The effect of bolt clearance and tolerances on the shear resistance of bolted connections subjected to uni-axial loading

A parametric study

FERNANDO DE ABREU ALMEIDA
The effect of bolt clearance and tolerances on the shear resistance of bolted connections subjected to uni-axial loading

Author: Fernando de Abreu Almeida
Supervisor: Dr. Bert Norlin

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in the Civil and Architectural Engineering Architecture and the Built Environment

December 12, 2018
Declaration of Authorship

I, Fernando de Abreu Almeida, declare that this thesis titled, “The effect of bolt clearance and tolerances on the shear resistance of bolted connections subjected to uni-axial loading” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: 

"Declaration of Authorship"
Abstract

Structural Engineering and Bridge Design
Architecture and the Built Environment

Master of Science

The effect of bolt clearance and tolerances on the shear resistance of bolted connections subjected to uni-axial loading

by Fernando DE ABREU ALMEIDA

The aim of this thesis was to investigate the effect of clearance and tolerance in bolted joints when there is a mismatch between the bolt holes. A parametric study with seven different cases was analyzed in this project; four double bolt configuration and three triple bolt configuration, with variation of the size of the bolt hole misalignment, the diameter of the bolt and the thickness of the plates. All analyses were performed with the aid of the FEM commercial software Abaqus, all the models were modelled with 3D brick elements.

Despite bolted connections being subject of several investigations, no study about this matter for structural engineering purposes had been performed before.

The results indicate that for connections with a low number of bolts a misalignment of the bolt clearance can cause a serious reduction in the ultimate bearing capacity of a joint and it indicates that the Eurocode 1993 1-8 might be overestimating the ultimate bearing capacity for some cases.

Keywords: Bolt Clearance, Bolt, FEM, Eurocode 1993, Parametric study
Acknowledgements

I would like to thank my Supervisor for his guidance and disposition to answer my every question, to direct me to the right direction and for the proposal of the thesis topic.

I would like to thank my family for their support throughout my entire academic journey, for being there for me and cheering for me every step of the way.

I would like to thank the scientific community for sharing their knowledge and providing me with the information required to complete this project.

My colleagues for the debate of ideas during the development of the project and for the laughs shared.

Last but not least, I would like to thank my girlfriend for the support and patience.
Contents

Declaration of Authorship iii
Abstract v
Acknowledgements vii

1 Introduction 1
1.1 Aim and Scope ........................................... 1
1.2 Assumptions and Limitations .............................. 2

2 Theory 5
2.1 Bolted Connections ..................................... 5
2.2 Steel .................................................... 7
2.3 Finite Element Method .................................. 8
2.4 Bolt Shear Test ......................................... 9
2.5 Eurocode ................................................ 10

3 Method 15
3.1 Analysis Procedure ..................................... 15
3.2 FEM Model .............................................. 17
3.2.1 Type of Analysis .................................... 17
3.2.2 Material Properties ................................ 17
3.2.3 Geometry ........................................... 19
3.2.4 Boundary Conditions and Load ...................... 21
3.2.5 Element Type ....................................... 22
3.2.6 Mesh Size ........................................... 22
3.2.7 Interaction ........................................... 24
3.2.8 Reaching convergence ............................... 25
3.3 Validation of the Model .................................. 26
3.4 Parametric Study ....................................... 28

4 Results 31
4.1 General Results ......................................... 31
4.2 Bearing Resistance Curves ............................... 34

5 Discussion and Further Research 37
5.1 Discussion ............................................... 37
5.2 Further Research ....................................... 37

Bibliography 39
A Detailed Results 41
B Developed Codes 49
## List of Figures

1.1 Bolt Hole Misalignment ........................................... 2
1.2 Contact Angle ..................................................... 3
1.3 Shear Plane ........................................................ 4
2.1 Bolt Shear .......................................................... 5
2.2 Bolt Pull-Through .................................................. 5
2.3 Plate Bearing ...................................................... 6
2.4 Plate Net-Section ................................................... 6
2.5 Plate Shearing ...................................................... 6
2.6 Plate Cleavage ..................................................... 7
2.7 S355 x G550 ........................................................ 8
2.8 Universal Testing Machine ........................................ 9
2.9 Schematic Shear Test .............................................. 10
2.10 EC 1993 1-8 spacing requirements ............................. 11
3.1 Double Bolt Misalignment ......................................... 16
3.2 Triple Bolt Misalignment .......................................... 16
3.3 Plate S355 - Stress x Strain ..................................... 18
3.4 Bolt 8.8 - Stress x Strain ......................................... 18
3.5 Plate S355 - True x Eng ........................................... 19
3.6 Bolt Geometry ...................................................... 19
3.7 Plate Geometries ................................................... 20
3.8 Symmetry Plate .................................................... 20
3.9 Full model x Symmetric model ................................... 21
3.10 Boundary Conditions .............................................. 21
3.11 Mesh Size .......................................................... 23
3.12 Mesh Study ........................................................ 23
3.13 Friction Study ...................................................... 25
3.14 Type of Load Application ........................................ 25
3.15 Validated Experiments ........................................... 27
3.16 Validation of Single Bolt ......................................... 27
3.17 Parametric Flowchart ............................................. 29
4.1 Bolt Failure - FEM Analysis ....................................... 32
4.2 Force x Displacement for several Δd .............................. 33
4.3 Double Bolt - Diameter = 12 mm ................................. 34
4.4 Double Bolt - Thickness = 10 mm ............................... 34
4.5 Diameter = 12 mm - Thickness = 05 mm ....................... 35
4.6 Diameter = 12 mm - Thickness = 10 mm ....................... 35
4.7 Diameter = 16 mm - Thickness = 10 mm ....................... 35
4.8 Triple Bolt - Thickness = 10 mm ............................... 36
A.1 ULS Curve - Double Bolt - Diameter = 16 mm - Thickness = 10 mm ............................... 42
A.2 Force x Displacement - $\Delta d = 0.0$ mm ........................... 42
A.3 Force x Displacement - $\Delta d = 0.2$ mm ........................... 43
A.4 Force x Displacement - $\Delta d = 0.4$ mm ........................... 43
A.5 Force x Displacement - $\Delta d = 0.6$ mm ........................... 44
A.6 Force x Displacement - $\Delta d = 0.8$ mm ........................... 44
A.7 Force x Displacement - $\Delta d = 1.0$ mm ........................... 45
A.8 Force x Displacement - $\Delta d = 1.2$ mm ........................... 45
A.9 Force x Displacement - $\Delta d = 1.4$ mm ........................... 46
A.10 Force x Displacement - $\Delta d = 1.6$ mm ........................... 46
A.11 Force x Displacement - $\Delta d = 1.8$ mm ........................... 47
A.12 Force x Displacement - $\Delta d = 2.0$ mm ........................... 47
A.13 Force x Displacement - All $\Delta d$ ................................. 48
List of Tables

<table>
<thead>
<tr>
<th>Table No.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Resistance values for bolts according to the EC 1993 1-8.</td>
<td>11</td>
</tr>
<tr>
<td>2.2</td>
<td>Minimum spacing for bolted connections according to the EC 1993 1-8.</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>Nominal Clearances for Bolts and Pins According to the EN 1090-2.</td>
<td>12</td>
</tr>
<tr>
<td>3.1</td>
<td>Studied Cases.</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>Comparison between experimental data and FEM analyses.</td>
<td>26</td>
</tr>
<tr>
<td>4.1</td>
<td>Double Bolt Results.</td>
<td>31</td>
</tr>
<tr>
<td>4.2</td>
<td>Triple Bolt Results.</td>
<td>32</td>
</tr>
<tr>
<td>4.3</td>
<td>Eurocode x FEM.</td>
<td>33</td>
</tr>
<tr>
<td>A.1</td>
<td>ULS for Double Bolt - Diameter = 16mm - Thickness = 10mm.</td>
<td>41</td>
</tr>
</tbody>
</table>
List of Abbreviations

SLS  Service Limit State
ULS  Ultimate Limit State
mm  millimeter
s  second
T  Tonne
MPa  Mega Pascal
kN  kilo Newton
rad  radians
FEM  Finite Element Method
DOF  Degrees of Freedom
D  Dimension
MPC  Multi-Point Constraint
# List of Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d$</td>
<td>Diameter of the Bolt</td>
<td>mm</td>
</tr>
<tr>
<td>$d_0$</td>
<td>Diameter of the Bolt Hole</td>
<td>mm</td>
</tr>
<tr>
<td>$t$</td>
<td>Thickness of the Plate</td>
<td>mm</td>
</tr>
<tr>
<td>$f_{yb}$</td>
<td>Yield Strength of the Bolt</td>
<td>MPa</td>
</tr>
<tr>
<td>$f_{ub}$</td>
<td>Ultimate Strength of the Bolt</td>
<td>MPa</td>
</tr>
<tr>
<td>$f_u$</td>
<td>Ultimate Strength of the Plate</td>
<td>MPa</td>
</tr>
<tr>
<td>$k$</td>
<td>Eurocode Factor</td>
<td>unitless</td>
</tr>
<tr>
<td>$e$</td>
<td>Eurocode Minimum Spacing Between Bolt and Edge</td>
<td>mm</td>
</tr>
<tr>
<td>$p$</td>
<td>Eurocode Minimum Spacing Between Bolts</td>
<td>mm</td>
</tr>
<tr>
<td>$\Delta d$</td>
<td>Bolt Hole Misalignment</td>
<td>mm</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Stress</td>
<td>MPa</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>Strain</td>
<td>mm/mm</td>
</tr>
<tr>
<td>$\theta_c$</td>
<td>Angle of Contact</td>
<td>rad</td>
</tr>
<tr>
<td>$\gamma_{M2}$</td>
<td>Safety Factor</td>
<td>unitless</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Eurocode Factor</td>
<td>unitless</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Steel is widely used in several different kinds of structures, both in small and large scale, from small machinery to large infrastructure projects; it is a material that revolutionized architecture, construction and manufacturing around the globe. In order to assemble all those structures different parts of steel must be jointed together; nowadays the main techniques to do so are either by using bolts, screws or by welding the different pieces together.

Even though bolted connections are commonly used in constructions, there still are uncertainties regarding aspects that affects them, this study intends to explore in more depth the influence of bolt clearance and mismatch of bolt holes in connections. Despite its industrial manufacturing, there are imperfections when creating steel structures and even more in the assembly between those different structures, leaving room for misalignment between the holes of the steel plates, as a consequence, seldom all the bolts will receive the loads equally, there will always be bolts that will receive a larger load or a smaller load, this research intents to grasp a better understanding of the effect of this misalignment and its consequences in bolted connections.

Currently several investigations are being performed about bolted connections, however there is a gap in the study of the effect of bolt clearance in bolted connections, the most prominent research found was by McCarthy et al., 2002, which is focused on composite plates for the aerospace industry. This research will focus on the effect of bolt clearance in steel structures for the construction industry. Other authors such as Kelly and Hallström, 2004 and Woo et al., 2017 also investigated the effect of bolt-hole clearance, but for other plates materials (not steel) and for other industries.

1.1 Aim and Scope

This study will dwell into the usage of bolts for joining different steel parts together, to do so it will idealize bolted connections to focus the investigation on this matter and avoid investigating non-relevant topics.

The EN 1993 Eurocode 3 is the European standard for steel structures, it has two different standards to address bolted connections; the EC 1993 1-3 and EC 1993 1-8, whereas the main difference between them is the thickness of the steel plates being jointed together. According to the EC 1993-1 plates whose thickness are equal or thicker than 4mm shall be analyzed with the EC 1993 1-8 and plates whose thickness are thinner than 4mm shall be analyzed with the EC 1993 1-3.

Due to the advance of technology thinner plates are being connected with self-screwed connectors, thus requiring no clearance, this report will not investigate this
kind of bolted connections. The report focus on thicker plates that require bolt clearance for assembly. The standard for bolt clearance can be found in the EN 1090-2 (Section 2.5).

This study will analyze in more depth, with the aid of FEM Analysis, the effect of bolt clearance and the misalignment of the bolt holes (Δd) have on bolted steel connections and how they affect the ULS of the structure. To perform this study a 3D FEM model was developed with the aid of a FEM commercial software, a total of 81 simulations were performed to evaluate this phenomenon.

Three different models were investigated, the benchmark (single bolt configuration), a double bolt configuration and a triple bolt configuration. Three different sizes of bolts were investigated (M12, M16 and M20)\(^1\) for all three bolt configuration.

FEM is a very powerful tool to study steel structures and bolted connections, since if properly used it can give cheap and very accurate data, it is a good alternative to expensive and time consuming experiments (Hedeyat, Afzadi, and Iranpour, 2017). Abaqus was the FEM Software used in this study. Abaqus is a software developed by Dassault Systèmes Simulia Corp and it is a very popular commercial application with years of validation and several users around the world.

A parametric study was developed with the aid of Matlab and Python programming language in order to perform the analyses and analyze them in an optimal manner. In Section 3.4 the author briefly suggests an strategy to automatize processes for parametric studies.

This study intends to present results about the influence of bolt clearance, tolerance and bolt hole mismatch in steel bolted connections, it also aims to give the reader of the paper a good understanding of 3D FEM modelling for such scenarios, to illustrate ways of setting up parametric studies for further research and to provide ways of developing the research further.

1.2 Assumptions and Limitations

In this study the followings assumptions and limitations are present:

\(^1\)A Bolt with a diameter of 12, 16 and 20 millimeters, respectively.
1.2. Assumptions and Limitations

- Only slip joint type A (shear connections) with no preload on the bolt (CEN, 2005) was analyzed, therefore the effect of bolt tightening is not contemplated in this project;

- The bolt, the washer, the head and the nut were modelled as a single smooth element and no threaded pitch was contemplated in the analyses;

- This study did not investigate the effect of the contact angle (Figure 1.2) between the bolt and the hole diameter, it was assumed the same contact angle for all the analyses;

- Only single shear plane (Figure 1.3) models were investigated, no double shear planes were investigated in the study;

- To simplify the gathering of results and the FEM model, damage was not modelled, however the failure was imposed with the plastic behavior of the material, after reaching the $f_{ub}$ the bolt did not contributed any longer to the resistance of the joint;

- The experimental data used to validate the FEM model was gathered by other authors and they had a different type of failure (plate and not bolt) than the one investigated in this research.

![Figure 1.2: Contact Angle.](image)
Figure 1.3: Shear Plane.
Chapter 2

Theory

2.1 Bolted Connections

In order to assemble a thick bolted connection the steel plates must have a large bolt hole to ensure the installation of the bolt.

When a structure has corners or holes a stress concentration occurs and whenever this happens there is a larger risk of failure in that particular area, because of this a bolt connection is a hot spot for failure and it has to be analyzed with attention, both the bolt and the plate are subjected to a possible failure and they can occur in a variety of ways; thus a proper verification of its maximum resistance of the bolt and the plate are fundamental for the design of a steel structure.

The most common types of bolted connections are the following:

- Bolt Shear (Figure 2.1).

![Figure 2.1: Bolt Shear.](image)

- Bolt Pull-Through (Figure 2.2).

![Figure 2.2: Bolt Pull-Through.](image)
• Plate Bearing (Figure 2.3).

Figure 2.3: Plate Bearing.

• Plate Net-Section (Figure 2.4).

