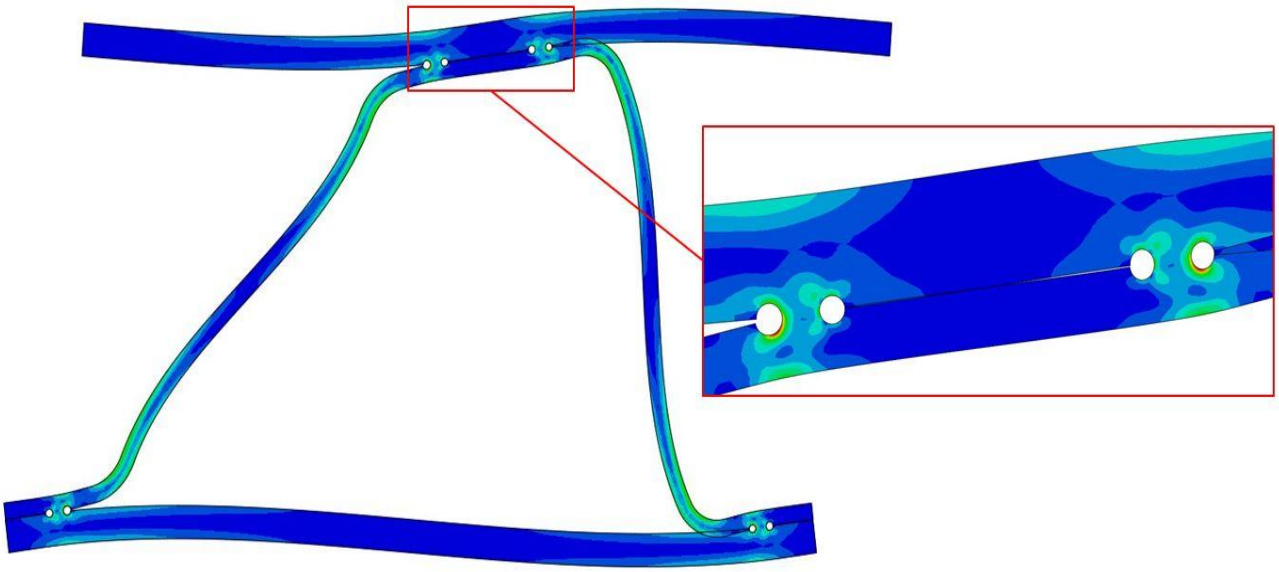
CHALMERS
UNIVERSITY OF TECHNOLOGY



# A parametric study of shear-induced fatigue in corrugated steel sandwich elements

Master's Thesis in the Master's Programme Structural Engineering and Building Technology

LOVISA PERSSON

Department of Civil and Environmental Engineering
*Division of Structural Engineering*
*Steel and Timber structures*
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2016
Master's Thesis BOMX02-16-55

MASTER'S THESIS BOMX02-16-55

# A parametric study of shear-induced fatigue in corrugated steel sandwich elements

*Master's Thesis in the Master's Programme Structural Engineering and Building Technology*

LOVISA PERSSON

Department of Civil and Environmental Engineering
*Division of Structural Engineering*
STEEL AND TIMBER STRUCTURES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2016

A parametric study of shear-induced fatigue in corrugated steel sandwich elements

*Master's Thesis in the Master's Programme Structural Engineering and Building Technology*

LOVISA PERSSON

© LOVISA PERSSON, 2016

Examensarbete BOMX02-16-55/ Institutionen för bygg- och miljöteknik,
Chalmers tekniska högskola 2016

Department of Civil and Environmental Engineering
Division of Structural Engineering
Steel and Timber structures
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: + 46 (0)31-772 1000

Cover:
Figure of the FE-model of a unit cell of the sandwich structure together with a close up of the welded area.

Chalmers Reproservice Göteborg, Sweden, 2016

A parametric study of shear-induced fatigue in corrugated steel sandwich elements

*Master's thesis in the Master's Programme Structural Engineering and Building Technology*

LOVISA PERSSON

Department of Civil and Environmental Engineering
Division of Structural Engineering
STEEL AND TIMBER STRUCTURES
Chalmers University of Technology

ABSTRACT

A probable replacer of the conventional steel bridge deck type that has shown promising features is the corrugated-core steel sandwich element. The steel-sandwich element is a modern type of high performing orthotropic steel deck consisting of two face sheets separated by a corrugated core.

