generations and 10 individuals)	109
Figure 3-79. Improvement of the fitness through the generations when optimizing the weight and damage (50 generations and 10 individuals)	109
Figure 3-80. Number of designs of bay 1 that change from one generation to the next one when optimizing the weight and damage (50 generations and 10 individuals)	110
Figure 3-81. Number of designs of bay 2 that change from one generation to the next one when optimizing the weight and damage (50 generations and 10 individuals)	110
Figure 3-82. Number of designs of bay 3 that change from one generation to the next one when optimizing the weight and damage (50 generations and 10 individuals)	111
Figure 3-83. Number of designs of bay 4 that change from one generation to the next one when optimizing the weight and damage (50 generations and 10 individuals)	111
Figure 3-84. Gamma constriction for the legs when optimizing the weight and damage (50 generations and 10 individuals)	112
Figure 3-85. Gamma constriction for the braces when optimizing the weight and damage (50 generations and 10 individuals)	113
Figure 3-86. Beta constrain for legs and braces when optimizing the weight and damage (50 generations and 10 individuals)	114
Figure 3-87. Tau constrain for legs and braces when optimizing the weight and damage (50 generations and 10 individuals)	115
Figure 6-1. Geometrical definitions for tubular joints [15]	129
Figure 6-2. Definition of geometric parameters for Y-joints and X-joints [15]	130
Figure 6-3. Definition of geometric parameters for K-joints [15]	130

1 Introduction

The design of offshore wind turbine poses challenging issues because this type of structures are highly dynamic [34] [50]. In particular, the design of support structures for wind turbines is a nontrivial task [39] due to the dynamic loads affecting this type of structure.

The structural behavior of the wind turbine is nonlinear due to unsteadiness in the flow, and the significant interaction between rotor dynamics and the motions of the support structures, for example, due to aerodynamic damping [37]. If a structural analysis aims to be accurate, it has to be performed in the time domain and it has to be integrated, which means, it is necessary to consider the complete, fully-coupled wind turbine. This is very important for reliable and cost-effective design [29] [40] [6].

Reduction of the cost of energy is a major challenge in offshore wind energy. This reduction must be carried out through the optimization of all the phases of a wind energy project. Current scenarios ask for a cost reduction of at least 20 percent, and for the year 2020 the reduction of the costs is expected to reach a value of 30-40 percent [16] [43] [44]. In spite of the support structure and foundation contribution only around 17 percent of total capital costs, this is an area with high potential for cost reduction [43] [44].

The structural optimization through computational tools, applied to wind turbine support structures can be a major enabling technology to realize these goals.

1.1 Background

1.1.1 Structural optimization

Structural optimization, as an engineering area, seeks to determine the most economical geometrical shapes satisfying the constraints imposed on the design [11], and it has always been an important concern when designing because finding better and more efficient designs is the task of an engineer.

Before the use of the computers the structures depended on the experience and intuition of an engineer [11]. Galileo Gallilei seems to be the first scientist who focused on the optimization theory in 1638 when he analyzed the optimum shape of a cantilever beam subject to a point load at its free end.

From those days, some scientists have contributed through their studies to the development of the structural optimization. In the beginning, the contributions were very theoretical and the results were not enough successful due to the lack of computational tools. This usually led to oversize the design in order to ensure a good response.

But in the last few decades, when powerful computers together with new structural analysis methods were developed, the structural optimization became more accurate and faster. As a result, the interest in this field has increased notably [42].

During these decades, there were a lot of researches focused on finding new methods to optimize structures which are subject to static loadcases [22] [10] [1]. The most efficient methods are gradient-based, but for transient problems (optimization of a structure for wind turbine), these methods are quite complex to apply [8] [9] [27].

The early literature on optimization of wind turbines has considered mainly static loadcases. For example, Bazeos et al [3] describe a detailed tower design for a 450 kW turbine using a single static loadcase, and Lavassas et al [30] considered eighteen different static loadcases for a 1.0 MW machine. Long et al [31] proposed optimization of the distance between the bottom-leg for a lattice-tower with a similar set of static loadcases. Torcinaro et al [46] extended the analysis by also including buckling and eigenfrequency checks.

On the other hand, frequency-domain has been used by some authors, particularly for preliminary design. For example, Thiry et al [45] used a genetic algorithm in order to optimize a monopole design, but they assumed a rigid rotor and no aerodynamic damping. Long and Moe [32] optimized the distance between bottom-leg for a lattice-tower, implementing aerodynamic damping as a linear dashpot element.

In the last years, integrated wind turbine simulations in the time domain have been used in order to optimize support structures. These studies are simulation-based [20]. Either the design sensitivities are obtained by finite-differences [2], or the optimization is performed with a gradient-free method. Ashuri [2] demonstrated the potential of integrated design and optimization using a set of thirteen loadcases of ten minutes each. He tested it for a typical North Sea site and the results showed that by increasing the cost of both the support structure and the rotor, and keeping the rating fixed to 5 MW, the cost

of energy could be decreased by a few percent. The convergence was reached in less than 5 iterations, but each of these iterations took almost two hours, fully parallelized, and only a circular tower was considered. Yoshida [54] optimized a circular tower of wind turbine using a genetic algorithm. Zwick et al [55] describe a novel algorithm that sizes each structural member independently (assuming that loads do not change significantly) within a certain tolerance, for a complex multi-membered support structure around 30-40 iterations are needed for the convergence, but only ten min loadcase was used for this method.

1.1.2 Genetic algorithm theory

1.1.2.1 Evolution theory

The theory of evolution was described by Charles Darwin [12] in 1959. The theory consists mainly on the evolution based on natural selection. In a hostile environment, only the fittest individual will survive. If an individual is not fitness enough will be removed. However, Darwin didn't know what the basis of the heredity was.

Some years later, in 1865, Mendel discovered that the characters were inherited discretely and were taken from either one parent or other, depending on their nature (dominant or recessive). This characters that could take several values were called gens (basic coding unit). The goal of his work was the set of the three laws of inheritance [33].

In 1882, Flemming investigated the process of cell division and distribution of the chromosomes in the nucleus. He described the chromosomes as filaments in which, chromatin from the nucleus was added during the cell division [17]. However, Flemming did not know the work of Mendel about inheritance, therefore he could not make the connection between his investigations and the genetic inheritance.

Watson and Crick [51] found in 1953 that the molecular basis of the gene is DNA (deoxyribonucleic acid). Chromosomes are composed of DNA, which means that the genes are in the chromosome.

All this facts constitute the theory of neo-Darwinism, in which, the history of the life is caused by a number of processes (reproduction, mutation, competition and selection) acting within populations.

1.1.2.2 Origin of the genetic algorithm

The genetic algorithm is a search technique based on the theory of evolution of Darwin. Since the individuals with best fitness in a population will survive to the next generation thanks to the capacity of adaptation.

The early studies on this method were carried out at the end of 50's decade by Box [5] and Friedman [19]. But due to the lack of computational tools, this discipline was not developed successfully until the 70's when the professor of electrical engineering and computer sciences, John Holland [24] started to develop this field at the University of Michigan.

At the University, Holland taught a course entitled "System Theory adaptation". In this course, his ideas about the implementation of the genetic algorithm started to be developed. For the development of these ideas the contribution of some students was very important.

The main objectives of Holland and his colleagues [13] were:

1. Explaining rigorously the adaptive process of natural processes
2. Designing artificial systems (programs) based on the natural systems

Some years later, David Edward Goldberg, a student of the University of Michigan, contacted with professor Holland in order to apply the genetic algorithm on industrial problems. In spite of the complexity of the problem, Goldberg finally reached his objective and he became one of the first engineers who applied this optimization method for this type of problem. After him, other applications of Holland also managed to create others application for the genetic algorithm and progressively this method started to get importance in the field of optimization process.

Finally, in 1985 The First International Conference on Genetic Algorithms was held in Pittsburgh, Pennsylvania and the original technique invented by Holland called "reproductive plans" become popular under the name "genetic algorithm".

1.1.2.3 Biological basics of genetic algorithm

All the organisms are formed by either one or more cells, and these cells have either one or more *chromosomes*. One chromosome can be divided in *gens* which are the responsible to define the characteristics of an individual such as color eyes, hair, etc. The different values a characteristic can take are

called *allele* [14]. And the location where an allele takes place within the chromosome is called *locus*. *Genotype* is the genetic constitution that an individual has with respect to a particular characteristic [36], and the *phenotype* is how this genotype is expressed in an individual (eyes or hair colour).

Most of the species store the chromosomes in pairs, which means that they are diploids. The human being, for example, each cell of the body has 23 pairs of chromosomes. By the reproduction of two individuals, crossover or recombination of chromosomes (genetic information) is carried out between the parents. The recombination consists of an exchange of genetic material between two individuals (father and mother). On the other hand, a mutation can occur (due to an error in the inheritance) and it is possible to find some genetic information in the kid that has not been inherited from the parents.

Once the biological terms are explained, it is easier to understand the similarities between biological world and the algorithm. In the following table, it is possible to see the correspondence between elements in biological terms and algorithm terms [48].

Natural System	Genetic Algorithm
Chromosome	String
Gen	Parameter or variable of the problem
Allele	Value of the parameter
Locus	Place within string
Genotype	Value coded
Phenotype	Value decoded
Individual	Solution
Generation	Cycle

Table 1-1. Correspondence between elements in biological and algorithm terms

In the genetic algorithms a chromosome is one solution of the problem. This chromosome is encoded by a string of bits. The most common representation is a binary code although there are different types of representation.

1.1.2.4 Representation encoded

When a genetic algorithm is designed to solve a specific problem, choosing the type of representation code is not an easy task. In general, for one problem different representation encoded can be used and all of them are able to carry out the process of optimization. But, the results obtained from all and

each of them can be very different despite of the operators used in the algorithm process being the same [48].

One of the most important decisions to take in this point is the type of codification. The representation encoded generally, can be either phenotypic or genotypic representation. If the genotypic representation is chosen, it is essential to assure a perfect connection between the genotype and phenotype values. To carry out this perfect connection the following conditions must be satisfied [48]:

- The correspondence between code representation and the proposed solution decoded must maintain a univocal relationship.
- Any alteration, perturbation or combination of the code representation must correspond to a solution of the problem.
- Any possible solution of the problem must have a valid value in the representation encoded space.
- Small variations in the genotype space must correspond with small modifications in the phenotype space.

The main advantage of choosing a phenotypic codification is that there is no need to establish any correspondence between two different levels (genotype and phenotype level) satisfying directly the 4 conditions mentioned previously. However, the main disadvantage is the difficulty of the operators. They must be elaborated very carefully.

Another classification criterion is the way the chromosome is organized in space [48]. Under this point of view, the representation encode can be classified in lineal and not lineal. The former is represented by a one-dimensional code, like a list or string. This string can be formed by binary values, integer values, etc. The latter, not lineal, is represented by a structure of two or more dimensions organized in trees, matrix, etc

Looking at the chromosome characteristics, the length of them is another important property. The length of the chromosome can stay constant or variable. In this case, the individuals can be represented by different length strings [53].

1.1.2.5 Development of the genetic algorithm

1.1.2.5.1 Structure of a genetic algorithm

The genetic algorithm starts from a set of individual called initial population. This population is generated randomly within reasonable values of the total search space. Then, the population is evaluated by a fitness function. Depending on the result of this evaluation, the individuals are selected to be combined and therefore, allowed to transfer their characteristics to the next generation which means that they will be selected for the crossover and mutation process. Once the new generation, is created, the process is repeated again until convergence is detected or when a specified maximum number of generations are reached [23].

The main steps to develop a genetic algorithm are described as follows [48]:

1. Decide the type of representation for the variables, the function fitness and the type of genetic operators.
2. Create an initial set (population) formed by n solutions (individuals), where n is the population size.
3. Evaluate the fitness for each individual.
4. Get a new population of individuals by repeating the following three steps:
 - a. Selection: choose two individuals (parents) of the population based on the fitness.
 - b. Crossover: the selected parents are combined in order to obtain two new individuals.
 - c. Mutation: with a probability lower than the mutation rate, any allele of the new individual can be mutated.
5. The best individuals are selected in order to form the population for the next generation.
6. Stop the process if the stop condition is reached. If not, go back to step 3. The stop condition can be either a maximum number or generation, or the achievement of the optimum solution.

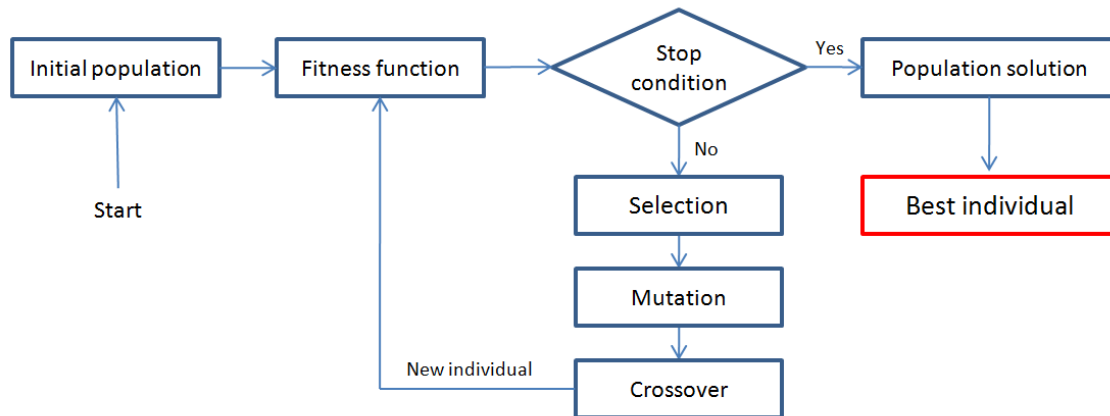


Figure 1-1. Flow chart of a simple genetic algorithm

1.1.2.5.2 Phases of a genetic algorithm

The genetic algorithm can be divided in 5 different phases to analyze easily how it works [23]. In the three first phases, decimal values for the variables are used without need to encode them. But for the fourth and fifth steps, the values have been encoded into binary code.

1.1.2.5.2.1 Initialization

It is the generation of the initial population of individuals. This population is created randomly within a range of values. But before that, two parameters must be set: the size population and the encoding representation.

The population size has an important influence on the efficiency of the algorithm so it is important to select it judiciously. If the size population is too small, cover the entire search space will be complicate, which means that the variability rate will be very low. A low variability can make the algorithm converge into a local optimum instead of a global optimum. This is the result of a premature convergence. On the other hand, a large population size decreases the rate of convergence, so the premature convergence problem disappears but in the worst scenario, if the population size is too large, divergence can happen. Besides, the higher the population size, the longer the processing time.

Regarding the encoded representation, the conditions that one code must satisfy to obtain success results from the optimization process is explained in the next sections. Since for the same problem, different representation can be used, it is important to choose one that fits properly to the type of problem and allows an efficient development of the optimization.

1.1.2.5.2.2 Evaluation

The objective of this phase is to determinate the suitability of each individual as the solution of the problem. To carry it out, a function called “fitness function” is defined. All the individuals are evaluated by this function and obtain a fitness value. But, before obtaining the fitness value, some constraints need to be satisfied. These constraints usually consist on simple inequations which let to establish some boundary conditions for the search of the optimum value of the optimization problem.

If the constraints are not satisfied, the individual obtains the minimum value of fitness which means that will not be selected for the crossover for the next generation. Otherwise, if the constraints are satisfied, the fitness of the individual is evaluated based on the fitness function. This fitness function depends on some variables and the goal of the GA is to obtain the best value for this function changing the value of these variables.

Compared with the natural world, the fitness function would be the capacity or ability of one individual. This ability will give to the individual the tool to survive in the environment. Therefore, the fitness function establishes the basis for selection of parents that will be mated together during the reproduction [7].

1.1.2.5.2.3 Selection

In this phase is defined the number of offspring that this individual will produce into the next generation [7] [48].

The number of offspring depends on the fitness of each individual. If an individual has a high fitness, the number of offspring that this individual will have for the crossover process will be higher. This means that the individual with higher fitness value, have higher probability of contributing one or more offspring in the next generation.

Therefore, the main objective of the selection process is taking the best solutions of the population and displacing out of this the individuals with lower fitness value (Victoria, 2006)

In this phase is important to decide if the selection process will have impact on the population size. Under the point of view, there are two options:

1. Population size keeps constant throughout the generations

2. Population size increases throughout the generations. The population size is equal to the previous population, plus the new offspring.

Studying the different types of selection for a genetic algorithm is not the goal of this work. However, the main concepts of some of them are presented as follow [48]:

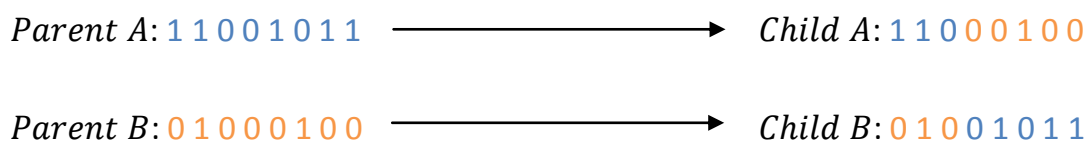
- Elitist selection: the selection of the fittest individual of each generation is guaranteed for the next generation.
- Roulette wheel Selection: A rate probability is assigned to the individual based on the fitness.
- Tournament Selection. Subgroups of individuals are formed within the population. This way, the members of each subgroup compete. The best individual of each group are selected.
- Generational selection: the individuals selected are always crossed over, which means that none of the individual is copied into the next generation directly as they are. They are always modified through the crossover operator.

1.1.2.5.2.4 Crossover

In this phase the genetic information is recombined with the intention of exploring new points in the space solution and reaching new and better results. Here the gene information contained in two selected parents is crossed over to generate two new children.

Like selection, there are different crossover operators. But, in general, the main idea is that some portions of the selected strings (as parents) are switched in order to generate two children [14] [41].

- Single point crossover: It is the simplest method. A crossover site is selected randomly between 1 and the string length. After that, the substrings of the parents, after and before the crossover site, are exchanged, generating the new two children. In the example, the crossover site is 3.



- Multipoint crossover: This method is very similar to the previous one. In this case, two (or more) crossover sites are selected. The crossover is carried out between these two (or more) points instead of one point. Supposing that the crossover site are 2 and 5, the children would be as follow:

Parent A: 1 1 0 0 1 0 1 1 \longrightarrow *Child A:* 1 1 0 0 0 0 1 1

Parent B: 0 1 0 0 0 1 0 0 \longrightarrow *Child B:* 0 1 0 0 1 1 0 0

- Uniform crossover: A random binary string is created. This string is called “mask”. This mask doesn’t mean anything, it is just a string of “0” and “1” with the same string length than the individuals [23]. With the information of the mask the children is created as follow:

Mask: 0 0 1 0 1 1 0 1

Parent A: 1 1 0 0 1 0 1 1 \longrightarrow *Child A:* 1 1 0 0 0 1 1 0

Parent B: 0 1 0 0 0 1 0 0 \longrightarrow *Child B:* 0 1 0 0 1 0 0 1

If the position i of the mask is “0”, $Child\ 1 = parent\ 1$ and $Child\ 2 = parent\ 2$ for that position. Otherwise, if the value is “1”, $Child\ 1 = parent\ 2$ and $Child\ 2 = parent\ 1$.

1.1.2.5.2.5 Mutation

This operator is introduced in order to encourage the development of new genetic material [26]. The mutation gives to the algorithm the chance to explore new points in the search space that has not been reached before or has been removed during the optimization process but still are not so bad (it is possible that a specific gene of one chromosome offers good results but since the other genes are bad, the whole individual gets poor fitness value). Therefore this operator increases the diversity of the population and has the effect inhibiting the possibility of converging to a local optimum instead a global optimum [23].

The mutation operator consists of just a random alteration of a bit value at a particular bit position in one string. As follow is showed how this operator works.

Original string: 0 1 0 1 1 0 0

Mutation site: 4 (for example)

String after mutation: 0 1 0 0 1 0 0

In order to control this operator, the probability of mutation is defined. This probability establishes the possibility of a bit of a string to be mutated [48]. If there is not mutation, the children are generated automatically after the crossover without any change. If mutation is carried out, one or more bits of the string are modified.

There are many works about the value of the mutation probability. But the results are very different ones from others, so it is difficult to find a universal valid value for all the genetic algorithms. It will depend on the type of the problem. For example, De Jong recommends a value of 0.001 [13], however, Grefenstette considers 0.01 [21], and Schaffer et al proposes a value from 0.005 to 0.01 [38].

On the other hand, Bach and Schutz [4] calculated a theoretic formula that modifies the value of the mutation probability based on the number of generation.

$$P_m = \left(2 + \frac{L - 2}{N} \cdot g \right)^{-1}$$

Where:

g = Generation

L = String length

N = Maximum number of generations

1.1.3 Advantages of genetic algorithm

Nowadays, the algorithms based on the biologic evolution of the organisms are considered as a robust and powerful search method. The genetic algorithm is not the only method based on biologic evolution. For instance, Evolution strategies (ES) is a method developed by Rechenberg in 1965 in order to find a method for numerical optimization [35]. Other examples of evolutionary algorithms are the Genetic Programming (GP) proposed by Koza [28] and Evolutionary programming (EP) created by Fogel [18].

Within the different methods based on the biologic evolution, the genetic algorithms have the best theoretical basis, and are better adapted biologically [48]. An important characteristic of these algorithms is that the approach is independent of the type of problem. This characteristic makes the genetic algorithm robust, and therefore, useful for any kind of problem. The main advantages are presented as follows:

- They are algorithms simple to understand and easy to implement.
- Specific knowledge about the problem where they are to be applied is not necessary.
- They consider some solutions of the problem simultaneously instead of working sequentially as the traditional methods.
- They work with design variables encoded. This way, the use of continuous or discrete type is allowed.
- The possibility of working with a large number of variables at the same time.
- There is an appropriate balance between the exploration and exploitation of the possible solutions.

The robustness of this method is an advantage due to the variety of problems where it can be applied to, but at the same time it is a disadvantage due to the lack of specialization on the type of problem. In spite of that, after years of development and implements on this method, the experience shows that the results obtained are successful and the computational cost is relatively competitive [48]. For this reason, the Genetic algorithms are becoming one of the most popular computational methods in the field of structures' optimization. Particularly, this method is very suitable for the optimization of wind turbine structure because it offers the chance to optimize complex structures with many parameters (such as the one considered in this study) in a relatively simple way.

1.2 Objective of study

The global aim of this master study is to develop and implement a genetic algorithm that allows the optimization of the topology and geometry of the jacket structure designed by Rambøll A/S [47] for the UpWind project [49].

And as partial objectives this report will go through:

- Developing a complete fatigue analysis based on Fedem response data and checking if the ultimate state limit is exceeded.
- Implementing the genetic algorithm with different initial parameters so the simulations are run with the most convenient input data.
- Implementing the genetic algorithm with different ways to calculate the objective function:
 - Optimization of the weight checking that the structure does not fail
 - Optimization of the damage of those designs that weigh less than the maximum value given as an input.
 - Alternatively use (one step at a time) weight optimization and damage optimization.
- Implementing three different simulations of the genetic algorithm:
 - Optimization of the locations of the nodes (diameter and thickness will remain constant).
 - Optimization of the diameter and thickness of the beams (the location of the nodes will not change).
 - Optimization of the locations of the nodes together with the diameter and thickness of the beams
- Finding the way to reduce the waiting time when combining the Fedem process with the fatigue analysis and the genetic algorithm.

1.3 Environmental conditions and properties of the jacket structure

The support structure that has been analyzed was defined, by Rambøll A/S [47] in the UpWind project for the 5MW baseline turbine as mentioned above. The four legged jacket has four levels of X-braces, accordingly mud braces, four central piles and a rigid block of concrete between jacket and tower, which represents the transition piece (Figure 1.2). The jacket legs are slightly inclined and the total height of the jacket from the mudline including the transition part is 70.15 meters, excluding the tower [49].

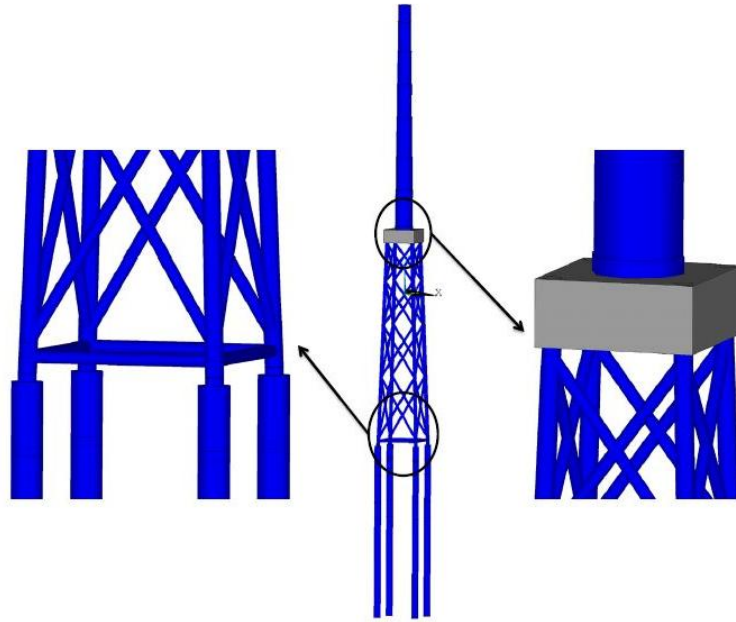


Figure 1-2. Detail of pile head (left) and transition part (right) [49]

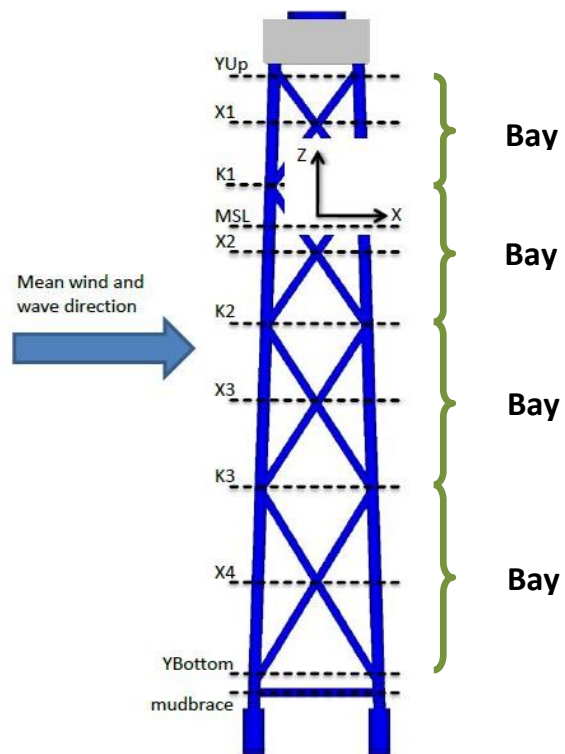


Figure 1-3. Front view of the support jacket structure [49].

Z-coordinate of levels [m]	
YUp	15.615
X1	10.262
K1	4.378
MSL (mean sea level)	0
X2	-1.958
K2	-8.922
X3	-16.371
K3	-24.614
X4	-33.373
YBottom	-43.127
Mudbrace	-44.001

Table 1-2. Z-location of levels of the support jacket structure

The geometry properties of the tubular members are described in Table 1-3 and Figure 1-4.

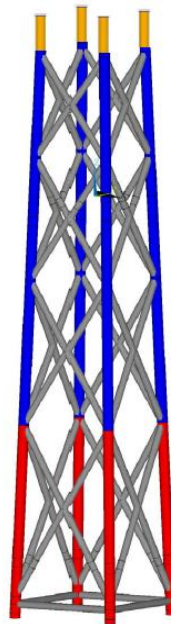


Figure 1-4. Member properties of UpWind Jacket Model [49].

Component	Color	Outer diameter [m]	Thickness [m]
X- and mud braces	Grey	0.8	20
Legs bays 2,3,4	Blue	1.2	35
Leg bay 1	Red	1.2	50
Leg crossing TP	Orange	1.2	40

Table 1-3. Geometric values assigned to each member of the structure

The steel properties that will be applied for the whole jacket are:

$$\rho_s = 7850 \text{ kg}/\text{m}^3 \quad E_s = 2,1 \cdot 10^{11} \text{ N}/\text{m}^2 \quad \nu_s = 0,3$$

The other aim of this section is to describe the environmental conditions that the support jacket structure is going to be exposed to. On one hand, these conditions depend on the water velocity, which we will be assumed as 10 m/s and on the other hand, on the wind speed. For the latter, a value of 10 m/s has been chosen since this seems a good compromise between on the one hand fatigue loading from waves (relevant mostly for lower wind speeds) and, on the other hand, fatigue loading from rotor excitations (higher for larger wind speeds). The environmental conditions, according to "UpWind Bases Design [6]", we are considering are:

$$\text{Significant wave height } (H_s [\text{m}]) = 1.48$$

$$\text{Peak spectral period } (T_p [\text{s}]) = 5.74$$

$$\text{Peakness for fatigue } [-] = 1$$

The jacket model used in this study is not fully realistic, e.g., being clamped at the mudline and, thereby being artificially too rigid. However, it is representative for the complexity of the problem in terms of the design space, and also, in terms of the dynamics of such structures.

2 Genetic algorithm proposed

The tuning of the genetic algorithm for this study has been carried out by testing three different ways of calculating the objective function. Both geometric parameter and topology were considered for these ways, while the properties of the material were kept constant.

2.1 Representation encoded

In order to develop a genetic algorithm, the first step consists on deciding the type of representation encoded for the variables.

Among the different types of representation, the binary encode has been chosen converting the variables of the problem into a string formed by “0” and “1”. This representation encode has been chosen mainly due to three reasons:

- These strings are very easy to create and manipulate, not only for the crossover operator but also for the mutation operator.
- It is the most common representation in genetic algorithms related to optimization of structures being obtained successful results.
- Historical reason, since this type of representation was the one used by Holland in the first genetic algorithms.

The string of binary code, just like the chromosome, has different segments, or genes that represent different features of a solution. Therefore, the number of genes is fixed by the number of variables of the problem [6]. In this work, the number of variables used in the optimization process of geometry of the jacket was sixteen; eight for the diameter and eight for the thickness (leg and brace per bay). On the other hand, for the topology optimization, the variables considered were the height of the bays since the total height and the footprint of the jacket were kept constant. Therefore, the number of variables were three, which correspond to the three k-joint K1, K2, K3 (Figure 1.3).

The length of the string in this work keeps constant over the iterations and its value depends on two factors [48]:

- Maximum and minimum values that represent the bounds of the search space.
- The desired precision for the representation of the different variables.

The number of bits required for a variable can be calculated as follows [41]:

$$l_i = \log_2 \left(\frac{x_i^{max} - x_i^{min}}{\varepsilon_i} \right)$$

Where:

x_i^{max} = Upper bound solution for variable i

x_i^{min} = Lower bound solution for variable i

ε_i = Desired precision in variable i

In the Table 2.1 the values obtained from the equation above are showed having chosen a precision of 1 mm.

Variable	Length string	Range of values [mm]
Diameter	11	$2^{11} = 2048$
Thickness	6	$2^6 = 64$
Z-position of K-joint	14	$\pm 2^{14-1} = \pm 8192$

Table 2-1. String length and range of values for the geometric variables

As a result, the representation of one individual consists on a binary string formed by 136 bits, in the geometry optimization (8 diameter variables · 11 bits + 8 thickness variables · 6 bits = 136 bits). For the topology optimization, the string length is 178 (136 bits + 3 Zposition · 14 bits = 178 bits).

2.2 Development

2.2.1 Initialization

The initial population is generated randomly within a range of values for the variables design (Table 2-2). The size of this population, N, will be fixed as a constant, like the string length, over the iterations.

		Diameter leg [mm]	Diameter brace [mm]	Thickness leg [mm]	Thickness brace [mm]	Weight [kg]
Geometry optimization	Min.	800	400	10	10	100196
	Max.	2847	2447	73	73	3590146
Geometry - topology opt.	Min.	300	100	10	5	23451
	Max.	2347	2147	73	68	2933400
Topology optimization	Min.	300	100	10	5	23451
	Max.	2347	2147	73	68	2933400

Table 2-2. Range of initial values for the geometric variables

2.2.2 Perform time domain simulation

For each individual of the population a loadcase analysis in the time domain is performed in order to assess ULS and FLS analysis for each joint in the structure. But before being performed, to avoid a waste of computational time, the individual is evaluated to check if it is within a range of validity established by the rules used for calculation of the stress concentration factors (SCF). If it is not, a new random individual is generated. This process is repeated until an N population size is reached.

After that, all structures are evaluated in FEDEM and damage is calculated under the specific environmental conditions imposed. The ULS and FLS calculation follow the IEC guideline [25].

2.2.3 Fitness evaluation

The fitness for each individual is evaluated considering the ULS and FLS requirements. The fitness can depend on several variables and can be defined in several ways. Particularly, in this study, the fitness calculation depends on 2 magnitudes:

- Weight: the function used to define the fitness of one individual is calculated as follows:

$$F_i = M_{max} - M_i$$

Where

F_i = Fitness of individual i

M_{max} = Maximum possible weight (Table 2.2)

M_i = Weight of individual i

- Damage: To estimate the damage of the structure a fatigue analysis according to the Recommended Practice DNV-RP-C203 [15] has been carried out (appendix A). The damage has been calculated for the different bays separately, taking values from 0 to 1 for allowed design (ULS and FLS requirements are fulfilled). This way, a value of 0 means that the damage for a bay is minimal, otherwise, a value of 1 is the maximum value of damage that one bay can withstand without exceeding the FLS. The function used in order to define the fitness for the damage is calculated as follows:

$$F_i = \sum_{i=1}^{N_b} D_i$$

Where

F_i = Fitness of individual i

N_b = Number of bays of the structure

D_i = Damage calculated for one bay of the structure

Since the damage is calculated for each bay, the total damage of the structure is the sum of the damage calculated for each bay separately. Therefore, the value of this function for an allowed design can take values from 0 to 4.

In addition, a linear scaling model has been used to improve the reproduction rate with better designs and depress the weaker ones. The scaling remaps the fitness values of each individual using the following equations:

$$F_i^{scaled} = a \cdot F_i + b$$

$$a = 2 \cdot \frac{F_{avg}}{F_{max} - F_{min}}$$

$$b = a \cdot F_{min}$$

Where

F_i^{scaled} = Scaled fitness of individual i

F_i = Fitness of individual i

F_{avg} = Fitness average of the population

F_{max} = Fitness maximum of the population

F_{min} = Fitness minimum of the population

2.2.4 Genetic operators

2.2.4.1 Selection

Once the individuals have been assigned a fitness value, f_i , they can be chosen from the population and recombined in order to produce the new design for the next generation. The probability of one individual to be chosen will be established according to their fitness scale. This probability is calculated as follows:

$$P_i = \frac{F_i^{Scaled}}{\sum_{i=1}^N F_i^{Scaled}}$$

Where:

P_i = Probability to be chosen of individual i

N = Size of the population

F_i^{Scaled} = Fitness scaled of individual i

Next step is the selection of pairs of parents. Two individuals (parents) are picked up randomly from the existing population for combining (“mating”) to produce two new individuals (children). But in this process, one condition is imposed: the selection of two parents who correspond to the same design is not allowed. This way, inbreeding is avoided and the diversity in the designs of children population is guaranteed.

2.2.4.2 Crossover

Having selected the pairs of parents, the crossover operator is the responsible for the genetic combination to generate the two new designs. In this study, the method used for this operator is the multipoint crossover.

Crossover sites determine the position in the string where the string is cut (Figure 2-1). Crossover sites are chosen randomly within the string length of the chromosome.

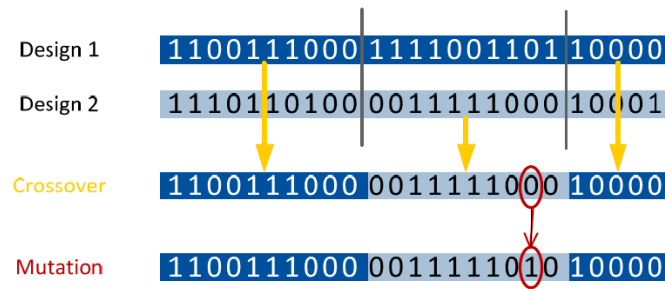


Figure 2-1. Example of crossover and mutation operators. For simplicity only two cuts (grey vertical lines) are shown and one mutation (red circles), and only 25 bits of the chromosome are shown

Parameters defined for crossover operator:

1. *Cut_int*: range for the amount of crossover sites.
2. *Cut*: two possible options: (1) the cuts for the crossover are picked in equally divided parts of the chromosome, or (2) the cuts are defined randomly within the whole chromosome.

2.2.4.3 Mutation

Mutation operator is introduced in the optimization process to encourage the development of new genetic material and to avoid on the other hand, avoids the possible loss of valuable genetic material during the optimization process [1].

This operator consists on the replacement of a single bit inside the string by its component (Figure 2-1). This process is controlled by the probability of mutation. In this study, a baseline mutation probability of 0.05 has been defined. But the probability of mutation varies over the generations and depends on the change of the fitness of the best individual. This way, if the fitness of the best individual has not changed for a specific number of generations, the value of the probability of mutation is increased 0.01. But there is a limit to this increase, which was established at 0.12.

Parameters defined for mutation operators:

- P_m : mutation probability.
- P_{m_max} : maximal mutation probability.
- Mut_gen : range of generations where mutation is applied.
- Mut_int : increase for mutation probability.
- Mut_last : number of generations without any change on the fitness of the best individual needed to increase the mutation probability.

2.2.4.4 Immigration

This operator was introduced in the genetic algorithm to further improve the convergence performance [52]. The objective of this operator is to increase the diversity in the population, and consequently, the chance to find a local optimum instead of a global is reduced.

During the immigration, completely new, randomly generated individuals are introduced to the gene pool and are directly used during the reproduction process.

Parameters defined for the immigration operator:

1. $Immi$: range of generations where immigration is applied.
2. $Immi_int$: interval of generations where immigration is applied.
3. $Immi_P$: number of immigrations per immigration process.

2.2.5 Flow chart of the genetic algorithm

In order to improve the development of the genetic algorithm, the chromosomes of the children were checked after crossover and mutation to confirm that (1) the design fulfills the constraints for SCF calculation and (2) the design was not used in an earlier generation. This is ensured by storing each used chromosome in a table. The reproduction process is repeated until the size of children population is equal to the prior population size N .

After evaluation of the fitness of a new population, the N fittest designs of the combined set of current (children) and prior (parent) generation are selected to form the parents (gene pool) for the next iteration of the reproduction process. This method ensures that the fitness will never decrease from one generation to the next and the individuals with high fitness will never become extinct [1].

