Optimizing the concrete load-bearing structure of high-rise buildings

Combining the ground structure method with a recursive resizing algorithm on a case study

B.W. Gerritsen

Technische Universiteit Delft

TU Delft

zonneveld ingenieurs

Optimizing the concrete load-bearing structure of high-rise buildings

Combining the ground structure method with a recursive resizing algorithm on a case study

by

B.W. Gerritsen

to obtain the degree of Master of Science
at the Delft University of Technology,

Student number: 4288122
Project duration: June 1, 2019 – May 6, 2020
Thesis committee:

Dr. ir. M.A.N. Hendriks,	TU Delft, chairman
Ir. R. Crielaard,	TU Delft
Ir. R. Verstralen,	Zonneveld ingenieurs b.v.
Ir. L.P.L. van der Linden,	TU Delft

TU Delft

zonneveld ingenieurs

Preface

This Msc. thesis concludes my masters in Structural Engineering with specialization Structural Mechanics of the faculty of Civil Engineering and Geosciences at the Delft University of Technology. Before you start reading my work, I would like to thank those people who contributed to the process of writing this thesis, the people that contributed to making my life as a student amazing and the people that supported me over the last few years.

I would like to thank Zonneveld ingenieurs for their hospitality, use of their resources, knowledge and information about Boston & Seattle (including the image on the front cover). In specific my supervisor from Zonneveld, Ir. R. Verstralen. Also, I would like to express my gratitudes to the other members of my committee: Dr. ir. M.A.N. Hendriks, Ir. R. Crielaard and Ir. L.P.L. van der Linden.

I want to thank everyone who contributed to my student life, the people who I met during going out, friends I made in South-Africa when I was doing a research project there, friends from the sport association, fellow student and friends from the university and anyone else who I met during studying. Even though I haven't seen them in a while in person due to the recent outbreak of the COVID-19 virus. Also, I want to thank Google who supported me by answering many of my questions during my (student) life.

Last but not least, I would like to thank my family and friends who supported me and gave me the motivation to complete this thesis. In specific, I want to thank Lise for all her feedback and support!

I wish you a pleasant and interesting reading!

B.W. (Wouter) Gerritsen
Delft, May 6, 2020

Abstract

The shortage in housing and office spaces, combined with the desire of people to live in densely populated cities results in a lack of space. A proposed solution can be found in the usage of high-rise buildings. Nowadays there is more awareness for the environment, thus this research tries to reduce the environmental impact of a high-rise building by optimizing the material used in load-bearing structures. This research aims to give designers and engineers more insight into the added value of structural optimization; in particular for the material usage in the load-bearing structure of high-rise buildings. The research objective is formulated as follows:

"What is the optimal topology for a reinforced concrete load-bearing structure, situated at the perimeter of a high-rise building when optimizing the material use?"

A building is classified as high-rise building when its roof is 70 m or more above ground level and accommodates work and/or living space. In literature the distinction is made between three sorts of optimization: size, shape and topology optimization. Topology optimization has the most freedom, therefore it is more likely to find a novel structure which minimizes the material use as much as possible. In specific the Ground Structure Method (GSM) is used. The initial ground structure is constructed by creating members between all nodes (full connectivity) which are located in the design space. The cross-sectional areas of the bars are the design variables in the optimization. An option is that the variables turn to zero, thus elements are deleted resulting in less material. The conventional GSM starts with the full connectivity as initial ground structure, deletes elements and calculates the new load distribution until a threshold is reached. In the end the load-bearing structure will consist of columns, braces and beams, which are located at the perimeter of the floor plan. Literature indicates that (high-rise) buildings which are mainly loaded by horizontal loads would be optimal if they contain arches in the load-bearing structure, or resemble the so-called Michell truss [5, 6, 49, 62]. However, the influence of the vertical load is often not taken into account, therefore this is investigated with the help of a parametric study. To compare the results with reality, a case study (Boston & Seattle, Rotterdam) is investigated.

The core of the research is the further development of the optimization code, based on the GSM, written by He et al. [31], where multiple load cases and demands for the material strengths are implemented. In contrast to deleting elements, the code uses an adaptive 'member adding' scheme, which is firstly proposed by Gilbert and Tyas [26]. This scheme solves problems faster than the conventional GSM, up to 8 times, and is able to solve large problems [31]. This code is extended during this research with new functions which implement demands for fire, second-order, buildability, flexural buckling and stiffness. Also it is possible to add self-weight to the optimization. The stiffness is implemented by adding a constraint to the displacement of the top of the building, wherefore a recursive resizing algorithm is written based on the article of Chan [17]. Thus the extended code exists of two optimizations, first a strength optimization and afterwards a stiffness optimization. The code is written in Python and for the purpose of post-processing exporting the data to Excel and Abaqus is possible.

With the help of the extended code, two design spaces are investigated. One design space is smaller, such that the computational time is low and many variations can be examined during the parametric study on the total vertical to total horizontal load ratio (v/h-ratio). The other is based on a case study for comparison with a realistic situation. The verification of the code shows that the extra functions work properly for the investigated problems. If the functions concerning the stiffness optimization, inclusion of self-weight, fire, buildability

and second-order are used, the extended code becomes unstable and the computation time increases enormously when optimizing the case study. Therefore it is chosen not to include during obtaining the results.

The results of the parametric study shows an expected pattern for the load-bearing structure, for a v/h-ratio below 4. The pattern consist of arches, originating from the Michell truss. The optimization of the case study, subjected to one realistic load combination, showed no clear pattern for the load-bearing structure. Post-processing steps, based on engineering judgement, are taken to clarify the solutions, which showed that the columns above the supports should be large in comparison to the other elements and that the arches are the most optimal structure. Therefore an ”optimized” load-bearing structure consisting of arches is proposed. The analysis of the ”optimized” load-bearing structure shows us that most of the elements meet the strength requirements. To find the optimal solution an iterative process would be needed, because increasing the cross-section of an element will decrease the stress but increase the stiffness, thus attracting more load.

The difference between the optimized load-bearing structure and the original load-bearing structure of the case study is that the optimized uses arches instead of (punched) structural walls and uses less material ($\pm$3%). From the post-processing of the results it is concluded that increasing the strength ratio (compression to tensile) to 1.0, decreasing the total v/h-ratio or ignoring the rigid-diaphragm working of the floor help clarify the results of the optimization.

This research extends the current literature with extra insights in the use of the Ground Structure Method in an optimization code. Also, it confirms that the arches (originating from Michel Truss) is an efficient manner to transfer the loads to the supports. However, more research in the influence of the supports, the design space and the material type on the clearness of the optimal solution is needed.

The advice for designers and engineers is to see what the possibilities are for arches to use in their load-bearing structure, because these are efficient in transferring the loads so material can be saved. The current version of the code needs to be made more user friendly, stabler and faster before it is recommended to be used by designers and engineers. The extended code is a first attempt to implement multiple rules from the Eurocode in a optimization code.

List of symbols

All values are in SI-units (unless specified otherwise): distances in meters, time in seconds, force in Newtons and mass in kilograms.

$\mathbf{B}$	[-]	Equilibrium matrix
A	[-]	Constant for calculating the slenderness limit
A_c	[m^2]	Total area of the concrete in the cross-section
A_s	[m^2]	Total longitudinal reinforcement area in the cross-section
B	[-]	Constant for calculating the slenderness limit
C	[-]	Constant for calculating the slenderness limit
E	[N/m^2]	Young's modulus of concrete
I	[m^4]	Moment of inertia
M_{01}, M_{02}	[Nm]	First-order end moments, $\lvert M_{02}\rvert \geq \lvert M_{01}\rvert$
W	[m]	Geometry parameter in 3D cantilver optimization problem
a	[m^2]	Vector containing all a_i
a_c	[m^2]	Cross-sectional area
a_i	[m^2]	Cross-sectional area of member i
$a_{i,1}$	[m^2]	Cross-sectional area of member i as result of optimization part one
a_1	[m^2]	Vector containing all $a_{i,1}$
b_c	[m]	Width of the cross-section
e_i	[m]	Surcharge eccentricity
e_{tot}	[m]	Total eccentricity
e_0	[m]	First-order-eccentricity
f	[N]	Vector containing all f_j
$f(x)$	[-]	Objective function depending on variable x
f_{cd}	[N/m^2]	Design compression strength
f_{ck}	[N/m^2]	Characteristic concrete compression strength
f_{ctd}	[N/m^2]	Design tensile strength
f_i^{sw}	[N]	The load due to self-weight of the member i
f_j	[N]	External force on node j
f_{yd}	[N/m^2]	Design yield strength of the reinforcement
f_{yk}	[N/m^2]	Characteristic strength of the reinforcement
f^k	[N]	Vector containing all f_j for load case k
f^{sw}	[N]	Vector containing information about the load due to self-weight of the members on the nodes
g	[m/s^2]	Gravity acceleration
$g_k(x)$	[-]	Inequality constrain function depending on variable x
g^u	[m]	Right-hand side of the inequality constraint (displacement of the top floor)
g^t	[m]	Left-hand side of the inequality constraint (displacement of the top floor)
h	[m]	Height of the building
h_c	[m]	Height of the cross-section
$h_j(x)$	[-]	Equality constrain function depending on variable x
h_s	[m]	Floor height
i	[m]	Radius of gyration
k	[-]	Load case number
l	[m]	Vector containing all l_i
l_i	[m]	Length of member i
l_m	[m]	length of a member
l_0	[m]	Buckling length

m	[-]	Amount of members
n	[-]	Amount of nodes
n_d	[-]	Number of design variables
n_g	[-]	Number of inequality constrains
n_h	[-]	Number of equality constrains
n_{rel}	[-]	Relative normal force, $N_{Ed}/(A_c f_{cd})$
p	[-]	Amount of load cases
q	[N]	Vector containing all q_i
q_i	[N]	Force in member i
$q_{u,i}$	[N]	Force in member i, caused by a unit load at the point of interest
q^k	[N]	Vector containing all q_i for load case k
r_m	[-]	Moment ratio, M_{01}/M_{02}
t	[-]	Iteration step
V	[m^3]	Volume
x_i	[-]	i^{th} Design variable
x_i^l	[-]	Lower bound of the design variable x_i
x_i^u	[-]	Upper bound of the design variable x_i
z_1	[-]	Lagrange multiplier
Φ	[-]	Reduction factor due to flexural buckling
α_{cc}	[-]	Reduction factor which considers long-term effects and unfavourable acting of a force on compression strength
α_{ct}	[-]	Reduction factor which considers long-term effects and unfavourable acting of a force on tension strength
γ_c	[-]	Material factor for concrete
γ_s	[-]	Material factor for steel
η	[-]	Relaxation parameter
λ	[-]	Slenderness
λ_{lim}	[-]	Slenderness limit
ρ	[kg/m^3]	Density of reinforced concrete
ϕ_{ef}	[-]	Effective creep coefficient
ω	[-]	Mechanical reinforcement ratio, $A_s f_{yd}/(A_c f_{cd})$

Contents

1 Introduction

1.1. Motive

The Netherlands has a housing shortage, approximately a shortage of 263.000 houses at the start of 2019 [66]. Also, the number of vacant office buildings is decreasing. More major cities in the Netherlands need to deal with a shortage in office-buildings [14, 23]. An option to resolve these deficits is to build new houses and office buildings [58]. These new buildings will occupy land, but the land is not only used for living and working but also for leisure, agriculture and nature (think of the "Groene Hart"). The Netherlands is a densely populated country (510 people per km^2 in 2018 [15]), which means that the land must be used efficiently. Also, people want to live and work in densely populated cities, where the land is even more scarce. Concluding, there is a lack of space in the Netherlands, especially in the urban environment. A solution is to increase the height of the buildings; high-rise buildings. These enable space for living or offices without taking a lot of square footage of the valuable land, using the land as efficiently as possible.

1.1.1. Background

Currently the tallest building in the world is located in Dubai, the Burj khalifa, with a height of 828 m shown in figure 1.1 [35].

Figure 1.1: Burj Khalifa [35]

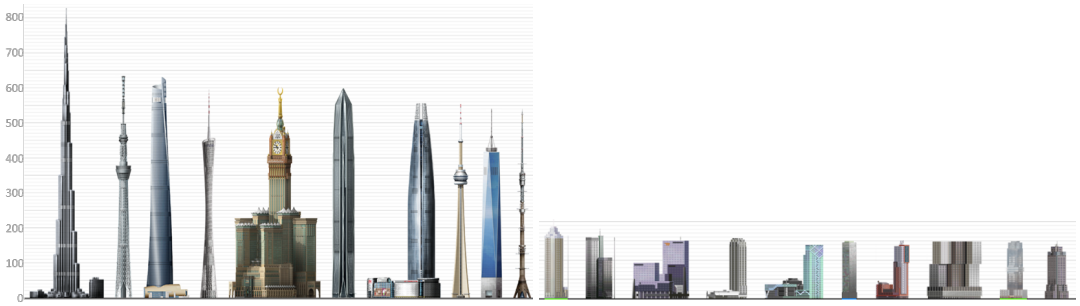
There are more super-tall buildings in the world, especially in the Middle East, Asia and North America. These continents have multiple buildings above 400 m. These buildings have among others residential, office, hotel or retail areas [16]. Next to these functions they also have an aesthetic value; the buildings are landmarks. These high-rise buildings can be seen from far and wide and quite some cities are known for their high-rise buildings (such as The Empire State Building in New York and the hotel Marina Bay Sands in Singapore).

In Table 1.1 the top 10 of cities with the highest number of buildings above 90 m is shown, which are all densely populated cities. In the Netherlands Rotterdam is the only city where people live in high-rise buildings on large scale. The city has 29 buildings above 90 m [69].

Table 1.1: Top 10 of cities with buildings above 90 m in the world [69]

City	Number of buildings (>90 m)
Hongkong	2939
New York	849
Tokyo	572
Shanghai	549
Bangkok	382
Chicago	321
Signapore	296
Sao Paulo	281
Seoul	273
Dubai	268

Figure 1.2a and 1.2b respectively show the top 10 highest buildings in the world and the Netherlands. In these figures, the height of the buildings is scaled, such that they are comparable. As mentioned before the tallest building in the world is the Burj Khalifa, the tallest building in the Netherlands will be the Zalmhaven Toren (215 m) when it's finished in mid-2020 [21].



(a) High-rise buildings in the world [47]

(b) High-rise buildings in the Netherlands [46]

Figure 1.2: Top 10 high-rise in the world (1.2a) and in the Netherlands (1.2b), scale is in meters

Currently, the municipalities in the Netherlands are adapting their definitions of high-rise (these can be found in Appendix A) and their policy on high-rise buildings to a more modern point of view [69]. For example, the municipality of The Hague has the demand that each building has its own "Haagse" signature meaning each building has to have a recognisable top [54]. With this vision, the municipality of The Hague creates landmarks in their city.

Limitations for high-rise buildings in the Netherlands are among others, the soil condition, minimal natural light in buildings, logistics, financial and the historic inner cities. Due to the soil conditions in the Netherlands, it is preferable to build on pile foundations such that the structure will not subside which could make the design more challenging. The minimal incidence of natural light makes the Dutch towers very slender for low heights. Next to

the design problems the logistics could also be problematic. As mentioned before, in the densely populated areas the ground is scarce. Thus when building in there areas, there is minimal space for equipment and storage. The Netherlands has cities with old historical centre, similar to other European cities but different than North-American cities. Since these centres aren't tall, it is not desirable to build tall buildings here because they won't fit in. At last, it is a difficult business case. The rent price per square meter is low in the Netherlands (compared to other major cities in the world) and the building costs are normally higher compared to those of low-rise buildings [17].

The market is asking for cheaper buildings, cheaper building methods, shorter building time and nowadays also for more sustainable buildings. A way to achieve this is modular building. Research concludes that the building time and waste on-site is reduced when building with modules [40]. Another possibility is to optimize high-rise buildings, leading to lowering the material use, which could result in lower costs and is better for the environment. Other research describes that the impact of buildings in use or under construction are the greatest indirect source of carbon emissions [3]. Proposed solutions are, adapting the architecture of the building such that less heating/cooling is needed and using double-layer glass facade [57]. Thus less material needs to be used or the material needs to be used more efficiently, such that the harm to the environment is mitigated and the costs are kept low.

1.1.2. Problem description

The shortage in housing and office space, combined with the desire of people to live in the densely populated cities results in a lack of space in urban areas. A proposed solution for this is high-rise buildings instead of low-rise buildings, such that the areas are efficiently used. Next to the demands from the buildings codes, market forces have begun to push the governments and the private sector towards renewable energy in most industrialized countries to create more awareness for the environment. Therefore, tall buildings should become more sustainable considering the environment, long-term economic growth, and human needs. This problem can be solved by reducing the environmental impact of a high-rise building with the help of high-performance design [3]. One of the components of high-performance design is the use of material. Optimizing high-rise buildings is a method to find possible material saving ideas, through using the material more efficient in the load-bearing structure. This will be the focus of this research, from which in Section 1.2 the research aim, objectives, methodology and scope are defined.

1.2. Research objective, methodology and scope

The research aims to give designers and engineers more insight into the added value of structural optimization. In particular for the case of material usage in the load-bearing structure of high-rise buildings. From this aim, the research objective(s) are formulated.

1.2.1. Research objective

The research objective is formulated as follows:

"What is the optimal topology for a reinforced concrete load-bearing structure, situated at the perimeter of a high-rise building when optimizing the material use?"

This objective will be investigated with a parametric study and a case study. Afterwards the results will be placed in perspective to draw conclusions for a more general case. The research objective is divided into eight sub-research questions, these are stated and explained in Paragraph 1.2.2. These sub-research questions are answered in the different chapters of this report.

1.2.2. Methodology

In this paragraph the methodology is explained. As mentioned before the main research question is divided into sub-research questions. Underneath each sub-research question there is a short explanation.

1. *What is the definition of a high-rise building and what kind of load-bearing structures exists?*
   The first question results in a definition of high-rise buildings and gives background information about the different load-bearing structures.

2. *What are the requirements to a high-rise structure, according to the Dutch code?*
   The Ultimate Limit State (ULS), Serviceability Limit State (SLS) requirements, fire and practical requirements are discussed.

3. *What kind of structural optimization methods exists and which suits the research objective the best?*
   A brief overview of the structural optimization methods, the conclusion is an optimization method which suits the research objective the best. Also, previous studies in structural optimization methods are presented.

4. *What are the characteristics of the case study regarding the structure?*
   A case study is chosen based on information from question one. The characteristics of the case study are noted: the floor surface, the height of the building, amount of floors, etc. Some of these characteristics are marked as demands to which the design space in the optimization must satisfy so that the case study and the optimized structure can be compared.

5. *Which boundary conditions, for the optimization problem, follow from the case study?*
   The characteristics are translated to boundary conditions which are used in the optimization and act as boundary conditions for the design space.

6. *How can the optimization problem, for different design spaces, be defined?*
   With the help of sub-question two and five the boundary conditions, supports, load combinations, material properties, constraints and design space are identified. These are written such that they can be implemented in the optimization code (written in Python 2.7).

7. *Does the optimization code work and what is the input for the optimizations?*
   The optimization code is used to optimize different design spaces. Firstly, the optimization is done with a floor plan where the results are known/can be predicted (with the use of literature from sub-question three). This shows that the optimization method works. Next, the input for the optimizations is stated.

8. *What is the difference between the optimized case study and the original case study and what is learned from this to benefit the design of future buildings?*
   Design spaces are optimized and results are presented, post-processing of the results is conducted if needed. Comparing the load-bearing structures, before and after optimization, will give insight into the usefulness of topological optimization for this particular case study. Furthermore, the comparison and the parametric study is used to translate the results to a more general case.

1.2.3. Scope

In this section the limitations of the research are discussed, below these are itemized and explained:

- The core
  The core will only have the necessary for practical reasons (such as space for elevators) and takes an appropriate part of the vertical load, but does not contribute to the stability of the building. The diaphragm function of the rigid floor is taken into account during the optimization, but instead of transferring it to the core it will transfer the wind load to the perimeter of the structure.
- Linear and non-linear behaviour
  During the optimization linear behaviour of the structure is assumed, the found structure from the optimization is analysed linear. The non-linear behaviour is not implemented in the optimization problem, because the optimization problem will become too complex. During the optimization concerning the displacement of the structure, cracked concrete, in elements under tension, is taken into account by reducing the Young's modulus with 50%.
- The foundation
  The transfer of forces from the building to the ground through a foundation will not be addressed in this research. The focus is on the building and not the interaction with the ground.
- The influence of and on other buildings
  The influence of and on other buildings are not taken into account. Thus the shielding of buildings surrounding the high-rise buildings is not taken into account (should never be taken into account according to NTA 4614-3:2012). The local and/or global increase of loads due to surrounding buildings on the high-rise is not taken into account, because this will complicate the translation of the insights from the case study to the general case.
- The Netherlands
  The chosen case study is a high-rise building in Rotterdam, the Netherlands. So the design rules for high-rise buildings of the Netherlands are used in this research, which complies with the Eurocode. For the wind load, the wind environment of the location of the case study is used. Although the NEN (Dutch version of Eurocode) prescribes calculations on earthquakes (human-induced or natural), these aren't taken into account.

1.3. Reading guide

In Chapter 2 high-rise buildings are discussed answering question one and two. In Chapter 3 the optimization methods are discussed and question three is answered. Question four and five are answered in Chapter 4, through discussing the case study in detail. The information from the previous chapters lead to a fully described optimization problem, stated in Chapter 5. The optimization problem is rewritten to an optimization code and is verified answering question six and seven. At last, question eight is partly answered in Chapter 6 by showing the results and a comparison between the optimized structure and the structure of the case study is made. In chapter 7 the result of the research is discussed and in Chapter 8 conclusions are drawn and some recommendations for further research are done. Also, Chapter 8 partly answers the last sub-research question. Figure 1.3 shows in which chapters which sub-research questions are discussed and how the chapters are related. Concluding, Chapter 2, 4 en 3 are part of the literature review and result in the input for the optimization. Chapter 5 covers the optimization code, Chapter 6 shows the results of the parametric study and the optimization of the case study. The discussion, comparison and design recommendations are done in 7 and 8.

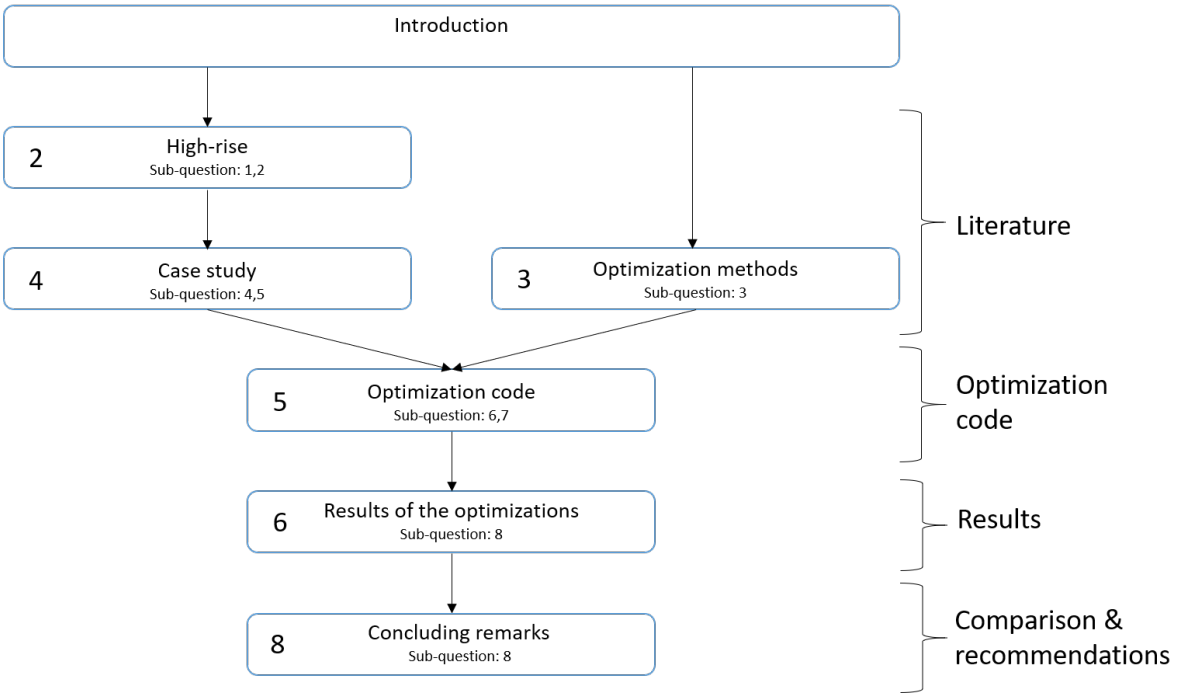


Figure 1.3: Visualisation of the reading guide

2

High-rise building

In this chapter the first two sub-research questions are answered, the first one: *"What is the definition of a high-rise building and what kind of load-bearing structures exists?"* is answered in Section 2.1 and 2.2. In these two sections, the definition of a high-rise in the Netherlands and the classification of the load-bearing structure is discussed. The second question: *"What are the requirements to a high-rise structure, according to the Dutch code?"* is answered in Section 2.3. At last, in Section 2.4, a summary of the discussed points is given.

2.1. Definition of high-rise

As mentioned in the scope, the codes of the Netherlands are used in this research. The definition of high-rise, according to the Dutch dictionary (Dikke van Dale) is *"high building, the opposite of low-rise building"*, which is an ambiguous definition. According to the general part of NTA 4641-1 (Nederlandse Technische Afspraak) a high-rise building is a building, which accommodates work and/or living space, and is 70 m or more above ground level.

Each municipality in the Netherlands has its own policy concerning high-rise, all have their definition of high-rise depending on among others the average height of the current city. For different municipalities the height, from which a building becomes high-rise, is shown in Appendix A together with some regulations at certain heights. The frontier in high-rise buildings in the Netherlands, Rotterdam, uses 70 m, which is the same as the definition of the NTA 4641-1. The definitions which state that high-rise buildings are buildings which are higher than the normal buildings are not convenient, because these depend on the location. Thus the definition of the NTA is adopted in this research because it is a leading document in the Dutch building climate concerning high-rise buildings and it's similar to the definition of the municipality of Rotterdam, which is a frontier on high-rise in the Netherlands. So, in this research a high-rise building is a building where the roof is 70 m or more above ground level and accommodates work and/or living space.

2.2. Load-bearing structure

The load-bearing structure of a building must ensure that the building is strong enough (doesn't collapse), is stiff enough (no excessive deformation and movement for the comfort of the users) and is stable enough (doesn't fall over) [53]. The different types of load-bearing structures will be discussed in Paragraph 2.2.1, in Paragraph 2.2.2 the core of a building is discussed.

2.2.1. Classification

In general, the lateral forces are critical for high-rise buildings, thus most classifications are based on the effectiveness in resisting lateral loads [53]. A rough distinction between different types of load-bearing structures can be made, which is shown in Figure 2.1.

These five systems will be discussed shortly, the frame structure uses stiff connections to construct moment resisting frames. At a certain height, it isn't profitable to build with this system, due to the high costs of producing stiff enough connections [53]. A core, shear walls which are located at the centre of the building, is a commonly used option. The core is used to (partly) ensure the stiffness and stability of the building, next to that the core is used to accommodate, among others, lifts and stairs. This system is very common to use in high-rise [25]. If a core isn't sufficient, the core can be reinforced with an outrigger (concrete, steel or a combination)[53]. The outriggers and the columns, located at the outside of the building, work together as a buttress, which reinforces the core[24]. The tube system can be defined as a three-dimensional system which uses the whole building, the columns at the perimeter of the building are used to resist the lateral loads [4, 53]. The megastructure extends the tube system with braced modules at every $10 - 25$ stories [25].

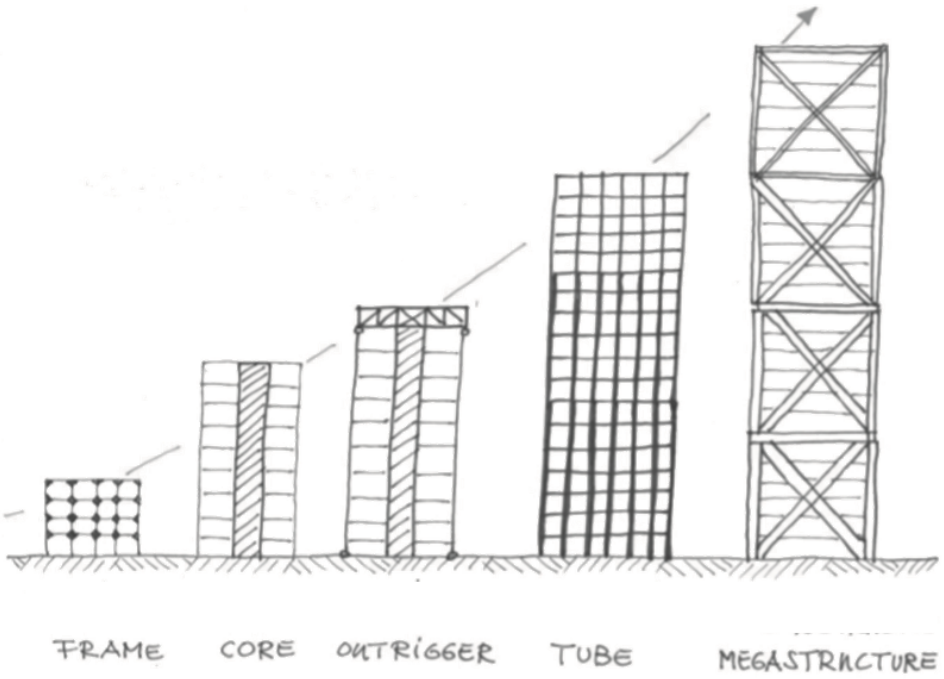


Figure 2.1: Classification of high-rise structural system [53]

In recent years the cost of manufacturing a diagrid system is decreased, through new technologies. Diagrid systems are increasingly used in high-rise buildings due to their structural efficiency [50]. The difference between a diagrid system and a megastructure is that the vertical supports are eliminated and the vertical force transport is done by the diagonals, shown in Figure 2.2.

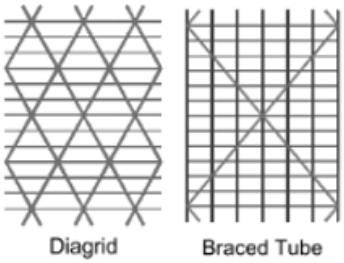


Figure 2.2: Diagrid vs. braced structure (adapted from [51])

Other classifications can be found in literature, such as classification based on the interior or exterior location of the primary lateral load-bearing structure [4]. Or classification which is based on steel and (reinforced) concrete [4, 71]. Each material has advantages and disadvantages. Steel systems are fast to construct and light (in comparison with concrete), thereby decreasing demand on the foundation. On the other hand, reinforced concrete systems are more resistant to fire, offer more damping and mass [71].

As mentioned before, in Section 1.1.1, high-rise buildings are, next to their normal function, landmarks. These buildings become more and more an art object, a good example of this is the work of Zaha Hadid architects. Some classify the designs of these extraordinary buildings as free- form design. These buildings are so complex that they often are designed and calculated with the help of the computer, which is becoming more and more normal in the civil engineering sector (think of BIM and FEM-calculations). Examples of landmarks with an extraordinary design are given in literature [36, 44], other designs of (optimized) high-rise buildings can be found in the article of Beghini et al. [6]. One of the conceptual designs is shown in Figure 2.3.

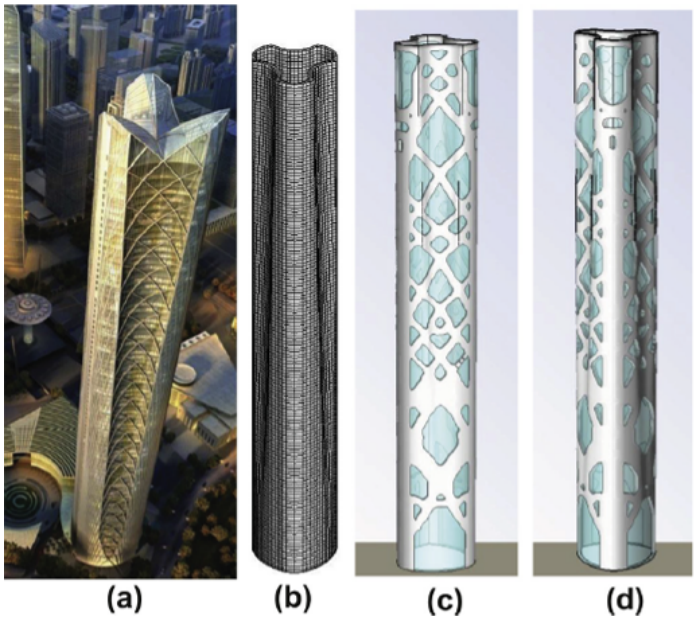


Figure 2.3: Illustration of the concept design for the Wuhan competition: (a) architectural rendering of the final design, (b) finite element mesh, (c and d) topology optimization results [6].

2.2.2. Core

A building which is stabilised through a core consists of at least one core, which is elastically clamped in the ground to ensure the stability and stiffness of the building. Next to that, there are other columns situated in the floor plan to transfer the vertical load. A building which has a rectangular shape could have more cores to withstand the turning of the building, through lateral loading [25, 53].

As mentioned in Paragraph 2.2.1 the core has more functions than providing resistance against loads. For the elevators and stairs are no regulations concerning the natural light, thus these are often placed in the middle of the building. Sound isolation and fire safety regulations result in a sturdy shaft around the stairs and elevators. Because these cores are so well protected, often they are used as an evacuation route in the building. A difficult design problem is the passages through the core, these are needed for people to use the core but weaken the structure[53].

The combination with an outrigger system increases the stability of the core, through which taller buildings can be designed. An example of this is the Shanghai Tower, which is 630 meters high and has approximately every 20 floors an outrigger system and a reinforced concrete core. In Figure 2.4 the tower is shown under construction. Another example of a tall building with a core combined with outrigger is ”The Shard” in London.

Figure 2.4: Shanghai tower under construction [56]

Stability cores are mainly distinguished through material, steel or (reinforced) concrete [25]. In this research, a (reinforced) concrete core is considered. As mentioned in Paragraph 1.2.3, the core will take an appropriate part of the vertical load and hold only the necessary (lift, stairs etc.). The size of the core and the magnitude of the load it will carry will be discussed in Chapter 4 where the case study is investigated. The rest of the load-bearing structure will consist of columns, braces and beams at the perimeter of the building.

2.3. Requirements

A building in the Netherlands is designed based on the Eurocode (and all appendixes), which is translated to dutch by the NEN. Therefore the Eurocode is used to find the requirements to which a high-rise building has to comply. In the Netherlands there is an addition, NTA (Nederlands Technische Afspraak), especially for high-rise buildings, the NTA 4641. The NTA 4641 gives extra information/demands for high-rise buildings. When it contradicts with the Eurocode, the Eurocode is normative. Only the requirements which are taken into account in this research are stated here, these are translated to constraints for the optimization problem in Section 5.4. Other requirements, from the NTA 4641, are stated in Appendix B.4. The requirements stated below are ultimate limit state (ULS), serviceability limit state (SLS), fire and buildability requirements.

The most important demands follow from safety, which ensures that the building will not collapse. A high-rise building has a high consequence class, due to the impact when the building collapses. Therefore several safety factors are included in the ULS calculations and there is a limit on the allowable stresses, under a predefined load. Another important aspect is the behaviour of the building during a fire. After the safety, the buildability is classified as most important, designing a building which can not be built is impractical. Other demands are SLS, these requirements ensure that the building is liveable. People can not get sick or scared when they are in the building, due to noticeable movement of the building. The requirements are sorted in two groups, ULS and fire resistance and SLS and buildability, which are explained in more detail in Paragraph 2.3.1 and 2.3.2.

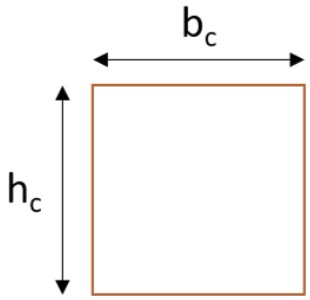


Figure 2.5: Cross-section of a structural element

In this research a square cross-section for structural elements is assumed, as pictured in Figure 2.5. This assumption is made to limit the size of the optimization problem, through making no difference in height (h_c) or width (b_c) of the cross-section. Therefore, when the surface is equal to a_c the following holds: $h_c = b_c = \sqrt{a_c}$. A result of this assumption is that the second moment of inertia (I) is the same in both directions and equal to $\frac{1}{12}a_c^2$ Also, it is assumed that the length for buckling (l_0) is the same as the length of the member (l_m).

2.3.1. Ultimate Limit State and fire

As a result of the consequence class 3, for high-rise buildings, additional methods and techniques are used to demonstrate the increased safety level. It is needed that a systematic risks analysis is executed, where both (extraordinary) foreseen as unforeseen dangerous events are considered. This will be further described in Appendix B.4 and will not be taken into account in the optimization.

The stresses in the reinforced concrete are limited to a design compression and tension stress, these are stated in the NEN-EN 1992-1-1 + C2. It is required that the structure can withstand certain loading, for example, wind, thermal, snow, etc. These are all stated in NEN-EN 1991 Actions on structures. In this research, the loads on the structure will be similar to the loads in the case study. The case study is a realised building, thus the calculations and the loads are assumed to be correct. This will be further discussed in Chapter 4. The following requirements are discussed: design strengths, second-order-effects, Flexural buckling and fire.

Design strengths
The design compression strength of concrete is depending on the concrete class and safety factors. The characteristic compression strength is divided by the material safety factor (γ_c) and multiplied with a factor which considers long-term effects and unfavourable acting of a force (α_{cc}). According to the Eurocode α_{cc} can be taken as 1.0 and the material factor is 1.5. The same holds for the tensile strength of concrete, here (α_{cc}) is called (α_{ct}).

$$\text{Design compression strength} : f_{cd} = \frac{f_{ck}}{\gamma_c} \tag{2.1}$$

$$\text{Design tensile strength} : f_{ctd} = \frac{f_{ctk;0.05}}{\gamma_c} \tag{2.2}$$

Second-order-effects
Second-order-effects should be taken into account in the design of a building. Due to second-order-effects extra forces and moments are added to elements of the structure. The impact of second-order-effects can be neglected when, for independent elements, the slenderness of a structural element (λ) is smaller than a certain limit (λ_{lim}). This is described in NEN-EN 1992-1-1 + C2:2011 (paragraph 5.8.3.1). The i stand for the radius of gyration.

$$\lambda \leq \lambda_{lim} \tag{2.3}$$

$$\frac{l_0}{i} \leq \lambda_{lim}$$

$$\frac{l_m\sqrt{12}}{\sqrt{a_c}} \leq \lambda_{lim} \tag{2.4}$$

The limit on the slenderness depends among others on the relative normal force, mechanical reinforcement ratio and effective creep coefficient. All of these are unknown before the optimization, therefore values can be adopted (described in the Eurocode). The conclusion is that the slenderness limit is 26.18, further explanation over the obtaining this limit is described in Appendix B.1 and visualised in Figure B.1.

$$\frac{l_m\sqrt{12}}{\sqrt{a_c}} \leq 26.18$$

$$a_c \geq \frac{l_m^2}{57.12} \tag{2.5}$$

Flexural buckling
Structural elements which are loaded by an axial force, need to be checked for flexural buckling. According to NEN-EN 1992-1-1+C2:2011 (paragraph 12.6.5.2), the compression strength of the column needs to be reduced by Φ. Φ is the reduction factor and depends on the height of the cross-section, eccentricity and the buckling length of the element.

$$\Phi = 1.14\left(1 - 2\frac{e_{tot}}{h_c}\right) - 0.02\frac{l_0}{h_c} \leq \left(1 - 2\frac{e_{tot}}{h_c}\right) \tag{2.6}$$

with:

e_{tot} = total eccentricity

When all required information is implemented in Equation 2.6, a reduction factor of 0.81 is found. The derivation of this value can be found in Appendix B.2.

$$f_{cd} = \Phi\frac{f_{ck}}{\gamma_c}$$

$$f_{cd} = 0.54 f_{ck} \tag{2.7}$$

Fire
Fire can influence the material properties, hence constraints for the size of load-bearing columns need to be satisfied. According to NTA-6414-3, users of a high-rise building need to have at least 60 min to evacuate the building. Therefore the load-bearing system of the building needs to resist the fire for at least 120 min. It is assumed that the columns are exposed to the fire on more than one side. The minimal column width depends on the load during the fire and the resistance of the column during normal circumstances. In this case these are unknown because the dimensions of the column are determined during the optimization. The NEN-EN 1992-1-2 + C1:2011 (paragraph 5.3.2) prescribes that, for a resistance of 120 min, the minimum column width is 250 mm. Thus the following needs to hold during optimization, visualised in Figure B.1 in Appendix B.3:

$$a_c \geq 0.0625\ \mathrm{m}^2 \tag{2.8}$$

These values are based on the traditional methods for consequence class 1 and 2. For high-rise buildings, consequence class 3, a more elaborate systematic analysis of the risks for the structure is needed, described in Appendix B.4 and NEN-EN 1990.

2.3.2. Serviceability Limit State and buildability
The minimal size for a column is 200 by 200 mm, according to NEN-EN 1992-1-1+C2/NB:2016 (paragraph 9.5.1.). When a building is built with prefab elements and the column is poured horizontally, the minimal size for a column is 150 by 150 mm according to NEN-EN 1992-1-1+C2/NB:2016 (paragraph 10.9.8.).

$$a_c \geq 0.0225\ \mathrm{m}^2 \tag{2.9}$$

This requirement is less stringent than the requirement of the fire resistance, thus satisfying the requirement for the fire resistance is sufficient.

Also, according to the NEN-EN 1992-1-1+C2:2011 (paragraph 5.3.1) a structural element is a column if the length is minimal three times the longest side of the cross-section. Next to that, the longest side of the cross-section must not be longer than four times the shortest side of the cross-section, which is satisfied through the use of a square cross-section. All derived equations (2.5,2.8,2.9) are based on the fact that these structural elements are columns. Also, in the perspective of a minimal daylight illuminance, there are only columns allowed and no walls, such that there is room for openings in the facade. This will improve daylight incidence. Although the requirements concerning a structural elements being a column is needed, it is neglected partly. The part concerning the minimal length is neglected, the columns at the lower floor will need to carry a big vertical load, so implementing a demand which limits the cross-sectional area could result in no solutions.

The deflection at the top of the building is limited to $\frac{h}{750}$, when the governing load combination is active with h being the height of the building in meters (according to NEN-EN 1990+A1+A1+C2/NB). The following three serviceability limit state requirements are not taken into account in the optimization but should be checked afterwards. The inter-story drift is limited to $\frac{h_s}{300}$, when the governing load combination is active and h_s is the smallest floor height (according to NEN-EN 1990+A1+A1+C2/NB).