Figure 2.4: Plate Net-Section.

• Plate Shearing (Figure 2.5).

Figure 2.5: Plate Shearing.
2.2. Steel

- Plate Cleavage (Figure 2.6).

![Figure 2.6: Plate Cleavage.](image)

More than one failure can happen at once, a plate bearing and net-section failure can happen simultaneously in the same specimen for example.

For the purpose of this research all the bolted connections were designed to force a shear failure of the bolt.

### 2.2 Steel

Steel structures are the object of study of this research, three different types steels were used to verify and model the experiment, a cold-formed thin steel plate (G550), and hot-formed thick steel plate (S355JR) and a medium-low carbon steel bolt.

Despite both plates being made of steel (G550 and S355JR) there are differences in their fabrications processes that influenced their structural behavior, a hot-rolled plate is molded in high temperatures, which allows it to have thicker plates and more malleable shapes. While the cold-rolled plate is formed at room temperature, by mechanically rolling over a steel plate, this process increases the overall resistance of the material by strain-hardening, however it also makes it thinner and does not allow make very unusual shapes of the material.

As a result of those different processes the overall behavior of the plates is quite different from one to the other. It is clear to see the higher resistance and lower ductility of the cold-formed plate in comparison to the hot-rolled plate in Figure 2.7.

The fabrication of a S355 plate can occur in different ways and depending of how it is done it can have an effect on the maximum spacing of bolts allowed by the EC 1993 1-8 and influence its maximum resistance according to the EC 1993 1-1, the "J" letter stands for an impact test resistance of 27 Joules and the "R" letter stands for that the impact test was conducted at a room temperature of 20°C Celsius, the fabrication of a S355JR plate is done conforming the EN 10025-2. To verify cold-formed members one should look at the EC 1993 1-3.

According to ASTM F3125 Standard the bolts used in construction are made of low or medium carbon steel, this composition indicates strength, hardness and other qualities of the structural component. The bolts are graded in this paper as M12, M16 and M20, the "M" letter stands for the metric and the number stands for the diameter of the bolt in millimeters.
2.3 Finite Element Method

According to Cook et al., 2002 the Finite Element Method is a method for solving field problems numerically, a field problem be a structural analysis, a heat transfer, mass transport and more. The concept of the method is to subdivide a large problem in smaller and smaller parts, thus achieving an accurate solution. In structural engineering a given structure can be idealized into several types of elements:

- **Truss**: A 2D element that can only intake axial stresses and cannot take moment into account, this element has 4 DOF;
- **Beam**: A 2D elements that like the can take axial stresses and moments into account, this element has 6 DOF;
- **2D Continuum**: 2D plane stress or plane strain element, it can have a triangular or rectangular shape. These elements can be linear or quadratic and the number of DOF varies according to they shape and type;
- **Plate**: A Plate is a 3D solid body where one of the directions (thickness) is much smaller than the other two, it handles only bending deformation, as a consequence has a zero curvature in its mid surface;
- **Shell**: A Shell is a 3D solid body where one of the directions (thickness) is much smaller than the other two, its loading causes bending and stretching deformation, as a consequence it has a non-zero curvature in its mid surface;
- **Solid**: A 3D element without any restriction regarding any of its directions, it can have a tetrahedral or hexahedral shape.

Truss and Beams elements can be quite useful for modelling frames, 2D continuum element can be used no analyze a 2D solid for example; plate and shell elements can be used for slabs, membranes and more; solid elements are usually used to model solid objects when shell or plate elements are not ideal.

For this particular type of analysis the best option is to model the experiment with solid elements, in Section 3.2.5 the type of solid element chosen for this experiment is explained in more detail.
2.4 Bolt Shear Test

To test the bearing capacity of a joint usually a testing machine is used (Figure 2.8). For a bolt shear test it is usually a tensile or an universal testing machine.

![Universal Testing Machine](Figure 2.8: Universal Testing Machine. (Source: MTS)](image)

To perform this kind of experiment usually two steel plates are clamped into the machine and joined together by a number of bolts, the machine applies a certain displacement per unit of time until the failure of the joint (Section 2.1). The data can be collected directly by the machine, to have a better accuracy of the data one can have external extensometers on the area of interest (i.e. bolts or overlapping plate) or another kind of external monitoring of displacement, such as video.

The displacement generates an uni-axial tensile load in one of the plates that translates into a shear load on the bolts, one of the steel plates is usually fixed while the other one must withstand the imposed displacement (Figure 2.9).
Chapter 2. Theory

FIGURE 2.9: Schematic Shear Test.

The FEM model (Section 3.2) developed on this study simulated this kind of experiment.

2.5 Eurocode

The EN 3 Eurocode 1993 is the code for steel structures in Europe, as previously stated the Eurocode has two different sections to deal with bolted connections, which are directly related to the thickness of the steel plate, if the plate is thinner than 4 millimeters, then the EC 1993 1-3 will provide the directions to design such connections, if the plate is 4 millimeters thick or thicker then the EC 1993 1-8 will provide the directions.

There are several aspects to take into account when designing bolted connections, one must mind the type of bolt that is being used, the type of load the connection is being subjected to, if the bolt spacing is adequate, if the bolt hole clearance is following the standard and the spacing requirements.

The EC 1993 1-8 has a standard of bolt classes to use, the most used kind of bolt is the 8.8 grade. To calculate the bolt grade resistance yield resistance one can multiply both numbers of the class and multiply them by 10 and the result will be in Mega Pascal, for example $8 \times 8 \times 10 = 640 \text{MPa}$ is the yield resistance of a 8.8 bolt suggested by the Eurocode. To calculate the ultimate resistance a similar approach can be used, by multiplying the number before the dot by 100, for example $10 \times 1000 = 1000 \text{MPa}$ is the ultimate of a 10.9 bolt grade suggested by the Eurocode. Those values (Table 2.1) are conservative in relation to experimental data.
2.5. Eurocode

Table 2.1: Resistance values for bolts according to the EC 1993 1-8.

<table>
<thead>
<tr>
<th>Bolt Class</th>
<th>$f_{yb}$ (MPa)</th>
<th>$f_{ub}$ (MPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6</td>
<td>240</td>
<td>320</td>
</tr>
<tr>
<td>4.8</td>
<td>320</td>
<td>400</td>
</tr>
<tr>
<td>5.6</td>
<td>300</td>
<td>500</td>
</tr>
<tr>
<td>5.8</td>
<td>400</td>
<td>500</td>
</tr>
<tr>
<td>6.8</td>
<td>480</td>
<td>600</td>
</tr>
<tr>
<td>8.8</td>
<td>640</td>
<td>800</td>
</tr>
<tr>
<td>10.9</td>
<td>900</td>
<td>1000</td>
</tr>
</tbody>
</table>

The EC 1993 1-8 defines minimum and maximum spacing for bolts (Table 2.2), the maximum spacing can be quite tolerant depending on the type of the plate and the environment of the structure. In the other hand, the minimum spacing is far less tolerant with its requirements. All cases of the simulated bolted connections were within the spacing boundaries given by the Eurocode (Figure 2.10), both the minimum and maximum spacing. The analyses were assumed to be in a non-corrosive environment. For the maximum spacing the Eurocode also makes a distinction of how the steel plate was produced, the used plate is a S355JR steel, which is made according to the EN 10025-2 and not the EN 10025-5, thus providing the following maximum spacing.

Table 2.2: Minimum spacing for bolted connections according to the EC 1993 1-8.

<table>
<thead>
<tr>
<th>Distance and Spacing</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>End distance $e_1$</td>
<td>$1,2d_0$</td>
<td>-</td>
</tr>
<tr>
<td>Edge distance $e_2$</td>
<td>$1,2d_0$</td>
<td>-</td>
</tr>
<tr>
<td>Spacing $p_1$</td>
<td>$2,2d_0$</td>
<td>Smaller of 14$t$ or 200 mm</td>
</tr>
<tr>
<td>Spacing $p_2$</td>
<td>$2,4d_0$</td>
<td>Smaller of 14$t$ or 200 mm</td>
</tr>
</tbody>
</table>

Figure 2.10: EC 1993 1-8 spacing requirements. (Source: EC 1993 1-8)

The main investigation of this study is the effect of bolt clearance in bolted connections. According to the EN 1090-2 the minimum values for the execution of bolted connections can be seen in Table 2.3, it shows that most of the commercial bolt diameters operates with a 2 millimeter bolt clearance, with the exception of the
12 and 14 millimeters diameter that can be accepted as 1 millimeter and larger than 27 millimeter which operates with a 3 millimeters clearance; the code states a value for oversized hole and when this happens the EC 1993 1-8 suggests to use a reduction factor of 0.8 for the calculation of the bearing resistance for the joint.

### Table 2.3: Nominal Clearances for Bolts and Pins According to the EN 1090-2.

<table>
<thead>
<tr>
<th>Nominal bolt pin diameter (mm)</th>
<th>Normal round holes (mm)</th>
<th>Oversized round holes (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>22</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>24</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>27 to 36</td>
<td>3</td>
<td>8</td>
</tr>
</tbody>
</table>

The Eurocode has different criteria for different types of loading and whether or not the bolt is preloaded. In total the EC 1993 1-8 has 5 types of categories of bolted connections, 3 for shear connections and 2 for tensions connections.

- **Category A**: Bearing type for shear connections. No preloading is required and bolts classes from 4.6 to 10.9 can be used;
- **Category B**: Slip Resistant at SLS for shear connections. Preloaded 8.8 or 10.9 bolts are used.
- **Category C**: Slip-resistant at ULS for shear connections. Preloaded 8.8 or 10.9 bolts are used;
- **Category D**: Non-preloaded bolts for tensions connections. Bolts classes from 4.6 to 10.9 can be used;
- **Category E**: Preloaded bolts for tensions connections. Preloaded 8.8 or 10.9 bolts are used.

For the analyses performed in this study a Category A type of joint was assumed. For this scenario the EC 1993 1-8 recommends two types of verification, a shear resistance of the bolt (Equation 2.1) and a bearing resistance (Equation 2.2).

Shear resistance per shear plane:

\[
F_{v,Rd} = \frac{\alpha_v f_{ub} A}{\gamma M^2}
\]  

(2.1)

Where:

- \( A \) is the area of the cross-section of the bolt;
- \( f_{ub} \) is the bolt ultimate resistance;
• $\alpha_v$ is 0.6 for classes 4.6, 5.6 and 8.8;
• $\alpha_v$ is 0.5 for classes 4.8, 5.8, 6.8 and 10.9;
• $\gamma_{M2}$ is a safety factor of 1.25\(^1\).

The bearing resistance for bolts:

$$ F_{b,Rd} = \frac{k_1 \alpha_b f_u d t}{\gamma_{M2}} $$

(2.2)

Where:

• $d$ is the diameter of the bolt;
• $f_u$ is the plate ultimate resistance;
• $\alpha_b$ is the smallest of $f_u / f_u$;
• for edge bolts: $k_1$ is the smallest of $2.8 \frac{d}{d_0} - 1.7, 1.4 \frac{d}{d_0} - 1.7$ or 2.5;
• for inner bolts: $k_1$ is the smallest of $1.4 \frac{d}{d_0} - 1.7$ or 2.5.

\(^1\)This value may change in a country National Annex.
Chapter 3

Method

This chapter aims to provide the reader with all the important information to understand what was done in this project and give insight in how to replicate it.

3.1 Analysis Procedure

To investigate the effect of the bolt hole misalignment a series of shear failure tests (Section 2.4) were simulated to evaluate its effect, varying the thickness, the diameter and bolt configuration (Table 3.1).

For each case 11 analyses were made, with a varying $\Delta d$ of 0.2 mm per simulation, going from 0.0 mm to 2.0 mm, summing up a total of 77 analyses, plus other 4 analyses for the single cases of the same bolt diameter and plate thickness, giving a total of 81 analyses for this project.

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Thickness (mm)</th>
<th>Diameter (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>Triple</td>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>Double</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>Triple</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>Double</td>
<td>10</td>
<td>16</td>
</tr>
<tr>
<td>Triple</td>
<td>10</td>
<td>16</td>
</tr>
<tr>
<td>Double</td>
<td>10</td>
<td>20</td>
</tr>
</tbody>
</table>

For the double bolt case a hole misalignment was modelled for one of the bolts (Figure 3.1) and as a result this bolt had to bear all the load for a displacement equal to the $\Delta d$ which caused a precocious failure in comparison to the other bolt.

For the triple bolt case a hole misalignment was modelled in one of the edge bolts (Figure 3.2) and as a result this bolt had to bear all the load for a displacement equal to the $\Delta d$ which caused a precocious failure in comparison to the other bolts.
Chapter 3. Method

Figure 3.1: Double Bolt Misalignment.

Figure 3.2: Triple Bolt Misalignment.
3.2 FEM Model

In order to properly use Abaqus and all the benefits of FEM analysis a considerable amount of research and validations were performed.

To develop an adequate 3D model it requires inputting several different parameters, such as: material properties, boundary conditions, element type, mesh size, interaction between surfaces, type of analysis and Load. Four different experiments were used to validate the developed FEM model, the experiments were gathered from three different papers and from two different authors. The validation of the model is better explained in Section 3.3.

Whenever using a software like Abaqus or similar it is extremely important to input consistent units, for this project the International System of Units (SI) was used in newton (N), millimeter (mm), ton (T) and second (s).

3.2.1 Type of Analysis

The experiment was conducted with implicit analysis; explicit analysis can be used as well, but one must be careful about the consistency of the results, since explicit analyses do not enforce equilibrium in every interaction as the implicit analysis does, the explicit results could lead to very unrealistic deformations and behavior. Whereas implicit analysis is more consistent because of its equilibrium method, however it comes with a cost of a higher computational use (Salamet and Garlok, 2010) and convergence difficulties.

Within the implicit modulus several different kind of analyses can be performed, for this study a static general analysis was used, however a dynamic implicit quasi-static analysis renders equally precise results and it can also be better to reach convergence at the cost of a longer analysis.

Convergence is the main issue regarding this type of analysis, since it deals with several non-linear problems, such as irregular shapes, interaction between different parts and non-linear materials. In the Section 3.2.8 strategies to overcome this issue are going to be discussed.

3.2.2 Material Properties

There are several ways of modelling materials in Abaqus, it ranges from simply Elastic Models up to more complex models, such as Viscoelasto-plastic with damage, for this analysis an Elastoplastic material with a forced failure was used to better grasp the results and ensure that the maximum load allowed would be easily identified. The experimental data for the S355 plate was gathered from Ribeiro, Santiago, and Rigueiro, 2016 and the experimental data for the 8.8 bolt grade was gathered from Silva and Griza, 2013.

A typical Stress x Strain curve is usually represented in engineering stress and engineering strain. An engineering stress is the stress calculated with the initial cross-section area of the specimen without taking into account the deformation due to the load, whereas the true stress takes into account the deformation of the cross-section.

Abaqus (Simulia, 2013) works with true stress and true strain, to convert one must use the following formulas:

\[ \sigma_{\text{true}} = \sigma_{\text{eng}} (1 + \epsilon_{\text{eng}}) \]  
\[ \epsilon_{\text{true}} = \ln(1 + \epsilon_{\text{eng}}) \]
If one does not use true stress strain Abaqus will underestimate the bearing capacity of the material and how the material deforms (Figure 3.5).
3.2. FEM Model

3.2.3 Geometry

Three different model shapes (Figure 3.6) were used to perform this experiment, a single bolt configuration, a double bolt configuration and a triple bolt configuration, in all scenarios only a single shear plane is considered.