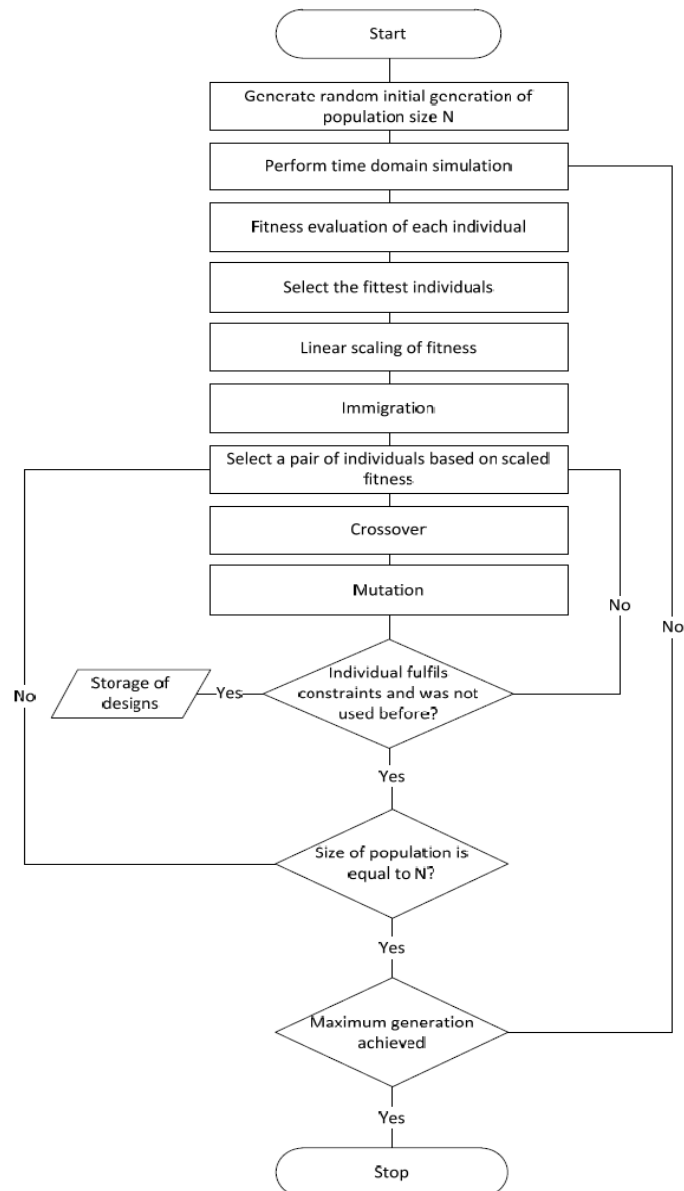


Figure 2-2. Flow chart of proposed genetic algorithm

2.3 Convergence of proposed algorithm

A weakness of the genetic algorithm was identified by plotting the dimensions of all tested children in an optimization run for legs and braces in a scatter diagram (Figure 2-3)

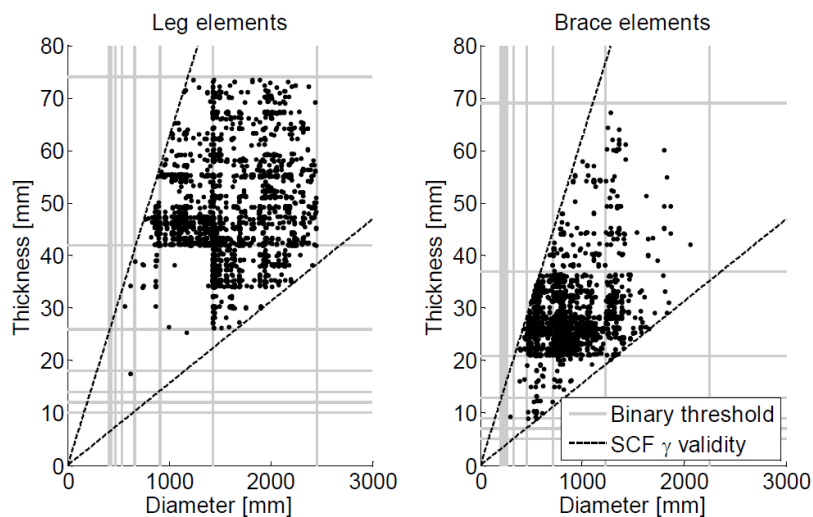


Figure 2-3. The challenges of binary thresholds

As can be seen in the diagram for the leg (clearer), there are some areas of the search space that are difficult to test. It looks as if some limitations are imposed for the search in the design space. Particularly, the value 1424 for the diameter and the value 42 for the thickness act as constraints. The reason for this behavior relies on the code used for representation of the design variables.

As the binary string of each individual may contain a 1 or 0 at the same position for the whole population, the crossover operator will not be able to change this value and further optimize the structure over such a so-called binary threshold.

The example shown is based on a minimum diameter of 400 mm and a minimum thickness of 10 mm for leg elements. In order to better understand the reason for these non-tested-areas, the values that act as constraints are analyzed.

$$1424 = 400 \text{ (minimum diameter value)} + 1024$$

$$42 = 10 \text{ (minimum thickness value)} + 32$$

The value 1024 converted into binary code looks as: 1000000000. However, the value just below (1023) looks as: 0111111111. With this example is easy to see why for a population where all the individuals have values above 1024, it is impossible to get children below this value since the value 0 for the first position of the string is not present in any of individuals. There is not chance to get a value 0 for the first position after crossover

operation. The same situation happens with the value 32 for the thickness, whose value in binary code is: 100000.

For these reasons, the algorithm is converging towards these limits. Only a few individual designs are tested with values below these limits. This premature convergence to local optimums has been solved successfully by increasing progressively the mutation rate when the fitness of the best individual has not changed for a specific number of iterations (possible local optimum has been found), and by the introduction of immigration operator.

3 Results

The results obtained from running the algorithm several times combining different objective functions with the two types of optimization that we have proposed for this thesis, topology optimization (nodes location) and geometric optimization (diameter and thickness) are shown in this section.

Three ways to calculate the fitness are defined. The first is based on the optimization of the weight, one has to make sure that the structure fulfills the requirements of the damage but the design with the lowest weight will be the best fit. The second objective function is based on the damage: designs which are heavier than the heaviest structure previously set, will not be analyzed. So, after rejecting the individuals that do not fulfill the requirements, the highest fitness value will be given to the one with the highest damage (considering that it won't brake). Third, the last objective function that will be analyzed is based on a combination of a weight optimization and a damage optimization. For the odd generations the fitness is assigned using the first objective function (weight optimization) and for the even generations the second objective function (damage optimization) is used.

In the following subsections the results from running these objective functions are presented. For every function a geometric optimization and a full optimization (changing the location of the nodes, the diameters and the thickness of the beams) are simulated. One of the objectives was to run also a simulations where only the locations of the nodes are changed but that it was not possible since the model from the OC4 project doesn't survive the loadcase used for that study. However, if a different model is tested it would probably work and it should be simulated in further studies.

The same type of figures have been created for each simulation along this section:

- Weight range: where the difference between the heaviest and lightest design of each generation can be observed. The aim of these figures is to appreciate how the structures become more alike.
- Evolution of geometric variables through the generations: where the convergence of the diameter and thickness to the optimum can be seen.
- Fitness representation: where the highest fitness of each generation is plotted showing how the designs improve with the time.

- Number of designs that change on every generation at every bay. In these cases four different figures are plotted, one per bay. The aim of those figures is to discover whether there is a bay that converges to the optimum faster than the others. In addition, lines are drawn every 20 generations, when immigration takes place. One immigration takes place every 10 generations until generation 200 but they are plotted every 20 iterations not to make the figures unreadable.
- Constrains importance: where some SCF parameter are represented together with the different values that the geometric variables (diameter or thickness, depending on the SCF parameter) take along the whole simulation. The aim of these figures is to determine which constrain is the most restrictive, all geometric variables will be inside the limits the SCF parameters draw (otherwise the FLS analysis will not be perform) but some of them will be closer to these limits than others. The SCF parameter that will be represented are:

$$0.2 \leq \beta = \frac{d}{D} \leq 1$$

$$0.2 \leq \tau = \frac{t}{T} \leq 1$$

$$8 \leq \gamma = \frac{D}{2 \cdot T} \leq 32$$

- Damage progress through the generations: where the damage for the best design of each generation is represented so that we can appreciate how it changes with the geometric variables change, when a new individual is introduced into the group of the best designs.
- Location of the nodes: where will be possible to look into the final location of the nodes of the 10 best designs (only when topology optimization is carried out).

3.1 Objective function based on the weight

The ideal weight that should be reached with this function is the minimum that has been established, but that is not possible due to the SCF constraints that are required by the FLS analysis. The best design that we will end up with is the lightest design that can survive the load actions during 20 years.

3.1.1 Geometry optimization

3.1.1.1 Weight range

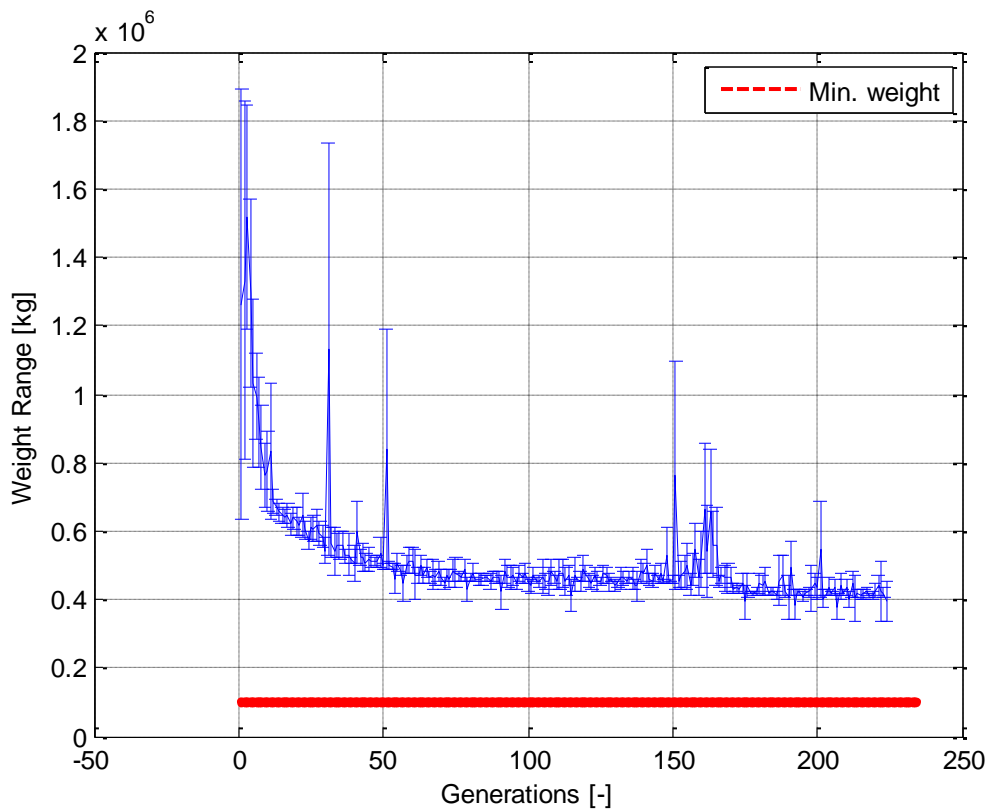


Figure 3-1. Weight of the structure [kg], when optimizing the weight (224 generations and 9 individuals)

The lightest design of the last generation weight 338.178 kg and the heaviest 453.455 kg, but the best one is in between and weight 420.122 kg.

The limit for the maximum weight is not represented because the designs that are heavier than the maximum are not analyzed and therefore this limit is never achieved.

3.1.1.2 Diameter progress

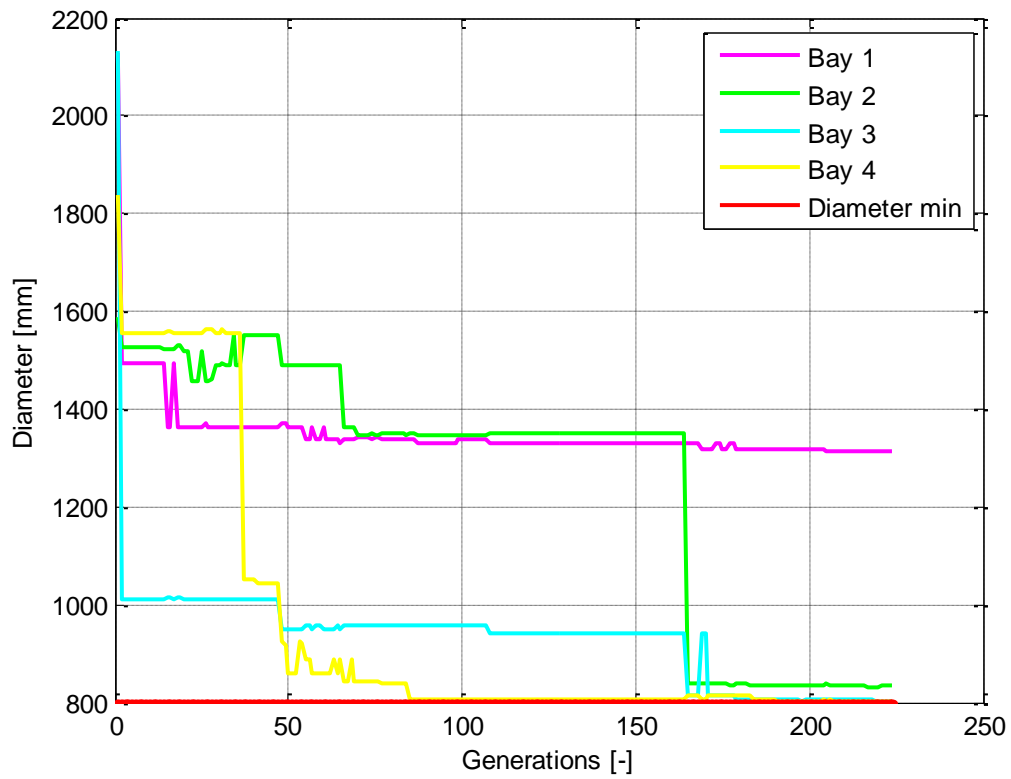


Figure 3-2. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight (224 generations and 9 individuals)

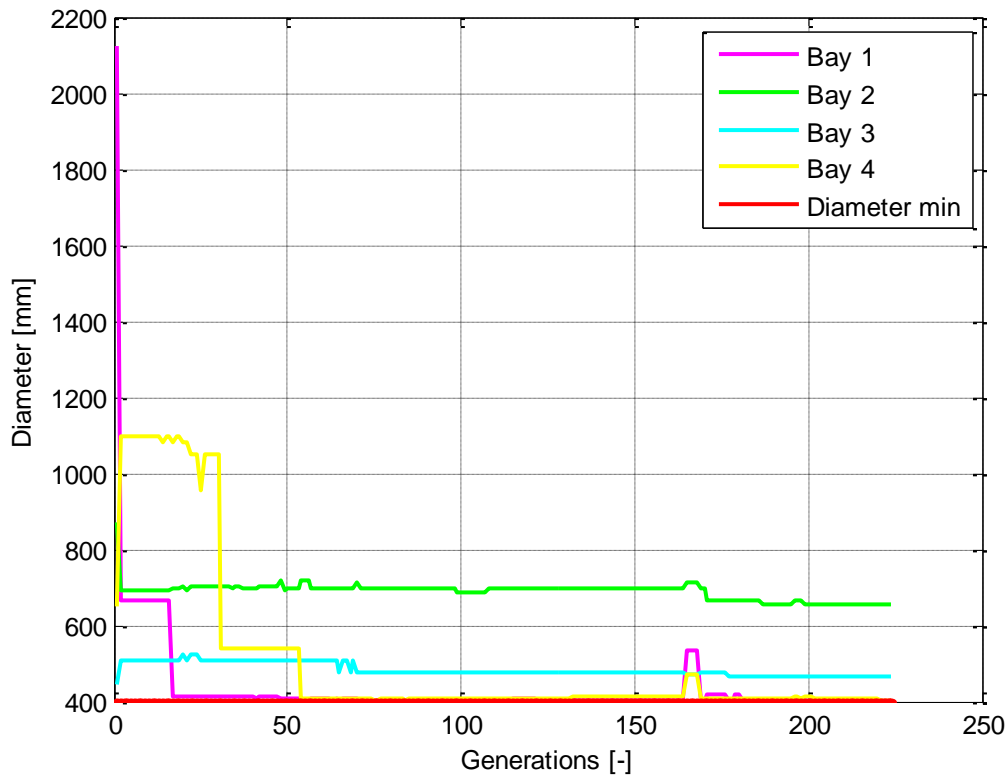


Figure 3-3. Evolution of the values of the diameter [mm] for the *braces* of every bay and minimum value that can be reach when optimizing the weight (224 generations and 9 individuals)

The exact values of diameter that the legs and braces take at the end of the simulation are presented in the table below:

		DIAMETER [mm]			
		Bay 1	Bay 2	Bay 3	Bay 4
Leg		1.312	836	804	804
Brace		404	656	406	401

Table 3-1. Values of diameter at the last generation when optimizing the weight and the geometry (224 generations and 9 designs)

3.1.1.3 Thickness progress

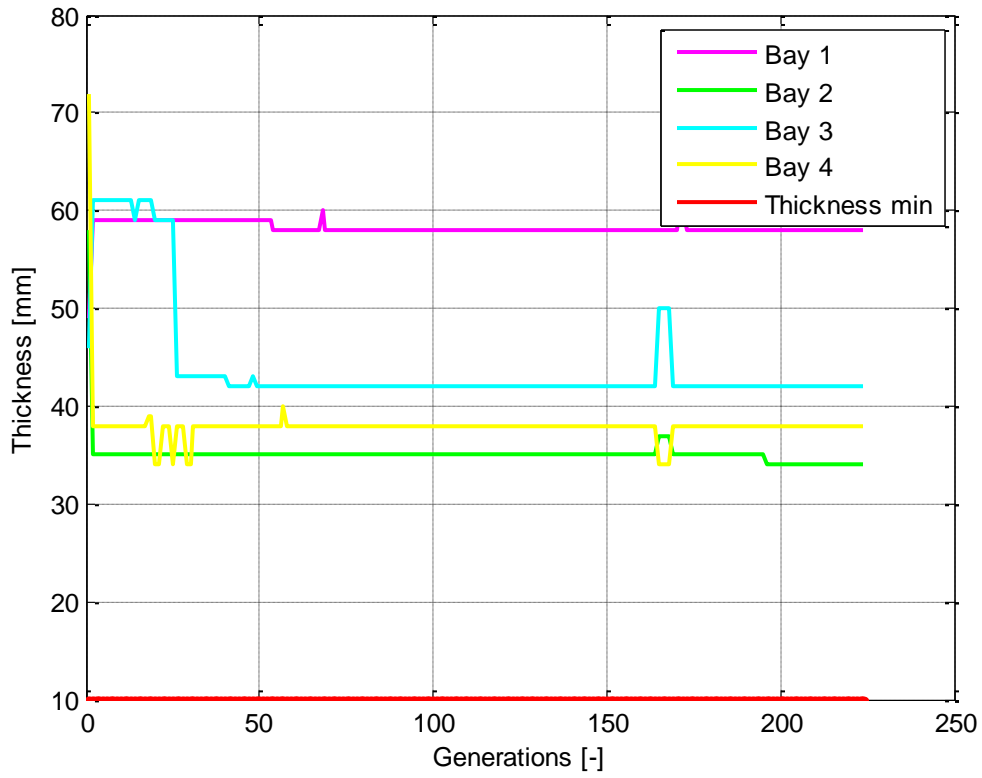


Figure 3-4. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight (224 generations and 9 individuals)

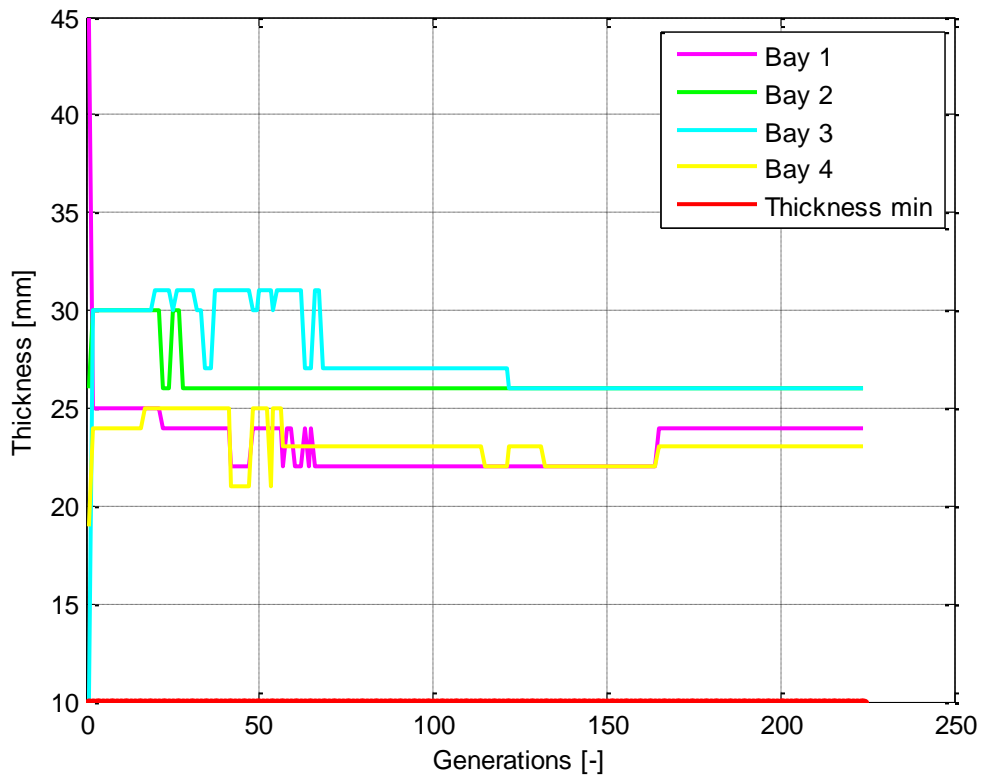


Figure 3-5. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight (224 generations and 9 individuals)

The exact values of thickness that the legs and braces take at the end of the simulation are presented in the table below:

	THICKNESS [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	58	34	42	38
Brace	24	26	26	23

Table 3-2. Values of thickness at the last generation when optimizing the weight and the geometry (224 generations and 9 designs)

3.1.1.4 Fitness

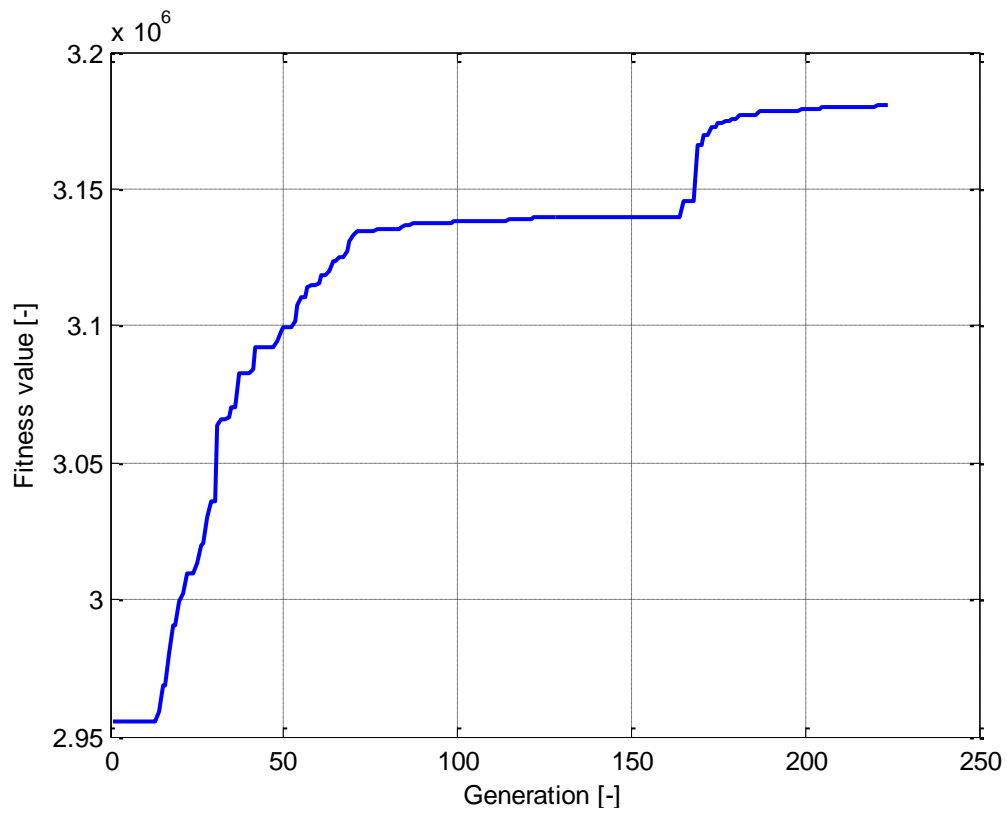


Figure 3-6. Improvement of the fitness through the generations when optimizing the weight (224 generations and 9 individuals)

3.1.1.5 Number of designs that change

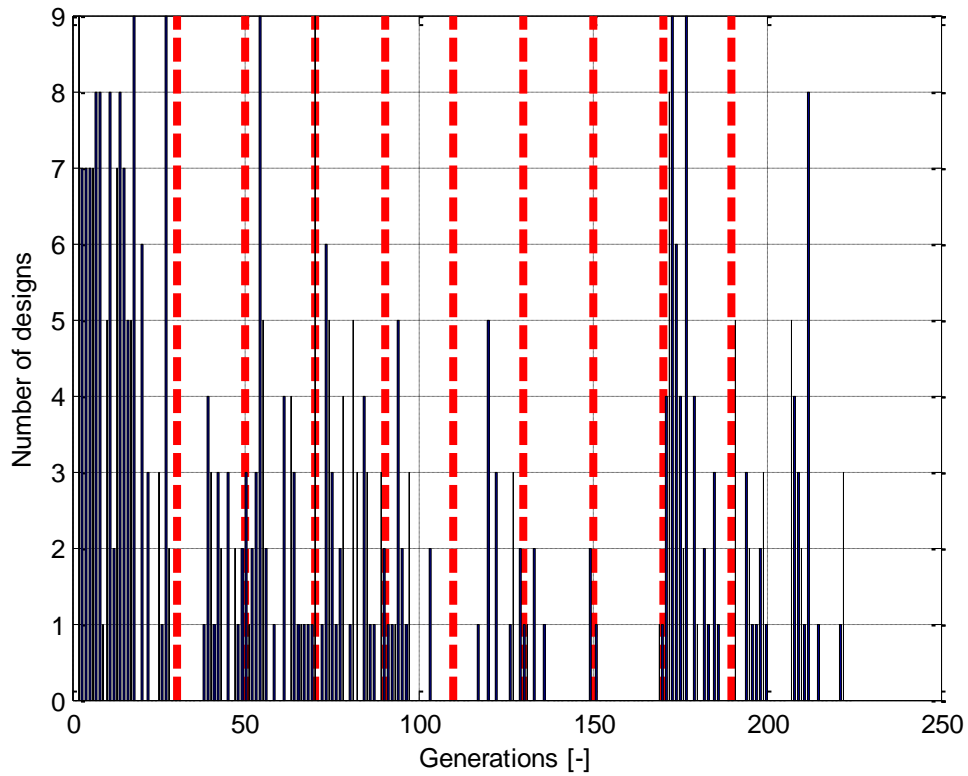


Figure 3-7. Number of designs of bay 1 that change from one generation to the next one when optimizing the weight (224 generations and 9 individuals)

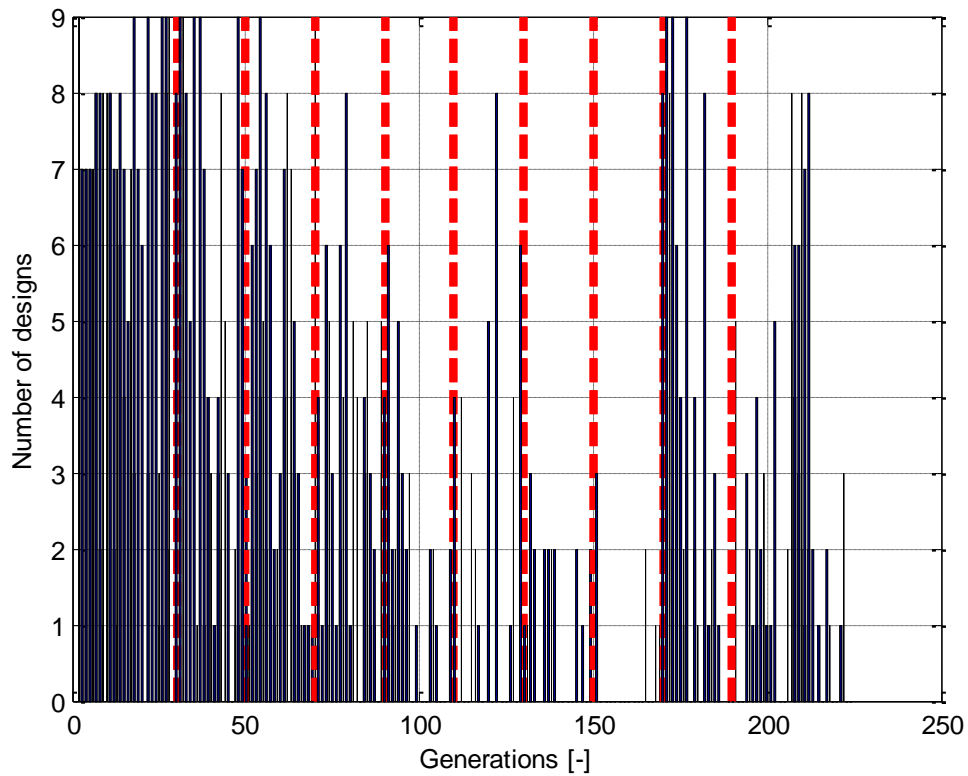


Figure 3-8. Number of designs of bay 2 that change from one generation to the next one when optimizing the weight (224 generations and 9 individuals)

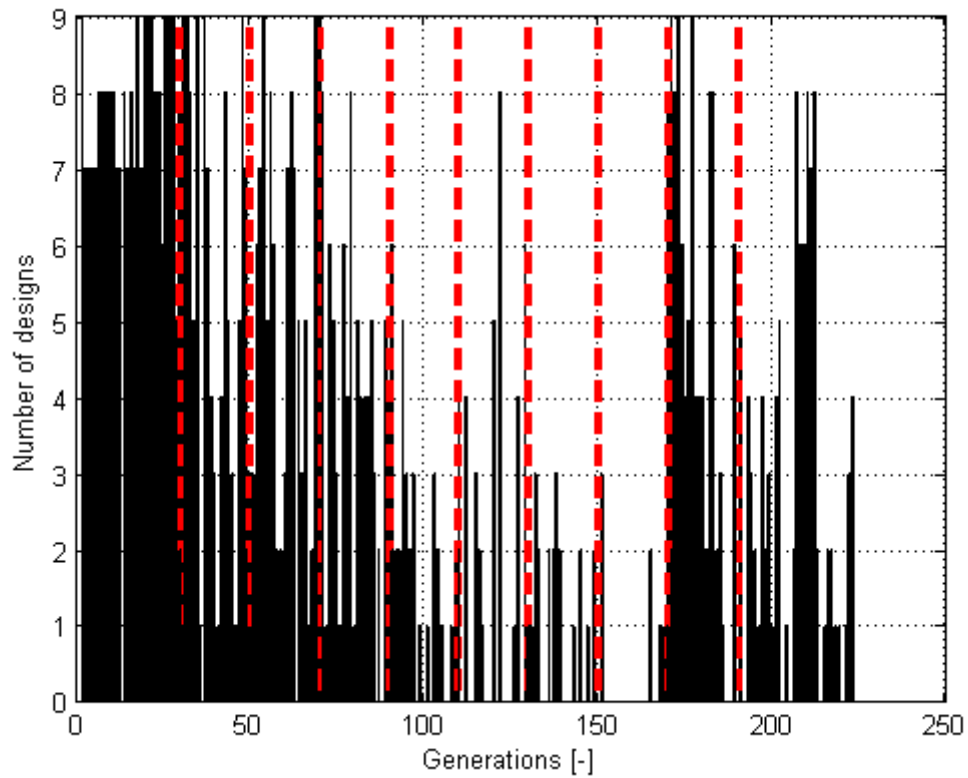


Figure 3-9. Number of designs of bay 3 that change from one generation to the next one when optimizing the weight (224 generations and 9 individuals)

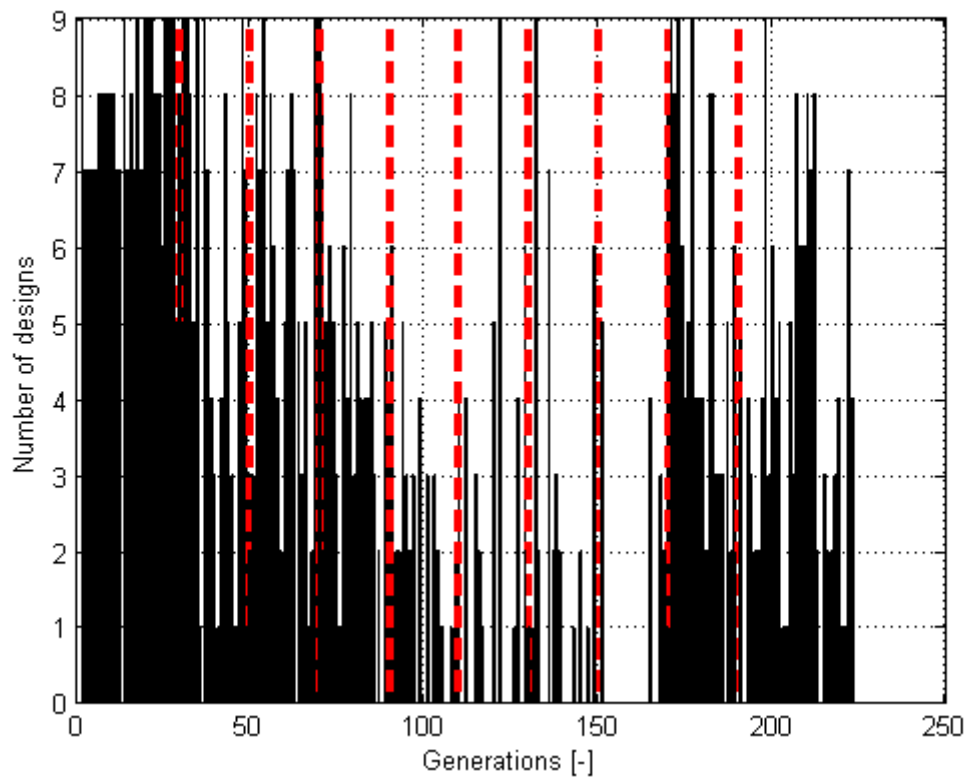


Figure 3-10. Number of designs of bay 4 that change from one generation to the next one when optimizing the weight (224 generations and 9 individuals)

3.1.1.6 SCF parameters constraints

3.1.1.6.1 Gamma parameter

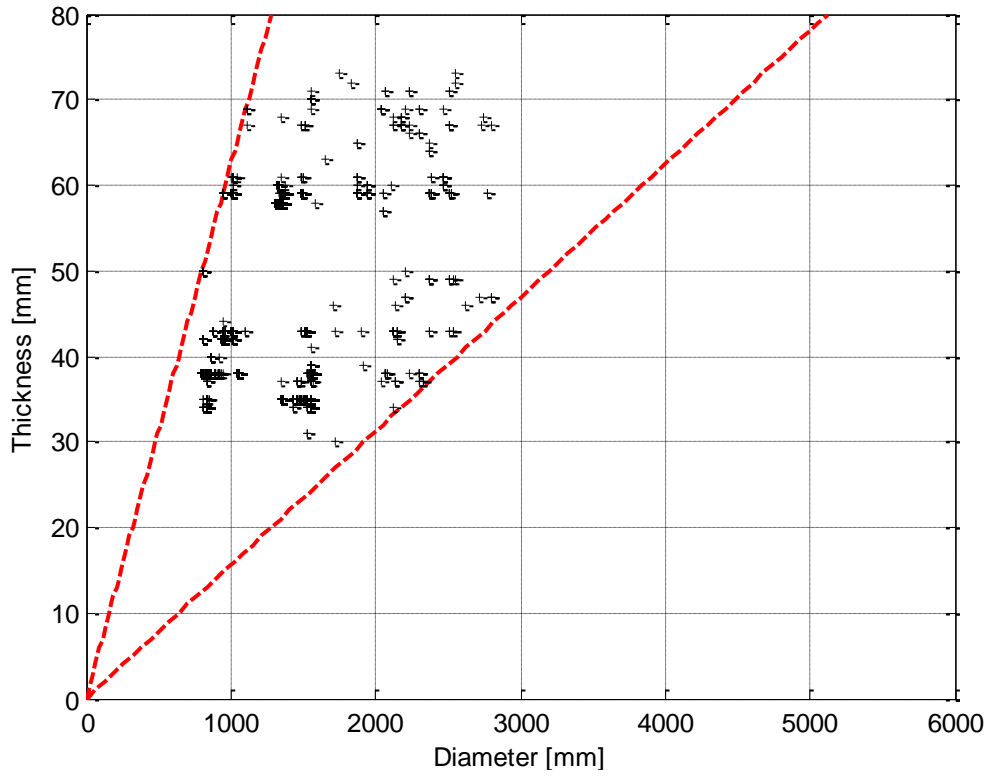


Figure 3-11. Gamma constriction for the legs when optimizing the weight (224 generations and 9 individuals)

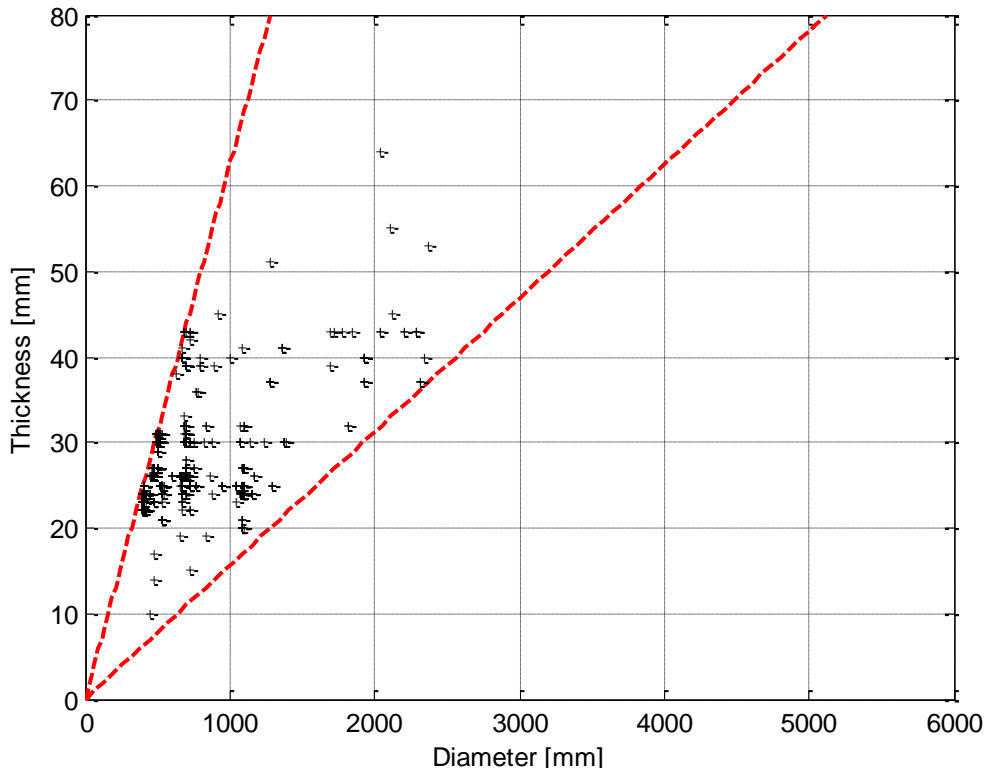


Figure 3-12. Gamma constriction for the braces when optimizing the weight (224 generations and 9 individuals)