The horizontal acceleration, in the longitudinal direction of the wind, of the building should be limited because these can lead to discomfort or an unsafe feeling. The acceleration which is admitted varies between the 0.1 and 0.5 $\mathrm{m/s}^2$, depending on the natural frequency of the building and its function (NTA 4614-3, section 8.5 and 8.6). The graph does determine the value is shown in Figure 2.6.

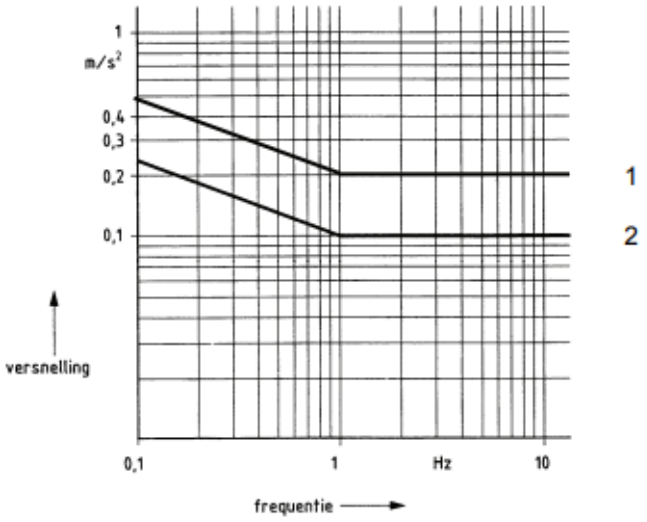


Figure 2.6: Limits of the maximal acceleration of a building, line 1 for work function and line 2 for buildings with an accommodation function (retrieved from NTA 4614-3, section 8.6)

For all types of building functions there is a minimal light value below which illuminance (on average) shall not fall. For example, an office area where people write, type, read and process data the minimal light there must be on average is 500 lx. These requirements can be found, for all types of functions, in NEN-EN 12464-1. The light in a room is produced by artificial light and daylight. It is mostly not accepted that all light is produced by artificial light and there are demands to the minimal amount of daylight which incidence, depending on its function. Bouwbesluit 2012 (paragraph 3.11 daylight) states that the minimal equivalent daylight surface is the maximum of 0.5 m^2 and 10% of the floor surface. In the Bouwbesluit extra requirements to which buildings in the Netherlands has to comply are stated. These values are used to check whether the daylight is sufficient.

2.4. Summary

The goal of this chapter is to answer the following two sub-research questions:

- *"When is a building a high-rise building and what kind of load-bearing structures are there?"*
- *"What are the requirements to a high-rise building, according to the Dutch code?"*

The definition of a high-rise building in this report is:

"A building is classified as high-rise building when it's roof is 70 m or more above ground level and accommodates work and/or living space."

In Paragraph 2.2.1 an overview of the different load-bearing structures and their classification is given. In this research only the necessary will be placed in the core, the core will take an appropriate part of the vertical load. The load-bearing structure at the perimeter of the building will take care of the stability, stiffness and the rest of the load.

The second sub-question is answered in Section 2.3, a list of requirements according to the Dutch code is given. These are based on two assumptions: cross-section is a square and the buckling length is equal to the length of the element. The first four of them are taken into account in the optimization phase, listed in order of importance:

1. In all calculations the building should be considered in consequence class 3;
2. The compression and tension design strengths are $f_{cd} = 0.54 f_{ck}$ and $f_{ctd} = \frac{f_{ctk;0.05}}{1.5}$.
3. The minimal cross-sectional area is the maximal value of 0.0625 m^2, 0.0225 m^2 and $\frac{l_m^2}{57.12}$ m^2 , these are visualised in Figure B.1 and B.2 in Appendix B.3;
4. The deflection at the top of the building is limited to $\frac{1}{750}$ of the height of the building;
5. The inter-story drift is limited to $\frac{1}{300}$ of the floor height;
6. For all types of areas, in a building, there is a minimal light value below which illuminance (on average) shall not fall. This light is composed of daylight and artificial light, thereby there are minimum values for the amount of daylight, which affect the openings in the facade. The minimal equivalent daylight surface is the maximum of 0.5 m^2 and 10% of the floor surface;
7. The horizontal acceleration, in the longitudinal direction of the wind, of the building which is admitted varies between the 0.1 and 0.5 m/s^2, depending on the natural frequency of the building.

3

Optimization methods

In this chapter, the following sub-research question is answered: *"What kind of structural optimization methods exists and which suits the research objective the best?"* Firstly the term optimization is explained in Section 3.1, afterwards the three different structural optimization techniques are discussed in Section 3.2. In Section 3.3 some studies on structural optimization are highlighted. At last, in Section 3.4 a summary of the above is given and the most suitable method for this research is chosen.

3.1. Optimization

Optimization is improving the design to achieve the best way of satisfying the need, with all the available means [55]. In other words, optimization is obtaining the best results, under certain circumstances [34]. The goal is to find a minimum or maximum through tweaking parameters. In mathematics the optimization problem is generally expressed as:

$$\begin{aligned} &\min_x f(x) \\ \text{Subject to: } &h_j(x) = 0, \quad j = 1, \dots, n_h \\ &g_k(x) \leq 0, \quad k = 1, \dots, n_g \\ &x_i^l \leq x_i \leq x_i^u, \quad i = 1, \dots, n_d \end{aligned}$$

with:

$f(x)$	=	objective function
$h_j(x)$	=	equality constrain function
$g_k(x)$	=	inequality constrain function
n_h	=	number of equality constrains
n_g	=	number of inequality constrains
n_d	=	number of design variables
x_i	=	i^{th} design variable
x_i^l	=	lower bound of the design variable x_i
x_i^u	=	upper bound of the design variable x_i

The collection of design alternatives is called the design space. Each design alternative is described with design variables. To find the best design alternative, every point within the design space or each alternative needs to be assessed with an objective or criterion [43]. The objective is minimized with the restriction that the constraints are satisfied, which are written as equality and inequality functions [29, 55]. Concluding, for optimization is needed: design variables, an objective (function), constrains depending on the design variables and possibly restrictions for the design variables.

Optimization is done for all sorts of problems in all sorts of working fields. For example, your navigation system finds the shortest or fastest route with constrains as avoiding tollways

and maximum speed. The best result is found through minimizing the time or distance. Another example, in the design of packaging products, optimization is used to find the design which uses the least material but still doesn't fail on strength or stability. Also, aerospace engineers use optimization to minimize the wind resistance of the aeroplane or to minimize the weight [34]. Thus, there are enormous possibilities for optimization in all work fields. Civil engineers also use optimization, e.g. for finding the stiffest or lightest structure, this is called structural optimization.

Structural optimization is the search for an optimal structure to carry a certain load while satisfying various constraints [43] or optimization techniques applied to structures [38]. Structural optimization can be divided into three classes, depending on geometric features: sizing, shape and topology optimization [2, 18, 32, 38, 43, 61, 64] and is discussed in Section 3.2.

3.2. Structural optimization

The three optimization classes, sizing, shape and topology, are explained in respectively Paragraph 3.2.1, 3.2.2 and 3.2.3.

3.2.1. Sizing optimization

Sizing or size optimization is a form of structural optimization where the geometry and form of the structure, boundary conditions and location of the members are fixed [2]. The optimization method optimizes design variables based on fixed conditions. An example of these design variables is some type of structural thickness, i.e. cross-sectional areas of structural members or the thickness distribution along a sheet [18]. Also, a set of design parameters of a beam cross-section can be the design variable, such as the moment of inertia (different directions), shear modulus, bending modulus, or torsion modulus, etc [64]. In Figure 3.1 an example of sizing optimization is shown. Here a simply supported truss structure is optimized. The structure after size optimization, shown on the right of Figure 3.1, has certain trusses which are enlarged (bold in Figure 3.2.1). Thus according to the optimization method, these trusses need to transfer the most force and are thus enlarged. Concluding, after sizing optimization the size of elements are changed such that the material is used efficiently while satisfying all prescribed conditions.

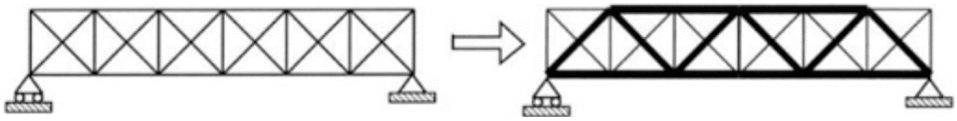


Figure 3.1: Before (left) and after (right) size optimization [9]

3.2.2. Shape optimization

Shape optimization (geometry optimization) has more freedom than sizing optimization, the geometry of the structure isn't fixed [2]. Shape optimization has the freedom to make changes to the boundaries of the design [41]. It optimizes the shapes of structural boundaries with a fixed topology and type. A continuum structure usually has boundaries which are described by geometrical curve such as a line, spline or arc. Geometrical curves have parameters, which can be adjusted such that the structural boundaries are changed. An example, the optimization of a simply supported beam with six holes, is shown in Figure 3.2. Inside the domain the material is reshaped, i.e. reshaping of the structural boundaries, while maintaining its topological properties [28]. In the case of the simply supported beam, it means that the shape of the six holes are adapted.

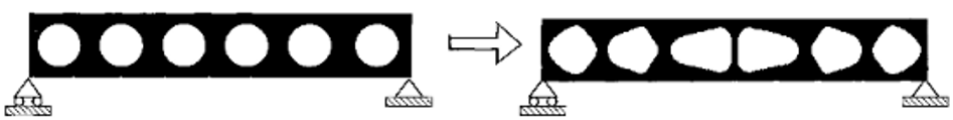


Figure 3.2: Before (left) and after shape (right) optimization (right) [9]

3.2.3. Topology optimization

Topology optimization searches for the optimal distribution of material within the structure. It is used to find a structural configuration that meets predefined criteria, such as constraints (on the stress, displacement, etc.), boundary conditions and loads [18]. Commonly, the process starts with a design space, a chunk of material, which consists of a large number of elements. Topology optimization method removes elements from the domain or translocate elements within the domain [34]. The solutions of the optimization can have any connectivity, shape or size [6]. Topology optimization combines the size and shape methods and adds something extra. In the case of the example in Figure 3.2, topology optimization finds the optimal number of holes in the beam, the optimal shape of these holes and the optimal area of the beam [61]. Topology optimization searches for the optimal design by determining the best locations and geometries of cavities in the design domains [18, 32]. In Figure 3.3 this is visualized, it shows that after optimization the material in the beam is significantly reduced and shows similarities with Figure 3.1. The main difference between the two methods is that the topology was fixed in size optimization and in topology optimization this topology was found as the best design alternatives (in this particular case).

Although this is the optimal solution, this is hard to build, due to the irregular shape of the beams. Often after topology optimization the found structure needs to be refined to more standard elements, thus the method is a good first estimation of where structural elements need to be located. In the future, through advances in 3d-printing of concrete and steel, this problem could be eliminated [13].

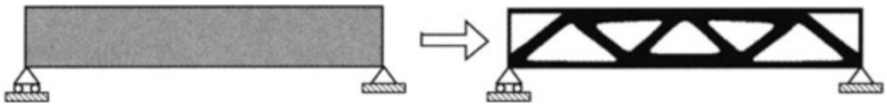


Figure 3.3: Before (left) and after (right) topology optimization (right) [9]

The classic approach in topology optimization is the homogenization approach in which a structural element is only defined by its volume, the loads acting on the element and design requirements (e.g. limitations on stress and/or strain). The initial design in the iterative design optimization procedure is a rough block of space in which we fill material in an optimal way (or we have a rough block of material and remove material) [8]. Another early form of topology optimization is the ground structure approach (GSM), where the design space is divided into nodes connected by trusses, which are sized to an optimal shape.

Currently, the most known methods of topology optimization are the solid isotropic material with penalization (SIMP) and the (bi-)evolutionary structural optimization ((B)ESO) [39, 65]. Next to these two methods there are more (less known) methods, each method has variations (adaption for certain cases or improvements) thus many more exist. A description of these two and the ground structure method (GSM) are given in the next paragraphs.

GSM

GSM (ground structure method) is proposed by Dorn et al. in 1964 [20]. GSM works with a design space consisting of nodes. Originally the initial ground structure is constructed by creating members between all nodes (also called full connectivity). The cross-sectional area of the bars are the design variables in the optimization and are constant over the length of one member. Depending on the location of the supports and the acting points of the loads, the numerical values for the variables are sought after. An option is that the variables turn to zero, thus elements are deleted, saving material. Originally this method start with the full connectivity as initial ground structure (shown on the left in Figure 3.4), deletes elements and calculates the new load distribution until a threshold is reached (e.g. the number of elements).

GSM is a method which divides the optimization problem into multiple size optimization problems, which combined give an optimized structure. This implies that the truss topology problem can be viewed as a standard sizing problem [9]. GSM finds an optimal solution, but the quality of the solution depends on the initial ground structure, the location of the nodes

and the connectivity of the bars [27]. An example is shown in Figure 3.4. Often this method is used for skeletal structures, such as a truss structure.

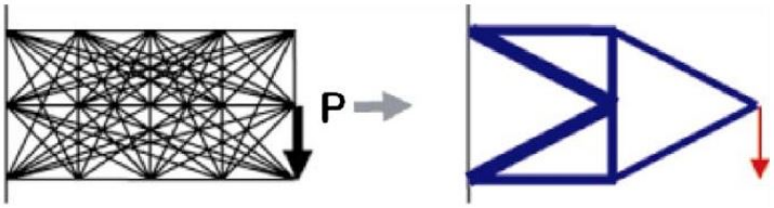


Figure 3.4: An example of a cantilever optimized by the Ground Structure Method. On the left the the nodes and all possible members, on the right the solution with each member optimized [67]

SIMP

SIMP (Solid Isotropic Microstructures with Penalization), a density-based approach. Usually, a density-based method divides the design space in 2D pixels (or 3D voxels) and to each pixel a variable is assigned. These variables can take the value one if the pixel must be filled with material and zero otherwise (discrete, either zero or one). The decision, if a pixel needs to be filled or not, depends on the solution of the optimization problem. The SIMP method is different than the normal density-based method because it works with continuous variables instead of discrete variables. The variable can be zero, one and any value between zero and one. The benefit is that the optimization problem is solved faster. But, translating this problem to reality is harder. If a variable is unequal to zero or one, it is not clear if there must be material or not. This problem is solved through penalizing the values unequal to zero or one, such that they converge to either one or zero. [7, 59]. An example is shown in Figure 3.5. Due to the use of continuous variables the faded edges occur because they have a value between zero and one.

(B)ESO

ESO (Evolutionary Structural Optimization) removes material from a given domain to find the optimal structure under applied loading and boundary conditions. This is an iterative process, in each step inefficient material is removed. The material is marked as inefficient if a threshold is not exceeded (for example a stress limit) [43, 60]. Note that according to the research of Zhou, the ESO method can produce non-optimal solutions to an optimization problem [70]. The Bi-Evolutionary Structural Optimization (BESO) does not only remove material, but can also add material. Therefore it can start with a minimal amount of material in contrast to ESO which uses an initially oversized structure [68].

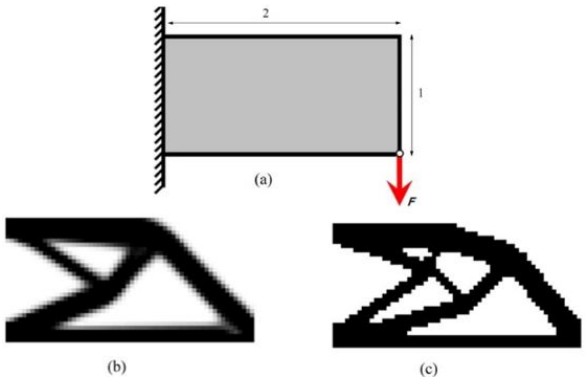


Figure 3.5: An example of a cantilever (a) optimized by the SIMP (b) en ESO (c) method [1]

For more topology optimization methods: Beghini's article describes shortly which types of topology optimization methods there are [6], E. Tyflopoulos et al. gives a detailed overview, with their strengths and weaknesses summarised in a Table [65].

3.3. Research in topology optimization

There has been extensive research in topology optimization, in this section some research is shortly discussed. There is a distinction made between two types of research, the research in improving topology optimization and the research in the application of topology optimization. Eschenauer describes a good overview of different researches in the field of improving topology optimization. There has been research in implementing, among others, non-linearly elastic materials, design depending loads, dynamic loads, multiple load cases, buckling, stiffness constraints, frequency constrains and manufacturing-type in topology optimization [22, 63, 64, 68].

A few examples for research in the application of topology optimization are, the minimum cost optimization of the northeast tower at Hong Kong station, the minimization of the number of bracing members in The Pinnacle Tower in London or the minimum or a cost minimization in the design of the 1 Dubai Tower in Dubai [2]. Beghini et al. describe in their paper how topology optimization combines architecture and engineering, through presenting case studies and concludes that topology optimization results in designs which embrace the structural engineering together with the architecture to create innovative aesthetically pleasing structures with evident structural engineering components [6]. Thus implementing the optimization in the design has benefits, but Kingman et al. have indicated in their paper that there are barriers for widespread implementation of topology optimization, namely the complex geometry of the optimized designs and the difficulty in solving problems involving nonlinear behaviour (such as buckling) and dynamics [36].

3.4. Summary

The goal of this chapter is to answer the following sub-research question: *"What kind of structural optimization methods are there and which suits the research objective the best?"*. In short, the following distinction between structural optimization methods are made:

- Sizing optimization;
- Shape optimization;
- Topology optimization.

The sizing optimization method is useful for optimization of a member of a load-bearing structure. Shape, or geometrical, optimization does change the shape but is limited to a certain topology through which it is not able to find the location of load-bearing members in the structure. According to Huang, topology optimization provides much more freedom, compared to the other two, and allows the designer to create totally novel and highly efficient conceptual designs. Topology optimization has more undetermined parameters, therefore more freedom, and the topological parameters have more influence on the optimization objective. Hence, more rewarding economically in comparison with sizing and shape optimization [32, 64]. A drawback of more undetermined parameters is the computational time, which is longer than for the other optimization methods, this should be kept in mind when searching for the optimal solution for the problem. All things considered, for gaining more insight into the added value of structural optimization, topology optimization is the most suitable. Topology optimization has the most freedom, therefore it's more likely to find a novel structure which minimizes the material use as much as possible.

Within topology optimization there are different methods, in this research the Ground Structure Method will be used. GSM is a method which divides the optimization problem into multiple size optimization problems, which combined give an optimized structure. In the end the load-bearing structure will consist of columns, braces and beams, which are located at the perimeter of the floor plan. When using methods, such as SIMP and ESO, which work with pixels or voxels, the solution is often quite complex. The problem afterwards is composing a design with buildable columns, beams and braces. This post-processing will be less hard when using the ground structure method because this method sizes the beams, braces and columns.

In Section 3.3 an overview of the literature is given, which indicates the research there is done in the use of topology optimization and the improvement op the topology optimization method. As mentioned before, in Paragraph 1.2.2, to perform the optimization Python is used. Abaqus works well with Python, It is possible to script an analysis in Python and execute this in Abaqus. Abaqus is mainly chosen because of practical reasons and for the good collaboration with Python.

4

Case Study

In this chapter the case study, Boston & Seattle, is discussed. The information about and images of the case study needed for this Chapter are obtained via Zonneveld Ingenieurs. In Section 4.1 the building is discussed: location, building method, floor plan, etc. Characteristics of the building; height, width, length, floor heights, etc. are summed up in Section 4.2, answering the sub-research question: *"What are the characteristics of the case study regarding the structure?"*. These characteristics are used as constraints in the optimization process and as boundary conditions of the design spaces in Chapter 5. At last, a summary is given, in Section 4.3, and an answer to the sub-research question: *"Which boundary conditions, for the optimization problem, follow from the case study?"* is given.

4.1. General information

The case study, Boston and Seattle, is situated at the Otto Reuchlinweg in Rotterdam, on the Wilhelminapier and are named after the former warehouses which were located here. The total height of the building (relative to the ground) is 74 m. The building consists of two towers, each containing 110 apartments connected by the part containing the parking and commercial spaces. As can be seen in Figure 4.1, Boston & Seattle is one of the lowest building compared to the other high-rise buildings on the Wilhelminapier.

Figure 4.1: Boston & Seattle

The building has 23 floors above ground level and 2 floors below ground. The two floors below ground are public parking spaces, on the 5^{th} till 23^{th} floor apartments are located. On the other floors are shops, restaurants and commercial spaces located. The construction is completed in September 2017 and took approximately 3 years. The tower on the left in Figure 4.1 is named Seattle and the mirrored version at the right is named Boston. The building is partly built with prefab elements and a part of the elements is poured on-site.

4.2. Characteristics

In Figure 4.3 a simplified impression of the floor plan (without internal walls) of the 11^{th} floor of the tower Seattle is shown. The figure shows the main dimensions of the building and the shape of the floor plan. In Figure 4.2 a 3D view of the floor plan (including internal walls) is shown. The widest part of the building is 24.9 m and the longest part is 36.90 m. On all four sides there are prefab balconies placed, at the notches in the floor plan (located at line 5 and between line D and E). Between line C and D the stairs are located in a core, of the size 3600 x 3000 mm and a second core between line D and E, for the elevators, is 4660 x 2690 mm. From the 6^{th} floor the outside dimensions of the building and the size of the core is the same (ignoring minor differences on the top levels). In Appendix C the 3D floor plans of the tower Seattle from the 6^{th} floor and higher are presented. The apartments in the lower floors are adapted, due to the rooftop of the commercial areas. As mentioned before the towers are mirrored versions of each other, so optimizing one of the two towers is sufficient to find the optimal design for both. Also, the building below the 6^{th} floor, which connects the two towers, has a floor plan and dimensions which comply with the demands for the commercial areas. Therefore during the optimization the lower part will not be considered and only the towers are investigated. As concluded before, it is not necessary to investigate both towers, only one is sufficient. Each floor from the 4^{th} up has a floor height of 3 m and an approximately floor surface of 670 m^2. The 6^{th} floor starts at 20.3 m above ground level and the roof of the building is at 74.3 m, thus the total height of the tower which will be investigated is 54 m. In the following two paragraphs the load-bearing structure and the loads are discussed.

4.2.1. Load-bearing structure

A 3D view of the 11^{th} floor of the Seattle tower is shown in Figure 4.2. All dark grey elements in the figure are prefab elements and all light grey elements are in-situ elements. The figure shows that all elements at the perimeter of the building are prefab elements.

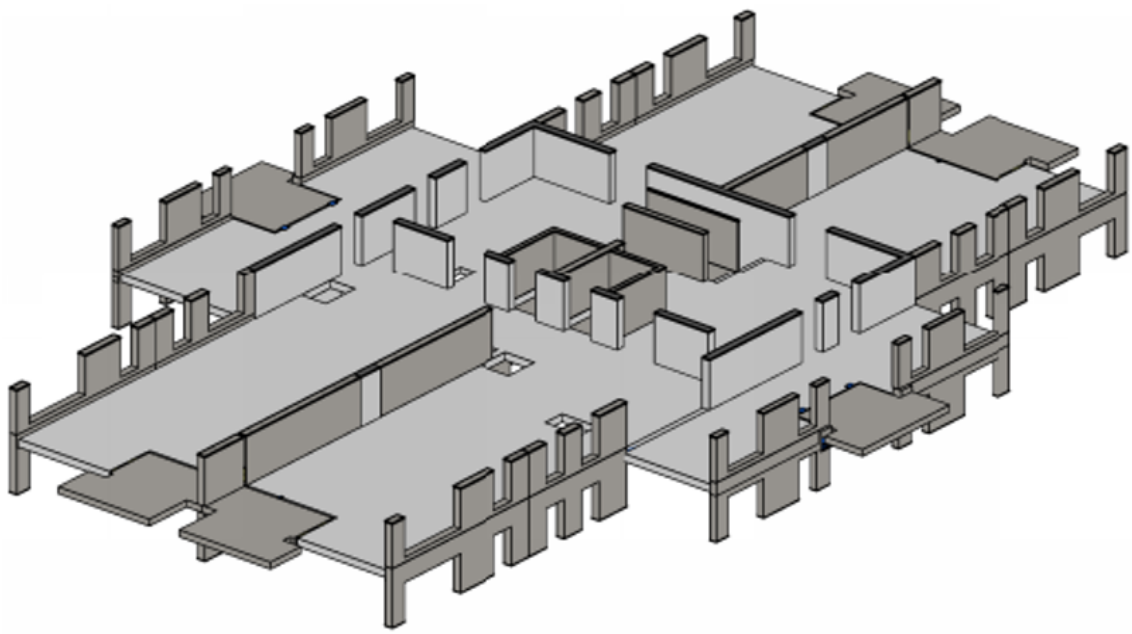


Figure 4.2: 3D view on the 11^{th} floor of tower Seattle

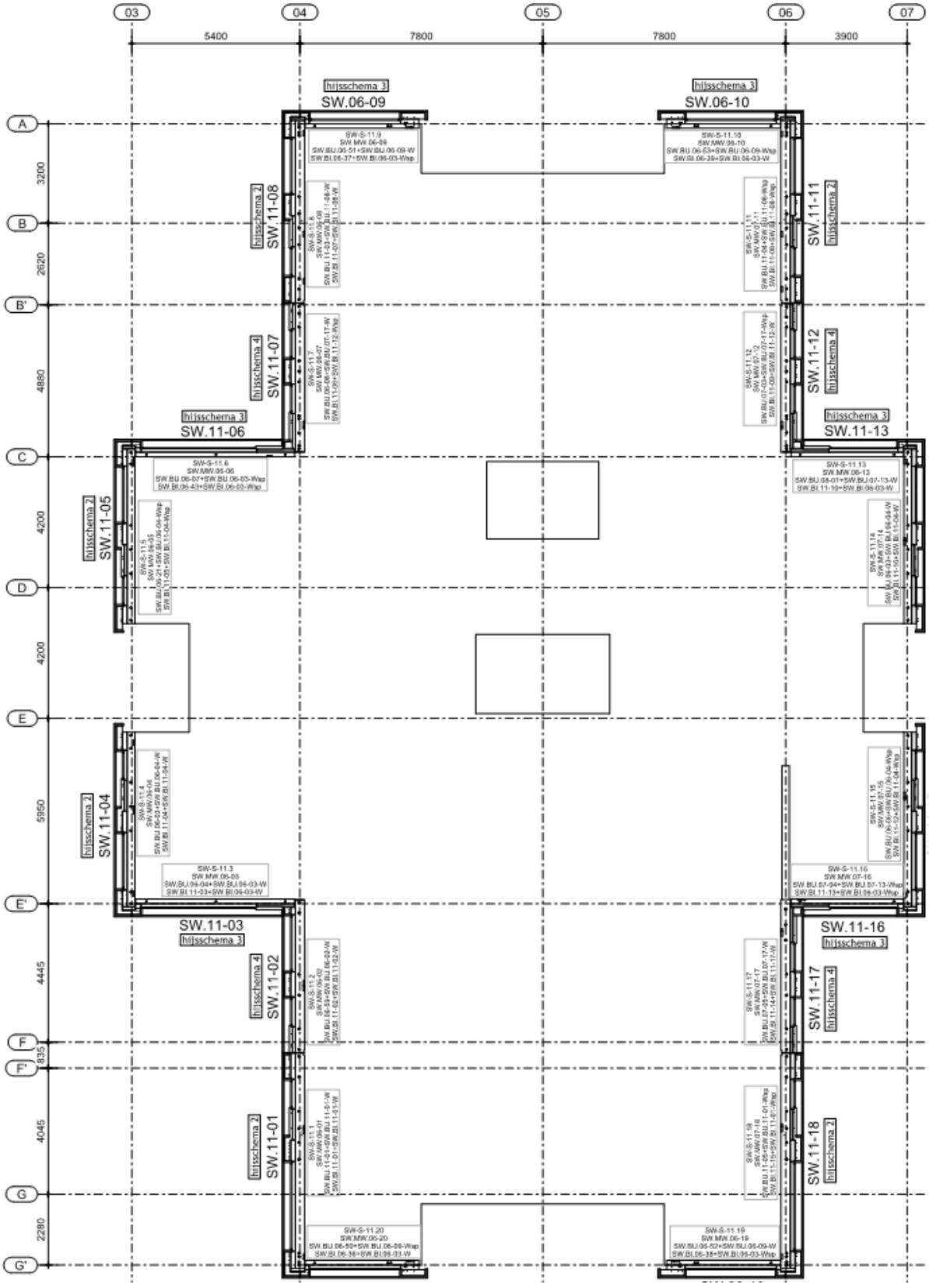


Figure 4.3: Simple floor plan of the 11^{th} floor of tower Seattle without internal elements, dimensions in millimetres

The main load-bearing structure consist of (punched) load-bearing walls divided over the floor plan, which acts as shear walls for stability. The load-bearing walls are placed in different parts of the floor plan; centrally around the stairs and elevators, inside the building functioning as room dividers and at the perimeter. These walls transfer the lateral and vertical load to the lower floors. From the 5^{th} floor and lower the load is transferred by load-bearing walls and columns to the foundation. The lower floors have a more open layout through the use of columns. This is practical for the commercial areas which are located here.

For the different structural elements different type of concrete is used, the elements and their corresponding concrete type is listed below:

- Prefab walls/columns are made of C45/55;
- Columns and beams are made of C55/67;
- The in-situ walls on the 8^{th} up to and including 23^{th} floor are made of C35/45;
- The in-situ walls on the -2^{th} up to and including 7^{th} floor are made of C55/67;
- The prefab floor is made of C35/45;
- The prefab balconies are made of C55/67.

4.2.2. Loads

In Table 4.1 the permanent and variable loads of each component is given. The parapet, front, side and end facade act over the whole perimeter of the building except for the location of the balcony. The balcony load only acts on the location where they are connected to the building.

Table 4.1: Vertical load (per square meter), including the self-weight of the floor, balcony and load-bearing elements in the side facade

	Perm. [kN/m^2]	Var. [kN/m^2]
Floor	8.50	1.50
Balcony	6.50	1.25
Internal walls	7.50	0.00
Side facade	11.20	0.00
End facade	5.30	0.00
Front	1.00	0.00
Parapet	5.17	0.00

As mentioned in Paragraph 4.2.1 and shown in Appendix C, there are some minor difference in the floor plans in the towers. This results in vertical loads which differ from floor to floor. For the 6^{th}, $7^{th}-21^{th}$, 22^{th}, 23^{th} floor and the roof, the loads are given in respectively Table F.1 till F.5 in Appendix F. The permanent load of the floor and balcony are including self-weight, the permanent load of the side facade consist of the weight of the load-bearing structure in the side facade and the finishing.

The horizontal load (wind) depends on the height and the direction of the wind. The most unfavourable direction is perpendicular on the facade. The wind can act on the long or short side of the building. The loads are taken from the case study and are shown in Table 4.2. These do included pressure of the wind on the facade and friction on the sides of the building. It is assumed that the wind load, acting on the facade, is redirected by the floors to the facades, therefore the wind loads are given per meter width floor.

Table 4.2: Wind load per square meter (including friction on the side-facade) on the long (parallel to grid line E) or short side (parallel to grid line 05), grid lines in Figure 4.3

	Load[kN/m^2]		Line load on floor [kN/m]	
Floor	Short side	Long side	Short side	Long side
6	1.83	1.93	5.50	5.80
7	1.83	1.93	5.50	5.80
8	1.83	1.93	5.50	5.80
9	1.83	1.93	5.50	5.80
10	1.97	1.93	5.90	5.80
11	2.10	1.93	6.30	5.80
12	2.23	1.93	6.70	5.80
13	2.37	1.93	7.10	5.80
14	2.37	2.20	7.10	6.60
15	2.37	2.20	7.10	6.60
16	2.37	2.20	7.10	6.60
17	2.37	2.20	7.10	6.60
18	2.37	2.20	7.10	6.60
19	2.37	2.20	7.10	6.60
20	2.37	2.20	7.10	6.60
21	2.37	2.20	7.10	6.60
22	2.37	2.20	7.10	6.60
23	2.43	2.20	7.30	6.60
roof	1.23	1.10	3.70	3.30

4.3. Summary

The information about Bosten & Seattle is obtained via Zonneveld Ingenieurs. The following two sub-research questions are answered in this chapter:

- *"What are the characteristics of the case study regarding the structure?"*
- *"Which boundary conditions, for the optimization problem, follow from the case study?"*

The main load-bearing structure consist of (punched) load-bearing walls divided over the floor plan, which acts as shear walls for stability. The load-bearing walls are placed in different parts of the floor plan; centrally around the stairs and elevators, inside the building functioning as room dividers and at the perimeter. These walls transfer the lateral and vertical load to the lower floors. This only holds for the two residential towers above the commercial areas, which are mirrored version of each other. During the optimization of the case study, only the residential towers are optimized. All elements at the perimeter of the building are prefab elements of concrete class C45/55, therefore the characteristics of this concrete class are used during the optimizations. The loads on the case study are shown in Table 4.1 and 4.2. The floor plan of the case study is shown in Figure 4.3 and the height of the tower (54 m).

5

Optimization code

In this chapter, the mathematical description of the optimization is derived, the structure of the Python code is explained, the Python code is verified and the principles of the optimization are stated. The different aspects are discussed in the following sections:

- In Section 5.1 the design variables, objective function and constraints for the optimization are discussed;
- In Section 5.2 the working of the Python code is explained;
- In Section 5.3 the code is verified;
- In Section 5.4 the parameters, loads, supports and design space for the case study are specified.

These sections combined answer the following two sub-research question:

- *"How can the optimization problem, for different design spaces, be defined?"*
- *"Does the optimization code work and what is the input for the optimizations?"*

As described in Chapter 4, the main load-bearing structure of the towers consist of punched load-bearing walls divided over the floor plan. These load-bearing walls act as shear walls for stability. The GSM uses a combination of a diagrid and braced tube system, both shown in Figure 2.2. Thus the punched shear walls at the perimeter are redesigned during the optimization, the other load-bearing elements do not change. Through using a different manner to design the load-bearing structure at the perimeter, new insights might occur, material might be saved and/or aesthetics of the building might improve.

Preferably all constraints are implemented in one optimization problem. A disadvantage of this is that the optimization problem becomes too complex or the solution space might become too limited. To control the optimization problem and prevent this, the problem is split into two parts. The goal of the first part is to find the optimal size of each member depending on the compression and tension strength (this might mean that a member has no size, therefore doesn't exist). The second part resizes these members such that the top displacement of the building is limited to $\frac{1}{750}$ of the height of the building, as defined in Section 2.3. This method is a recursive resizing algorithm, each iteration step the variables are resized with the help of information from the previous step and the sensibility of the constraint to change of the variable. The mathematics of both steps are explained in detail in Section 5.1. Often post-processing with simplification is needed to be able to actually build the found structure. Post-processing options are for example: (small) structural elements can be combined or the simplification to the optimization can be made to clarify the results.

5.1. Mathematical description code

In Section 2.3 all requirements, taken into consideration for this research, are listed on importance. These need to be converted to constraints, such that they can be implemented in the optimization process. In the first part of the optimization, the maximum tension and compression stresses, the minimal values for the cross-sectional area are taken into account. This part is also called the ”strength” optimization. The deflection at the top of the building is the constraint in the second part of the optimization, called ”stiffness” optimization. The constraints concerning displacement of the top of the building, inter-storey drift and maximum horizontal acceleration should be checked afterwards (for example with the use of a FEM-model). During the optimization, a negative internal force is compression and a positive internal force is tension. In Figure 5.1 the used coordinate system is shown wherein the XY-plane parallel is to the floors of the building. The degree of freedoms are translation in the x-, y- and z-direction.

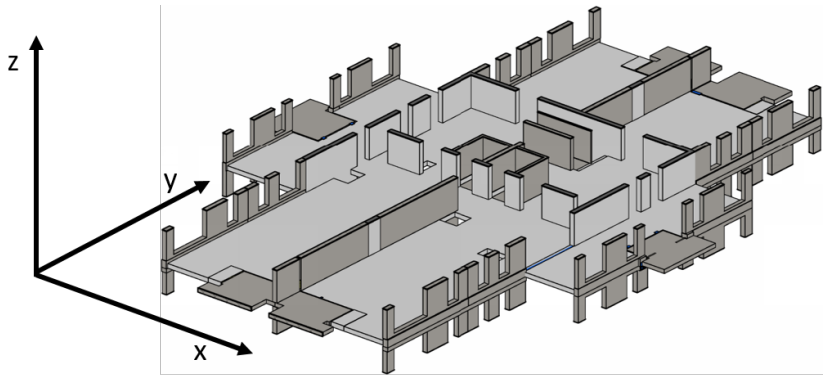


Figure 5.1: 3D coordinate system

5.1.1. Optimal member sizing and distribution

The mathematics of this paragraph are based on the principles discussed in the article of He et al. [31]. The goal of this strength optimization is finding the optimal distribution of the members over the design space, their cross-section area and their corresponding internal force, such that as little as possible material is used. In Section 2.3 it is assumed that the buckling lengths of the individual members are equal to the length of each member. This implies that all members are connected with hinges. In the Ground Structure Method this is the case, all structural elements are connected with hinges, thus the assumption is acceptable.

The first part of the optimization has two variables: the internal forces in the members and the cross-sectional areas of the members. The internal forces in the members are denoted in vector q, this vector contains the internal force for all m members. The cross-sectional area of the members are stored in the vector a and has, similar to q, m entries.

$$q = \left[q_1; q_2; \dots; q_m\right]^T$$
$$a = \left[a_1; a_2; \dots; a_m\right]^T$$

The objective of the optimization is to minimize the use of material, thus to minimize the volume of the material. The volume of the material is equal to the summation of the volume of each member, or the dot product of the transposed vector containing all lengths (l) and the vector with the cross-section areas of the members. The vector l has the same shape and amount of entries as the vector q and a.

$$f(a) = \sum_{i=1}^{m} l_i a_i = l^T \cdot a \tag{5.1}$$

The constraints mentioned in Section 2.3 are rewritten, such that the optimization problem can be written as in Section 3.1. The tension and compression strength constraints are satisfied, by limiting the internal force (including flexural buckling).

$$q_i \geq -0.54 f_{\text{ck}} \cdot a_i \tag{5.2}$$

$$q_i \leq \frac{f_{\text{ctk; 0.05}}}{1.5} \cdot a_i \tag{5.3}$$

These equations are only valid if the values for a_i, f_{ck} and $f_{\text{ctk; 0.05}}$ are greater or equal to zero. f_{ck} and $f_{\text{ctk; 0.05}}$ are properties of a certain type of material, thus always larger then zero. As derived in Section 2.3, there is a minimal surface-area:

$$a_i \geq \max\left(0.0625; \frac{l_i^2}{57.12}\right) \tag{5.4}$$

At last, to solve the problem, the equilibrium between the internal forces and the external forces is used. The external forces act at the nodes and the internal forces act in the members and need to be converted to loads on the nodes. Where the external forces work on the nodes in three main directions (X, Y and Z), the internal forces can be inclined. These need to be converted to the X-, Y- and Z-direction such that they can be equalized with the external forces. This conversion of the internal force is done by an equilibrium matrix **B**. The matrix **B** has the size $3n\ x\ m$, with n the amount of nodes. The matrix has $3n$ rows because each node has an x-, y- and z-component. Note that this holds that there are no moments and shear forces included in this optimization, only normal forces.

$$\mathbf{B} \cdot q = f \tag{5.5}$$

With vector f, containing all external force information, of the size $2n$ x 1, which is predefined before optimization starts. Combining the objective function and all constraint, described in equation 5.1 - 5.5, in the format described in Section 3.1 the strength optimization problem is defined.

$$\begin{aligned}
&\min_{q,a} \quad l^T \cdot a \\
\text{Subject to: } &\mathbf{B} \cdot q = f \\
&a_i \geq \max\left(0.0625; \frac{l_i^2}{57.12}\right), \quad i = 1 \dots m \\
&q_i \geq -0.54 f_{\text{ck}} \cdot a_i \\
&q_i \leq \frac{f_{\text{ctk; 0.05}}}{1.5} \cdot a_i
\end{aligned}$$

The constraints to this optimization problem can be extended with multiple load cases. Also, it is possible to take the self-weight as external force into account. In all load cases the constraints concerning the force equilibrium and tension and compression strength must be satisfied, thus these must be checked for each load case. Different load cases are denoted with the superscript k, which could have a value between 1 and p, with p as the total number of load cases.

The load due to the self-weight of a member can be calculated through multiplying the mass of the member with the gravity acceleration (denoted as g). The load is divided evenly over the two nodes where the member is connected to. The force vector is needed as input for the optimization, thus the self-weight is calculated after the optimization and taken into account in the next iteration step.

$$f_i^{\mathrm{sw}} = \rho g a_i l_i \tag{5.6}$$

The mass is calculated through multiplying the volume with the density of reinforced concrete (denoted as ρ). For each node the loads through the self-weight of the members are summed up and inserted in the vector f^{sw}. The force due to the self-weight works in the negative z-direction. The vector f^{sw} is of the same size as the external force vector f ($2n$ x 1) and is added to the force equilibrium. The characteristic strength divided by their material factor is the design strength, for the compression f_{cd} and for tension f_{ctd}. The optimization problem can be further specified by implementing this information.

$$\begin{aligned}
&\min_{q,a} \quad l^T \cdot a \\
\text{Subject to: } &\mathbf{B} \cdot q^k = f^k + f^{\mathrm{sw}}, \quad k = 1 \dots p \\
&a_i \geq \max\left(0.0625; \frac{l_i^2}{57.12}\right), \quad i = 1 \dots m \\
&q_i \geq -0.81 f_{\mathrm{cd}} \cdot a_i \\
&q_i \leq f_{\mathrm{ctd}} \cdot a_i
\end{aligned}$$

5.1.2. Recursive Resizing Algorithm

The stiffness optimization, wherefore a recursive resizing algorithm is used, is based on the article of Chan [17]. The stiffness optimization has only one variable: the cross-sectional area of the members. The amount of members, their location and their size together with the internal forces is the solution to the strength optimization and is input for the Recursive Resizing Algorithm (RRA). The objective function is still the same, minimizing the volume, Equation 5.1. The constraint in the RRA is that the top displacement of the building should be less or equal to $\frac{1}{750}$ of the height, as mentioned in Section 2.3. The constraints for the maximum stresses and minimal size are already taken into account in strength optimization. Through using the results of the strength optimization as input for the stiffness optimization, the constraints of that optimization are already satisfied. This only holds if the cross-sectional area of each member is larger then the cross-sectional area found in the strength optimization. Note that the increase in self-weight, through an increase in cross-sectional area, is not taken into account in this part of the optimization. The increase in volume of the total structure results in an increase the load through self-weight, therefore the minimum and maximum stresses should be checked again after this stiffness optimization.

$$\begin{aligned}
&\min_{a} \quad \sum_{i=1}^{m} l_i a_i \\
\text{Subject to: } &a_i \geq a_{i,1} \\
&\left| \sum_{i=1}^{m} q_{\mathrm{u},i} \frac{q_i l_i}{E a_i} \right| \leq \frac{h}{750}
\end{aligned} \tag{5.7}$$

with:

$a_{i,1}$ = Cross-sectional area of member i as result of optimization part one
E = Young's modulus of concrete
$q_{\mathrm{u},i}$ = Force in member i, caused by a unit load at the point of interest

To find the displacement of the top of the building, the virtual work method for truss structures is used. The virtual work method states that the summation of all members elongation or shortening multiplied with the force in the member, due to a unit load at the point of interest, is equal to the displacement in the point of interest [48]. The point of interest

is the point where the user is interested in the magnitude of the displacement (often the location where the biggest displacement is expected). Therefore the constrained minimization problem is formulated and shown in Equation 5.7.