In order to reduce the number of variables most of the parameters were created in function of the diameter of the bolt (d) and the thickness of the plate (t), thus making sure the tests are standardized and following the EC 1993 1-8 spacing criteria.

Figure 3.5: Plate S355 - True x Eng.

Figure 3.6: Bolt Geometry.
To reduce computational time symmetry was used, a single analysis of a symmetric triple Bolt configuration took hours, the use of symmetry saved time for data processing and it allowed to run more analyses and to investigate more. As it can be seen on the Figure 3.9, the use of symmetry did not affect the results.

One important distinction is that if one uses symmetry, one must remember the force will be half of what it actually is. When using symmetry one must mind the position of the bolt to ensure the analysis is truly symmetric.
3.2. **FEM Model**

3.2.4 **Boundary Conditions and Load**

All the analyses were conducted with the same boundary conditions, regardless of the number of bolts, the bottom plate was considered fully pinned (all the displacements on the X, Y and Z-axis were imposed as zero and they were free to rotate), while the top plate had a roller support (the displacements on the Y and Z-axis were imposed as zero, they could rotate in the X, Y and Z-axis and the X-axis was free to move).

A displacement was prescribed on the X-axis of the top plate, for a single bolt plate a displacement of 5mm was imposed, for a double or triple bolt plate the imposed displacement was 6mm.
Chapter 3. Method

To emulate symmetry the following boundary conditions were imposed, the displacement on the Y-axis was imposed as 0 and the rotations on the Y and Z-axis were also imposed as zero.

There are several ways of enforcing boundary conditions and displacements in Abaqus and they all produce virtually the same results if properly used. The author suggests that for the boundary conditions enforce them in the desired surface and for parametric studies the use of MPC Beam Constraint for the application of the load, this will allow to program both the application of the displacement and later on the gathering of the Force x Displacement curve without having to worry about the number of nodes or the mesh size, other kinds of constraints (such as Equation) or applying a load directly to a surface will generate virtually the same results, however for programming parametric models it can be simpler to use an MPC (Beam) Constraint.

Abaqus works better with prescribed displacements than prescribed loads, this is due to the nature of the FEM solution \( F = k \times d \), by doing so the convergence rate will be higher and it will take less time to calculate the problem.

3.2.5 Element Type

All the experiments were conducted using 3D solid elements. For this shape and type of analysis the best elements of choice are Brick (hexahedral) elements. Wedge (tetrahedral) elements can be useful for very irregular geometries, however their accuracy is low in comparison with Brick elements, so the author suggests to use them carefully.

Abaqus has a variety of Brick Elements to choose from, ranging from C3D8R (8-noded brick element with reduced integration) to C3D20H (20-noded brick element with hybrid formulation), as explained by Salamet and Garlok, 2010, 20-noded elements are better for Elastic material problems whereas 8-noded elements are better suited for Plastic material problems.

C3D8R Elements are not well suited for contact, whenever there is contact between parts one should avoid using C3D8R elements, because it will not account for their contact and the results will not be accurate; C3D8 (Eight-node brick element) Elements are better suited for contact and generally more accurate than C3D8R Elements, but C3D8 has a problem with shear-locking and if the problem analyzed is a bending dominated problem these elements should be used with care, for the experiment conducted in this research an uni-axial load was applied, thus being a tensile dominated problem and shear-locking was not an issue in this scenario. C3D8I (Incompatible mode eight-node brick element) elements work well with contact zones and do not have the shear-locking problem, however they have extra nodes than C3D8 elements, which means more computational time and might lead to extra difficulties for reaching convergence when dealing with contact between different parts.

For this project all the elements used were C3D8 for the plates and the bolts, one can use C3D8R in parts without any contact, the results were virtually identical. During the research the author noted that using the same elements lead to a slightly higher convergence rate than mixing different elements, thus the choice of always using C3D8 elements.

3.2.6 Mesh Size

The size of the mesh is fundamental for the analysis accuracy and speed, for this study two different mesh sizes were used, one for the areas of interest (bolt and
3.2. FEM Model

plate near the bolt holes) and another for the rest of the plates. The size of the area of interest for this study varied accordingly to the diameter of the hole to make sure the data would be reliable.

A study about the accuracy of the analysis and the time of the analysis was conducted; five different mesh combinations were tested and the choice of mesh size was based on accuracy and the time of the analysis. The combination chosen was a seed size of 1 element per 1.0 mm in the areas of interest (bolt and the plates surrounding area of bolt) and 1 element per 4.0 mm in the areas that were not so interesting for the analysis, since it yielded very good results without demanding as much time as a finer mesh configuration would (Figure 3.12).

![Figure 3.11: Mesh Size.](image)

![Figure 3.12: Mesh Study.](image)
3.2.7 Interaction

The interaction between the different parts is fundamental for the proper modelling and it defines how the model will behave. To define the contact between surfaces one must chose a Master and a Slave surface, this choice can impact the results and the convergence of the analysis. As a general rule the Master surface should always be the stiffer part and/or the part with the coarser mesh, if a load is applied in one of two equal elements (like the plates of this study), one should choose the part that is taking the load as the Master.

Another important factor to keep in mind when defining the Master and the Slave surface is that the Master surface is allowed to penetrate the Slave surface, the opposite, on the other hand, is not allowed.

For this problem there are two possible solutions in Abaqus. The first and most popular is to use a General Contact approach, where Abaqus assumes that all parts can touch with each other with a single friction for all parts, but it can lead to convergence error when used with bolt clearances in implicit analysis, it does not allow the user to customize different frictions for different surfaces and also demands more computational time for an analysis.

A second approach is a Surface-to-Surface modelling, which allows the user to select different interaction properties between surfaces and whether or not they will interact at all. In the Surface-to-Surface approach two options can be chosen, Finite Sliding or Small Sliding.

Small Sliding will consider the relationship between Master and Slave nodes in the beginning of the analysis and will keep this relationship throughout the analysis without changing it. This option can be useful when dealing with low displacement or when disregarding clearance, but when clearance is involved this option is not ideal and leads to convergence problems.

The other option is the Finite Sliding, which allows for larger displacements, this option takes more computational time in comparison to the Small Sliding, since it has an interactive calculation between the surfaces, but provides a better solution for clearance problems. The Finite Sliding can be discretized by "Surface to Surface" or by "Node to Surface", "Surface to Surface" is more accurate and it was the option chosen for this experiment. For an implicit analysis the approach recommended is the Surface-to-Surface Finite Sliding with friction.

The interaction properties between the elements were modelled with Normal and Tangential Behavior. A study of friction (Figure 3.13) was conducted to see which configuration would yield to more accurate results, the study showed that when disregarding the bolt clearance, a very low friction estimates the overall behavior well and the final load, but overestimates the initial force in comparison to the measured data by Konkong and Phuvoravan, 2017; a study considering the clearance and with a very low friction (0.01) presents a similar behavior to the experimental data, but it underestimates the force applied; a simulation considering the bolt clearance with a higher friction (0.2, which is also the maximum feasible value for steel) estimates the final load and entire behavior of the specimen very well and a higher friction helps to reach convergence.

For this study the contact between elements all were modelled with a friction coefficient of 0.2.
3.2. FEM Model

Another interaction used for this model was an MPC (Beam) Constraint to impose a prescribed displacement in the top plate, if used properly it will render the same results as of a load applied directly to the surface, as it can be seen in Figure 3.14.

3.2.8 Reaching convergence

To reach convergence is a common problem when dealing with bolted connections and clearance, there are several strategies one can use to overcome those issues using an implicit analysis. The author advises the Abaqus user to start with a simple model and add the nonlinearities as the user better grasps the problems behavior.

A few suggestions for overcoming the convergence issue are:

- Allow the non-linear effects of large deformations and displacements;
• If possible, start with contact before applying any load;
• Insert a Dissipation Energy Factor;
• Allow a very large number of increments (such as $10^6$);
• Allow for very small steps to happen (such as $10^{-10}$);
• Insert friction in the contact between the parts;
• Apply a Displacement instead of a Load;
• Use a dynamic implicit quasi-static analysis if a static general analysis is not converging.

One must keep in mind that whenever inserting friction or a dissipation energy factor, for example, it can affect the quality of the results, so it is important to verify how they affect the results to model them properly.

Another alternative to overcome convergence problems is using an explicit analysis, however to investigate matter with this strategy, it is advised to mind the time of the step to make sure the analysis is working as a Quasi-static analysis and if the release of internal and kinetic energy of the structure is working a quasi-static analysis would (where the kinetic energy is less than 5% of the internal energy as a thumb rule).

For this kind of simulation the author suggests to only use explicit analysis if one has a lot of experimental data to properly model the behavior.

### 3.3 Validation of the Model

To have extra certainty about the accuracy of the proposed model for the investigation, four different experiments were modelled with the same configurations as the ones used in the investigation.

Given the lack of data with single shear plane bolted connections with hot-rolled plates, the date used for validation was of cold-formed steel plates, whereas the failures (Section 2.1) for those analyses are of the plate and not of the bolt. The experiments used for the validation were conducted by Konkong and Phuvoravan, 2017, Konkong, 2017 and Hedeyat, Afzadi, and Iranpour, 2017.

The bolt configurations of the experiments were: two single bolts, one double bolt and a quadruple bolt (Figure 3.15). All the experiments used G550 cold formed steel plates and either 8.8 or 10.9 bolts, all of them had one single shear plane.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Average Results (kN)</th>
<th>FEM Result (kN)</th>
<th>$P_{test}/P_{FEM}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>22.92</td>
<td>22.50</td>
<td>1.019</td>
</tr>
<tr>
<td>Single</td>
<td>10.75</td>
<td>10.85</td>
<td>0.991</td>
</tr>
<tr>
<td>Double</td>
<td>21.25</td>
<td>21.04</td>
<td>1.010</td>
</tr>
<tr>
<td>Quadruple</td>
<td>37.40</td>
<td>40.98</td>
<td>0.913</td>
</tr>
</tbody>
</table>

As it can be seen in Table 3.2 and Figure 3.16, the FEM model captured the behavior of the experiment and estimated the final load with very good precision. These
results provided confidence that the outcomes of the simulations for a thicker hot-rolled steel plate are reliable.

**Figure 3.15:** Validated Experiments.

**Figure 3.16:** Validation of Single Bolt.
3.4 Parametric Study

To conduct any parametric study it means to run several simulations with small differences between them, to evaluate the effect of changing one single parameter.

In order to perform a parametric study in Abaqus several simulations must be performed, given the complexity of the model each simulation can take over several hours to be completed and the modelling a 3D structure can be time consuming as well. To surpass those issues and perform an optimized analysis, the author used programming tools like Python and Matlab.

Python is the programming language used for Abaqus and it is a very popular programming language, with a minimum knowledge of programming skills one can program python scripts to generate an Abaqus model, run a simulation and extract the data of interest in an automatized process.

Matlab is a very powerful calculation tool that can calculate desired parameters for the user and can control Abaqus and run simulations without having to use the Abaqus Graphic User Interface, this allows the user to have a faster analysis, to perform them in a loop, to extract the results automatically and, if the user computer does not dispose of hundreds of gigabytes of free space, to delete non interesting data generated by Abaqus to avoid running out of space during the calculations.

As a good practice the author suggests the use of this routine (Figure 3.17) or a similar one to perform parametric analyses.

One example of the scripts developed for this project can be found in the Appendix B.
Figure 3.17: Parametric Flowchart.
Chapter 4

Results

In this chapter the results obtained from the FEM analyses will be presented. The results consists of Force X Displacement curves, comparisons between the FEM analyses and the Eurocode and Ultimate Bearing Curves for different types of joint configuration in relation to the bolt hole misalignment.

4.1 General Results

In this chapter the results for the 7 different cases simulated (Table 3.1) are presented in a tabular matter (Table 4.1 and Table 4.2). All the results were extracted from the Abaqus output files with the aid of an Matlab script and then transferred to Excel to plot the graphs.

The values from the tables represent the Ultimate Bearing Capacity in comparison to the benchmark result (when $\Delta d$ is equal to 0.0 mm).

<table>
<thead>
<tr>
<th>$\Delta d$ (mm)</th>
<th>$t = 5\text{ mm}$</th>
<th>$t = 10\text{ mm}$</th>
<th>$t = 10\text{ mm}$</th>
<th>$t = 10\text{ mm}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d = 12\text{ mm}$</td>
<td>$d = 12\text{ mm}$</td>
<td>$d = 16\text{ mm}$</td>
<td>$d = 20\text{ mm}$</td>
</tr>
<tr>
<td>0.00</td>
<td>100.0%</td>
<td>100.0%</td>
<td>100.0%</td>
<td>100.0%</td>
</tr>
<tr>
<td>0.20</td>
<td>98.0%</td>
<td>97.9%</td>
<td>98.5%</td>
<td>97.2%</td>
</tr>
<tr>
<td>0.40</td>
<td>96.2%</td>
<td>95.4%</td>
<td>96.8%</td>
<td>93.2%</td>
</tr>
<tr>
<td>0.60</td>
<td>94.2%</td>
<td>92.2%</td>
<td>95.2%</td>
<td>79.8%</td>
</tr>
<tr>
<td>0.80</td>
<td>90.4%</td>
<td>84.1%</td>
<td>93.1%</td>
<td>61.3%</td>
</tr>
<tr>
<td>1.00</td>
<td>86.3%</td>
<td>72.9%</td>
<td>89.1%</td>
<td>51.5%</td>
</tr>
<tr>
<td>1.20</td>
<td>78.0%</td>
<td>58.3%</td>
<td>83.1%</td>
<td>51.5%</td>
</tr>
<tr>
<td>1.40</td>
<td>70.8%</td>
<td>51.3%</td>
<td>76.1%</td>
<td>51.4%</td>
</tr>
<tr>
<td>1.60</td>
<td>59.1%</td>
<td>51.3%</td>
<td>68.1%</td>
<td>51.5%</td>
</tr>
<tr>
<td>1.80</td>
<td>50.9%</td>
<td>51.3%</td>
<td>56.5%</td>
<td>51.5%</td>
</tr>
<tr>
<td>2.00</td>
<td>50.9%</td>
<td>51.4%</td>
<td>51.2%</td>
<td>51.6%</td>
</tr>
</tbody>
</table>
Chapter 4. Results

Table 4.2: Triple Bolt Results.