The purpose of this thesis was to evaluate the influence of different geometrical properties on the fatigue life using the Effective Notch Stress (ENS) method. Furthermore, the influence of the geometrical properties on the transverse shear stiffness, in the direction perpendicular to the core, was also investigated.

First, a literature study on the structural behaviour of corrugated-core steel sandwich elements was performed. Thereafter, the parametric study was carried out by using script-based modelling of a unit sandwich cell based on the geometrical inputs and loaded in shear. Last, the output from the parametric study was post-processed to identify the most important parameters with respect to the fatigue life and the transverse shear stiffness.

The most important parameters are identified and their influences are discussed. The parameters with largest influence on the fatigue life are the thickness of the corrugation, $t_c$ , the length of the upper horizontal part of the core, $f_2$, the thickness of the upper face plate, $t_2$, and the distance between the welds, $d_w$. For the transverse shear stiffness, the angle of the corrugation, $\theta$ and the thickness of the corrugation, $t_c$ were found to have the largest influence. Moreover, a numerical approach for determining the effective notch stress under a unit shear force and the transverse shear stiffness are presented.

Key words: steel sandwich element, bridge deck, effective notch stress method, effective notch stress, transverse shear stiffness

En parameterstudie av skjuvinducerad utmattning i korrugerade stålsandwichelement

Examensarbete inom masterprogrammet Structural Engineering and Building Technology

LOVISA PERSSON

Institutionen för bygg- och miljöteknik
Avdelningen för Konstruktionsteknik
Stål- och träbyggnad
Chalmers tekniska högskola

## SAMMANFATTNING

En trolig ersättare till det konventionella brodäcket av stål som visat lovande egenskaper är det korrugerade sandwichelementet. Detta sandwichelement, helt i stål, är en modern typ av ett högpresterande ortotropt ståldäck som består av två styva ytplåtar separerade av en korrugerad kärna.

Syftet med detta examensarbete var att undersöka inflytandet av olika geometriska egenskaper på livslängen med avseende på utmattning enligt Effective Notch Stress (ENS) – metoden. Vidare, så undersöktes även inflytandet av de olika geometriska egenskaperna på skjuvstyvheten vinkelrätt mot korrugeringen.

Först utfördes en litteraturstudie av det strukturella beteendet för korrugerade sandwichelement. Därefter genomfördes parameterstudien genom att med scriptbaserad modellering skapa en enhetscell av sandwich elementet. Enhetscellen baserades på geometrisk indata och lastades i skjuvning. Sist, efterbehandlades resultaten från parameterstudien för att identifiera de viktigaste parametrarna med avseende på utmattning och skjuvstyvhet.

De viktigaste parametrarna identifieras och deras inflytande diskuteras. De parameterar med störst inflytande på utmattningslivslängden är tjockleken av korrugeringen, $t_c$, längden av den horisontella delen av korrugeringen, $f_2$, tjockleken av den övre ytplåten, $t_2$ och avståndet mellan svetsarna, $d_w$. För skjuvstyvheten så har korrugeringsvinkeln, $\theta$ och tjockleken av korrugeringen, $t_c$ störst inflytande. Dessutom presenteras en numerisk metod för att bestämma spänningen ”effective notch stress” under en enhetskjuvning samt den tvärgående skjuvstyvheten.

Nyckelord: stålsandwichelement, brodäck, effective notch stress-metoden, effective notch stress, tvärgående skjuvstyvhet

# Contents

# Preface

This master thesis evaluates the influence of cross-sectional geometrical properties on the fatigue life and the transverse shear stiffness of a corrugated-core sandwich element. The work was carried out during the spring 2016 at the Division of Structural Engineering, Department of Civil and Environmental Engineering at Chalmers University of Technology.