The method to find the recursive resizing algorithm is based on the article of Chan [17], with background information from the book of Boyd and Vandenberghe [12]. The optimal criteria for the constrained optimization problem can be obtained indirectly by transforming the constrained problem to an unconstrained Lagrangian function, temporarily omitting the minimal value of the design variable. The unconstrained Lagrangian function which involves the objective function and the deflection constraint associated with a corresponding Lagrangian multiplier (z_1) is as follows:

$$\mathcal{L}(a_i) = \sum_{i=1}^{m} l_i a_i + z_1 \left(\left| \sum_{i=1}^{m} q_{\mathrm{u},i} \frac{q_i l_i}{E a_i} \right| - \frac{h}{750} \right) \tag{5.8}$$

By differentiating the Lagrangian function to the design variable (a_i) and setting the derivative to zero (see Appendix D), the necessary stationary optimally conditions can be obtained. These can be used to find the recursive relation to resize the design variable. The Lagrangian multiplier (z_1) can be interpreted as a sensitivity factor that measures the importance of the corresponding constraint to the optimal design.

$$\frac{\partial \mathcal{L}(a_i)}{\partial a_i} = l_i - z_1 \frac{g^t}{|g^t|} q_{\mathrm{u},i} \frac{q_i l_i}{E a_i^2} = 0 \tag{5.9}$$

$$\text{with} \quad g^t = \sum_{i=1}^{m} q_{\mathrm{u},i} \frac{q_i l_i}{E a_i}$$

$$\frac{\partial \mathcal{L}(a_i)}{\partial a_i} = z_1 \frac{g^t}{|g^t|} q_{\mathrm{u},i} \frac{q_i}{E a_i^2} - 1 = 0 \tag{5.10}$$

The found equation 5.10 is a stationary solution to the unconstrained Lagrangian function, this can be used to derive a recursive relation for the design variable. The following recursive resizing equation is found:

$$a_i^{t+1} = a_i^t \left(1 + \frac{1}{\eta} \left(z_1 \frac{g^t}{|g^t|} q_{\mathrm{u},i} \frac{q_i}{E a_i^2} - 1 \right)_t \right) \tag{5.11}$$

Where η a relaxation parameter is, which can be adapted to control the rate of convergence, the t stands for the iteration step and t+1 is the next iteration step. This equation holds only if the value of the design variable is bigger than the found value in the first optimization step, $a_i \geq a_{i,1}$. To use this recursive resizing equation, first the Lagrangian multiplier need to be calculated, which is done through considering the sensitivity of the constraint due to changes in the design variable.

$$g^{\mathrm{u}} - |g^t| = \sum_{i=1}^{m} \frac{\mathrm{d}|g^t|}{\mathrm{d}a_i} (a_i^{t+1} - a_i^t) \tag{5.12}$$

with:

g^{u} = Right-hand side of the inequality constraint, $\frac{h}{750}$
g^t = Left-hand side of the inequality constraint at time step t, $\sum q_{u,i} \frac{q_i l_i}{E a_i}$ with: $i = 1 \ldots m$

All elements of the equation are known, thus a solution for z_1 in time step t can be found. All values below are the values from time step t. Equation 5.11 is filled in Equation 5.12 and written out in Equation 5.13.

$$\frac{h}{750} - |g^t| = \sum_{i=1}^{m} \left[-\frac{g^t}{|g^t|} q_{\mathrm{u},i} \frac{q_i l_i}{E a_i^2} \right] \cdot \frac{a_i}{\eta} \cdot \left[z_1 \frac{g^t}{|g^t|} q_{\mathrm{u},i} \frac{q_i}{E a_i^2} - 1 \right] \tag{5.13}$$

Concluding, when Equation 5.13 is rewritten, z_1 is found.

$$z_1 = \frac{\eta \left(|g^t| - \frac{h}{750} \right) + \Sigma \left[\frac{g^t}{|g^t|} q_{u,i} \frac{q_i l_i}{E a_i} \right]}{\Sigma \left[\frac{{g^t}^2}{|g^t|^2} q_{u,i}^2 \frac{q_i^2 l_i}{E^2 a_i^3} \right]} \quad \text{with: } i = 1 \dots m \tag{5.14}$$

The Lagrange multiplier (Equation 5.14) is implemented in the recursive resizing equation (Equation 5.11). In this case the Lagrange multiplier should always be positive [17], this needs to be verified in Paragraph 5.3. η is defined by the user and should be tested by trial-and-error. For now the value is chosen as 5, if during optimization turns out that it converges to fast or to slow, this value should be adapted.

5.2. Structure of the code

As mentioned in Paragraph 1.2.2, the optimization code is written in Python. For solving the problem the package CVXPY (version 1.0.25) in combination with MOSEK (version 9.1.10) is used [19, 52]. The code, which is used for the optimization, is adapted from the code written by He et al. [31]. This code uses an adaptive 'member adding' scheme, which is firstly proposed by Gilbert and Tyas [26]. This scheme solves problems faster than the conventional GSM, up to 8 times, and allows to solve large problems (e.g. $>$ 1000000 members) [31]. All units in the code are SI-units: meters, seconds, kilogram and newtons.

The strength optimization, described in Paragraph 5.1.1, is solved by using an adaptive 'member adding' scheme. To be able to understand the code, the adaptive 'member adding' scheme must be understood. A short explanation will be given here, for a more detailed explanation the articles by He et al. and Gilbert and Tyas or the book of Boyd and Vandenberghe are recommended [12, 26, 31]. More information about solving constrained optimization problems and Lagrange multiplier methods can be found in the books of Bertsekas and Ito [10, 33].

The adaptive "member adding" scheme initially starts with only a reduced set of members to solve the optimization problem. The remaining potential members are added in small doses until all potential members satisfy a constraint. The constraint is obtained using the duality principle [26]. It rewrites the original optimization problem, with the help of the Lagrange multiplier method (also used in the RRA), to an unconstrained optimization problem. Solving this unconstrained optimization problem will result in a constraint to which all potential members need to comply to solve the initial optimization problem. In this case it is a limitation on the virtual strain.

At the start of the strength optimization the input is given by the user. The needed input is a design space, tension strength, compression strength, load cases and supports. Also, the constraints and objective function are defined, as determined in Paragraph 5.1.1. The design space is discretized using nodes. All possible members are created, which is called "full connectivity", each node is connected to each node, excluding overlapping lines. An example of "full connectivity" is shown in Figure 5.3. Then an initial set of members is formed, each member which is shorter than a user-defined limit (in the optimizations this limit is set to $\sqrt{2}$) is added to the initial set. The problem is fully defined and the 'member adding' loop starts, the optimization problem is solved with the initial set, which returns information about the design variables a_i and q_i. For all members which are not part of the initial set, the virtual strain is calculated. The top 5 % (user-defined), of the members which violates the limit, is added to the initial set. Then the optimization problem is solved with this newly found 'initial set' and the process described above is repeated until no member violates the limit, thus the optimal solution is found for the first part of the optimization. This is visualised in Figure 5.2.

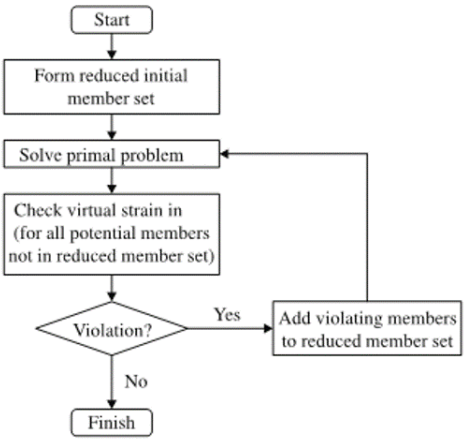


Figure 5.2: Flowchart for the adaptive 'member adding' process [31]

The number of elements which is added after each iteration step, depends on the user-defined (in this research 5 %) percentage pf the number of elements which violates the limit. Thus after each iteration step the number of elements added to the set changes.

The stiffness optimization uses the recursive resizing algorithm. The found optimal structure in the strength optimization combined with the optimization problem, defined in Paragraph 5.1.2, is input for this optimization. This optimization problem resulted in two equations, 5.11 and 5.14. With the help of these two equations each iteration step the design variable, the cross-sectional area of the members, is resized. In each iteration step the displacement of the structure is calculated with the use of the virtual work method and checked that it does not exceed the limit. If it exceeds the limit, the Lagrange multiplier (z_1) and the new (resized) cross-sectional areas are calculated. With the newly found cross-sectional areas the displacement of the structure is calculated. If the new displacement of the structure does exceed the limit, the iteration starts over, the Lagrange multiplier is calculated, etc. This iteration process continues until the constraint concerning the displacement of the top of the building is satisfied. Each time that the new cross-sectional areas are calculated, they must be bigger than the previously cross-sectional area, otherwise the previous value is used.

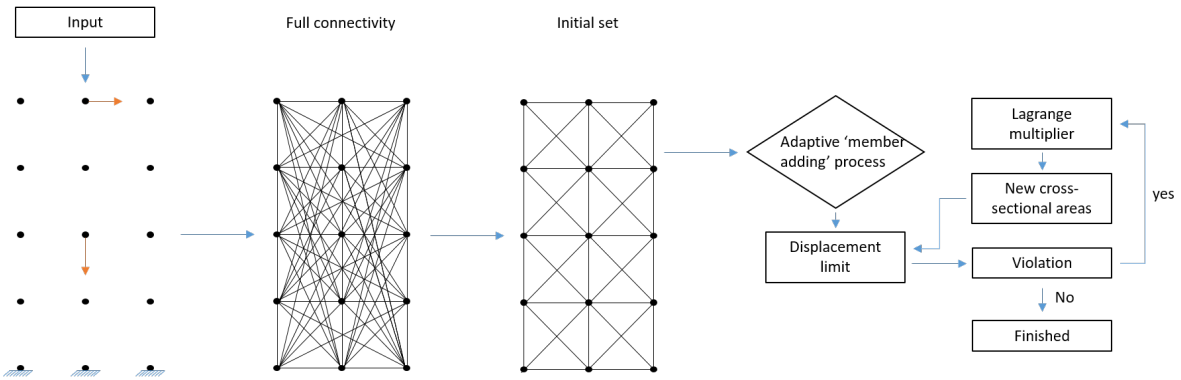


Figure 5.3: Code structure

The diamond with adaptive 'member adding' process in Figure 5.3 represents figure 5.2. During the stiffness optimization, all elements which are in tension are assigned a reduced Young's modulus of 50% of the normal Young's modulus of concrete. In this manner, the reduced stiffness of the cracked concrete is taken into account. It is possible to plot the initial set of members, the forces in each load case, the supports, each iteration step and the optimized structure. Note that plotting takes time, so the speed of the optimization is reduced when plotting all of the mentioned possibilities.

5.3. Verification of the code

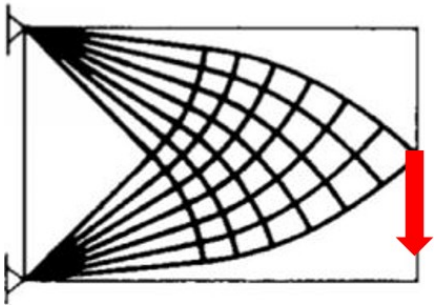
In this paragraph the optimization code is verified, checked whether the boundaries which are imposed during the optimization are satisfied. Also, literature is used to check whether the solution of the optimization code makes sense. All figures have colours indicating whether an element is in compression or tension. Yellow means compression, green means tension and grey means that in different load cases the elements are in tension and in compression. This holds for all figures which are generated for this research, all other figures are retrieved from literature.

All information which is gathered during an optimization is stated in an output file (.xsls-file), the Python script, which generates this file, is shown in Appendix E.5. This output file is used to check whether the imposed boundaries (maximal tension, maximal compression, minimal value for the surface of members) holds during the optimization. An example of this output file is given in Figure E.9 up and till E.15 in Appendix E.5. From these output files it can be concluded that the imposed boundaries are satisfied. In Paragraph 5.1.2 is stated that the Lagrange multiplier should always be positive, during the optimization Python prints for each iteration the Lagrange multiplier so this can be monitored.

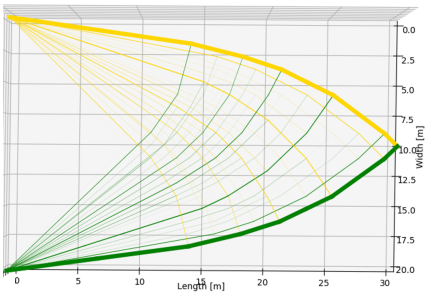
Note that a disadvantage of the 3D figures in Python is that they can only be saved in one viewpoint, not interactive. Thus for the post-processing a script is written to import the .xlsx-file in Python to plot the interactive figures or to post-process and/or view the data in Abaqus, see Section E.6.

5.3.1. 2D cantilever

A known optimization problem is the cantilever beam subjected to a point load in 2D, which results in the Michell truss (shown in Figure 5.4a). In Figure 5.4b the solution generated by the code described in Section 5.1 and 5.2.



(a) Michell truss in the design domain [37, 49]



(b) Solution to the cantilever problem in 2D

Multiple researchers have optimized this 2D cantilever, two examples of this are shown in Appendix E.1 together with an enlarged Figure 5.4b. This shows that the code is capable of optimizing a 2D problem.

5.3.2. RRA displacement

The RRA is based on the Lagrange multiplier method, during the RRA the displacement is calculated with the virtual work method. To verify the working of this method a simple problem is optimized and analysed in Abaqus to see whether the found displacement of the code corresponds with the found displacement in Abaqus. A Python code to make an import file, of the found optimized structure, for Abaqus is given in the Appendix E.6 (Figure E.19 and E.20). The code has a build-in function which allows the user to select elements which are larger than a threshold and import only those into Abaqus. Using this function might lead to problems in the analysis. The imported structure in Abaqus can be incomplete, due to enforcing a minimal surface-area limit for the elements. Therefore the program might not be able to analyse the structure.

In Figure 5.5 the problem statement, the solution containing elements larger than 250 mm^2 and the solution containing all elements are shown. The 250 mm^2 is chosen, such that the really small elements do not cloud the figures of the solutions. The distance between nodes vertically of horizontally is 1 m, the load is a point load of 50 kN, the supports are shown by a triangle in the problem statement and the strength (tension and compression) is set to 20 N/mm^2 in the optimization code

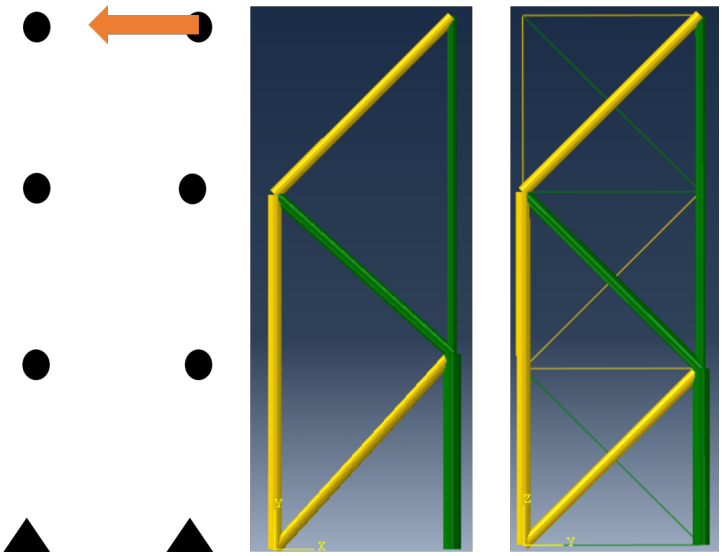


Figure 5.5: From left to right: the problem statement, the solution containing elements larger than 250 mm^2 and the solution containing all elements

Figure 5.6 shows three different general static analysis conducted in Abaqus, where truss elements are used (more specific, element T3D2). Respectively results of the analysis when the structure is made by hand, results of the analysis when elements larger than 250 mm^2 are taken into account (imported with the code) and results of the analysis when all elements are taken into account (imported with the code).

When importing the structural elements with the code each structural element is assigned as a truss element (T3D2) and connected to other truss elements with a hinge. This could results in a kinked solution (middle figure in Figure 5.6). When the solution is made by hand, successive structural elements are meshed as one truss element, resulting in a smoother result (left figure in Figure 5.6). Preferably the situation of the hand-made results is used, but with the currently used importing method this is unfortunately not possible. The displacement using the virtual work method when elements larger than 250 mm^2 are taken into account is $1.229 \cdot 10^{-3}$ m and when taking all elements into account is $1.196 \cdot 10^{-3}$ m. These are close to the found displacements shown in Figure 5.6, all found displacements are in a range of $\pm 3\%$ from each other. The rate of convergence in the RRA depends on the relaxation parameter, for now for the relaxation parameter a value of 5 is assumed. Trail and error should point out if this a correct value for the relaxation parameter and thus the rate of convergence. This is further discussed in Appendix E.2.

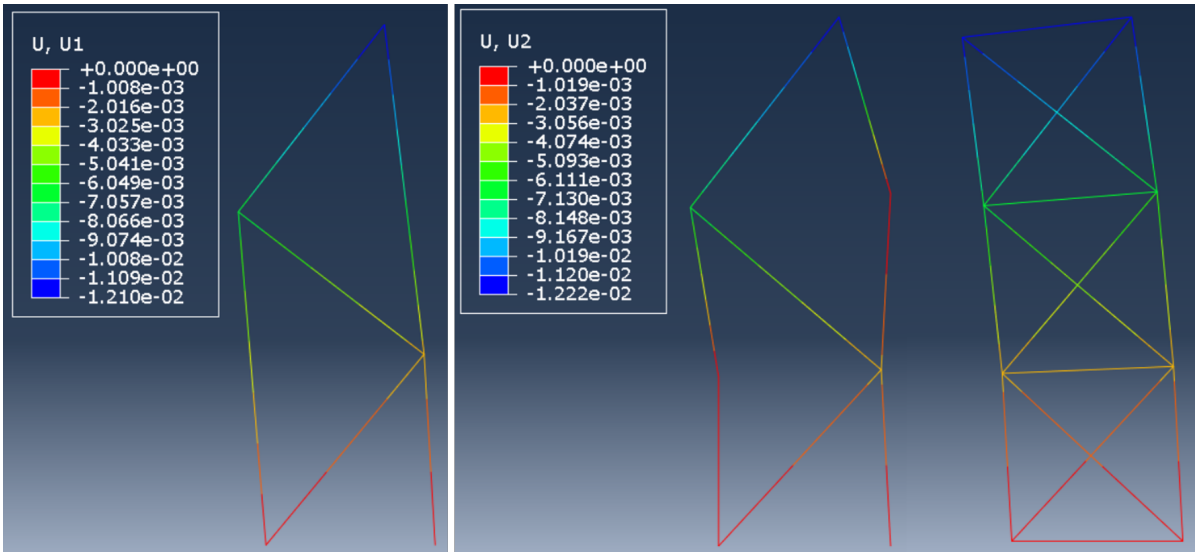


Figure 5.6: The legend is in meters, from left to right: results of analysis when structure is made by hand, results of analysis when elements larger than 250 mm^2 are taken into account (imported with the code) and results of the analysis when all elements are taken into account (imported with the code).

5.3.3. 3D cantilever

A more interesting case, for this research, is a 3D cantilever. In the article of Kwok a 3D cantilever is optimized, by using a PSL-based optimization method, the problem and the result are shown in Figure 5.7 [37].

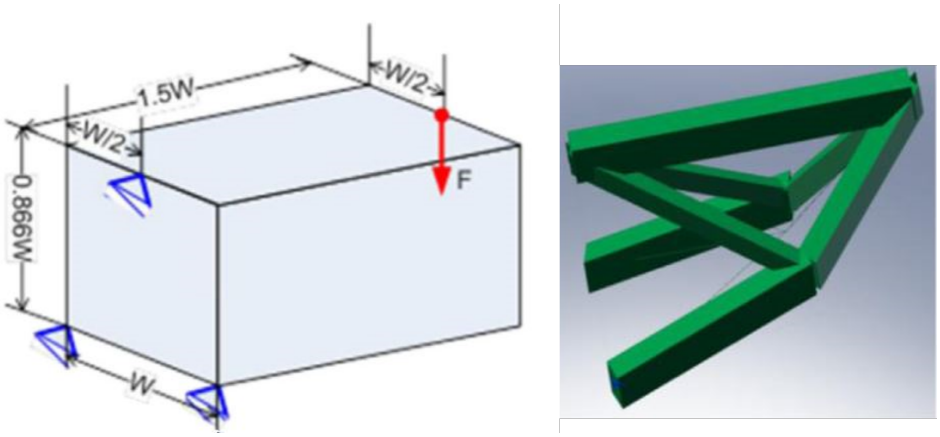
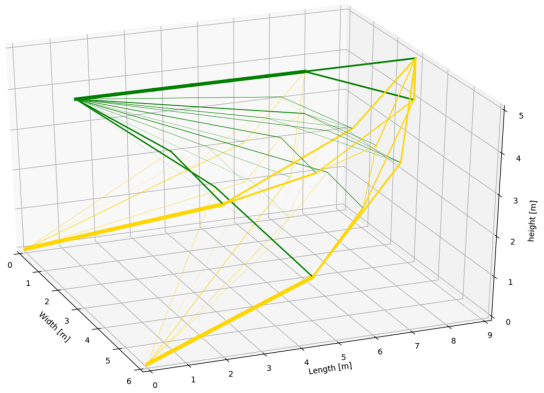


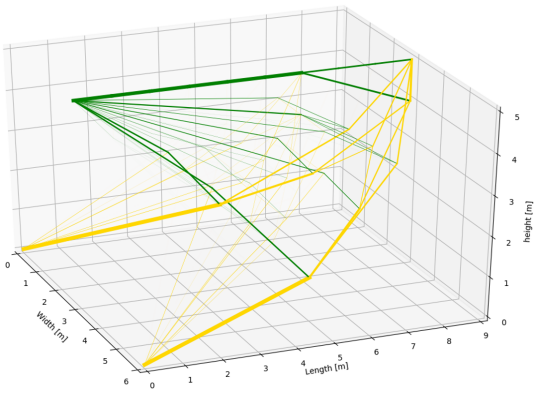
Figure 5.7: 3D cantilever optimization, on the left the problem statement and on the right the solution [37]

This optimization is repeated with the optimization code created in this research. The W in Figure 5.7 is taken as 6, the $0.866W$ is rounded to 5, the tension and compression strength are assumed to be the same and the load is chosen in the same order of magnitude. There is no boundary set for the minimal surface of a member. To test the influence of the self-weight, the problem is optimized twice, once with self-weight excluded and once with self-weight included. The results are respectively shown in Figure 5.8a and 5.8b.

Both figures show great similarities with each other and with Figure 5.7, the main difference is the size of the elements. This is not clearly shown in the figure because the elements are scaled with respect to the maximal member surface. But, the volume of the total structure is bigger (an increase of 6%) and it took the optimizer more iterations to find the solution, these iterations and found volumes are shown in Table E.1 in Appendix E.3.

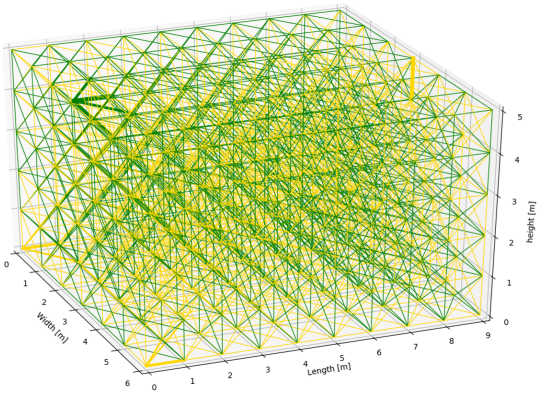


(a) Optimized 3D cantilever subjected to a point load, self-weight excluded (vol = 20.52m^3)

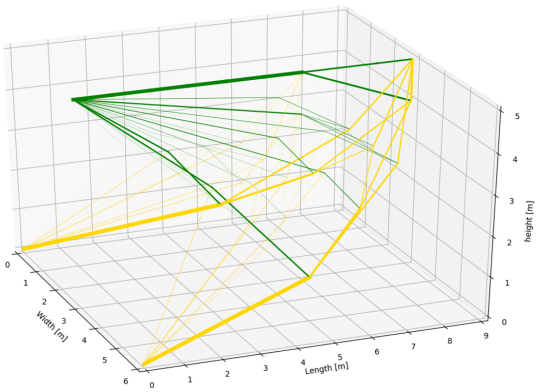


(b) Optimized 3D cantilever subjected to a point load, self-weight included (vol = 21.84m^3)

The next step in the verification of the code is to check what the influence of the imposed minimal boundary on the surface of the members is. The same 3D cantilever problem as before is optimized, but now the boundaries described in Equation 5.4 is implemented. In Figure 5.9a the solution is shown. A problem with the initial set arises, the initial set contains all elements which are smaller or equal to $\sqrt{2}$ meter (user-defined). When the whole initial set takes the minimal value imposed by the boundary conditions, the optimization code finds a solution without iterating. This means that all elements in the initial set remain in the solution, which results in the solution shown in Figure 5.9a. An option would be to erase the elements which are close to the minimum boundary in the next iteration step. This only works if the elements are bigger than the minimum boundary, through a big load on the structure. Otherwise all elements will be erased because they need to be at least the minimal value and the code erases all elements with this minimal value. The 3D cantilever problem is reproduced with a greater load, to show that this reasoning holds and the solution is shown in Figure 5.9b.



(a) Solution to the 3D cantilever problem with the imposed boundaries



(b) Solution to the 3D cantilever problem with the imposed boundaries and the solution with the imposed boundaries and the elements are erased, but subjected to a big load

From the observations above it is concluded that the function to implement a demand for the minimal surface-area of the structural elements does not work properly. Therefore the demand for the minimal surface-area should be enforced during the post-processing.

5.3.4. Verification with literature

To further check whether the solutions the code finds make sense, an optimization of a high-rise building is executed. In Figure 5.10 the solutions of different high-rise optimizations provided by literature are shown. All are high-rise buildings with approximately a ratio of the height and width of 6. Subfigure (a) is the initial problem for (b) and (c), (d) is a 3D solution to this initial problem (through copying a 2D problem to all four sides). Subfigure (e) has instead of two small forces at the top edge of the building, one combined force at the top center of the building (all other forces remain the same as in subfigure (a)). At last subfigure (f) is a 3D solution to a wind load (from one side) using symmetry. From these solutions can be concluded that the shape of the Michell truss returns (or could be described as a high-waisted X frame, diamond shape structure and arches). In all these figures the minimal value for the surface of the members and the self-weight is not taken into account. Also, the tension and compression strength are assumed to be the same and only the wind load is taken into account. Therefore, during the parametric study, it is investigated what the influence of the vertical force is on the solution of the optimization.

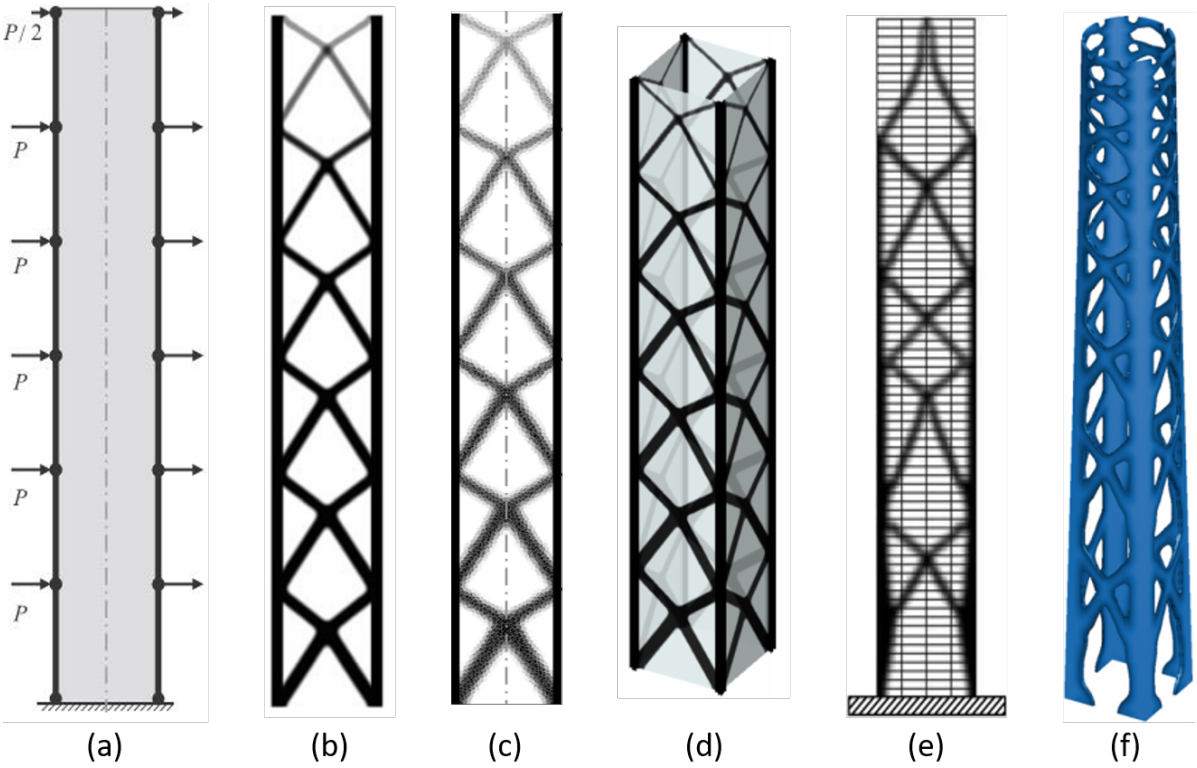


Figure 5.10: High-rise building optimization with (a) initial problem [6, 62], (b) 2D solution[6, 62], (c) 2D solution[5], (d) 3D solution[62], (e) 2D solution to a slightly different problem[11] and (f) 3D conceptual design for Lotte tower [63]

To see whether the created optimization code has logical outputs, this high-rise optimization problem is recreated, the results of the 2D and 3D problem are respectively shown in Figure 5.11 and 5.12. The diamond shape is returning in all solutions (2D and 3D) and is comparable to the solutions in Figure 5.10. The diamond shape can be seen in reality in the diagrid structure, which is described in Paragraph 2.2.1 and shown in Figure 2.2. The main difference between the diagrid diamond shape and the high-waisted X is the sizes of the angles. The diamond in a diagrid structure is a shape with two symmetry axis and the high-waisted X has one symmetry axis. In Appendix E.4 the results in Figure 5.12 are compared to the results of a Grasshopper plugin Peregrine, which is based on the same principles as this research. Here as well, the diamond shape is a returning phenomenon.

When increasing the size of structures analysed, the size of the optimization problems increase. Therefore more variables need to be taken into account, which increases the calcu-

lation time. Also, another issue is occurring, occasionally the code returns an error instead of a solution. The error is given by the solver (MOSEK), which solves the strength optimization. The error message is ”UNKNOWN”, according to the MOSEK documentation this often is a numerical problem [52]. One could change the MOSEK parameters, which control for example the termination criteria, this might have an effect on the stability and speed of the optimization.

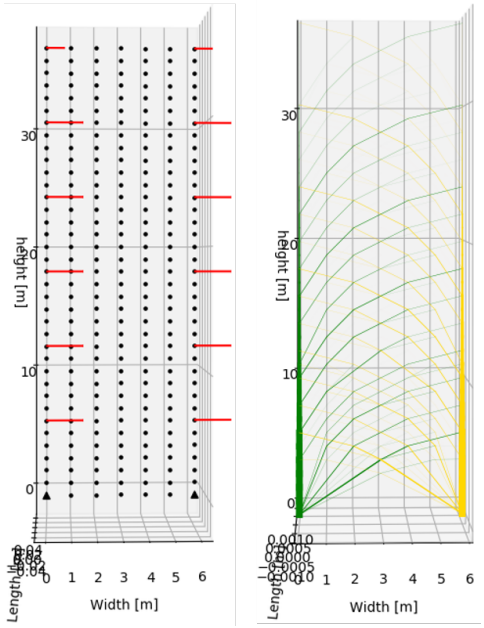


Figure 5.11: Problem statement and solution to 2D high-rise optimization problem

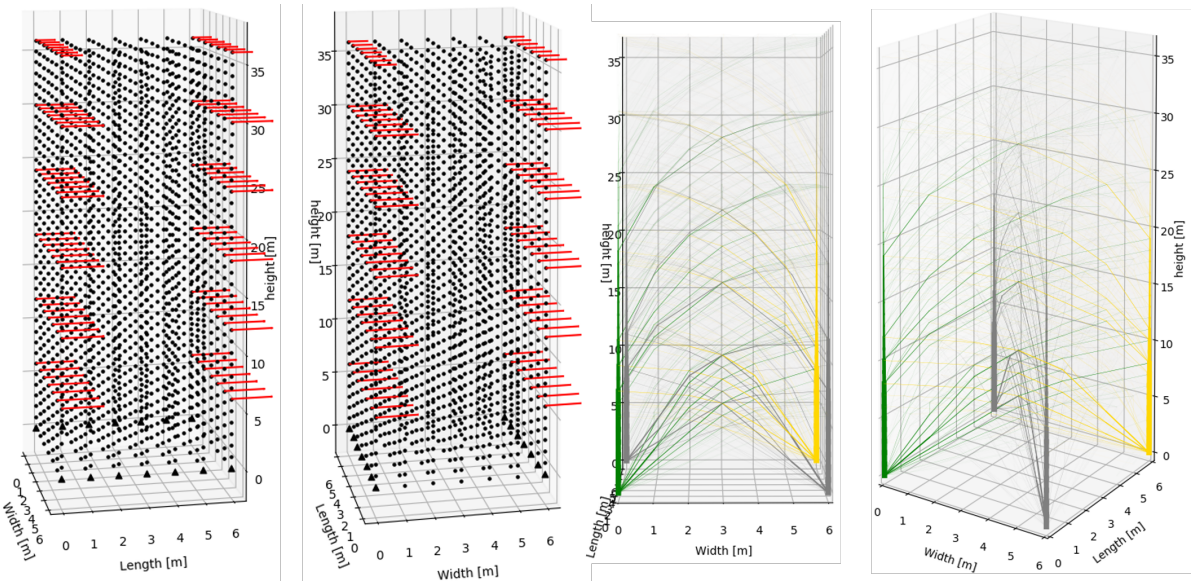


Figure 5.12: Problem statement and solution to 3D high-rise optimization problem, from left to right: load case 1, load case 2, side view solution, 3D view solution

5.4. Principles of the optimization

Information for this paragraph is obtained from Chapter 4 and retrieved from the Eurocode. The values for parameters, the loads, supports and the design spaces are discussed in this section.

5.4.1. Parameters

The following constants are used during the optimizations from which the results are shown in Chapter 6.

- The density of concrete, ρ, is 2500 $\mathrm{kg/m^3}$;
- The Young's modulus of concrete, E_{cm}, is 36000 $\mathrm{N/mm^2}$;
- The gravity acceleration, g, is 9.81 $\mathrm{m/s^2}$;
- The compression strength, f_{ck}, is 45 $\mathrm{N/mm^2}$;
- The relaxation parameter η is taken as 5.

The tension strength, $f_{\mathrm{ctk;0.05}}$, of concrete C45/55 is 2.7 $\mathrm{N/mm^2}$. Due to the low tensile strength of concrete and brittle failure behaviour, reinforcement is always applied and often the tension strength of concrete is neglected. Therefore the reinforcement should be able to take all the axial force and Equation 2.2 should be adjusted. It is assumed that the minimal area of reinforcement steel in the column is 1% of the cross-section area. Thus the minimal tension stress is the minimal tension force divided by the area of the cross-section. The characteristic strength (f_{yk}) for reinforcement steel is 500 $\mathrm{N/mm^2}$ and the material factor (γ_s) for steel is 1.15. Thus the input values for the design compression and tension strength are:

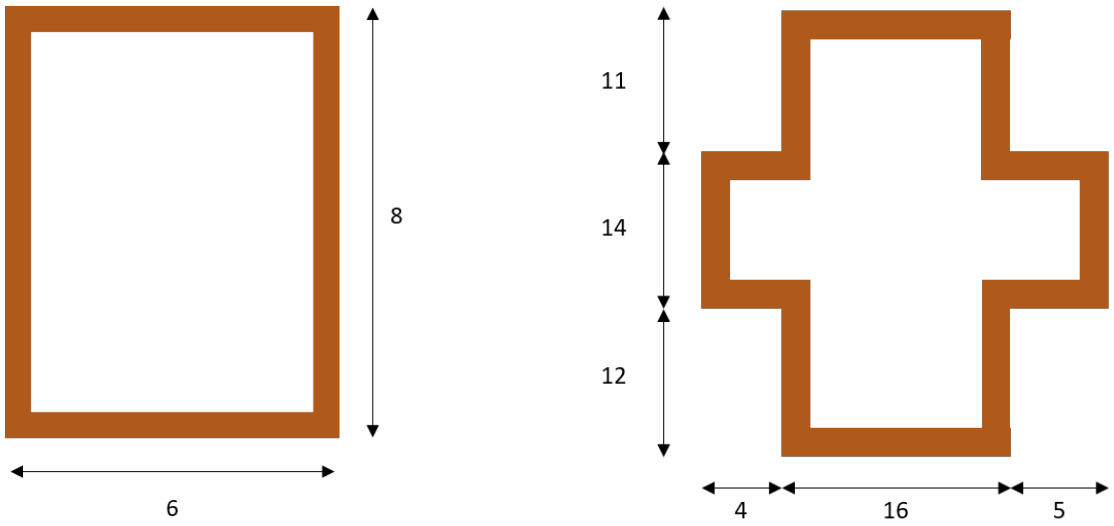
$$\text{Tensile design Strength :} \qquad f_{\mathrm{ctd}} = 0.01 a_c \frac{f_{\mathrm{yk}}}{\gamma_s} \frac{1}{a_c} = 4.35 \ \mathrm{N/mm^2} \tag{5.15}$$

$$\text{Compression design Strength :} \qquad f_{\mathrm{cd}} = 0.54 f_{\mathrm{ck}} = 24.75 \ \mathrm{N/mm^2} \tag{5.16}$$

5.4.2. Design space

As mentioned in Section 3.1, the design space contains all possible solutions. The design space is limited through using only structural elements at the perimeter of the building, causing the design space to be a tube with a certain floor plan. In this case two design spaces are considered, a rectangular design space with a height of 24 m with a floor plan as shown in Figure 5.13a for the parametric study and a design space based on the case study with a height of 54 m shown in Figure 5.13b. The choice for the parametric study is a smaller design space than the case study, such that calculation time wouldn't be a bottleneck. All dimensions are round to integers, such that distances between nodes is always 1 m.

Floors are often used to transfer the horizontal force to the structural elements which take care of the stability of the building. The floors act like rigid diaphragm. As assumed in the Paragraph 1.2.3 the core doesn't take any part in the stability, thus the floors would transfer the horizontal forces to the facade which transfer the force to the supports. To make the optimization more realistic, the floors are added to the design space (the floor height is 3 m). The rigid diaphragm action of floors can be implemented through making the floor elements significantly stiffer than the other elements. But the strength optimization is only based on the equilibrium of force, thus implementing a rigid diaphragm is not possible. However, it is possible to insert floors in the optimization which transfer the horizontal forces to the facades parallel to the direction of the force. In Appendix E.7 is shown that the floors ensure that the horizontal loads are transferred to the facade. The structural elements in the facade can only be in the plane of the facade.



(a) The rectangular floor plan (dimensions in meters)

(b) The floor plan for the case study (dimensions in meters)

Figure 5.13: Floor plans for the design spaces

5.4.3. Loads and supports

It is assumed that at each corner of the building a support is located. Thus the floor plan in Figure 5.13a has 4 supports and in Figure 5.13b has 12 supports. The loads from the case study are used during the optimization, which is described in Paragraph 4.2.2 and shown per floor (in tables) in Appendix F.1. These tables contain all vertical loads (including self-weight of the elements) on the case study. These are adapted, such that the loads which act on the facade are known. The core and the internal walls take an appropriate part of the vertical load (50%) of the floors and the build-in load-bearing structure in the side facade is taken out as load, because this will be redesigned. The adapted vertical load tables can be found in Appendix F.2. In Table 5.1 the total vertical load per floor during the optimization is noted.

Table 5.1: Total vertical load on the floors during optimization

Floor	Perm. total [kN]	Var. total [kN]
6^{th}	5187.31	581.38
7^{th} - 21^{th}	5227.28	589.33
22^{th}	5170.73	578.45
23^{th}	5032.21	554.90
Roof	2025.98	0.00
Total	95825.42	10554.60

It is assumed that the vertical load is evenly distributed over the whole perimeter. This assumption is made to simplify the input in the optimization, therefore all nodes have the same vertical loading. The total load is found by summing all loads per floor and shown in Table 5.1. In the case study (Figure 5.13b) the nodes are 1 m apart, therefore there are 124 nodes in the perimeter of the building. The building is 54 m high, thus in total there are $55 \cdot 124 = 6696$ nodes in the facade of the building. This holds that each node would have 14.05 kN permanent load and 1.55 kN variable load. The wind load is given in Table 4.2. Just as the vertical loads, for the optimization the wind loads are simplified. Therefore all nodes have the same loading, which makes it easier to implement in the optimization. The simplification of the wind load is based on the overturning moment.