<table>
<thead>
<tr>
<th>$\Delta d$ (mm)</th>
<th>$\Delta d = 12$ mm</th>
<th>$\Delta d = 20$ mm</th>
<th>$\Delta d = 20$ mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>0.20</td>
<td>97.71%</td>
<td>96.60%</td>
<td>95.69%</td>
</tr>
<tr>
<td>0.40</td>
<td>95.20%</td>
<td>93.61%</td>
<td>90.20%</td>
</tr>
<tr>
<td>0.60</td>
<td>92.67%</td>
<td>88.91%</td>
<td>73.94%</td>
</tr>
<tr>
<td>0.80</td>
<td>88.06%</td>
<td>78.59%</td>
<td>66.93%</td>
</tr>
<tr>
<td>1.00</td>
<td>82.15%</td>
<td>67.11%</td>
<td>66.96%</td>
</tr>
<tr>
<td>1.20</td>
<td>73.15%</td>
<td>67.04%</td>
<td>67.11%</td>
</tr>
<tr>
<td>1.40</td>
<td>66.62%</td>
<td>67.02%</td>
<td>67.08%</td>
</tr>
<tr>
<td>1.60</td>
<td>66.60%</td>
<td>66.83%</td>
<td>67.04%</td>
</tr>
<tr>
<td>1.80</td>
<td>66.69%</td>
<td>66.83%</td>
<td>66.98%</td>
</tr>
<tr>
<td>2.00</td>
<td>66.80%</td>
<td>66.92%</td>
<td>66.90%</td>
</tr>
</tbody>
</table>

Figure 4.1 helps to better understand what happened in the analyses, essentially due to the bolt hole mismatch one of the bolts received all the load for a given $\Delta d$, as a consequence this bolt yielded sooner than the other one and compromised the total bearing capacity of the joint. In the upper left picture (1) is before the load is applied, the upper right (2) model presents the left bolt already under stress, in the bottom left (3) model the left bolt has already failed and now the right bolt is bearing the load, in the bottom left (4) model both bolts have failed.

The force displacement curves in Figure 4.2 explain how the bolt hole misalignment affected the total bearing resistance, where once the first bolt failed the total resistance of the joint was highly compromised. It is possible to observe how the bolt hole misalignment can affect the overall resistance of the joint.

Figure 4.1: Bolt Failure - FEM Analysis.
4.1. General Results

The Table 4.3 makes a comparison between the bearing capacity of the Eurocode and the bearing capacity of the FEM worst case scenario (Δd = 2.0 mm), it is important to highlight that the FEM model needs further verification regarding those values, given some assumptions and simplifications of the model the final load could be different, however they do illustrate the possibility of the Eurocode overestimating the ultimate bearing capacity of the joint.

<table>
<thead>
<tr>
<th>Bolt</th>
<th>2 Bolts EC (kN)</th>
<th>FEM (kN)</th>
<th>Diff</th>
<th>3 Bolts EC (kN)</th>
<th>FEM (kN)</th>
<th>Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>M12</td>
<td>86.9</td>
<td>60.3</td>
<td>44.1%</td>
<td>130.3</td>
<td>117.7</td>
<td>10.7%</td>
</tr>
<tr>
<td>M16</td>
<td>154.5</td>
<td>107.3</td>
<td>43.9%</td>
<td>231.6</td>
<td>209.5</td>
<td>10.6%</td>
</tr>
<tr>
<td>M20</td>
<td>241.3</td>
<td>167.0</td>
<td>44.5%</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Detailed results for the double bolt case with a diameter of 16 millimeters and a thickness of 10 millimeters case can be seen on Appendix A.
4.2 Bearing Resistance Curves

The following figures display the maximum capacity curves for all seven cases listed in Table 3.1, comparing the effect of the bolt hole misalignment by bolt diameter, plate thickness and bolt configuration.

**Figure 4.3:** Double Bolt - Diameter = 12 mm.

**Figure 4.4:** Double Bolt - Thickness = 10 mm.
4.2. Bearing Resistance Curves

**Figure 4.5:** Diameter = 12 mm - Thickness = 0.5 mm.

**Figure 4.6:** Diameter = 12 mm - Thickness = 10 mm.

**Figure 4.7:** Diameter = 16 mm - Thickness = 10 mm.
Figure 4.8: Triple Bolt - Thickness = 10 mm.
Chapter 5

Discussion and Further Research

5.1 Discussion

As it can be seen in the results presented in Chapter 4 two main conclusions can be drawn.

First, the misalignment of bolt holes can vastly influence the overall resistance of a bolted connection. Even in the M12 bolt, in which the EN 1090-2 states that a clearance of only 1 millimeter is accepted in contrast to larger bolt diameters that require at least 2 millimeters, the bolt hole misalignment can have a huge influence on the ULS of the joint depending on the thickness of the plate.

Both the diameter and the thickness have an effect in this phenomena, given that the thicker the plate, the faster the resistance would diminish, whereas the inverse occurred with the diameter, the larger the diameter the smoother was the reduction of the strength. The EC 1993 1-8 states that if a bolt hole has an oversized clearance a reduction factor of 0.8 should be applied to account for the oversize for the bearing resistance for the bolt, however it does not contemplate the number of bolts, how they are distributed in the plate (only if they are edge or inner bolts) or for the shear resistance of the bolt.

It is important to notice that the Eurocode uses underestimated values for the $f_{ub}$ of the bolt, has safety factors and the FEM analysis has some simplifications of its own that have influenced the final bearing capacity, however when analyzing Table 4.3 one can notice that for worst case scenarios the Eurocode can be overestimating the ULS.

The second conclusion is that the more bolts a connections has the less significant the effect of the mismatch might be, in a scenario where there are only two bolts bearing all the load and one fails, the resistance of the connection will suffer drastically, whereas a with a larger number of bolts, the reduction will not be as significant since there will be a large number of bolts left to bear the load.

This research aimed to investigate the effect of bolt clearance and its mismatch on bolted connections, the results indicates that it has a significant effect for some cases and that the current European standard does not take the bolt hole misalignment into account and, as such, it could be overestimating the bearing capacity of bolted connections in some scenarios scenarios.

5.2 Further Research

The author suggests the following for continuing the research:

- Use experimental data to validate the findings;
• Investigate other kinds of slip-joints and observing if the pre-loading of the bolts has any effect in the bearing resistance of misalignment bolted connections;

• Model a more detailed bolt and observe if it has any effect in the phenomena investigated;

• Investigate how different contact angles influence the final resistance in this scenario, however when doing so symmetry would not be possible;

• Investigate the same problem for a double shear plane configuration;

• The disposition of the bolts studied was always the same, an interesting topic of study would be the effect of bolt hole misalignment in different disposition and in different numbers;

• Model damage behavior in the bolt material. To simplify the gathering of results and the FEM model, damage was not modelled, however the failure was imposed with the plastic behavior of the material.
Bibliography


Appendix A

Detailed Results

In this appendix the results for one of the cases studied can be found, a double bolt case with a diameter of 16 millimeters and a thickness of 10 millimeters. A Force X Displacement graph for every single $\Delta d$, a graph combining all of the Force X Displacement and a curve representing the ULS of the structure in comparison to the best case scenario ($\Delta d = 0$).

The Force x Displacement data for all the analyses can be found here:

https://www.dropbox.com/s/r287xf8m8zv527y/FdAA_Excel.zip?dl=0

<table>
<thead>
<tr>
<th>$\Delta d$</th>
<th>ULS (kN)</th>
<th>Diff (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>209</td>
<td>100</td>
</tr>
<tr>
<td>0.2</td>
<td>204</td>
<td>98</td>
</tr>
<tr>
<td>0.4</td>
<td>199</td>
<td>95</td>
</tr>
<tr>
<td>0.6</td>
<td>192</td>
<td>92</td>
</tr>
<tr>
<td>0.8</td>
<td>176</td>
<td>84</td>
</tr>
<tr>
<td>1.0</td>
<td>152</td>
<td>73</td>
</tr>
<tr>
<td>1.2</td>
<td>122</td>
<td>58</td>
</tr>
<tr>
<td>1.4</td>
<td>107</td>
<td>51</td>
</tr>
<tr>
<td>1.6</td>
<td>107</td>
<td>51</td>
</tr>
<tr>
<td>1.8</td>
<td>107</td>
<td>51</td>
</tr>
<tr>
<td>2.0</td>
<td>107</td>
<td>51</td>
</tr>
</tbody>
</table>
**Figure A.1:** ULS Curve - Double Bolt - Diameter = 16 mm - Thickness = 10 mm.

**Figure A.2:** Force x Displacement - \( \Delta d = 0.0 \) mm.
Appendix A. Detailed Results

**Figure A.3:** Force x Displacement - $\Delta d = 0.2$ mm.

**Figure A.4:** Force x Displacement - $\Delta d = 0.4$ mm.
Figure A.5: Force x Displacement - $\Delta d = 0.6$ mm.

Figure A.6: Force x Displacement - $\Delta d = 0.8$ mm.
Figure A.7: Force x Displacement - $\Delta d = 1.0$ mm.

Figure A.8: Force x Displacement - $\Delta d = 1.2$ mm.
Figure A.9: Force x Displacement - $\Delta d = 1.4$ mm.

Figure A.10: Force x Displacement - $\Delta d = 1.6$ mm.
**Figure A.11:** Force x Displacement - $\Delta d = 1.8$ mm.

**Figure A.12:** Force x Displacement - $\Delta d = 2.0$ mm.
Figure A.13: Force x Displacement - All $\Delta d$. 
Appendix B

Developed Codes

For each type of simulation (Single, Double and Triple Bolt) a folder with similar files were created. In said folder the following codes were developed:

- **Main Control**: A Matlab file that controlled the input parameters for the simulation and executed the other files in order to perform the parametric study.
- **Var**: A python file with the variables that were used to create the Abaqus File.
- **All Parts**: A python script that generated a CAE file with the variables from the Var.py file and executed the analysis.
- **Material**: A python file with material properties that the All Parts Script read to define the analysis material. It is easier to edit properties by doing so.
- **Mesh**: A python file with the mesh configuration that the All Parts Script read to define the analysis mesh. It saves time to have it on a different file to do a mesh study using the Graphic User Interface.
- **Gather Data**: A python file that opens the ODB file calculated by the All-Parts.py and extracts the data of interest.
- **Delete**: A python file that deletes non-interesting files produced by Abaqus once the simulations are over.
- **Data Extraction**: A Matlab file developed to extract all the calculate results in an organized manner to Excel spreadsheets. To use this script one must use at least a 2017 Matlab version.

The author is disponibilizing the entirety of the files in this dropbox folder:

https://www.dropbox.com/s/zut9g48aph1kp3m/FdAA_Exjobb.zip?dl=0

MainControl.m:

```matlab
%% Fernando de Abreu Almeida - KTH - Master Thesis %
% First Attempt: 26-09-2018
% Last Update: 12-10-2018
% Script for Double Bolt Configuration
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Clear Command
clear, close all, clc;

%% Input Variables
% Units in N, mm
```
Eccent    = 0.0;
% Eccentricity between Bolt Holes
R_B      = 6.0;
% Radius of the Bolt
%
for R_B  = 6:2:10
for Eccent = 0:0.2:0.8

Clear    = 2.0;
% Clearance of the Bolt

Thickness = 10.0;
% Thickness of the Plate_Top

MeshSize = 1.0;
% Size of the Mesh Around the Holes
MeshPlate = 4.0;
% Size of the Mesh on the Rest of the Plate

ElType1    = 1.0;
% 1 - C3D8, 2- C3D8R or 3 - C3D8I on the Surface with Interaction
ElPlate1 = 1.0;
% 1 - C3D8, 2- C3D8R or 3 - C3D8I on the Surface without Interaction

FrBP = 0.2;
% Friction between Bolt and Plate
FrPP = 0.2;
% Friction between Plates

PrDisp = 6.0;
% Prescribed Displacement

Analysis = 1.0;
% To run an analysis set it as 1.0

Mat = 2.0;
% It controls the materials used
% 1 - G550 Plate 8.8 Bolt
% 2 - S355 Plate 8.8 Bolt
% 3 - To be Defined

% % % % % Variables % % % % %

e1     = 6*R_B;
% Distance between Bolt and end plate (x-axis)
e2     = 8*R_B;
% Distance between Bolt and end plate (y-axis)
p1     = 10*R_B;
% Distance between Bolts

PartSpacing = 10.0;
Appendix B. Developed Codes

% Size of the partition around the Holes
if R_B > 4.0
    PartSpacing = 15.0;
end
if R_B > 7.0
    PartSpacing = 20.0;
end

DistH1 = e1;
% Distance from the edge hole 1
DistH2 = DistH1+p1;
% Distance from the edge hole 2
Overlap = DistH1+DistH2;
% The overlap between the Plates

Length = Overlap+10*R_B;
% Length of the Plate_Top
Width = e2;
% Width of the Plate

R_HN = 2*R_B;
% Radius of the Washer
H_B = 2*Thickness;
% Height of the Bolt (thickness of the plate)
H_HN = Thickness;
% Height of the Washer (plus head and nut)

% if Eccent > Clear
% Debugger for Eccentricity
% Eccent = Clear;
% Only use it if you are not in a Loop
% end

%%% Insert Variables in a Phyton File
mo='noGUI'; % Do not open Abaqus
% mo='script'; % Open Abaqus
delete('./Var.py');
% The . indicates that the file is located in the same directory
fid = fopen('./Var.py', 'w');
% w indicates that it is going to re-write the file
fprintf(fid,'Length = %0.12f\n',Length);
% the n indicates that it is a number, for text use s
fprintf(fid,'Width = %0.12f\n',Width);
fprintf(fid,'R_B = %0.12f\n',R_B);
fprintf(fid,'Clear = %0.12f\n',Clear);
fprintf(fid,'Thickness = %0.12f\n',Thickness);
fprintf(fid,'DistH1 = %0.12f\n',DistH1);
fprintf(fid,'DistH2 = %0.12f\n',DistH2);
fprintf(fid,'Eccent = %0.12f\n',Eccent);
fprintf(fid,'PartSpacing = %0.12f\n',PartSpacing);
fprintf(fid,'Overlap = %0.12f
',Overlap);
fprintf(fid,'MeshSize = %0.12f
',MeshSize);
fprintf(fid,'ElType1 = %0.12f
',ElType1);
fprintf(fid,'MeshPlate = %0.12f
',MeshPlate);
fprintf(fid,'ElPlate1 = %0.12f
',ElPlate1);
fprintf(fid,'R_HN = %0.12f
',R_HN);
fprintf(fid,'H_B = %0.12f
',H_B);
fprintf(fid,'H_HN = %0.12f
',H_HN);
fprintf(fid,'FrBP = %0.12f
',FrBP);
fprintf(fid,'FrPP = %0.12f
',FrPP);
fprintf(fid,'PrDisp = %0.12f
',PrDisp);
fprintf(fid,'Analysis = %0.12f
',Analysis);
fprintf(fid,'Mat = %0.12f
',Mat);
fclose(fid);

%% Run Abaqus with the General File
unix(['abaqus cae ',mo,'=C:\General_Assembly\Abaqus_Files\AllParts.py']);
Unix system
system(['abaqus cae ',mo,'=\AllParts.py']);
%Windows system % Executes and runs the job
system(['abaqus cae ',mo,'=\GatherData.py']);
%Windows system % Extracts the information from the .odb

end
% End of the Loop for the Eccent
end
% End of the Loop for the Radius

%%% Deletes Files that are not interesting for the Post-Processing
system('=\Delete.py');
% Runs Python Script

clearvars Analysis ans Clear DistH1 DistH2 e1 e2 Eccent ElType1 Elplate1
clearvars fid FrBP FrPP H_B H_HN Length Mat MeshPlate MeshSize mo
clearvars Overlap p1 PartSpacing PrDisp R_B R_HN Thickness Width
Appendix B. Developed Codes

Var.py:

Length = 192.000000000000
Width = 48.000000000000
R_B = 6.000000000000
Clear = 2.000000000000
Thickness = 10.000000000000
DistH1 = 36.000000000000
DistH2 = 96.000000000000
Eccent = 0.800000000000
PartSpacing = 15.000000000000
Overlap = 132.000000000000
MeshSize = 1.000000000000
ElType1 = 1.000000000000
MeshPlate = 4.000000000000
ElPlate1 = 1.000000000000
R_HN = 12.000000000000
H_B = 20.000000000000
H_HN = 10.000000000000
FrBP = 0.200000000000
FrPP = 0.200000000000
PrDisp = 6.000000000000
Analysis = 1.000000000000
Mat = 2.000000000000

AllParts.py:

    # Created by Fernando de Abreu Almeida 27-9-2018
    from part import *
    from material import *
    from section import *
    from assembly import *
    from step import *
    from interaction import *
    from load import *
    from mesh import *
    from optimization import *
    from job import *
    from sketch import *
    from visualization import *
    from connectorBehavior import *

    # ============== Input =================== #
    import os
    #script_path = os.path.dirname(os.path.abspath( __file__ ))
    script_path = os.getcwd()
    execfile(script_path+'\Var.py')
    # Gets the address of the file and executes the right Var.py
    # They all have to be in the same folder to work
# os.chdir(r"C:\General_Assembly\Results")
#folder = os.path.dirname(os.path.dirname(os.path.abspath( __file__ )))
folder = os.path.dirname(os.path.abspath(script_path))
os.chdir(folder+'\Results')

# Mesh, defines the type of Element
ElType = C3D8
if ElType1 ==2:
  ElType = C3D8R
elif ElType1 == 3:
  ElType = C3D8I

ElPlate = C3D8R
if ElPlate1 ==1:
  ElPlate = C3D8
elif ElPlate1 == 3:
  ElPlate = C3D8I

mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=500.0)
mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(0.0, 0.0),
  point2=(Length, Width))

mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
  Length-DistH1-Eccent, 0.0), point1=(Length-DistH1+R_B+Clear/2.0-Eccent, 0.0))

mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
  Length-DistH2, 0.0), point1=(Length-DistH2+R_B+Clear/2.0, 0.0))

mdb.models['Model-1'].sketches['__profile__'].autoTrimCurve(curve1=
  mdb.models['Model-1'].sketches['__profile__'].geometry[7], point1=(
  Length-DistH2, -(R_B+Clear/2.0)))

mdb.models['Model-1'].sketches['__profile__'].autoTrimCurve(curve1=
  mdb.models['Model-1'].sketches['__profile__'].geometry[5], point1=(
  Length-DistH2, 0.0))

mdb.models['Model-1'].sketches['__profile__'].autoTrimCurve(curve1=
  mdb.models['Model-1'].sketches['__profile__'].geometry[6], point1=(
  Length-DistH1, -(Length-DistH1+R_B+Clear/2.0)))

mdb.models['Model-1'].sketches['__profile__'].autoTrimCurve(curve1=
  mdb.models['Model-1'].sketches['__profile__'].geometry[9], point1=(
  Length-DistH1, 0.0))

mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Plate_Bottom', type=
  DEFORMABLE_BODY)

mdb.models['Model-1'].parts['Plate_Bottom'].BaseSolidExtrude(depth=Thickness, sketch=
  mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']

# =========== Plate Bottom =========== #
# =========== Geometry ================ #

mdb.models['Model-1'].parts['Plate_Bottom'].DatumPlaneByPrincipalPlane(offset=
  PartSpacing, principalPlane=XZPLANE)

mdb.models['Model-1'].parts['Plate_Bottom'].DatumPlaneByPrincipalPlane(offset=
  PartSpacing, principalPlane=YZPLANE)
Length-DistH1+PartSpacing, principalPlane=YZPLANE)
mdb.models['Model-1'].parts['Plate_Bottom'].DatumPlaneByPrincipalPlane(offset=Length-DistH1-PartSpacing, principalPlane=YZPLANE)
mdb.models['Model-1'].parts['Plate_Bottom'].DatumPlaneByPrincipalPlane(offset=Length-DistH2+PartSpacing, principalPlane=YZPLANE)
mdb.models['Model-1'].parts['Plate_Bottom'].DatumPlaneByPrincipalPlane(offset=Length-DistH2-PartSpacing, principalPlane=YZPLANE)
mdb.models['Model-1'].parts['Plate_Bottom'].DatumPlaneByPrincipalPlane(offset=Length-Overlap, principalPlane=YZPLANE)

mdb.models['Model-1'].parts['Plate_Bottom'].PartitionCellByDatumPlane(cells=mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask(('[#1 ]', ), ), datumPlane=mdb.models['Model-1'].parts['Plate_Bottom'].datums[2])

mdb.models['Model-1'].parts['Plate_Bottom'].PartitionCellByDatumPlane(cells=mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask(('[#3 ]', ), ), datumPlane=mdb.models['Model-1'].parts['Plate_Bottom'].datums[7])

mdb.models['Model-1'].parts['Plate_Bottom'].PartitionCellByDatumPlane(cells=mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask(('[#5 ]', ), ), datumPlane=mdb.models['Model-1'].parts['Plate_Bottom'].datums[6])

mdb.models['Model-1'].parts['Plate_Bottom'].PartitionCellByDatumPlane(cells=mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask(('[#3 ]', ), ), datumPlane=mdb.models['Model-1'].parts['Plate_Bottom'].datums[5])

mdb.models['Model-1'].parts['Plate_Bottom'].PartitionCellByDatumPlane(cells=mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask(('[#c ]', ), ), datumPlane=mdb.models['Model-1'].parts['Plate_Bottom'].datums[4])

mdb.models['Model-1'].parts['Plate_Bottom'].PartitionCellByDatumPlane(cells=mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask(('[#3 ]', ), ), datumPlane=mdb.models['Model-1'].parts['Plate_Bottom'].datums[3])

# ================ Mesh =============================== #
mdb.models['Model-1'].parts['Plate_Bottom'].setElementType(elemTypes=(ElemType(elemCode=ElPlate, elemLibrary=STANDARD, secondOrderAccuracy=OFF,
Appendix B. Developed Codes

kinematicSplit=AVERAGE_STRAIN, hourglassControl=DEFAULT,
distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),
ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask((
  ['#a000'], ), ), ))
mdb.models['Model-1'].parts['Plate_Bottom'].setElementType(elemTypes=(ElemType(
  elemCode=ElType, elemLibrary=STANDARD, secondOrderAccuracy=OFF,
distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),
ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask((
  ['#5ff0'], ), ), ))
mdb.models['Model-1'].parts['Plate_Bottom'].setMeshControls(algorithm=
MEDIAL_AXIS, regions=
  mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask((
  ['#2100'], ), ), ))
mdb.models['Model-1'].parts['Plate_Bottom'].Set(cells=
  mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask((
  ['#2100'], ), ), name='Set-Mesh')
mdb.models['Model-1'].parts['Plate_Bottom'].seedPart(deviationFactor=0.1,
minSizeFactor=0.1, size=MeshPlate)
mdb.models['Model-1'].parts['Plate_Bottom'].seedEdgeBySize(constraint=FINER,
deviationFactor=0.1, edges=
  mdb.models['Model-1'].parts['Plate_Bottom'].edges.getSequenceFromMask((
  '#6067c71 #633fe720 #80d80000 #1000000'), ), ), minSizeFactor=0.1, size=MeshSize)
mdb.models['Model-1'].parts['Plate_Bottom'].generateMesh()

# ============== Sets and Surfaces =============== #
mdb.models['Model-1'].parts['Plate_Bottom'].Set(faces=
  mdb.models['Model-1'].parts['Plate_Bottom'].faces.getSequenceFromMask((
  '#120004 #2a002080'), ), ), name='Set-SymmBP')
mdb.models['Model-1'].parts['Plate_Bottom'].Set(faces=
  mdb.models['Model-1'].parts['Plate_Bottom'].faces.getSequenceFromMask((
  '#000000 #480000'), ), ), name='Set-BCBP')
mdb.models['Model-1'].parts['Plate_Bottom'].Surface(name='Surf-BP_Top',
side1Faces=
  mdb.models['Model-1'].parts['Plate_Bottom'].faces.getSequenceFromMask((
  '#80a12048 #4010504'), ), ))
mdb.models['Model-1'].parts['Plate_Bottom'].Surface(name='Surf-BP_SL',
side1Faces=
  mdb.models['Model-1'].parts['Plate_Bottom'].faces.getSequenceFromMask((
  '#0 #10000000'), ), ))
mdb.models['Model-1'].parts['Plate_Bottom'].Surface(name='Surf-BP_SR',
side1Faces=
  mdb.models['Model-1'].parts['Plate_Bottom'].faces.getSequenceFromMask((
  '#0 #4000000'), ), ))
mdb.models['Model-1'].parts['Plate_Bottom'].Surface(name='Surf-BP_Bot',
side1Faces=
  mdb.models['Model-1'].parts['Plate_Bottom'].faces.getSequenceFromMask((
  '#22409090 #8004841'), ), ))

##
# ============= Plate Top ======================= #
# ============== Geometry ============================== #

mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=500.0)

mdb.models['Model-1'].sketches['__profile__'].rectangle(point1=(0.0, 0.0),
point2=(Length, Width))

mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
DistH1, 0.0), point1=(DistH1+R_B+Clear/2.0, 0.0))

mdb.models['Model-1'].sketches['__profile__'].CircleByCenterPerimeter(center=(
DistH2, 0.0), point1=(DistH2+R_B+Clear/2.0, 0.0))

mdb.models['Model-1'].sketches['__profile__'].autoTrimCurve(curve1=
mdb.models['Model-1'].sketches['__profile__'].geometry[6], point1=(
DistH1, -R_B-Clear/2.0))

mdb.models['Model-1'].sketches['__profile__'].autoTrimCurve(curve1=
mdb.models['Model-1'].sketches['__profile__'].geometry[5], point1=(
DistH1, 0.0))

mdb.models['Model-1'].sketches['__profile__'].autoTrimCurve(curve1=
mdb.models['Model-1'].sketches['__profile__'].geometry[7], point1=(
DistH2, -R_B-Clear/2.0))

mdb.models['Model-1'].sketches['__profile__'].autoTrimCurve(curve1=
mdb.models['Model-1'].sketches['__profile__'].geometry[9], point1=(
DistH2, 0.0))

mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Plate_Top', type=
DEFORMABLE_BODY)

mdb.models['Model-1'].parts['Plate_Top'].BaseSolidExtrude(depth=Thickness, sketch=
mdb.models['Model-1'].sketches['__profile__'])

del mdb.models['Model-1'].sketches['__profile__']

# ================ Datum and Partition =================== #

mdb.models['Model-1'].parts['Plate_Top'].DatumPlaneByPrincipalPlane(offset=
PartSpacing , principalPlane=XZPLANE)

mdb.models['Model-1'].parts['Plate_Top'].DatumPlaneByPrincipalPlane(offset=
DistH1-PartSpacing , principalPlane=YZPLANE)

mdb.models['Model-1'].parts['Plate_Top'].DatumPlaneByPrincipalPlane(offset=
DistH1+PartSpacing , principalPlane=YZPLANE)

mdb.models['Model-1'].parts['Plate_Top'].DatumPlaneByPrincipalPlane(offset=
DistH2-PartSpacing , principalPlane=YZPLANE)

mdb.models['Model-1'].parts['Plate_Top'].DatumPlaneByPrincipalPlane(offset=
DistH2+PartSpacing , principalPlane=YZPLANE)

mdb.models['Model-1'].parts['Plate_Top'].DatumPlaneByPrincipalPlane(offset=
Overlap , principalPlane=YZPLANE)

mdb.models['Model-1'].parts['Plate_Top'].PartitionCellByDatumPlane(cells=
mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask(('[#1 ]', ), ),
datumPlane= mdb.models['Model-1'].parts['Plate_Top'].datums[2])

mdb.models['Model-1'].parts['Plate_Top'].PartitionCellByDatumPlane(cells=
mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask(('[#3 ]', ), ),
datumPlane= mdb.models['Model-1'].parts['Plate_Top'].datums[3])

mdb.models['Model-1'].parts['Plate_Top'].PartitionCellByDatumPlane(cells=
mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask(('[#5 ]', ), ),
datumPlane=
Appendix B. Developed Codes

mdb.models['Model-1'].parts['Plate_Top'].datums[4]

mdb.models['Model-1'].parts['Plate_Top'].PartitionCellByDatumPlane(cells=
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#[14 ]', ), ), datumPlane=
    mdb.models['Model-1'].parts['Plate_Top'].datums[5])

mdb.models['Model-1'].parts['Plate_Top'].PartitionCellByDatumPlane(cells=
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#[3 ]', ), ), datumPlane=
    mdb.models['Model-1'].parts['Plate_Top'].datums[6])

mdb.models['Model-1'].parts['Plate_Top'].PartitionCellByDatumPlane(cells=
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#[c ]', ), ), datumPlane=
    mdb.models['Model-1'].parts['Plate_Top'].datums[7])

# =============== Material Properties ================================ #

mdb.models['Model-1'].Material(name='Material-Plate')

mdb.models['Model-1'].materials['Material-Plate'].Elastic(table=((213000.0, 0.3), ))

mdb.models['Model-1'].materials['Material-Plate'].Plastic(table=((607.0, 0.0), (670.0, 0.058)))

mdb.models['Model-1'].HomogeneousSolidSection(material='Material-Plate', name=
    'Section-PlateTop', thickness=None)

mdb.models['Model-1'].parts['Plate_Top'].SectionAssignment(offset=0.0, offsetField='',
    offsetType=MIDDLE_SURFACE, region=Region(cells=mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask(mask=('
        [#fff ]', ), )), sectionName='Section-PlateTop', thicknessAssignment=FROM_SECTION)

# =============== Mesh ================================ #

mdb.models['Model-1'].parts['Plate_Top'].seedPart(deviationFactor=0.1, minSizeFactor=0.1, size=MeshPlate)

mdb.models['Model-1'].parts['Plate_Top'].setElementType(elemTypes=(ElemType(
    elemCode=ElPlate, elemLibrary=STANDARD, secondOrderAccuracy=OFF, kinematicSplit=AVERAGE_STRAIN, hourglassControl=DEFAULT, distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),
    ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#[3 ]', ), ), ))

mdb.models['Model-1'].parts['Plate_Top'].setElementType(elemTypes=(ElemType(
    elemCode=ElType, elemLibrary=STANDARD, secondOrderAccuracy=OFF, distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),
    ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#[fffc ]', ), ))

mdb.models['Model-1'].parts['Plate_Top'].setMeshControls(algorithm=MEDIAL_AXIS, regions=mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
    ['#48 ]', ), ))

mdb.models['Model-1'].parts['Plate_Top'].Set(cells=
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#[48 ]', ), ), name='Set-Mesh_TP')

mdb.models['Model-1'].parts['Plate_Top'].seedEdgeBySize(constraint=FINER,
deviationFactor=0.1, edges=
mdb.models['Model-1'].parts['Plate_Top'].edges.getSequenceFromMask((
    '#e4c40000 #fe68181b #80d8061f #1 ', ), ),
minSizeFactor=0.1, size=MeshSize)

mdb.models['Model-1'].parts['Plate_Top'].generateMesh()