First of all, I would like to thank my supervisor Peter Nilsson for his guidance and support throughout the entire project. Gratitude is given to my examiner, Associate Professor Mohammad Al-Emrani, for his assistance and helpful feedback. I would also like to thank WSP Bridge and Hydraulic Design in Gothenburg for the possibility to carry out my work at their office. Special thanks to Daniel Josefsson at WSP for all the help I have received during the development of my Python-scripts. Finally, I would like to thank Scanscot Technology for the sponsorship of a license to BRIGADE/Plus 6.1 and for their fast and helpful support.

Gothenburg, June 2016
Lovisa Persson

# Notations

**Roman upper case letters**

| | |
|---|---|
| $A_c$ | Area, per unit width, of corrugation cross section perpendicular to corrugation axis |
| $D_x, D_y$ | Bending stiffness for corrugated-core sandwich plate, per unit width, in x- and y-direction respectively |
| $D_{xy}$ | Twisting stiffness for corrugated-core sandwich plate, per unit width |
| $D_{Qx}, D_{Qy}$ | Transverse shear stiffness for corrugated-core sandwich element, per unit width, in x- and y-direction respectively |
| $E_c$ | Modulus of elasticity of core material |
| $E_f$ | Modulus of elasticity of face sheet material |
| $E_x, E_y$ | Stiffness of plate, per unit width, in x- and y-direction respectively |
| $G_c$ | Shear modulus of elasticity of core material |
| $G_f$ | Shear modulus of elasticity of face sheet material |
| $G_{xy}$ | Shear stiffness of plate in xy-plane, per unit width |
| $H$ | Horizontal shear force acting on the element in the global system |
| $I$ | Moment of inertia of width 2p of cross section parallel to yz-plane, taken about centroidal axis parallel to y-axis |
| $I_c$ | Moment of inertia, per unit width, of corrugation cross-sectional area about middle plane |
| $M_x, M_y$ | Bending moment, per unit width, in x- and y-direction respectively |
| $M_{xy}$ | Twisting moment, per unit width |
| $N_x, N_y$ | Normal force, per unit width, in x- and y-direction |
| $N_{xy}$ | Shear force acting in x- and y-directions, per unit width |
| $Q_x, Q_y$ | Transverse shear force acting on cross-sections parallel to yz-plane and xz-plane respectively |
| $R$ | Radius of corrugation |
| $S$ | Non-dimensional coefficient depending upon shape of corrugation, relative proportions of sandwich cross section, and the material properties of the component parts |

**Roman lower case letters**

| | |
|---|---|
| $b$ | Width of sandwich plate in x-direction |
| $d_w$ | Distance between welds |
| $f_1$ | Length of lower horizontal part of the core |
| $f_2$ | Length of upper horizontal part of the core |
| $l$ | Length of one corrugation leg measured along the centre line |
| $p$ | Half of the corrugation pitch |
| $s$ | Coordinate measured along centre line of corrugation leg |
| $t_1$ | Thickness of lower face plate |
| $t_2$ | Thickness of upper face plate |
| $t_c$ | Thickness of corrugated-core sheet |

| | |
|---|---|
| $t_f$ | Thickness of each face sheet |
| $t_w$ | Thickness of weld |
| $h$ | Distance between middle surfaces of face sheets |
| $h_c$ | Depth of corrugation, measured vertically from centre line at crest to centre line at trough |
| $q$ | Intense of lateral loading |
| $s$ | Constraint factor |
| $v$ | Displacement in y-direction |
| $w$ | Deflection of middle-surface of plate, measured in z-direction |
| $x, y, z$ | Orthogonal coordinates |

**Greek lower case letters**

| | |
|---|---|
| $\gamma_{xz}, \gamma_{yz}$ | Shear-strain angles due to shears $Q_x$ and $Q_y$ respectively |
| $\eta_1$ | Distance from the edge of the lower horizontal part of the core to the weld |
| $\eta_2$ | Distance from the edge of the upper horizontal part of the core to the weld |
| $\nu_c$ | Poisson's ratio of core material |
| $\nu_f$ | Poisson's ratio of face sheet material |
| $\nu_x, \nu_y$ | Poisson's ratio associated with bending, x- and y-direction respectively |
| $\rho$ | Real notch radius |
| $\rho^*$ | Micro-support length |
| $\rho_f$ | Effective notch radius |
| $\theta$ | Angle of corrugation |
| $\varphi_x, \varphi_y$ | The slopes of the normal to the middle plane of the sandwich plate about yz- and xz-planes respectively |