Table 5.2: Wind load and overturning moment per meter floor on the long (parallel to grid line E) or short side (parallel to grid line 05), grid lines in Figure 4.3

		Line load on floor [kN/m]		Overturning moment [kNm/m]	
Floor	height [m]	Short side	Long side	short side	long side
6	0.00	5.50	5.80	0.00	0.00
7	3.00	5.50	5.80	16.50	17.40
8	6.00	5.50	5.80	33.00	34.80
9	9.00	5.50	5.80	49.50	52.20
10	12.00	5.90	5.80	70.80	69.60
11	15.00	6.30	5.80	94.50	87.00
12	18.00	6.70	5.80	120.60	104.40
13	21.00	7.10	5.80	149.10	121.80
14	24.00	7.10	6.60	170.40	158.40
15	27.00	7.10	6.60	191.70	178.20
16	30.00	7.10	6.60	213.00	198.00
17	33.00	7.10	6.60	234.30	217.80
18	36.00	7.10	6.60	255.60	237.60
19	39.00	7.10	6.60	276.90	257.40
20	42.00	7.10	6.60	298.20	277.20
21	45.00	7.10	6.60	319.50	297.00
22	48.00	7.10	6.60	340.80	316.80
23	51.00	7.30	6.60	372.30	336.60
roof	54.00	3.70	3.30	199.80	178.20
Total	513.00			3406.50	3140.40

The wind creates an overturning moment, the wind at the top of the building has more impact than the wind at the lower part of the building. To calculate the general load value per node, the current value per kN/m is multiplied with the height of that floor (6^{th} floor = 0 m, roof = 54 m). This is divided by the sum of all heights of floors (513 m) to find the equal wind load per node which generates the same amount of overturning moment:

- Line load on the short side: $\frac{3406.50}{513.00} = 6.64\,\text{kN/m}$;
- Line load on the long side: $\frac{3140.40}{513.00} = 6.12\,\text{kN/m}$;

5.5. Summary

This chapter answers the following two sub-research question:

- *"How can the optimization problem, for different design spaces, be defined?"*
- *"Does the optimization code work and what is the input for the optimizations?"*

The mathematical description of the optimization code and it's functions is given in Section 5.1. The structure of the Python code is explained in Section 5.2. All things considered, it is concluded that the code works properly in 3D. A known problem is imposing the boundary for the minimal surface of each element, therefore it is concluded that this boundary should be enforced during post-processing. Also, during the verification of the code with literature, the code occasionally returns an error instead of a solution. The error is given by the solver (MOSEK), which solves the strength optimization. The error message is "UNKNOWN", according to the MOSEK documentation this often is a numerical problem [52]. One could change the MOSEK parameters, which control for example the termination criteria, this might have an effect on the stability and speed of the optimization.

The literature used to verify the code is based on horizontal loads. Therefore during the parametric study, the effect of the vertical force is investigated. The optimization problem of the case study is defined with the loads, supports, parameters and design space described in Section 5.4. Two design spaces are investigated, shown in Figure 5.13a and 5.13b, respectively for the parametric study and for the optimization of the case study. The supports are at every corner of the building, 4 for the design space with a rectangular floor plan and 12 for the design space with a floor plan based on the case study. The following parameters are used during the optimization of the case study and design space with a rectangular floor plan:

- The density of concrete, ρ, is 2500 $\mathrm{kg/m^3}$;
- The Young's modulus of concrete, E_{cm}, is 36000 $\mathrm{N/mm^2}$;
- The gravity acceleration, g, is 9.81 $\mathrm{m/s^2}$;
- The relaxation parameter η is taken as 5;
- Tensile design strength, f_{ctd}, is 4.35 $\mathrm{N/mm^2}$;
- Compression design strength, f_{cd}, is 24.75 $\mathrm{N/mm^2}$.

The following loads are used in the optimization of the case study:

- Each node in the facade is loaded with 14.05 kN permanent load and 1.55 kN variable load;
- Wind load on the short side is 6.64 kN/m at every floor;
- Wind load on the long side is 6.12 kN/m ar every floor.

6

Results of the optimizations

In this chapter the results of the parametric study and the optimization of the case study are discussed. The ratio between the total vertical and horizontal load is investigated and reported in Section 6.1. Section 6.2 contains the results of the optimization of the case study. In Section 6.3 and 6.4 post-processing of the results are shown. At last Section 6.5 analyses a found load-bearing structure for the case study. The sub-research question (partly) answered in this chapter is *"What is the difference between the optimized case study and the original case study and what is learned from this to benefit the design of future buildings?"*. The first part, concerning the difference between the original and optimized case study is answered in this chapter. The second part, concerning the benefit of future designs is answered in Chapter 8.

All optimizations are executed in Python, together with optimization packages CVX (version 1.0.25) and MOSEK (version 9.1.10) on a laptop with an Intel(R) Core(TM) i7-6700HQ processor and 8 GB RAM-memory. The strength optimization is explained and mathematically described in Paragraph 5.1.1 is decisive for the lay-out of the structural members. The stiffness optimization only increases certain members to ensure the demand for displacement is satisfied. Therefore, in the following optimizations only the strength optimization is executed. During the optimization of the case study often the numerical error occurs (as discussed in Paragraph 5.3.4) when the self-weight is included and when there is a demand imposed for the minimal surface-area of a strutural element. Also the calculation times are increased by using these functions. Thus these two functions are not taken into account during obtaining the results for this chapter, to mitigate the occurrence of the error and shortening the calculation time. Some other points of attention for these optimizations:

- Yellow means compression and green means tension in the figures;
- The elements shown in the figures are not scaled to reality, but their mutual ratio is correct;
- In the figures of the solution only elements bigger than $2.5 \cdot 10^{-3}$ m^2 ($50x50$ mm) are shown, unless stated otherwise in the caption.
- The generated data, Python codes, images and Abaqus files can be found with the following DOI: 10.4121/uuid:d5a42352-a417-4082-aa0f-87047812bfed

6.1. Ratio vertical and horizontal load

There are two cases investigated to see what influence the ratio of the total vertical load and the total horizontal load is (denoted by v/h-ratio). The two cases which are investigated are:

1. Design space with a rectangle floor plan, the wind load acts in the positive x-direction;
2. Design space with a rectangle floor plan, two load cases, respectively: the wind load acts in the positive x-direction and the wind load acts in the positive y-direction.

The wind load acts only at nodes located at heights where a floor is situated and the vertical load acts on all nodes which are situated in the facade of the structure.

6.1.1. One load case

Figure 6.1 shows the top view and the side view of the problem. The vertical loads are left out such that wind loads and supports are visible in the figures.

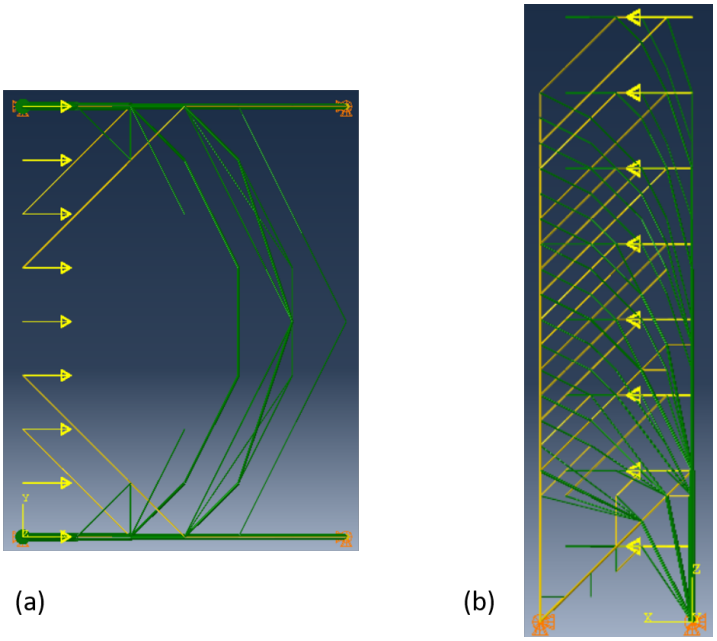


Figure 6.1: (a)Top view of the solution including loading (yellow arrow) and support (orange triangles), (b) side view of the solution including loading (yellow arrow) and support (orange triangles)

The following v/h-ratios are investigated in this scenario: $0, 0.1, 0.2, 0.4, 0.8, 1, 2, 4, 8$. A selection $(0, 0.4, 1, 2, 4, 8)$ is shown in this paragraph, the others $(0.1, 0.2, 0.8)$ are shown in Appendix G.1. Figure 6.2 shows the solutions for the facade parallel to the x-axis and Figure 6.3 shows the solutions for the facade perpendicular to the x-axis.

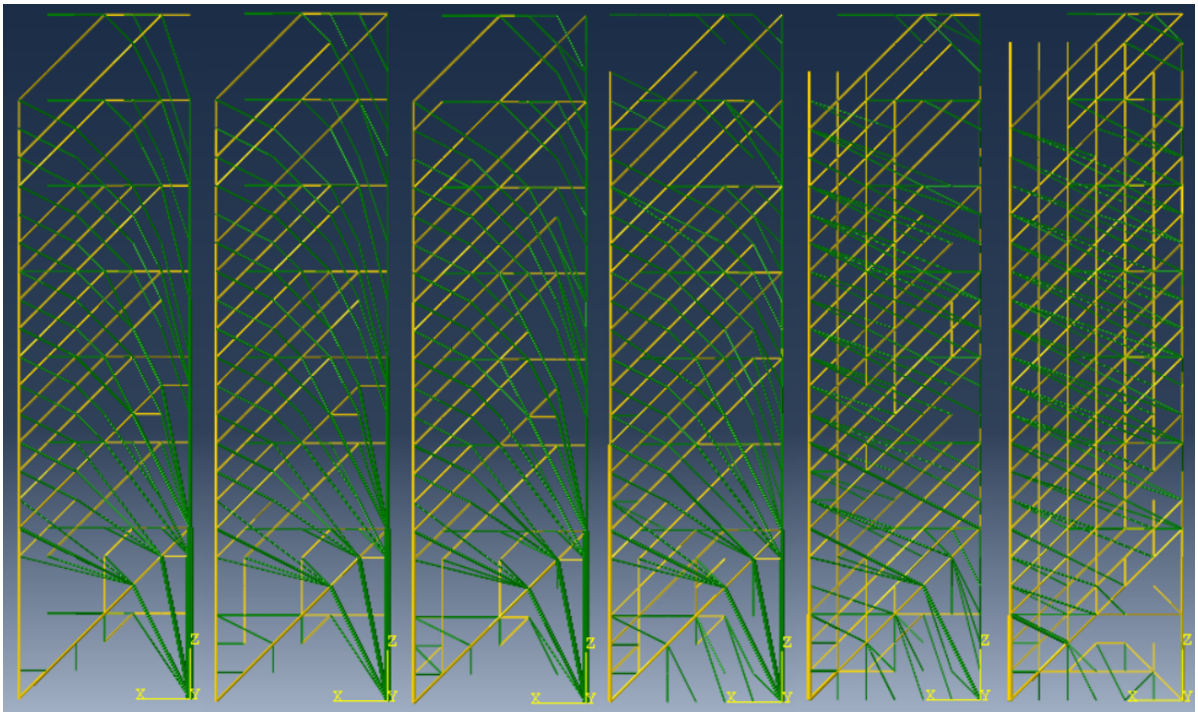


Figure 6.2: Facade parallel to the x-axis and the direction of the wind load, from left to right: v/h-ratio = 0, 0.4, 1, 2, 4, 8

When the v/h-ratio is increased, Figure 6.3 shows that more and more vertical and inclined elements arise to transfers the vertical loads to the supports in the corner. As proved in Appendix E.7, the floors redirects the load to the facade parallel to the direction of the load. These facades transfer the loads to the supports. This is done via the shape of a Michell truss (Figure 5.4a), which can clearly be seen in Figure 6.2 when the v/h-ratio is lower than 4. When the ratio becomes 4 or larger the arches of the Michell truss can't clearly be seen, the figures get distorted. However, the arches are changed to long inclined structural elements which span over the width of the structure. Figures 6.3 shows only vertical and inclined elements to transfer the vertical load to the supports.

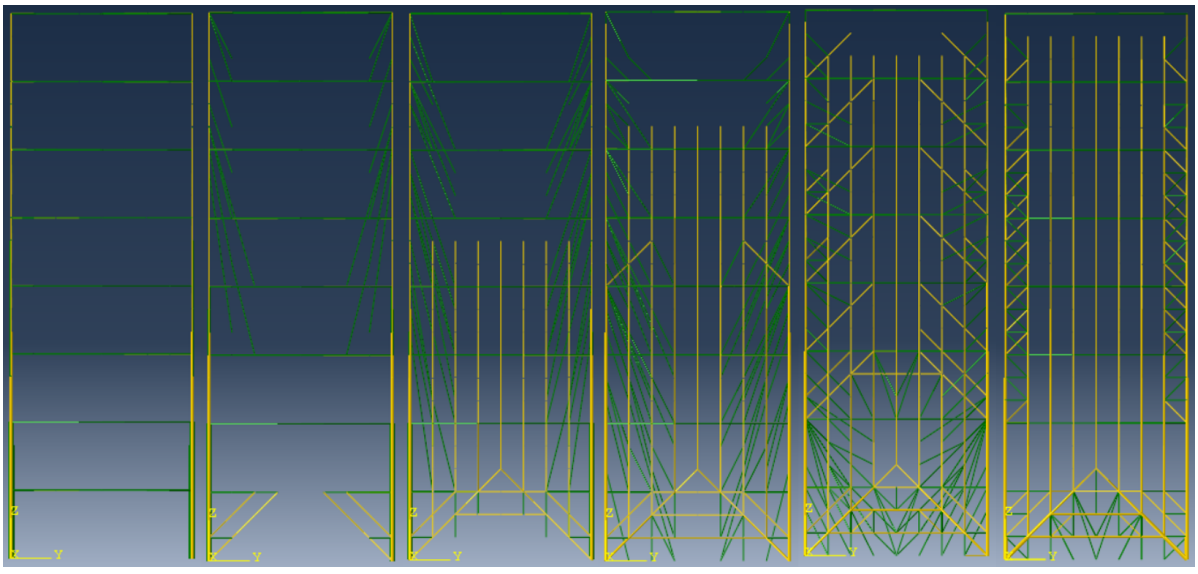


Figure 6.3: Facade parallel to the y-axis and the direction of the wind load, from left to right: v/h-ratio = 0, 0.4, 1, 2, 4, 8

6.1.2. Two load cases

Figure 6.4 show the top view with respectively load case 1 and load case 2. Load case 1 is the wind load in the positive x-direction and load case 2 is the wind load in positive y-direction. Load case 2 is the only addition made to the optimization discussed in Paragraph 6.1.1. The solutions, for the v/h-ratio's 0, 0.2, 1 and 4, are shown in Figure 6.5 and 6.6 and these satisfies both load cases.

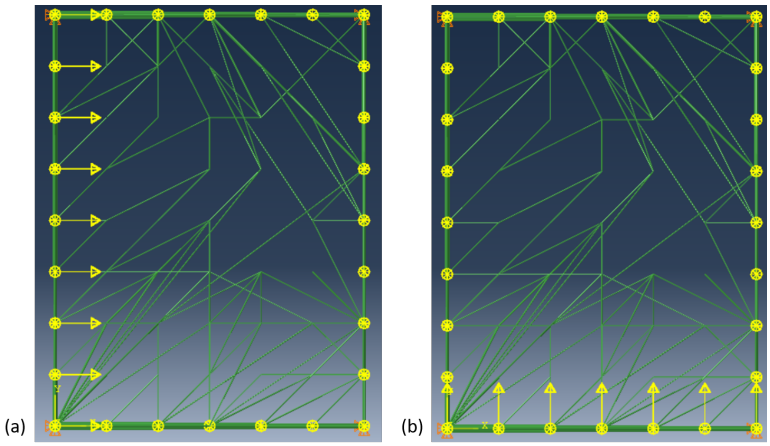


Figure 6.4: (a) Top view of the solution with load case 1, (b)top view of the solution with load case 2

In the solution with one load case, the facade parallel to the load direction contained the arches to redirect the load to the supports. Thus with two load cases it is expected that both facades will show arches. Looking at the facade parallel to the x-axis (Figure 6.5) the arches are visible till the v/h-ratio of 1, but a lot less clear than in Figure 6.2.

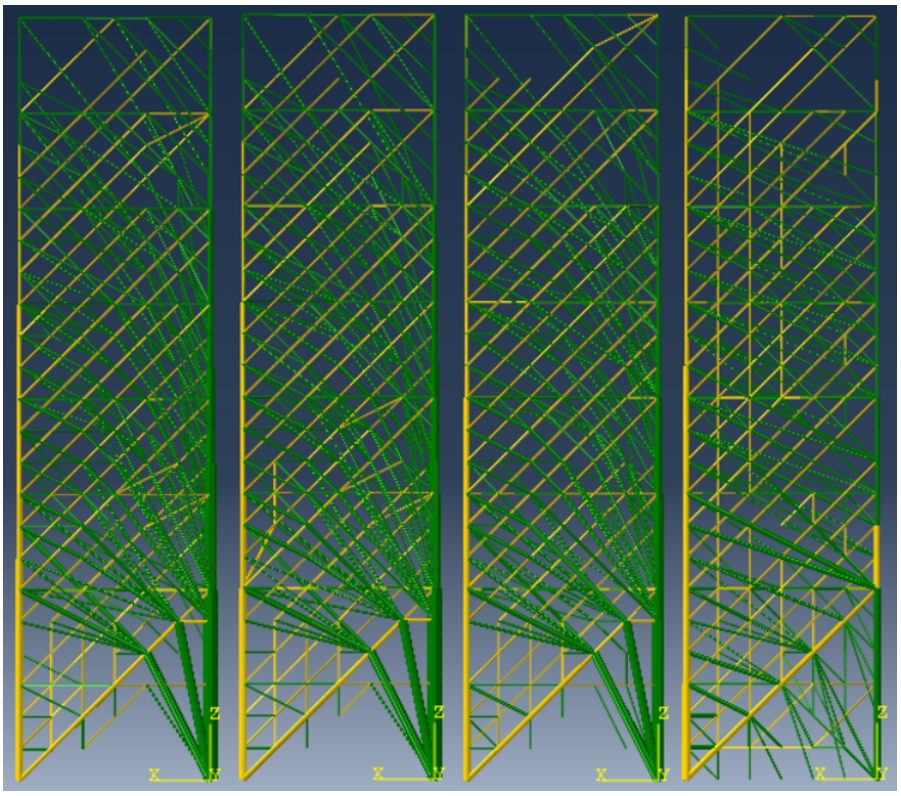


Figure 6.5: Facade parallel to the x-axis, from left to right: v/h-ratio = 0, 0.2, 1, 4

In the facade parallel to the y-axis (Figure 6.6), the arches or Michel truss shape is not recognisable independent of the v/h-ratio. Next to that, when the v/h-ratio increases more and more compression elements (yellow elements) are appearing to transfer the vertical loads to the supports.

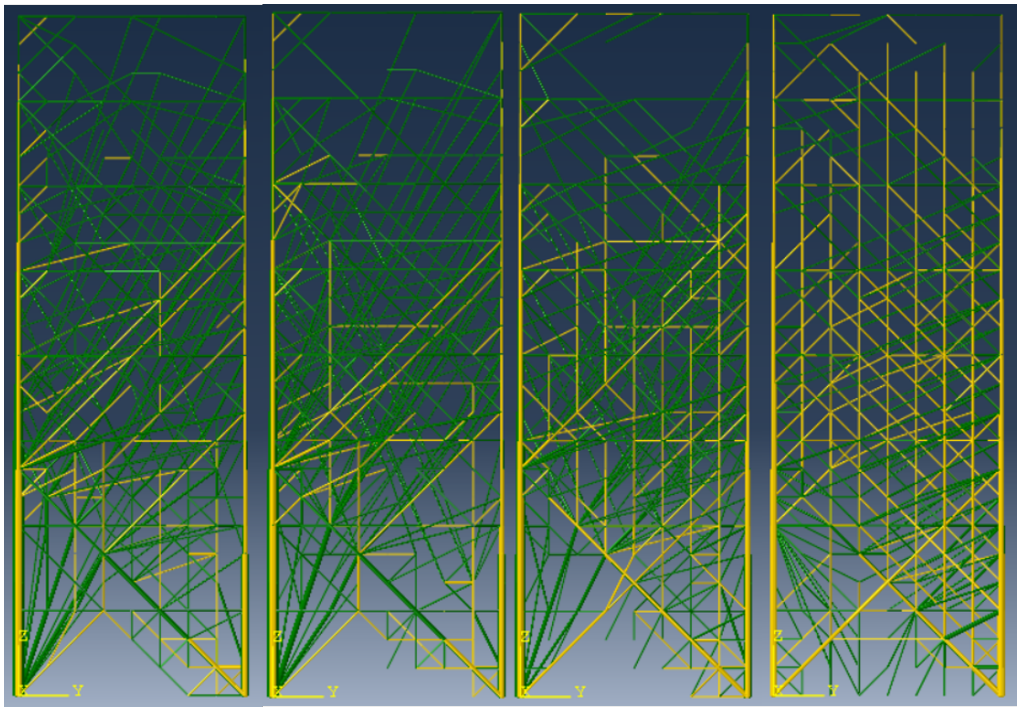


Figure 6.6: Facade parallel to the y-axis, from left to right: v/h-ratio = 0, 0.2, 1, 4

6.2. Optimizing case study

The second design space, which is investigated, is the design space based on the case study, shown in Figure 5.13b. The code used for the optimization of this design space is shown in Appendix H. In the first load case the wind acts in the positive x-direction on the structure (Figure 6.7) and the vertical load works on all nodes in the facade. The v/h-ratio is 0.33, therefore it is expected that facade parallel to the x-axis shows the arches from the Michel truss, which is confirmed by Figure 6.7. In the facade perpendicular to the direction of the load are only inclined elements which transfer the vertical load to the supports (see Figure 6.7).

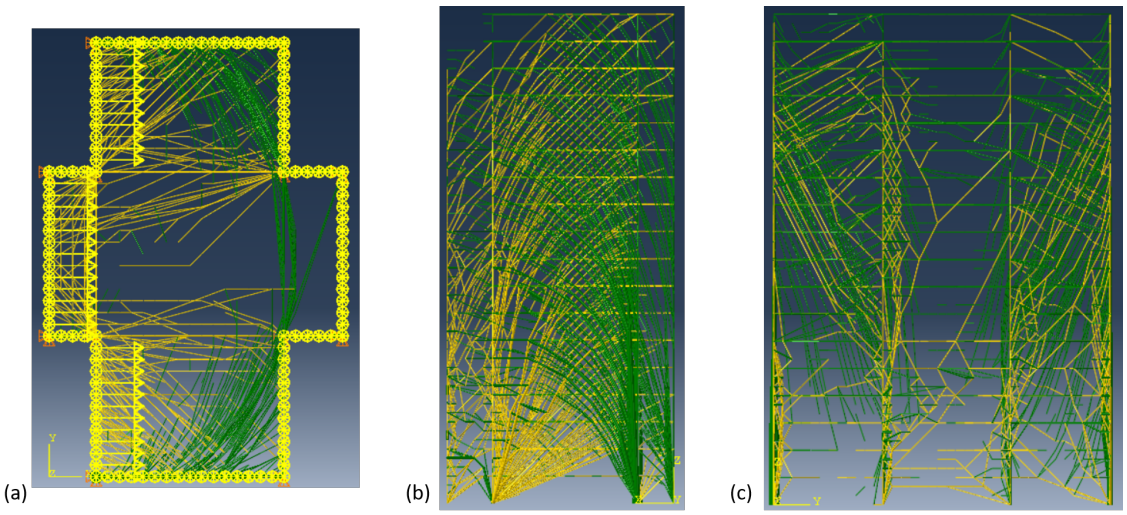


Figure 6.7: (a) Top view of the case study with the wind load in x-direction, v/h-ratio = 0.33, (b) facade parallel to x-axis with v/h = 0.33, (c) facade parallel to y-axis with v/h = 0.33

In the second load case the wind acts in the positive y-direction on the structure (Figure 6.8) and the vertical load works on all nodes in the facade. The v/h-ratio is 0.50, therefore it is expected that facade parallel to the y-axis shows the arches from the Michel truss, which is confirmed by Figure 6.8. In the facade perpendicular to the direction of the loads are only inclined elements which transfer the vertical load to the supports (see Figure 6.8).

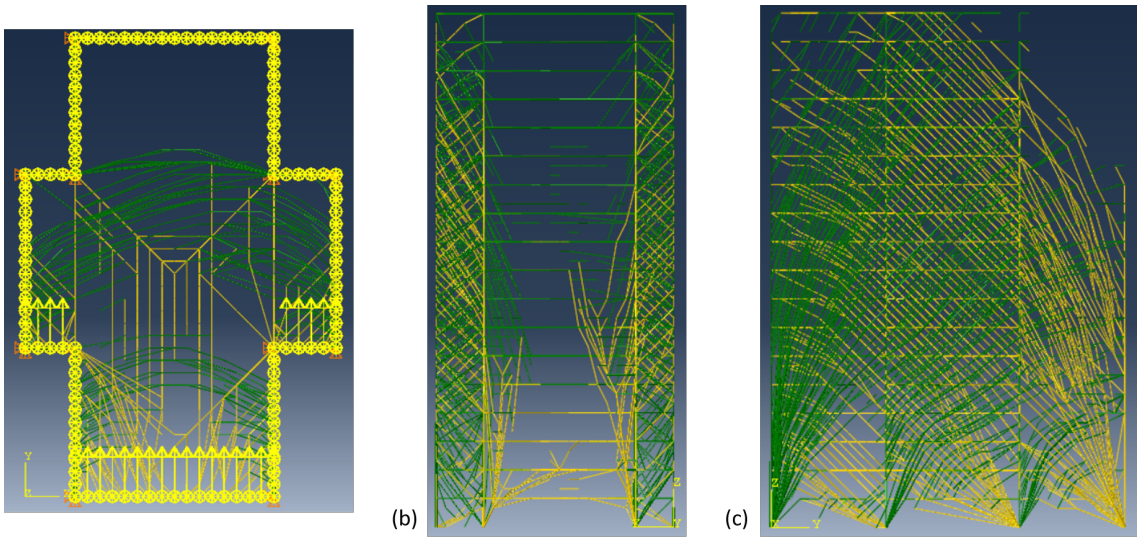


Figure 6.8: (a) Top view of the case study with the wind load in y-direction, v/h-ratio = 0.50, (b) facade parallel to x-axis with v/h = 0.50, (c) facade parallel to y-axis with v/h = 0.50

To investigate the optimal shape for the case study under realistic loads, the loads need to be combined with load combinations. Using equation 6.10(a) en 6.10(b) of the NEN-EN 1990+A1+A1/C2 and the corresponding national annex. The equations and corresponding factors for consequence class 3 are shown in Appendix F.3. The wind could load a building in four different ways, in the positive x and y-direction and in the negative x- and y-direction. For simplicity it is assumed that the case study is symmetric in a vertical line (at half of the width) and a horizontal line (at half of the length). Therefore only two types of wind need to be investigated and then symmetry can be used to find the final solution. The two load cases, where wind is in positive x- and y-direction (as shown in Figure 6.7 and 6.8), are investigated. The following three load combinations, for wind in the positive x-direction and in the positive y-direction are of importance:

1. 6.10(a): $1.5 \cdot \text{perm.} + 1.65 \cdot 0.4 \cdot \text{Var.} + 1.65 \cdot 0.0 \cdot \text{Wind}$
2. 6.10(b): $1.3 \cdot \text{perm.} + 1.65 \cdot 0.4 \cdot \text{Wind} + 1.65 \cdot \text{Var.}$
3. 6.10(b): $0.9 \cdot \text{perm.} + 1.65 \cdot \text{Wind}$

The maximum compression force will occur if the vertical force (in the negative z-direction) is at it's greatest, either the first or the second load combination. The maximum tension stress will occur if the vertical force is the smallest and the wind is the greatest, load combination three. The vertical force acts on all nodes in the facade. The Figures 6.7 (a) and 6.8 (a) show on which nodes the wind load acts. The number of nodes, where the loads on acts are stated in Table 6.1 and are retrieved from the Excel-output file:

Table 6.1: Amount of nodes on which the forces act

Type of load	Nodes
Vertical	6820
Horizontal in x-direction	684
Horizontal in y-direction	468

The load combinations and the loads per node (retrieved from Paragraph 5.4.3) are combined, the total vertical and horizontal load per node, as well as the v/h-ratio, are shown in Table 6.2. The total loads per node are multiplied with their amount of nodes on which they act, stated in Table 6.1 to find the v/h-ratio.

Table 6.2: 6 load combinations, the unit of the loads are kN/node

L.C.	Perm.	Var.	Wind	Total v	Total h	v/h-ratio
1, x-direction	14.05	1.55	6.12	22.10	0.00	-
2, x-direction	14.05	1.55	6.12	19.29	10.10	19.04
3, x-direction	14.05	1.55	6.12	12.65	10.10	12.48
2, y-direction	14.05	1.55	6.64	19.29	10.96	25.65
3, y-direction	14.05	1.55	6.64	12.65	10.96	16.82

The results in Paragraph 6.1.1 and 6.1.2 showed that optimizing a problem with multiple load cases clouded the solution and v/h-ratio above 2.0 would give a solution containing long inclined members. The lowest v/h-ratio in Table 6.2 is 12.48, thus the solution probably would contain the long inclined members and no clear pattern.

Preferably all the load cases are all implemented in an optimization, unfortunately the laptop used does not have enough RAM-memory to conduct such an elaborated optimization. Conducting such an optimization would also take quite some time, for example the optimization where the solution is shown in Figure 6.7 took more than 24 hours. Therefore running all load cases separately and combining them afterwards isn't done because it's a

time costly procedure. An option would be to increase the distance between nodes, thus decrease the grid density. This would need some adaption of the dimensions of the case study, the dimensions need to be divisible by 2. This has not been done to keep unity in the results. Another aspect which would case/caused problems and delays during obtaining the results is the occasionally ”UNKNOWN” error by MOSEK, just as mentioned in Section 5.3. These occurred multiple times during obtaining the results for this chapter.

To see what type of solution the high values for the v/h-ratio will give, the case study is optimized with one of the load combinations (the one with v/h-ratio 19.04). The results of this optimization are shown in Figure 6.9. This figure does not show a clear pattern of how the forces are redirected to the supports, therefore post-processing of the results are needed.

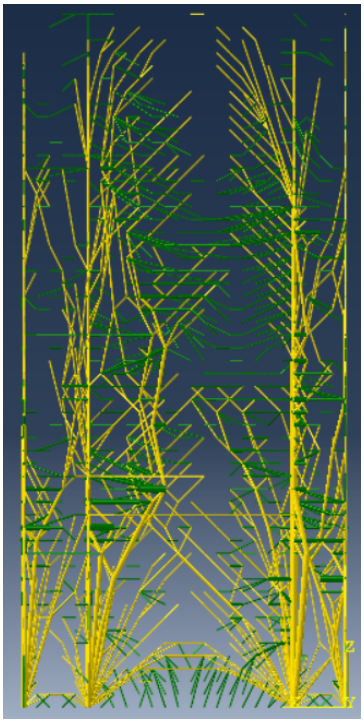


Figure 6.9: Solution to optimization of the case study subjected to the load combination with v/h-ratio of 19.04, facade parallel to x-axis

6.3. Post-processing of the results

To see whether a pattern in the load-bearing structure can be found, the results found in Section 6.1 and 6.2 are post-processed, the optimizations are simplified or parameters are changed. The following steps are taken:

- The results of the case study are sorted by element size, only elements which are bigger than certain limits are shown (post-processing);
- The acting points of the wind loads are located on the edge of the building (and not as line load on the width of the building), such that the wind loads directly works at the facade (simplifying the optimization);
- The influence of the ratio of the maximal tension stress and compression stress is investigated (parameters are changed).

6.3.1. Sorted by element size

All figures shown in this chapter only show the elements which are larger than $2.5 \cdot 10^{-3}$ m^2, these do not give a clear result. Therefore the solution for the case study might contain a pattern but is not shown due to the limit which is set. In this paragraph the limit is varied to see whether a pattern surfaces. In the Figure 6.10 and 6.11 the same solution as in Figure 6.9 is shown, except that the limit is adapted. Thus more ore less elements are shown, instead of all elements larger than $2.5 \cdot 10^{-3}$ m^2 this limit is set to, among others, $6.25 \cdot 10^{-2}$ m^2 and $3.75 \cdot 10^{-2}$ m^2. These figures show that the vertical elements above the supports need to be large in comparison to the horizontal and inclined elements. Furthermore no additional observations can be made.

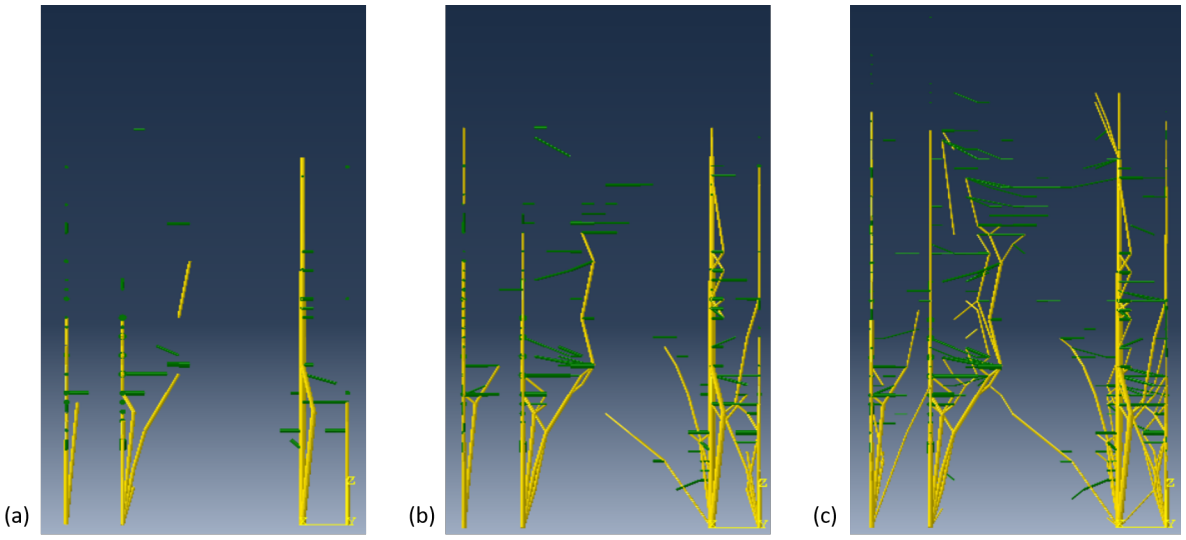


Figure 6.10: Solution with elements larger than (a) $6.25{\cdot}10^{-2}$ m^2 (250x250 mm), (b) $3{\cdot}10^{-2}$ m^2 ($\pm$173x173 mm), (c) $1.5{\cdot}10^{-2}$ m^2 ($\pm$122x122 mm)

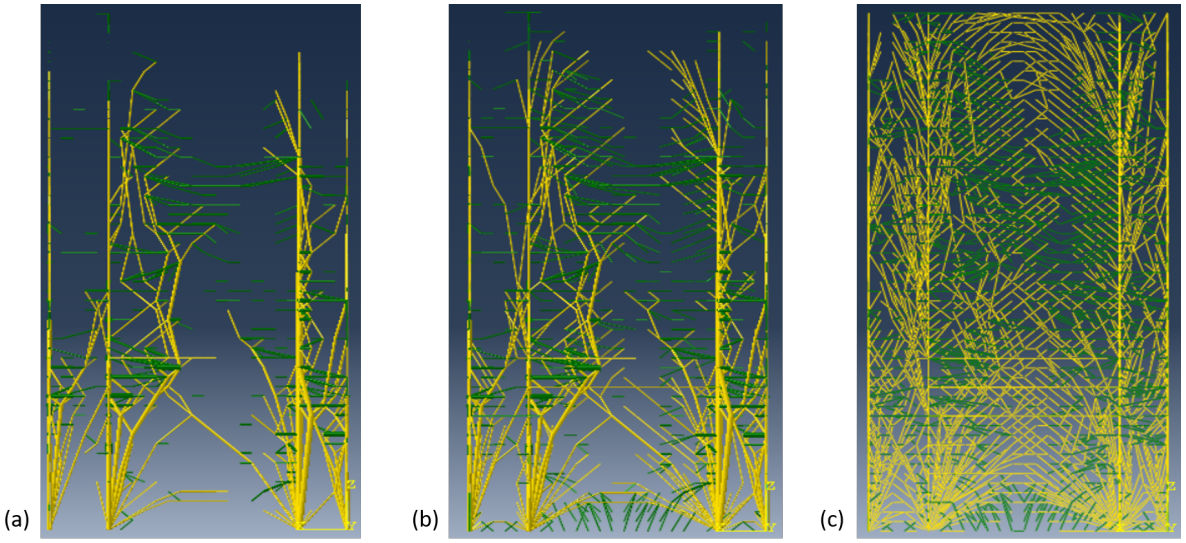


Figure 6.11: Solution with elements larger than (a) $7.5{\cdot}10^{-3}$ m^2 ($\pm$87x87 mm), (b) $3.75{\cdot}10^{-3}$ m^2 ($\pm$61x61 mm), (c) $6.25{\cdot}10^{-4}$ m^2 (25x25 mm)

6.3.2. Acting points of the wind load

The acting points of the wind loads are located on the corners of the building and not as line load over the width of the building. This ensures that the wind load works directly at the facade parallel to the direction of the force and the floors are no longer needed to redistributed the forces to the facades. This simplification of the wind load is done in the optimization of the design space with the rectangle floor plan loaded by two load cases(as described in Paragraph 6.1.2), when there is no vertical load. Figure 6.12 shows the results.

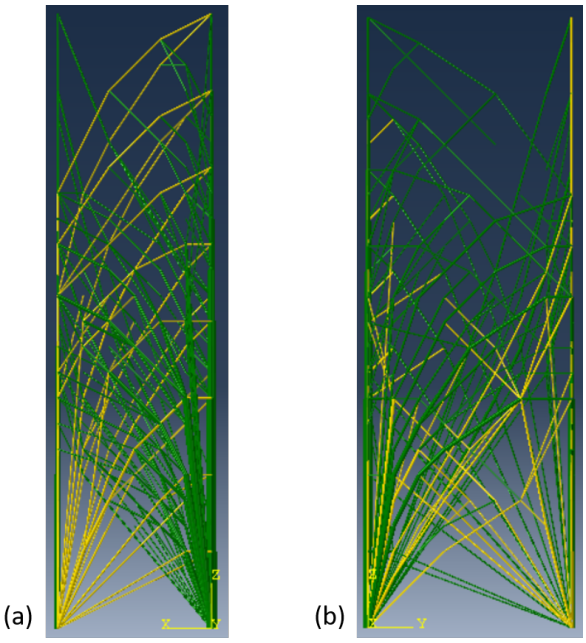


Figure 6.12: (a) The facade parallel to the x-axis, (b) the facade parallel to the y-axis

Figure 6.12 shows compression arches instead of compression diagonals which can be seen in in Figure 6.5 (most left with v/h-ratio = 0). The tension arches are still easily recognisable in the facade parallel to the x-axis. In the facade parallel to the y-axis (Figure 6.12) tension arches are more recognisable than the solution shown in Figure 6.6 (most left with v/h-ratio = 0).

6.3.3. Influence of the ratio of the maximal stresses

In the Paragraph 6.3.2 it is observed that changing the acting points of the wind loads to the corners of the structure has a positive effect on the clarity of the solution. Therefore, these manner of loading the structure is kept during this study to the influence of the ratio of the maximal stresses. The same design space loaded by only wind as in Paragraph 6.3.2 is optimized, only the maximal tension stress is increased. The maximal tension stress is 4.35 N/mm^2 and the maximal compression stress is 24.75 N/mm^2 in all optimzation till now, a tension/compression ratio of 0.18. The tension stress is increased to 13.5 N/mm^2 and 24.75 N/mm^2, respectively tension/compression ratios of 0.55 and 1.0.

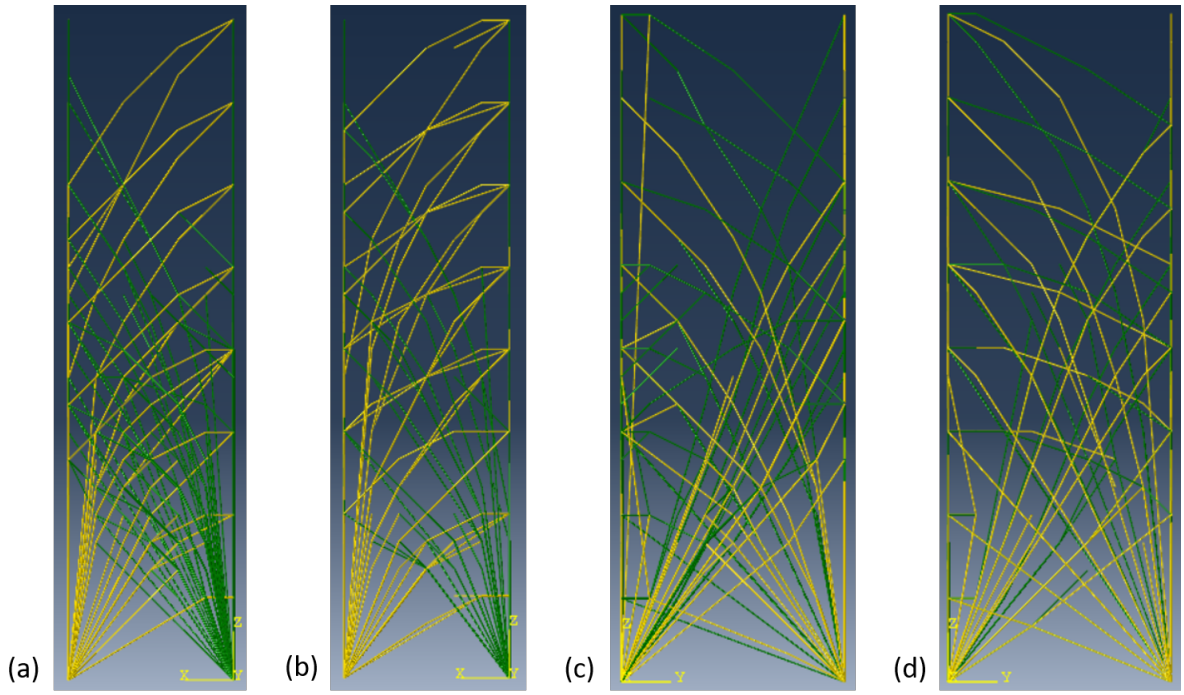


Figure 6.13: (a) The facade parallel to the x-axis with a tension/compression ratio of 0.55, (b) the facade parallel to the x-axis with a tension/compression ratio of 1.0, (c) the facade parallel to the y-axis with a tension/compression ratio of 0.55, (d) the facade parallel to the y-axis with a tension/compression ratio of 1.0

When the maximal tension stress is set to 24.75 N/mm^2, thus the tension/compression ratio is set to 1.0, there are less elements needed to transfer all force and the solution becomes less clouded. The tension and compression arches are clearly seen in both facades (parallel to x-axis en y-axis), shown in Figure 6.13. The patterns are similar to what can be found in literature, shown in Figure 5.10.