3.1.1.6.2 Beta parameter

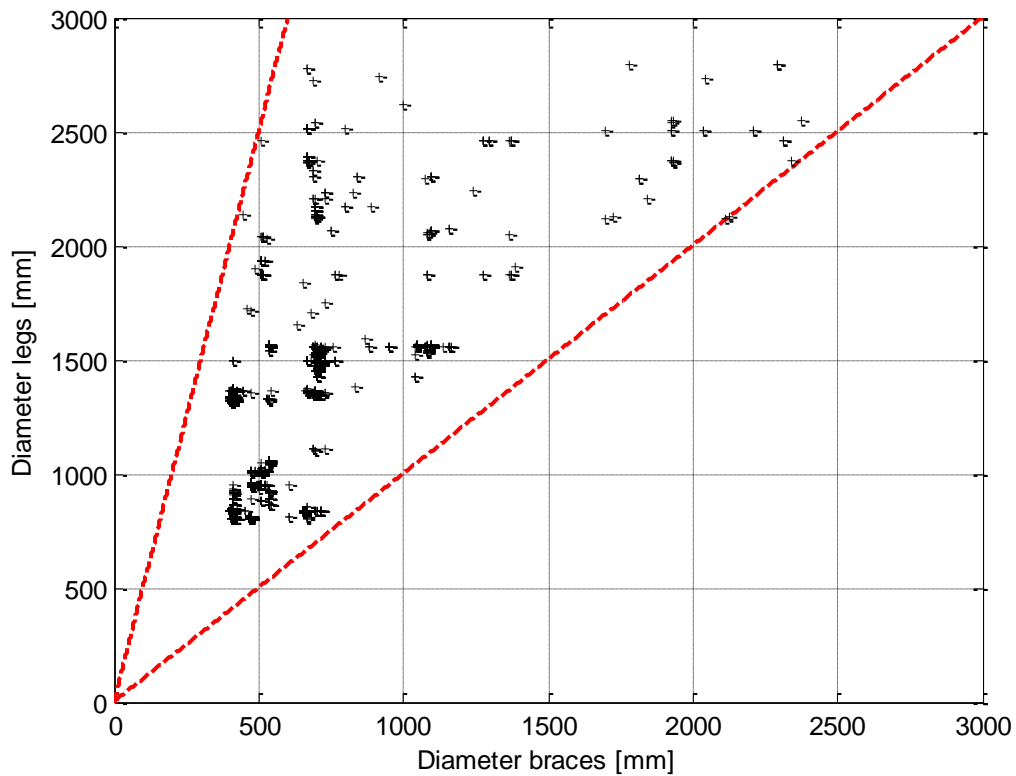


Figure 3-13. Beta constrain for legs and braces when optimizing the weight (224 generations and 9 individuals)

3.1.1.6.3 Tau parameter

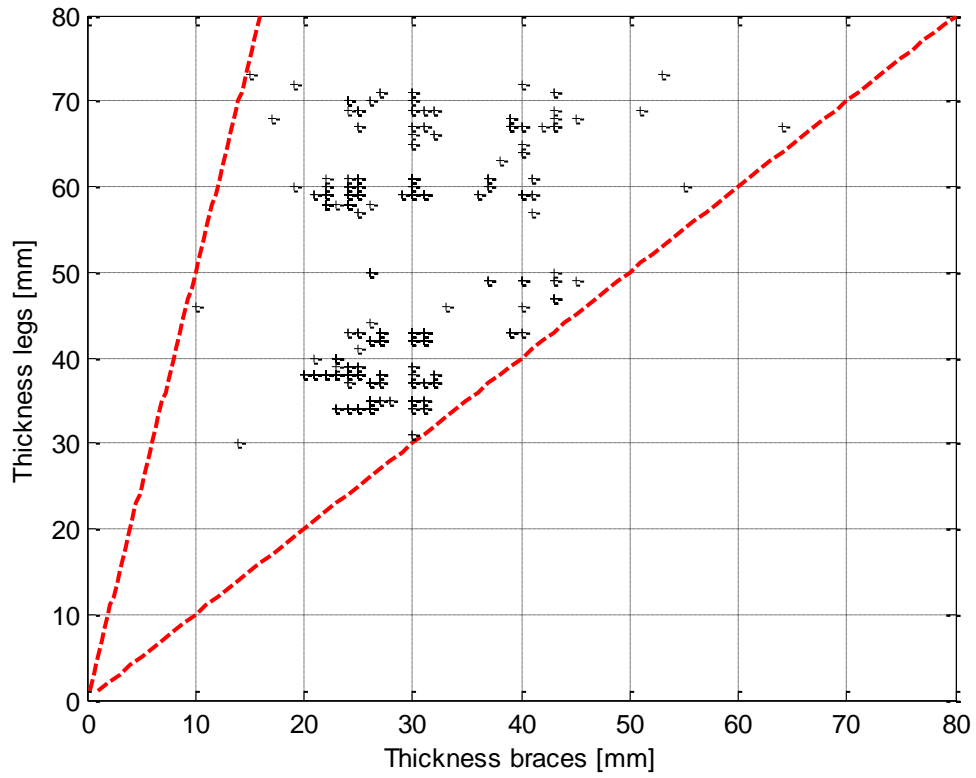


Figure 3-14. Tau constrain for legs and braces when optimizing the weight (224 generations and 9 individuals)

3.1.2 Topology and geometry optimization

3.1.2.1 Weight range

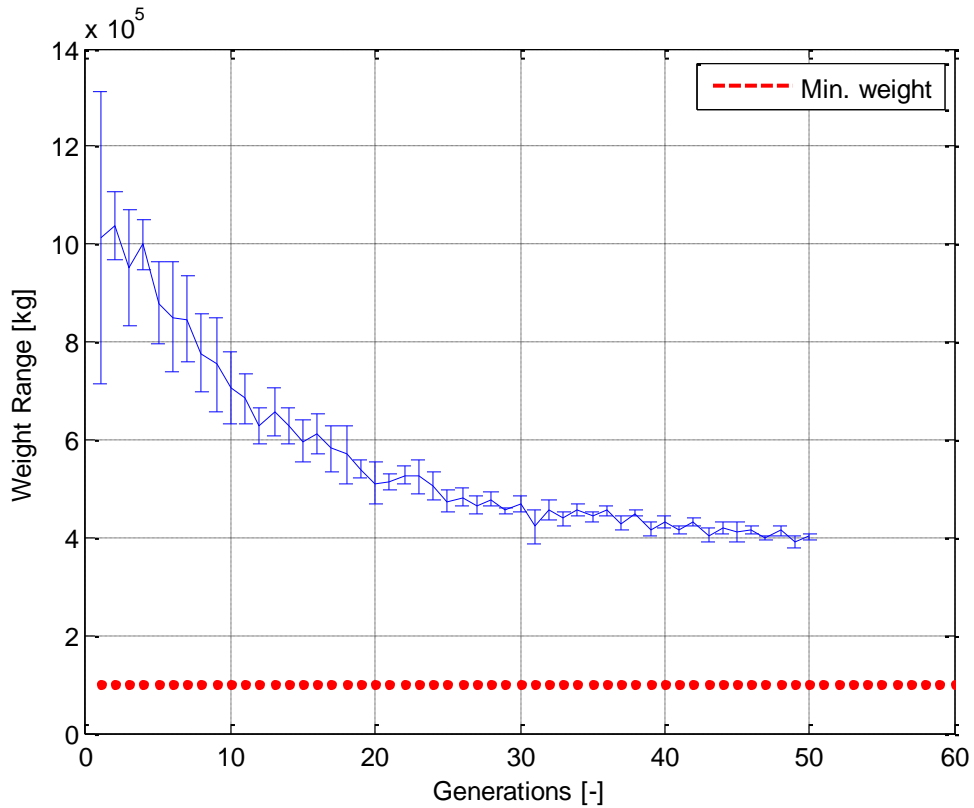


Figure 3-15. Weight of the structure [kg], when optimizing the weight (50 generations and 10 individuals)

The lightest design of the last generation weight 396.608 kg and the heaviest 407.988 kg , but the best one is in between and weight 396.608 kg.

If we compare the best design of this objective function with the one that only optimize the weight turns out that the optimum design when optimizing both, topology and geometry is lighter.

3.1.2.2 Diameter progress

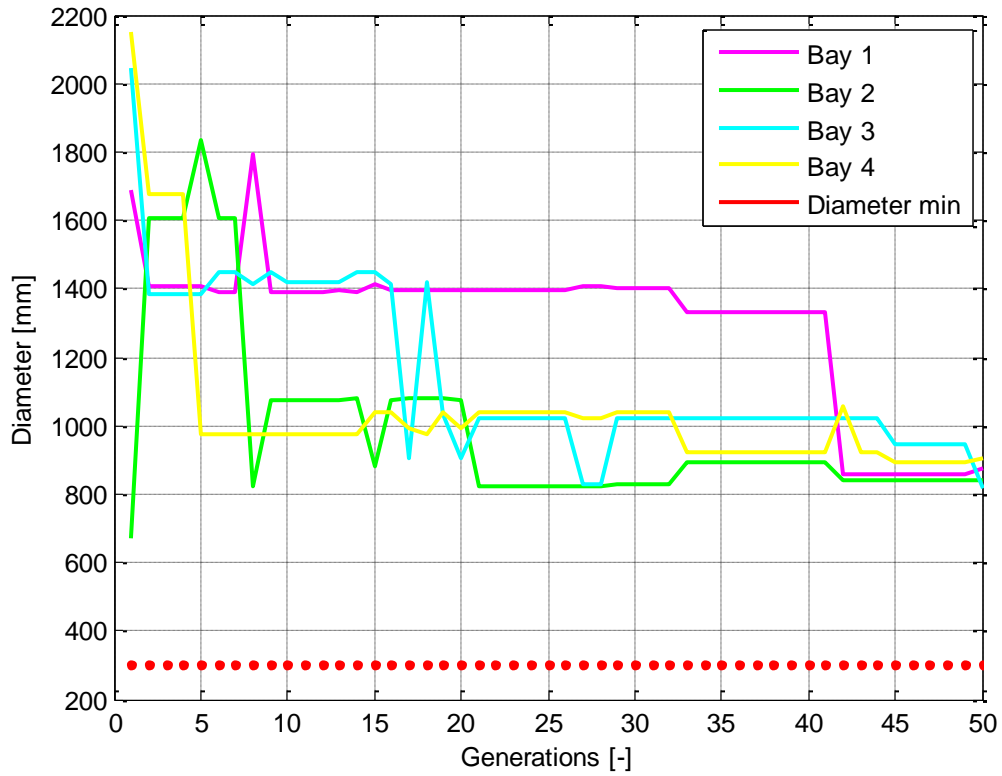


Figure 3-16. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight (50 generations and 10 individuals)

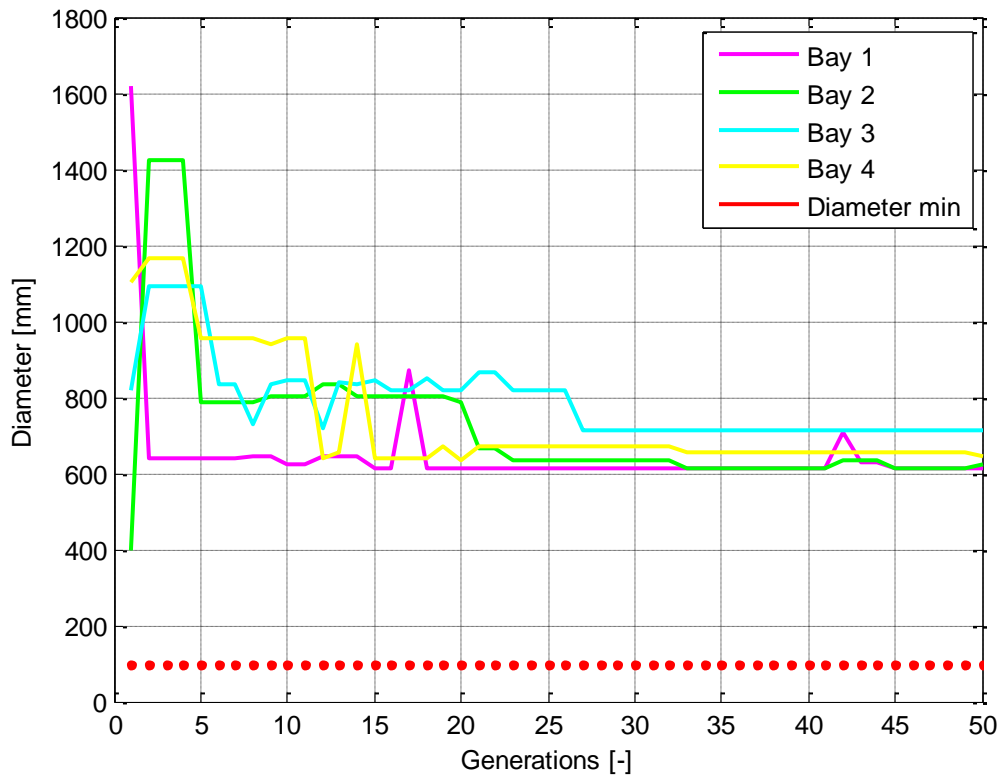


Figure 3-17. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight (50 generations and 10 individuals)

The exact values of diameter that the legs and braces take at the end of the simulation are presented in the table below:

		DIAMETER [mm]			
		Bay 1	Bay 2	Bay 3	Bay 4
Leg		875	840	819	907
Brace		613	623	713	647

Table 3-3. Values of diameter at the last generation when optimizing the weight the topology and the geometry (50 generations and 10 designs)

3.1.2.3 Thickness progress

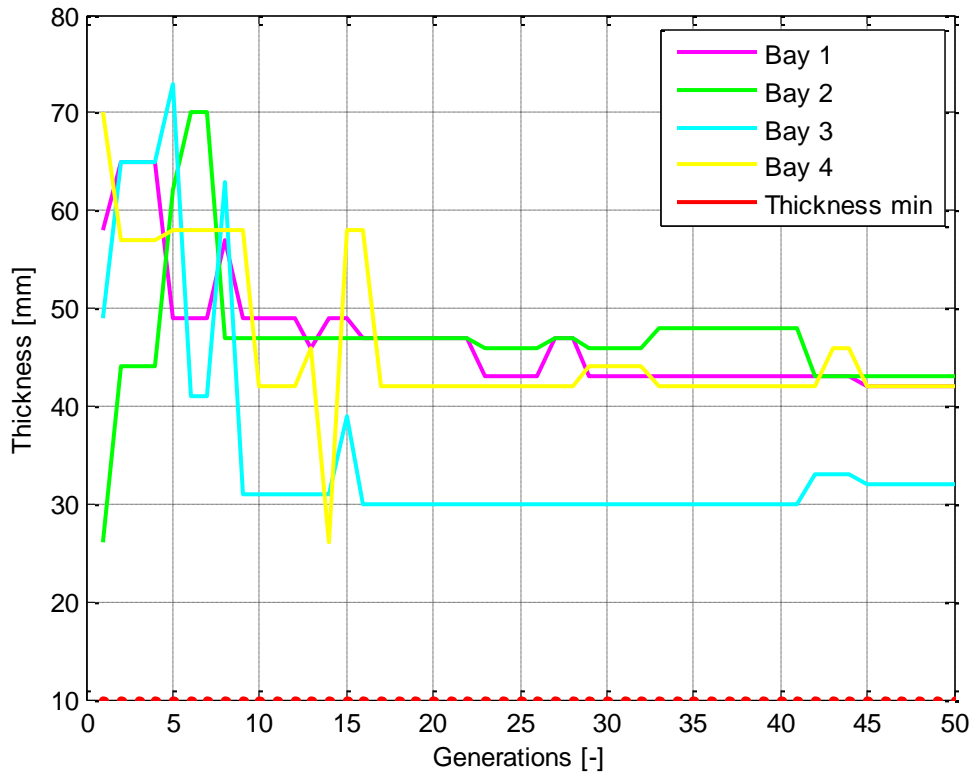


Figure 3-18. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight (50 generations and 10 individuals)

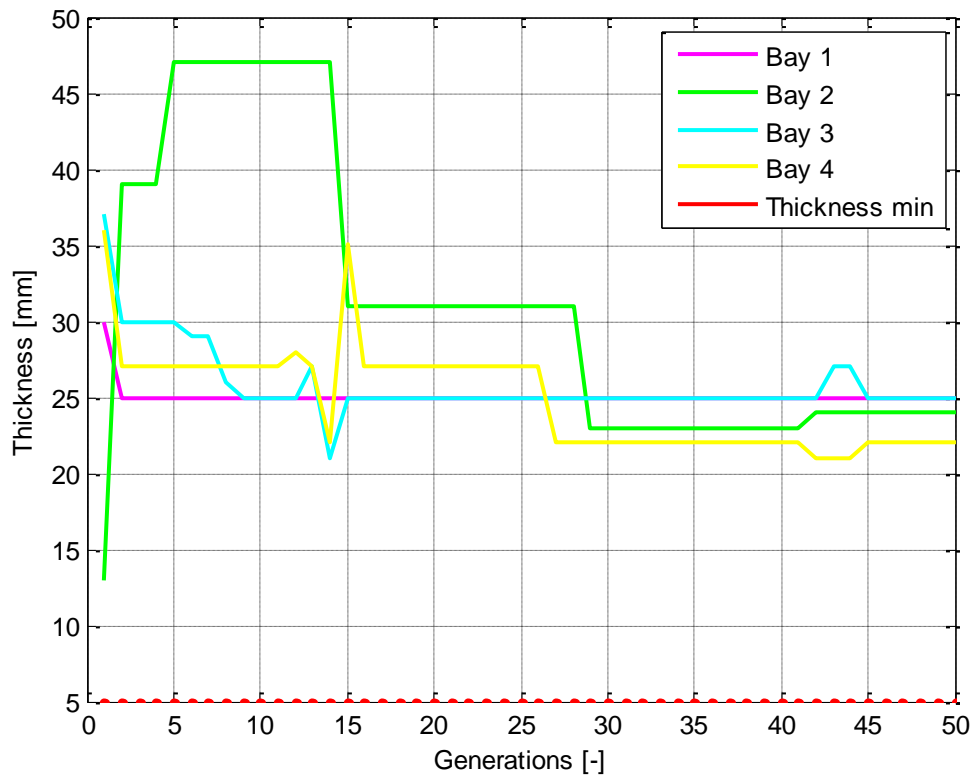


Figure 3-19. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight (50 generations and 10 individuals)

The exact values of thickness that the legs and braces take at the end of the simulation are presented in the table below:

		THICKNESS [mm]			
		Bay 1	Bay 2	Bay 3	Bay 4
Leg		42	43	32	42
Brace		25	24	25	22

Table 3-4. Values of thickness at the last generation when optimizing the weight the topology and the geometry (50 generations and 10 designs)

3.1.2.4 Topology overview

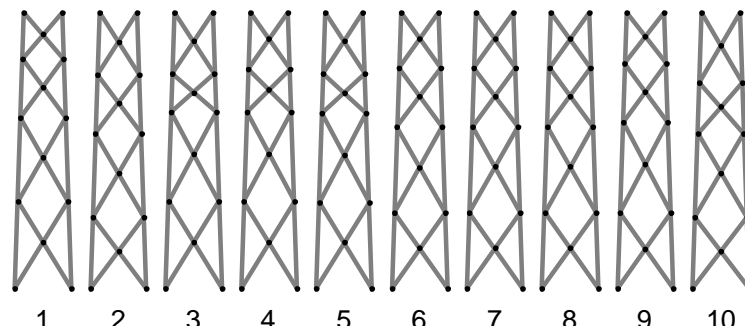


Figure 3-20. Location of the nodes of the 10 best designs when optimizing the weight (50 generations and 10 individuals)

And the length of the bays of the best design is written in the table that follows:

LENGTH [mm]				
	Bay 1	Bay 2	Bay 3	Bay 4
Length	18.353	17.964	12.484	9.9977

Table 3-5. Values of the length of the bays of the best design at the last generation when optimizing the weight the topology and the geometry (50 generations and 10 designs)

3.1.2.5 Fitness

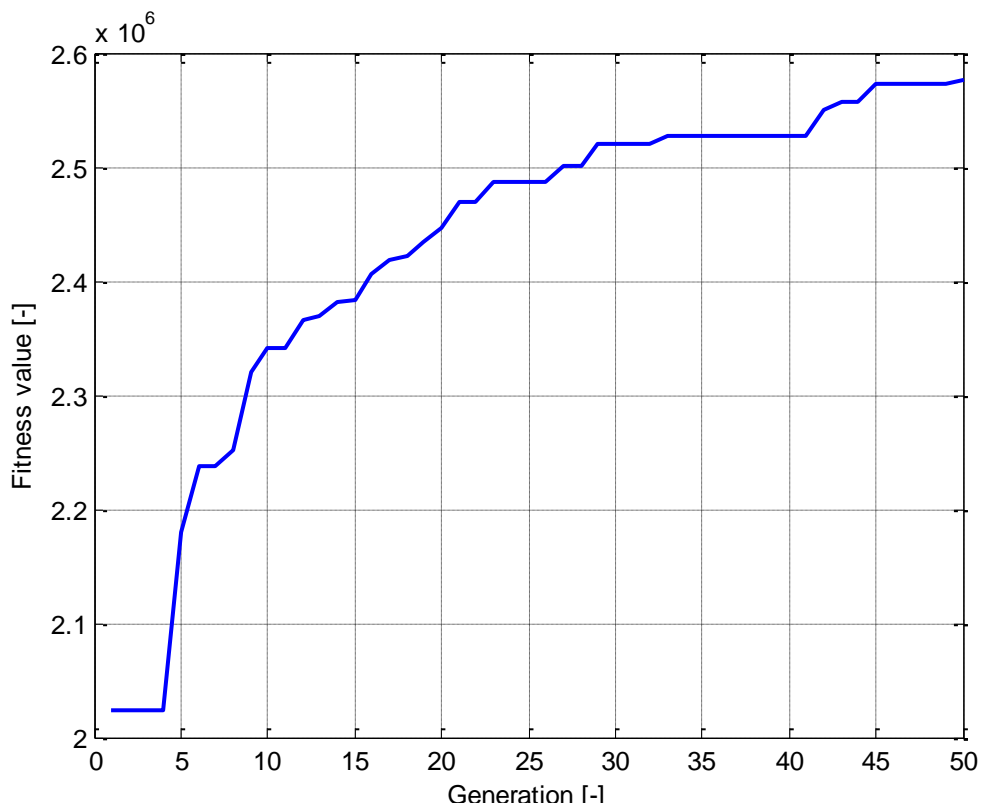


Figure 3-21. Improvement of the fitness through the generations when optimizing the weight (50 generations and 10 individuals)

3.1.2.6 Number of designs that change

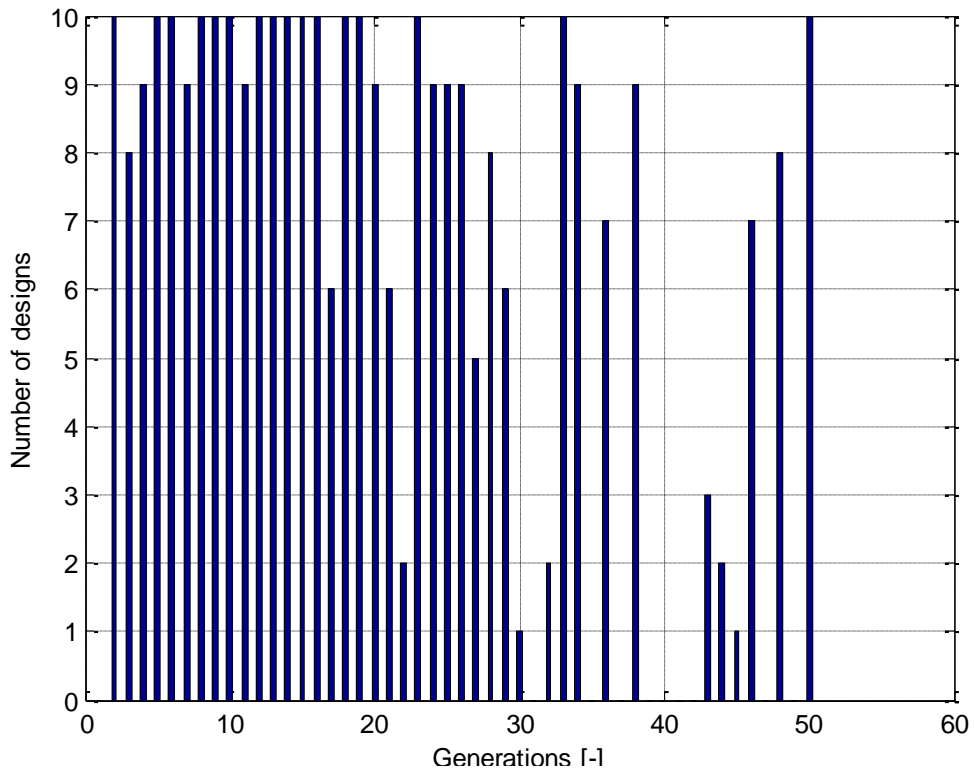


Figure 3-22. Number of designs of bay 1 that change from one generation to the next one when optimizing the weight (50 generations and 10 individuals)

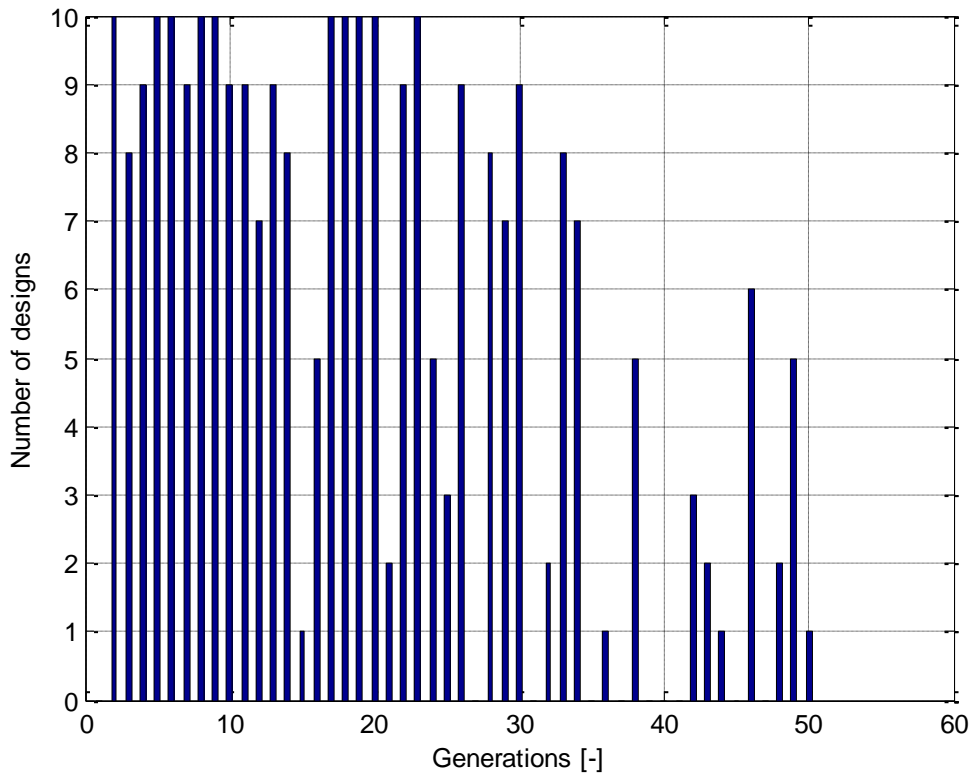


Figure 3-23. Number of designs of bay 2 that change from one generation to the next one when optimizing the weight (50 generations and 10 individuals)

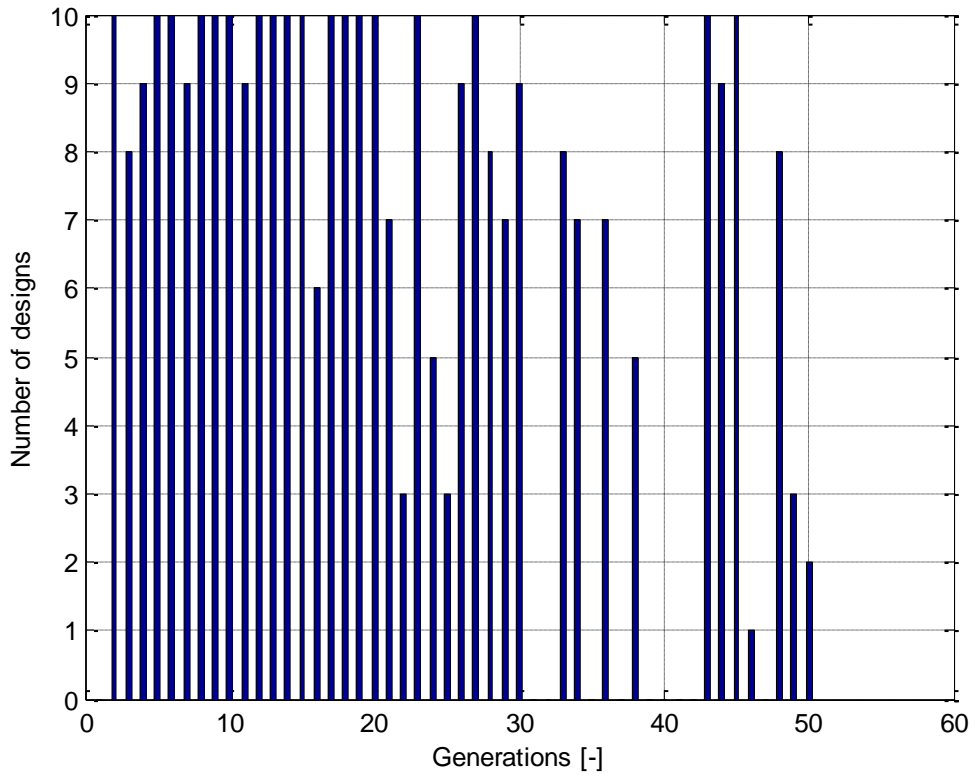


Figure 3-24. Number of designs of bay 3 that change from one generation to the next one when optimizing the weight (50 generations and 10 individuals)

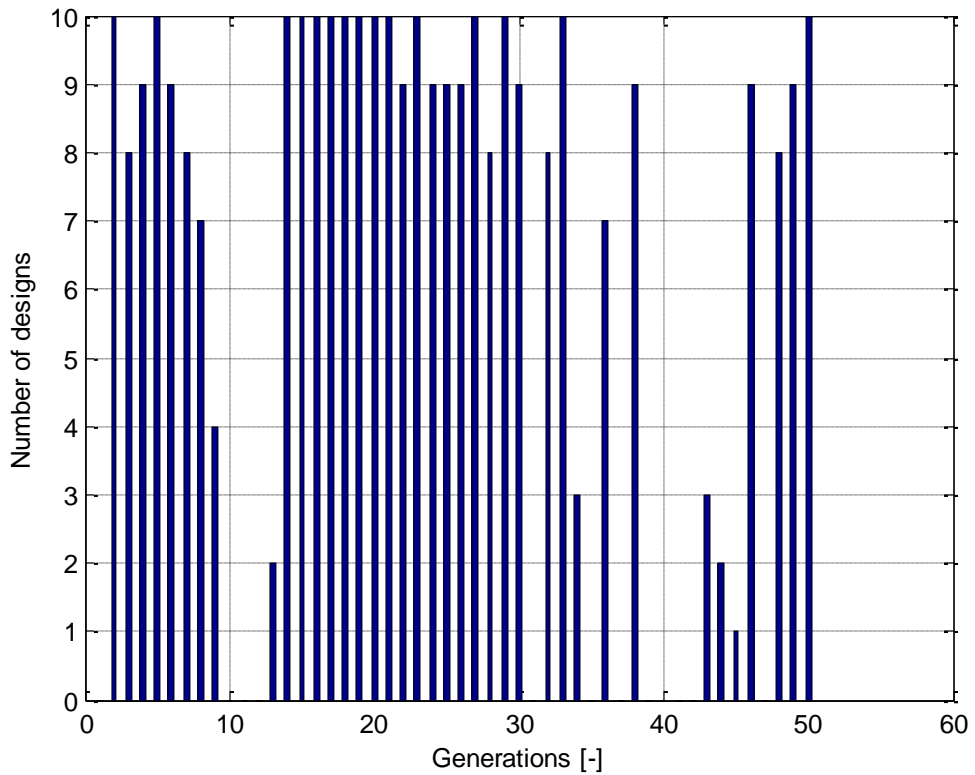


Figure 3-25. Number of designs of bay 4 that change from one generation to the next one when optimizing the weight (50 generations and 10 individuals)

3.1.2.7 SCF parameters constrains

3.1.2.7.1 Gamma parameter

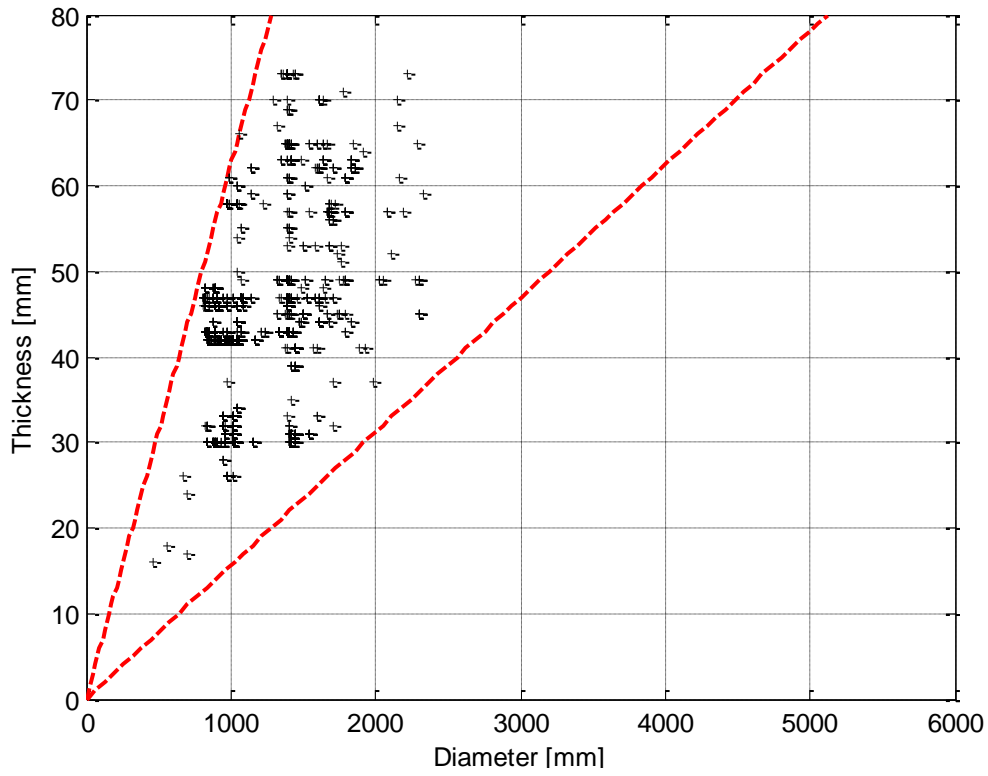


Figure 3-26. Gamma constriction for the legs when optimizing the weight (50 generations and 10 individuals)

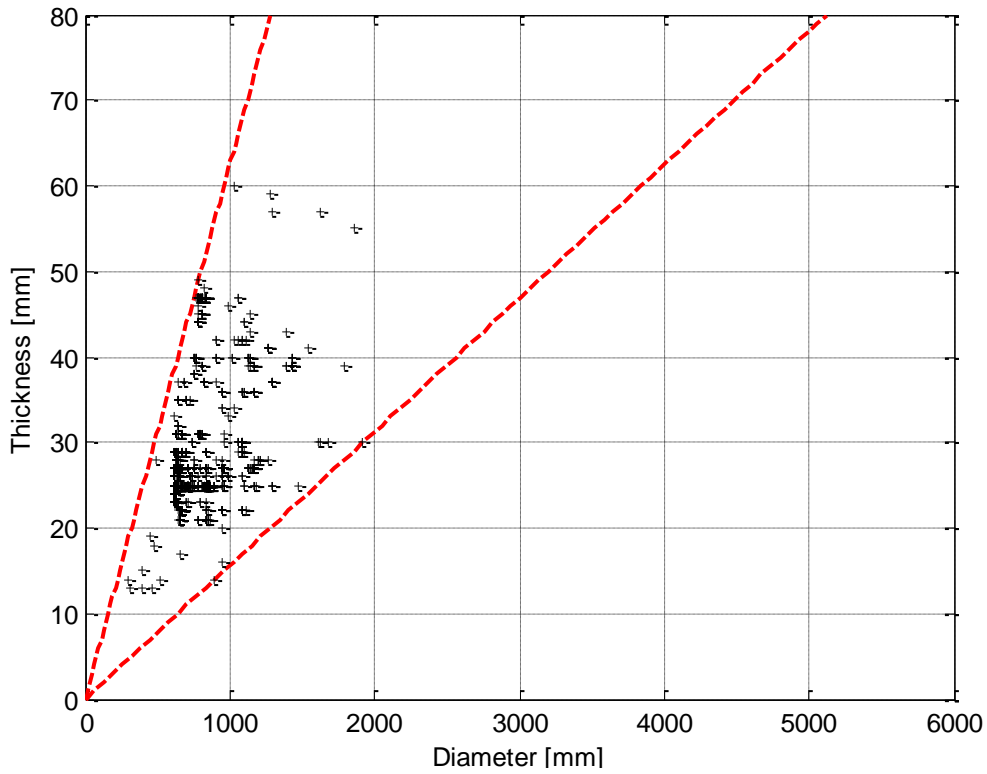


Figure 3-27. Gamma constriction for the braces when optimizing the weight (50 generations and 10 individuals)

3.1.2.7.2 Beta parameter

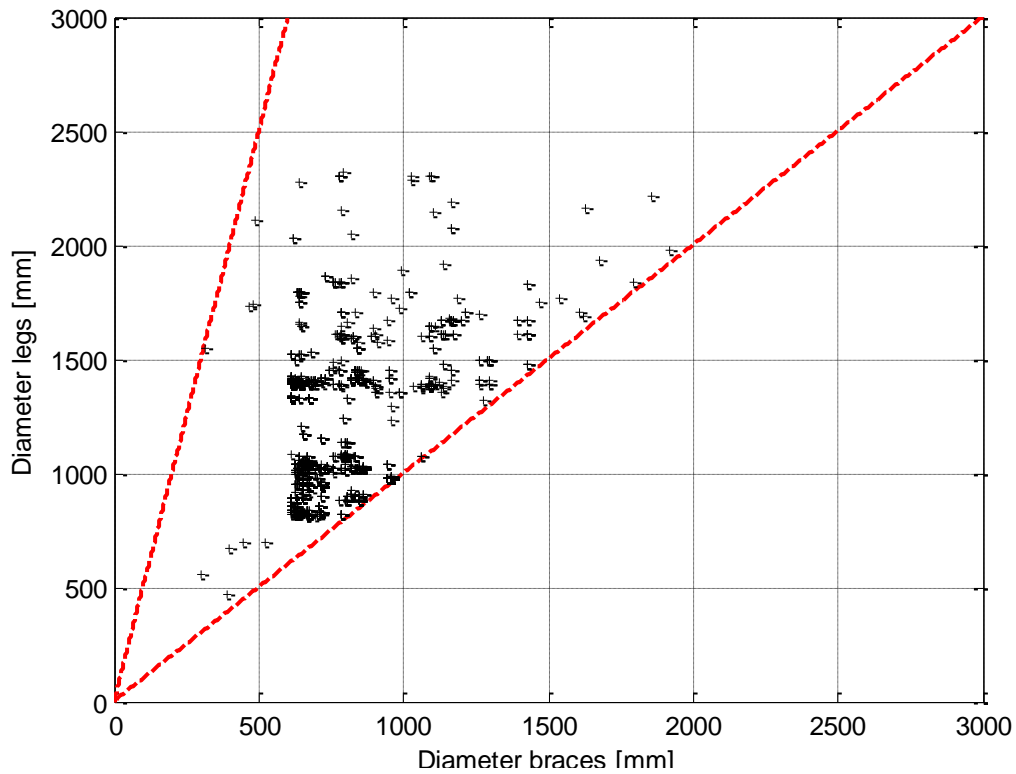


Figure 3-28. Beta constrain for legs and braces when optimizing the weight (50 generations and 10 individuals)

3.1.2.7.3 Tau parameter

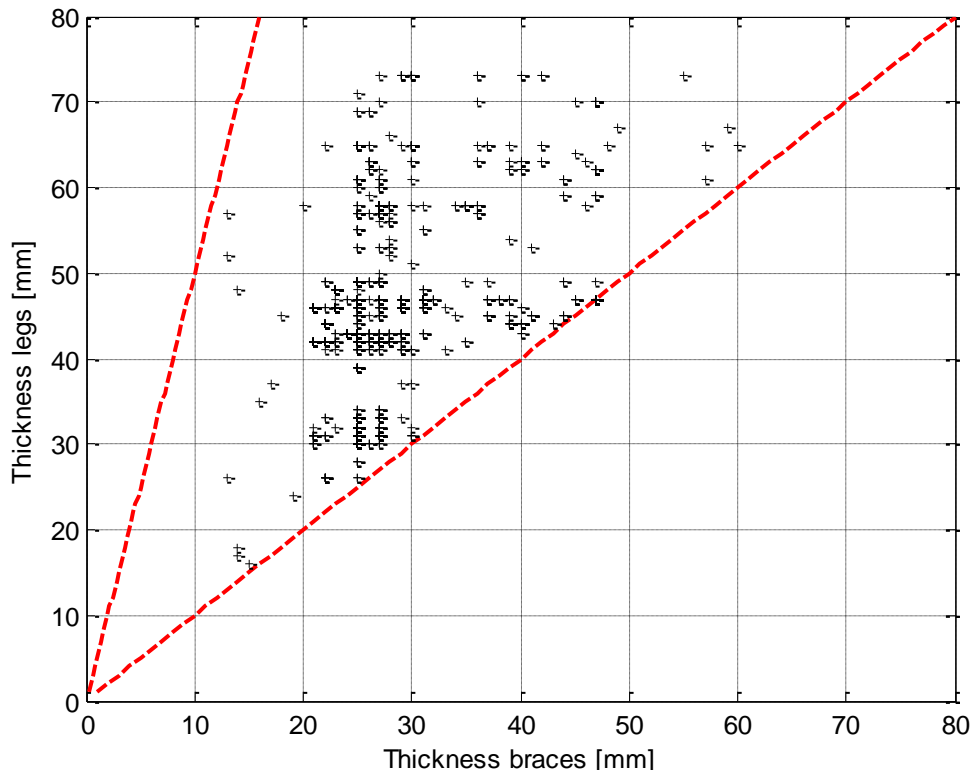


Figure 3-29. Tau constrain for legs and braces when optimizing the weight (50 generations and 10 individuals)

3.2 Objective function based on the damage

After running the algorithm with this objective function we will end up with the design that has the highest damage among all the designs that had been tested. In a way that means that the weight is also optimize because as the damage increases the weight decreases.