# ================ Sets and Surfaces =============== #

mdb.models['Model-1'].parts['Plate_Top'].Set(faces=
    mdb.models['Model-1'].parts['Plate_Top'].faces.getSequenceFromMask((
    '#44000900 #2a00200 ', ), ), name='Set-SymmTP')

mdb.models['Model-1'].parts['Plate_Top'].Set(faces=
    mdb.models['Model-1'].parts['Plate_Top'].faces.getSequenceFromMask((
    '#0 #120000 ', ), ), name='Set-BCTP')

mdb.models['Model-1'].parts['Plate_Top'].Surface(name='Surf-TP_Top',
    side1Faces=
    mdb.models['Model-1'].parts['Plate_Top'].faces.getSequenceFromMask((
    '#22409090 #4010502 ', ), ))

mdb.models['Model-1'].parts['Plate_Top'].Surface(name='Surf-TP_SL',
    side1Faces=
    mdb.models['Model-1'].parts['Plate_Top'].faces.getSequenceFromMask((
    '#0 #100000 ', ), ))

mdb.models['Model-1'].parts['Plate_Top'].Surface(name='Surf-TP_SR',
    side1Faces=
    mdb.models['Model-1'].parts['Plate_Top'].faces.getSequenceFromMask((
    '#0 #400000 ', ), ))

mdb.models['Model-1'].parts['Plate_Top'].Surface(name='Surf-TP_Bot',
    side1Faces=
    mdb.models['Model-1'].parts['Plate_Top'].faces.getSequenceFromMask((
    '#81212048 #8004844 ', ), ))

##

# ======================= Bolt Left ========== #
# Model Bolt ================ #

mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=500.0)

mdb.models['Model-1'].sketches['__profile__'].ConstructionLine(point1=(0.0,
    -250.0), point2=(0.0, 250.0))

mdb.models['Model-1'].sketches['__profile__'].FixedConstraint(entity=
    mdb.models['Model-1'].sketches['__profile__'].geometry[2])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(0.0, H_B+2.0*H_HN),
    point2=( R_HN, H_B+2.0*H_HN))

mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(addUndoState=
    False, entity1=mdb.models['Model-1'].sketches['__profile__'].geometry[3])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(0.0, H_B+2.0*H_HN),
    point2=( R_HN, H_B+2.0*H_HN))

mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[4])

mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[3], entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[4])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_HN, H_B+2.0*H_HN),
    point2=( R_HN, H_B+H_HN))


Appendix B. Developed Codes
Appendix B. Developed Codes

In the context of the developed codes, several constraints and lines are defined within the sketch of the model. Below are the detailed steps:

1. **Vertical Constraint**:
   - `mdb.models['Model-1'].sketches['__profile__'].VerticalConstraint(addUndoState=False, entity=mdb.models['Model-1'].sketches['__profile__'].geometry[5])`

2. **Perpendicular Constraint**:
   - `mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(addUndoState=False, entity1=mdb.models['Model-1'].sketches['__profile__'].geometry[4], entity2=mdb.models['Model-1'].sketches['__profile__'].geometry[5])`
   - `mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_HN, H_B+H_HN), point2=(R_B, H_B+H_HN))`

3. **Horizontal Constraint**:
   - `mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(addUndoState=False, entity=mdb.models['Model-1'].sketches['__profile__].geometry[6])`

4. **Line**:
   - `mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_B, H_B+H_HN), point2=(R_B, H_HN))`

5. **Vertical Constraint**:
   - `mdb.models['Model-1'].sketches['__profile__'].VerticalConstraint(addUndoState=False, entity=mdb.models['Model-1'].sketches['__profile__].geometry[7])`

6. **Perpendicular Constraint**:
   - `mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(addUndoState=False, entity1=mdb.models['Model-1'].sketches['__profile__].geometry[6], entity2=mdb.models['Model-1'].sketches['__profile__].geometry[7])`
   - `mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_B, H_HN), point2=(R_HN, H_HN))`

7. **Horizontal Constraint**:
   - `mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(addUndoState=False, entity=mdb.models['Model-1'].sketches['__profile__].geometry[8])`

8. **Line**:
   - `mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_HN, H_HN), point2=(R_HN, 0.0))`

9. **Vertical Constraint**:
   - `mdb.models['Model-1'].sketches['__profile__'].VerticalConstraint(addUndoState=False, entity=mdb.models['Model-1'].sketches['__profile__].geometry[9])`

10. **Perpendicular Constraint**:
    - `mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(addUndoState=False, entity1=mdb.models['Model-1'].sketches['__profile__].geometry[8], entity2=mdb.models['Model-1'].sketches['__profile__].geometry[9])`
    - `mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_HN, 0.0), point2=(0.0, 0.0))`

Finally, a part is defined with the following specifications:

- `mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Bolt-L', type=DEFORMABLE_BODY)`
Appendix B. Developed Codes

```python
mdb.models['Model-1'].parts['Bolt-L'].BaseSolidRevolve(angle=180.0,
    flipRevolveDirection=OFF, sketch=
    mdb.models['Model-1'].sketches['__profile__'])
del mdb.models['Model-1'].sketches['__profile__']

# ============== Datum and Partition ====================== 
mdb.models['Model-1'].parts['Bolt-L'].DatumPlaneByPrincipalPlane(offset=H_HN,
    principalPlane=XZPLANE)
mdb.models['Model-1'].parts['Bolt-L'].DatumPlaneByPrincipalPlane(offset=H_B+H_HN,
    principalPlane=XZPLANE)
mdb.models['Model-1'].parts['Bolt-L'].PartitionCellByDatumPlane(cells=
    mdb.models['Model-1'].parts['Bolt-L'].cells.getSequenceFromMask((('[#1 ]',
    ), ), datumPlane=mdb.models['Model-1'].parts['Bolt-L'].datums[2])
mdb.models['Model-1'].parts['Bolt-L'].PartitionCellByDatumPlane(cells=
    mdb.models['Model-1'].parts['Bolt-L'].cells.getSequenceFromMask((('[#2 ]',
    ), ), datumPlane=mdb.models['Model-1'].parts['Bolt-L'].datums[3])

# ============ Material ================================ 
mdb.models['Model-1'].Material(name='Material-Bolt')
mdb.models['Model-1'].materials['Material-Bolt'].Elastic(table=((205000.0,
    0.3), ))
mdb.models['Model-1'].materials['Material-Bolt'].Plastic(table=((930.0, 0.0),
    (952.0, 0.039)))
mdb.models['Model-1'].HomogeneousSolidSection(material='Material-Bolt', name=
    'Section-BoltL', thickness=None)

mdb.models['Model-1'].parts['Bolt-L'].SectionAssignment(offset=0.0,
    offsetField='', offsetType=MIDDLE_SURFACE, region=Region(
    cells=mdb.models['Model-1'].parts['Bolt-L'].cells.getSequenceFromMask(
        mask=('[#7 ]', ), ), sectionName='Section-BoltL', thicknessAssignment=
    FROM_SECTION)

# ================= Mesh ==================== 
mdb.models['Model-1'].parts['Bolt-L'].setElementType(elemTypes=(ElemType(
    elemCode= ElType, elemLibrary=STANDARD, secondOrderAccuracy=OFF,
    distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),
    ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
    mdb.models['Model-1'].parts['Bolt-L'].cells.getSequenceFromMask((('[#7 ]',
    ), ), ), ))

mdb.models['Model-1'].parts['Bolt-L'].seedPart(deviationFactor=0.1,
    minSizeFactor=0.1, size=MeshSize)

mdb.models['Model-1'].parts['Bolt-L'].setMeshControls(algorithm=MEDIAL_AXIS,
    regions=mdb.models['Model-1'].parts['Bolt-L'].cells.getSequenceFromMask((
    '[#7 ]', ), ))

mdb.models['Model-1'].parts['Bolt-L'].generateMesh()

# ================== Sets and Surfaces =============== 
mdb.models['Model-1'].parts['Bolt-L'].Set(faces=
    mdb.models['Model-1'].parts['Bolt-L'].faces.getSequenceFromMask((
    '[#6036 ]', ), ), name='Set-SYMM'))
```
Appendix B. Developed Codes

```python
mdb.models['Model-1'].parts['Bolt-L'].Surface(name='Surf-BL_Shank',
    side1Faces=
    mdb.models['Model-1'].parts['Bolt-L'].faces.getSequenceFromMask(['[#200 ]',
    ]),
)

mdb.models['Model-1'].parts['Bolt-L'].Surface(name='Surf-BL_N',
    side1Faces=
    mdb.models['Model-1'].parts['Bolt-L'].faces.getSequenceFromMask(['[#100 ]',
    ]),
)

mdb.models['Model-1'].parts['Bolt-L'].Surface(name='Surf-BL_H',
    side1Faces=
    mdb.models['Model-1'].parts['Bolt-L'].faces.getSequenceFromMask(['[#400 ]',
    ]),
)

##
# ===================== Bolt Right =================== #
# ================ Model Bolt ==================================== #
mdb.models['Model-1'].ConstrainedSketch(name='__profile__', sheetSize=500.0)
mdb.models['Model-1'].sketches['__profile__'].ConstructionLine(point1=(0.0,-
    250.0), point2=(0.0, 250.0))

mdb.models['Model-1'].sketches['__profile__'].FixedConstraint(entity=
    mdb.models['Model-1'].sketches['__profile__'].geometry[2])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(0.0, 0.0), point2=(
    0.0, H_B+2.0*H_HN))

mdb.models['Model-1'].sketches['__profile__'].VerticalConstraint(addUndoState=
    False, entity=mdb.models['Model-1'].sketches['__profile__'].geometry[3])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(0.0, H_B+2.0*H_HN),
    point2=(
    R_HN, H_B+2.0*H_HN))

mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(
    addUndoState=False, entity=
    mdb.models['Model-1'].sketches['__profile__'].geometry[4])

mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(
    addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[3],
    entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[4])

mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(
    addUndoState=False, entity=
    mdb.models['Model-1'].sketches['__profile__'].geometry[4])

mdb.models['Model-1'].sketches['__profile__'].VerticalConstraint(addUndoState=
    False, entity=mdb.models['Model-1'].sketches['__profile__'].geometry[5])

mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(
    addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[4], entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[5])

mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(
    addUndoState=False, entity=
    mdb.models['Model-1'].sketches['__profile__'].geometry[5])

mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(
    addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[5], entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[6])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_B, H_B+H_HN),
    point2=(
    R_B, H_HN))
```

False, entity=mdb.models['Model-1'].sketches['__profile__'].geometry[7])

mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[6], entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[7])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_B, H_HN), point2=(R_HN, H_HN))

mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(addUndoState=False, entity=
    mdb.models['Model-1'].sketches['__profile__'].geometry[8])

mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[7], entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[8])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_HN, H_HN), point2=(R_HN, 0.0))

mdb.models['Model-1'].sketches['__profile__'].VerticalConstraint(addUndoState=False, entity=mdb.models['Model-1'].sketches['__profile__'].geometry[9])

mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[7], entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[8])

mdb.models['Model-1'].sketches['__profile__'].Line(point1=(R_HN, 0.0), point2=(0.0, 0.0))

mdb.models['Model-1'].sketches['__profile__'].HorizontalConstraint(addUndoState=False, entity=
    mdb.models['Model-1'].sketches['__profile__'].geometry[10])

mdb.models['Model-1'].sketches['__profile__'].PerpendicularConstraint(addUndoState=False, entity1=
    mdb.models['Model-1'].sketches['__profile__'].geometry[9], entity2=
    mdb.models['Model-1'].sketches['__profile__'].geometry[10])

mdb.models['Model-1'].Part(dimensionality=THREE_D, name='Bolt-R', type=DEFORMABLE_BODY)

mdb.models['Model-1'].parts['Bolt-R'].BaseSolidRevolve(angle=180.0, flipRevolveDirection=OFF, sketch=
    mdb.models['Model-1'].sketches['__profile__'])

del mdb.models['Model-1'].sketches['__profile__']
Appendix B. Developed Codes

# ============ Material ================================ #
mdb.models['Model-1'].Material(name='Material-Bolt')
mdb.models['Model-1'].materials['Material-Bolt'].Elastic(table=((205000.0, 0.3), ))
mdb.models['Model-1'].materials['Material-Bolt'].Plastic(table=((930.0, 0.0), (952.0, 0.039)))
mdb.models['Model-1'].HomogeneousSolidSection(material='Material-Bolt', name='Section-BoltR', thickness=None)

# ================= Mesh ==================== #
mdb.models['Model-1'].parts['Bolt-R'].setElementType(elemTypes=(ElemType(elemCode=ElType, elemLibrary=STANDARD, secondOrderAccuracy=OFF, distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD), ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(mdb.models['Model-1'].parts['Bolt-R'].cells.getSequenceFromMask((['#7'], ), ), ) )
mdb.models['Model-1'].parts['Bolt-R'].seedPart(deviationFactor=0.1, minSizeFactor=0.1, size=MeshSize)
mdb.models['Model-1'].parts['Bolt-R'].setMeshControls(algorithm=MEDIAL_AXIS, regions=mdb.models['Model-1'].parts['Bolt-R'].cells.getSequenceFromMask((['#7'], ), ) )
mdb.models['Model-1'].parts['Bolt-R'].generateMesh()

# = = = = = Sets and Surfaces = = = = = #
mdb.models['Model-1'].parts['Bolt-R'].Set(faces= mdb.models['Model-1'].parts['Bolt-R'].faces.getSequenceFromMask((['#6036'], ), ), name='Set-SYMM')
mdb.models['Model-1'].parts['Bolt-R'].Surface(name='Surf-BR_Shank', side1Faces= mdb.models['Model-1'].parts['Bolt-R'].faces.getSequenceFromMask((['#200'], ), ) )
mdb.models['Model-1'].parts['Bolt-R'].Surface(name='Surf-BR_N', side1Faces= mdb.models['Model-1'].parts['Bolt-R'].faces.getSequenceFromMask((['#100'], ), ) )
mdb.models['Model-1'].parts['Bolt-R'].Surface(name='Surf-BR_H', side1Faces= mdb.models['Model-1'].parts['Bolt-R'].faces.getSequenceFromMask((['#400'], ), ) )

##

# = = = = = = = = = = Assembly = = = = = = = = = = #
# = = = = = = = = = = Assembly = = = = = = = = = = = = = #
mdb.models['Model-1'].rootAssembly.DatumCsysByDefault(CARTESIAN)
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Bolt-L-1', part=mdb.models['Model-1'].parts['Bolt-L'])
mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Bolt-R-1', part=mdb.models['Model-1'].parts['Bolt-R'])
Appendix B. Developed Codes

mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Plate_Bottom-1', part=mdb.models['Model-1'].parts['Plate_Bottom'])

mdb.models['Model-1'].rootAssembly.Instance(dependent=ON, name='Plate_Top-1', part=mdb.models['Model-1'].parts['Plate_Top'])

# Generates Assembly
mdb.models['Model-1'].rootAssembly.instances['Bolt-R-1'].translate(vector=(50.0, 0.0, 0.0))

mdb.models['Model-1'].rootAssembly.instances['Plate_Bottom-1'].translate(vector=(100.0, 0.0, 0.0))

mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].translate(vector=(300.0, 0.0, 0.0))