6.4. Schematisation by hand

In the previous sections the results are shown and an attempt to simplify the results is made. These attempts did gave insight in what could be the optimal load-bearing structure. The idea of this section is to create two potential load-bearing structures (in 2D), based on previous results. These are compared and there is concluded which one is better based on volume and occurring stresses. In Section 6.5 the found load-bearing structure is applied to the case study and checked whether this is an improvement in comparison with the original case study. In the following figures all positive values are tension and all negatives values are compression.

An recurring element in the results is the inclined compression element, therefore this will come back in both potential load-bearing structures. In Paragraph 6.1.1 long inclined structural elements which span over the width of the structure are observed, these are included in the load-bearing option one. These are implemented as elements over the width of the structure which are located between floors, see Figure 6.14(a). Another important aspect which comes forward in the results are arches, these are included in the second load-bearing option (Figure 6.14(b)). The wind load can come from both directions, therefore symmetry is applied to the two models (result is shown in Figure 6.14(c) and 6.16(d)).

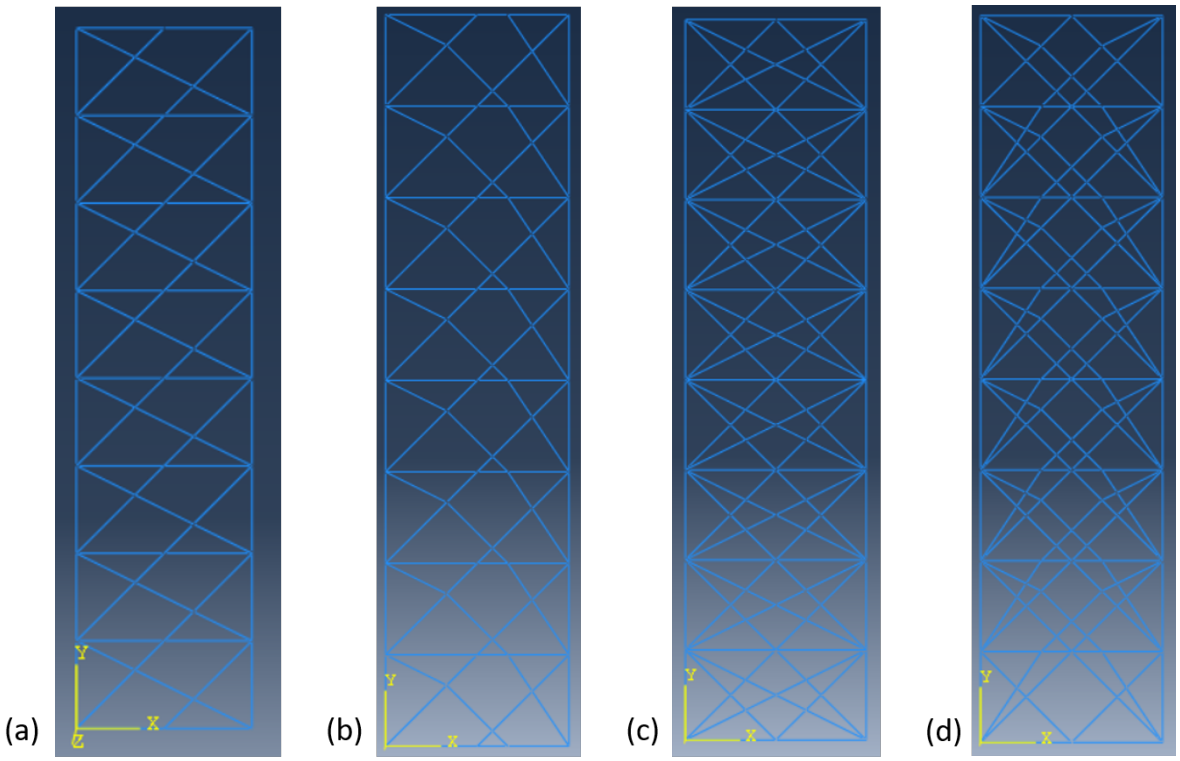


Figure 6.14: (a) Load-bearing structure option 1, (b) load-bearing structure option 2, (c) load-bearing structure option 1 with symmetry applied, (d) load-bearing structure option 2 with symmetry applied

After applying the symmetry to the second load-bearing option, the inclined compression elements are redundant. The arches will transfer the forces downwards, thus these are deleted from the model. Also the arches are reshaped such that the kink is at floor level, the adapted load-bearing structure is shown in Figure 6.15.

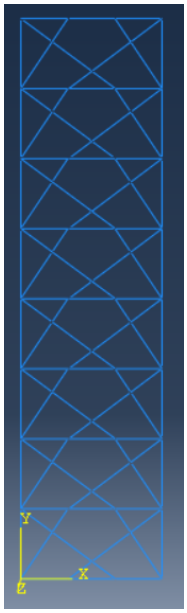


Figure 6.15: Adapted load-bearing structure option 2 with symmetry applied

Both load-bearing structure options (Figure 6.14(c) and 6.15) are subjected to realistic vertical loads similar to the case study and wind loads, derived from the NEN-EN 1991-1-4+A1+C2. The load-combinations which are used for the case study, described in Section 6.2, are also applied to both load-bearing options. For the case of simplicity all elements have the same surface, 0.0625 m^2 (250x250 mm). The results of the linear static analysis in Abaqus are shown in Appendix G.2. The maximum tension and compression stress found in the analysis and the volume of each option are shown in Table 6.3.

Table 6.3: Option 1 vs. Option 2, stresses and volumes

	Max. tension [N/mm^2]	Max. Compression [N/mm^2]	Volume [m^3]
Option 1	4.4	19	345.10
Option 2	2.6	29	239.69

Option 2 uses ±30% less material than option 1 and has a lower maximum tension stress. The maximum compression stress is ±50% higher. The material used in the case study is reinforced concrete, which can withstand compression well but not tensile. Thus a high compression stress and a low tensile stress is preferred. Next to the stresses, the material used is important. The less material there is used the better. Therefore option 2 is chosen as the best option for the load-bearing structure, it has the lowest material use and the preferred stress ratio. In the next section the case study is build with this load-bearing structure.

6.5. New load-bearing structure case study

The grid with arches found in Section 6.4 is applied to the case study. In this section the proposed load-bearing grid for the case study is analysed in Abaqus. In all figures in this section the positive values are tension and all negatives values are compression. A top view, the facade parallel to the x-axis and the facade parallel to the y-axis are shown in Figure 6.16. The dimensions of the the floor plan is equal to the dimensions of the design space for the case study (Figure 5.13b). In this analysis only elements of the type T3D2 are used. T3D2 are truss elements where only normal forces occur.

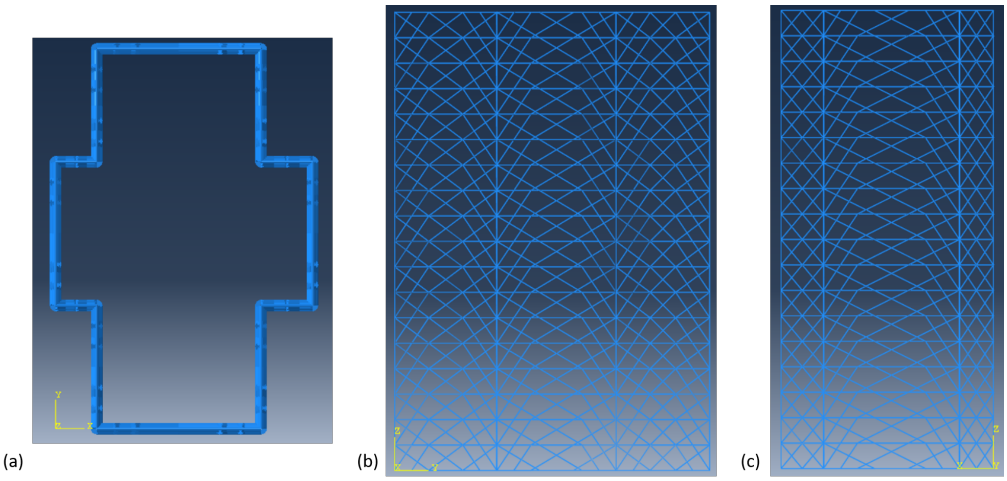


Figure 6.16: (a) Top view of the optimized case study, (b) geometry of the facade parallel to the x-axis of the optimized case study, (c) geometry of the facade parallel to the y-axis of the optimized case study

Just as in the optimization (see Section 5.4) the supports are placed at each corner of the structure. Also the loads are only applied on the corners of the structure, to simplify the analysis. The wind load is divided evenly (50-50) in a pushing and pulling force on the structure. Only the load combinations in x-direction (Table 6.2) are applied during the analysis. First all elements are given the same cross-sectional area of 1 m^2. This will give all elements the same stiffness and therefore all forces are equally distributed and this will show us which elements will attract the most load. Certain inclined and horizontal elements in the facade perpendicular to the load-direction are attracting the loads, as shown in Figure 6.17

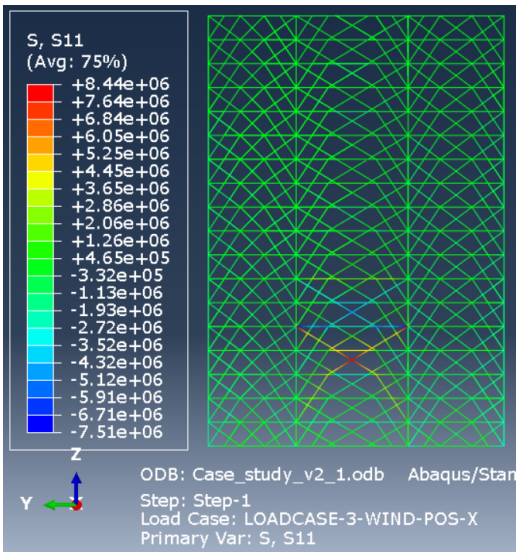


Figure 6.17: Side view of the solution, loaded by load combination 3 in x-direction (Table 6.2), the results are in N/m^2

As deduced in Paragraph 6.3.1 the vertical elements (columns above the supports) need to be larger in comparison to the inclined and horizontal elements. This will cause the stiffness of these elements to increase and thus attract more load. A benefit of this is that the inclined and horizontal elements are relieved of part of their load. An educated guess for the cross-sectional area of the elements is made as follow:

- Vertical elements of floor $1-12$ have a cross-sectional area of $1\ \mathrm{m}^2$ ($1x1$ m);
- Vertical elements of floor $13-18$ have a cross-sectional area of $0.5\ \mathrm{m}^2$ ($\pm 0.7x0.7$ m);
- All horizontal and inclined elements have a cross-sectional area of $0.0625\ \mathrm{m}^2$($0.25x0.25$ m), minimal value due to fire resistance (Section 2.3).

The results due to the change in element size are shown in Figure 6.18. The figure only shows the normative load combination (load combination 3 in x-direction, see Table 6.2) for the facade parallel to the x-direction and y-direction. All elements are below the $24.75\ \mathrm{N/mm}^2$ compression stress and almost all elements are below the $4.35\ \mathrm{N/mm}^2$ tension stress (as defined in Section 5.4). Only some elements (grey coloured in the figure) are above the maximum tension stress.

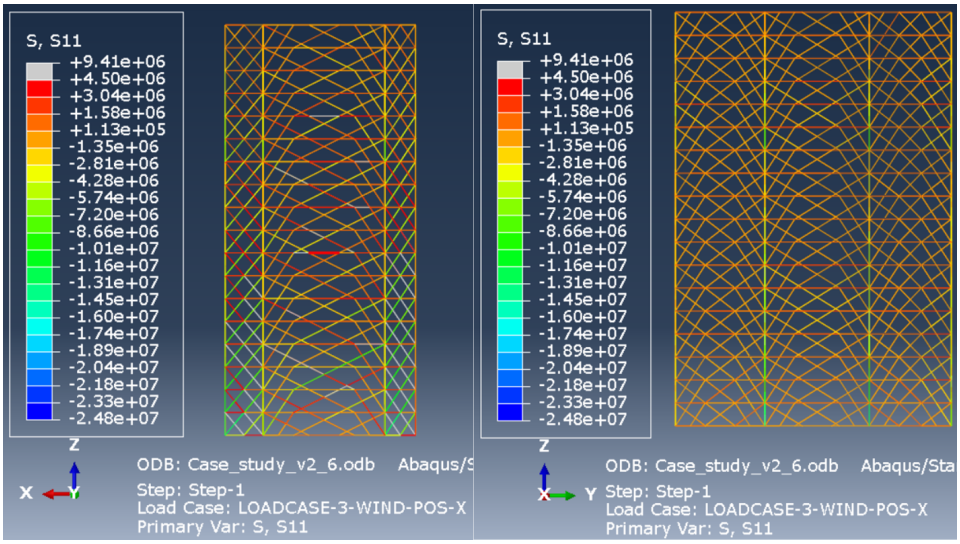


Figure 6.18: Load combination 3, on the left the facade parallel to the x-axis and on the right the facade parallel to the y-axis, the results are in $\mathrm{N/m}^2$

For now this is the best solution, to find the optimal solution a iteration process would be needed. As mentioned before: the stiffer an element is the more load it attracts, thus the thicker the element (in comparison to the others) to keep the stress low the more load it attracts. This would need an optimization on itself, due to time problems this is not done and the current solution (as shown in Figure 6.18) is used to see how much material there has been used. In the case study is $579.97\ \mathrm{m}^3$ used and of the optimized case study as shown in Figure 6.18 is $562.02\ \mathrm{m}^3$. Thus $\pm 3\%$ less material is used in the optimized structure.

6.6. Summary

For, roughly said, v/h-ratio below 4 the Michell truss is recognisable when optimizing the design space with a rectangle floor plan, above that ratio it is not recognisable anymore. The lowest v/h-ratio found in the governing load-combinations is 12.71. The results of the design space with a rectangular floor plan (Section 6.1) showed that v/h-ratio of 4 and higher will not give a clear pattern for the load-bearing structure. The solution of the case study, subjected to one load combination, showed us that this is the case. Preferably all the load cases are all implemented in an optimization, unfortunately the laptop used does not have enough RAM-memory to conduct such an elaborated optimization. Running all load cases separately and combining them afterwards could be used in future purposes when using a laptop with limited RAM memory. Another aspect which would case/caused problems and delays during obtaining the results is the occasionally ” UNKNOWN” error by MOSEK. These occurred multiple times during obtaining the results for this chapter.

To see whether a pattern in the load-bearing structure can be found, the results found in Section 6.2 and 6.1 are post-processed, the optimization are simplified or parameters are changed. The following steps are taken:

1. The results of the case study are sorted by element size, only elements which are bigger than a certain limits are shown (post-processing);

2. The acting points of the wind loads are located on the edge of the building (and not as line load on a floor), such that the wind loads directly works at the facade (simplifying the optimization);

3. The influence of the ratio of the maximal tension stress and compression stress is investigated (parameters are changed).

The first step showed that the columns above the supports should be large in comparison to the other elements. The second and third step showed that the arches are the most optimal structure. With this information two possible load-bearing structures are made and analysed. After comparison the most efficient structure turns out to be arches. The load-bearing structure of the case study is schematised with arches and analysed.

With the chosen cross-sectional areas and geometry, described in Section 6.5, all elements are below the 24.75 N/mm^2 compression stress and almost all elements are below the 4.35 N/mm^2 tension stress. A small number of elements are above the maximum tension stress. To find the optimal solution a iteration process would be needed, because increasing the cross-section of an element will decrease the stress but increase the stiffness, thus attract more load. In the case study is 579.97 m^3 of concrete used and of the optimized case study as shown in Figure 6.18 is 562.02 m^3 of concrete used.

The sub-research question investigated in this chapter: *”What is the difference between the optimized case study and the original case study and what is learned from this to benefit the design of future buildings?”* can be partly answered. The optimized case study uses arches instead of (punched) structural walls and uses less material ($\pm$3%). The second part, concerning the benefit of future designs is answered in Chapter 8.

7

Discussion

In the discussion four points are discussed, the comparison between the original and found load-bearing structure, the calculation time and numerical errors, the method of the stiffness optimization and the clearness of the pattern for the load-bearing structure.

Comparison between original and found load-bearing structure
In Section 6.5 there is stated that the optimized load-bearing structure uses ±3% less material. When the comparison between the optimized load-bearing structure and the original load-bearing structure of the case study is made the following points should be kept in mind:

- Light incidence, acceleration of the building, inter-story drift, displacement of the top of the building and the 2nd order effect are not verified for the optimized load-bearing structure. However, the minimal size for structural elements due to fire resistance and flexural buckling (based on the assumption of square cross-sections) are verified;
- Not all load combinations are taken into account during the analysis of the load-bearing structure, in this case three load combinations are taken into account. This could affect the maximal occurring stress in the analysis;
- In the analysis of the optimized load-bearing structure truss elements are used, because this smoothens the transition from the solution of the GSM to the analysis. Truss elements are only based on normal forces, there are no moments and shear forces taken into account. Which in reality do occur in the structural elements;
- The found load-bearing structure uses ±3% less material. The dimensions of the elements are an educated guess in the found load-bearing structure and can be further optimized to increase this material savings.

Calculation time and numerical errors
During optimization of the design space similar to the case study long calculations times (up to 66 h) and errors, which likely indicate numerical problems, occur. These problems arise when the functions concerning the inclusion of self-weight, the demand for a minimal surface-area and the stiffness optimization are included in the optimization. Therefore these newly developed functions are not used in the optimization of the case study. Long computational times were expected due to the use of topology optimization (as mentioned in Chapter 3), but not lto this extent.

Method of the stiffness optimization
In Section 5.3 is shown that the stiffness optimization, based on a Recursive Resizing Algorithm (RRA), works properly. However, two notes have to be made:

- Section 5.3 shows that there is much uncertainty about the relaxation parameter and therefore a value is assumed. It is also shown that the relaxation parameter has an influence on the rate of convergence and final displacement;
- The input for the stiffness optimization is the output of the strength optimization;internal loads and surface area of each structural element. These output values of the strength optimization depends, among others, on Ultimate Limit State loads (ULS). Thus the stiffness optimization is optimizing the structure based on ULS loads. According to the Eurocode displacement demands should be based on Serviceability Limit State (SLS). SLS requires lower partial factors for the load combinations, therefore the stiffness optimization will give a solution which is excessive.

Clearness of the pattern for the load-bearing structure
The parametric study and the optimization of the case study, respectively Section 6.1 and 6.2 did result in solutions with a great number of elements. These solution did not give a clear pattern for the load-bearing structure. Some points which influence the results and explain why the results were not that clear are stated:

- The chosen design space are not slender, the case study (asymmetric floor plan) has a slenderness of 1.46 and the design space for the parametric study (symmetric floor plan) has a slenderness of 3. When a building becomes more slender, the ratio between the total vertical and total horizontal (v/h-ratio) becomes smaller. The wind load increases with the increase of height, but the vertical load on each floor stays the same. This decrease of the ratio would benefit the clearness of the results, as shown in Section 6.1. The results of the case study and the parametric study show that for low v/h-ratio a load-bearing structure consisting of arches similar to most literature. As mentioned in Paragraph 3.2.3, the quality of the solution depends on the grid refinement. During all optimization this is unchanged, between all nodes (horizontal and vertical) is kept 1 m. During both optimization it is assumed that at each corner a support is located. This limits the possibilities for load-bearing structure to redirect the loads and changing this could potentially show new patterns for load-bearing structures;
- Mostly the Ground Structure Method (GSM) is used for skeletal structures (Paragraph 3.2.3). The original load-bearing structure of the case study consists of (punched) load-bearing walls. The idea was that using a different method of designing the load-bearing structure might give new insights to save materials, but this might has a negative effect on the clearness of the solution;
- Concrete has a low tensile strength in comparison to its compression strength. Post-processing of the results showed (Paragraph 6.3.3) that when the maximum allowable stress are equal the results are much more clear, arches occur. Thus due to the difference in the chosen strengths the solutions become less clear. When using steel, which has a higher strength than reinforce concrete, less material would be needed to transfer the loads to the supports and therefore less (and/or smaller) structural elements;
- The floors distribute the horizontal load over the facade parallel to the horizontal load, as discussed in Paragraph 5.4.2. When the floors are not taken into account, through changing the acting points of the horizontal load to the corners of the structure (Paragraph 6.3.2), the solution becomes more clear.

8

Concluding remarks

In this chapter conclusions are drawn and recommendations are made for future research.

8.1. Conclusions

First the main research question is answered, then other conclusions which are drawn from this research are stated. At last the addition of this research to the current literature and the advice for designers and engineers is stated. The main research question is:

"What is the optimal topology for a reinforced concrete load-bearing structure, situated at the perimeter of a high-rise building when optimizing the material use?"

The optimal topology for a reinforced concrete load-bearing structure, situated at the perimeter of a high-rise building consist of arches based on the improvement of the material use by ±3% when using the newly found load-bearing structure for the case study. Other conclusions drawn from this research are:

- The asymmetric floor plan of the case study and its dimensions combined with the the loads, location of the supports and grid density are not a good match with the optimization code. The whole is too complicated to find a clear pattern for the load-bearing structure;
- The optimization code is suitable to give more insight in the manner the loads are transferred to the supports in situations where there is a low total vertical to total horizontal load ratio;
- Multiple options are shown which help improving the clearness of the solution. These options are: increasing the maximum stress ratio (compression to tension) to 1.0, decreasing the total vertical to total horizontal load ratio or ignoring the rigid-diaphragm working of the floor (through simplifying the horizontal loads as point loads on the corners).

This research extends the current literature with extra insights in the use of the Ground Structure Method in an optimization code. This optimization code has functions which can include self-weight, a stiffness optimization and flexural buckling. Also, it confirms that the arches (originating from Michel Truss) is an efficient manner to transfer the loads to the supports.

The advice for designers and engineers is to see what the possibilities are for arches to use in their load-bearing structure, because these are efficient in transferring the loads. The extended code including flexural buckling, fire safety, stiffness optimization, buildability, self-weight and second-order effects provides a first attempt to implement the rules from the Eurocode. The code becomes more usable in practice when more rules from the Eurocode are closely followed, implemented and verified.

8.2. Recommendations

In the following points recommendations are stated for researchers:

- At this point it is not recommended to use the current version of the optimization code for designing of structures. However if the following points will be improved this might change:
  - Adding a Graphical User Interface (GUI);
  - The speed needs to be increased;
  - Mitigate the negative effect of the extra functions (created in this research) on the speed and stability;
  - Implementing more requirements from the Eurocode, such as inter-storey drift.

  Adding a GUI makes the optimization code more accessible for people to use in practise. Another option to make it more accessible is to improve the speed of the optimization or a computer with more CPU and/or RAM can be used. To improve the speed of the code, it could be rewritten for multiprocessing or it can be rewritten such that the amount of loops is minimized (instead of loops, matrix multiplications are preferable because these are faster).

  As mentioned in the discussion, the added functions to the code cause stability and calculations time issues when optimizing the design space similar to the case study. The stability of the code can be improved by changing the solver or by changing the parameters of the solver (MOSEK).

- To improve the knowledge about the Ground Structure Method in structural optimization more research should be done into the following aspects:
  - The influence of the location and the sensitivity of boundary conditions on the optimal solution;
  - The influence of the design space on the optimal solution, this included the shape of the floor plan and the slenderness of the design space;
  - The dependency of the clearness of the solution on the type of material used, steel might give a clearer solution as the post-processing indicates (Paragraph 6.3.3).

  If further research into optimization of high-rise buildings is executed and only the maximal strengths and multiple load cases are needed a possibility is to use the plug-in Peregrine for Grasshopper. This plug-in is released during the time of this research, mid-November 2019. Grasshopper is based on the same mathematical principles as this research but is user-friendly through a GUI.

- More investigation into the relaxation parameter and the Recursive Resizing Algorithm is needed to make sure the stiffness optimization works properly. An optimal value for the relaxation parameter can be sought-after, through which the increase in volume due to stiffness optimization is minimized. Also, The RRA can be elaborated with dependency on the type of cross-section of the elements, therefore the moment of inertia.

- An extension of this research is possible through further optimizing the found load-bearing structure of the case study and see whether more material can be saved. Also, this analysis can be made more realistic by implementing beam elements which take moments and shear forces into account.

Bibliography

[1] Meisam Abdi, Ian Ashcroft, and Ricky Wildman. Topology optimization of geometrically nonlinear structures using an evolutionary optimization method. *Engineering Optimization*, 50(11):1850–1870, 2018.

[2] M. Aldwaik and H. Adeli. Advances in optimization of highrise building structures. *Structural and Multidisciplinary Optimization*, 50(6):899–919, 2014.

[3] M.M. Ali and P.J. Armstrong. Overview of sustainable design factors in high-rise buildings. In *Proc. of the CTBUH 8th World Congress*, pages 3–5, 2008.

[4] M.M. Ali and K.S. Moon. Structural developments in tall buildings: current trends and future prospects. *Architectural science review*, 50(3):205–223, 2007.

[5] Lauren L Beghini, Alessandro Beghini, William F Baker, and Glaucio H Paulino. Integrated discrete/continuum topology optimization framework for stiffness or global stability of high-rise buildings. *Journal of Structural Engineering*, 141(8):04014207, 2014.

[6] L.L. Beghini, A. Beghini, N. Katz, W.F. Baker, and G.H. Paulino. Connecting architecture and engineering through structural topology optimization. *Engineering Structures*, 59: 716–726, 2014.

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

[8] Martin Philip Bendsøe and Noboru Kikuchi. Generating optimal topologies in structural design using a homogenization method. *Computer methods in applied mechanics and engineering*, 71(2):197–224, 1988.

[9] Mp. Bendsoe and O. Sigmund. *Topology optimization: theory, methods and applications*. Berlin; New York: Springer, 2003.

[10] Dimitri P Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic press, 2014.

[11] S Bobby, SMJ Spence, E Bernardini, and A Kareem. Performance-based topology optimization for buildings under wind and seismic hazards. In *Structures Congress 2015*, pages 2218–2229, 2015.

[12] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[13] D Brackett, I Ashcroft, and R Hague. Topology optimization for additive manufacturing. In *Proceedings of the solid freeform fabrication symposium, Austin, TX*, volume 1, pages 348–362. S, 2011.

[14] CBRE. Komende jaren groot tekort aan kantoorruimte den haag, 2019. URL https://nieuws.cbre.nl/komende-jaren-groot-tekort-aan-kantoorruimte-den-haag/.

[15] CBS. Bevolking; kerncijfers, 2018. URL https://statline.cbs.nl/Statweb/publication/?DM=SLNL&PA=37296ned&D1=68&D2=0%2c10%2c20%2c30%2c40%2c50%2c60%2c67-68&VW=T.

[16] The Skyscraper Center. 100 tallest completed buildings in the world by height to architectural top, 2019. URL https://www.skyscrapercenter.com/buildings.

[17] C. Chan. Optimal lateral stiffness design of tall buildings of mixed steel and concrete construction. *The Structural Design of Tall Buildings*, 10(3):155–177, 2001.

[18] P.W. Christensen and A. Klarbring. *An introduction to structural optimization*, volume 153. Springer Science & Business Media, 2008.

[19] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 2016. URL http://stanford.edu/~boyd/papers/pdf/cvxpy_paper.pdf. To appear.

[20] W. Dorn, Gomory R., and M. Greenberg. Automatic design of optimal structures. *J. de Mecanique*, 3:25–52, 1964.

[21] S. van Eerden. Hoogtepunt voor binnenstedelijke woningbouw. *Cement*, 2017.

[22] Hans A Eschenauer and Niels Olhoff. Topology optimization of continuum structures: a review. *Applied Mechanics Reviews*, 54(4):331–390, 2001.

[23] FACTO. Dreigend tekort aan kwalitatief aanbod in de kantorenmarkt, 2019. URL https://facto.nl/onderzoek-kantorenmarkt-2018/?vakmedianet-approve-cookies=1&_ga=2.132792078.129100565.1563175556-2102634130.1557747494.

[24] F. Fu. *Advanced modelling techniques in structural design*. John Wiley & Sons, 2015.

[25] F. Fu. *Design and analysis of tall and complex structures*. Butterworth-Heinemann, 2018.

[26] Matthew Gilbert and Andrew Tyas. Layout optimization of large-scale pin-jointed frames. *Engineering Computations*, 20(8):1044–1064, 2003.

[27] Takao Hagishita and Makoto Ohsaki. Topology optimization of trusses by growing ground structure method. *Structural and Multidisciplinary Optimization*, 37(4):377–393, 2009.

[28] J. Haslinger and R.A.E. Mäkinen. *Introduction to Shape Optimization - Theory, Approximation, and Computation*. Society for Industrial and Applied Mathematics, 2003. ISBN 978-0-89871-536-1. URL https://app.knovel.com/hotlink/toc/id:kpISOTAC02/introduction-shape-optimization/introduction-shape-optimization.

[29] Behrooz Hassani and Ernest Hinton. *Homogenization and structural topology optimization: theory, practice and software*. Springer Science & Business Media, 2012.

[30] L He, T Pritchard, M Gilbert, and H Lu. *Peregrine User Manual*. LimitState Limited, Sheffield, 2019.

[31] Linwei He, Matthew Gilbert, and Xingyi Song. A python script for adaptive layout optimization of trusses. *Structural and Multidisciplinary Optimization*, pages 1–13, 2019.

[32] X. Huang and M. Xie. *Evolutionary topology optimization of continuum structures: methods and applications*. John Wiley & Sons, 2010.

[33] Kazufumi Ito and Karl Kunisch. *Lagrange multiplier approach to variational problems and applications*, volume 15. Siam, 2008.

[34] M. Kalanchiam and B. Mannai. Topology optimization of aircraft fuselage structure. *World Academy of Science, Engineering and Technology*, 77:110–114, 2013.

[35] Burj Khalifa. Facts & figures, 2019. URL https://www.burjkhalifa.ae/en/the-tower/facts-figures/.

[36] James J Kingman, Konstantinos Daniel Tsavdaridis, and Vassilli V Toropov. Applications of topology optimization in structural engineering: High-rise buildings and steel components. *Jordan Journal of Civil Engineering*, 159(3097):1–23, 2015.

[37] Tsz-Ho Kwok, Yongqiang Li, and Yong Chen. A structural topology design method based on principal stress line. *Computer-Aided Design*, 80:19–31, 2016.

[38] M. Langelaar and F. van Keulen. Engineering optimization concepts and applications [powerpoint slide lecture 1], 2018/2019.

[39] M. Langelaar and F. van Keulen. Engineering optimization concepts and applications [powerpoint slide lecture 13], 2018/2019.

[40] R.M. Lawson, R.G. Ogden, and R. Bergin. Application of modular construction in high-rise buildings. *Journal of architectural engineering*, 18(2):148–154, 2011.

[41] Q.Q. Liang, Y.M. Xie, and G.P. Steven. A performance index for topology and shape optimization of plate bending problems with displacement constraints. *Structural and Multidisciplinary Optimization*, 21(5):393–399, 2001.

[42] Limitstate3d. Peregrine, 01 2020. URL https://limitstate3d.com/peregrine.

[43] L.P.L. van der Linden, A. Andrejevic, J. Spiegeler, R. Crielaard, G. Torpiano, W. Riedijk, M. Beekman, R. Winter, and R. Titulaer. Parametric engineering, 2018. version 1.0.

[44] Massimo Majowiecki. The free form design (ffd) in steel structural architecture–aesthetic values and reliability. *Steel Construction: Design and Research*, 1(1):3–15, 2008.

[45] P. Martínez, Pascual Martí-Montrull, and Osvaldo Querin. Growth method for size, topology, and geometry optimization of truss structures. *Structural and Multidisciplinary Optimization*, 33:13–26, 01 2007. doi: 10.1007/s00158-006-0043-9.

[46] Skyscraper Source Media. Tallest buildings of the netherlands, 2019. URL http://skyscraperpage.com/diagrams/?searchID=85671391.

[47] Skyscraper Source Media. Tallest buildings in the world, 2019. URL http://skyscraperpage.com/diagrams/?searchID=206.

[48] T.H.G. Megson. *Structural and Stress Analysis (2nd Edition)*. Elsevier, 2005. ISBN 978-0-7506-6221-5. URL https://app.knovel.com/hotlink/toc/id:kpSSAE0005/structural-stress-analysis/structural-stress-analysis.

[49] Anthony George Maldon Michell. Lviii. the limits of economy of material in frame-structures. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 8(47):589–597, 1904.

[50] Giulia Milana, Konstantinos Gkoumas, and Franco Bontempi. Sustainability concepts in the design of high-rise buildings: the case of diagrid systems. *Design in Civil and Environmental Engineering*, page 170, 2014.

[51] Kyoung-Sun Moon, Jerome J Connor, and John E Fernandez. Diagrid structural systems for tall buildings: characteristics and methodology for preliminary design. *The structural design of tall and special buildings*, 16(2):205–230, 2007.

[52] APS Mosek. The mosek optimization software. *Online at http://www. mosek. com*, 54 (2-1):5, 2010.

[53] R. Nijsse. *Dictaat draagconstructies II*. TU Delft, 2014. Artikelnummer 06917410004.

[54] Municipality of Den Haag. Haagse hoogbouw eyeline en skyline, 2017.

[55] Panos Y Papalambros and Douglass J Wilde. *Principles of optimal design: modeling and computation*. Cambridge university press, 2000.

[56] photorator. Shanghai tower under construction nov, 2019. URL https://photorator.com/photo/9852/shanghai-tower-under-construction-nov-.

[57] M.H. Rafiei and H. Adeli. Sustainability in highrise building design and construction. *The Structural Design of Tall and Special Buildings*, 25(13):643–658, 2016.

[58] B. Roberts. Converted housing. a solution to the housing shortage?, 2019. URL https://www.hollandtimes.nl/articles/national/converted-housing-a-solution-to-the-housing-shortage/.

[59] George IN Rozvany. Aims, scope, methods, history and unified terminology of computer-aided topology optimization in structural mechanics. *Structural and Multidisciplinary optimization*, 21(2):90–108, 2001.

[60] D. Rusakevičius and R. Belevičius. Optimization of laminated bending plates. *Journal of Civil Engineering and Management*, 8(3):143–152, 2002.

[61] O. Sigmund. Topology optimization: a tool for the tailoring of structures and materials. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 358(1765):211–227, 2000.

[62] Lauren L Stromberg, Alessandro Beghini, William F Baker, and Glaucio H Paulino. Topology optimization for braced frames: combining continuum and beam/column elements. *Engineering Structures*, 37:106–124, 2012.

[63] L.L. Stromberg, A. Beghini, W.F. Baker, and G.H. Paulino. Application of layout and topology optimization using pattern gradation for the conceptual design of buildings. *Structural and Multidisciplinary Optimization*, 43(2):165–180, 2011.

[64] Y. Sui and X. Peng. *Modelling, solving and application for topology optimization of continuum structures: ICM method based on step function*. Butterworth-Heinemann, 2017.

[65] Evangelos Tyflopoulos, David Tollnes Flem, Martin Steinert, and Anna Olsen. State of the art of generative design and topology optimization and potential research needs. *DS 91: Proceedings of NordDesign 2018, Linköping, Sweden, 14th-17th August 2018 DESIGN IN THE ERA OF DIGITALIZATION*, 2018.

[66] Capital value and abf research. De woning(beleggings)markt in beeld 2019, 2019.

[67] Ali R Yildiz and Kazuhiro Saitou. Topology synthesis of multicomponent structural assemblies in continuum domains. *Journal of Mechanical Design*, 133(1):011008, 2011.

[68] Virginia Young, Osvaldo M Querin, GP Steven, and YM Xie. 3d and multiple load case bi-directional evolutionary structural optimization (beso). *Structural optimization*, 18(2-3): 183–192, 1999.

[69] Zandbelt&vandenBerg. Een studie naar nederlandse hoogbouwcultuur. Rotterdam, 2008. ISBN: 978-90-809293-4-0.

[70] M Zhou and GIN Rozvany. On the validity of eso type methods in topology optimization. *Structural and Multidisciplinary Optimization*, 21(1):80–83, 2001.

[71] J. Zils and J. Viis. An introduction to high rise design. *Structure Magazine*, 2003.

A

High-rise definitions according to Dutch municipalities

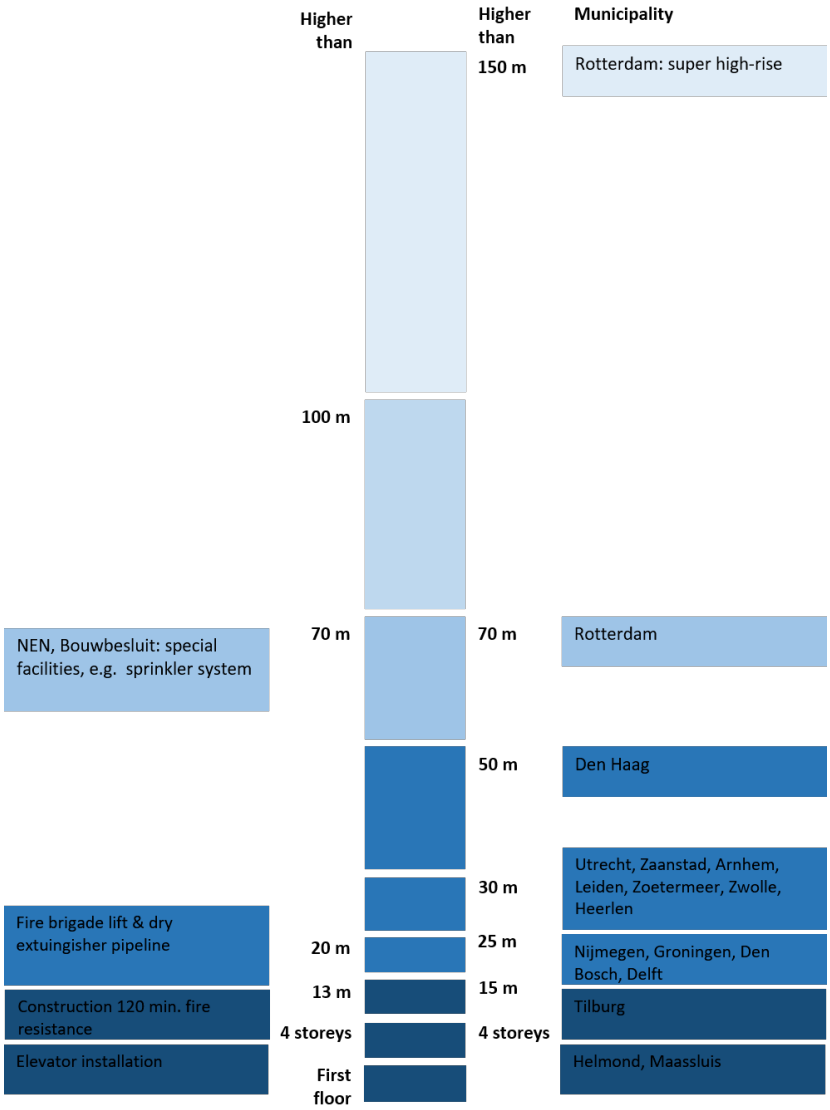


Figure A.1: High-rise definitions with some obligations according to NTA 4614-1 (October 2012) adapted from [69]

B

Requirements

This appendix is an addition of Section 2.3. Detailed derivation of the equations in Section 2.3 are described here. The equations are visualised in Section B.3. At last, the requirements which are not taken into account in this research for a high-rise building are stated in B.4.