3.2.1 Geometry optimization

3.2.1.1 Weight range

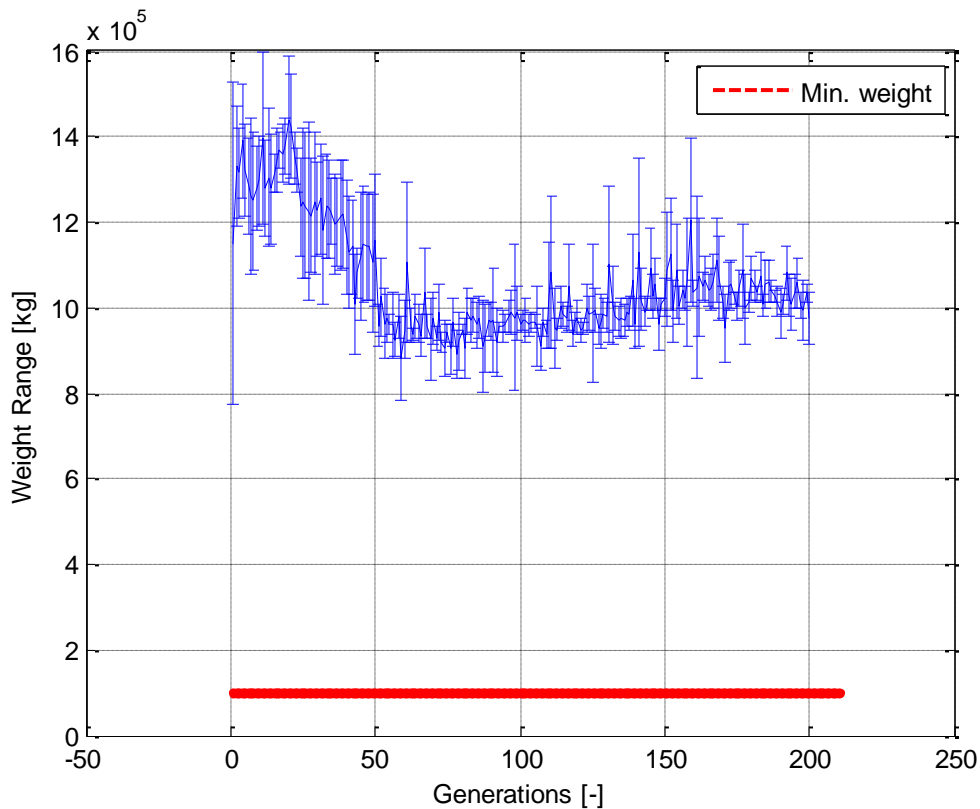


Figure 3-30. Weight of the structure [kg], when optimizing the damage (200 generations and 9 individuals)

The lightest design of the last generation weight 915.563 kg and the heaviest 1.036.481 kg , but the best one is in between and weight 1.009.061 kg.

3.2.1.2 Diameter progress

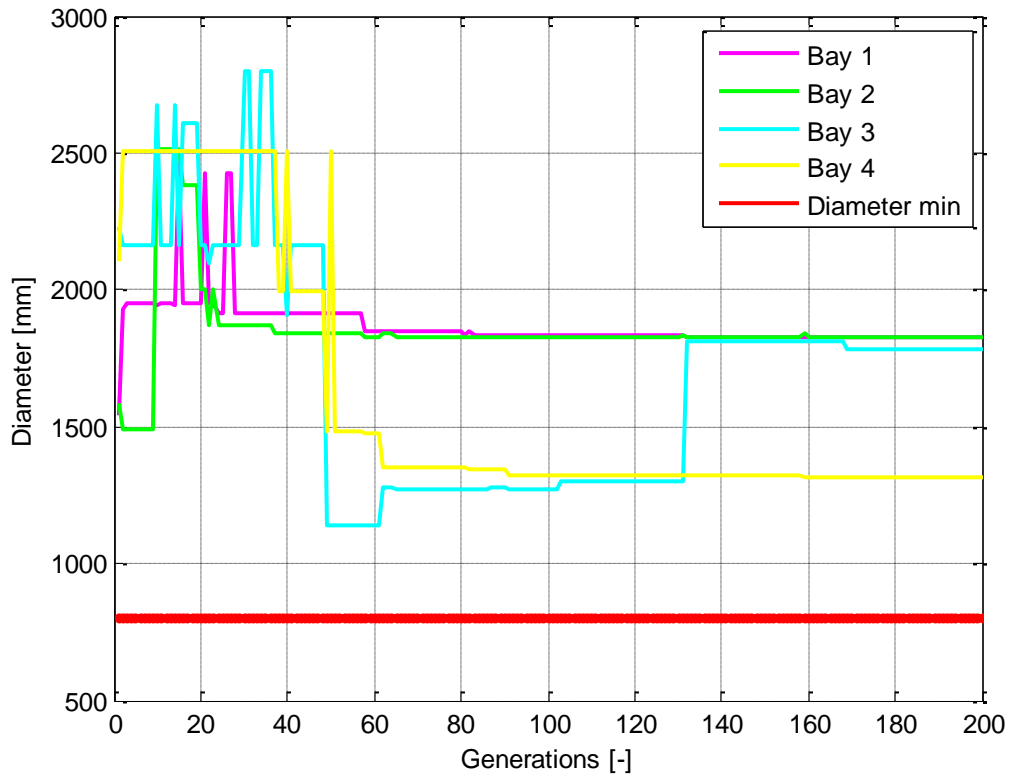


Figure 3-31. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the damage (200 generations and 9 individuals)

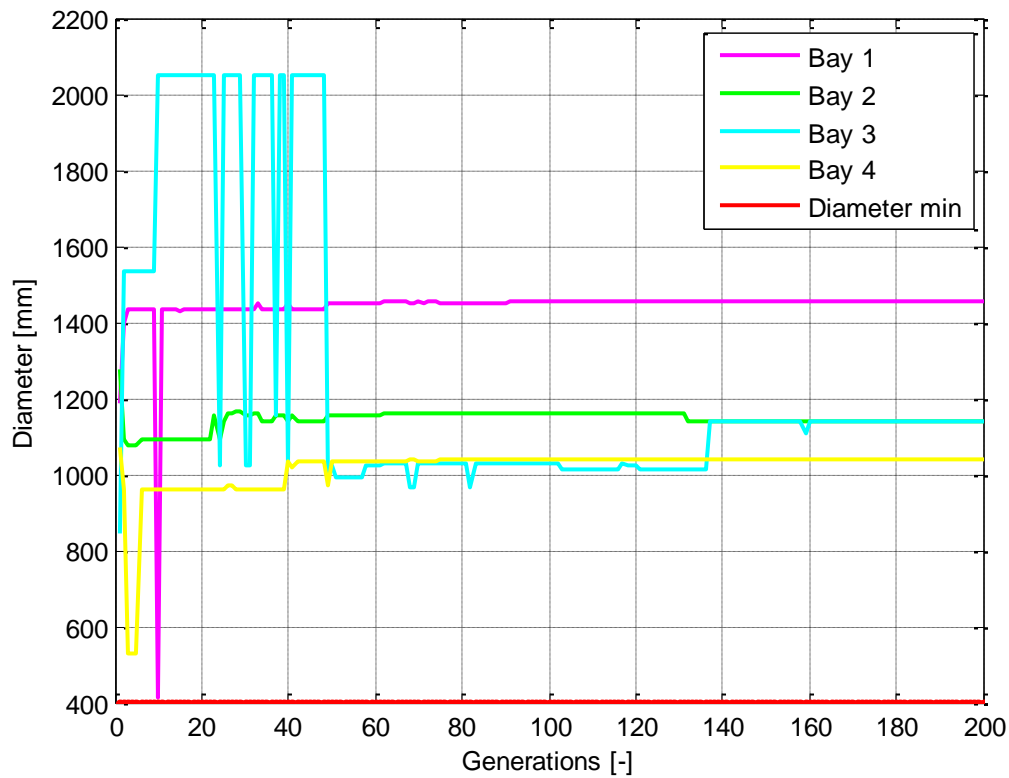


Figure 3-32. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the damage (200 generations and 9 individuals)

The exact values of diameter that the legs and braces take at the end of the simulation are presented in the table below:

	DIAMETER [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	1.828	1.827	1.782	1.312
Brace	1.455	1.142	1.141	1.039

Table 3-6. Values of diameter at the last generation when optimizing the damage and the geometry (200 generations and 9 designs)

3.2.1.3 Thickness progress

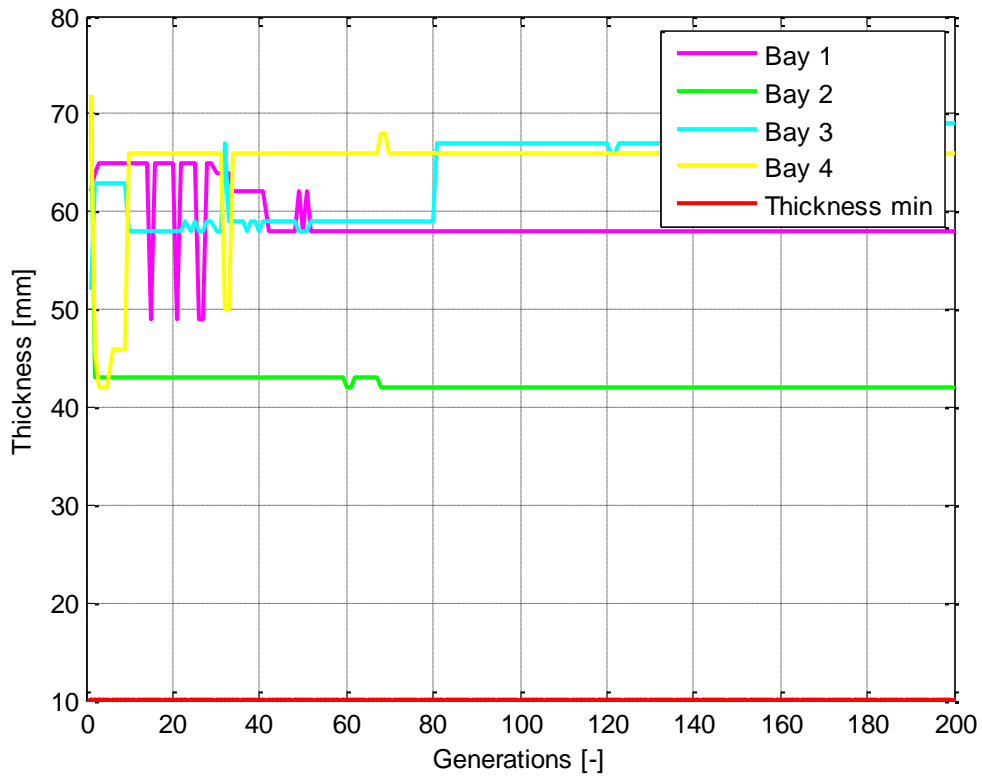


Figure 3-33. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the damage (200 generations and 9 individuals)

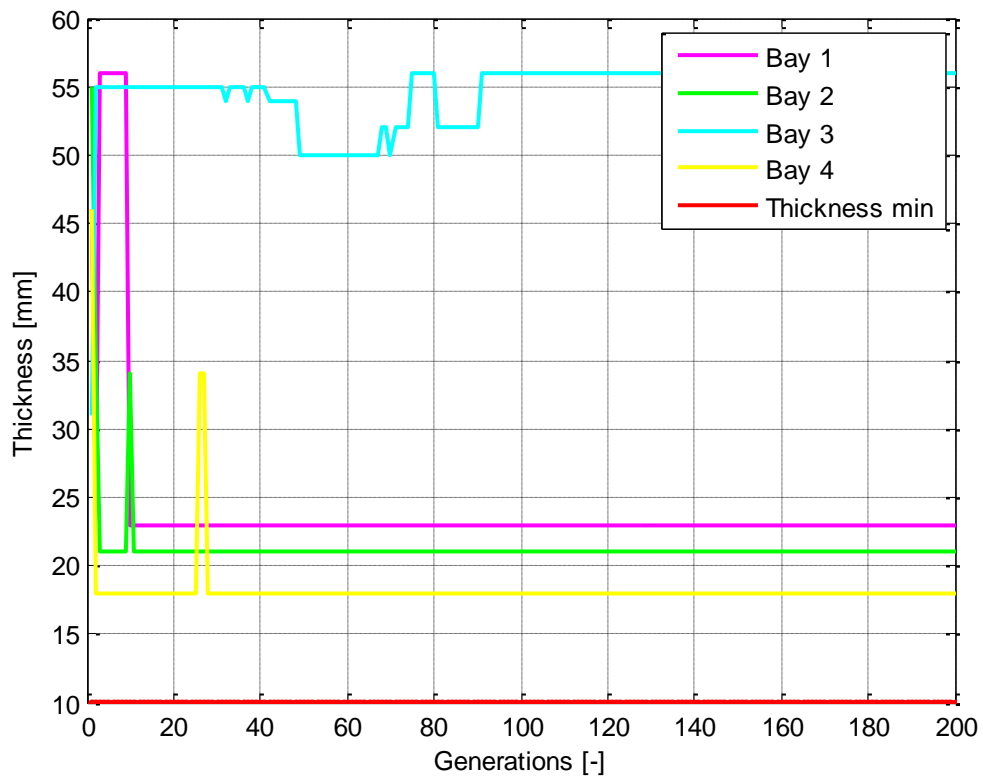


Figure 3-34. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the damage (200 generations and 9 individuals)

The exact values of thickness that the legs and braces take at the end of the simulation are presented in the table below:

	THICKNESS [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	58	42	69	66
Brace	23	21	56	18

Table 3-7. Values of thickness at the last generation when optimizing the damage and the geometry (200 generations and 9 designs)

3.2.1.4 Fitness

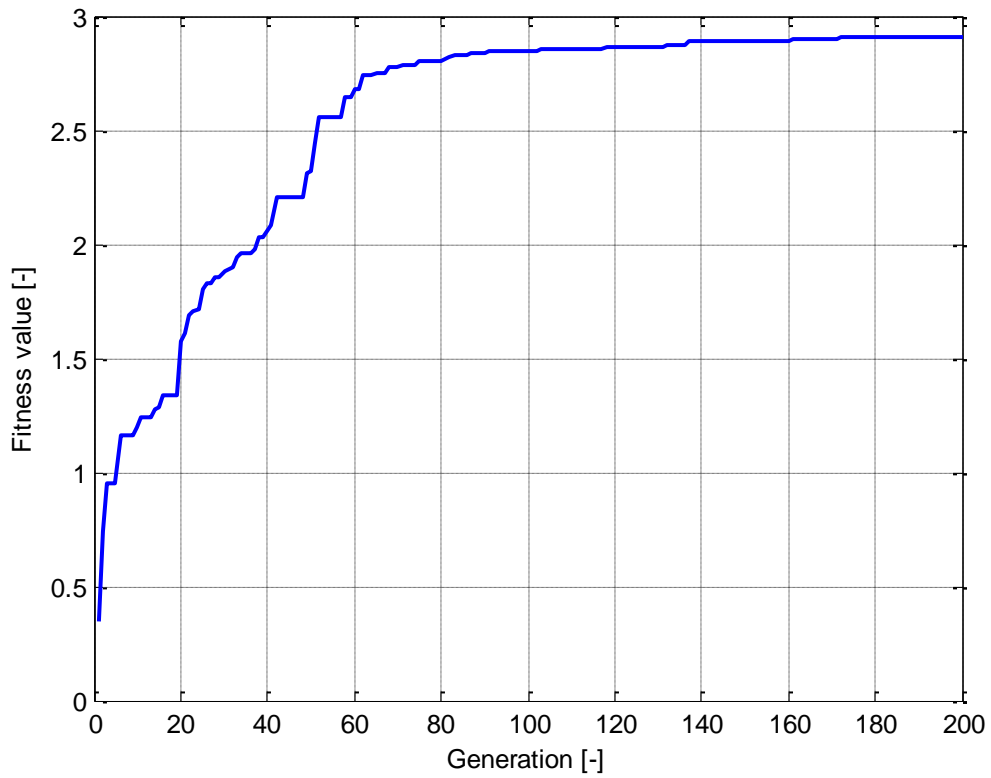


Figure 3-35. Improvement of the fitness through the generations when optimizing the damage (200 generations and 9 individuals)

3.2.1.5 Number of designs that change

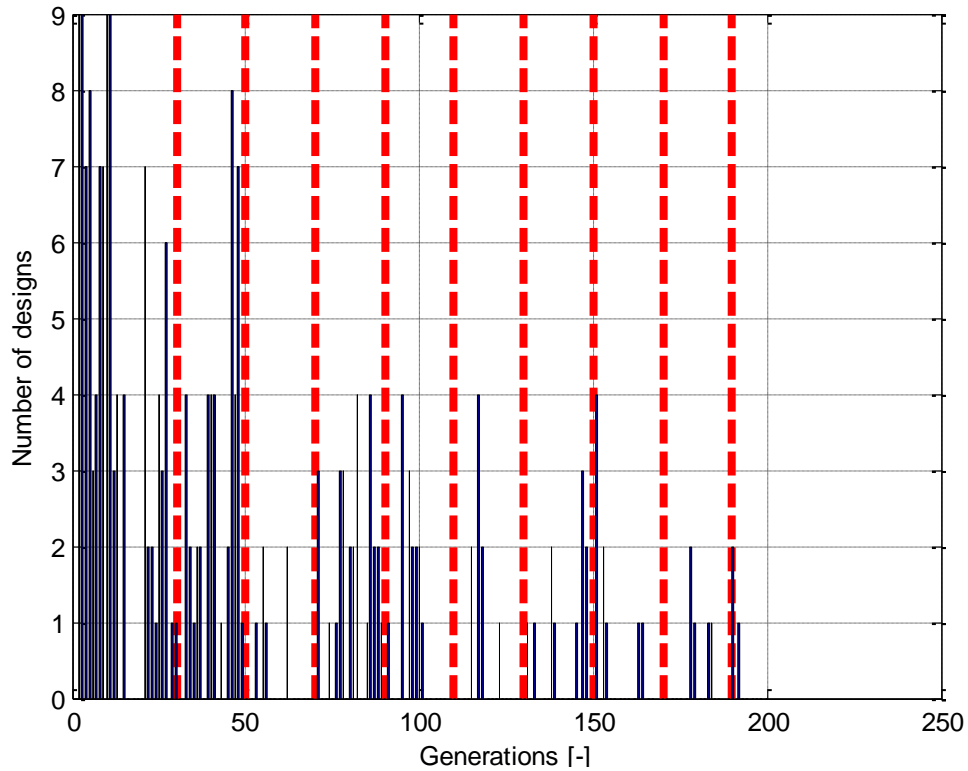


Figure 3-36. Number of designs of bay 1 that change from one generation to the next one when optimizing the damage (200 generations and 9 individuals)

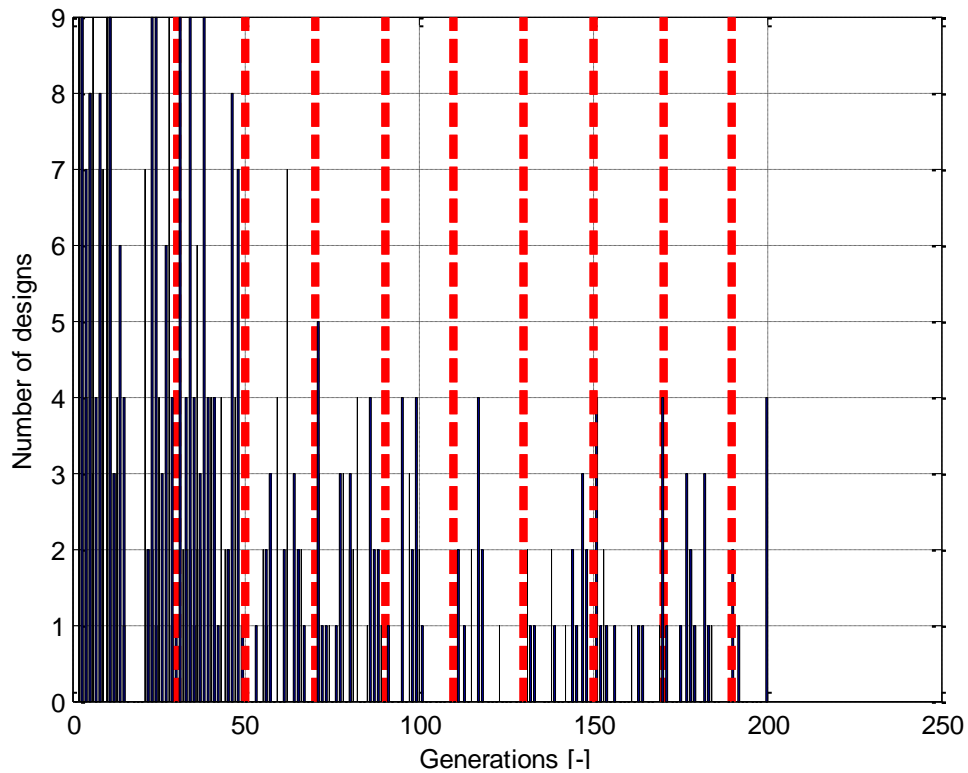


Figure 3-37. Number of designs of bay 2 that change from one generation to the next one when optimizing the damage (200 generations and 9 individuals)

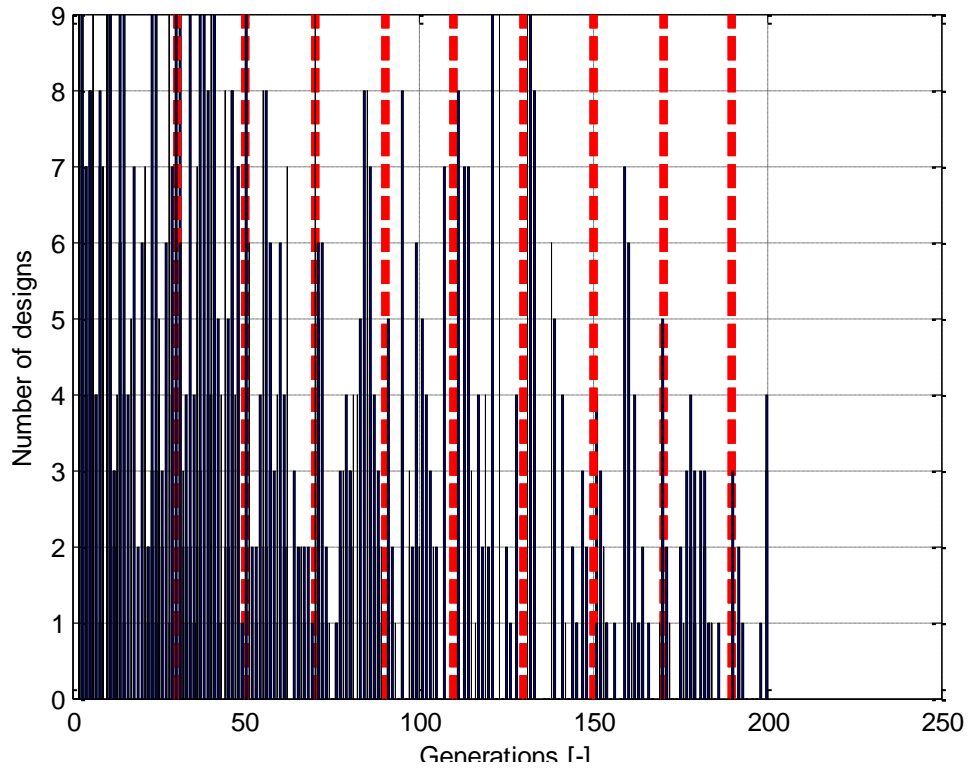


Figure 3-38. Number of designs of bay 3 that change from one generation to the next one when optimizing the damage (200 generations and 9 individuals)

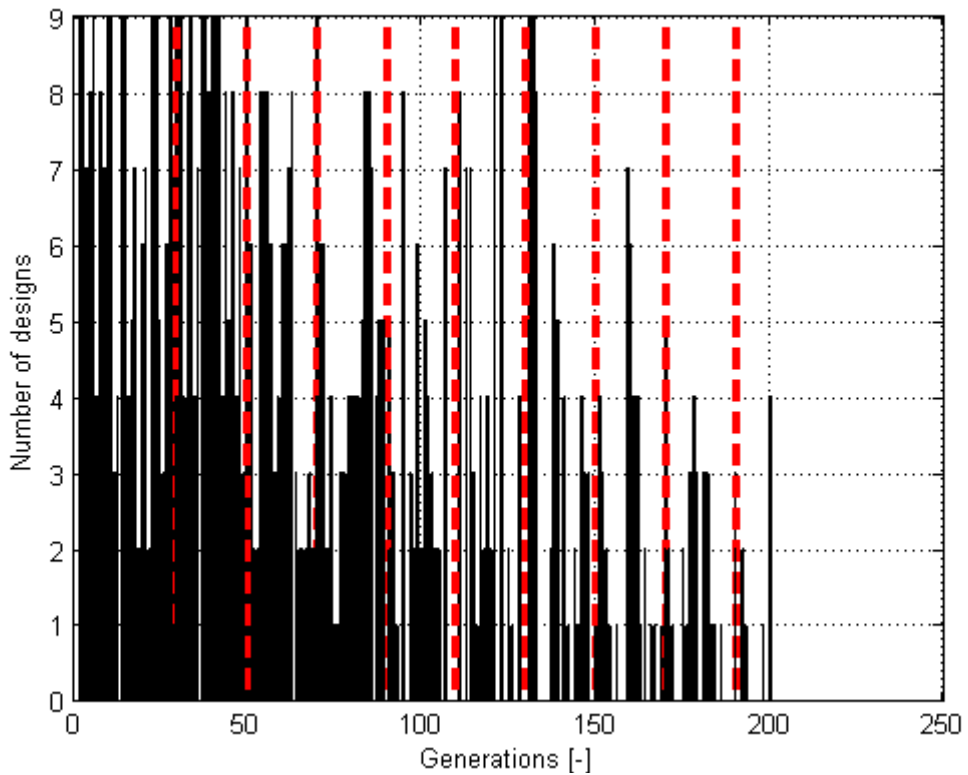


Figure 3-39. Number of designs of bay 4 that change from one generation to the next one when optimizing the damage (200 generations and 9 individuals)

3.2.1.6 SCF parameters constrains

3.2.1.6.1 Gamma parameter

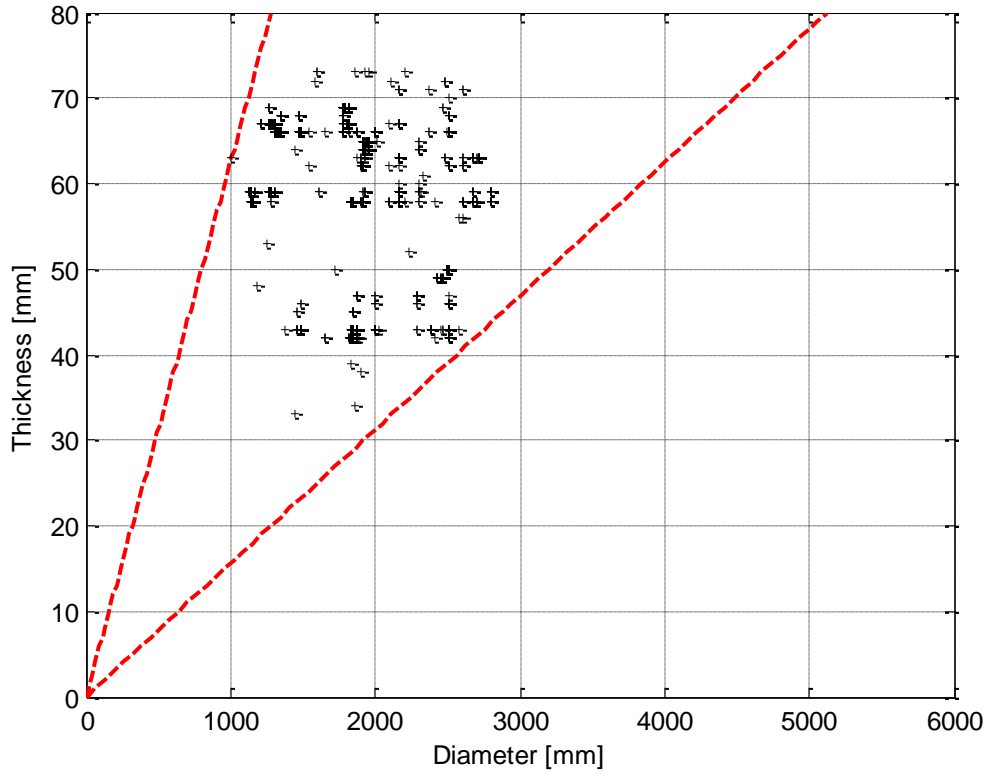


Figure 3-40. Gamma constriction for the legs when optimizing the damage (200 generations and 9 individuals)

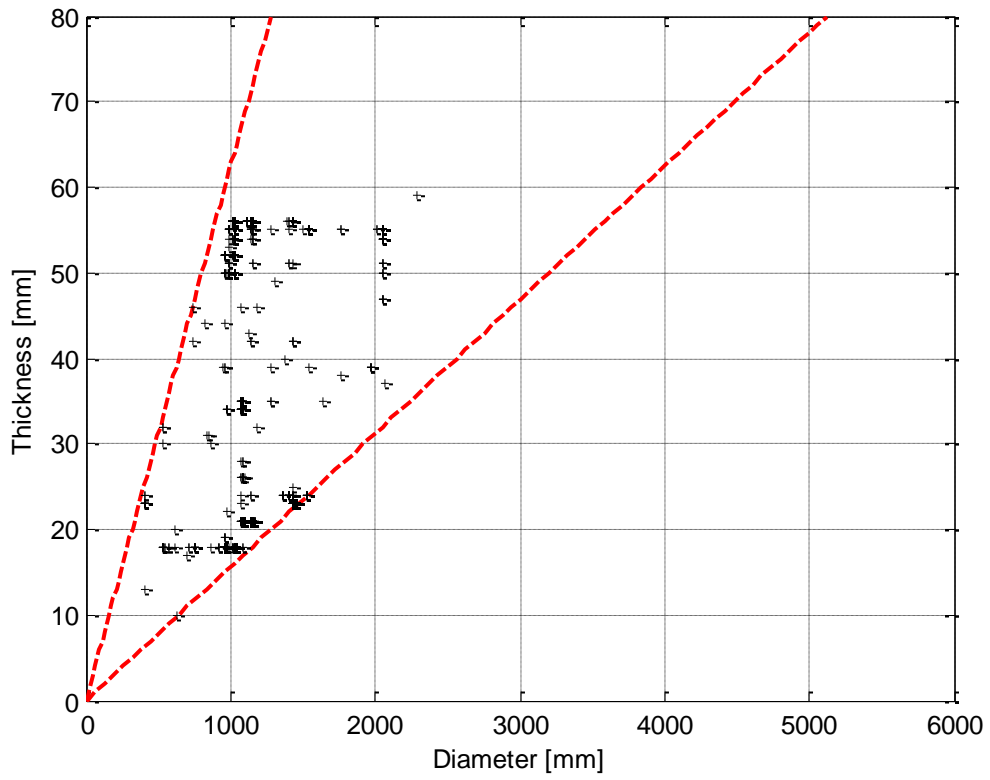


Figure 3-41. Gamma constriction for the braces when optimizing the damage (200 generations and 9 individuals)

3.2.1.6.2 Beta parameter

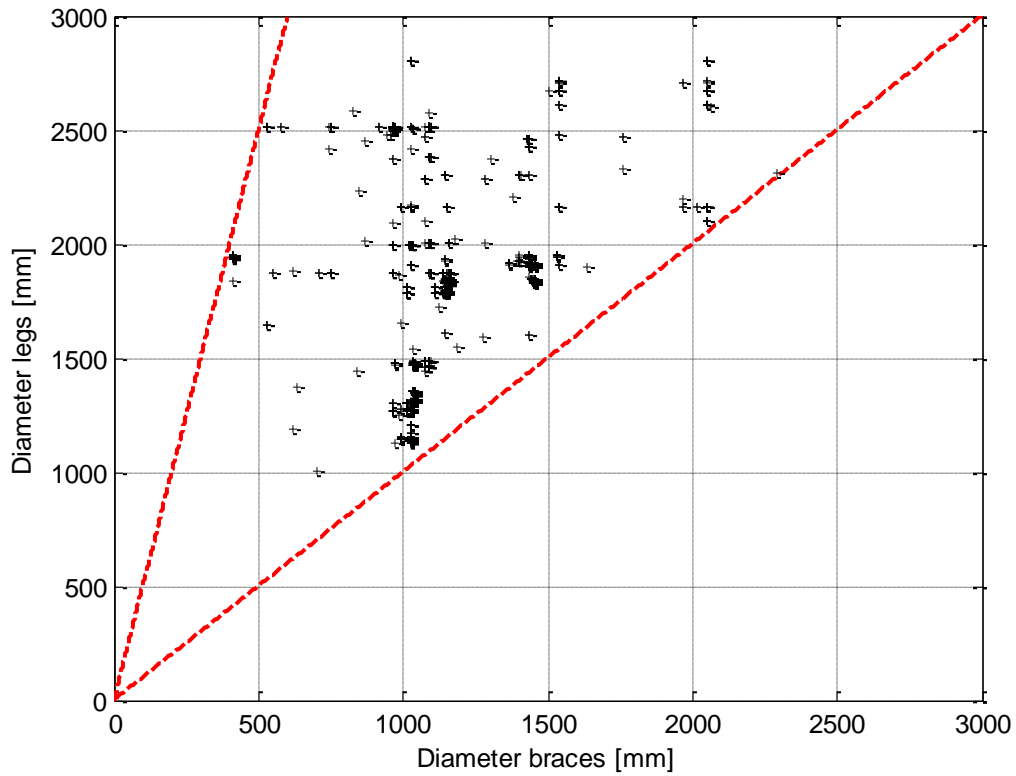


Figure 3-42. Beta constrain for legs and braces when optimizing the damage (200 generations and 9 individuals)

3.2.1.6.3 Tau parameter

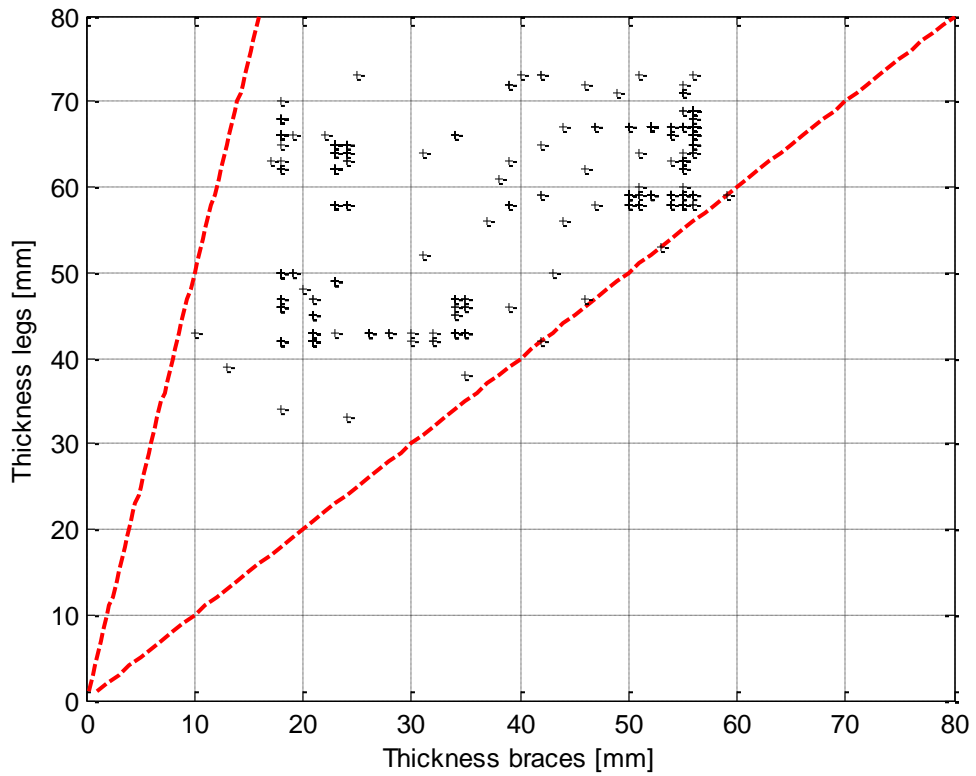


Figure 3-43. Tau constrain for legs and braces when optimizing the damage (200 generations and 9 individuals)

3.2.2 Topology and geometry optimization

3.2.2.1 Weight range

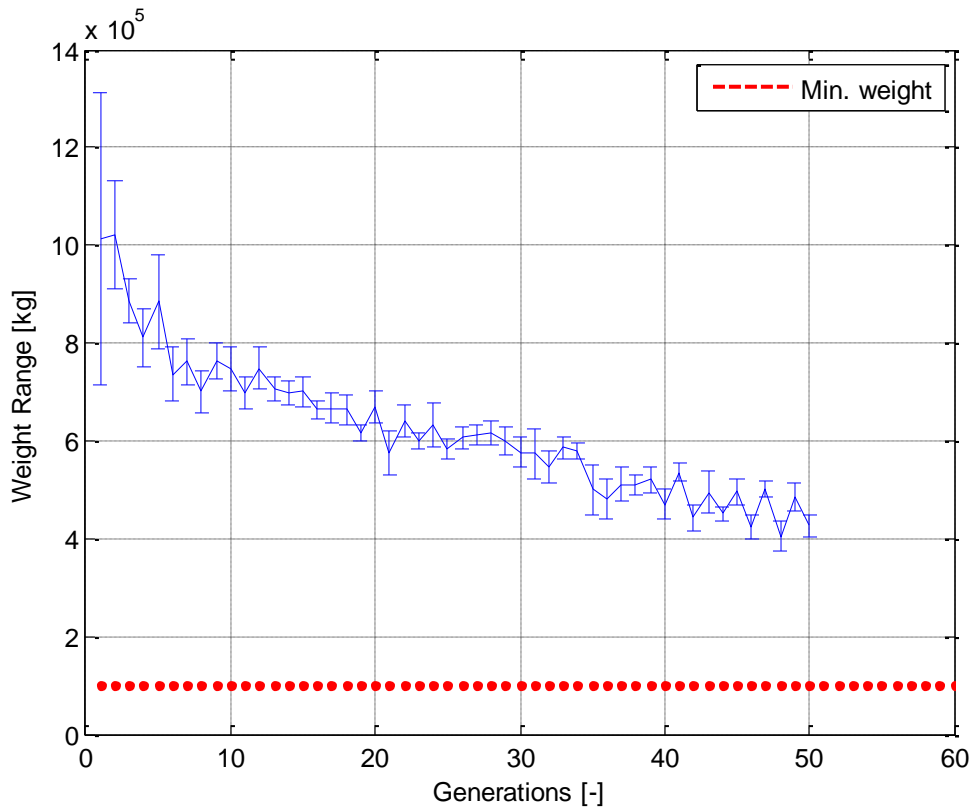


Figure 3-44. Weight of the structure [kg], when optimizing the damage (50 generations and 10 individuals)

The lightest design of the last generation and the one that fits the best weight 404.978 kg and the heaviest 447.146 kg.