**Abbreviations**

| | |
|---|---|
| ENS | Effective notch stress method |
| FE | Finite Element |
| GMAW | Gas Metal Arc Welding |
| HAZ | Heat Affected Zone |
| HLAW | Hybrid Laser-Arc Welding |
| LW | Laser Welding |
| S-N | Stress-Number |

# 1 Introduction

## 1.1 Background

During the last 4 decades the design development of steel bridge decks has been almost unchanged. The design most commonly used in bridge applications is a conventional orthotropic steel deck that consist of a top plate stiffened with longitudinal stiffeners that are either open or closed. The production of these decks demands a considerable amount of time and effort due to complex manual welding of details. This gives a high production cost and reduces the competitiveness against other deck designs. The complex joints also cause stress concentrations resulting in a reduced fatigue life. Therefore the application of the conventional orthotropic steel deck has been sparse and only used when minimal self-weight is essential like in long-span and movable bridges (Bright & Smith, 2007).

A probable replacer of the conventional deck type that has shown promising features is the corrugated steel sandwich element. The steel sandwich element is a light-weight high performing orthotropic steel deck consisting of two face plates separated by a corrugated core. The structure possess a high strength-to-weight ratio, it distributes loads in a more favourable manner and has an enhanced fatigue performance (Beneus & Koc, 2014).

The development of welding techniques has created possibilities for both a more efficient structural configuration and for increasing the automatization of the production. Techniques like laser welding and hybrid laser arc welding (HLAW) makes it possible to connect the core with the face plates from the outside. Even though the many benefits of the steel sandwich element it is not used in bridge applications at present. The element is not fully developed for this purpose and needs further research. (Palmkvist & Sandberg, 2015).

An area of research for the sandwich element is the transverse shear stiffness. Since the sandwich element has a light-weight and rather flexible core the shear stiffness of a sandwich element will always be less than of a homogenous member with the same bending stiffness. As a result of the flexible core, the sandwich element usually experiences substantial deflection due to shear. Furthermore, this deflection influences the magnitude of the stress close to the weld connecting the core to the face plates. One question that arises is; what influence on the stress close to the weld has the geometrical properties of the corrugated sandwich element?

## 1.2 Purpose and Aim

The purpose of this thesis was to evaluate the influence of the cross-sectional geometrical properties on the fatigue life of a corrugated core sandwich element using the Effective Notch Stress (ENS) method. Furthermore, the influence of the geometrical properties on the transverse shear stiffness, in the direction perpendicular to the core, was also investigated.

The aim of this thesis was to identify the most important parameters influencing the stress, close to the weld, and the transverse shear stiffness. The aim was also to present a model based on a numerical approach for determining the effective notch stress under a unit shear force and the transverse shear stiffness. The geometrical properties considered in the parametric study can be seen in Figure 1.1.