# Plate Position
mdb.models['Model-1'].rootAssembly.translate(instanceList=('Plate_Top-1', ), vector=(-(300.0+DistH1-(100.0+Length-DistH2)), 0.0, Thickness))

# Bolts Position
mdb.models['Model-1'].rootAssembly.rotate(angle=270.0, axisDirection=(R_HN/2.0, 0.0, 0.0), axisPoint=(-R_HN/2.0, 0.0, 0.0), instanceList=('Bolt-L-1', 'Bolt-R-1'))

mdb.models['Model-1'].rootAssembly.translate(instanceList=('Bolt-L-1', ), vector=(100.0+Length-DistH2-Clear/2.0, 0.0, H_B+H_HN))

mdb.models['Model-1'].rootAssembly.translate(instanceList=('Bolt-R-1', ), vector=(50.0+Length-DistH1-Clear/2.0, 0.0, H_B+H_HN))

# =================== Interaction ===================== #

mdb.models['Model-1'].ContactProperty('IntProp-BP')

mdb.models['Model-1'].interactionProperties['IntProp-BP'].TangentialBehavior(dependencies=0, directionality=ISOTROPIC, elasticSlipStiffness=None, formulation=PENALTY, fraction=0.005, maximumElasticSlip=FRACTION, pressureDependency=OFF, shearStressLimit=None, slipRateDependency=OFF, table=((FrBP, ), ), temperatureDependency=OFF)

mdb.models['Model-1'].interactionProperties['IntProp-BP'].NormalBehavior(allowSeparation=ON, clearanceAtZeroContactPressure=0.0, constraintEnforcementMethod=PENALTY, contactStiffness=DEFAULT, contactStiffnessScaleFactor=FrBP, pressureOverclosure=HARD, stiffnessBehavior=LINEAR)

mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE, clearanceRegion=omit, createStepName='Initial', datumAxis=None, initialClearance=OMIT, interactionProperty='IntProp-BP', master=mdb.models['Model-1'].rootAssembly.instances['Bolt-L-1'].surfaces['Surf-BL_H'], name='Int-BLH', slave=mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].surfaces['Surf-TP_Top'], sliding=FINITE, thickness=ON)

mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE, clearanceRegion=omit, createStepName='Initial', datumAxis=None, initialClearance=OMIT, interactionProperty='IntProp-BP', master=mdb.models['Model-1'].rootAssembly.instances['Bolt-L-1'].surfaces['Surf-BL_Shank'], name='Int-BLST', slave=mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].surfaces['Surf-TP_Top'], sliding=FINITE, thickness=ON)
Appendix B. Developed Codes

['Surf-TP_SL'], sliding=FINITE, thickness=ON)
mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
clearanceRegion=None, createStepName='Initial', datumAxis=None,
initialClearance=OMIT, interactionProperty='IntProp-BP', master=
mdb.models['Model-1'].rootAssembly.instances['Bolt-L-1'].surfaces
['Surf-BL_Shank'], name='Int-BLSB', slave=
mdb.models['Model-1'].rootAssembly.instances['Plate_Bottom-1'].surfaces
['Surf-BP_SL'], sliding=FINITE, thickness=ON)
mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
clearanceRegion=None, createStepName='Initial', datumAxis=None,
initialClearance=OMIT, interactionProperty='IntProp-BP', master=
mdb.models['Model-1'].rootAssembly.instances['Bolt-L-1'].surfaces['Surf-BL_N'],
name='Int-BLN', slave=
mdb.models['Model-1'].rootAssembly.instances['Plate_Bottom-1'].surfaces
['Surf-BP_Bot'], sliding=FINITE, thickness=ON)
mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
clearanceRegion=None, createStepName='Initial', datumAxis=None,
initialClearance=OMIT, interactionProperty='IntProp-BP', master=
mdb.models['Model-1'].rootAssembly.instances['Bolt-R-1'].surfaces['Surf-BR_H'],
name='Int-BRH', slave=
mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].surfaces
['Surf-TP_Top'], sliding=FINITE, thickness=ON)
mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
clearanceRegion=None, createStepName='Initial', datumAxis=None,
initialClearance=OMIT, interactionProperty='IntProp-BP', master=
mdb.models['Model-1'].rootAssembly.instances['Bolt-R-1'].surfaces['Surf-BR_Shank'],
name='Int-BRST', slave=
mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].surfaces
['Surf-TP_SR'], sliding=FINITE, thickness=ON)
mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
clearanceRegion=None, createStepName='Initial', datumAxis=None,
initialClearance=OMIT, interactionProperty='IntProp-BP', master=
mdb.models['Model-1'].rootAssembly.instances['Bolt-R-1'].surfaces['Surf-BR_Shank'],
name='Int-BRSB', slave=
mdb.models['Model-1'].rootAssembly.instances['Plate_Bottom-1'].surfaces
['Surf-BP_SR'], sliding=FINITE, thickness=ON)
mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
clearanceRegion=None, createStepName='Initial', datumAxis=None,
initialClearance=OMIT, interactionProperty='IntProp-BP', master=
mdb.models['Model-1'].rootAssembly.instances['Bolt-R-1'].surfaces['Surf-BR_N'],
name='Int-BRN', slave=
mdb.models['Model-1'].rootAssembly.instances['Plate_Bottom-1'].surfaces
['Surf-BP_Bot'], sliding=FINITE, thickness=ON)
mdb.models['Model-1'].SurfaceToSurfaceContactStd(adjustMethod=NONE,
clearanceRegion=None, createStepName='Initial', datumAxis=None,
initialClearance=OMIT, interactionProperty='IntProp-BP', master=
mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].surfaces
['Surf-TP_Bot'], name='Int-PP', slave=
mdb.models['Model-1'].rootAssembly.instances['Plate_Bottom-1'].surfaces['Surf-BP_Top'], sliding=FINITE, thickness=ON)
mdb.models['Model-1'].ContactProperty('IntProp-PP')
Appendix B. Developed Codes

mdb.models['Model-1'].interactionProperties['IntProp-PP'].TangentialBehavior(
dependencies=0, directionality=ISOTROPIC, elasticSlipStiffness=None,
formulation=PENALTY, fraction=0.005, maximumElasticSlip=FRACTION,
pressureDependency=OFF, shearStressLimit=None, slipRateDependency=OFF,
table=((FrPP, ), ), temperatureDependency=OFF)

mdb.models['Model-1'].interactionProperties['IntProp-PP'].NormalBehavior(
allowSeparation=ON, clearanceAtZeroContactPressure=0.0,
constraintEnforcementMethod=PENALTY, contactStiffness=DEFAULT,
contactStiffnessScaleFactor=FrPP, pressureOverclosure=HARD,
stiffnessBehavior=LINEAR)

mdb.models['Model-1'].interactions['Int-PP'].setValues(adjustMethod=NONE,
bondingSet=None, contactTracking=TWO_CONFIG, enforcement=SURFACE_TO_SURFACE,
initialClearance=OMIT, interactionProperty='IntProp-PP', sliding=FINITE,
thickness=ON)

# ============== Boundary Conditions =============== #

mdb.models['Model-1'].YsymmBC(createStepName='Initial', localCsys=None, name='BC-Sym_BL', region=
mdb.models['Model-1'].rootAssembly.instances['Bolt-L-1'].sets['Set-SYMM'])

mdb.models['Model-1'].YsymmBC(createStepName='Initial', localCsys=None, name='BC-Sym_BR', region=
mdb.models['Model-1'].rootAssembly.instances['Bolt-R-1'].sets['Set-SYMM'])

mdb.models['Model-1'].YsymmBC(createStepName='Initial', localCsys=None, name='BC-Sym_TP', region=
mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].sets['Set-SymTP'])

mdb.models['Model-1'].YsymmBC(createStepName='Initial', localCsys=None, name='BC-Sym_BP', region=
mdb.models['Model-1'].rootAssembly.instances['Plate_Bottom-1'].sets['Set-SymBP'])

mdb.models['Model-1'].DisplacementBC(amplitude=UNSET, createStepName='Initial',
distributionType=UNIFORM, fieldName='', fixed=OFF, localCsys=None, name='BC-7',
region=
mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].sets['Set-BCTP']
, u1=SET, u2=SET, u3=SET, ur1=UNSET, ur2=UNSET, ur3=UNSET)

mdb.models['Model-1'].DisplacementBC(amplitude=UNSET, createStepName='Initial',
distributionType=UNIFORM, fieldName='', fixed=OFF, localCsys=None, name='BC-TP',
region=
mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].sets['Set-BCTP']
, u1=UNSET, u2=SET, u3=SET, ur1=UNSET, ur2=UNSET, ur3=UNSET)

# =============== Step Creation =============== #

mdb.models['Model-1'].StaticStep(adaptiveDampingRatio=0.05,
continueDampingFactors=False, initialInc=0.01, maxInc=0.1, maxNumInc=100000
, minInc=1e-10, name='Step-1', nlgeom=ON, previous='Initial',
stabilizationMagnitude=0.0002, stabilizationMethod=
DISSIPATED_ENERGY_FRACTION)

# =============== Load =============== #

mdb.models['Model-1'].DisplacementBC(amplitude=UNSET, createStepName='Step-1',
distributionType=UNIFORM, fieldName='', fixed=OFF, localCsys=None, name='BC-7',
region=
mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].sets['Set-BCTP']
# == MPC Constraint ==#

```
# ================ MPC Constraint ================== #
mdb.models['Model-1'].rootAssembly.DatumPointByMidPoint(point1=
    mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].InterestingPoint(
        mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].edges.findAt((
            100.0+2.0*Length-Overlap, 0.0, 1.5*Thickness), ), MIDDLE),
    point2=
    mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].InterestingPoint(
        mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].edges.findAt((
            100.0+2.0*Length-Overlap, Width, 1.5*Thickness), ), MIDDLE))

mdb.models['Model-1'].rootAssembly.ReferencePoint(point=
    mdb.models['Model-1'].rootAssembly.datums[10])

mdb.models['Model-1'].rootAssembly.Set(name='m_Set-1', referencePoints=(
    mdb.models['Model-1'].rootAssembly.referencePoints[11], ))

mdb.models['Model-1'].MultipointConstraint(controlPoint=
    mdb.models['Model-1'].rootAssembly.sets['m_Set-1'], csys=None, mpcType=
    BEAM_MPC, name='Constraint-MPC', surface=
    mdb.models['Model-1'].rootAssembly.instances['Plate_Top-1'].sets['Set-BCTP'],
    userMode=DOF_MODE_MPC, userType=0)

mdb.models['Model-1'].boundaryConditions['BC-7'].setValues(region=
    mdb.models['Model-1'].rootAssembly.sets['m_Set-1'])

mdb.models['Model-1'].boundaryConditions['BC-TP'].setValues(region=
    mdb.models['Model-1'].rootAssembly.sets['m_Set-1'])
```

# == Material Change ==#

```
# =============== Material Change ================= #
execfile(script_path+'\Material.py')

""
Externally changes the material
""

# == Job ==#

```
# =============== Job ============================== #

Ti = str(Thickness)
Ma = str(Mat)
Cl = str(Clear)
RB = str(R_B)
EC = str(Eccent)

Ti = str.replace(Ti,'.','-')
Ma = str.replace(Ma,'.','-')
Cl = str.replace(Cl,'.','-')
RB = str.replace(RB,'.','-')
EC = str.replace(EC,'.','-')

""
It allows to name the Files, it should be named in such a way that
identification is very easy
""

job_name = 'Double_'+'Ti'_+'Ma'_+'Cl'_+'RB'_+'EC'
```
Appendix B. Developed Codes

mdb.Job(atTime=None, contactPrint=OFF, description='', echoPrint=OFF, explicitPrecision=SINGLE, getMemoryFromAnalysis=True, historyPrint=OFF, memory=90, memoryUnits=PERCENTAGE, model='Model-1', modelPrint=OFF, multiprocessingMode=DEFAULT, name=job_name, nodalOutputPrecision=SINGLE, numCpus=5, numDomains=5, numGPUs=5, queue=None, resultsFormat=ODB, scratch='', type=ANALYSIS, userSubroutine='', waitHours=0, waitMinutes=0)

mdb.jobs[job_name].setValues(numCpus=5, numDomains=5)

# ============= Saving the Model ==================

mdb.saveAs(pathName=folder+'\Results\'+job_name+'.cae')

# ============= Runs Job ===================

if Analysis==1:
    mdb.jobs[job_name].submit(consistencyChecking=OFF)

Material.py:

    # Created by Fernando de Abreu Almeida 27-9-2018

from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *

# ============== Input ===================

import os
#script_path = os.getcwd()
execfile(script_path+'\Var.py')
# Gets the address of the file and executes the right Var.py
# They all have to be in the same folder to work

mdb.models['Model-1'].Material(name='Material-Bolt')
mdb.models['Model-1'].materials['Material-Bolt'].Elastic(table=((205000.0, 0.3), ))
mdb.models['Model-1'].materials['Material-Bolt'].Plastic(table=((930.0, 0.0), (952.0, 0.039)))

mdb.models['Model-1'].Material(name='Material-Plate')
mdb.models['Model-1'].materials['Material-Plate'].Elastic(table=((213000.0, 0.3), ))
mdb.models['Model-1'].materials['Material-Plate'].Plastic(table=((607.0, 0.0), 0.3), ))
Appendix B. Developed Codes

如果您需要，我可以帮助您详细阅读和理解这份文档。
Appendix B. Developed Codes

```
mdb.models['Model-1'].parts['Plate_Bottom'].setMeshControls(algorithm=MEDIAL_AXIS, regions=
    mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask((
        '#21 ', ), ))

mdb.models['Model-1'].parts['Plate_Bottom'].Set(cells=
    mdb.models['Model-1'].parts['Plate_Bottom'].cells.getSequenceFromMask((
        '#21 ', ), ), name='Set-Mesh')

mdb.models['Model-1'].parts['Plate_Bottom'].seedPart(deviationFactor=0.1, minSizeFactor=0.1, size=MeshPlate)

mdb.models['Model-1'].parts['Plate_Bottom'].seedEdgeBySize(constraint=FINER, deviationFactor=0.1, edges=
    mdb.models['Model-1'].parts['Plate_Bottom'].edges.getSequenceFromMask((
        '#6067c7 #633fe7 #80d80000 #1 ', ), ), minSizeFactor=0.1, size=MeshSize)

mdb.models['Model-1'].parts['Plate_Bottom'].generateMesh()