3.2.2.2 Diameter progress

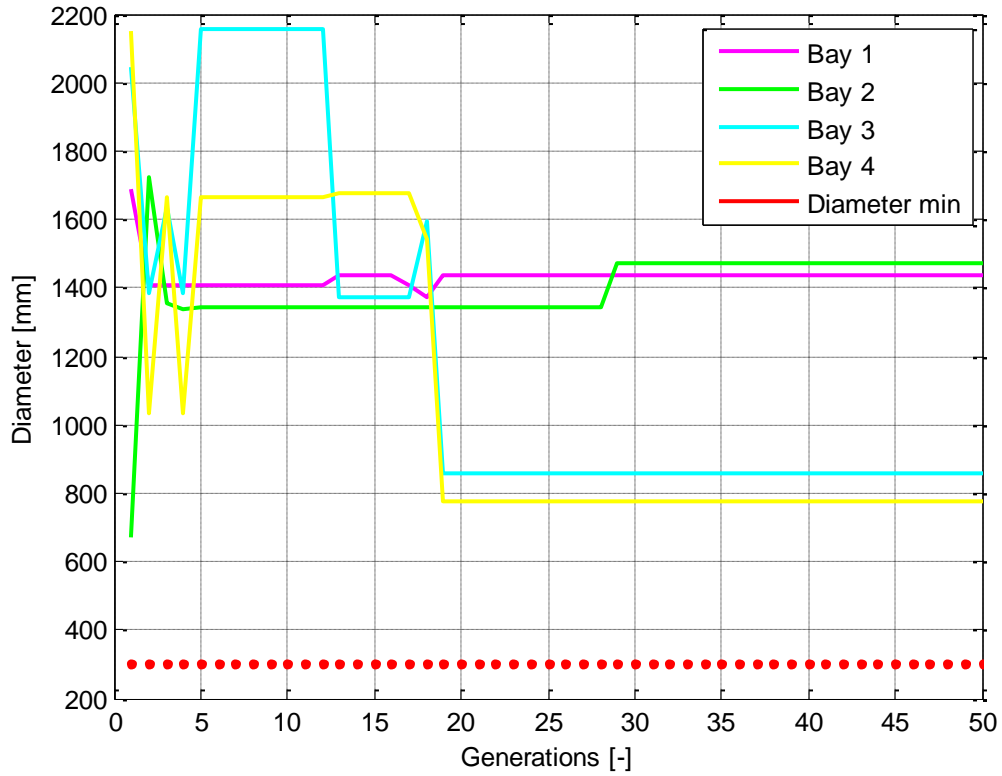


Figure 3-45. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the damage (50 generations and 10 individuals)

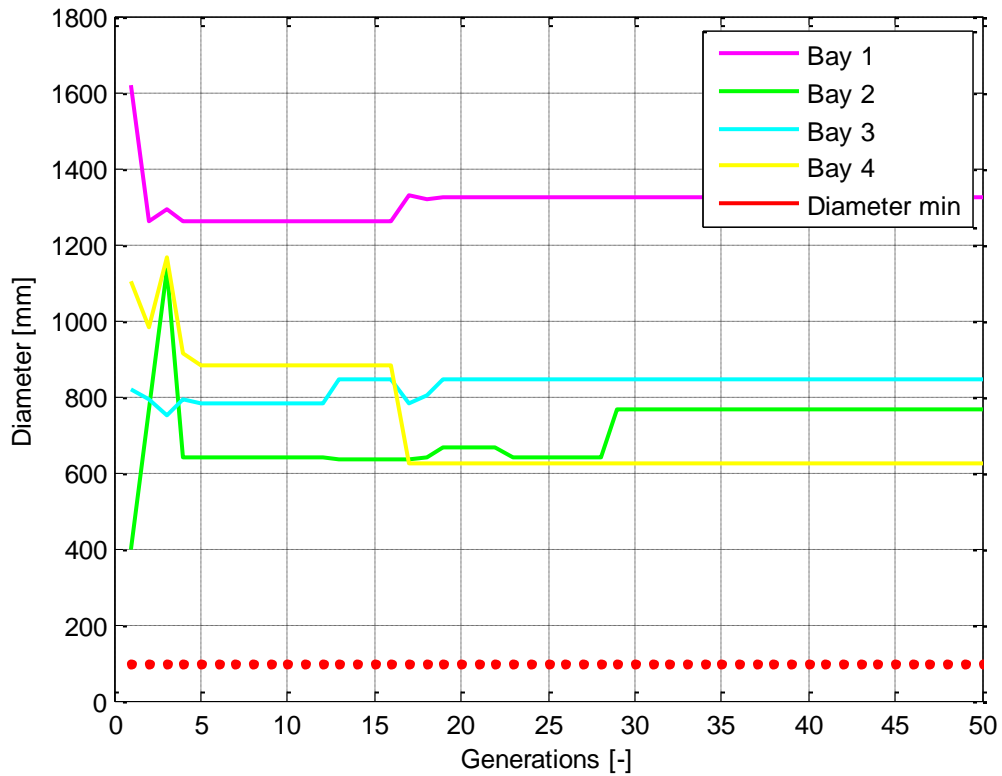


Figure 3-46. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the damage (50 generations and 10 individuals)

The exact values of diameter that the legs and braces take at the end of the simulation are presented in the table below:

	DIAMETER [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	1.438	1.474	855	778
Brace	1.322	768	844	625

Table 3-8. Values of diameter at the last generation when optimizing the damage the topology and the geometry (50 generations and 10 designs)

3.2.2.3 Thickness progress

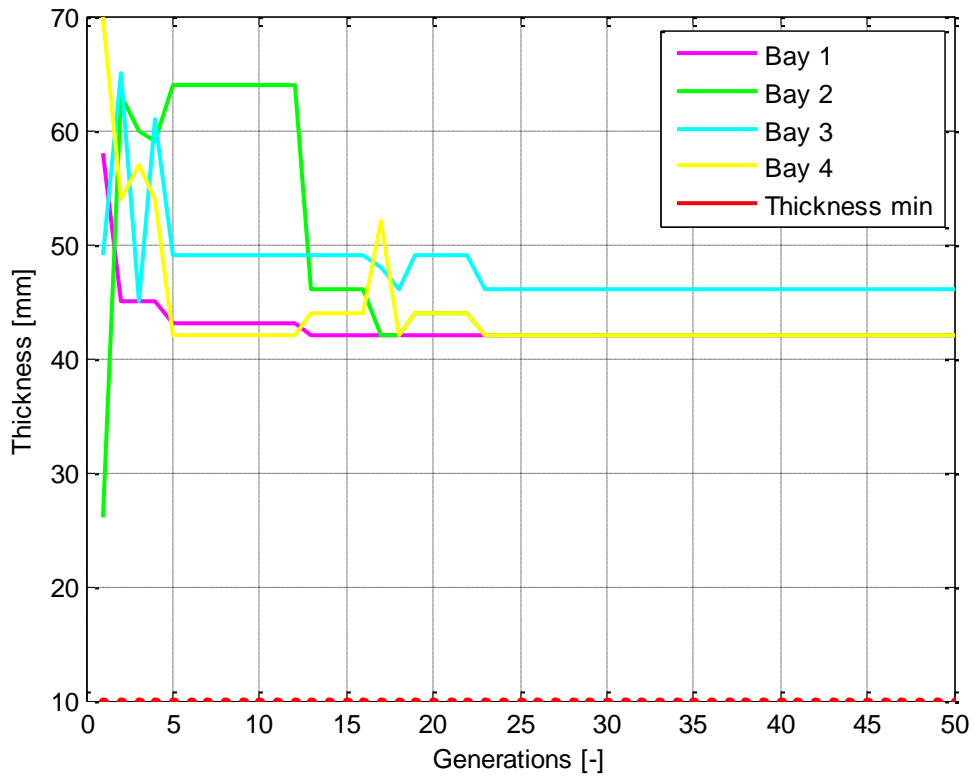


Figure 3-47. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the damage (50 generations and 10 individuals)

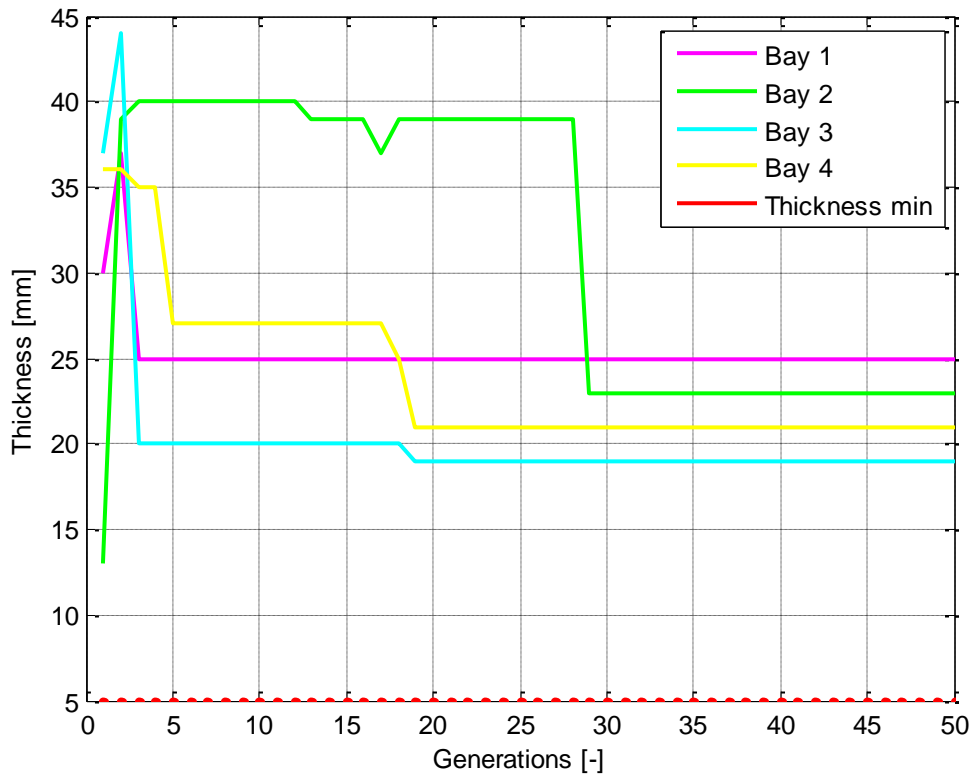


Figure 3-48. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the damage (50 generations and 10 individuals)

The exact values of thickness that the legs and braces take at the end of the simulation are presented in the table below:

	THICKNESS [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	42	42	46	42
Brace	25	23	19	21

Table 3-9. Values of thickness at the last generation when optimizing the damage the topology and the geometry (50 generations and 10 designs)

3.2.2.4 Topology overview

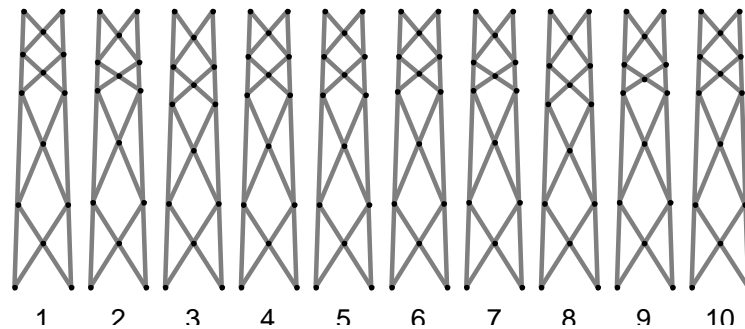


Figure 3-49. Location of the nodes of the 10 best designs when optimizing the weight (50 generations and 10 individuals)

And the length of the bays of the best design is written in the table that follows:

LENGTH [mm]				
	Bay 1	Bay 2	Bay 3	Bay 4
Length	17.555	23.873	8.132	9.218

Table 3-10. Values of the length of the bays of the best design at the last generation when optimizing the damage the topology and the geometry (50 generations and 10 designs)

3.2.2.5 Fitness

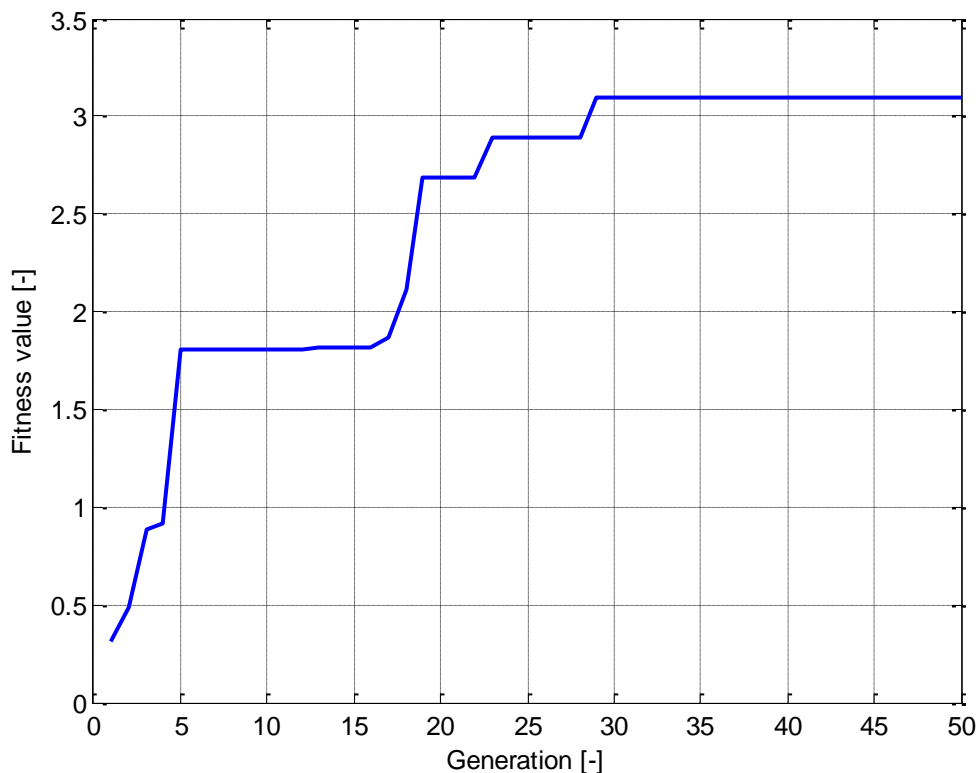


Figure 3-50. Improvement of the fitness through the generations when optimizing the damage (50 generations and 10 individuals)

3.2.2.6 Number of designs that change

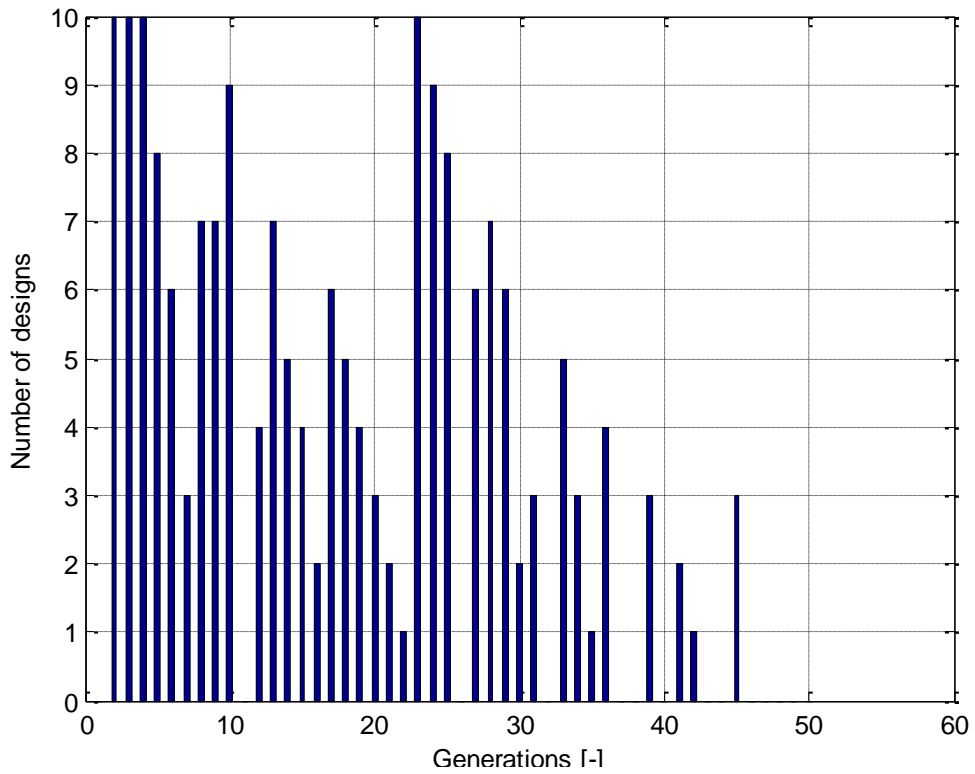


Figure 3-51. Number of designs of bay 1 that change from one generation to the next one when optimizing the damage (50 generations and 10 individuals)

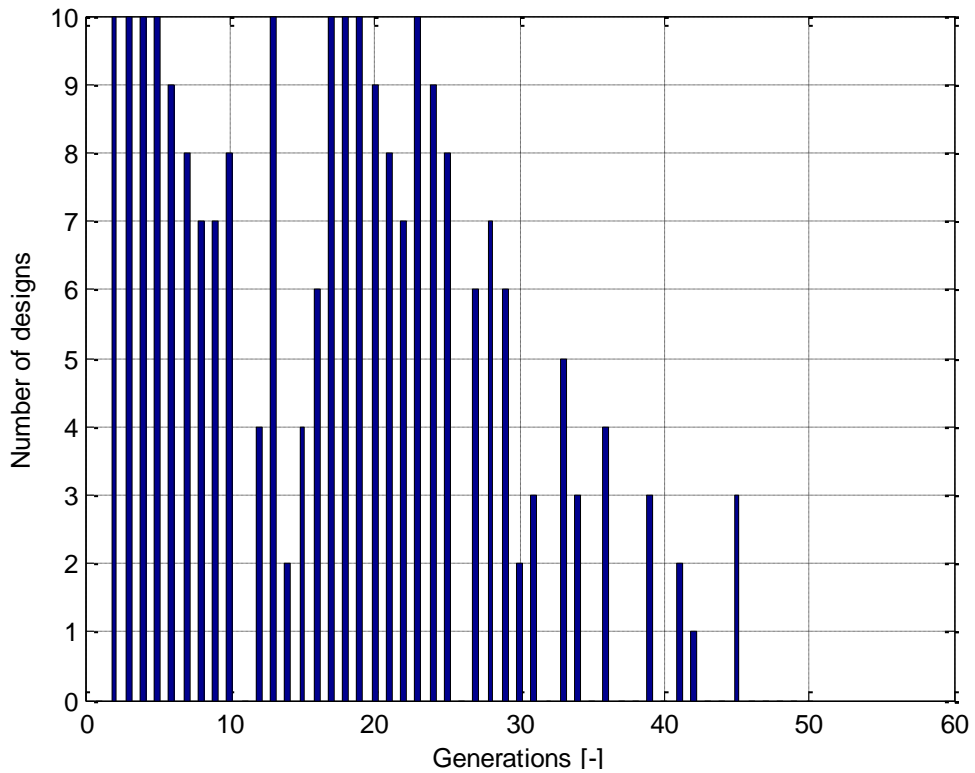


Figure 3-52. Number of designs of bay 2 that change from one generation to the next one when optimizing the damage (50 generations and 10 individuals)

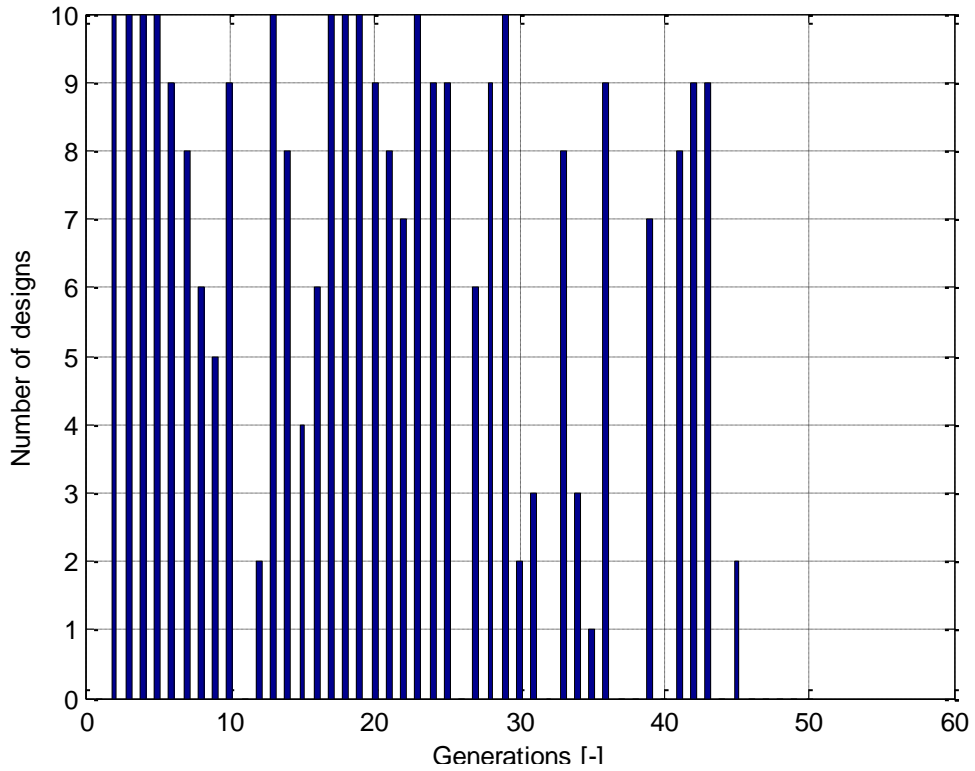


Figure 3-53. Number of designs of bay 3 that change from one generation to the next one when optimizing the damage (50 generations and 10 individuals)

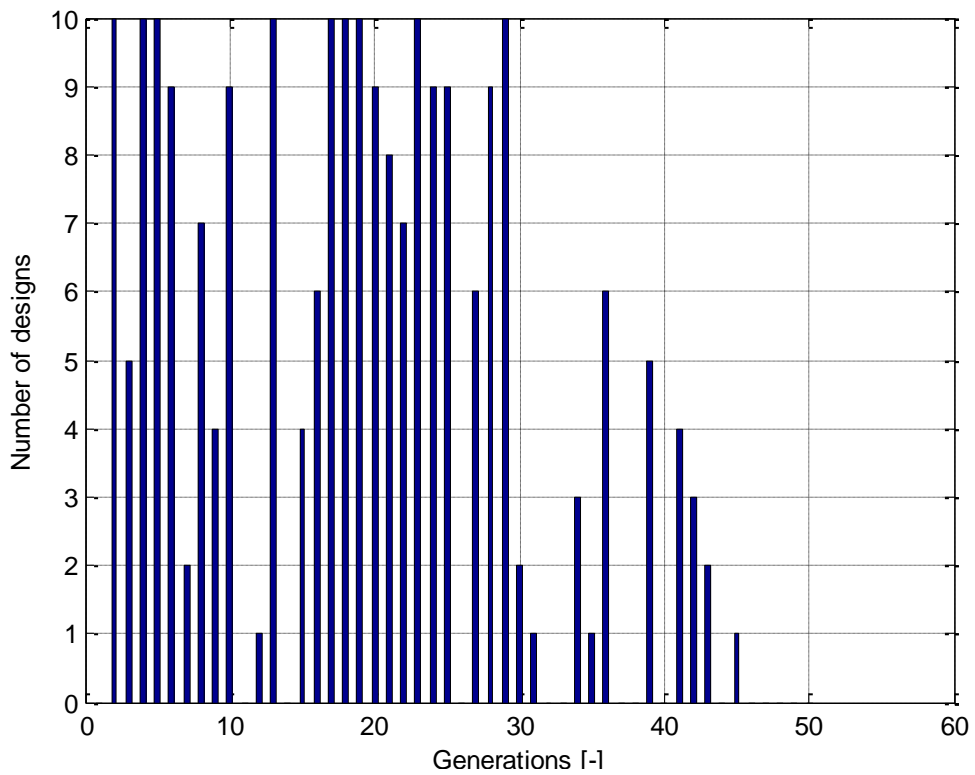


Figure 3-54. Number of designs of bay 4 that change from one generation to the next one when optimizing the damage (50 generations and 10 individuals)

3.2.2.7 SCF parameters constrains

3.2.2.7.1 Gamma parameter

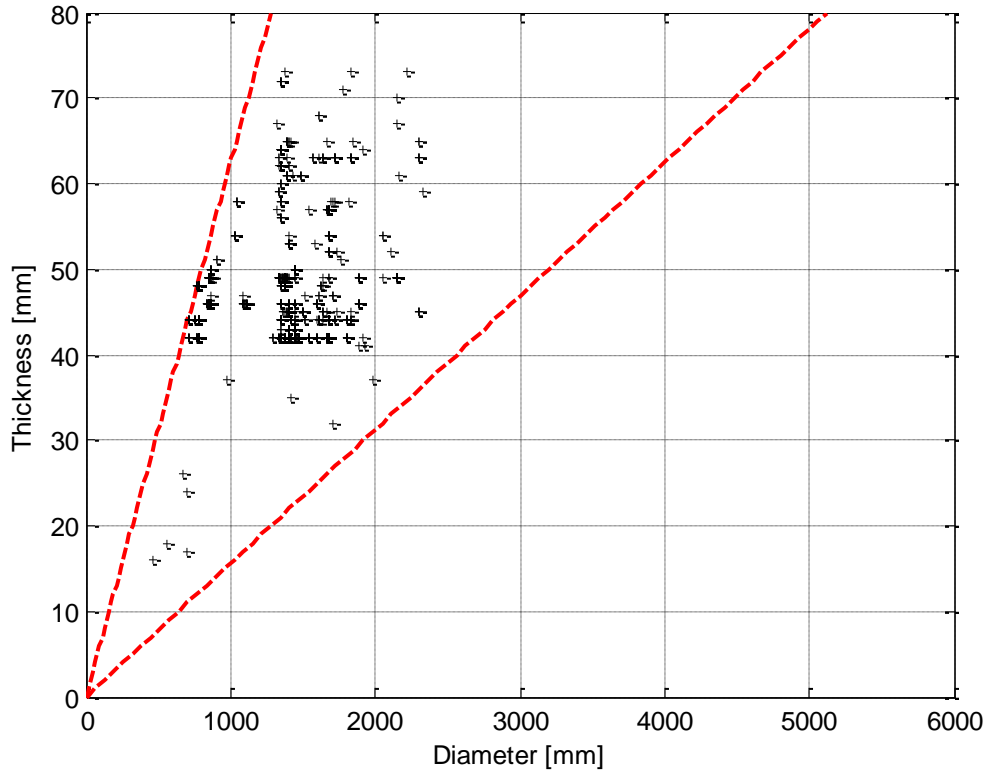


Figure 3-55. Gamma constriction for the legs when optimizing the damage (50 generations and 10 individuals)

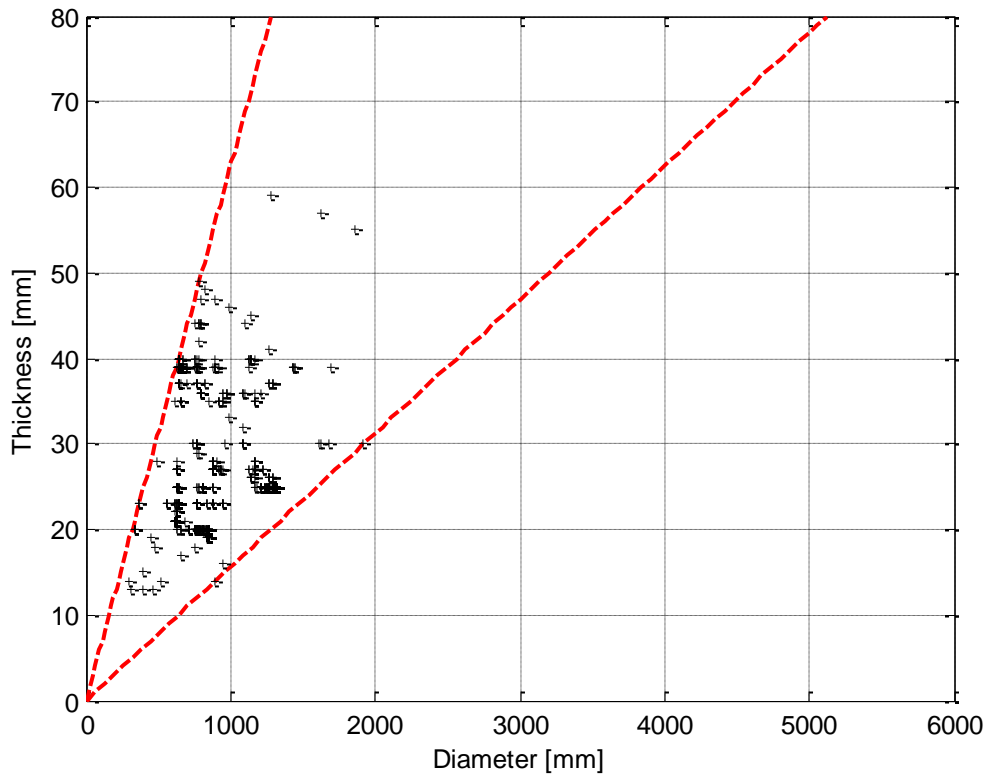


Figure 3-56. Gamma constriction for the braces when optimizing the damage (50 generations and 10 individuals)

3.2.2.7.2 Beta parameter

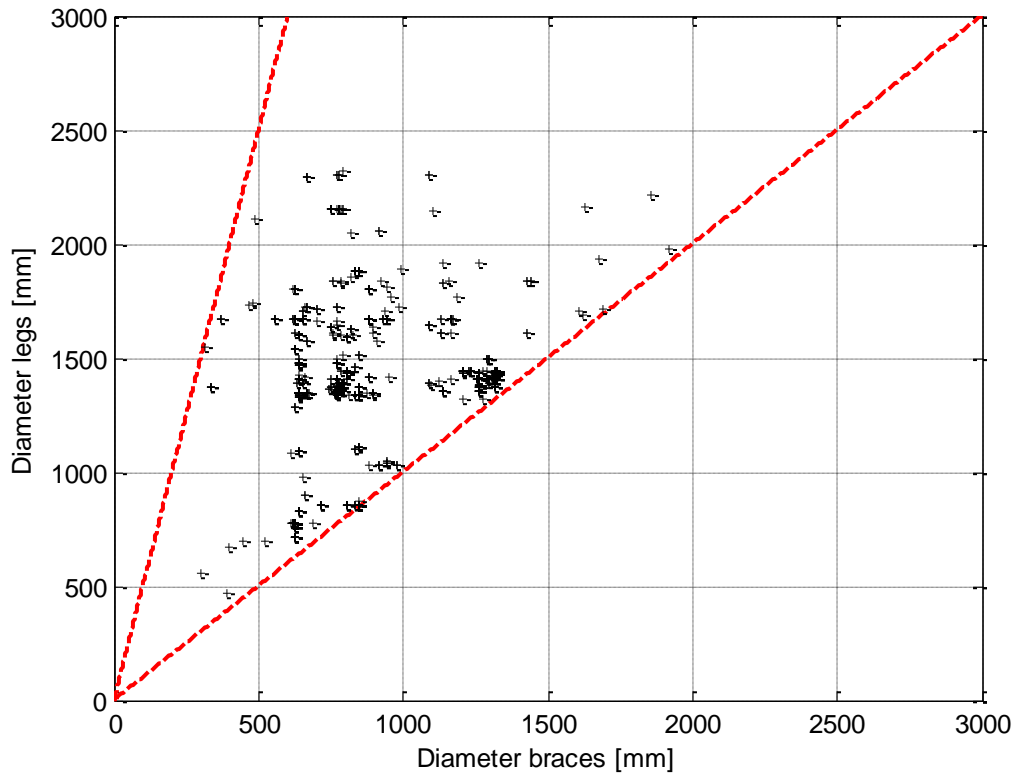


Figure 3-57. Beta constrain for legs and braces when optimizing the damage (50 generations and 10 individuals)

3.2.2.7.3 Tau parameter

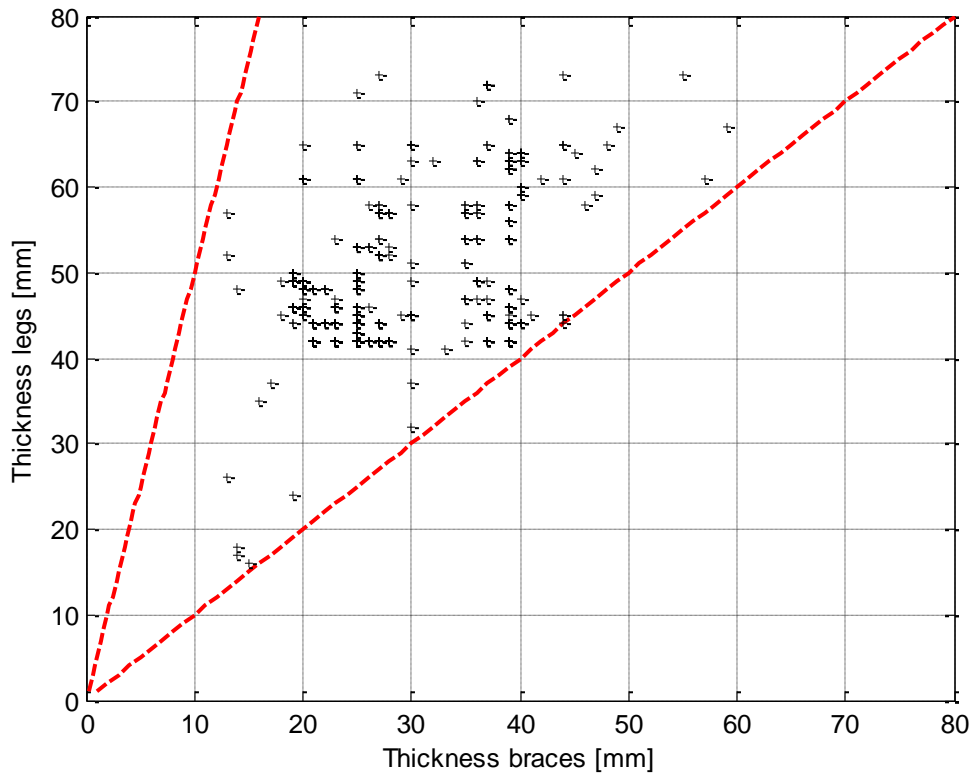


Figure 3-58. Tau constrain for legs and braces when optimizing the damage (50 generations and 10 individuals)

3.3 Different objective function for each step

For the odd iterations a weight optimization is carried out and for the even the damage is optimized. Of course for the odd generations only the structures with damage under the limits are analyzed and when the generation is even if the design is heavier than the maximum weight is rejected.

3.3.1 Geometry optimization

3.3.1.1 Weight range

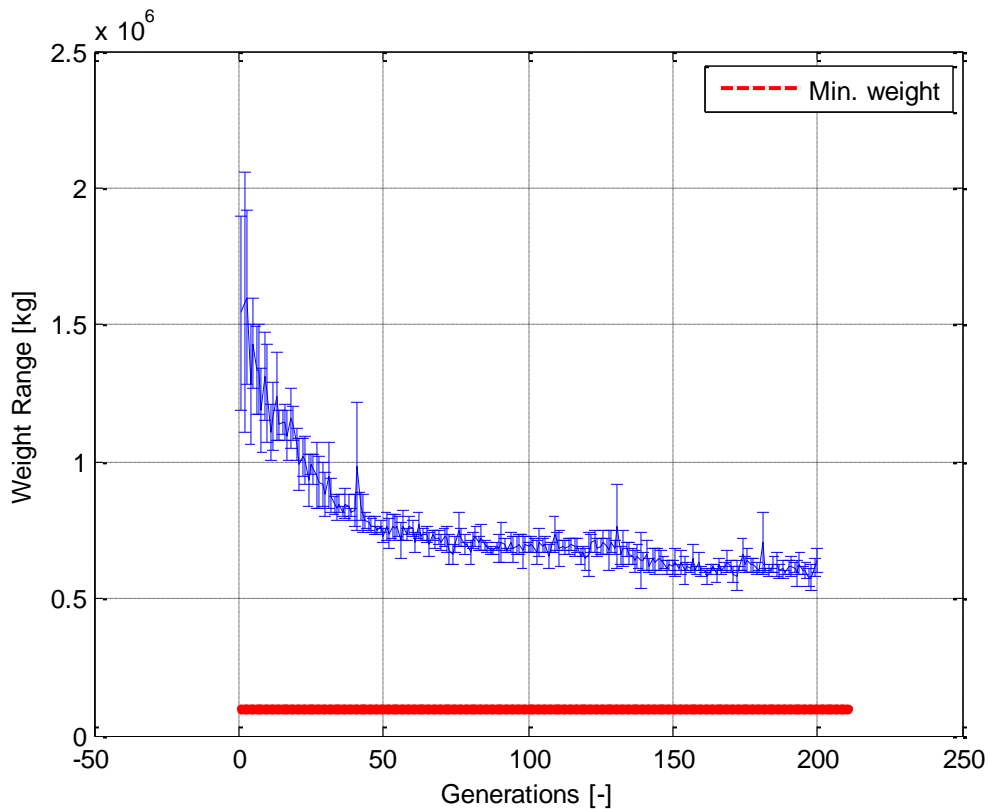


Figure 3-59. Weight of the structure [kg], when optimizing the weight and damage (200 generations and 9 individuals)

The lightest design of the last generation weight 595.331 kg and the heaviest 683.734 kg , but the best one is in between and weight 595.715 kg.