B.1. Slenderness limit

The calculation of the slenderness limit is described in NEN-EN 1992-1-1 + C2: 2011 (paragraph 5.8.3.1)

$$\lambda_{lim} = \frac{20ABC}{\sqrt{n_{rel}}} \tag{B.1}$$

with:

A	=	$1/(1+0.2\phi_{ef})$ (if unknown $A = 0.7$ can be assumed)
B	=	$\sqrt{1+2\omega}$ (if unknown $B = 1.1$ can be assumed)
C	=	$1.7 - r_m$ (if unknown $C = 0.7$ can be assumed)
ϕ_{ef}	=	effective creep coefficient
ω	=	$A_s f_{yd}/(A_c f_{cd})$, mechanical reinforcement ratio
A_s	=	total longitudinal reinforcement area in the cross-section
A_c	=	total area of the concrete in the cross-section
f_{yd}	=	design yield strength of the reinforcement
n_{rel}	=	$N_{Ed}/(A_c f_{cd})$, relative normal force
r_m	=	M_{01}/M_{02}, moment ratio
M_{01}, M_{02}	=	first-order end moments, $\lvert M_{02}\rvert \geq \lvert M_{01}\rvert$

The values for A and B are unknown, thus taken respectively as 0.7 and 1.1. In optimization the structure is simplified to a truss structure, therefore the r_m is zero and C is equal to 1.7. The goal of the optimization is to minimize the material use, thus the material needs to be used as efficient as possible. Therefore the value of n_{rel} will be close to 1.0, which is the most conservative value.

$$\lambda_{lim} = \frac{20 \cdot 0.7 \cdot 1.1 \cdot 1.7}{\sqrt{1.0}} = 26.18 \tag{B.2}$$

B.2. Flexural buckling

As described in Paragraph 2.3.1 the compression strength of the column needs to be reduced by Φ. Φ depends on the height of the cross-section, eccentricity and the buckling length of the element. In this section is the most conservative value for Φ is derived, such that the elements will not fail on flexural buckling.

$$\Phi = 1.14\left(1 - 2\frac{e_{tot}}{h_c}\right) - 0.02\frac{l_0}{h_c} \leq 1 - 2\frac{e_{tot}}{h_c} \tag{B.3}$$

The total eccentricity is found through summing up the first-order-eccentricity and a surcharge-eccentricity.

$$e_{tot} = e_0 + e_i \tag{B.4}$$

with:

e_0 = first-order-eccentricity
e_i = surcharge-eccentricity

The surcharge-eccentricity is the second-order-eccentricity, this is taken into account with a slenderness limit described in Appendix B.1 and Paragraph 2.3.1 thus not in the flexural buckling. The first-order-eccentricity is equal to the height of the cross-section divided by 30, which always must be bigger or equal to 20 mm. In Section 2.3 the assumption is made that the buckling length is equal to the length of the member and that the cross-section is a square. Rewriting Equation B.4 gives:

$$e_{tot} = \max\left[\frac{\sqrt{a_c}}{30}; 20\right] \tag{B.5}$$

This can be implemented in the formula for Φ and simplified.

$$\Phi = 1.14\left(1 - 2\frac{\max\left[\frac{\sqrt{a_c}}{30}; 20\right]}{\sqrt{a_c}}\right) - 0.02\frac{l_m}{\sqrt{a_c}} \leq \left(1 - 2\frac{\max\left[\frac{\sqrt{a_c}}{30}; 20\right]}{\sqrt{a_c}}\right) \tag{B.6}$$

$$\Phi = \min\left[1.14 - 0.02\frac{l_m}{\sqrt{a_c}} - \max\left[\frac{19}{250}; \frac{45.6}{\sqrt{a_c}}\right]; 1 - \max\left[\frac{1}{15}; \frac{40}{\sqrt{a_c}}\right]\right] \tag{B.7}$$

In the total eccentricity the maximal value needs to be determined. When the cross-sectional area (a_c) is greater or equal to 0.36 m^2 the $\frac{\sqrt{a_c}}{30}$ is maximum, otherwise 20 mm is maximum. The minimal value for a_c is 0.0625 m^2, due to the fire resistance. Equation B.7 is split up in two parts:

$$\Phi = \min\left[1.064 - 0.02\frac{l_m}{\sqrt{a_c}}; \frac{14}{15}\right] \quad \text{if } \sqrt{a_c} \geq 600 \text{ mm} \tag{B.8}$$

$$\Phi = \min\left[1.14 - 0.02\frac{l_m}{\sqrt{a_c}} - \frac{45.6}{\sqrt{a_c}}; 1 - \frac{40}{\sqrt{a_c}}\right] \quad \text{if } 250 \leq \sqrt{a_c} \leq 600 \text{ mm} \tag{B.9}$$

The minimal value for Φ in Equation B.8 and B.9 is found if the ratio of the l_m and $\sqrt{a_c}$ is maximal. The maximal ratio can be deduced, by rewriting Equation 2.5.

$$\frac{l_m}{\sqrt{a_c}} \leq 7.56 \tag{B.10}$$

Implemented in the formula for Φ:

$$\Phi = \min\left[0.91; 0.93\right] = 0.91 \quad \text{if } \sqrt{a_c} \geq 600 \text{ mm} \tag{B.11}$$

$$\Phi = \min\left[0.99 - \frac{45.6}{\sqrt{a_c}}; 1 - \frac{40}{\sqrt{a_c}}\right] \quad \text{if } 250 \leq \sqrt{a_c} \leq 600 \text{ mm} \tag{B.12}$$

The minimal value for Equation B.12 is found by minimising $\sqrt{a_c}$. As the fire requirements prescribe the minimal value is 250 mm.

$$\Phi = 0.91 \quad \text{if } \sqrt{a_c} \geq 600 \text{ mm} \tag{B.13}$$

$$\Phi = 0.81 \quad \text{if } 250 \leq \sqrt{a_c} \leq 600 \text{ mm} \tag{B.14}$$

Concluding, elements which have a width or height smaller than 600 mm are not uncommon in high-rise buildings. Therefore the assumption for the reduction factor Φ is 0.81.

B.3. Visualisation

In this section all limits on the cross-sectional area, written as equation in Section 2.3, are visualized.

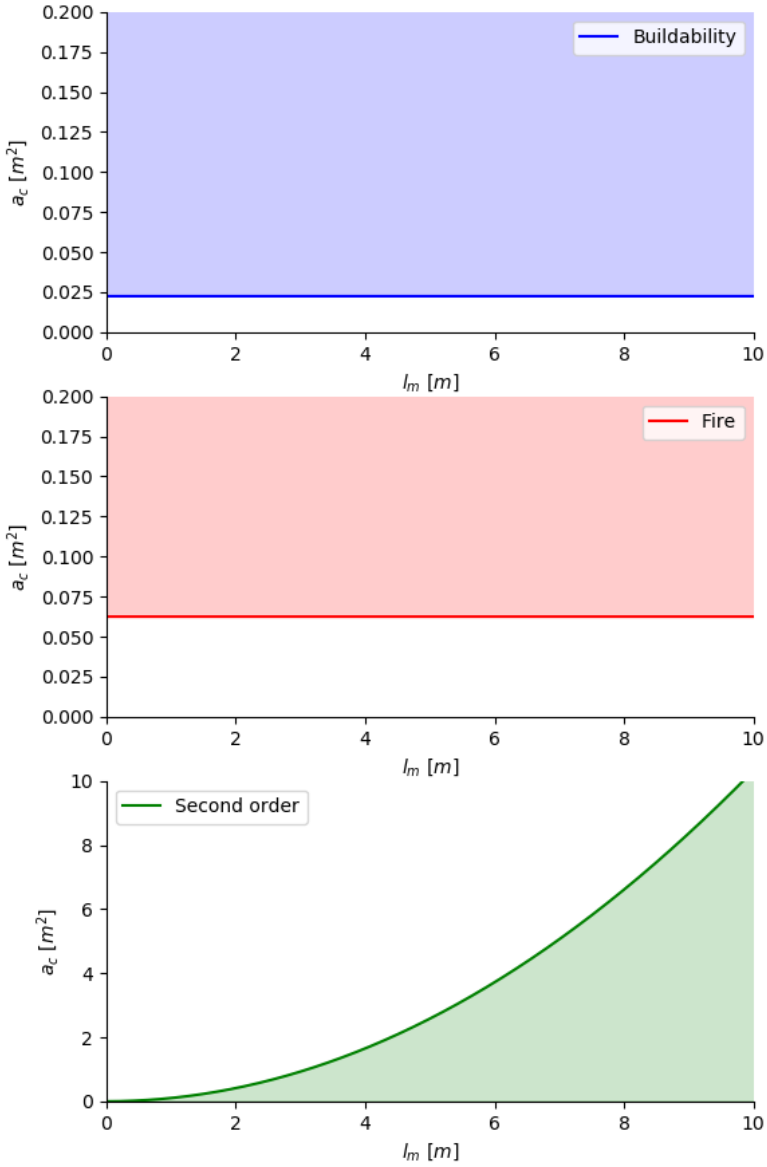


Figure B.1: The shaded areas in the graphs are the combinations of lengths and cross-section areas which are allowed according to requirements following from buildability (top), fire resistance (center), second-order-effects (bottom)

When all three requirements are active an area gives the possible combinations of the length of the member and area of the cross-section, shown in Figure B.2.

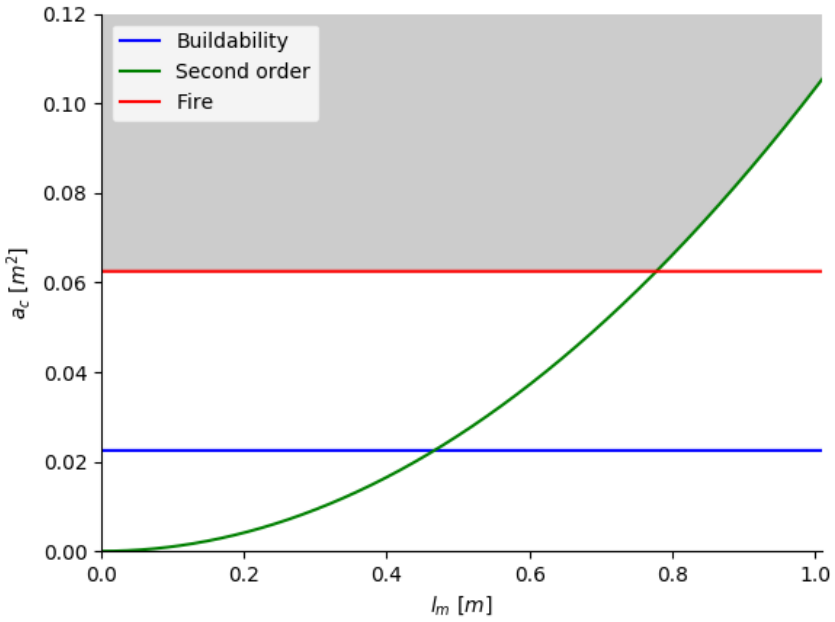


Figure B.2: All limits on the cross-sectional area implemented in one graph and grey shaded what is allowable according to all three limits

B.4. Other requirements

In this appendix, an insight into the requirements for high-rise buildings, according to NTA 4614, can be found. Most of these requirements are not mentioned in Section 2.3 and are not taken into account in this research. The six sections of the NTA 4614 are shortly discussed, the first part is the general part, from which the most important demands are:

- A high-rise building belongs to consequence class 3;
- The high-rise building must add value to the environment, which can be done by being a landmark or orientation point;
- It is recommended that the main load-bearing structure is built for a life span of 100 years;
- The building generates new traffic flows, which should be integrated into the existing traffic flows;

In the second part, the demands concerning evacuation of the high-rise building with stairs and elevators are stated. Such as the maximum evacuation time is 60 min (only with stairs, only with elevators, or a combination of stairs and elevators), assuming the main load-bearing structure withstands the fire for at least 120 min. In the NTA 4641-3, the structural safety is discussed, for example the following requirements:

- The main load-bearing structure should be able to withstand the fire for at least 120 min;
- There must be a second load path, such that the building doesn't collapse when the first load path is damaged or unavailable;
- There must be in- and external control during the build of the main load-bearing construction;
- The influence on the surrounding buildings should be investigated, this involves among other light incidence and wind loads;

As mentioned in Paragraph 2.3.1, (extraordinary) foreseen and unforeseen dangerous events need to be considered in a systematic risks analysis. For each extraordinary event, the required reliability level and depth of calculations need to be determined. Therefore, it might be necessary to use non-linear models and/or dynamic calculations to perform the systematic risks analysis. Designing for extraordinary events should be addressed in two manners:

- based on known extraordinary load (e.g. explosions and impact load) and
- based on limiting the extent of local collapse.

The structure should be designed, such that the know extraordinary loads are resisted. The building should be robust enough. Also, a non-structural approach is possible, limiting or preventing the load from happening by safety measurements. If an unforeseen load acts on the structure, whereby local structural elements fail, the structure should still function. This can be achieved by designing an alternative load path or using tension rods. Next to that, key elements need to be designed to withstand imaginary extraordinary loads (defined in the NEN-EN 1991-1-7+C1+A1: 2015).

In the fourth part of the NTA 4641 the demands to the elevator(s) are stated, among others the functionality, comfort, energy use and traffic handling. Part five discusses the facade and the maintenance of the facade. Which states the demands regarding wind pressure, water tightness, sound insulation, fire spread, etc. Next to that, there are demands to the installations which are used for the maintenance of the facade. For example, how the electricity and water are supplied during the maintenance of the facade. Another important aspect of high-rise buildings is discussed in part 5, namely that the shape and/or dimensions of the building can disturb radar signals. This should be taken into account during the design, especially when the building will be realised close to airport and radio towers.

The sixth and last part of the NTA 4641 discusses the building integrated installations. Installations such as a lightning conductor (in- and external part), sewerage, water supply system for drinking water and the fire department. Also, a control list for the following points is included;

- Rainwater drainage installation;
- Wastewater installation;
- Gas installation;
- Heating installation;
- Ventilation, cooling and air treatment installation;
- Security installation.

C

Floor plans

In this Appendix the 3D floor plans of several floors of the tower Seattle are shown. Might the reader be interested in more detailed drawings of the case study, one can contact the author which can supply the drawings or redirect you to organisations which can supply these. The more detailed drawings with dimensions are not included in this report, because of the readability. The 3D floor plan of the 6^th, 7^{th}, $12^{th} - 21^{th}$, 22^{th}, 23^{th} floor and the roof are included in this appendix. The 3D floor plan for $8^{th} - 11^{th}$ floor can be found in Figure 4.2. Just as all the information about the case study, this information is supplied by Zonneveld Ingenieurs.

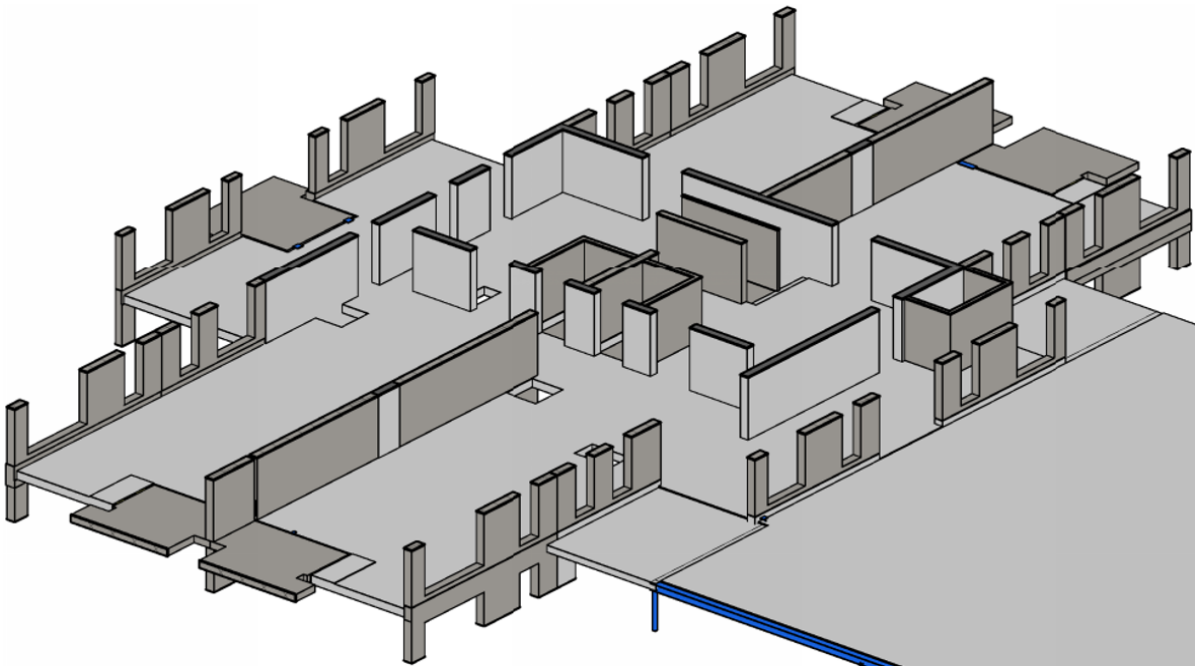


Figure C.1: 3D view of the 6^{th} floor of tower Seattle, on the right the roof of the low-rise can be seen

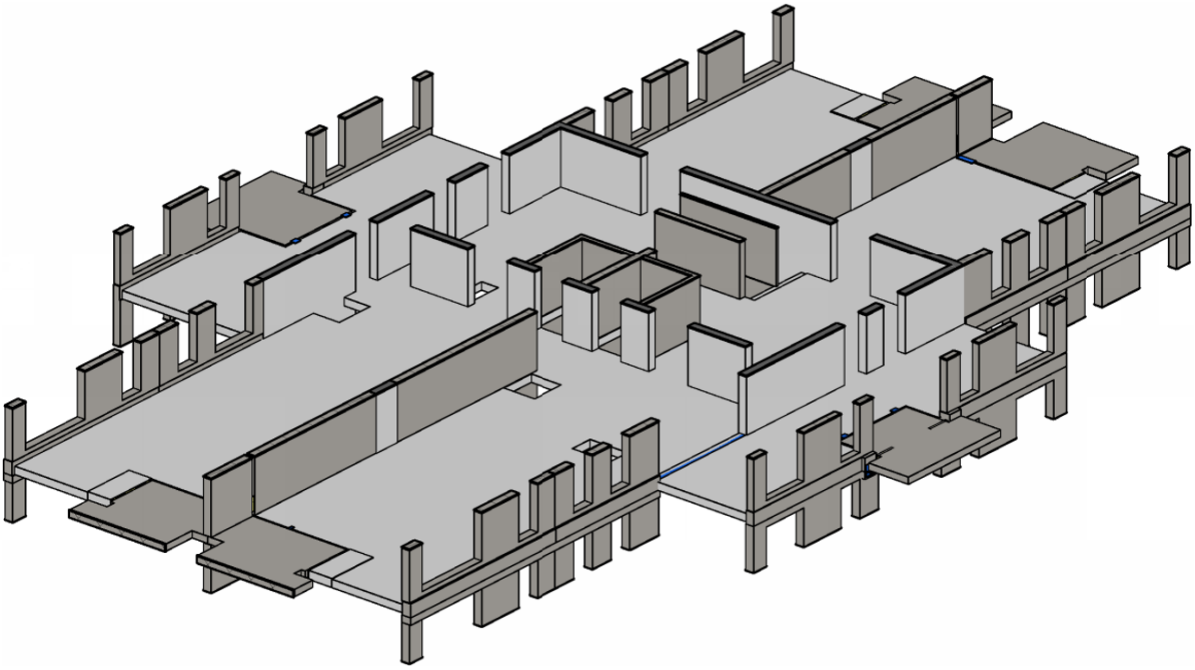


Figure C.2: 3D view of the 7^{th} floor of tower Seattle

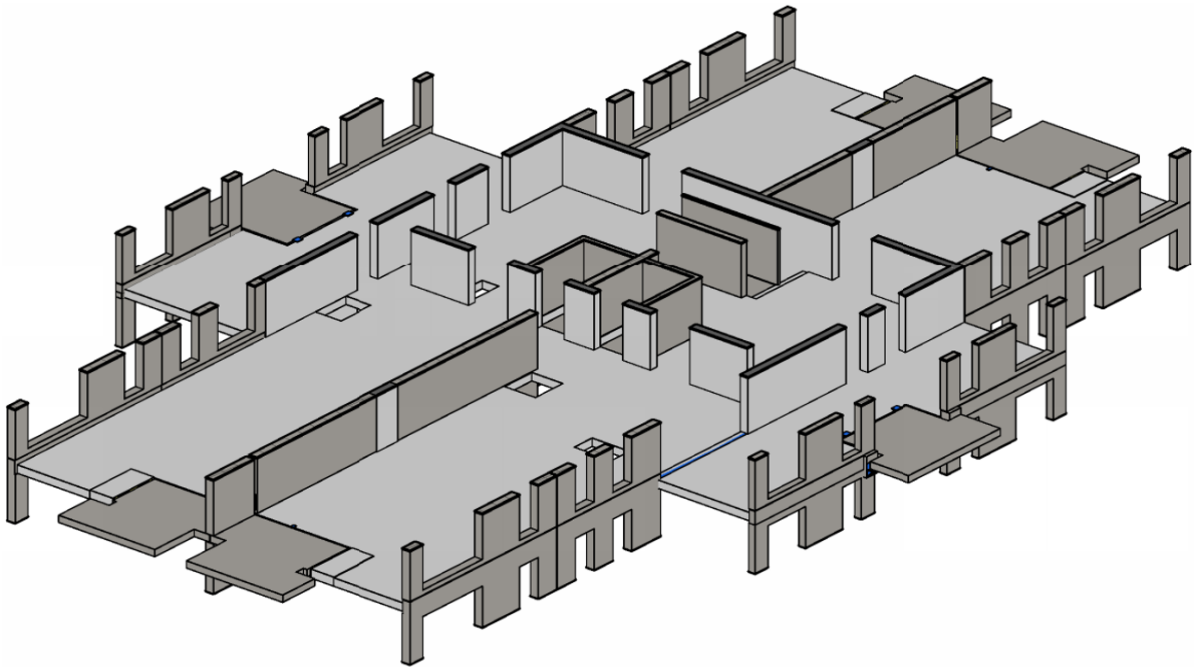


Figure C.3: 3D view of the 12^{th} till 21^{th} floor of tower Seattle

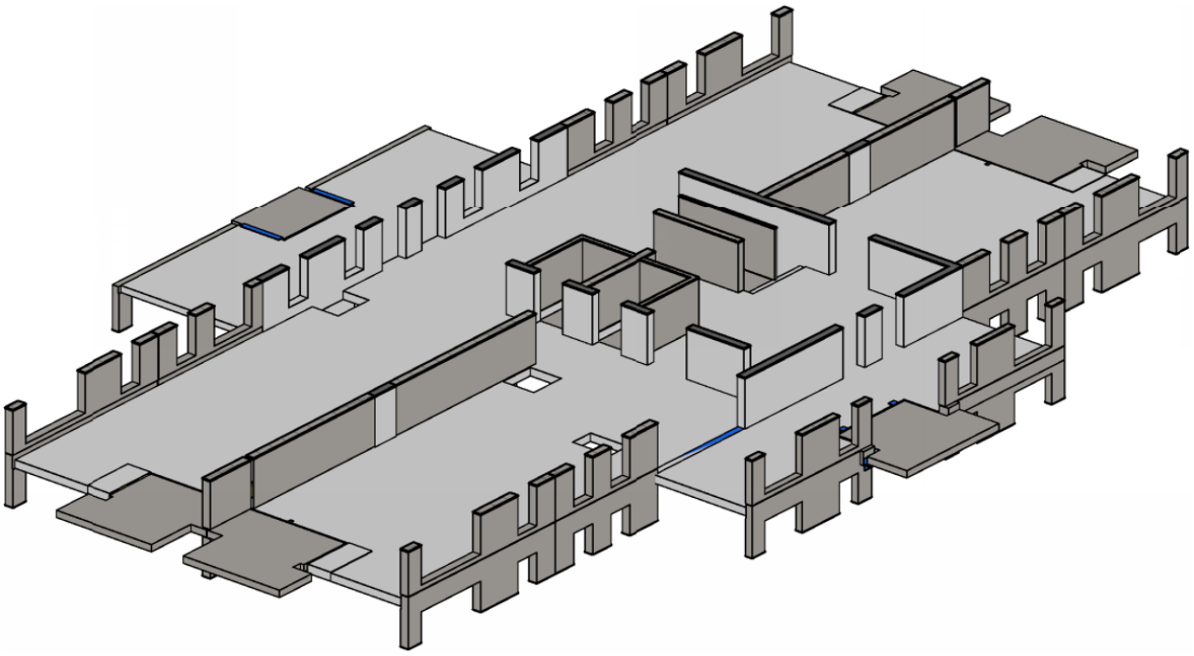


Figure C.4: 3D view of the 22^{th} floor of tower Seattle

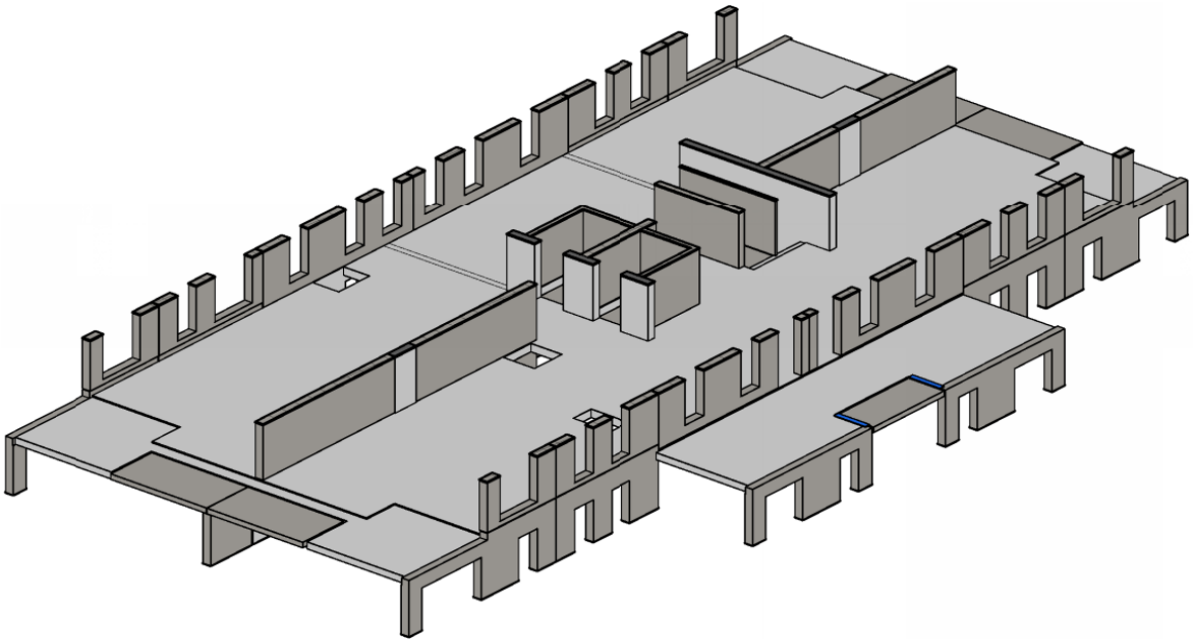


Figure C.5: 3D view of the 23^{th} floor of tower Seattle

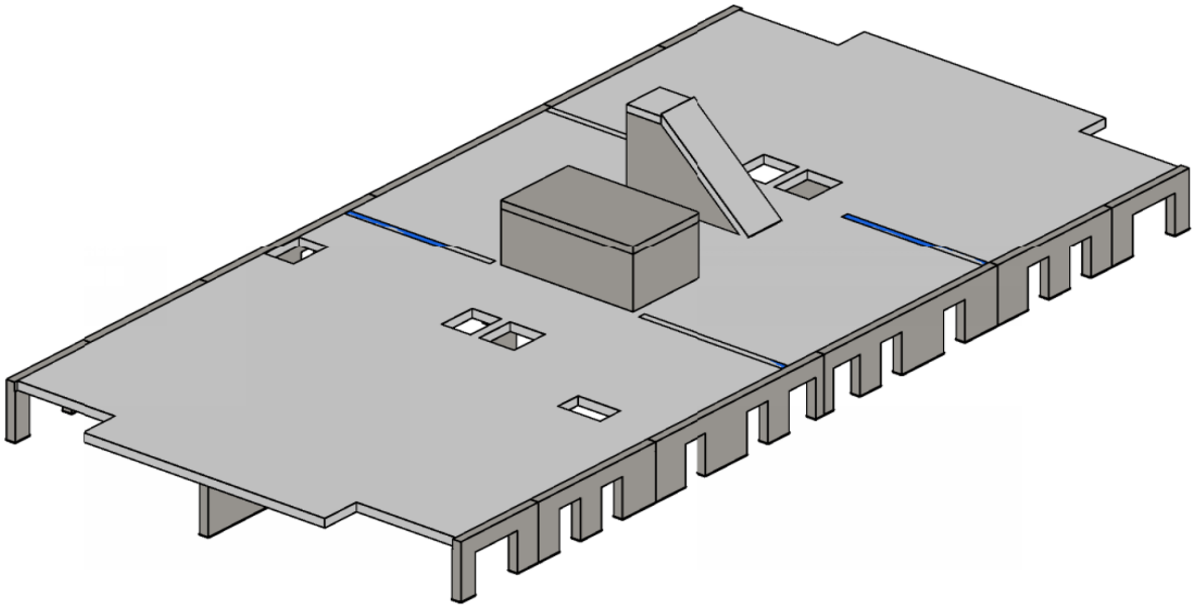


Figure C.6: 3D view of the roof of tower Seattle

D

Differentiation of Lagrange function

The Lagrange function consists of two parts, the summation, which leads to the total volume of the structure and the difference between the actual displacement and allowable displacement.

$$\mathcal{L}(a_i) = \sum_{i=1}^{m} l_i a_i + z_1 \left(\left| \sum_{i=1}^{m} q_{u,i} \frac{q_i l_i}{E a_i} \right| - \frac{h}{750} \right) \tag{D.1}$$

To differentiate the Lagrange both parts can be differentiated separately and summed up. The first part is differentiated as follows:

$$\sum_{i=1}^{m} l_i a_i = (l_1 a_1 + l_2 a_2 + \cdots + l_m a_m) \tag{D.2}$$

$$\frac{\partial}{\partial a_1} \sum_{i=1}^{m} l_i a_i = l_1 \tag{D.3}$$

$$\frac{\partial}{\partial a_2} \sum_{i=1}^{m} l_i a_i = l_2 \tag{D.4}$$

Therefore the following can be concluded:

$$\frac{\partial}{\partial a_i} \sum_{i=1}^{m} l_i a_i = l_i \tag{D.5}$$

The second part of the Lagrange function consists of a summation within an absolute function, therefore the chain rule is used. First the second part is rewritten:

$$z_1 \left(\left| \sum_{i=1}^{m} q_{unit,i} \frac{q_i l_i}{E a_i} \right| - \frac{h}{750} \right) \Leftrightarrow z_1 \left| \sum_{i=1}^{m} q_{u,i} \frac{q_i l_i}{E a_i} \right| - z_1 \frac{h}{750} \tag{D.6}$$

z_1 is the Lagrange multiplier and a constant, so $z_1 \frac{h}{750}$ is a constant and will disappear when differentiating.

$$\frac{\partial}{\partial a_i}\left[z_1\left|\sum_{i=1}^{m} q_{u,i}\frac{q_i l_i}{E a_i}\right|\right] \Leftrightarrow z_1 \frac{\partial |u|}{\partial u}\frac{\partial u}{\partial a_i} \tag{D.7}$$

$$\text{with} \quad u = \sum_{i=1}^{m} q_{u,i}\frac{q_i l_i}{E a_i} \quad \text{and} \quad \frac{\partial |u|}{\partial u} = \frac{u}{|u|}$$

From equation D.7 the derivative of u is deduced.

$$\frac{\partial u}{\partial a_i}\sum_{i=1}^{m} q_{u,i}\frac{q_i l_i}{E a_i} = -q_{u,i}\frac{q_i l_i}{E a_i^2} \tag{D.8}$$

Combining all information gives the derivative of the second part.

$$\frac{\partial}{\partial a_i}\left[z_1\left|\sum_{i=1}^{m} q_{u,i}\frac{q_i l_i}{E a_i}\right|\right] \Leftrightarrow -z_1\frac{u}{|u|}q_{u,i}\frac{q_i l_i}{E a_i^2} \tag{D.9}$$

Concluding, the derivative of the Lagrange function is:

$$\frac{\partial \mathcal{L}(a_i)}{\partial a_i} = l_i - z_1\frac{g^t}{|g^t|}q_{u,i}\frac{q_i l_i}{E a_i^2} \tag{D.10}$$

$$\text{with} \quad g^t = \sum_{i=1}^{m} q_{u,i}\frac{q_i l_i}{E a_i}$$

g^t is the Left-hand side of the inequality constraint, concerning the top floor displacement.

E

Verification of optimization code

This appendix is an extension of Paragraph 5.3 and consists of extra information over the 2D cantilever problem, the relaxation parameter, the 3D cantilever problem, a comparison with Grasshopper plugin, the Excel output file, post-processing and the rigid diaphragm action of floors.

E.1. 2D cantilever beam

In the article of Kwok, two different optimization methods are used to analyse the cantilever problem (Figure 5.4a). In both the grid is refined to see what influence this has, all solutions are similar to the Michell truss and shown in Figure E.1 [37].

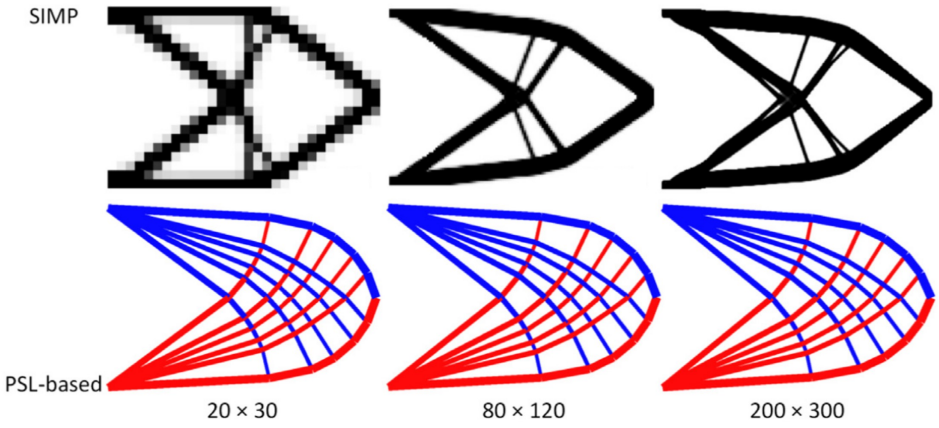


Figure E.1: Solutions according to SIMP and PSL-based optimization methods [37], the dimensions are stated below each figure ($20x30$ means that the vertical axis is divided in 20 elements and the horizontal axis is divided in 30 elements)

Martinez also investigated the effect of grid refined, the results of his research are shown in Figure E.2 [45].

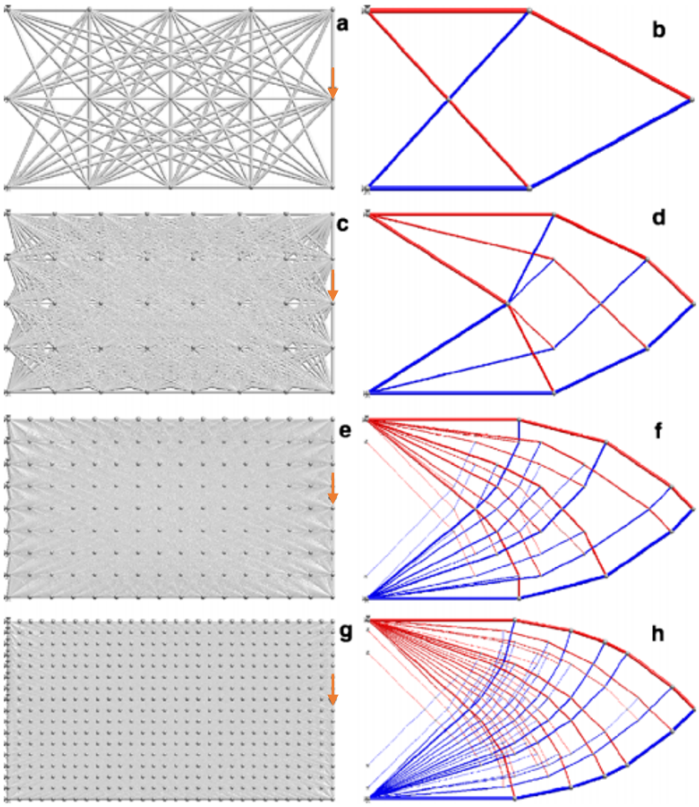


Figure E.2: A cantilever beam with a point load on the end (right side) and supports on the left side. Figure a,c,e and g show the initial ground structure with increasing nodes and b, d, f and h show the corresponding optimal solution [45]

The result of this research is displayed in Figure E.3 and has the size of $20x30$ meter with the distance between the nodes is 1 meter. During this optimization the self-weight of the elements and the lower limit for the surface of a member are not taken into account.

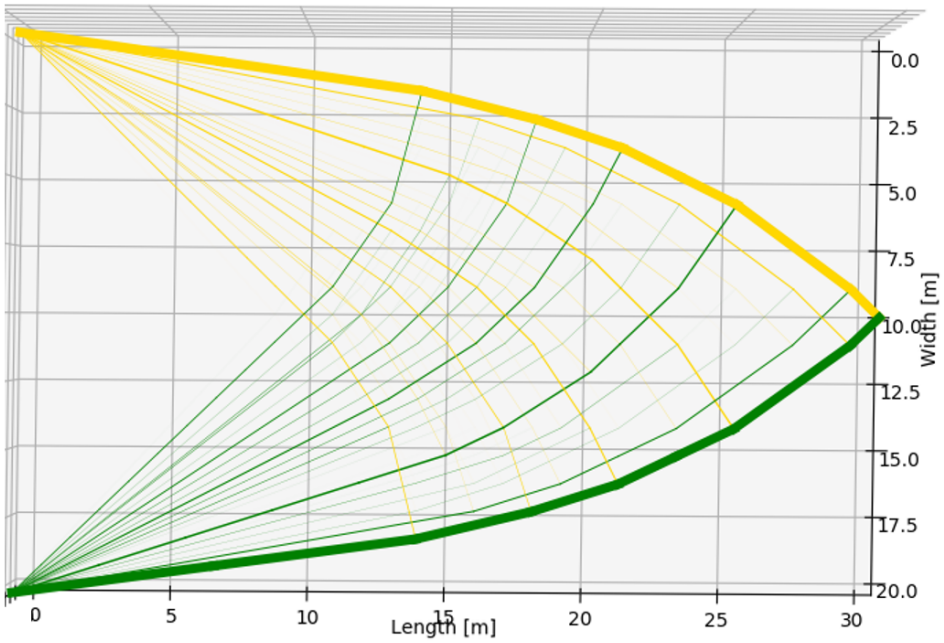


Figure E.3: Solution to the cantilever problem in 2D with the use of the developed code

E.2. Relaxation parameter

To investigate the influence of the relaxation parameter on the rate of convergence, optimizations for a 3D cantilever (Paragraph 5.3.3) and a high-rise building (2D) (the problem statement is shown in Figure 5.11) are executed for several different relaxation parameters. The maximum displacement for the 3D cantilever is 12 mm and for the 2D high-rise building 48 mm. The results are shown in graphs, the cantilever in Figure E.4 and the high-rise building in E.5. Both figures show that for multiple values of the relaxation parameter the displacement after optimization is equal to the (or real close to) their maximum displacement. Also, for the high-rise building no solution could be found with a positive relaxation parameter. Therefore only negative values are used. For the lower values of the relaxation parameter the displacements (for the high-rise building) are random. When the relaxation parameter is smaller than approximately −1.5 smoother curves arise, just as in the graph of the cantilever. There is no clear pattern in this, thus for now it is assumed that the value for the relaxation parameter is 5. Trail and error during optimization should show if this is a correct assumption.

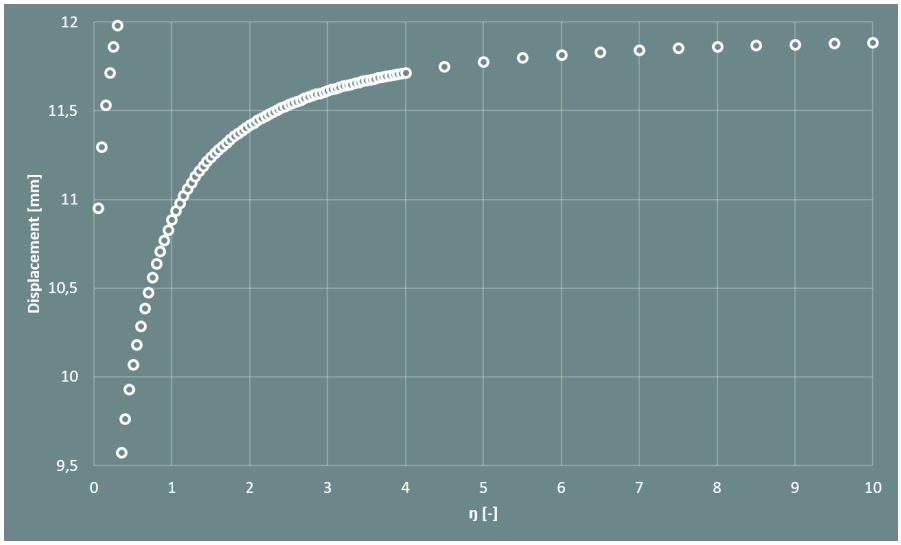


Figure E.4: Relaxation parameter vs. displacement after optimization for a 3D cantilever (described in Paragraph 5.3.3).

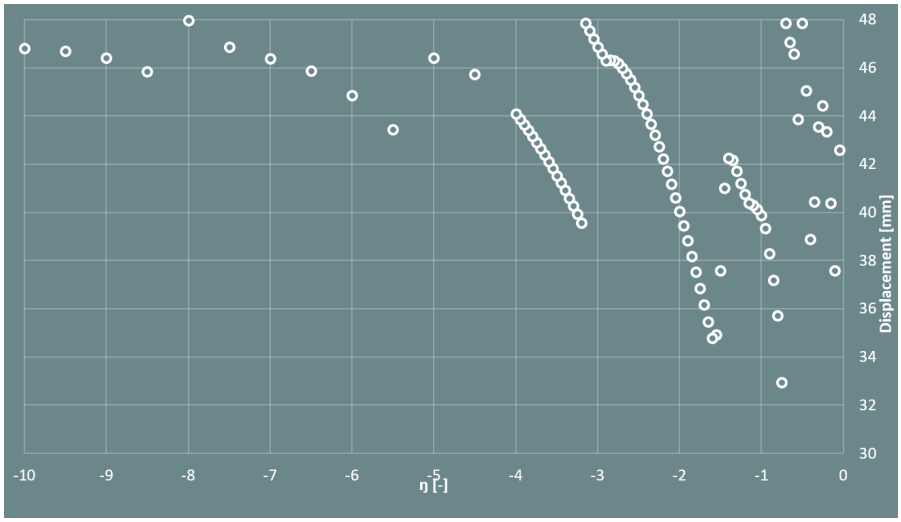


Figure E.5: Relaxation parameter vs. displacement after optimization for a 2D high-rise building (shown in Figure 5.11).

E.3. 3D cantilever beam

In this section some supporting information for the 3D cantilever beam problem is given. In Table E.1 the volume per iteration step is given, the first column shows the iteration step, the second the volume in each iteration when self-weight is excluded and the third shows the volume in each iteration when self-weight is included.

Table E.1: Volume per iteration for the 3D cantilever beam problem with self-weight excluded and included

iteration	Volume (excluded) [m^3]	Volume (included) [m^3]
1	24.60000344	24.60000344
2	22.00761907	23.46364921
3	20.84398293	22.34595427
4	20.64154029	22.0370592
5	20.53621435	21.89095809
6	20.53010305	21.85966795
7	20.52148692	21.79604521
8	20.52148624	21.84239761
9	-	21.84684469
10	-	21.84686674

The table clearly shows that, due to the inclusion of self-weight the volume is increased and that there are more iterations needed to find the optimal volume.

E.4. Grasshopper plugin Peregrine

As part of the UK government-funded BUILD-OPT research project Peregrine has been developed [42]. In Figure E.6 an example of an optimization with this plugin is shown.