# ================ Plate Top =============================== #

tuple='[#5ff ', ), ), ))

mdb.models['Model-1'].parts['Plate_Top'].seedPart(deviationFactor=0.1, minSizeFactor=0.1, size=MeshPlate)

mdb.models['Model-1'].parts['Plate_Top'].setElementType(elemTypes=(ElemType(  
    elemCode=ElPlate, elemLibrary=STANDARD, secondOrderAccuracy=OFF,  
    kinematicSplit=AVERAGE_STRAIN, hourglassControl=DEFAULT,  
    distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),  
    ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#3 ', ), ), ))

mdb.models['Model-1'].parts['Plate_Top'].setElementType(elemTypes=(ElemType(  
    elemCode=E1Type, elemLibrary=STANDARD, secondOrderAccuracy=OFF,  
    distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),  
    ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#fff ', ), ), ))

mdb.models['Model-1'].parts['Plate_Top'].setMeshControls(algorithm=MEDIAL_AXIS, regions=
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#48 ', ), ))

mdb.models['Model-1'].parts['Plate_Top'].Set(cells=
    mdb.models['Model-1'].parts['Plate_Top'].cells.getSequenceFromMask((
        '#48 ', ), ), name='Set-Mesh_TP')

mdb.models['Model-1'].parts['Plate_Top'].seedEdgeBySize(constraint=FINER, deviationFactor=0.1, edges=
    mdb.models['Model-1'].parts['Plate_Top'].edges.getSequenceFromMask((
        '#e4c40000 #fe68181b #80d8061f #1 ', ), ), minSizeFactor=0.1, size=MeshSize)

mdb.models['Model-1'].parts['Plate_Top'].generateMesh()

# ================ Bolt L ==================== #

mdb.models['Model-1'].parts['Bolt-L'].setElementType(elemTypes=(ElemType(  
    elemCode= ElType, elemLibrary=STANDARD, secondOrderAccuracy=OFF,  
    distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),  
    ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
    mdb.models['Model-1'].parts['Bolt-L'].cells.getSequenceFromMask((
        '#7 ', ), ), ))
```
Appendix B. Developed Codes

```python
mdb.models['Model-1'].parts['Bolt-L'].seedPart(deviationFactor=0.1,
minSizeFactor=0.1, size=MeshSize)
mdb.models['Model-1'].parts['Bolt-L'].setMeshControls(algorithm=MEDIAL_AXIS,
regions=mdb.models['Model-1'].parts['Bolt-L'].cells.getSequenceFromMask((
'#[7 ]', ), ))
mdb.models['Model-1'].parts['Bolt-L'].generateMesh()

# ================= Bolt R ==================== #
mdb.models['Model-1'].parts['Bolt-R'].setElementType(elemTypes=(ElemType(
  elemCode= ElType, elemLibrary=STANDARD, secondOrderAccuracy=OFF,
distortionControl=DEFAULT), ElemType(elemCode=C3D6, elemLibrary=STANDARD),
ElemType(elemCode=C3D4, elemLibrary=STANDARD)), regions=(
  mdb.models['Model-1'].parts['Bolt-R'].cells.getSequenceFromMask(('[#7 ]', ), ), ))
mdb.models['Model-1'].parts['Bolt-R'].seedPart(deviationFactor=0.1,
minSizeFactor=0.1, size=MeshSize)
mdb.models['Model-1'].parts['Bolt-R'].setMeshControls(algorithm=MEDIAL_AXIS,
regions=mdb.models['Model-1'].parts['Bolt-R'].cells.getSequenceFromMask((
'#[7 ]', ), ))
mdb.models['Model-1'].parts['Bolt-R'].generateMesh()

GatherData.py:

```
# =============== Job =============== #

Ti = str(Thickness)
Ma = str(Mat)
Cl = str(Clear)
RB = str(R_B)
EC = str(Eccent)

Ti = str.replace(Ti,'.','-')
Ma = str.replace(Ma,'.','-')
Cl = str.replace(Cl,'.','-')
RB = str.replace(RB,'.','-')
EC = str.replace(EC,'.','-')

"""
It allows to name the Files, it should be named in such a way
that identification is very easy
"""

job_name = 'Double_'+Ti+'_'+Ma+'_'+Cl+'_'+RB+'_'+EC

from abaqus import *
from abaqusConstants import *
session.Viewport(name='Viewport: 1', origin=(0.0, 0.0),
width=354.719787597656,
height=209.902770996094)
session.viewports['Viewport: 1'].makeCurrent()
session.viewports['Viewport: 1'].maximize()

from caeModules import *
from driverUtils import executeOnCaeStartup
executeOnCaeStartup()

session.viewports['Viewport: 1'].partDisplay.geometryOptions.setValues(
referenceRepresentation=ON)
o1 = session.openOdb(
    name=folder+'\Results\'+job_name+'.odb')
session.viewports['Viewport: 1'].setValues(displayedObject=o1)

odb = session.odbs[folder+'\Results\'+job_name+'.odb']
session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
     NODAL, ((COMPONENT, 'RF1'), )), ('U', NODAL, ((COMPONENT, 'U1'), ))), ),
nodeSets=('M_SET-1', ))
x1 = session.xyDataObjects['RF:RF1 PI: ASSEMBLY N: 1']
x0 = session.xyDataObjects['U:U1 PI: ASSEMBLY N: 1']
session.writeXYReport(fileName='H:/Thesis/Analysis_Result/'+job_name+'.txt',
xyData=(x0, x1))

# One can use the same principle to take pictures and save them
# the same goes for graphs

os.chdir('H:/Thesis/Analysis_Result')
session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
     NODAL, ((COMPONENT, 'RF1'), )), ('U', NODAL, ((COMPONENT, 'U1'), ))), ),
nodeSets=('M_SET-1', ))
xy1 = session.xyDataObjects['U:U1 PI: ASSEMBLY N: 1']
Appendix B. Developed Codes

```python
xy2 = session.xyDataObjects['RF:RF1 PI: ASSEMBLY N: 1']
xy3 = combine(xy1, xy2)
xy3.setValues(
    sourceDescription='combine ( "U:U1 PI: ASSEMBLY N: 1","RF:RF1 PI: ASSEMBLY N: 1" )
)
tmpName = xy3.name
session.xyDataObjects.changeKey(tmpName, 'XYData-1')
xy1 = session.xyDataObjects['U:U1 PI: ASSEMBLY N: 1']
xy2 = session.xyDataObjects['RF:RF1 PI: ASSEMBLY N: 1']
xy3 = combine(xy1, xy2)
xy3 = session(xyPlot('XYPlot-1'))
chartName = xy3.charts.keys()[0]
chart = xy3.charts[chartName]
c1 = session.Curve(xyData=xy3)
chart.setValues(curvesToPlot=(c1, ), )
session.viewports['Viewport: 1'].setValues(displayedObject=xyp)
session.printToFile(fileName=job_name, format=PNG, canvasObjects=(
    session.viewports['Viewport: 1'], ))

Delete.py:

```
Appendix B. Developed Codes

DataExtraction.m:

%% Import data from text file.
% Script for importing data from the following text file:
% Auto-generated by MATLAB on 2018/10/09 14:11:59

% Use: select a directory, in this directory only Abaqus txt files must be
% present. UPDATE: extra files are allowed, avoid having other .txt files,
% besides the Abaqus Results.

% First Matlab will gather the names of every file, after it will
% gather all the relevante information (in this case Force X Displacement)
% and save it in an array for later usage.

% Matlab will create a folder with the date and time to store the .xlsx
% It also extracts all the data for an Excel Sheet.

% Matlab creates a general excel sheet and a particular one for each type
% of single bolt analysis

% Modified by Fernando de Abreu Almeida
% Latest Update: 14-10-2018

%% Cleaning all Variables
  clear all; close all; clc

% % % USE MATLAB 2017 or latest version % % %

%% Create a New Folder
  cd 'H:\';
  % Directory of the new Folder
  t = datestr(now);
  % Name of the New Folder, is the date and time
  t = strrep(t,' ','_');
  t = strrep(t,':','-');
  % Adjusts the name for some acceptable
  mkdir (t);
  % Creates the New Folder

%% Getting the Name of the Files
  directory = 'H:\Thesis\Analysis_Result\';
  % Where the files are located
  a=dir(directory);
  % Command Gathers the name of the files
  b=rmfield(a,'folder');
  % From b to d it organizes in the matrix structure needed
  b=rmfield(a,'date');
  % To extract the data from txt to matlab
  b=rmfield(b,'isdir');
  b=rmfield(b,'datenum');
  b=rmfield(b,'bytes');
c = struct2cell(b);

[m,n] = size(c);

index = 1;
% Counter inside the counter to avoid size problems
empty = 1;
% Counts the number of files in the folder that are not .txt
for i=1:n
% Statement that only saves .txt text to be later on extracted
  k(i) = strfind(c(1,i),'.txt');
  % Finds .txt in the variables, if there arent the cell will be empty
  g = cell2mat(k(i));
  if isempty(g)
    empty = empty+1;
  else
    d(1,index)=c(1,i);
    index = index+1;
  end
end

[m,n] = size(d);
% Initialize variables.
Index1 = 1;
% Index for the logical point of cells from A to Z
Index2 = 1;
% Internal Counters for the Excel extraction
Index3 = 1;
for i=1:n
% A loop for all of the files in the folder (again the folder should only contain the .txt files)
  file = [directory];
  % Directory where the files are located
  fjoint = strcat(file,d(1,i));
  % Combines the directory to the file
  filename = cell2mat(fjoint);
  % Transform the array into a cell for the other commands
delimiter = ', ';
startRow = 3;

  %% Format string for each line of text:
  % column2: double (%f)
  % column3: double (%f)
  % For more information, see the TEXTSCAN documentation.
  formatSpec = '%*s%f%f%s%s*\n\r';

  %% Open the text file.
  fileID = fopen(filename,'r');
%% Read columns of data according to format string.
%% This call is based on the structure of the file used to generate this
%% code. If an error occurs for a different file, try regenerating the code
%% from the Import Tool.
textscan(fileID, '%[^\n\r]', startRow-1, 'ReturnOnError', false);
dataArray = textscan(fileID, formatSpec, 'Delimiter', delimiter,
                   'MultipleDelimsAsOne', true, 'EmptyValue', NaN, 'ReturnOnError', false);

%% Close the text file.
fclose(fileID);

%% Create output variable
e = cell2mat(d(1,i));
name = erase(e,'.txt');
e = strrep(e,'_','S');
e = strrep(e,'-','d');
e = erase(e,'.txt');
% Organizes the array name into something acceptable to Matlab
data.(e) = [dataArray{1:end-1}];
% Creates an Array with the Disp and the Force
data.(e)(:,2)=data.(e)(:,2)*2;
% Multiplies the Force by 2, adjusting the symmetry
top.(e)=max(data.(e)(:,2));
f{i} = e;

Alphabet = ['A' 'B' 'C' 'D' 'E' 'F' 'G' 'H' 'I' 'J' 'K'
           'L' 'M' 'N' 'O' 'P' 'Q' 'R' 'S' 'T' 'U' 'V' 'W' 'X' 'Y' 'Z'];
[o,p] = size(data.(e));
% O is the variable of interest
excelsheet = strcat('H:\','t', '\All_Files.xlsx');
if Index1 < 26
  % Inserts Data of Interest for the Cells A to Z
  Alph = strcat(string(Alphabet(Index1)),'1');
  Bet = strcat(string(Alphabet(Index1)),'2:',string(Alphabet(Index1+1)),string(o+1));
  xlswrite(excelsheet ,{name},'Sheet1',Alph);
  xlswrite(excelsheet ,data.(e),'Sheet1',Bet);
else % Inserts Data of Interest to Cells AA...
  Indexx = ceil (Index3/26);
  Alph = strcat(string(Alphabet(Indexx)),string(Alphabet(Index2)),'1');
  % The Cell for the Name
  Bet = strcat(string(Alphabet(Indexx)),string(Alphabet(Index2)),'2:',
             string(Alphabet(Indexx)),string(Alphabet(Index2+1)),string(o+1));
  % The Cells for the Data
  xlswrite(excelsheet ,{name},'Sheet1',Alph);
  xlswrite(excelsheet ,data.(e),'Sheet1',Bet);
  Index2 = Index2+2;
  Index3 = Index3+2;
end

if Index2 > 26 % zero the Index2 so it will never go above 26
Index2 = 1;
end
Index1 = Index1+2;

%% Clear temporary variables
clearvars filename delimiter startRow formatSpec fileID dataArray ans;
end

%% Count for identifying the number of Single Bolt Analysis
cont = 1;
% Counters for the data Extraction
Index1 = 1;
% Index for the logical point of cells from A to Z
Index2 = 1;
Index3 = 1;
for i=1:n
% Statement that only saves the Single Bolts Codes
y(i) = strfind(f(i),'Single');
% Find Singles in the variables, if there arent the cell will be empty
z = cell2mat(y(i));
if isempty(z)
else
    Singles(cont)=f(i);
    cont = cont+1;
end
end

[mm,nn] = size(Singles);
cont = 1;
cont2 = 1;
for i=1:nn
% A loop for the number of "Single" Analysis in the Folder
Singles(i) = erase(Singles(i),'Single');
exsheet = strrep(Singles(i),'S','_');
exsheet = strrep(exsheet,'d','-');
exxlsxsheet = strcat('H:\',t,'\',exsheet ,'.xlsx');
% Name of the Excel Sheet for Single Bolt
xlsxsheet = cell2mat(xlsxsheet);
% Adjust the type of variable for the formula

Index1 = 1;
% Index for the logical point of cells from A to Z
Index2 = 1;
Index3 = 1;
cont2 = 1;
for j=1:n
% A loop for all the .txt files, it will read through them
% and collect the ones for the "Single Bolt"
yy = strfind(f(j),Singles(i));
zz = cell2mat(yy);
e = cell2mat(d(1,j));
name = erase(e,'.txt');
    if isempty(zz)
        else
            Seq(cont)=f(j);
            cont = cont+1;
            [o,p] = size(data.(cell2mat(f(j))));
            % 0 is the variable of interest

if Index1 < 26
    % Inserts Data of Interest for the Cells A to Z
    Alph = strcat(string(Alphabet(Index1)),'1');
    Bet = strcat(string(Alphabet(Index1)),'2:',string(Alphabet(Index1+1)));
    name2 = f(j);
    xlswrite(xlsxsheet ,{name},'Sheet1',Alph);
    xlswrite(xlsxsheet ,data.(cell2mat(f(j))),'Sheet1',Bet);
else
    % Inserts Data of Interest to Cells AA...
    Indexx = ceil (Index3/26);
    name2 = f(j);
    Alph = strcat(string(Alphabet(Indexx)),'1');
    % The Cell for the Name
    Bet = strcat(string(Alphabet(Indexx)),'2:',string(Alphabet(Indexx+1)),string(o+1));
    % The Cells for the Data
    xlswrite(xlsxsheet ,{name},'Sheet1',Alph);
    xlswrite(xlsxsheet ,data.(cell2mat(f(j))),'Sheet1',Bet);
    Index2 = Index2+2;
    Index3 = Index3+2;
end

if Index2 > 26
    % zero the Index2 so it will never go above 26
    Index2 = 1;
end
Index1 = Index1+2;

xlswrite(xlsxsheet ,{name},'Max',strcat('A',string(cont2)));
xlswrite(xlsxsheet ,top.(cell2mat(f(j))),'Max',strcat('B',string(cont2)));
cont2 = cont2+1;
end
end % End of the loop for all the elements
end % End of the loop for the "Single Bolt"

%% Opens Excel Sheet
% winopen('H:\Matlab_Ab.xlsx');
% Opens the Excel Sheet

%% Clear non-interesting variables
Appendix B. Developed Codes

clearvars a Alph Alphabet b Bet c cont cont2 d directory e empty excelsheet
clearvars exsheet f file fjoint g i index Index1 Index2 Index3 indexx j
clearvars Seq Singles t xlsxsheet y yy z zz k m mm n nn name o p