3.3.1.2 Diameter progress

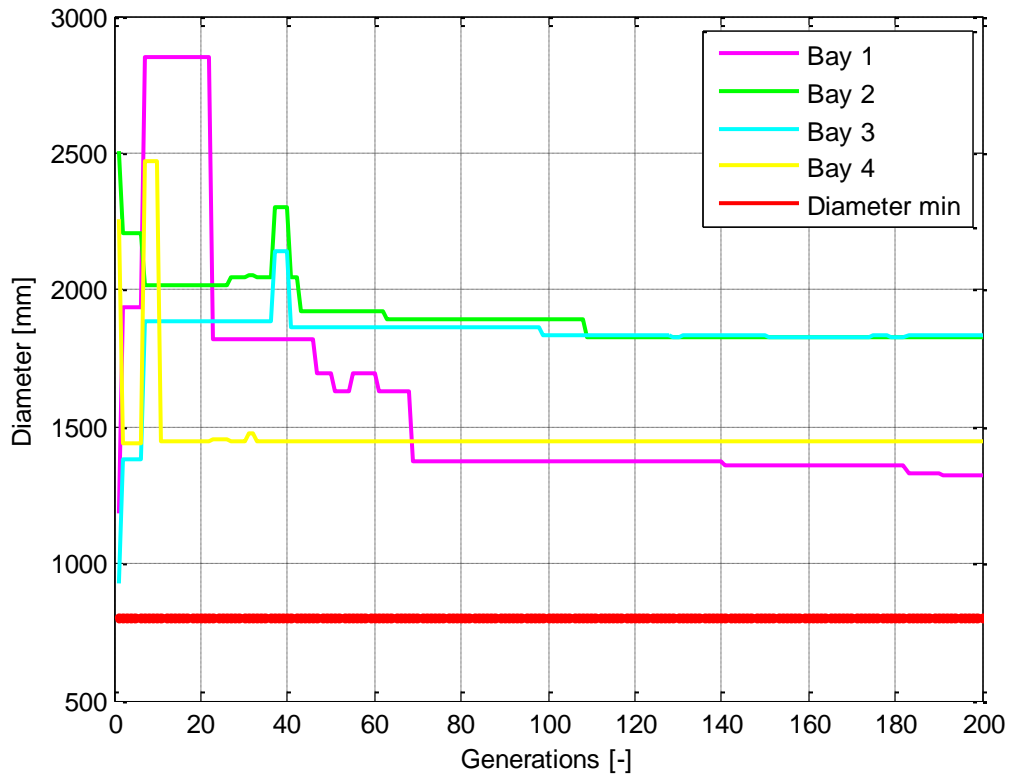


Figure 3-60. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight and damage (200 generations and 9 individuals)

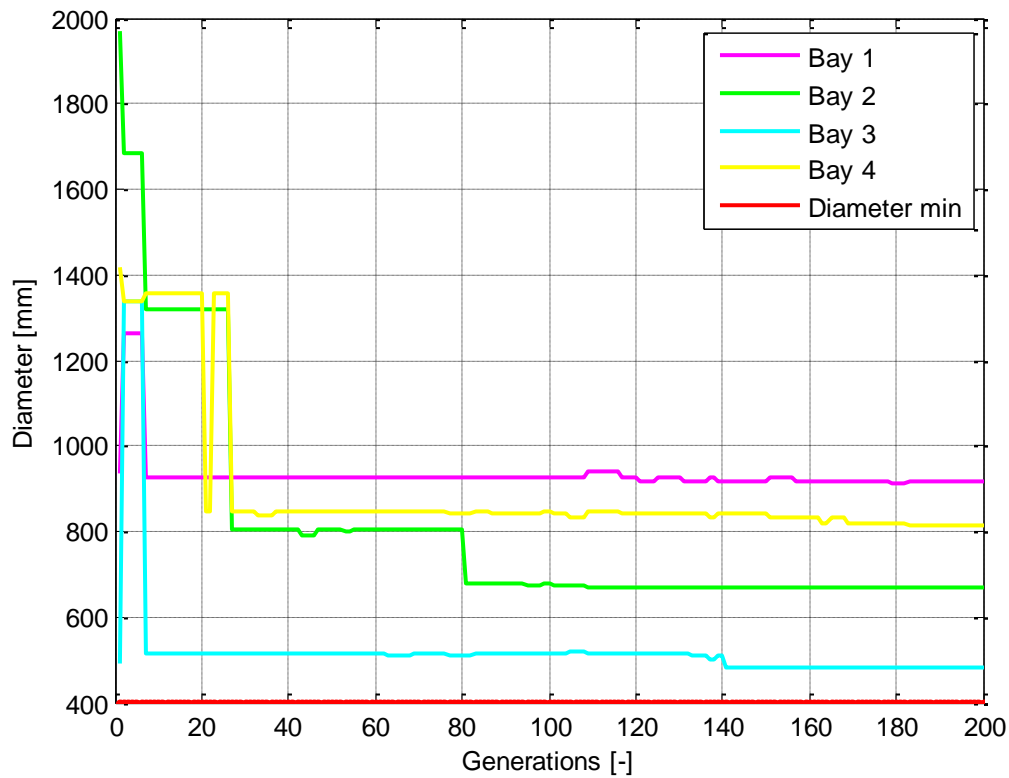


Figure 3-61. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight and damage (200 generations and 9 individuals)

The exact values of diameter that the legs and braces take at the end of the simulation are presented in the table below:

	DIAMETER [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	1.325	1.825	1.834	1.443
Brace	917	667	482	816

Table 3-11. Values of diameter at the last generation when optimizing the weight, the damage and the geometry (200 generations and 9 designs)

3.3.1.3 Thickness progress

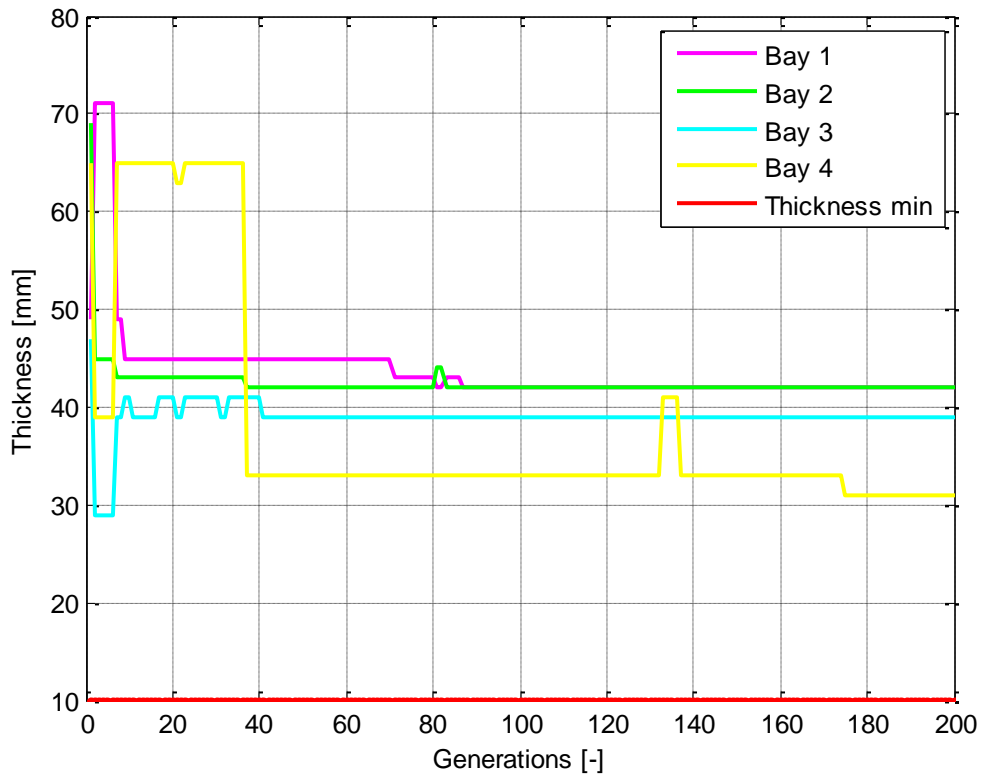


Figure 3-62. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight and damage (200 generations and 9 individuals)

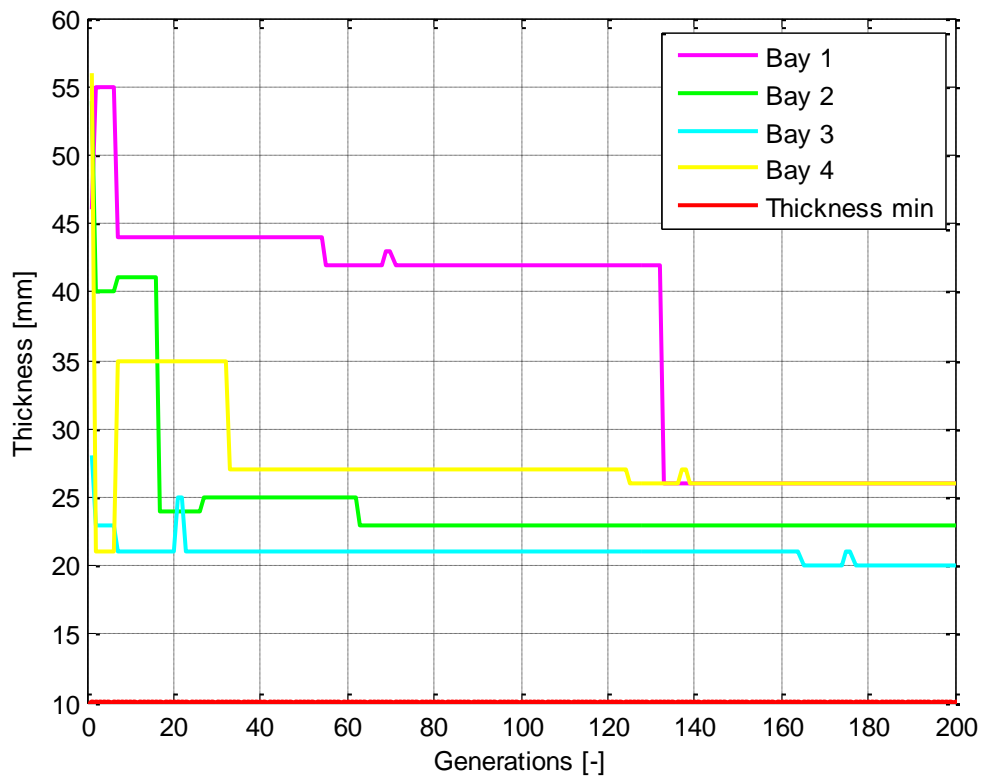


Figure 3-63. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight and damage (200 generations and 9 individuals)

The exact values of thickness that the legs and braces take at the end of the simulation are presented in the table below:

	THICKNESS [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	42	42	39	31
Brace	26	23	20	26

Table 3-12. Values of thickness at the last generation when optimizing the weight, the damage and the geometry (200 generations and 9 designs)

3.3.1.4 Fitness

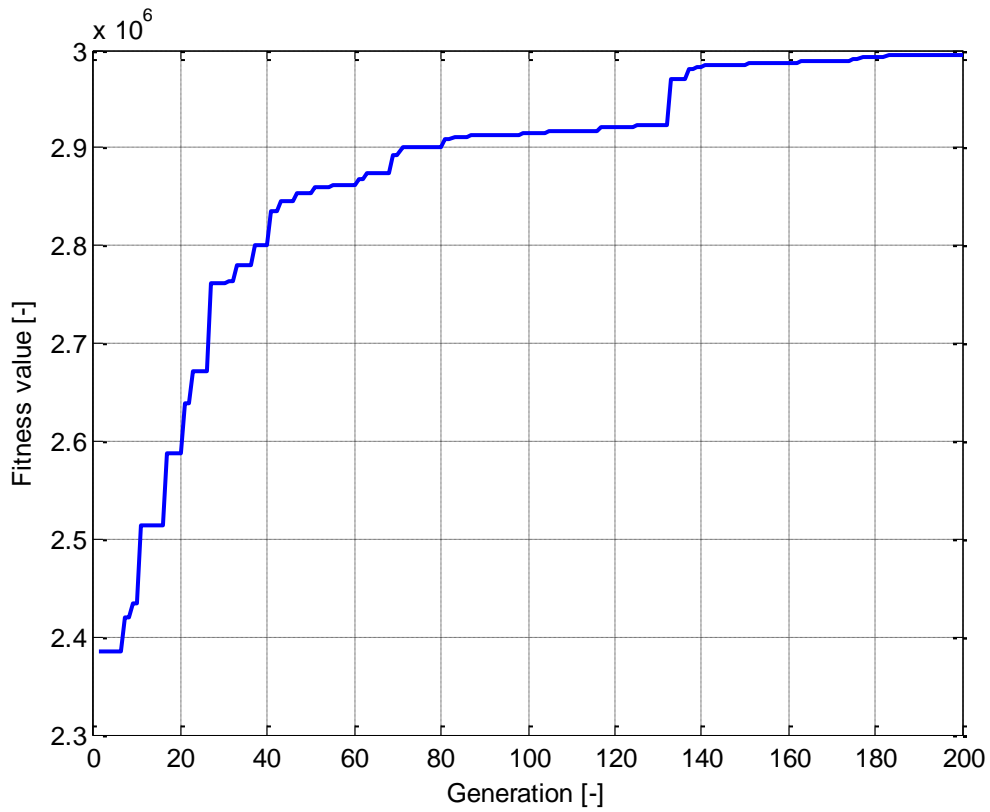


Figure 3-64. Improvement of the fitness through the generations when optimizing the weight and damage (200 generations and 9 individuals)

3.3.1.5 Number of designs that change

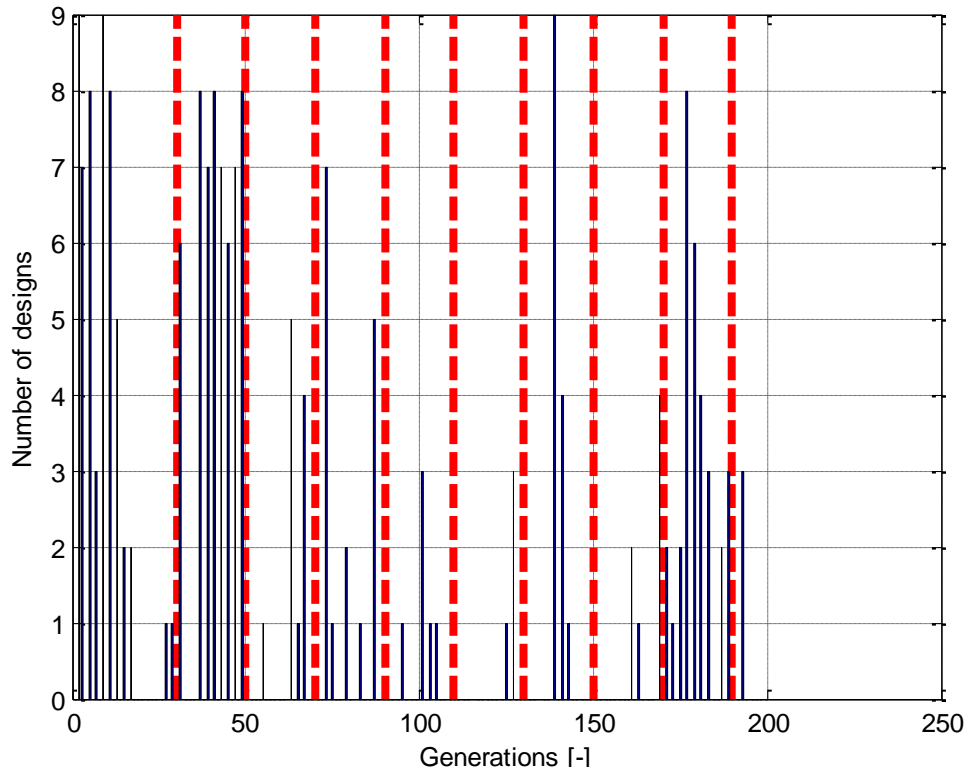


Figure 3-65. Number of designs of bay 1 that change from one generation to the next one when optimizing the weight and damage (200 generations and 9 individuals)

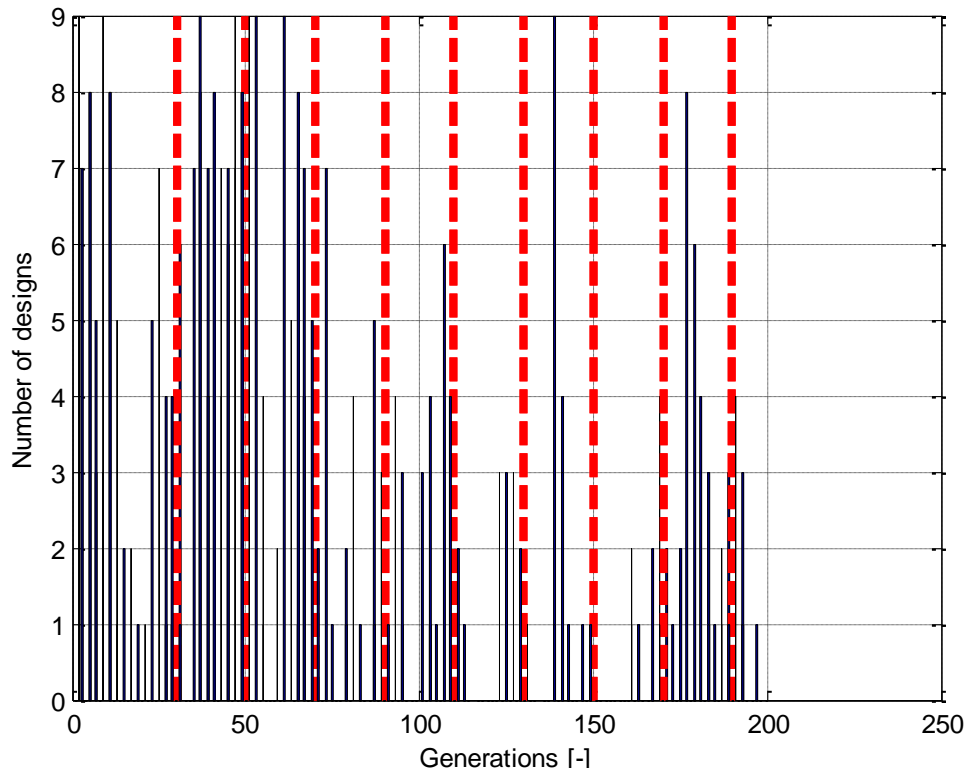


Figure 3-66. Number of designs of bay 2 that change from one generation to the next one when optimizing the weight and damage (200 generations and 9 individuals)

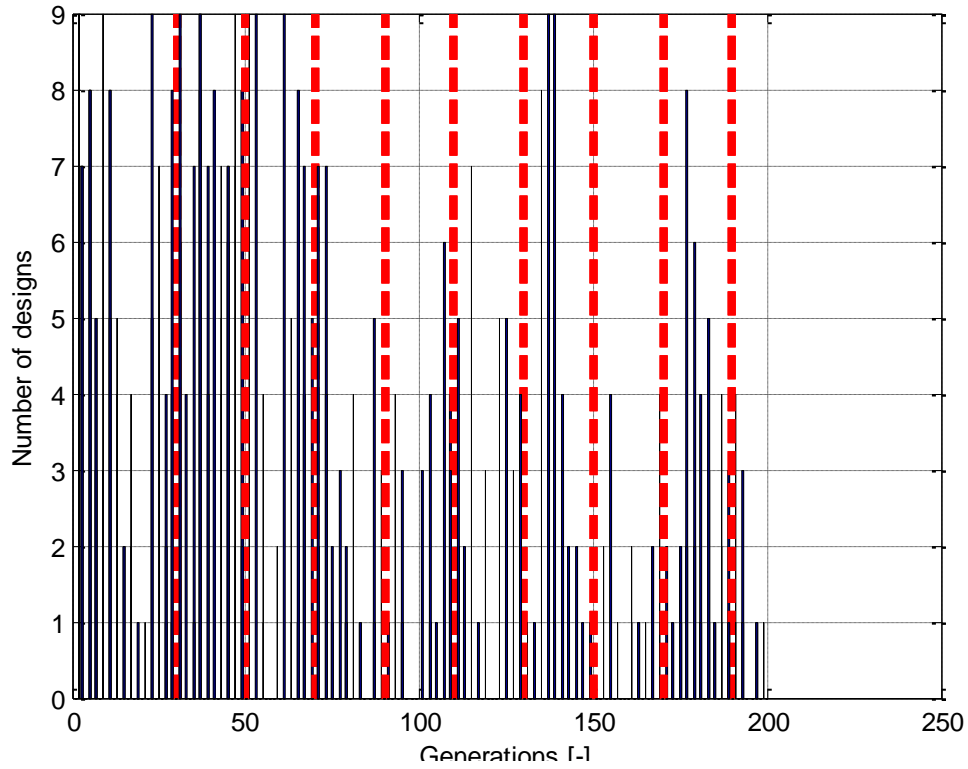


Figure 3-67. Number of designs of bay 3 that change from one generation to the next one when optimizing the weight and damage (200 generations and 9 individuals)

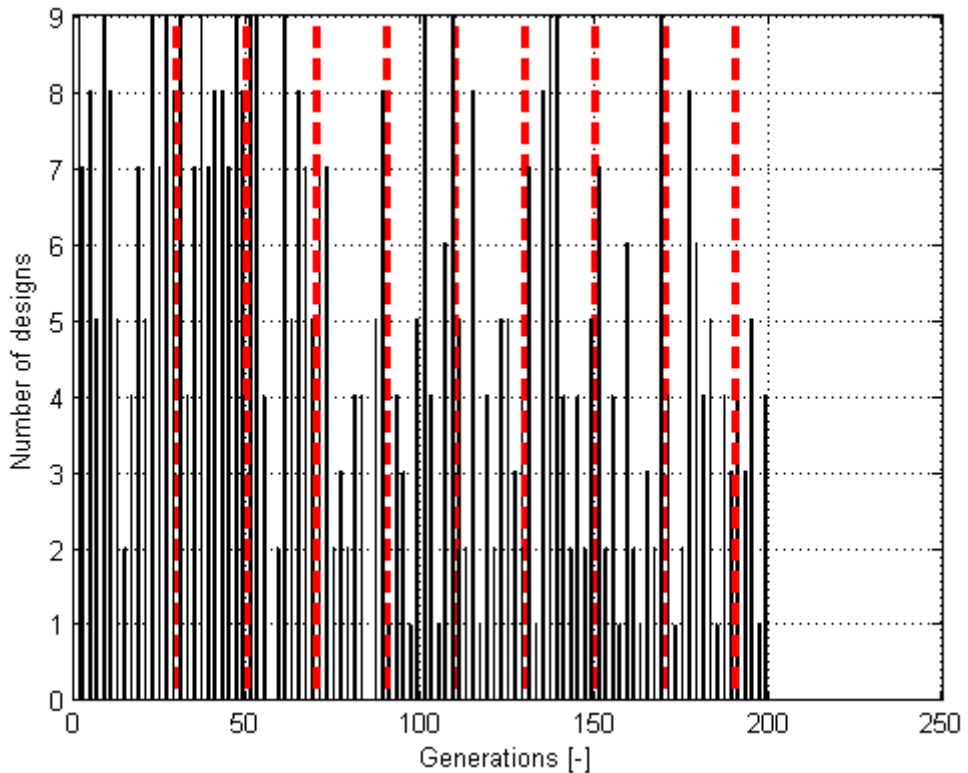


Figure 3-68. Number of designs of bay 4 that change from one generation to the next one when optimizing the weight and damage (200 generations and 9 individuals)

3.3.1.6 SCF parameters constraints

3.3.1.6.1 Gamma parameter

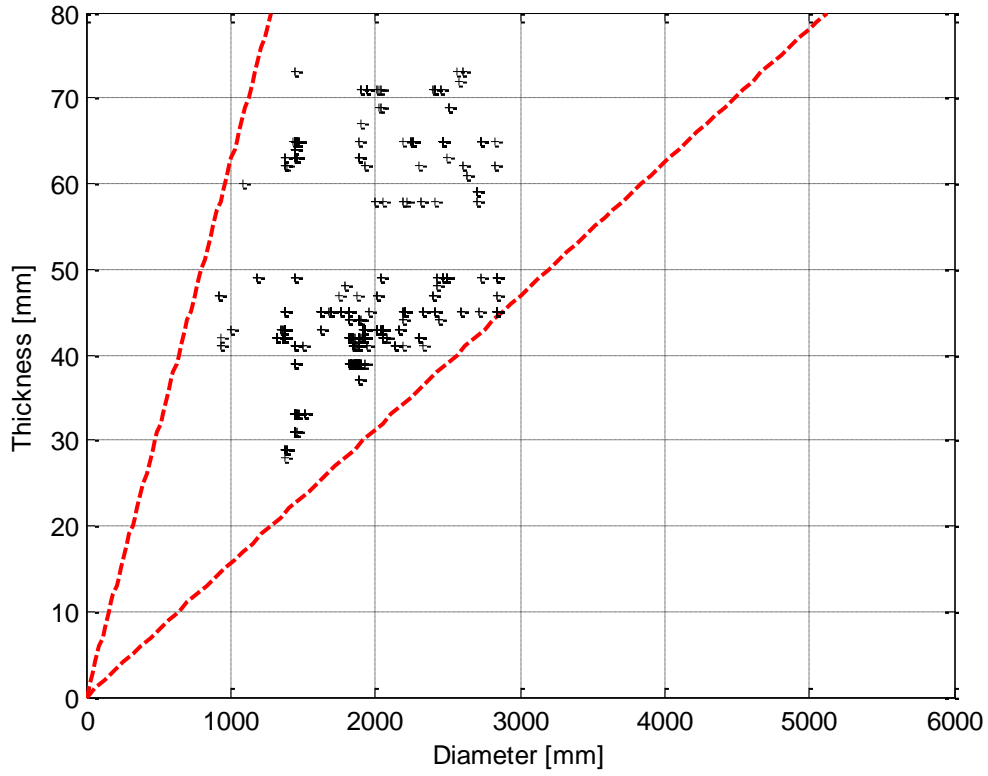


Figure 3-69. Gamma constriction for the legs when optimizing the weight and damage (200 generations and 9 individuals)

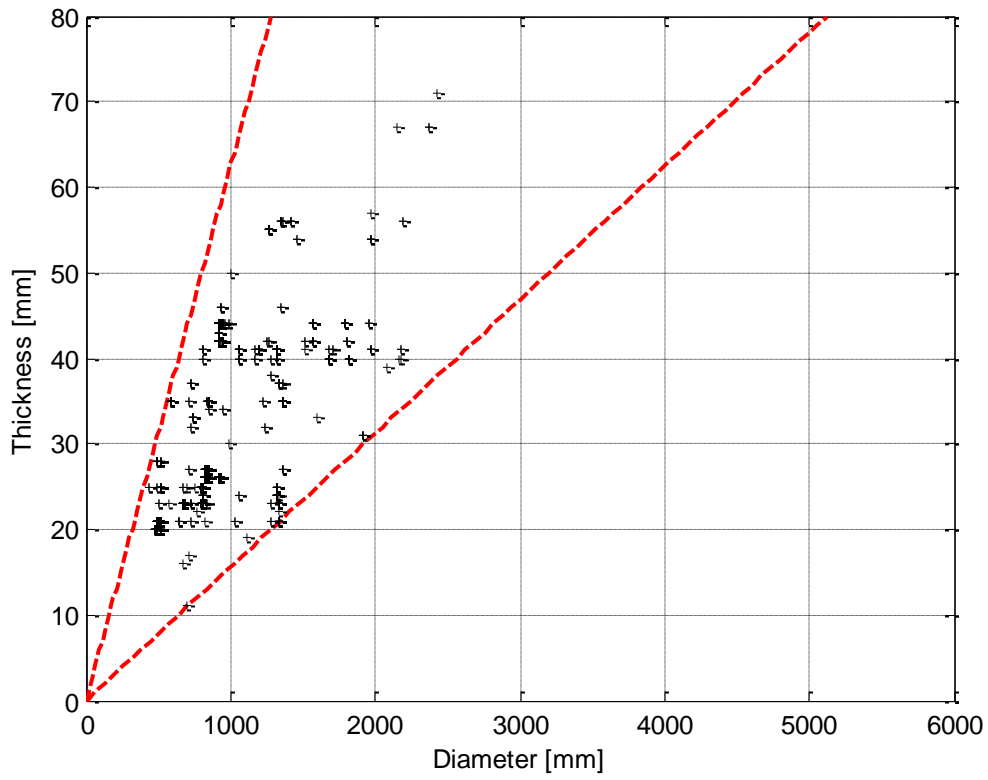


Figure 3-70. Gamma constriction for the braces when optimizing the weight and damage (200 generations and 9 individuals)

3.3.1.6.2 Beta parameter

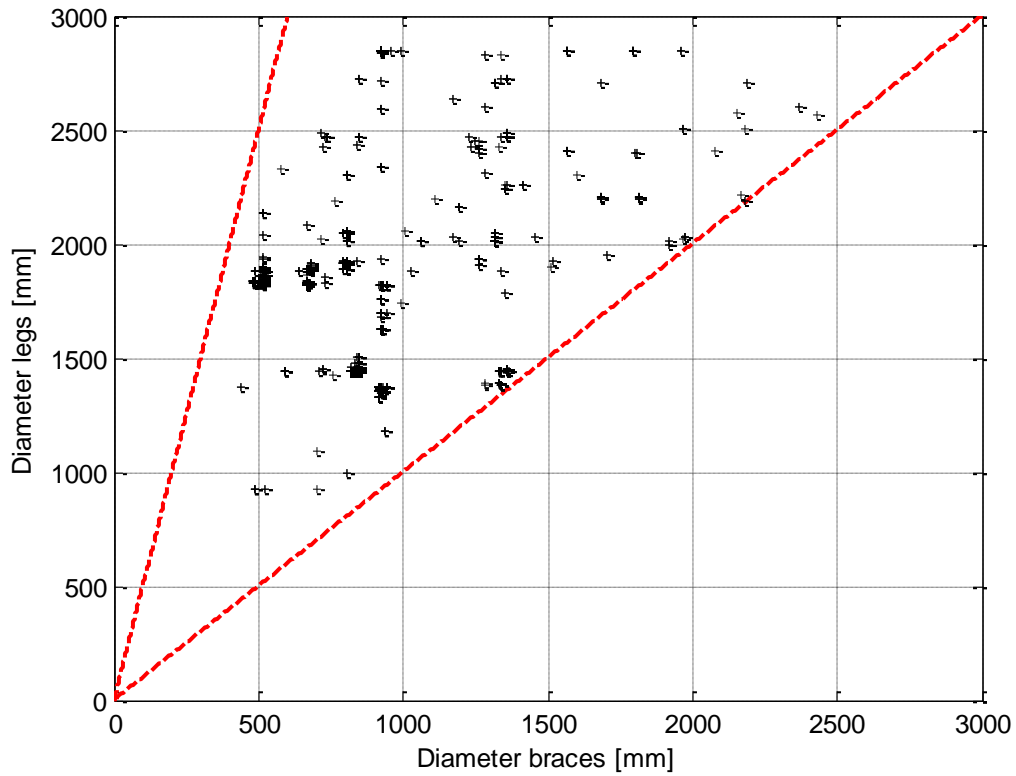


Figure 3-71. Beta constrain for legs and braces when optimizing the weight and damage (200 generations and 9 individuals)

3.3.1.6.3 Tau parameter

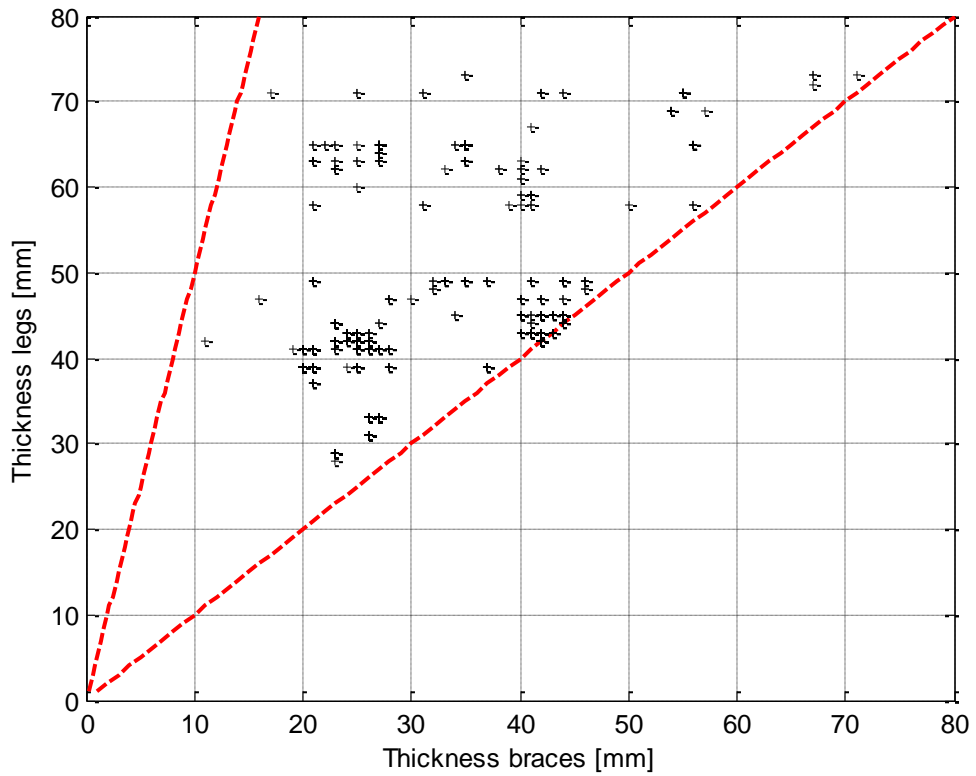


Figure 3-72. Tau constrain for legs and braces when optimizing the weight and damage (200 generations and 9 individuals)

3.3.2 Topology and geometry optimization

3.3.2.1 Weight range

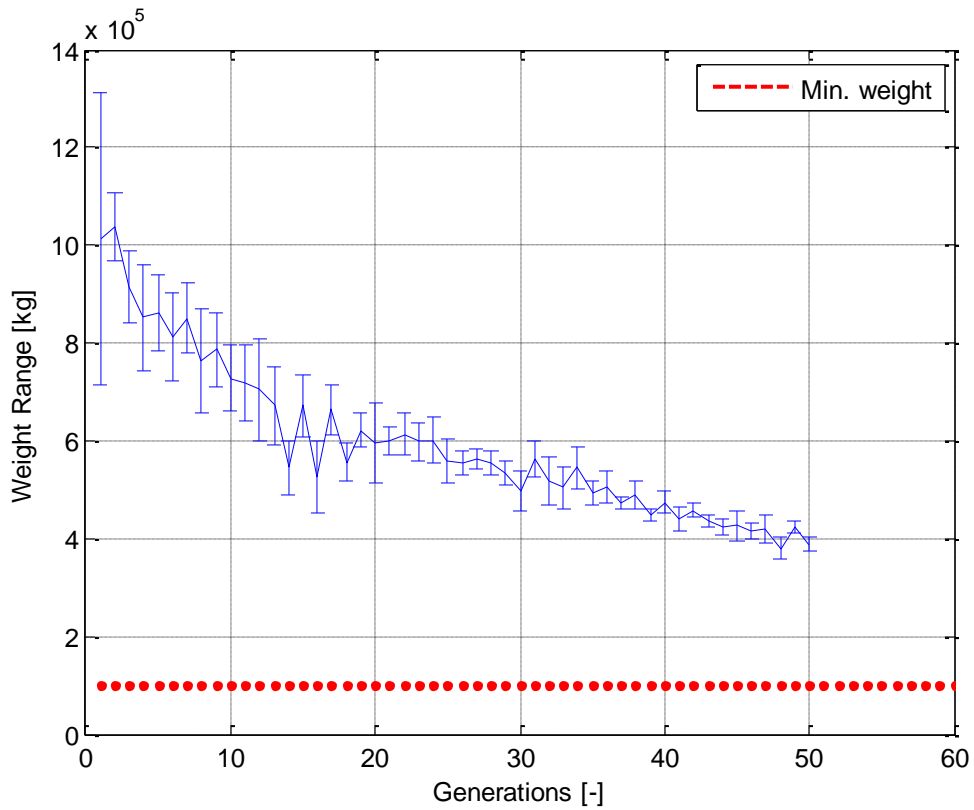


Figure 3-73. Weight of the structure [kg], when optimizing the weight and damage (50 generations and 10 individuals)

The lightest design of the last generation and the one that fits the best weight 373.350 kg and the heaviest 402.382 kg.