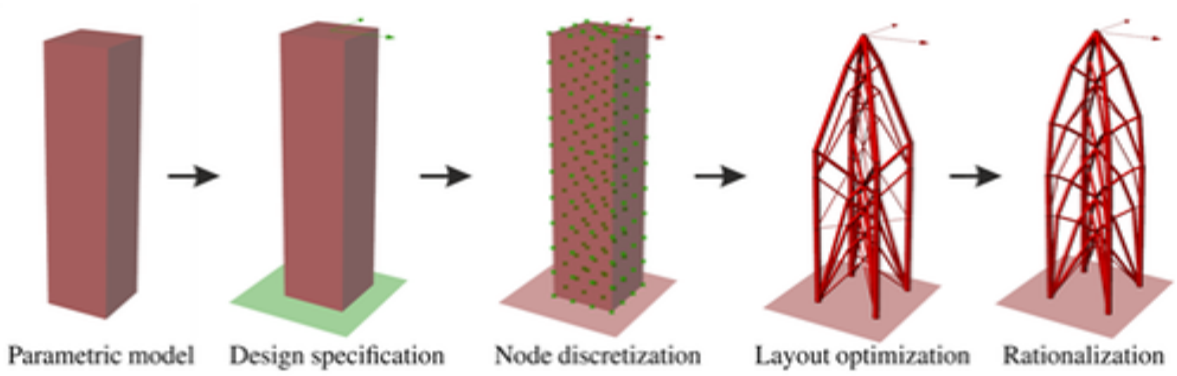


Figure E.6: Steps in the structural layout optimization plugin Peregrine [42]

As concluded in Paragraph 5.3, the shape of the Michell truss or the diamond shape is a returning shape during different optimizations. Also, the second to last and last image in Figure E.6 show the diamond shape of the Michell truss. Compare these two images with the solutions in Figure 5.12 and great similarities can be seen.

A note concerning the plugin, in the manual is stated that in the current version, among other things, it is not possible to prescribe limits on structural deflection and buckling is not yet considered. A workaround for both is to lower the stress limits [30].

E.5. Excel output file

The Python code creates a folder and sets the directory to this folder (user-defined), the code is shown in Figure E.7.

```
def makefolder(height, length, width):
    date = datetime.datetime.now()
    x,i = 20,0
    while i <= x:
        try:
            os.mkdir("C:\Users\Harry\.spyder/"+str(date.year)+"-"+str(date.month)+"-"+str(date.day)+\
                     '_height'+str(height)+'_length'+str(length)+'_width'+str(width)+"v"+str(i))
        except OSError:
            i += 1
        else:
            path = "C:\Users\Harry\.spyder/"+str(date.year)+"-"+str(date.month)+"-"+str(date.day)+\
            '_height'+str(height)+'_length'+str(length)+'_width'+str(width)+"v"+str(i)
            break
    os.chdir(path)
```

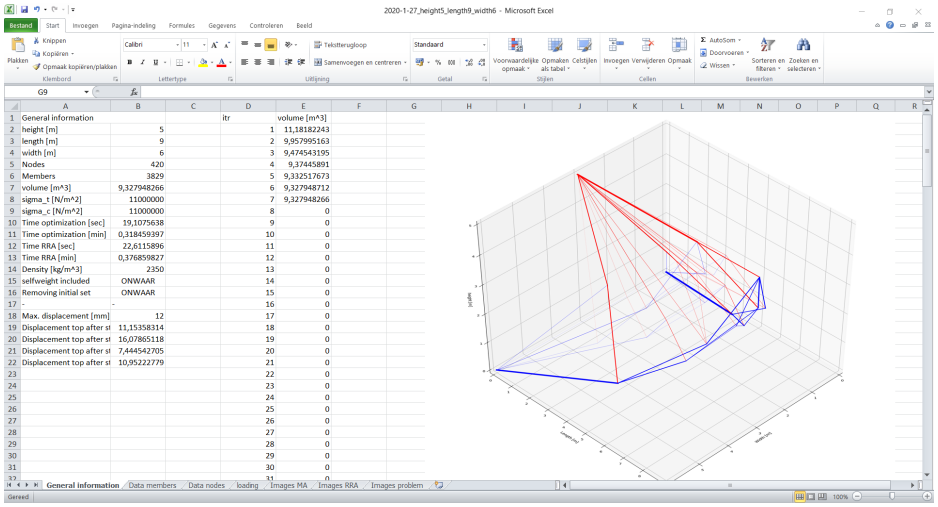
Figure E.7: Function to make folder and set directory to this folder (Python)

This function should be runned before the optimization, then all information (figures and output file) will be placed in this folder. The output file is generated with the script shown in Figure E.8.

```
def savedata(Nd, Cn, a, q, height, length, width, vol_t, vol, f, dof, a_new, sf, R, inRRA, st, sc,\
             startMA, endMA, timeplotma, startRRA, endRRA, timeplotRRA):
    date = datetime.datetime.now()
    workbook = xlsxwriter.Workbook(str(date.year)+'-'+str(date.month)+'-'+str(date.day)+'_height'+\
                                   str(height)+'_length'+str(length)+'_width'+str(width)+'.xlsx')
    worksheet1 = workbook.add_worksheet("General information")
    worksheet2 = workbook.add_worksheet("Data members")
    worksheet3 = workbook.add_worksheet("Data nodes")
    worksheet4 = workbook.add_worksheet("loading")
    worksheet5 = workbook.add_worksheet("Images MA")
    worksheet6 = workbook.add_worksheet("Images RRA")
    worksheet7 = workbook.add_worksheet("Images problem")
    worksheet1.set_column(0,0,20)
    worksheet1.set_column(1,10,12)
    worksheet2.set_column(0,1,20)
    worksheet2.set_column(2,3,6)
    worksheet2.set_column(4,10,20)
    worksheet3.set_column(0,5,8)
    worksheet7.set_column(0,0,10)
    worksheet2.freeze_panes(1,0)
    worksheet3.freeze_panes(1,0)
    worksheet4.freeze_panes(1,0)
    name = ["height [m]", "length [m]", "width [m]", "Nodes", "Members", "volume [m^3]","sigma_t [N/m^2]",\
    "sigma_c [N/m^2]", "Time optimization [sec]", "Time optimization [min]", "Time RRA [sec]",\
    "Time RRA [min]", "Density [kg/m^3]", "selfweight included", "Removing initial set", "jc", "-", \
    "Max. displacement [mm]","""Displacement top after strength optimization (no reduction for members in\
    tension) [mm]""","""Displacement top after strength optimization (with reduction for members in tension)\
    [mm]""","""Displacement top after stiffness optimization (no reduction for members in tension) [mm]""",\
    "Displacement top after stiffness optimization (with reduction for members in tension) [mm]"]
    data = [height, length, width, len(Nd), len(Cn), vol, st, sc, endMA-startMA-timeplotma, \
            (endMA-startMA-timeplotma)/60, endRRA-startRRA-timeplotRRA, (endRRA-startRRA-timeplotRRA)/60,\
            2500, sf, R, jc, "-", inRRA[0], inRRA[1], inRRA[2], inRRA[3], inRRA[4]]

    worksheet1.merge_range(0,0,0,1,"General information")
    worksheet1.write(0, 3, "itr")
    worksheet1.write(0, 4, "volume [m^3]")
    worksheet1.insert_image(1,7,"MA_3D.png")
    for i in range(len(name)):
        worksheet1.write(1+i, 0, name[i])
        worksheet1.write(1+i, 1, data[i])
    for i in range(len(vol_t)):
        worksheet1.write(1+i, 3, i+1)
        worksheet1.write(1+i, 4,vol_t[i])
    worksheet2.write(0,0, "Surface [m^2]")
    worksheet2.write(0,1, "Surface after RRA [m^2]")
    worksheet2.write(0,2, "Node 1")
    worksheet2.write(0,3, "Node 2")
    worksheet2.write(0,4, "length [m]")
    for k in range(len(q)):
        worksheet2.write(0,5+k, "q in " + str(k+1) + "'th load case [N]")
        for i in range(len(a)):
            worksheet2.write(i+1,5+k, q[k][i])
            worksheet2.write(i+1, 0 ,a[i])
            worksheet2.write(i+1,1, a_new[i])
            worksheet2.write(i+1,2, Cn[i,0])
            worksheet2.write(i+1,3, Cn[i,1])
            worksheet2.write(i+1,4, Cn[i,2])

    worksheet3.write(0,0, "Node")
    worksheet3.write(0,1, "X-value")
    worksheet3.write(0,2, "Y-value")
    worksheet3.write(0,3, "Z-value")
    for i in range(len(Nd)):
        worksheet3.write(i+1,0, i)
        worksheet3.write(i+1,1, Nd[i,0])
        worksheet3.write(i+1,2, Nd[i,1])
        worksheet3.write(i+1,3, Nd[i,2])

    worksheet4.merge_range(0,0,0,3,"Dof")
    worksheet4.write(1,0, "Node")
    worksheet4.write(1,1, "X")
    worksheet4.write(1,2, "Y")
    worksheet4.write(1,3, "Z")
    for k in range(len(q)):
        worksheet4.merge_range(0,4+(k*3),0,6+(k*3),"Load in " + str(k+1) + "'th load case [N]")
        worksheet4.write(1,4+k*3, "X")
        worksheet4.write(1,5+k*3, "Y")
        worksheet4.write(1,6+k*3, "Z")
        for i in range(len(f[0])/3):
            worksheet4.write(2+i,4+k*3, f[k][3*i])
            worksheet4.write(2+i,5+k*3, f[k][3*i+1])
            worksheet4.write(2+i,6+k*3, f[k][3*i+2])
            worksheet4.write(2+i,0,i)
            worksheet4.write(2+i,1, dof[3*i])
            worksheet4.write(2+i,2, dof[3*i+1])
            worksheet4.write(2+i,3, dof[3*i+2])
    fignameMA = ["MA_3D.png", "MA_topview.png", "MA_sideview_width.png", "MA_sideview_length.png"]
    fignameRRA = ["RRA_3D.png", "RRA_topview.png", "RRA_sideview_width.png", "RRA_sideview_length.png"]
    fignamePr = []
    for i in range(len(fignameMA)):
        worksheet5.insert_image(0,10*i,fignameMA[i])
        worksheet6.insert_image(0,10*i,fignameRRA[i])
    for j in range(np.size(f)/3/len(Nd)):
        fignamePr += ["startproblem_load_case_"+str(j+1)+"_3D.png", "startproblem_load_case_"+str(j+1)+\
                      "_topview.png", "startproblem_load_case_"+str(j+1)+"_sideview_width.png", \
                      "startproblem_load_case_"+str(j+1)+"_sideview_length.png"]
        worksheet7.write(24*j+12,0, "load case "+str(j+1))
        for i in range(len(fignameMA)):
            worksheet7.insert_image(27*j,10*i+1,fignamePr[i+j*4])
    workbook.close()
```

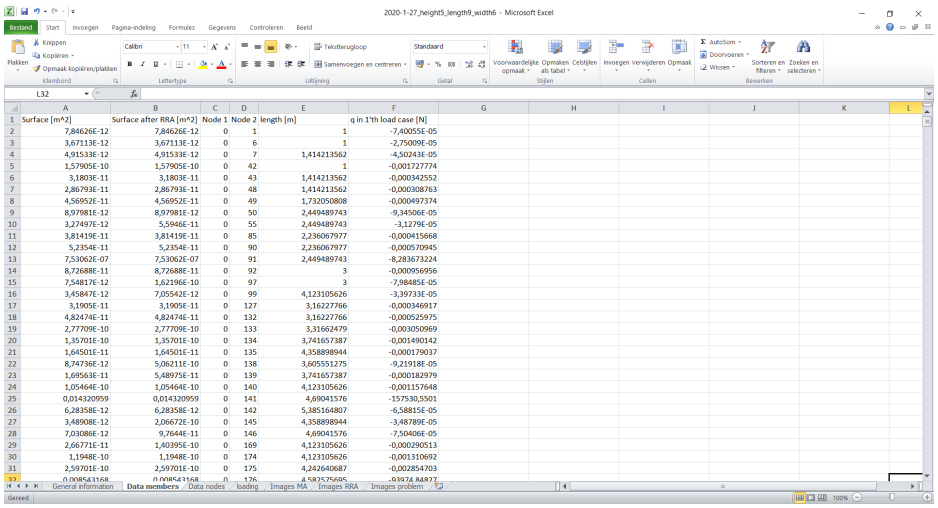
Figure E.8: Python script for generating the Excel output file

The Excel file has seven worksheets, the general information is stated on the first worksheet, including a 3D image of the optimized structure. The most information is quite straightforward, except for the ”Removing initial set”. If this parameter is True (or ”WAAR” in Dutch), it holds that there is a minimal boundary set for the surface of the members. Next to that, members can be removed from the initial set. The second worksheet shows all the information of the individual members. On the third worksheet, named ”Data nodes” the coordinates of the nodes are stated. The fourth worksheet contains information about the loading and the degrees of freedom of each node. The last three worksheets contain images of the solution after the strength part, stiffness part and an image of the initial problem. In all three worksheets, there are 2 side-views, 1 top-view and 1 3D-view. Figure E.9 up and till E.15 show examples of these worksheets. This output file can be used to post-process the found solution to the optimization problem. An option could be to plot the interactive figures again in Python or to load the solution into Abaqus for extra analysis. The script for loading it into Python and plotting is given in Appendix E.6.



1	General information		itr	volume [m^3]	
2	height [m]	5	1	11,18182243	
3	length [m]	9	2	9,957995163	
4	width [m]	6	3	9,474543195	
5	Nodes	420	4	9,37445891	
6	Members	3829	5	9,332517673	
7	volume [m^3]	9,327948266	6	9,327948712	
8	sigma_t [N/m^2]	11000000	7	9,327948266	
9	sigma_c [N/m^2]	11000000	8	0	
10	Time optimization [sec]	19,1075638	9	0	
11	Time optimization [min]	0,318459397	10	0	
12	Time RRA [sec]	22,6115896	11	0	
13	Time RRA [min]	0,376859827	12	0	
14	Density [kg/m^3]	2350	13	0	
15	selfweight included	ONWAAR	14	0	
16	Removing initial set	ONWAAR	15	0	
17	-	-	16	0	
18	Max. displacement [mm]	12	17	0	
19	Displacement top after st	11,15358314	18	0	
20	Displacement top after st	16,07865118	19	0	
21	Displacement top after st	7,444542705	20	0	
22	Displacement top after st	10,95222779	21	0	
23			22	0	
24			23	0	
25			24	0	
26			25	0	
27			26	0	
28			27	0	
29			28	0	
30			29	0	
31			30	0	
32			31	0	

Figure E.9: Excel output file worksheet 1



	A	B	C	D	E	F
1	Surface [m^2]	Surface after RRA [m^2]	Node 1	Node 2	length [m]	q in 1'th load case [N]
2	7,84626E-12	7,84626E-12	0	1	1	-7,40055E-05
3	3,67113E-12	3,67113E-12	0	6	1	-2,75009E-05
4	4,91533E-12	4,91533E-12	0	7	1,414213562	-4,50243E-05
5	1,57905E-10	1,57905E-10	0	42	1	-0,001727774
6	3,1803E-11	3,1803E-11	0	43	1,414213562	-0,000342552
7	2,86793E-11	2,86793E-11	0	48	1,414213562	-0,000308763
8	4,56952E-11	4,56952E-11	0	49	1,732050808	-0,000497374
9	8,97981E-12	8,97981E-12	0	50	2,449489743	-9,34506E-05
10	3,27497E-12	5,5946E-11	0	55	2,449489743	-3,1279E-05
11	3,81419E-11	3,81419E-11	0	85	2,236067977	-0,000415668
12	5,2354E-11	5,2354E-11	0	90	2,236067977	-0,000570945
13	7,53062E-07	7,53062E-07	0	91	2,449489743	-8,283673224
14	8,72688E-11	8,72688E-11	0	92	3	-0,000956956
15	7,54817E-12	1,62196E-10	0	97	3	-7,98485E-05
16	3,45847E-12	7,05542E-12	0	99	4,123105626	-3,39733E-05
17	3,1905E-11	3,1905E-11	0	127	3,16227766	-0,000346917
18	4,82474E-11	4,82474E-11	0	132	3,16227766	-0,000525975
19	2,77709E-10	2,77709E-10	0	133	3,31662479	-0,003050969
20	1,35701E-10	1,35701E-10	0	134	3,741657387	-0,001490142
21	1,64501E-11	1,64501E-11	0	135	4,358898944	-0,000179037
22	8,74736E-12	5,06211E-10	0	138	3,605551275	-9,21918E-05
23	1,69563E-11	5,48975E-11	0	139	3,741657387	-0,000182979
24	1,05464E-10	1,05464E-10	0	140	4,123105626	-0,001157648
25	0,014320959	0,014320959	0	141	4,69041576	-157530,5501
26	6,28358E-12	6,28358E-12	0	142	5,385164807	-6,58815E-05
27	3,48908E-12	2,06672E-10	0	145	4,358898944	-3,48789E-05
28	7,03086E-12	9,7644E-11	0	146	4,69041576	-7,50406E-05
29	2,66771E-11	1,40395E-10	0	169	4,123105626	-0,000290513
30	1,1948E-10	1,1948E-10	0	174	4,123105626	-0,001310692
31	2,59701E-10	2,59701E-10	0	175	4,242640687	-0,002854703
32	0,008543168	0,008543168	0	176	4,582575695	-93974,84877

Figure E.10: Excel output file worksheet 2

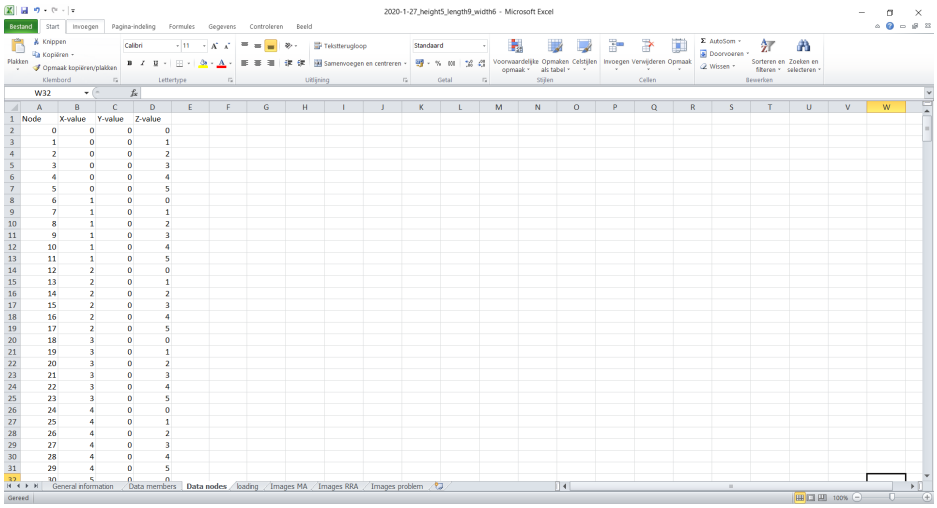


Figure E.11: Excel output file worksheet 3

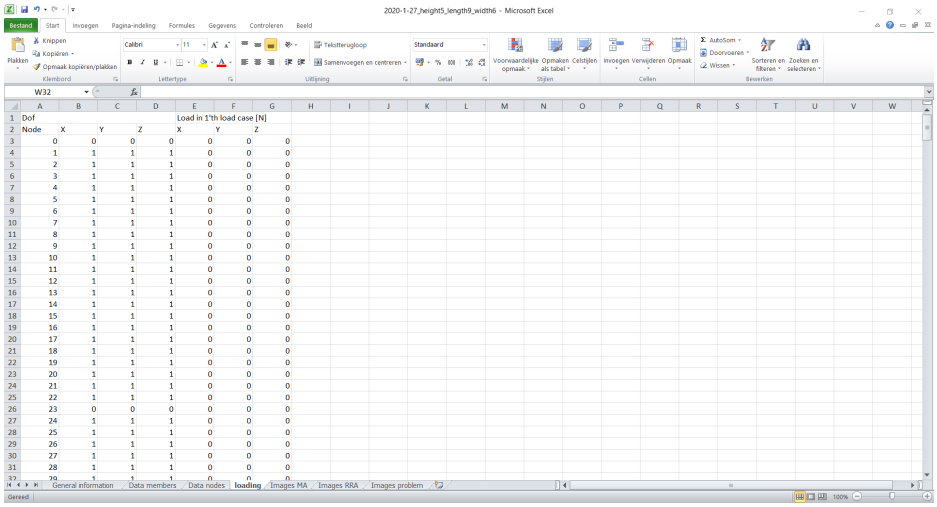


Figure E.12: Excel output file worksheet 4

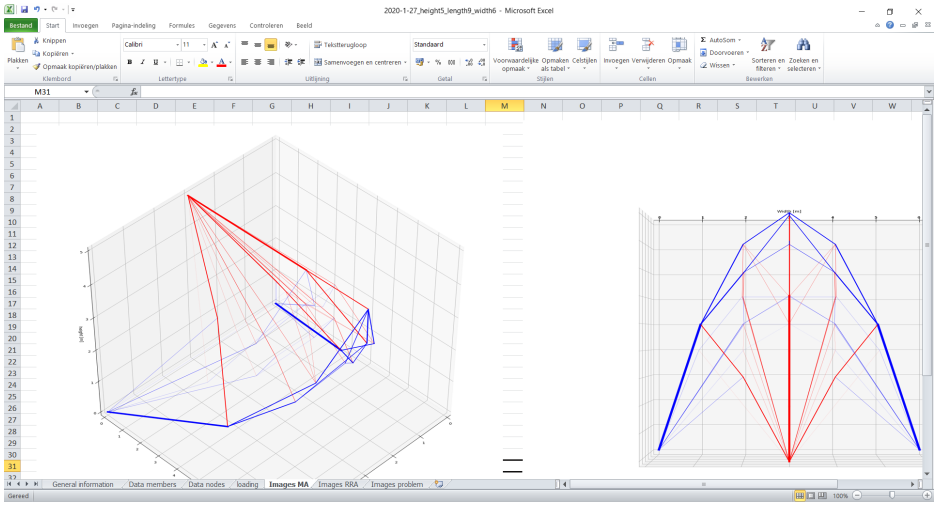


Figure E.13: Excel output file worksheet 5

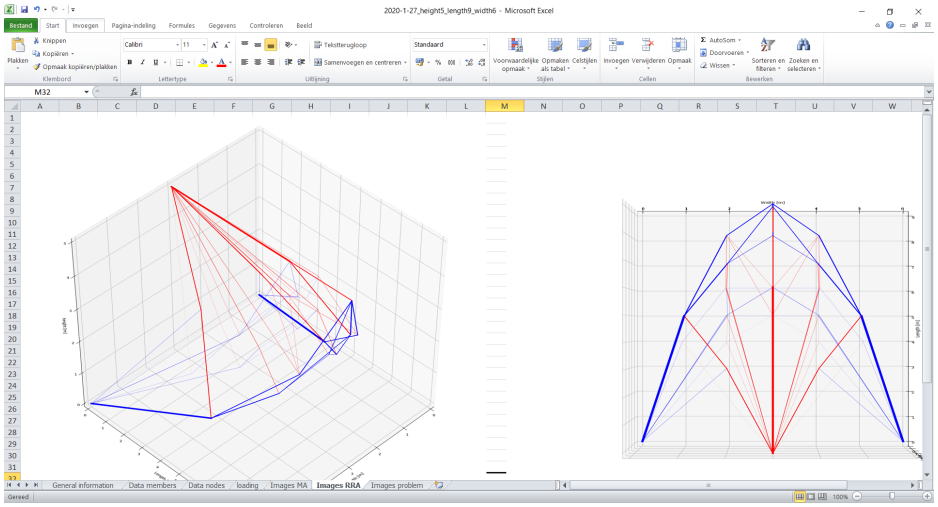


Figure E.14: Excel output file worksheet 6

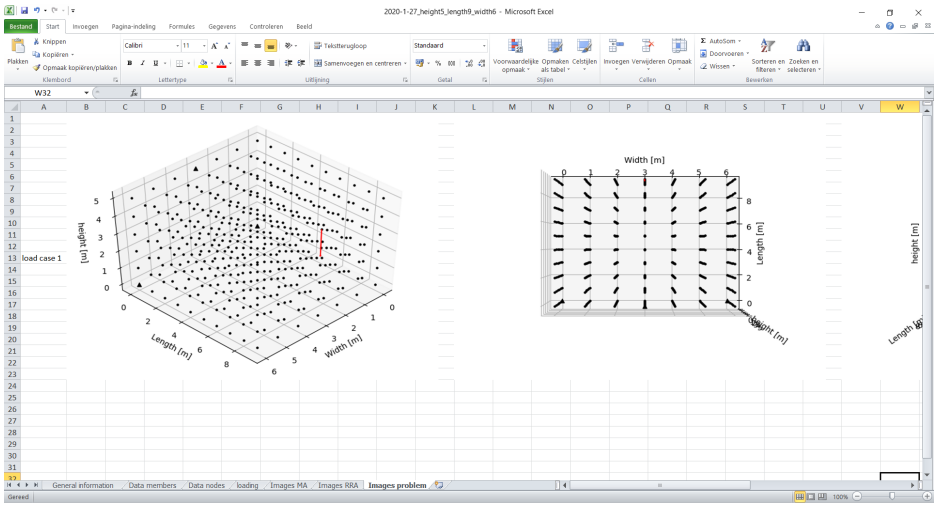


Figure E.15: Excel output file worksheet 7

E.6. Post processing

The output file, which is generated using the codes shown in Section E.5 can be imported in Python, such that interactive figures can be generated again. The code which imports the Excel file in Python is shown in Figure E.16.

```
def loadexcel(year, month, day, height, length, width, version):
    path = "C:/Users/Harry/.spyder/case study horizontal vertical/"+str(year)+"-"+str(month)+"-"+str(day)+\
    '_height'+str(height)+'_length'+str(length)+'_width'+str(width)+"v"+str(version)

    fn = "/"+str(year)+"-"+str(month)+"-"+str(day+2)+'_height'+str(height)+'_length'+str(length)+'_width'+\
        str(width)+".xlsx"
    os.chdir(path)
    Gi, Dm = pd.read_excel(path+fn, sheet_name = "General information"),\
        pd.read_excel(path+fn, sheet_name = "Data members")
    Dn, Ld = pd.read_excel(path+fn, sheet_name = "Data nodes"), pd.read_excel(path+fn, sheet_name = "loading")

    Nodes, Density = int(Gi.iloc[3,1]), Gi.iloc[12,1]
    a, a_new = Dm.iloc[:,0], Dm.iloc[:,1]
    Nd, Cn = np.zeros(shape = [Nodes, 3]), np.zeros(shape = [len(a), 3])
    q, f = np.zeros(shape = [np.shape(Dm)[1]-5, len(Dm)]), np.zeros(shape = [np.shape(Dm)[1]-5, Nodes*3])
    dof = np.zeros(Nodes*3)
    for i in range(3):
        Nd[:,i] = Dn.iloc[:,i+1]
        Cn[:,i] = Dm.iloc[:,i+2]
    for k in range (5, np.shape(Dm)[1]):
        for i in range(np.shape(q)[1]):
            q[k-5,i] = Dm.iloc[i,k]
        for i in range(Nodes):
            for j in range(4,7):
                f[k-5,i*3+j-4] = Ld.iloc[i+1,:][j+(k-5)*3]
    for i in range(Nodes):
        for j in range(1,4):
            dof[i*3+j-1] = Ld.iloc[i+1,:][j]
    return Nd, Cn, a, a_new, q, Density, dof, f
```

Figure E.16: Code to import the output Excel file in Python

The code for plotting the interactive figure of the initial problem is shown in Figure E.17 and the code for plotting the solution in an interactive figure is shown in Figure E.18.

```
## Visualize start of the member adding problem ##
def plotstartproblem(Nd, q, Cn, f, a, height, length, width, dof):
    for k in range (len(q)):
        fig = plt.figure()
        axs  = fig.add_subplot(111, projection = '3d')
        for i in range(len(Nd)): #all points
            if (dof[3*i] == 0.0 and dof[3*i+1] == 0.0 and dof[3*i+2] == 0.0): #points with constraint
                axs.scatter3D(Nd[i,0], Nd[i,1], Nd[i,2], c="k", marker="^", s = 25)
            else:
                axs.scatter3D(Nd[i,0],Nd[i,1],Nd[i,2], c="k", marker='o', s = 5)
            if f[k][3*i] != 0: #force x-direction (width)
                axs.quiver(Nd[i,0], Nd[i,1], Nd[i,2], 1, 0, 0, length=f[k][3*i]/np.max(np.abs(f[k]))*1.5,\
                           arrow_length_ratio = 0.01, colors="r", normalize=False)
            if f[k][3*i+1] != 0: #force y-direction (length)
                axs.quiver(Nd[i,0], Nd[i,1], Nd[i,2], 0, 1, 0, length=f[k][3*i+1]/np.max(np.abs(f[k]))*1.5\
                           , arrow_length_ratio = 0.05, colors="r", normalize=False)
            if f[k][3*i+2] != 0: #force z-direction (height)
                axs.quiver(Nd[i,0], Nd[i,1], Nd[i,2], 0, 0, 1, length=f[k][3*i+2]/np.max(np.abs(f[k]))*1.5,\
                           arrow_length_ratio = 0.05, colors="r", normalize=False)
        axs.set_xlabel('Width [m]')
        axs.set_ylabel('Length [m]')
        axs.set_zlabel('height [m]')
        plt.show()
        plt.savefig("startproblem_load_case_"+ str(k+1)+"_generated_afterwards.png")
```

Figure E.17: Python code for plotting the interactive figure of the initial problem

```python
## Visualize truss ##
def plotTruss(Nd, Cn, a, q, threshold, str, height, length, width):
    tk = 5/max(a)
    fig1 = plt.figure()
    fig1 = plt.figure(figsize = (15,20))
    ax  = fig1.add_subplot(111, projection = '3d')
    for i in [i for i in range(len(a)) if a[i] >= threshold]:
        if all([q[lc][i]>=0 for lc in range(len(q))]): c = 'r'
        elif all([q[lc][i]<=0 for lc in range(len(q))]): c = 'b'
        else: c = 'tab:gray'
        pos = Nd[Cn[i, [0, 1]].astype(int), :]
        ax.plot([pos[0,0],pos[1,0]],[pos[0,1],pos[1,1]],[pos[0,2],pos[1,2]],linewidth = a[i] * tk, \
                color = c)
    ax.set_xlabel('Width [m]')
    ax.set_ylabel('Length [m]')
    ax.set_zlabel('height [m]')
    ax.set_zlim(0,height)
    ax.set_ylim(0,length)
    ax.set_xlim(0,width)
    plt.show()
    plt.savefig(str + "_generated_afterwards.png")
```

Figure E.18: Python code for plotting the interactive figure of the solution

Another option can be to post-process the found solution in a FEM-program. A script to analyse the found structure in Abaqus can be found in Figure E.19 and E.20. Keep in mind, if you're using the student version of Abaqus, that it only works for less than 1000 nodes.

```python
def LoadAbaqus(Nd, Cn, a, a_new, q, Density, dof, f, limit):
## is a comment, * is important for reading the input file in Abaqus
    for k in range(len(f)):
        fi = open("Abaqus_input_file_load_case"+str(k+1)+".inp", "w+")
        fi.write("*Heading\n")
        fi.write("**\n")
        fi.write("** PARTS\n")
        fi.write("*Part, name=Part-1\n")
    ### Nodes and elements ###
        fi.write("*Node\n")
        for i in range(len(Nd)):
            fi.write("      "+str(i+1)+",   "+str(Nd[i,0])+",   "+str(Nd[i,1])+",   "+str(Nd[i,2])+"\n")
        fi.write("*Element, type=T3D2\n")
        for i in range(len(Cn)):
            if a[i] >= limit:
                fi.write(str(i+1)+", "+str(int(Cn[i,0])+1)+", "+str(int(Cn[i,1])+1)+"\n")
    ### Section assignment ###
        fi.write("**\n")
        for i in range(len(Cn)):
            if a[i] >= limit:
                #fi.write("*Nset, nset=_PickedSet"+str(i+1)+", internal \n")
                #fi.write(str(int(Cn[i,0])+1)+", "+str(int(Cn[i,1])+1)+"\n")
                fi.write("*Elset, elset=_PickedSet"+str(i+1)+", internal \n")
                fi.write(str(i+1)+"\n")
        for i in range(len(Cn)):
            if a[i] >= limit:
                fi.write("** Section: Section-"+str(i+1)+"\n")
    #### uitbreiden naar meerdere load-cases moet nog gebeuren ###
                if q[k][i] <= 0:
                    fi.write("*Solid Section, elset=_PickedSet"+str(i+1)+", material=Concrete_uncracked\n")
                else:
                    fi.write("*Solid Section, elset=_PickedSet"+str(i+1)+", material=Concrete_cracked\n")
                fi.write(str(a[i])+",\n")
        fi.write("*End Part\n")
        fi.write("**\n")
    ### Assmebly ###
        fi.write("** ASSEMBLY\n")
        fi.write("**\n")
        fi.write("*Assembly, name=Assembly\n")
        fi.write("**\n")
        fi.write("*Instance, name=Part-1-1, part=Part-1\n")
        fi.write("*End Instance\n")
        fi.write("**\n")
        for i in range(len(Nd)):
            fi.write("*Nset, nset =_PickedSet"+str(i+1)+", internal, instance=Part-1-1\n")
            fi.write(" "+str(i+1)+",\n")
        fi.write("*End Assembly\n")
```

Figure E.19: Python code for running an analysis in Abaqus part 1

```
### Materials ###
    fi.write("**\n")
    fi.write("** MATERIALS\n")
    fi.write("**\n")
    fi.write("*Material, name=Concrete_cracked\n")
    fi.write("*Density\n")
    fi.write(str(Density)+",\n") #2350.,
    fi.write("*Elastic\n")
    fi.write("1.8e+10, 0.4\n")
    fi.write("*Material, name=Concrete_uncracked\n")
    fi.write("*Density\n")
    fi.write(str(Density)+",\n") #2350.,
    fi.write("*Elastic\n")
    fi.write("3.6e+10, 0.4\n")

### Step including Boundary conditions and loads ###
    fi.write("**STEP: Step-1\n")
    fi.write("*Step, name=Step-1, nlgeom=NO\n")
    fi.write("*Static\n")
    fi.write("1., 1., 1e-05, 1.\n")
    fi.write("**\n")
    fi.write("** BOUNDARY CONDITIONS\n")
    fi.write("**\n")
    for i in range(1,len(dof)+1):
        if dof[i-1] == 0:
            fi.write("*Boundary\n")
            if (i/3. - math.floor(i/3))*3. == 0.0:
                direction = 3
            elif round((i/3.-math.floor(i/3.))*3.) == 2.0:
                direction = 2
            else:
                direction = 1
            fi.write("_PickedSet"+str(int(math.ceil(i/3.)))+", "+str(direction)+", "+str(direction)+"\n")
    fi.write("**\n")
    fi.write("** LOADS\n")
    fi.write("**\n")
    fi.write("** Name: Point Load   Type: Concentrated force\n")
    fi.write("*Cload\n")
    for i in range(1,len(Nd)*3+1):
        if f[k][i-1] != 0:
            if i/3. - math.floor(i/3.) == 0.0:
                direction = 3
            elif round((i/3.-math.floor(i/3.))*3) == 2.0:
                direction = 2
            else:
                direction = 1
            fi.write("_PickedSet"+str(int(math.ceil(i/3.)))+", "+str(direction)+", "+str(f[k][i-1])+"\n")
```

```
### Output Request (standard) ###
    fi.write("**\n")
    fi.write("** OUTPUT REQUESTS\n")
    fi.write("**\n")
    fi.write("*Restart, write, frequency=0\n")
    fi.write("**\n")
    fi.write("** FIELD OUTPUT: F-Output-1\n")
    fi.write("**\n")
    fi.write("*Output, field, variable=PRESELECT\n")
    fi.write("**\n")
    fi.write("** HISTORY OUTPUT: H-Output-1\n")
    fi.write("**\n")
    fi.write("*Output, history, variable=PRESELECT\n")
    fi.write("*End Step\n")
    fi.close()
```

Figure E.20: Python code for running an analysis in Abaqus part 2

This code writes an XXXX.inp file, which can be imported in Abaqus, via File →Import→ Model…. Search the XXXX.inp file and import this, the problem will be loaded to Abaqus. Now the model can be adapted if needed, note that the mesh can only be adapted by the mesh edit tools. If a job is created and submitted the structure is analysed and the results are shown in the visualisation module of Abaqus. The code generates for each load case a different input file, therefore each load case can be analysed separately in Abaqus.

E.7. Rigid diaphragm action of floors

To show that the rigid diaphragm action of the floors works properly, a test case is optimized. The test case has the design space shown in Figure 5.13a and has four supports (each corner). The parameters described in Paragraph 5.4.1 are used, the load is a point load at the top of the building, on the half of the length, as shown in Figure E.21 (the point load is the yellow arrow).

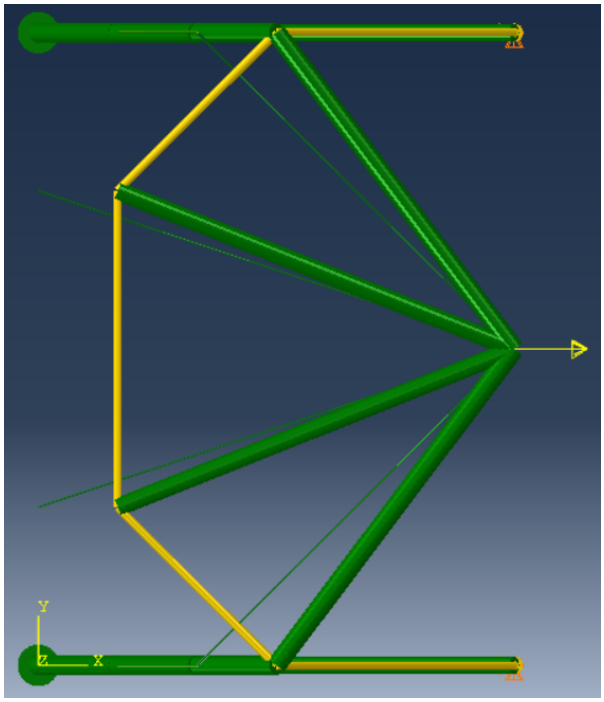


Figure E.21: Top view of solution with point load and supports visible

In the top view of the solution the elements are positioned such that they redirect the point load to the facade which is parallel to the direction of the point load. The point load works in the x-direction, thus the facade which is parallel to the x-axis redirects the point load to the supports (see Figure E.22). The optimal solution to transfer the load in the facade is arches, which are similar to the arches in the Michell truss (Figure 5.4a). The facade perpendicular to the load direction (parallel to the y-axis) is shown in Figure E.23, which clearly shows that this facade does nothing to transfer the loads to the supports. It can be concluded that the rigid diaphragm action of the floor works.

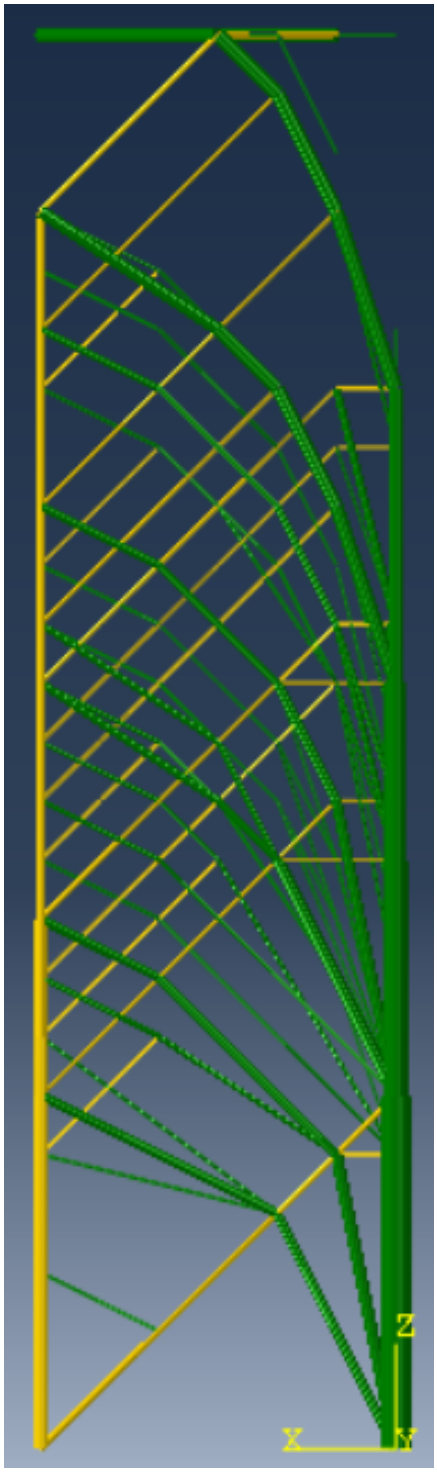


Figure E.22: Solution of the facade parallel to x-axis

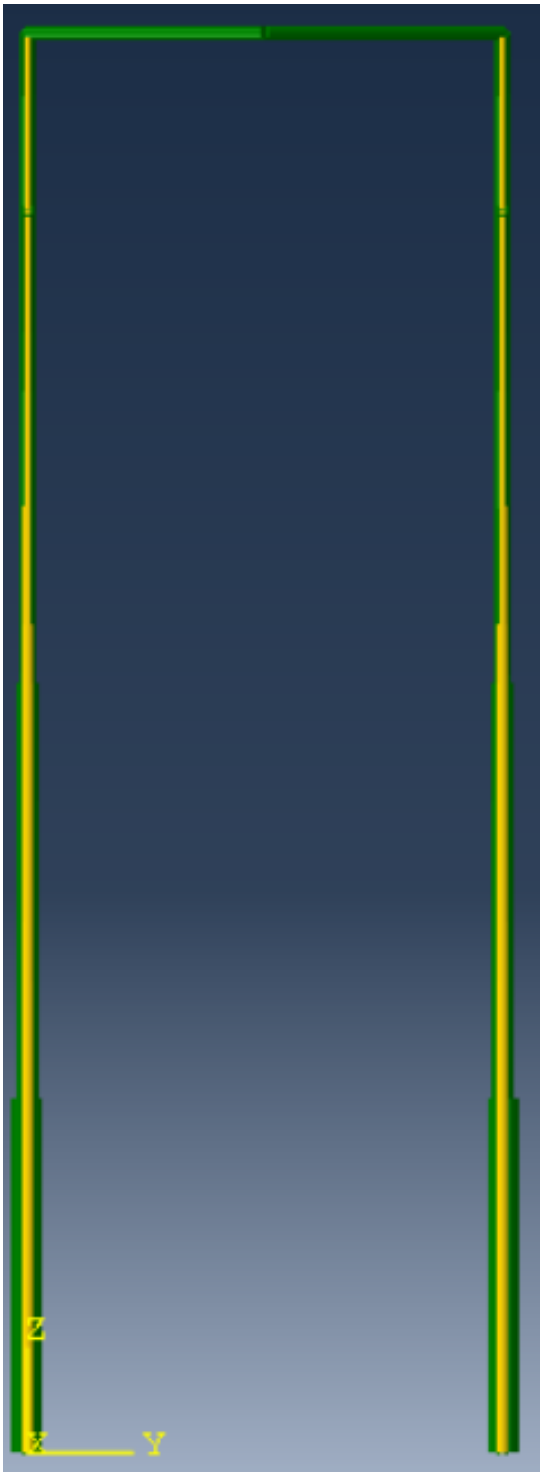


Figure E.23: Solution of the facade parallel to y-axis

F

Loads

In this appendix the loads for the different floors are calculated and shown in Table F.1 till F.5. These are the loads which are used to design the case study and will be used to construct a FEM-model.