3.3.2.2 Diameter progress

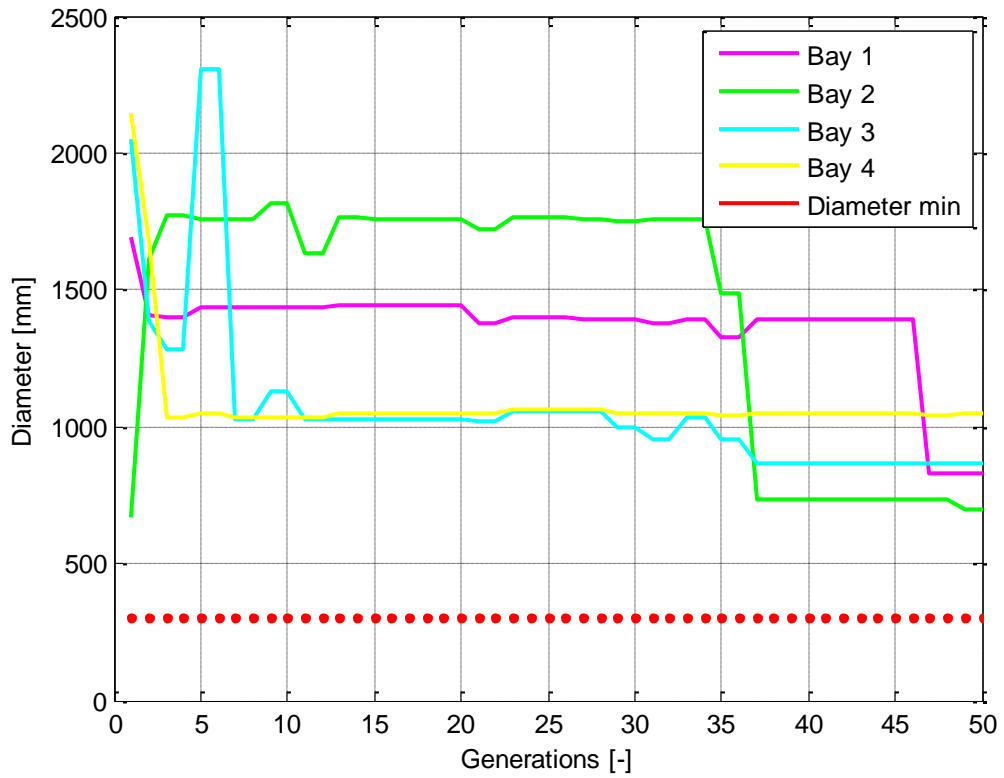


Figure 3-74. Evolution of the values of the diameter [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight and damage (50 generations and 10 individuals)

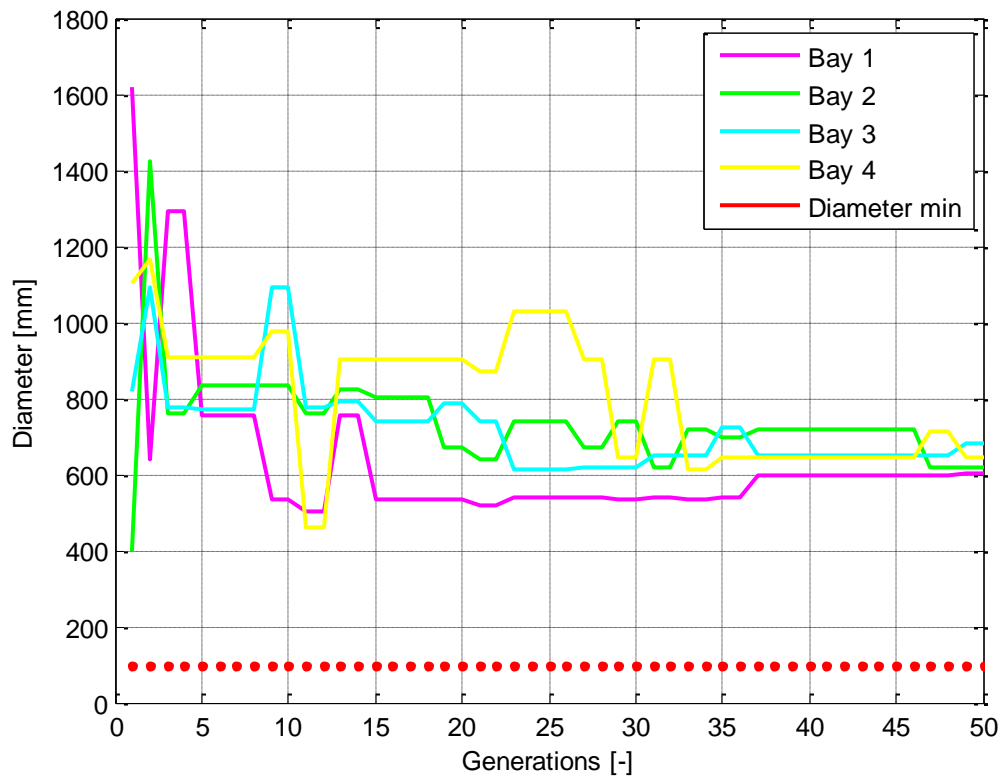


Figure 3-75. Evolution of the values of the diameter [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight and damage (50 generations and 10 individuals)

The exact values of diameter that the legs and braces take at the end of the simulation are presented in the table below:

	DIAMETER [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	830	699	867	1.051
Brace	602	618	683	647

Table 3-13. Values of diameter at the last generation when optimizing the weight, the damage, the topology and the geometry (50 generations and 10 designs)

3.3.2.3 Thickness progress

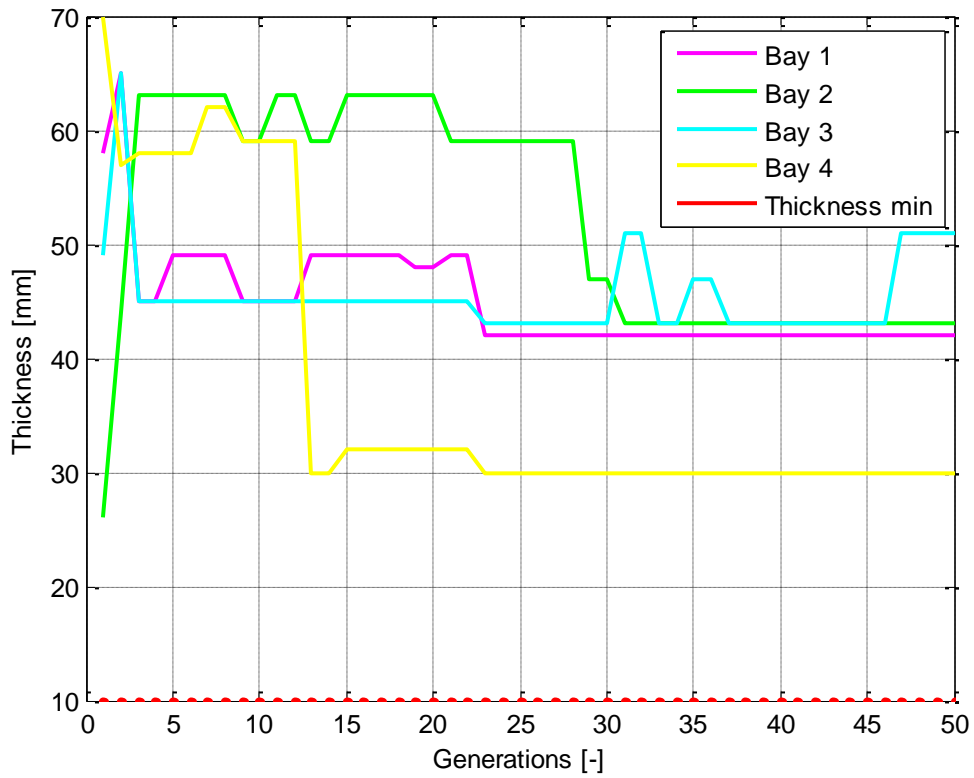


Figure 3-76. Evolution of the values of the thickness [mm] for the legs of every bay and minimum value that can be reach when optimizing the weight and damage (50 generations and 10 individuals)

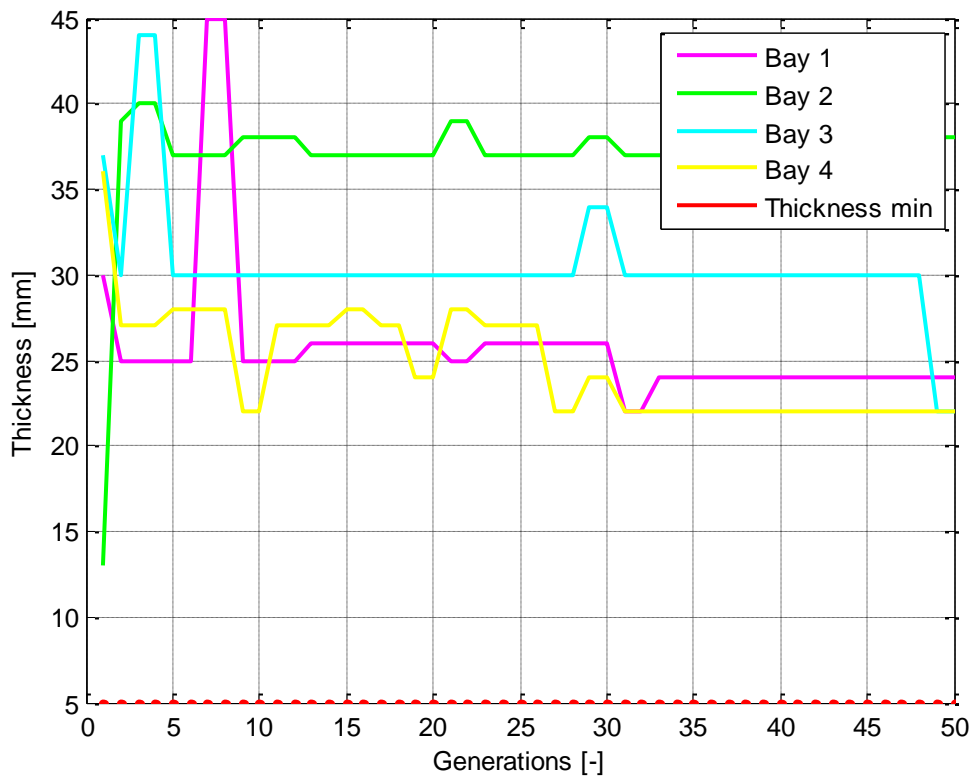


Figure 3-77. Evolution of the values of the thickness [mm] for the braces of every bay and minimum value that can be reach when optimizing the weight and damage (50 generations and 10 individuals)

The exact values of thickness that the legs and braces take at the end of the simulation are presented in the table below:

	THICKNESS [mm]			
	Bay 1	Bay 2	Bay 3	Bay 4
Leg	42	43	51	30
Brace	24	38	22	22

Table 3-14. Values of thickness at the last generation when optimizing the weight, the damage ,the topology and the geometry (50 generations and 10 designs)

3.3.2.4 Topology overview

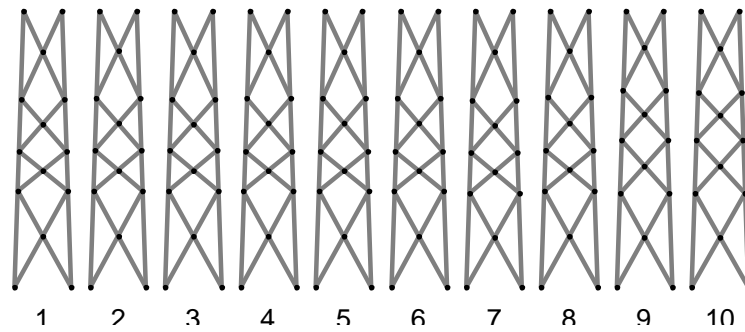


Figure 3-78. Location of the nodes of the 10 best designs when optimizing the weight and damage (50 generations and 10 individuals)

And the length of the bays of the best design is written in the table that follows:

LENGTH [mm]				
	Bay 1	Bay 2	Bay 3	Bay 4
Length	20.336	8.559	10.919	18.964

Table 3-15. Values of the length of the bays of the best design at the last generation when optimizing the weight, the damage, the topology and the geometry (50 generations and 10 designs)

3.3.2.5 Fitness

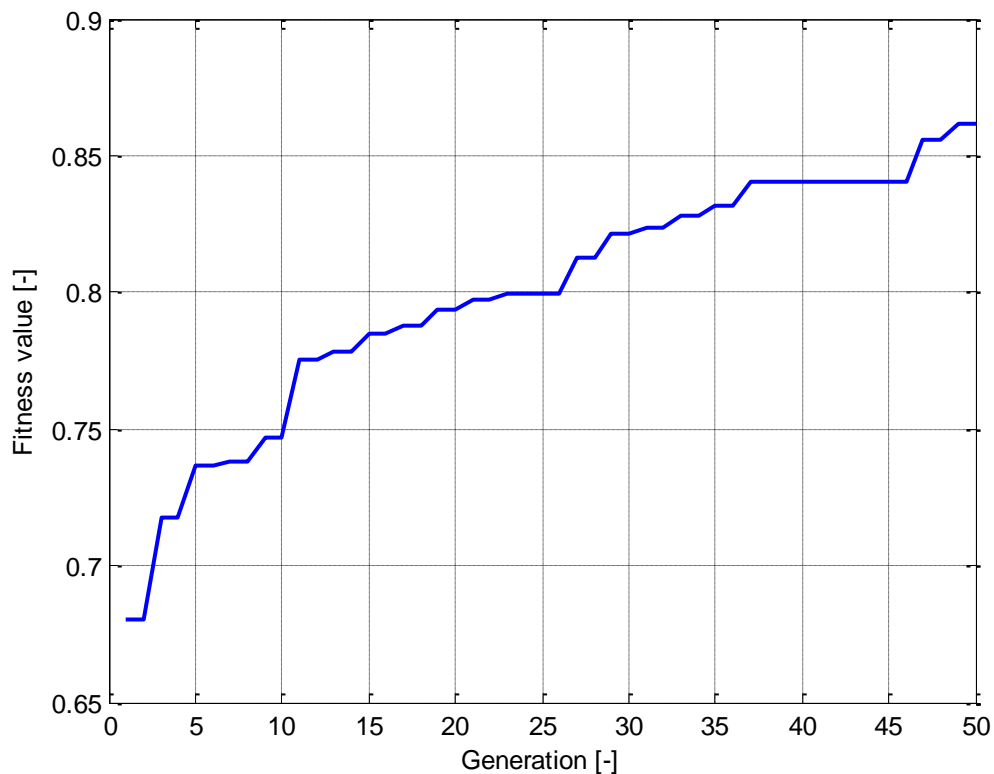


Figure 3-79. Improvement of the fitness through the generations when optimizing the weight and damage (50 generations and 10 individuals)

3.3.2.6 Number of designs that change

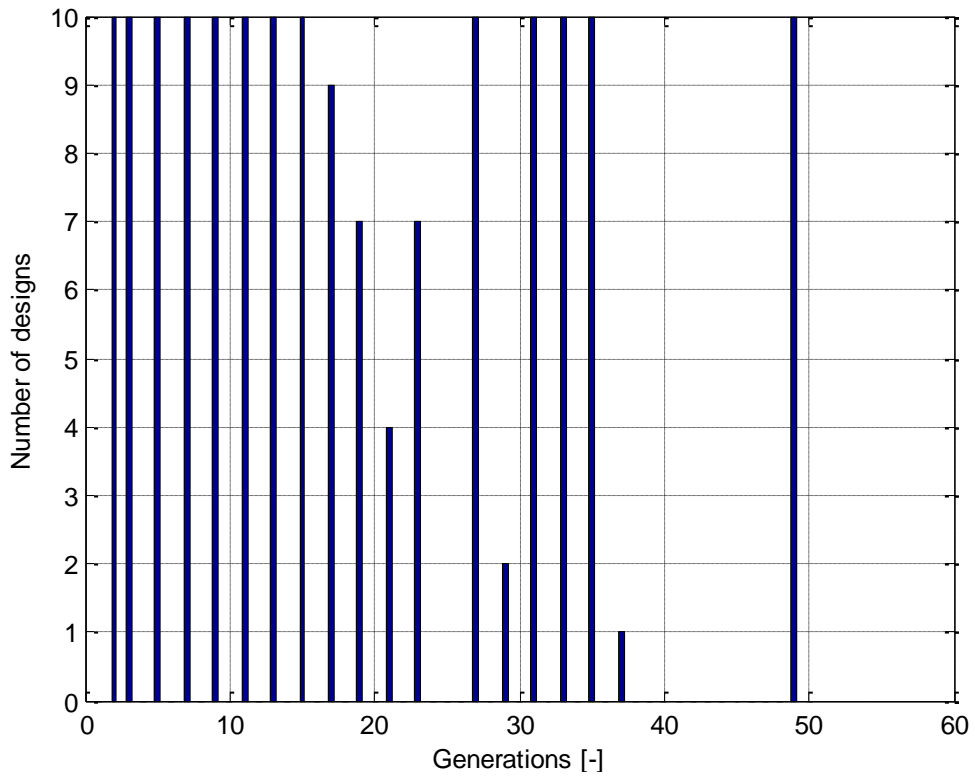


Figure 3-80. Number of designs of bay 1 that change from one generation to the next one when optimizing the weight and damage (50 generations and 10 individuals)

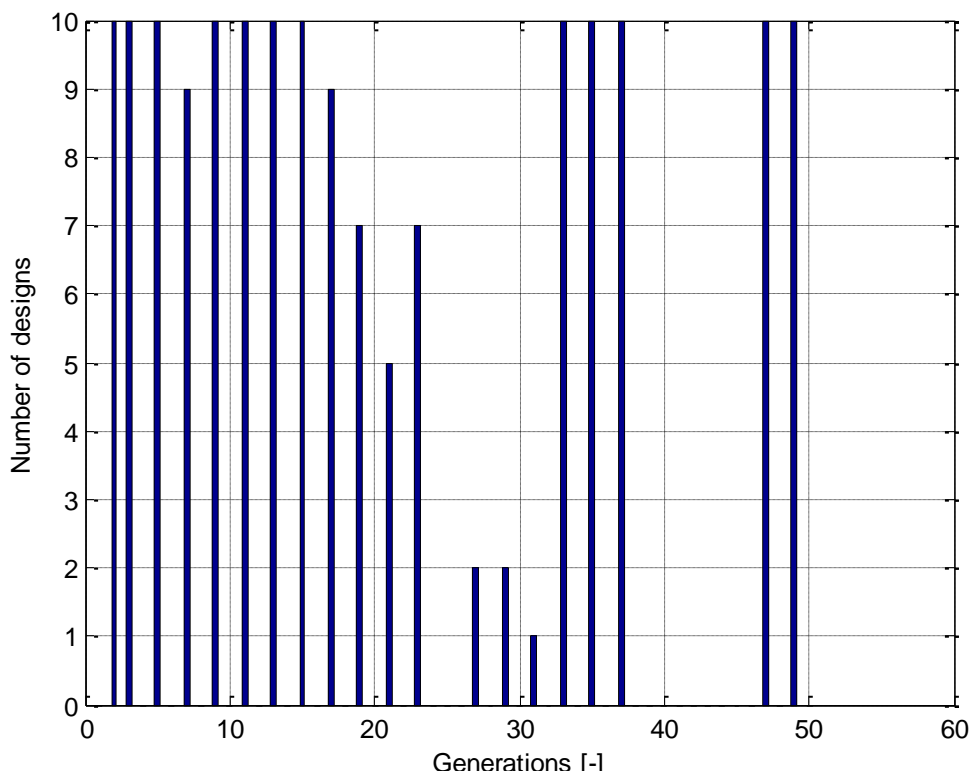


Figure 3-81. Number of designs of bay 2 that change from one generation to the next one when optimizing the weight and damage (50 generations and 10 individuals)

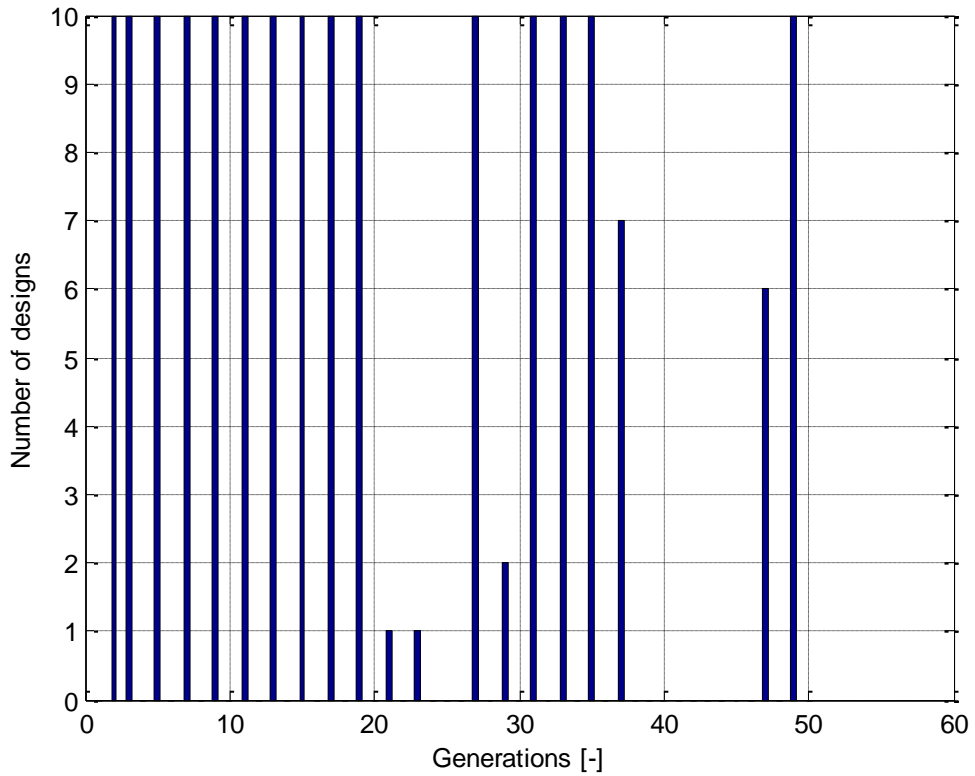


Figure 3-82. Number of designs of bay 3 that change from one generation to the next one when optimizing the weight and damage (50 generations and 10 individuals)

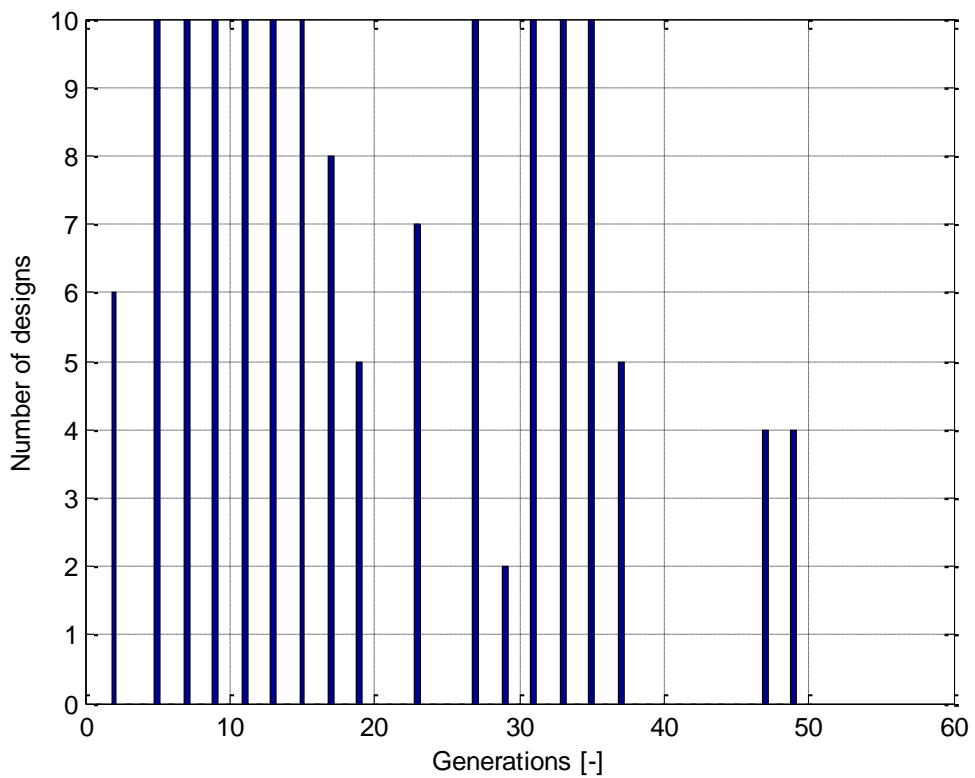


Figure 3-83. Number of designs of bay 4 that change from one generation to the next one when optimizing the weight and damage (50 generations and 10 individuals)

3.3.2.7 SCF parameters constrains

3.3.2.7.1 Gamma parameter

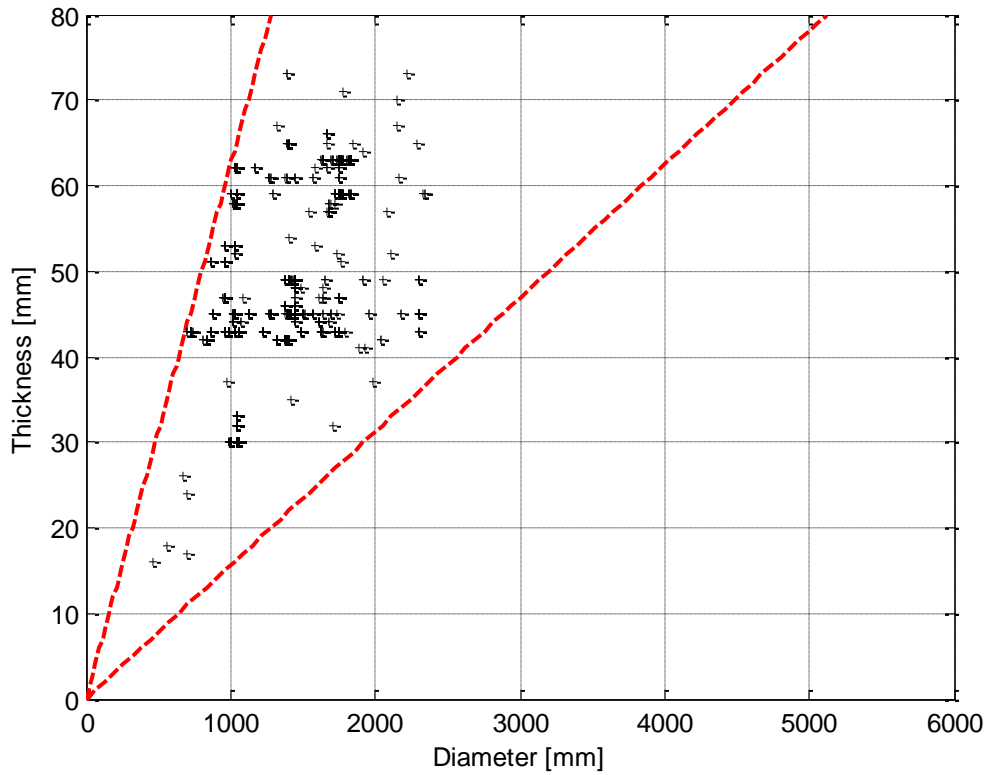


Figure 3-84. Gamma constriction for the legs when optimizing the weight and damage (50 generations and 10 individuals)

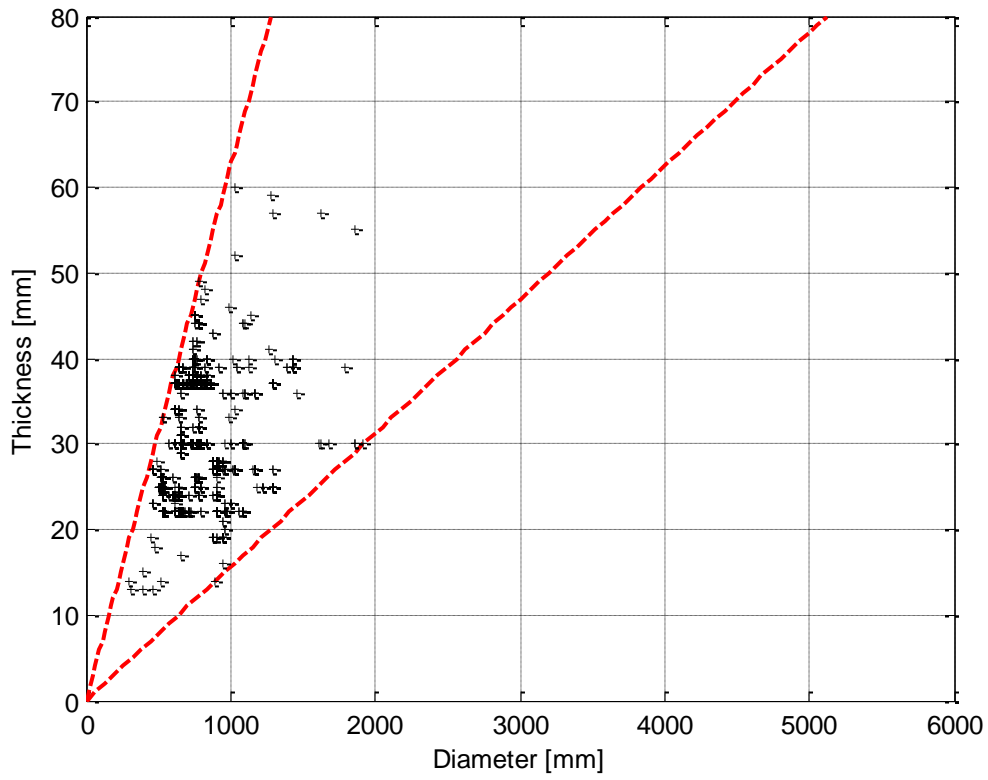


Figure 3-85. Gamma constriction for the braces when optimizing the weight and damage (50 generations and 10 individuals)

3.3.2.7.2 Beta parameter

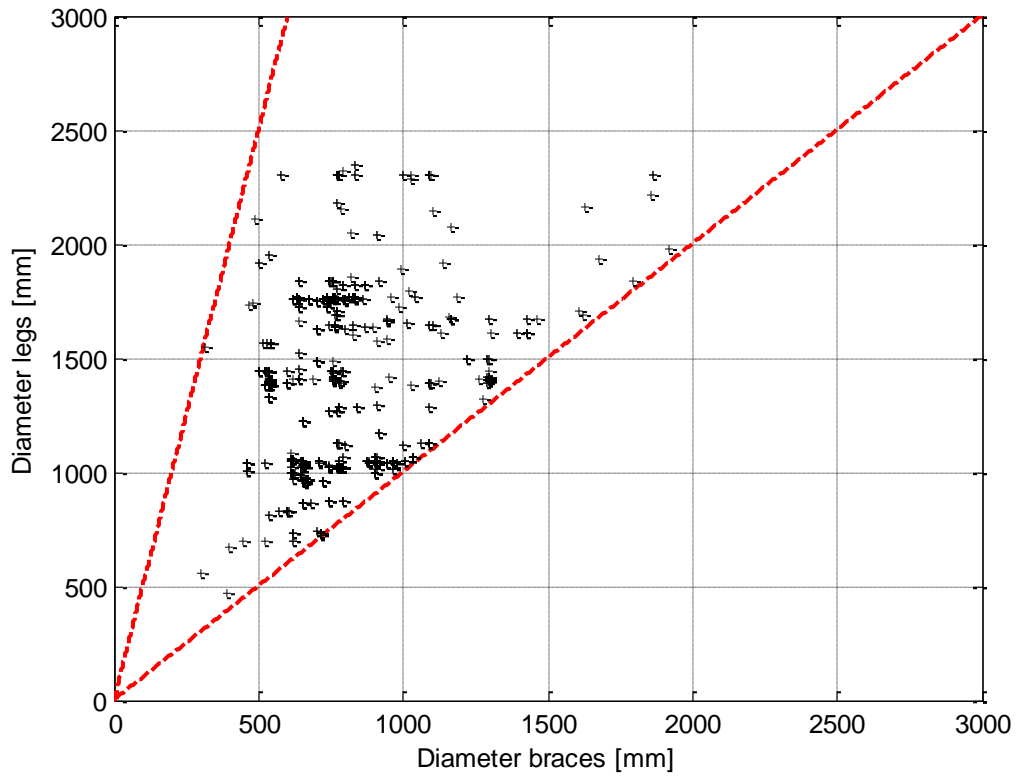


Figure 3-86. Beta constrain for legs and braces when optimizing the weight and damage (50 generations and 10 individuals)

3.3.2.7.3 Tau parameter

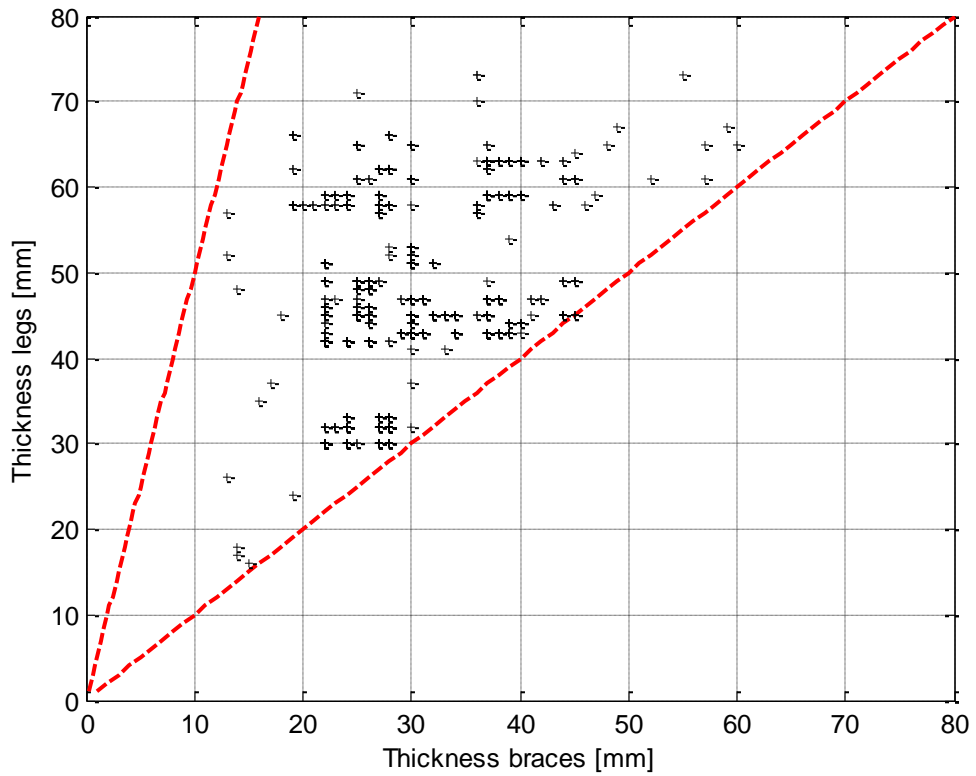


Figure 3-87. Tau constrain for legs and braces when optimizing the weight and damage (50 generations and 10 individuals)

4 Conclusion

The results presented above demonstrate that a genetic algorithm is a good tool to optimize a support jacket structure for offshore wind turbine. But even though all the simulations seem to find an optimum, not all of them are equally efficient. It must be clarified that the optima that have been found in the difference simulations are not a guarantee of an optimal design. To ensure that the design is a global optimum and not local the algorithm must be run several times [1].

In this thesis the simulations are limited by 30 s loadcase, hence, for a realistic application, a set of different loadcases and longer simulations would be needed in order to attain a more accurate analysis of the structural performance.

When changes on the topology are included similar characteristics of weight, diameter and thickness were found. Nevertheless, a combination of topology and geometry optimization seems to be more realistic since also the length of the beams affects the performance of the structure and the total weight.

The objective function based on the weight optimization will be discussed comparing the speed convergence first when only the geometry is changed and then when also the topology is implemented. In both cases the diameters of the legs are the ones that take longer to converge, so the rest of the variables will be used in the discussion. When implementing the topology optimization together with the geometry optimization an extra individual is included in the population ($N=10$). This is something that can help the speed convergence but still, the difference in the number of generations needed to achieve the same values is too high (one hundred) to conclude that the extra individual can compensate the faster convergence of the topology optimization.

The same holds true for the objective function based on the damage optimization. The geometry optimization is not faster than the topology and geometry optimization, but the number of generations is smaller, not only because significantly less iterations are needed on the geometry optimization, until it holds up, but also because this number of iterations is reduced for the topology and geometry optimization.

However this faster convergence troublesome when only the geometry is optimized as evidenced when the weight that has been calculated from both objective functions is compared. If the objective is to find the lightest design that survive over 20 years, then definitely damage optimization is not an option if only the geometry is changed because the best design is much heavier than the best design of the weight optimization.

Finally step-by-step objective function. Seems that is having more problems to find good designs. Lots of ups and downs can be seen on the progress of the diameter and thickness figures although similar values for the variables are found (similar to the rest of the objective functions results) but not as fast as the weight optimization when the locations of the nodes and the geometry are changed.

To conclude which will be the best design it would be necessary to run the algorithm for more generations until the fitness does not improve or increase the size of the population, which would be better since it increases the variability. But choosing the best design depends on what you want to minimize. If the costs are the main concern then a light design is the best. But maybe is a good idea to take the dimensions of the beams into consideration if they are very different from each other as it would be a serious inconvenience. Also, if the dimensions are not on the catalogue of the suppliers and need to be design especially for the project. In those cases it could be better to consider a heavier structure but with normalize dimensions as the pieces will be easier for the supplier to make them and will be available earlier at the construction site.

Now, if the figures with the number of new designs per generations are observed it is possible to conclude that the immigration is, definitively, introducing variability new designs are introduced in the reproduction every ten generations and consequently, in the following generations the number of new designs will increase. The mutation makes big changes in the string when the fitness is hold up so that the algorithm does not end in a local optimum. But sometimes it is not enough and the immigration is a very good way to help the mutation prevent the genetic algorithm from being hold up in a local optimum.

Regarding the SCF constraints it is difficult to determine whether one of them is more restricted than the other because all of them prevent a very light design to be analyzed. This is because if it does not fulfill the SCF requirements it will fail the fatigue analysis. Nevertheless, these constraints are more

restricted than the minimum values that have been established for the diameter and thickness because the design with the minimum characteristics does not fulfill the SCF requirements.

In conclusion, it has been demonstrated that a genetic algorithm can be used to optimize a support jacket for wind turbines structures and the properties of the optimize design are as follows:

- Between 800 and 900 mm for the diameter of the legs and from 400 to 500 mm for the diameter of the braces.
- Between 40 and 50 mm for the thickness of the legs and from 20 to 30 mm for the thickness of the braces.
- The length of the bays should be decreasing as the bay gets closer to the transition part.
- The weight will varies depending on the dimensions, but it should be around $4 \cdot 10^5$ kg.

5 References

- [1] Arora, J.S., 2012, *Introduction to optimum design*, Academic Press.
- [2] Ashuri, T., 2012, *Beyond classical upscaling: integrated aeroservoelastic design and optimization of large offshore wind turbines*, Ph.D. thesis, Delft University of Technology
- [3] Bazeos, N., Hatzigeorgiou, G.D., Hondros, I.D., Karamaneas, H., Karabalis, D.L., and Beskos, D.E., 2002, *Static, seismic and stability analyses of a prototype wind turbine steel tower*, Eng. Struct., 24, pp. 1014-1025.
- [4] Bäch T, Schütz M., 1996, *Intelligent mutation rate control in canonical genetic algorithms*. Foundations of Intelligent Systems, 1079, pp. 158-167.
- [5] Box GEP. Evolutionary operation, 1957, *A method for increasing industrial productivity*. Journal of the Royal Statistical Society, 6(2), pp. 81-101.
- [6] Böker, C., 2009, *Load simulation and local dynamics of support structures for offshore wind turbines*, Ph.D. thesis, Leibniz University Hannover.
- [7] Chipperfield A., Fleming P., Pohlheim H., Fonseca, C., *Genetic Algorithm Toolbox User's Guide for matlab*, Department of Automotic Control and Systems Engineering, University of Sheffield.
- [8] Choi, K.K., and Kim, N.H., 2005a, *Structural sensitivity analysis and optimization 1 – linear systems*, Springer.
- [9] Choi, K.K., and Kim, N.H., 2005b, *Structural sensitivity analysis and optimization 2 – nonlinear systems and applications*, Springer.
- [10] Christensen, P.W., and Klarbring, A., 2009, *An introduction to structural optimization*, Springer.
- [11] Coello Coello, Carlos A. Rudnick, Mitchael. Christiansen, Alan D., 1994, *Using genetic algorithms for optimal design of trusses*. Tulane University (New Orleans).
- [12] Darwin C., 1859, *On the Origin of Species*. Jhon Murray, Albemarle Street: London.
- [13] De Jong K., 1975, *Analysis of behaviour of a class of genetic adaptative systems*, Ph. D. thesis, University of Michigan, Ann Arbor.

- [14] Deb, Kalyanmoy. Gulati, Surendra, 2000, *Design of Truss-Structures for Minimum Weight using Genetic Algorithms*, Finite Elements in Analysis and Design. 37(5), pp. 447-465.
- [15] Det Norske Veritas, 2010, *Fatigue design of offshore wisteel structures*, Recommended Practice DNV-RP-C203, Det Norske Veritas, Høvik.
- [16] European Wind Energy Association, 2005, *Prioritising wind energy research – strategic research agenda of the wind energy sector*, European Wind Energy Association, Brussels.
- [17] Flemming W., 1882, *Zellsubstanz, Kern und Zelltheilung*. FCW Vogel: Leipzig.
- [18] Fogel LJ, Owens AJ, Walsh MJ., 1966, *Artificial intelligence through simulated evolution*. Wiley, New York.
- [19] Friedman GJ., 1959, *Digital simulation of an evolutionary process*, General Systems Yearbook, 4, pp. 171-184.
- [20] Gosavi, A., 2010, *Simulation-based optimization – parametric optimization techniques and reinforcement learning*, Kluwer Academic.
- [21] Grefenstette JJ. ,1986, *Optimization of control parameters for genetic algorithms*, IEEE Transactions on Systems, Man, and Cybernetics, 16(1), pp. 122-128.
- [22] Haftka, R.T., Gürdal, Z., 1991, *Elements of structural optimization*, Springer.
- [23] Harinath Babu Kamepalli, 2001, *The optimal basics for GAs*, IEEE Potentials, pp. 25-27.
- [24] Holland JH., 1975, *Adaptations in natural and artificial systems*, University of Michigan Press, Ann Arbor.
- [25] International Electrotechnical Commission, 2009, *Wind turbines – part 3: design requirements for offshore wind turbines*, International Standard IEC-61400:3, International Electrotechnical Commission, Geneva.
- [26] Jenkins, W.M., 1991, *Towards structural optimization via the genetic algorithm*, Comp. Struct., 40(5), pp. 1321-1327.

- [27] Kang, B.S., Park, G.J., and Arora, J.S., 2006, *A review of optimization of structures subjected to transient loads*, Struct. Multidisc. Optim., 31, pp. 81-95.
- [28] Koza J., 1992, *Genetic Programming*. MIT Press: Cambridge.
- [29] Kühn, M., 2001, *Dynamics and design optimisation of offshore wind energy conversion systems*, Ph.D. thesis, Delft University of Technology.
- [30] Lavassas, I., Nikolaidis, G., Zervas, P., Efthimiou, E., Doudoumis, I.N., and Baniotopoulos, C.C., 2003, *Analysis and design of the prototype of a steel 1-MW wind turbine tower*, Eng. Struct., 25, pp. 1097-1106.
- [31] Long, H., Moe, G., Fischer, T., 2012, *Lattice towers for bottom-fixed offshore wind turbines in the ultimate limit state: variation of some geometric parameters*, J. Offshore Mech. Arct. Eng., 134, pp. 021202:1-13.
- [32] Long, H., and Moe, G., 2012, *Preliminary design of bottom-fixed lattice offshore wind turbine towers in the fatigue limit state by the frequency domain method*, J. Offshore Mech. Arct. Eng., 134, pp. 031902:1-10.
- [33] Mendel G., 1865, *Versuche uber Pflanzen-Hybriden*. *Verh. Naturforsch. Ver. Brunn*, 4, pp. 3-47.
- [34] Quarton, D.C., 1998, *The evolution of wind turbine design analysis – a twenty-year progress review*, Wind Energy, 1, pp. 5-24.
- [35] Rechenberg I., 1965, *Cybernetic Solution Path of an Experimental Problem*. Ministry of Aviation. Royal Aircraft Establishment, UK.
- [36] Rothlauf, Franz., 2006. *Representations for Genetic and Evolutionary Algorithms*, Berlin: Springer.
- [37] Salzmann, D.J.C., and Van der Tempel, J., 2005, *Aerodynamic damping in the design of support structures for offshore wind turbines*, Proc. Copenhagen Offshore Conf., Copenhagen.
- [38] Schaffer J, Caruana R, Eshelman L, Das R., 1989, *A study of control parameters affecting online performance of genetic algorithms for function optimization*. Proceedings of the 3rd International Conference on Genetic Algorithms, pp. 51-60.

- [39] Schaumann, P., Böker, C., Bechtel, A., and Lochte-Holtgreven, S., 2011, “*Support structures of wind energy converters,*” Environmental wind engineering and design of wind energy structure, C. Baniotopoulos, C. Borri, and T. Stathopoulos, eds., Springer.
- [40] Seidel, M., Ostermann, F., Curvers, A.P.W.M., Kühn, M., Kaufer, D., and Böker, C., 2009, *Validation of offshore load simulations using measurement data from the DOWNVInD project*, Proc. European Offshore Wind Conf., Stockholm.
- [41] Sivanandam S. N. Deepa S. N., 2008. *Introduction to Genetic Algorithms*. Berlin: Springer.
- [42] Soh, Che-Kiong. Yang, Jiaping, 1998. *Optimal Layout of Bridge Trusses by Genetic Algorithms*. Computer-Aided Civil and Infrastructure Engineering, 13(4), pp. 247-254.
- [43] The Carbon Trust, 2008, *Offshore wind power: big challenge, big opportunity – maximizing the environmental, economic and security benefits*, Report CTC743, The Carbon Trust, London.
- [44] The Crown Estate, 2012, *Offshore wind cost reduction – pathways study*, Report, The Crown Estate, London.
- [45] Thiry, A., Bair, F., Buldgen, L., Raboni, G., and Rigo, P., 2011, *Optimization of monopile offshore wind structures*, Advances in Marine Structures, C.G. Soares and W. Fricke, eds., Taylor & Francis, pp. 633-642.
- [46] Torcinaro, M., Petrini, F., and Arangio, S., 2010, *Structural offshore wind turbines optimization*, Proc. Eartj Space 2010, Honolulu, HI, pp. 2130-2142.
- [47] Vemula, N. K., DeVries, W., Fischer, T., Cordle, A., and Schmidt, B., 2010, *Design solution for the upwind reference offshore support structure*. Upwind deliverable D4.2.6 (WP4: Offshore Foundations and Support Structures), Technical report, Ramøll Wind energy.
- [48] Victoria, M., 2006, *Optimización de Forma y Topología con Malla Fija y Algoritmos Genéticos*, Ph.D. thesis, University of Cartagena, Spain.
- [49] Vorpahl, F., Popko, W., Kaufer, D., 2013, *Description of a basic model of the “UpWind reference jacket” for code comparison in the OC4 project under IEA*

Wind Annex 30, Fraunhofer Institute for Wind Energy and Energy System Technology (IWES).

- [50] Vorpahl, F., Schwarze, H., Fischer, T., Seidel, M., and Jonkman, J., 2013, *Offshore wind turbine environmen, loads, simulation, and design*, WIREs Energy Environ., 2, pp. 548-570.
- [51] Watson JD, Crick, FHC., 1953, *Molecular Structure of Nucleic Acids*. Nature, 171, pp. 737 738.
- [52] Wen Xian Yang, 2004, *An improved genetic algorithm adopting immigration operator*, Journal Intelligente Data Analisis, 8(4), pp. 385-401.
- [53] Woon SY, Tong L, Querin OM, Steven GP., 2003, *Knowledge-based algoritms in fixed-grid GA shape optimization*, International Journal for Numerical Methods in Engineering, 58, pp. 643-660.
- [54] Yoshida, S., 2006, *Wind turbine tower optimization method using a genetic algorithm*, Wind Eng., 30, pp. 453-470.
- [55] Zwick, D., Muskulus, M., and Moe, G., 2012, *Iterative optimization approach for the design of full-height lattice towers for offshore wind turbines*, Energy Procedia, 24, pp. 297-304.

6 Appendix A. Fatigue Calculations

To estimate the fatigue damage of the structure we have carried out a fatigue analysis according to the **Recommended Practice DNV-RP-C203** [15] which is based on "fatigue tests and fracture mechanics". All points which are "potentially a source of fatigue cracking", call hot spots, will be evaluated by a fatigue method based on S-N data, "determined by fatigue testing". In this case, these critical points are located at the connection of the different beams (braces and chords) of the jacket. The stress that is generate on the surface of the hot spot is call hot spot stress and together with the T - curve is used to estimate the fatigue life of the structure ("number of stress cycles at a particular magnitude require to cause fatigue failure of the component").

"The fatigue life may be calculated based on the S-N fatigue approach under the assumption of linear cumulative damage (Palmgren-Miner rule)":

$$D = \sum_{i=1}^k \frac{n_i}{N_i}$$

where

D = accumulated fatigue damage

n_i = number of stress cycles in stress block i

N_i = number of cycles to failure at constant stress range $\Delta\sigma_i$

k = number of blocks. Shouldn't be under 20

The S-N curves are "obtained from fatigue tests", as mentioned before, "and are associated with a 97.7% probability of survival". The equation that defines an S-N curve is:

$$\log N = \log \bar{a} - m \cdot \log \Delta\sigma$$

where

N = predicted number of cycles to failure for stress range $\Delta\sigma$

$\Delta\sigma$ = stress range with unit MPa

m = negative inverse slope of S-N curve

$\log \bar{a}$ = intercept of log N-axis by S-N curve

These parameters will take different values depending on the joints we are analyzing, meaning that there will be varied S-N curves. For tubular joints we have to use:

S-N curve	N		N > 10 ⁶ cycles
	≤ 10 ⁶ cycles		
	m_1	$\log \bar{a}_1$	$\log \bar{a}_2$ $m_2 = 5.0$
T	3.0	11.764	15.606

Table 6-1. T-curve in seawater [15]

Once we know which values we have to use for the parameters that define an S-N curve we will focus on how to calculate the stress cycles and the constant stress ranges.

To calculate the constant stress range $\Delta\sigma$ of each block, which will define the stress distribution at each hot spot, we use the rainflow counting. The input for that procedure of stress range counting is a time series of hot spot stresses.

Since bending moments (in-plane and out of plane) are acting together with the axial load the stress should be calculated as a "summation of the single stress components from axial, in-plane and out of plane action" at the crown and the saddle of both, brace and chord, consequently we will have eight stress values at one single joint:

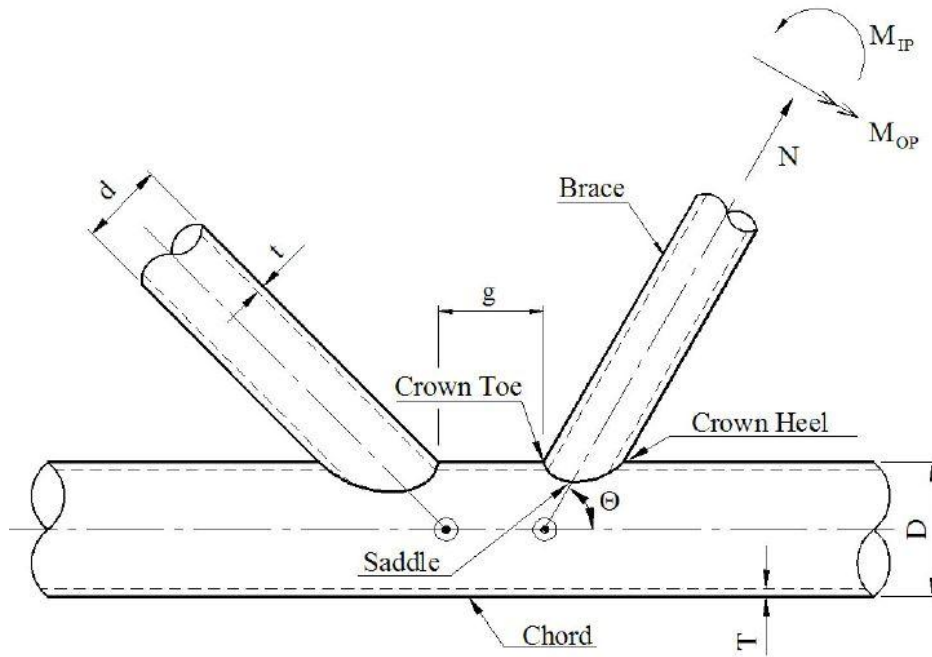


Figure 6-1. Geometrical definitions for tubular joints [15]

These stresses are given by:

$$\sigma_1 = SCF_{AC} \cdot \sigma_x + SCF_{MIP} \cdot \sigma_{my} \quad (\text{Crow Heel})$$

$$\sigma_3 = SCF_{AS} \cdot \sigma_x - SCF_{MOP} \cdot \sigma_{mz} \quad (\text{Saddle})$$

$$\sigma_5 = SCF_{AC} \cdot \sigma_x - SCF_{MIP} \cdot \sigma_{my} \quad (\text{Crow Toe})$$

$$\sigma_7 = SCF_{AS} \cdot \sigma_x + SCF_{MOP} \cdot \sigma_{mz} \quad (\text{Saddle})$$

where

σ_x = maximum nominal stress due to axial load

σ_{my} = maximum nominal stress due to in-plane moment

σ_{mz} = maximum nominal stress due to out of plane moment

SCF_{AC} = stress concentration factor at the crow for axial load

SCF_{AS} = stress concentration factor at the saddle for the axial load

SCF_{MIP} = stress concentration factor for in-plane moment

SCF_{MOP} = stress concentration factor for out of plane moment

When calculating the SCFs a definition of some geometric parameters is needed for the three types of tubular joints.

For *Y-joints* and *X-joints*:

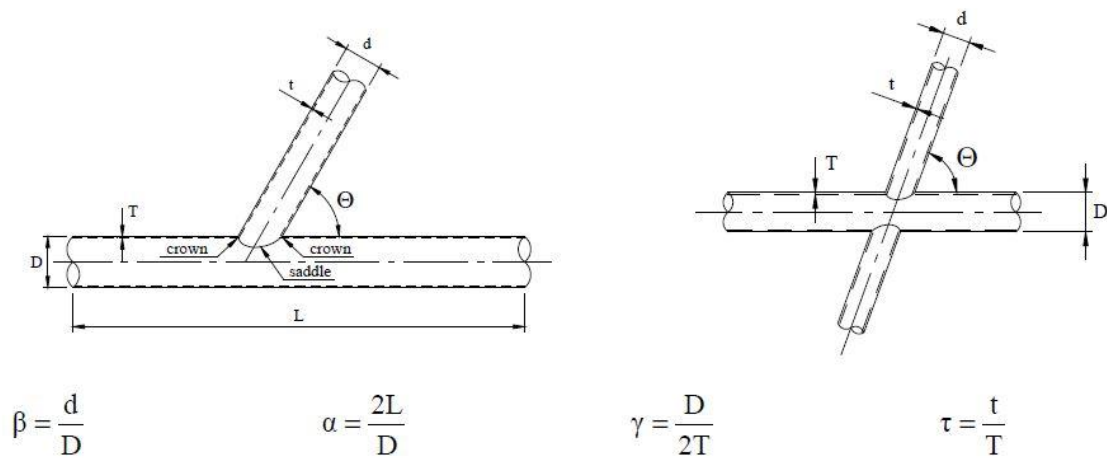


Figure 6-2. Definition of geometric parameters for Y-joints and X-joints [15]

and for *K-joints*:

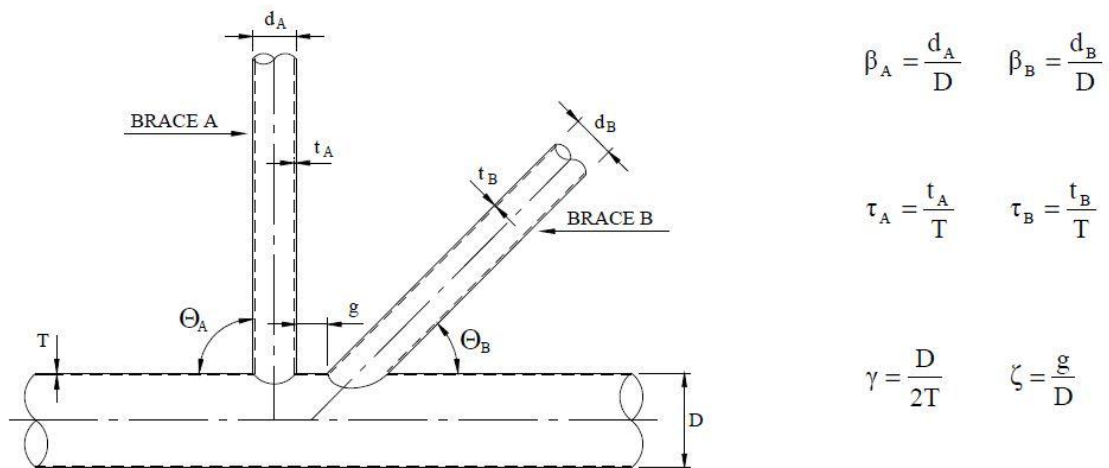


Figure 6-3. Definition of geometric parameters for K-joints [15]

All geometric parameters will have to fulfill the following restrictions:

$$0.2 \leq \beta \leq 1.0$$

$$0.2 \leq \tau \leq 1.0$$

$$8 \leq \gamma \leq 32$$

$$4 \leq \alpha \leq 40$$

$$20^\circ \leq \theta \leq 90^\circ$$

$$\frac{-0.6 \cdot \beta}{\sin \theta} \leq \zeta \leq 1.0$$

if the joint in consideration exceed at least one of restrictions above, new diameter and thickness will be chosen.

In addition there are four Short chord correction factors that will be apply when < 12 :

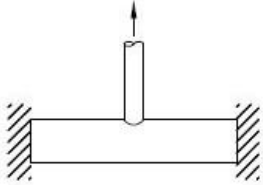
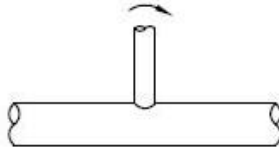
$$F1 = 1 - (0,83 \cdot \beta - 0,56 \cdot \beta^2 - 0,02) \cdot \gamma^{0,23} \cdot e^{(-0,21 \cdot \gamma^{-1,16} \cdot \alpha^{2,5})}$$

$$F2 = 1 - (1,43 \cdot \beta - 0,97 \cdot \beta^2 - 0,03) \cdot \gamma^{0,04} \cdot e^{(-0,71 \cdot \gamma^{-1,38} \cdot \alpha^{2,5})}$$

$$F3 = 1 - 0,55 \cdot \beta^{1,8} \cdot \gamma^{0,16} \cdot e^{(-0,49 \cdot \gamma^{-0,89} \cdot \alpha^{1,8})}$$

$$F4 = 1 - 1,07 \cdot \beta^{1,88} \cdot e^{-0,16 \cdot \gamma^{-1,06} \cdot \alpha^{2,4}}$$

Below the equations to calculate de SCFs are shown:

SIMPLE TUBULAR Y JOINTS		
LOAD TYPE	SCF equations	Short chord correction
<p>Axial load</p> 	<p>Chord saddle</p> $1.45 \cdot \beta \cdot \tau^{0.85} \cdot \gamma^{(1-0.68\cdot\beta)} \cdot (\sin \theta)^{0.7}$	F1
	<p>Chord crown</p> $\gamma^{0.2} \cdot \tau \cdot (2.65 + 5 \cdot (\beta - 0.65)^2) + \tau \cdot \beta \cdot (0.25 \cdot \alpha - 3) \cdot \sin \theta$	None
	<p>Brace saddle</p> $1. + \gamma \cdot \tau^{0.52} \cdot \alpha^{0.1} \cdot (0.187 - 1.25 \cdot \beta^{1.1} \cdot (\beta - 0.96)) \cdot (\sin \theta)^{(2.7-0.01\cdot\alpha)}$	F1
	<p>Brace crown</p> $3 + \gamma^{1.2} \cdot (0.12 \cdot e^{-4\cdot\beta} + 0.011 \cdot \beta^2 - 0.045) + \beta \cdot \tau \cdot (0.1 \cdot \alpha - 1.2)$	None
<p>In-plane bending</p> 	<p>Chord crown</p> $1.45 \cdot \beta \cdot \tau^{0.85} \cdot \gamma^{(1-0.68\cdot\beta)} \cdot (\sin \theta)^{0.7}$	None
	<p>Brace crown</p> $1 + 0.65 \cdot \beta \cdot \tau^{0.4} \cdot \gamma^{(1.09-0.77\cdot\beta)} \cdot (\sin \theta)^{(0.06\cdot\gamma-1.16)}$	None
<p>Out-of-plane</p>	<p>Chord saddle</p> $\gamma \cdot \tau \cdot \beta \cdot (1.7 - 1.05 \cdot \beta^3) \cdot (\sin \theta)^{1.6}$	F3

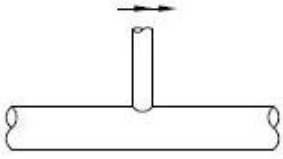
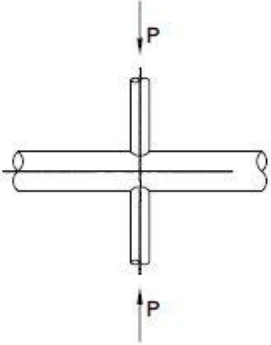
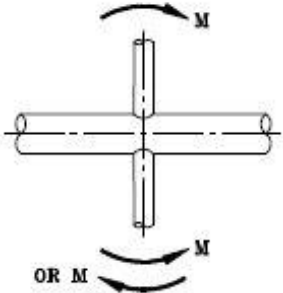
	<p>Brace saddle</p> $\tau^{-0.54} \cdot \gamma^{-0.05} \cdot (0.99 - 0.47 \cdot \beta + 0.08 \cdot \beta^4 \cdot \gamma \cdot \tau \cdot \beta \cdot 1.7 - 1.05 \cdot \beta^3 \cdot \sin \theta) 1.6$	<p>F3</p>
---	---	-----------

Table 6-2. Stress concentration factors for simple Y-joints [15]

SIMPLE TUBULAR X JOINTS	
LOAD TYPE	SCF equations
<p style="text-align: center;">Axial load</p> 	<p>Chord saddle</p> $3.7 \cdot \gamma \cdot \tau \cdot \beta \cdot (1.10 - \beta^{1.8}) \cdot (\sin \theta)^{1.7}$ <p>Chord crown</p> $\gamma^{0.2} \cdot \tau \cdot (2.65 + 5 \cdot (\beta - 0.65)^2) - 3 \cdot \tau \cdot \beta \cdot \sin \theta$ <p>Brace saddle</p> $1 + 1.9 \cdot \gamma \cdot \tau^{0.5} \cdot \beta^{0.9} \cdot (1.09 - \beta^{1.7}) \cdot (\sin \theta)^{2.5}$ <p>Brace crown</p> $3 + \gamma^{1.2} \cdot (0.12 \cdot e^{-4 \cdot \beta} + 0.011 \cdot \beta^2 - 0.045)$
<p style="text-align: center;">In-plane bending</p> 	<p>Chord crown</p> $1.45 \cdot \beta \cdot \tau^{0.85} \cdot \gamma^{(1-0.68 \cdot \beta)} \cdot (\sin \theta)^{0.7}$ <p>Brace crown</p> $1 + 0.65 \cdot \beta \cdot \tau^{0.4} \cdot \gamma^{(1.09-0.77 \cdot \beta)} \cdot (\sin \theta)^{(0.06 \cdot \gamma - 1.16)}$

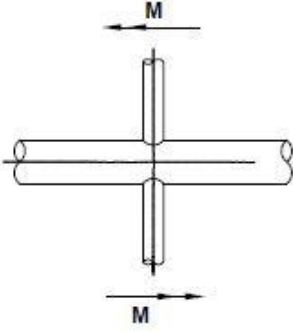
<p>Out of plane bending</p> 	<p>Chord saddle</p> $\gamma \cdot \tau \cdot \beta \cdot (1.561.34 \cdot \beta^4) \cdot (\sin \theta)^{1.6}$ <p>Brace saddle</p> $\tau^{-0.54} \cdot \gamma^{-0.005} \cdot (0.99 - 0.47 \cdot \beta + 0.08 \cdot \beta^4) \cdot \gamma \cdot \tau \cdot \beta \cdot (1.561.34 \cdot \beta^4) \cdot (\sin \theta)^{1.6}$
---	--

Table 6-3. Stress concentration factors for simple X-joints [15]

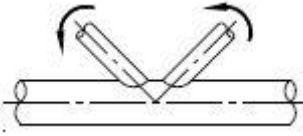
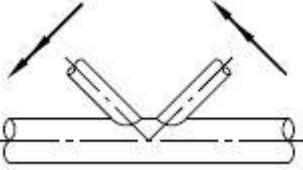
<p>In-plane bending</p> 	<p>Chord crown</p> $1.45 \cdot \beta \cdot \tau^{0.85} \cdot \gamma^{(1-0.68 \cdot \beta)} \cdot (\sin \theta)^{0.7}$ <p>Brace crown</p> $1 + 0.65 \cdot \beta \cdot \tau^{0.4} \cdot \gamma^{(1.09-0.77 \cdot \beta)} \cdot (\sin \theta)^{(0.06 \cdot \gamma - 1.16)}$	
<p>Out of plane bending</p> 	<p>Chord saddle SCF adjacent to brace A:</p> $[\gamma \cdot \tau \cdot \beta \cdot (1.7 - 1.05 \cdot \beta^3) \cdot (\sin \theta)^{1.6}]_A \cdot (1 - 0.08 \cdot (\beta_B \cdot \gamma)^{0.5} \cdot e^{-0.8 \cdot x}) + [\gamma \cdot \tau \cdot \beta \cdot (1.7 - 1.05 \cdot \beta^3) \cdot (\sin \theta)^{1.6}]_B \cdot (1 - 0.08 \cdot (\beta_A \cdot \gamma)^{0.5} \cdot e^{-0.8 \cdot x}) \cdot (2.05 \cdot \beta_{max}^{0.5} \cdot e^{-1.3 \cdot x})$ <p>Where:</p> $x = 1 + \frac{\zeta \cdot \sin \theta_A}{\beta_A}$ <p>Brace A saddle SCF</p> $\tau^{-0.54} \cdot \gamma^{-0.05} \cdot (0.99 - 0.47 \cdot \beta + 0.08 \cdot \beta^4) \cdot [\gamma \cdot \tau \cdot \beta \cdot (1.7 - 1.05 \cdot \beta^3) \cdot (\sin \theta)^{1.6}]_A \cdot (1 - 0.08 \cdot (\beta_B \cdot \gamma)^{0.5} \cdot e^{-0.8 \cdot x}) + [\gamma \cdot \tau \cdot \beta \cdot (1.7 - 1.05 \cdot \beta^3) \cdot (\sin \theta)^{1.6}]_B \cdot (1 - 0.08 \cdot (\beta_A \cdot \gamma)^{0.5} \cdot e^{-0.8 \cdot x}) \cdot (2.05 \cdot \beta_{max}^{0.5} \cdot e^{-1.3 \cdot x})$	<p>F4</p> <p>F4</p>

Table 6-4. Stress concentration factors for simple K-joints [15]

7 Appendix B. Optimization algorithm m-files. Main script

INPUT

```
% GA parameter

% General

N          = 9;          % Population size

l_chro(1)  = 11;        % Chromosome length for Diameter

l_chro(2)  = 6;         % Chromosome length for Thickness

gen_max    = 1000;     % Maximum number of generations

last_impro = 0;        % Parameter to record last improvement

% Mutation

pm         = 0.01;     % Mutation probability

pm_max     = 0.12;     % Maximal mutation probability

mut_gen    = [30 200]; % Range where mutation is applied

mut_int    = 0.01;     % Increase for mutation probability

mut_last   = 20;      % How long does the fitness to be equal to
increase the mutation probability?

% Immigration

Immi       = [10 200]; % Range where immigration is applied

Immi_int   = 10;      % Interval for immigration

Immi_P     = 6;       % Defines number of immigrations per
immigration process

% Crossover
```

```

        cut int      = [6 8];           % Define the range for the amount of cut
        locations per crossover

        cut          = 1;               % Shall the cuts for the crossover be
        picked in equally divided parts of the chromosome [1] or randomly within the
        whole chromosome [0]?

% Jacket parameter

        N_bay       = 4;               % Number of bays

        Minimum(1,1) = 800;            % Minimal diameter leg [mm]

        Minimum(1,2) = 10;             % Minimal thickness leg [mm]

        Minimum(2,1) = 400;            % Minimal diameter brace [mm]

        Minimum(2,2) = 10;             % Minimal thickness brace [mm]

        Dlight      = 100196;          % Lightest possible design [kg]

        Dheavy      = 3590146;         % Heaviest possible design [kg]

% Simulation parameter

        modelfile    = ['E:\2014_GeneticOptimization\model\FedemRun'
        num2str(N) '\OC4-master-' num2str(N) '.fmm'];    % Path for FEDEM model file

        timedata     = [60, 0.05, 30, 20, 100];
        % Time series cut off [s], simulation timestep [s], analysis length [s],
        lifetime scale [y], number of bins for Palmgren-Miner

        timedata(1,6) = timedata(1,4) * 365.25*24*60*60 / timedata(1,3);
        % Scale number of cycles to lifetime

% Initial values

        for p=1:N

                for n=1:N_bay

```

```

% Pick initial values until a valid design is created

% (validity is based on validity range for SCFs)

Valid=0;

while Valid==0

    Valid=1;

    for i=1:2

        for c =1:2

            DIM{1,p}{n,i}(c) =
randi([Minimum(i,c),2^l_chro(c)+Minimum(i,c)-1]);

        end

    end

    % Check geometrical parameters (used for SCF calculation) to
get rid of invalid designs

    % beta

    beta=DIM{1,p}{n,2}(1)/DIM{1,p}{n,1}(1);

    % tau

    tau=DIM{1,p}{n,2}(2)/DIM{1,p}{n,1}(2);

    % gamma

    gamma(1)=DIM{1,p}{n,1}(1)/2/DIM{1,p}{n,1}(2);

    gamma(2)=DIM{1,p}{n,2}(1)/2/DIM{1,p}{n,2}(2);

    % Check validity range

    % Set validity to false (zero), if the geometrical

    % parameters do not fulfil the requirements (DNV-RP-C203)

    if beta < 0.2 || beta > 1.0 || tau < 0.2 || tau > 1.0 ||
min(gamma(:)) < 8 || max(gamma(:)) > 32

        Valid=0;

    end

```

```
end  
  
end  
  
end
```

OPTIMIZATION

```
for gen=1:gen_max  
  
    % Convert [mm] to [m] in DIM matrix (necessary for FEDEM)  
  
    DIMm = cell(1,N);  
  
    for p=1:N  
  
        for n=1:N_bay  
  
            for i=1:2  
  
                DIMm{1,p}{n,i} = DIM{gen,p}{n,i}/1000;  
  
            end  
  
        end  
  
    end  
  
end  
  
% Run time-domain simulation in FEDEM  
  
runFEDEM(modelfile,gen,N,DIMm)  
  
% Calculate weight  
  
W = calcWEIGHT(N,DIMm);  
  
WEIGHT{gen}=W;  
  
% Calculate damage  
  
D = calcDAMAGE(modelfile,gen,N,DIMm,timedata);
```

```

DAMAGE{gen}=D;

fprintf(['Generation ' num2str(gen) '\n']);

% Evaluation of fitness

if mod (gen,2)==1 % evaluate fitness based on weight

    for p=1:N

        if all (D{1,p}(:,1)<=1)

            % Evaluate only designs with damage <=1

            FIT{gen} (p,1)=Dheavy-sum(W{1,p}(:,1));

        else

            % Set fitness to zero, if damage doesn't fulfil the

            % requirements

            FIT{gen} (p,1)=0;

        end

    end

else % evaluate fitness based on damage

    for p=1:N

        if all (D{1,p}(:,1)<=1)

            if all (W{1,p}(:,1)<=Dheavy)

                % Evaluate only designs with weight <=Dheavy

                FIT{gen} (p,1)=sum(D{1,p}(:,1));

            else

                FIT{gen} (p,1)=0;

            end

        else

            % Set fitness to zero, if weight doesn't fulfil the

```



```
        % requirements

        FIT{gen}(p,1)=0;

    end

end

end

% Selection of leading designs

% To ensure a continuous improvement of the designs, the best
% designs of the current generation and the generation prior to it
% are selected

if gen>1

% Matrix LEAD includes the fitness of the current and the prior
% generation

LEAD=[FIT{gen-1}(:,1); FIT{gen}(:,1)];

    for p=1:N

        % Search for the individual with the highest fitness

        [fit_ind,ind]=max(LEAD);

        FIT{gen}(p,1)=fit_ind;

        % Set the fitness of the picked individual to -1 to ensure
        % that it will not be picked again

        LEAD(ind)=-1;

        % Write the dimension of the selected design in the DIM

        % matrix

        if ind<=N

            DIM{gen,p}=DIM{gen-1,ind};

        else


```

```

        DIM{gen,p}=CHILD{gen-1,ind-N};

        end

    end

end

% Scaling of fitness

% Calculated min, max, mean fitness of current generation

f_raw_min=min(FIT{gen}(:,1));

f_raw_max=max(FIT{gen}(:,1));

f_raw_mean=mean(FIT{gen}(:,1));

% Save maximal fitness of generation

    fit_max(gen)=f_raw_max;

% Record improvement of fitness

    if f_raw_max > max(fit_max(1:gen-1))

        last_impro=gen;

    end

% Linear scalling

    sc_a=2*f_raw_mean/(f_raw_max-f_raw_min);

    sc_b=sc_a*f_raw_min;

    for p=1:N

        FIT{gen}(p,2)=sc_a*FIT{gen}(p,1)+sc_b;

    end

% Fitness plot

% This part is plotting the figure and writting some output

```

```

        if gen>1

            figure(1)

            hold on

            title(['Fitness for generation ' num2str(gen) '; Last
improvement: ' num2str(last_impro) '; Weight of best Design: '
num2str(round(Dheavy-f_raw_max)/1000) 't'],'fontsize',16)

            plot((1:gen),fit_max)

            hold off

            fprintf(['Maximal Fitness ' num2str(f_raw_max) '\n']);

            fprintf(['Best weight ' num2str(round(Dheavy-f_raw_max)/1000)
'\n']);

            fprintf(['Last improvement ' num2str(last_impro) '\n']);

            fprintf(['Mutation probability ' num2str(m) '\n\n']);

        end

% Mutation

% The mutation probability is based on changes in fitness and will not
% necessarily be applied for the whole calculation

        if gen>mut_gen(1) && gen<=mut_gen(2)

            if (max(FIT{gen}(:,1))-max(FIT{gen-mut_last}(:,1)))==0;

                m=m+mut_int;

                % Mutation probability greater pm_max is not allowed

                if m>pm_max

                    m=pm_max;

                end

            else

                m=pm;

            end

        else

```

```

        m=pm;

    end

    % Convert dimension to binary code

    for p=1:N

        for n=1:N_bay

            for i=1:2

                for c=1:2

                    GENE{1,p}{n,i}{c}=dec2bin(DIM{gen,p}{n,i}(c)-
Minimum(i,c),1_chro(c));

                    if n==1 && i==1 && c==1

                        CHROMOSOME{1,p}=GENE{1,p}{n,i}{c};

                    else

                        CHROMOSOME{1,p}=[CHROMOSOME{1,p}
GENE{1,p}{n,i}{c}];

                    end

                end

            end

        end

        end

    % Save the designs in binary code in matrix MODEL before

    % crossover (necessary for comparison - to avoid already

    % calculated designs)

    MODEL{gen,p}=CHROMOSOME{1,p};

    % Immigration

    % Defines how many immigrations shall be picked

    if gen==Immi(1) && p>N-Immi_P && gen<=Immi(2)

```

```

        % Create a new random design
        CHROMOSOME{1,p}=dec2bin(randi([0 1]));

        for x=2:8*sum(l_chro(:))

            CHROMOSOME{1,p}=[CHROMOSOME{1,p} dec2bin(randi([0 1]))];

        end

        % Set the value for the next generation with immigration

        if p==N

            Immi(1)=Immi(1)+Immi_int;

        end

    end

end

end

% Reproduction process

for p=1:N

    % Continue reproduction process until a valid design is created

    % (validity is based on validity range for SCFs)

    Valid=0;

    while Valid==0

        % Selection of parents

        prop = cumsum([0 FIT{gen}(:,2) ./sum(FIT{gen}(:,2))] );

        prop(end) = 1e3*eps + prop(end);

        [a,Par(p,1)] = histc(rand,prop);

        [a,Par(p,2)] = histc(rand,prop);

        % Select another parent in case of equal parent

        while Par(p,1)==Par(p,2)

            [a,Par(p,2)] = histc(rand,prop);

        end

    end

end

```

```
% Crossover

% Crossover can take place at different cuts. The
% variable rand_cut defines at how many locations the
% design (CHROMOSOME) shall be cut. It can be picked
% randomly in a predefined range or can be defined by
% the user

rand_cut=randi(cut_int);

clear cross

if rand_cut==0

    % No crossover if rand_cut is set to 0

    CHROMOSOME{2,p}=CHROMOSOME{1,Par(p,1)};

else

    % The cut location can be picked within equally
    % divided CHROMOSOME parts or randomly picked
    % within the whole chromosome

    for s=1:rand_cut

        if cut==0

            % Randomly picked within the whole
            % chromosome

            cross(s)=randi(8*sum(l_chro(:)));

            cross=sort(cross);

        elseif cut==1

            % CHROMOSOME is equally divided by numbers
            % of cuts. The first cut location will be
            % picked within the first interval.
```

```

                                cross(s)=randi([round(1+(s-
1)*8*sum(l_chro(:))/rand_cut) round(s*8*sum(l_chro(:))/rand_cut)]);

                                end

                                end

                                % Putting the new CHROMOSOME together

                                % The new CHROMOSOME is temporarily saved in the

                                % second line, since we can not safely say that

                                % this is a valid design

                                CHROMOSOME{2,p}='';

                                i=1;

                                for x=1:rand_cut

                                    if x==1

                                        CHROMOSOME{2,p}=[CHROMOSOME{2,p}
CHROMOSOME{1,Par(p,i)}(1:cross(1))];

                                        else

                                        CHROMOSOME{2,p}=[CHROMOSOME{2,p}
CHROMOSOME{1,Par(p,i)}(cross(x-1)+1:cross(x))];

                                        end

                                    if i==1

                                        i=2;

                                    else

                                        i=1;

                                    end

                                end

                                end

                                CHROMOSOME{2,p}=[CHROMOSOME{2,p}
CHROMOSOME{1,Par(p,i)}(cross(x)+1:8*sum(l_chro(:)))]);

                                end

```

```

% Mutation

%Check every digit

for x=1:8*sum(l_chro(:))

    if rand(1)<=m

        % Change digit based on mutation probability

        if strcmp(CHROMOSOME{2,p}(x:x),'1')==1

            CHROMOSOME{2,p}(x:x)='0';

        else

            CHROMOSOME{2,p}(x:x)='1';

        end

    end

end

end

% Cut CHROMOSOME to get diameter and thickness per bay

for n=1:N_bay

    for i=1:2

        for c=1:2

            GENE{2,p}{n,i}{c}=CHROMOSOME{2,p}(1+(n-1)*2*sum(l_chro(:))+(i-1)*sum(l_chro(:))+(c-1)*l_chro(1):(n-1)*2*sum(l_chro(:))+(i-1)*sum(l_chro(:))+(c-1)*l_chro(1)+l_chro(c));

        end

    end

end

% Check geometrical parameters (used for SCF calculation) to
get rid of invalid designs

Valid=1;

for n=1:N_bay

    D=bin2dec(GENE{2,p}{n,1}{1})+Minimum(1,1);

```



```

T=bin2dec(GENE{2,p}{n,1}{2})+Minimum(1,2);

d=bin2dec(GENE{2,p}{n,2}{1})+Minimum(2,1);

t=bin2dec(GENE{2,p}{n,2}{2})+Minimum(2,2);

% beta

    beta=d/D;

% tau

    tau=t/T;

% gamma

    gamma(1)=D/2/T;

    gamma(2)=d/2/t;

% Check validity range

    % Set validity to false (zero), if the geometrical

    % parameters do not fulfil the requirements (DNV-

RP-C203)

    if beta < 0.2 || beta > 1.0 || tau < 0.2 || tau >

1.0 || min(gamma(:)) < 8 || max(gamma(:)) > 32

        Valid=0;

    end

end

% Check if design was already used

for igen=1:gen-1

    for ip=1:N

        if MODEL{igen,ip}==CHROMOSOME{2,p}

            Valid=0;

        end

    end

end

end

```

```

        end

    end

    % Convert dimension to decimal numbers (for next generation)

    for p=1:N

        for n=1:N_bay

            for i=1:2

                for c=1:2

DIM{gen+1,p}{n,i}(c)=bin2dec(GENE{2,p}{n,i}{c})+Minimum(i,c);

CHILD{gen,p}{n,i}(c)=bin2dec(GENE{2,p}{n,i}{c})+Minimum(i,c);

                end

            end

        end

    end

    % Scatter plot

    % extract data of all new children

    clear Dnew

    Dnew = cell(1,2);

    for t=1:2

        ind = 0;

        for p=1:N

            for n=1:N_bay

                ind = ind + 1;

                Dnew{1,t}(ind,1) = CHILD{gen,p}{n,t}(1,1)+0.5*rand;
% diameter

                Dnew{1,t}(ind,2) = CHILD{gen,p}{n,t}(1,2)+0.5*rand;
% thickness
            end

        end

    end

```

```

        end

    end

end

% scatter plot

% definitions

clear c pl

Pmax(1) = 3000;

Pmax(2) = 80;

c{1}    = [0.8 0.8 0.8];

c{2}    = 'green';

c{3}    = 'blue';

Lim(1)  = Minimum(1,1);

Lim(2)  = Minimum(2,1);

figure(2)

for t=1:2

    subplot(1,2,t)

        hold on

        % plot binary limitations

        plot([0 Pmax(1)], [10 10], 'color', c{1})

        for T=1:6

            plot([0 Pmax(1)], [2^T+10 2^T+10], 'color', c{1})

        end

        plot([Lim(t) Lim(t)], [0 Pmax(2)], 'color', c{1})

        for D=1:11

            plot([2^D+Lim(t) 2^D+Lim(t)], [0
Pmax(2)], 'color', c{1})

        end

        % plot SCF limitations

```

```

        Dg      = 0:Pmax(1);

        Tg08    = Dg/(2*8);

        Tg32    = Dg/(2*32);

        pl(3) = plot(Dg,Tg08,'red');

                plot(Dg,Tg32,'red')

        % plot data points

                pl(t) =
scatter(Dnew{1,t}(:,1),Dnew{1,t}(:,2),20,c{t+1},'fill');

        hold off

        axis([0 Pmax(1) 0 Pmax(2)])

        xlabel('Diameter [mm]')

        ylabel('Thickness [mm]')

        end

        subtitle(['Population selection, all children of '
num2str(gen) ' generations'])

        legend(pl,'Legs','Braces','SCF \gamma','Location','SouthEast')

        % Save some output

        if gen>1

                save(['results\Generation' num2str(gen) 'N'
num2str(N)], 'DIM', 'FIT', 'GENE', 'LEAD', 'CHILD', 'DAMAGE', 'WEIGHT')

        end

end

```