F.1. Vertical loads on case study

Table F.1: Vertical loads on the 6^{th} floor

	Surface $[m^2]$	Perm. $[kN/m^2]$	Perm. total$[kN]$	Var. $[kN/m^2]$	Var. total$[kN]$
Floor	661.50	8.50	5622.75	1.50	992.25
Balcony	68.20	6.50	443.30	1.25	85.25
Internal walls	183.76	7.50	1378.18	0.00	0.00
Side facade	204.60	11.20	2291.52	0.00	0.00
End facade	105.00	5.30	556.50	0.00	0.00
Front	116.70	1.00	116.70	0.00	0.00
Parapet	33.84	5.17	175.05	0.00	0.00

Table F.2: Vertical loads on the 7^{th} up and till 21^{th} floor

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	657.60	8.50	5589.60	1.50	986.40
Balcony	76.90	6.50	499.85	1.25	96.13
Internal walls	183.76	7.50	1378.18	0.00	0.00
Side facade	204.60	11.20	2291.52	0.00	0.00
End facade	105.00	5.30	556.50	0.00	0.00
Front	116.70	1.00	116.70	0.00	0.00
Parapet	33.84	5.17	175.05	0.00	0.00

Table F.3: Vertical loads on the 22^{th} floor

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	657.60	8.50	5589.60	1.50	986.40
Balcony	68.20	6.50	443.30	1.25	85.25
Internal walls	183.76	7.50	1378.18	0.00	0.00
Side facade	204.60	11.20	2291.52	0.00	0.00
End facade	105.00	5.30	556.50	0.00	0.00
Front	116.70	1.00	116.70	0.00	0.00
Parapet	33.84	5.17	175.05	0.00	0.00

Table F.4: Vertical loads on the 23^{th} floor

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	611.70	8.50	5199.45	1.50	917.55
Balcony	76.90	6.50	499.85	1.25	96.13
Internal walls	183.76	7.50	1378.18	0.00	0.00
Side facade	204.60	11.20	2291.52	0.00	0.00
End facade	105.00	5.30	556.50	0.00	0.00
Front	116.70	1.00	116.70	0.00	0.00
Parapet	33.84	5.17	175.05	0.00	0.00

Table F.5: Vertical loads on the roof

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	476.70	8.50	4051.95	0.00	0.00

F.2. Vertical loads during optimization

During optimization the loads which act on the load-bearing structure at the perimeter of the building are of interest. Therefore the vertical loads during optimization are different than the vertical loads described in Section F.1. The following points are changed:

- The load-bearing structure incorporated in the side facade will be redesigned, therefore not taken into account as load, but the self-weight of the side facade will be taken into account in the optimization. Therefore the side facade and end facade will impose the same load, 5.30 kN/m^2;
- The load due to the self-weight of the internal walls is carried by themselves;
- The internal walls and the core will carry half of the vertical forces imposed by the floors (permanent and variable loads);
- It is assumed that the total load (permanently and variable) is divided over the perimeter of the building during optimization.

With this information the tables above are adjusted and shown in Table F.6 till F.10. There are small difference in the loads between the 6^{th} up and till the 23^{th} floor, contrarily the roof has less than half of the vertical load. The total loads per floor (permanent and variable) are noted in Table 5.1.

Table F.6: Vertical loads on the 6^{th} floor during optimization

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	330.75	8.50	2811.38	1.50	496.13
Balcony	68.20	6.50	443.30	1.25	85.25
Facade	309.60	5.30	1640.88	0.00	0.00
Front	116.70	1.00	116.70	0.00	0.00
Parapet	33.84	5.17	175.05	0.00	0.00
Total[kN]			5187.31		581.38

Table F.7: Vertical loads on the 7^{th} up and till 21^{th} floor during optimization

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	328.80	8.50	2794.80	1.50	493.20
Balcony	76.90	6.50	499.85	1.25	96.13
Facade	309.60	5.30	1640.88	0.00	0.00
Front	116.70	1.00	116.70	0.00	0.00
Parapet	33.84	5.17	175.05	0.00	0.00
Total[kN]			5227.28		589.33

Table F.8: Vertical loads on the 22^{th} floor during optimization

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	328.80	8.50	2794.80	1.50	493.20
Balcony	68.20	6.50	443.30	1.25	85.25
Facade	309.60	5.30	1640.88	0.00	0.00
Front	116.70	1.00	116.70	0.00	0.00
Parapet	33.84	5.17	175.05	0.00	0.00
Total[kN]			5170.73		578.45

Table F.9: Vertical loads on the 23^{th} floor during optimization

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	305.85	8.50	2599.73	1.50	458.78
Balcony	76.90	6.50	499.85	1.25	96.13
Facade	309.60	5.30	1640.88	0.00	0.00
Front	116.70	1.00	116.70	0.00	0.00
Parapet	33.84	5.17	175.05	0.00	0.00
Total [kN]			5032.21		554.90

Table F.10: Vertical loads on the roof during optimization

	Surface [m^2]	Perm. [kN/m^2]	Perm. total [kN]	Var. [kN/m^2]	Var. total [kN]
Floor	238.35	8.50	2025.98	0.00	0.00

F.3. Load combinations

This section includes the equations and partial factors from the Eurocode.

$$\sum_{j\geq 1} \gamma_{G,j} G_{k,j} \text{ "+" } \gamma_P P \text{ "+" } \gamma_{Q,1}\psi_{0,1} Q_{k,1} \text{ "+" } \sum_{i>1} \gamma_{Q,i}\psi_{0,i} Q_{k,i} \qquad (6.10a)$$

$$\sum_{j\geq 1} \xi_j \gamma_{G,j} G_{k,j} \text{ "+" } \gamma_P P \text{ "+" } \gamma_{Q,1} Q_{k,1} \text{ "+" } \sum_{i>1} \gamma_{Q,i}\psi_{0,i} Q_{k,i} \qquad (6.10b)$$

Figure F.1: Equation 6.10(a) and 6.10(b) from NEN-EN 1990+A1+A1/C2

CC	Blijvende en tijdelijke ontwerpsituaties	Blijvende belastingen		Overheersende veranderlijke belasting	Veranderlijke belastingen gelijktijdig met de overheersende	
		Ongunstig	Gunstig		Belangrijkste (indien aanwezig)	Andere
1	(Vgl. 6.10a)	1,2 $G_{k,j,sup}$ [a]	0,9 $G_{k,j,inf}$		1,35 $\psi_{0,1} Q_{k,1}$	1,35 $\psi_{0,i} Q_{k,i}$ (i >1)
	(Vgl. 6.10b)	1,1 $G_{k,j,sup}$ [b]	0,9 $G_{k,j,inf}$	1,35 $Q_{k,1}$		1,35 $\psi_{0,i} Q_{k,i}$ (i >1)
3	(Vgl. 6.10a)	1,5 $G_{k,j,sup}$ [a]	0,9 $G_{k,j,inf}$		1,65 $\psi_{0,1} Q_{k,1}$	1,65 $\psi_{0,i} Q_{k,i}$ (i >1)
	(Vgl. 6.10b)	1,3 $G_{k,j,sup}$ [b]	0,9 $G_{k,j,inf}$	1,65 $Q_{k,1}$		1,65 $\psi_{0,i} Q_{k,i}$ (i >1)

[a] Bij vloeistofdrukken met een fysiek beperkte waarde mag zijn volstaan met 1,2 $G_{k,j,sup}$.

[b] Deze waarde is berekend met ξ = 0,89.

Figure F.2: Partial factors for consequence class 1 and 3 from NEN-EN 1990+A1+A1/C2/NB

Belasting	ψ_0	ψ_1	ψ_2
Voorgeschreven belastingen in gebouwen, categorie			
Categorie A: woon- en verblijfsruimtes	0,4	0,5	0,3
Categorie B: kantoorruimtes	0,5	0,5	0,3
Categorie C: bijeenkomstruimtes	0,6/0,4 [a]	0,7	0,6
Categorie D: winkelruimtes	0,4	0,7	0,6
Categorie E: opslagruimtes	1,0	0,9	0,8
Categorie F: verkeersruimte, voertuiggewicht ≤ 30 kN	0,7	0,7	0,6
Categorie G: verkeersruimte [b], 30 kN < voertuiggewicht ≤ 160 kN	0,7	0,5	0,3
Categorie H: daken	0	0	0
Sneeuwbelasting	0	0,2	0
Belasting door regenwater	0	0	0
Windbelasting	0	0,2	0
Temperatuur (geen brand)	0	0,5	0

[a] De waarde 0,6 geldt voor delen van het gebouw die in geval van een calamiteit zwaar kunnen worden belast door een mensenmenigte (vluchtroutes, trappen enz.); de waarde 0,4 geldt in overige gevallen.

[b] Met verkeersruimte wordt in dit geval een ruimte bedoeld waar voertuigen kunnen rijden, bijvoorbeeld parkeergarages.

Figure F.3: ψ-factors for buildings from NEN-EN 1990+A1+A1/C2/NB

G

Extra results

In this appendix some extra results are shown of the optimizations done in Chapter 6.

G.1. V/h-ratio

In Figure G.1 and G.2 the v/h-ratios $0, 0.1, 0.2, 0.4, 0.8, 1$ for respectively the facade parallel to the x-axis and perpendicular to the x-axis are shown. These results are an extension of the results shown in 6.1.1.

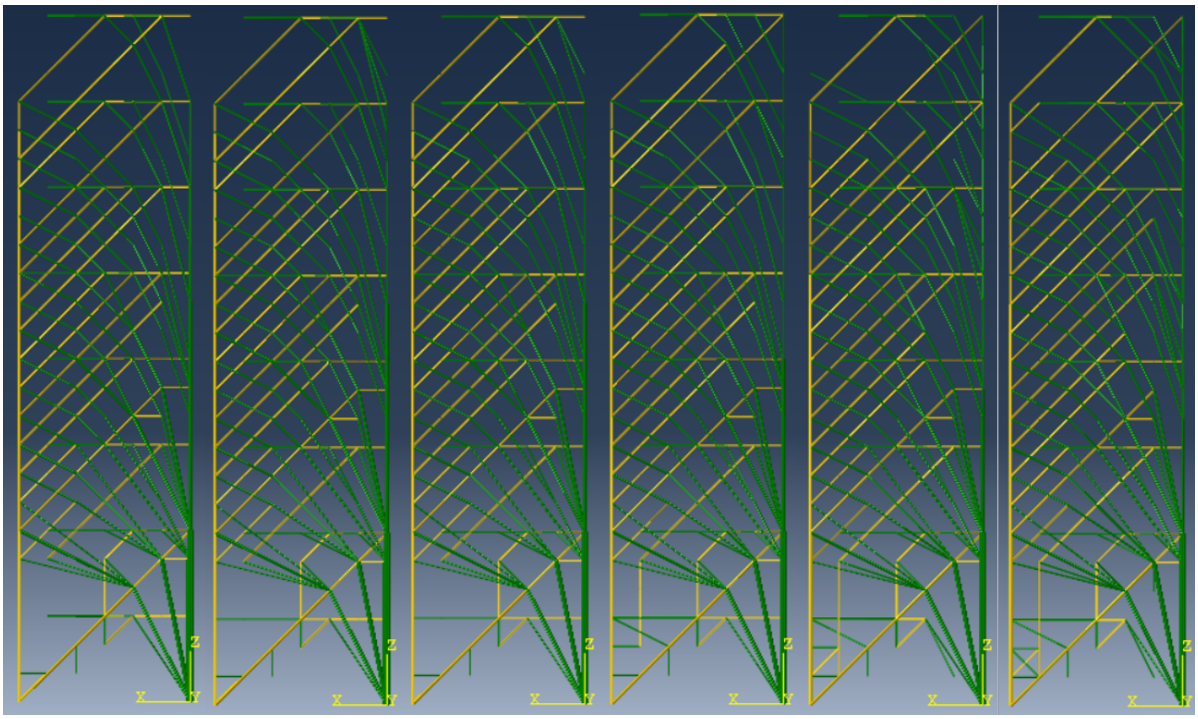


Figure G.1: Facade parallel to the x-axis and the direction of the wind load, from left to right: v/h-ratio = $0, 0.1, 0.2, 0.4, 0.8, 1$

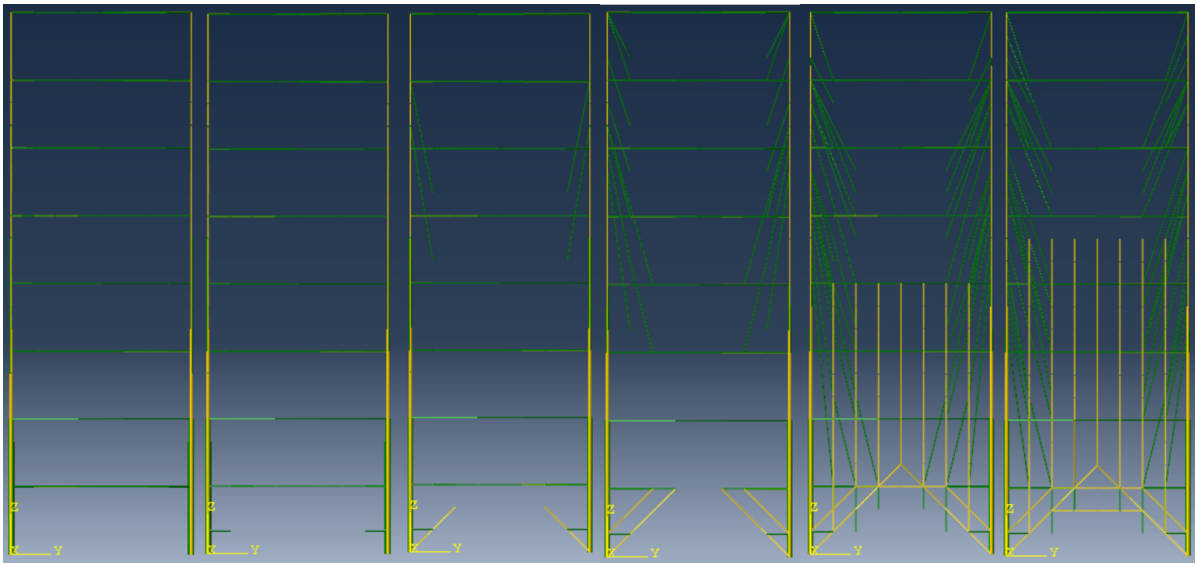


Figure G.2: Facade parallel to the y-axis and perpendicular to the direction of the wind load, from left to right: v/h-ratio = $0, 0.1, 0.2, 0.4, 0.8, 1$

G.2. Results load-bearing structure options

In the following figures the Abaqus static analysis of the two load-bearing options are shown.

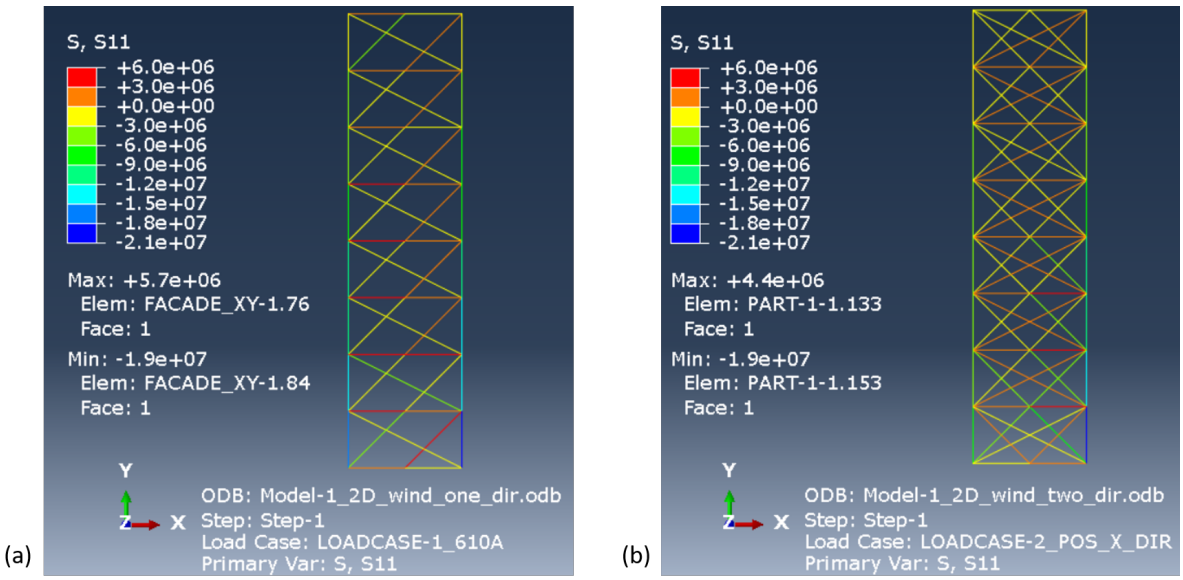
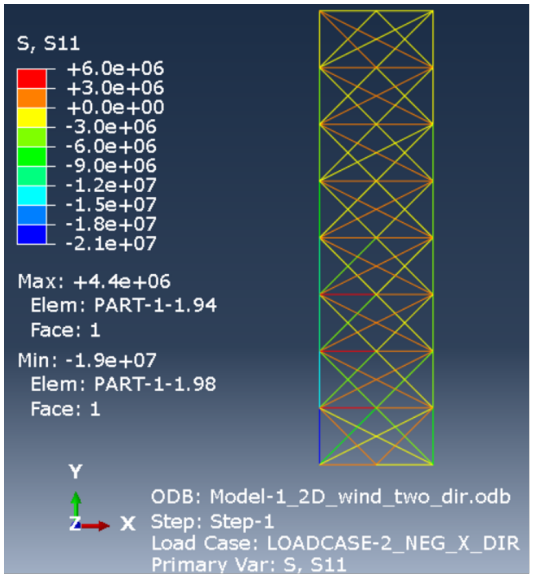


Figure G.3: (a) Internal stresses for load combination 1, option 1, (b) internal stresses for load combination 2 (wind in positive x-direction), option 1

(a)



(b)

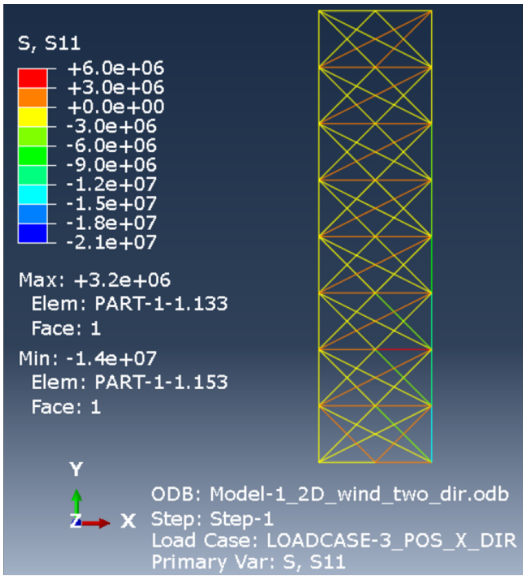


Figure G.4: (a) Internal stresses for load combination 2 (wind in negative x-direction), option 1, (b) internal stresses for load combination 3 (wind in positive x-direction), option 1

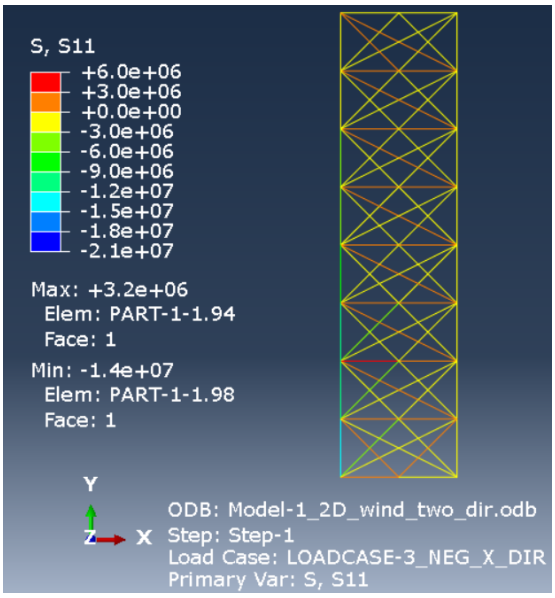
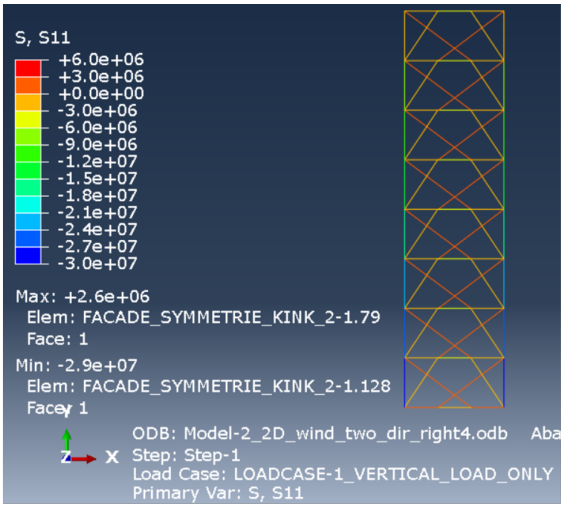


Figure G.5: Internal stresses for load combination 3 (wind in negative x-direction), option 1

(a)



(b)

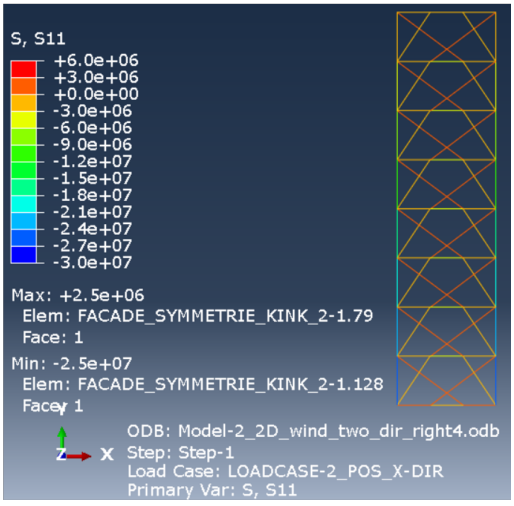
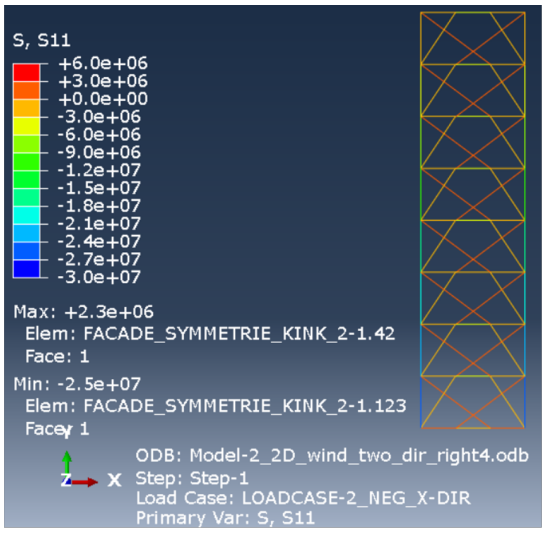


Figure G.6: (a) Internal stresses for load combination 1, option 2, (b) internal stresses for load combination 2 (wind in positive x-direction), option 2

(a)



(b)

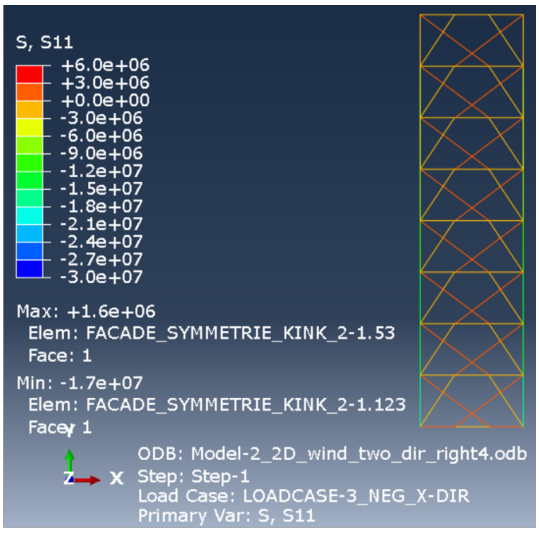


Figure G.7: (a) Internal stresses for load combination 2 (wind in negative x-direction), option 2, (b) internal stresses for load combination 3 (wind in positive x-direction), option 2

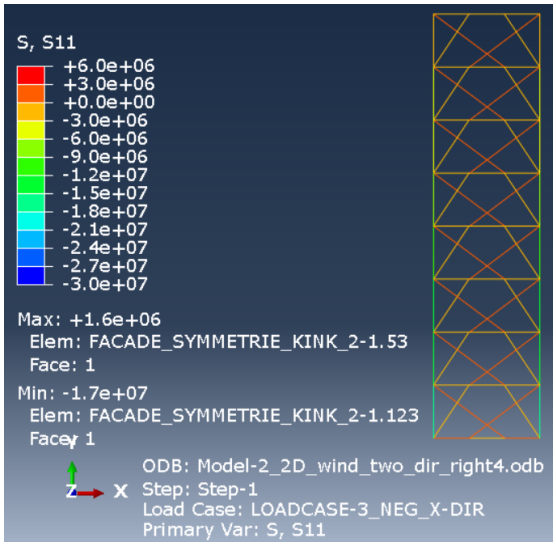


Figure G.8: Internal stresses for load combination 3 (wind in negative x-direction), option 2

H

Code

Before the optimization code can be executed certain packages need to be imported (under certain names). The code including the import of the packages are shown in Figure H.1.

```
import fractions
from math import ceil
import itertools
from scipy import sparse
import numpy as np
import cvxpy as cvx
import matplotlib.pyplot as plt
from shapely.geometry import Point, LineString, Polygon, shape
from mpl_toolkits.mplot3d import Axes3D
from timeit import default_timer as timer
import xlsxwriter
import datetime
import os
## Calculate equilibrium matrix B ##
def calcB(Nd, Cn, dof):
    m, n1, n2 = len(Cn), Cn[:,0].astype(int), Cn[:,1].astype(int)
    l, X, Y, Z = Cn[:,2], Nd[n2,0]-Nd[n1,0], Nd[n2,1]-Nd[n1,1], Nd[n2,2]-Nd[n1,2]
    d0, d1, d2, d3, d4, d5 = dof[n1*3], dof[n1*3+1], dof[n2*3], dof[n2*3+1], dof[n1*3+2], dof[n2*3+2]
    s = np.concatenate((-X/l * d0, -Y/l * d1, -Z/l * d4, X/l * d2, Y/l * d3, Z/l * d5))
    r = np.concatenate((n1*3, n1*3+1, n1*3+2, n2*3, n2*3+1, n2*3+2))
    c = np.concatenate((np.arange(m), np.arange(m), np.arange(m), np.arange(m), np.arange(m), np.arange(m)))
    return sparse.coo_matrix((s, (r, c)), shape = (len(Nd)*3, m))

## Solve linear programming problem ##
def solveLP(Nd, Cn, f, dof, st, sc, f_sw, width, height, jc, rho, sf, R):
    l = np.array([col[2] + jc for col in Cn])
    B = calcB(Nd, Cn, dof)
    a = cvx.Variable(len(Cn))
    if R is True:
        q, eqn, cons= [],  [], [a >= 1.2*np.maximum(np.power(l,2)/57.11,np.ones(len(l))*0.0625)]
    else:
        q, eqn, cons= [],  [], [a >= 1e-12]
    obj = cvx.Minimize(np.transpose(l) * a) #objective function
    for k, fk in enumerate(f):
        q.append(cvx.Variable(len(Cn)))
        if sf == True:
            eqn.append(B * q[k] == (fk + f_sw) * dof) #self-weight
        else:
            eqn.append(B * q[k] == fk * dof)
        cons.extend([eqn[k], q[k] >= -sc * a , q[k] <=  st * a])
    prob = cvx.Problem(obj, cons)
    vol = prob.solve(solver = cvx.MOSEK, mosek_params={"MSK_IPAR_INTPNT_SOLVE_FORM":1, \
        "MSK_IPAR_INTPNT_BASIS":0, "MSK_IPAR_INTPNT_MAX_ITERATIONS":1000000, \
        "MSK_DPAR_INTPNT_TOL_DSAFE": 10, "MSK_DPAR_INTPNT_TOL_PSAFE": 10}, verbose = True)
    q = [np.array(qi.value).flatten() for qi in q]
    a = np.array(a.value).flatten()
    u = [-np.array(eqnk.dual_value).flatten() for eqnk in eqn]
    return vol, a, q, u, l, B
```

```
## Main function ##
# Trussopt(m, m, m, N/m2, N/m2, kg/m3, True/False, True/False)
def trussopt(height, length, width, st, sc, jc, rho =2350, sf = True, R = False):
    makefolder(height,length,width)
    poly = Polygon([(width-21,0),(width-5,0),(width-5,length-25),(width,length-25),(width,length-11),\
           (width-5,length-11),(width-5,length),(width-21,length),(width-21,length-11),(0,length-11),\
           (0,length-25),(width-21,length-25)])
    xv, yv, zv = np.meshgrid(np.linspace(0,width, width+1), np.linspace(0,length,length+1), np.linspace(0, \
                             height, height+1))
    pts = [Point(xv.flat[i], yv.flat[i], zv.flat[i]) for i in range(xv.size)]
    Nd = []
    for pt in pts:
        if poly.boundary.intersects(pt): #only the points at the edge of the structure
            Nd.append([pt.x, pt.y, pt.z, True])
        elif fractions.gcd(pt.z,3) == 3 and pt.z != 0 and pt.within(poly):
            Nd.append([pt.x, pt.y, pt.z, False])
    Nd, dof, f, PML, Nodes = np.array(Nd), np.ones((len(Nd),3)), [], [], []
########### Load and support conditions ##########
    for i, nd in enumerate(Nd):             #support
        if nd[2] == 0: ## corners for case study (small version)
            if (nd[0] == 4 or nd[0] == 20) and (nd[1] == 0 or nd[1] == 12 or nd[1] == 26 or nd[1] == length):
                dof[i,:] = [0,0,0]
            if (nd[0] == 0 or nd[0] == width) and (nd[1] == 12 or nd[1] == 26):
                dof[i,:] = [0,0,0]
        if nd[3] == True:
            if (fractions.gcd(nd[2], 3) == 3 and (nd[0] == 0 or nd[0] == 4) and nd[2] != 0):
                if nd[0] == 4 and (nd[1] == 12 or nd[1] == 26):
                    f += [0, 0, -21.21e3]
                else:
                    f += [3.49e3, 0, -21.21e3]
            else:
                f += [0, 0, -21.21e3]
        else:
            f += [0, 0, 0]
    for i, nd in enumerate(Nd):             #Load case 3
        if nd[3] == True:
            if (fractions.gcd(nd[2], 3) == 3 and (nd[0] == 0 or nd[0] == 4) and nd[2] != 0):
                if nd[0] == 4 and (nd[1] == 12 or nd[1] == 26):
                    f += [0, 0, -10.96e3]
                else:
                    f += [12.88e3, 0, -10.96e3]
            else:
                f += [0, 0, -10.96e3]
        else:
            f += [0, 0, 0]
## Check dual violation ##
def stopViolation(Nd, PML, dof, st, sc, u, a, CN1, jc, R = False):
## here is Cn the members which are not used and CN1 are the members which are used ##
    if R is True: # set back the vlalues which are lower than de lower limit #
        for i in range(len(a)):
            if a[i] <= 0.99*max(CN1[i,2]**2/57.11, 0.0625):
                nnPML = np.intersect1d(np.where(CN1[i,0] == PML[:,0]),np.where(CN1[i,1] == PML[:,1]))
                PML[nnPML,3] = False
    lst = np.where(PML[:,3]==False)[0]
    Cn = PML[lst]
    l = Cn[:,2] + jc
    B = calcB(Nd, Cn, dof).tocsc()
    y = np.zeros(len(Cn))
    for uk in u:
        yk = np.multiply(B.transpose().dot(uk) / l, np.array([[st], [-sc]])) #Duality constrain
        y += np.amax(yk, axis=0)
    vioCn = np.where(y>1.0001)[0]
    vioSort = np.flipud(np.argsort(y[vioCn]))
    num = int(ceil(min(len(vioSort), 0.05*max( [len(Cn)*0.05, len(vioSort)])))) #The amount of new members
    #which are added to the structure: 5% of biggest exeedings
    for i in range(num):
        PML[lst[vioCn[vioSort[i]]]][3] = True
    return num == 0
```

```
## Create the 'ground structure' ##
    for i, j in itertools.combinations(range(len(Nd)), 2):
        dx, dy, dz = abs(Nd[i][0] - Nd[j][0]), abs(Nd[i][1] - Nd[j][1]), abs(Nd[i][2] - Nd[j][2])
        if fractions.gcd(fractions.gcd(int(dx), int(dy)),int(dz)) == 1 or jc != 0:
            seg = LineString([(Nd[i][0],Nd[i][1],Nd[i][2]),(Nd[j][0],Nd[j][1],Nd[j][2])])
            if dz == 0 and Nd[i][2] !=0 and fractions.gcd(Nd[i][2],3) == 3 and seg.within(poly):
                PML.append([i, j, np.sqrt(dx**2 + dy**2 + dz**2), False])
            if Nd[j][3] == Nd[i][3] == True: #and ((dy == 0) or (dx == 0): #creates facades
                if seg.within(poly.boundary):
                    PML.append([i, j, np.sqrt(dx**2 + dy**2 + dz**2), False])
    PML, dof = np.array(PML), np.array(dof).flatten()
    f = [f[i:i+len(Nd)*3] for i in range(0, len(f), len(Nd)*3)]
    print('Nodes: %d Members: %d' % (len(Nd), len(PML)))
    for pm in [p for p in PML if p[2] <= 1.42]:
        pm[3] = True
## Start the 'member adding' loop ##
    vol_t = np.zeros(151)
    for itr in range(1, 150):
        g = 9.81 #gravity
        rho = 2500 #density
        if itr == 1:
          f_sw = np.zeros(len(Nd)*3)
        Cn = PML[PML[:,3] == True]
        vol, a, q, u, l, B = solveLP(Nd, Cn, f, dof, st, sc, f_sw, width, height, jc, rho, sf=sf, R = R)
        vol_t[itr-1] = vol
        f_sw = np.zeros(len(Nd)*3)  #Adding self weight to the problem
        for i in range(len(Nd)):
          for j, cn in enumerate(Cn):
            if (cn[0] == i or cn[1] == i) and Nd[i][3] == True:
              f_sw[3*i+2] += -rho*g*cn[2]/2*abs(a[j])
        print("Itr: %d, vol: %f, mems: %d" % (itr, vol, len(Cn)))
        startplot = timer()
        CN1 = Cn
        if itr == 1:
          plotstartproblem(Nd, q, Cn, f, a, height, length, width, dof)
        ## plots truss structure after each iterations, time costly for big systems
        #plotTruss(Nd, Cn, a, q, max(a)*1e-5, "Itr:" + str(itr), height, length, width)
        if stopViolation(Nd, PML, dof, st, sc, u, a, CN1, jc, R): break
        if abs(vol_t[itr-2]-vol_t[itr-1]) <= max(vol_t[itr-2], vol_t[itr-1])*5e-3: break

    print("Finished, the volume is %f" % (vol))
    plotTruss(Nd, Cn, a, q, max(a)*1e-2, "MA", height, length, width)
    endplot = timer()
    plottimema = endplot-startplot
    return a, q, l, height, length, width, Nd, Cn, vol_t, st, sc, dof, plottimema, vol, f, sf, R, jc
## solving for unit load with q ##
def solveq(Nd, Cn, f, dof, st, sc):
    l = np.array([col[2] for col in Cn])
    B = calcB(Nd, Cn, dof)
    a = cvx.Variable(len(Cn))
    q, eqn, cons= [],  [], [a >= 0]
    obj = cvx.Minimize(np.transpose(l) * a)
    for k, fk in enumerate(f):
        q.append(cvx.Variable(len(Cn)))
        eqn.append(B * q[k] == fk * dof)
        cons.extend([eqn[k], q[k] >= -sc * a , q[k] <=  st * a])
    prob = cvx.Problem(obj, cons)
    vol = prob.solve("SCS")
    q = [np.array(qi.value).flatten() for qi in q]
    a = np.array(a.value).flatten()
    u = [-np.array(eqnk.dual_value).flatten() for eqnk in eqn]
    return vol, a, q, u

## solving for unit load with q ##
def calcq(height, length, width, st, sc, Nd, Cn, dof):
    f = []
    for i, nd in enumerate(Nd):
#### Here the location of interest and a unit load ####
        if (nd[0] == width and nd[1] == length/2 and nd[2] == height):
          f += [1,0,0]
        else:
          f += [0,0,0]
    f = [f[i:i+len(Nd)*3] for i in range(0, len(f), len(Nd)*3)]
    vol, a, q, u = solveq(Nd, Cn, f, dof, st, sc)
    #plotTruss(Nd, Cn, a, q, max(a)*1e-5, "distribution_q_unit_load",  height, length, width)
    return q
```

```
    z1 = (eta*(abs(maxTr)-maxU) + np.sum((maxTr/abs(maxTr))*q_unit[0]*q[0]*l/E/a_old))/(np.sum (q_unit[0]*\
         q_unit[0]*q[0]*q[0]*l/E/E/(a_old**3)))
    a_new = np.zeros(len(a_old))
    infac = np.zeros(len(a_old))
    for i in range(len(a_old)):
      infac[i] = 1+1/eta*(z1*maxTr*q_unit[0][i]*q[0][i]/E/a_old[i]**2/abs(maxTr)-1)
    a_newt = a_old*infac
    for i in range(len(a_old)):
      if a_old[i] >= a_newt[i]:
        a_new[i] = a_old[i]
      else:
        a_new[i] = a_newt[i]
### No reduction for cracked concrete
    maxT1, op1, maxT1r = np.zeros(len(q)), np.zeros(len(q)), np.zeros([len(q),len(q[0])])
    for k in range(len(q)):
      maxT1[k] = np.sum(q_unit[0] * q[k]*l/a_new/E)
### reduction for cracked concrete
      for i in range(len(q[0])):
        if q[k][i] >= 0:
          maxT1r[k][i] = 2*q_unit[0][i]*q[k][i]*l[i]/a_new[i]/E
        else:
          maxT1r[k][i] = q_unit[0][i]*q[k][i]*l[i]/a_new[i]/E
      op1[k] = np.sum(maxT1r[k])
    maxT1r, maxT1 = np.max(op1), np.max(maxT1)
    print("itr = "+ str(T+1))
    print("The Lagrange multiplier is                    %f" % z1)
    print("The relaxation parameters is                  %f" % round(eta,2))
    print("The displacement of the top floor is"+str((round(abs(maxTr),5)*1e3))+ " mm, before iteration")
    print("The displacement of the top floor is"+str((round(abs(maxT1r),5)*1e3))+ " mm, after iteration")
    print("The allowable displacement of the top floor is    "+str((round(maxU,5)*1e3))+ " mm")
    if round(abs(maxT1r),5) <= round(maxU,5):
      startplotRRA = timer()
      plotTruss(Nd, Cn, a_new, q, max(a_new)*10**-3, "RRA", height, length, width)
      endplotRRA = timer()
      timeplotRRA = endplotRRA-startplotRRA
      inRRA[3], inRRA[4] = maxT1, maxT1r
      return a_new, timeplotRRA, inRRA*1000
      break
    if T == TT-1:
        return a_new, 0, inRRA*1000
        break
    else: #turn on if you want to plot each iteration step
      #plotTruss(Nd, Cn, a_new, q, max(a)*1e-5, "RRA_itr:" + str(T+1), height, length, width)
      if abs(round(abs(maxT1r),5) - round(abs(maxTr),5)) <= 1e-5:
        print("increase the relaxation parameter")
        eta = eta + 1.0
      a_old = a_new
##### Recursive Resizing Algorithm ####
def RRA(a, q, l, height, length, width, Nd, Cn, st, sc, dof, eta):
  print("start RRA")
  E, maxU, inRRA = 36e9, height/750.0, np.zeros(5)
  q_unit = calcq(height, length, width, st, sc, Nd, Cn, dof)
  print("end_q_unit")
  TT = 2
  for T in range(TT):
    if T == 0:
      a_old = a
### No reduction for cracked concrete
    maxT, op, maxTr = np.zeros(len(q)), np.zeros(len(q)), np.zeros([len(q),len(q[0])])
    for k in range(len(q)):
      maxT[k] = np.sum(q_unit[0] * q[k]*l/a_old/E)
### reduction for cracked concrete
      for i in range(len(q[0])):
        if q[k][i] >= 0:
          maxTr[k][i] = 2*q_unit[0][i]*q[k][i]*l[i]/a_old[i]/E
        else:
          maxTr[k][i] = q_unit[0][i]*q[k][i]*l[i]/a_old[i]/E
      op[k] = np.sum(maxTr[k])
    maxTr = np.max(op)
    maxT = np.max(maxT)
    if T == 0:
        inRRA[0], inRRA[1], inRRA[2] = maxU, maxT, maxTr
    if round(abs(maxTr),5) <= round(maxU,5):
      startplotRRA = timer()
      plotTruss(Nd, Cn, a_old, q, max(a_old)*10**-3, "RRA", height, length, width)
      print(" ")
      print("No iterations are needed, displacement is      "+ str(round(abs(maxTr),5)*1e3)+ "mm")
      print("The allowable displacement of the top floor is "+ str((round(maxU,5)*1e3))+ "mm")
      a_new = a_old
      endplotRRA = timer()
      timeplotRRA = endplotRRA-startplotRRA
      return a_new, timeplotRRA, inRRA*1000
      break
```

```
startMA = timer()
### Starting the strength optimization ###
a, q, l, height, length, width, Nd, Cn, vol_t, st, \
sc, dof, timeplotma, vol, f, sf, R, jc = trussopt(54, 37, 25, 4.35e6, 24.75e6, 0, sf= False, R = False)
endMA = timer()

startRRA = timer()
## Starting the stiffness optimization ###
a_new, timeplotRRA, inRRA = RRA(a, q, Cn[:,2], height, length, width, Nd, Cn, st, sc, dof, eta = 5)
endRRA = timer()

savedata(Nd, Cn, a, q, height, length, width, vol_t, vol, f, dof, a_new, sf, R, inRRA, st, sc,\
                startMA, endMA, timeplotma, startRRA, endRRA, timeplotRRA)
```

Figure H.1: Optimization code in Python to optimize the case study