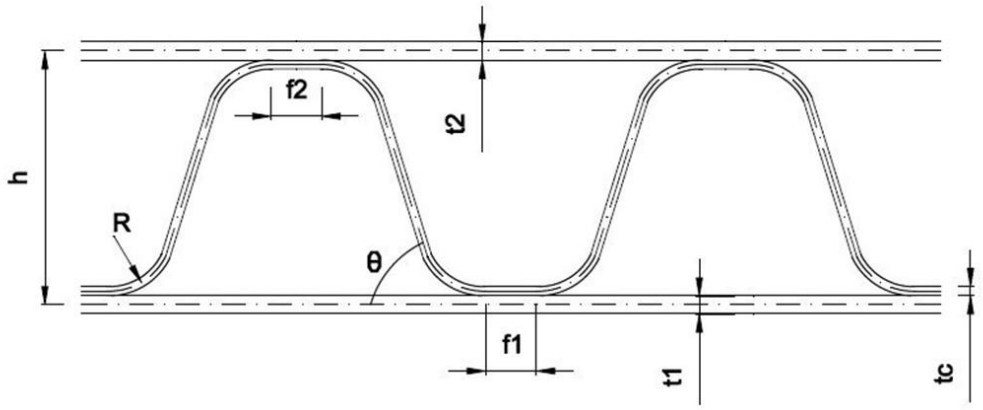



*Figure 1.1* *Cross-section of corrugated-core sandwich element including the geometrical properties. The thickness of the welds $t_w$ and the distance between the welds $d_w$ are not visualized in this figure but are also considered in the parametric study (Nilsson, 2015).*

## 1.3 Method

A literature study was performed that include a brief introduction to sandwich structures and the structural behaviour of corrugated-core steel sandwich plates. The literature study also include the theory of the effective notch method as well as a brief description of factorial design and regression analysis used for post-processing of the parametric study output.

The parametric study was carried out by creating a Python-script to automatically model a unit cell of the sandwich structure in BRIGADE/Plus 6.1. The analysis performed with the Python-script was a comparison between one and two welds, a fractional factorial design and a regression analysis based on a multi-level full factorial design matrix. All the analyses were based on the same Python-script with small changes to fit the required analysis.

Furthermore, the output from the parametric study was post-processed using Mathcad or Matlab to identify the most important parameters with respect to the fatigue life and the transverse shear stiffness. Mathcad was used in the factorial design to calculate the main effects and to visualize these in plots. The Matlab-script regstats.m was used to perform the regression analysis.

## 1.4 Limitations

The joining of the core to the face sheets for corrugated-core steel sandwich elements can be performed with a various number of welds. However, in this study only joining with one or two welds was considered. The contact between the face plate and core is excluded in this study. The reason for this conservative assumption is that a gap in the weld itself and/or the deformation of the plates from the welding easily eliminate the contact.

The evaluation of the geometrical parameters was limited to fatigue life and transverse shear stiffness in the direction perpendicular to the core. The fatigue assessment is performed using the ENS-method. Only stresses in the top-notches are considered within this thesis. However, the result can also be used for the bottom as the effects are mirrored there.

The shear deformation in the direction perpendicular of the core direction has been shown to influence the fatigue life to a high extent, and will be the only load effect considered within this thesis.

## 1.5 Outline

In chapter 2, the theory and methods used within this thesis are presented. It covers the basics about sandwich structures and the structural behaviour of corrugated-core steel sandwich plates. The effective notch method and its use in fatigue assessment are discussed as well as factorial design and regression analysis.

In chapter 3, the Python-scripts created for the parametric study are described. Also the modelling choices made for the FE-model of one-unit cell are presented.

Chapter 4 covers the results obtained from the three different analyses made; the comparison between one weld and two welds, the fractional factorial design and the full factorial design.

In chapter 5, the most important parameters influencing the stress close to the weld and the transverse shear stiffness are discussed.

In chapter 6, the conclusions regarding the purpose and aim of the thesis are presented.

Chapter 7 contains the references used within this thesis. Appendices containing supplementary information are attached in the end.

# 2 Literature study

The literature study presented in this chapter includes information, theories and methods used to obtain the result of this thesis. Initially a brief introduction to sandwich structures and steel sandwich plates in particular is presented. Then sandwich plate theory is discussed where the expressions of the elastic constants used to describe the fundamental properties of a corrugated-core sandwich plate is presented. Furthermore, information about the effective notch stress method and its application in fatigue assessment are provided. Last, brief descriptions of free body cut, factorial design and regression analysis are presented.

## 2.1 Sandwich structures

A sandwich structure is a structure that consists of two stiff surface plates separated by a core material with a lower stiffness (Säynajäkangas & Taulavouri, 2004). By separation of the two surface plates the flexural stiffness of the cross-section is noticeably increased with only a minor increase of the weight (SANDCORE, u.d.). This gives the sandwich structure the possibility to obtain a high strength-to-weight ratio (Säynajäkangas & Taulavouri, 2004). The basic concept for a sandwich structure is that the surface plates will resist load from bending while the core resists load from shear.

## 2.2 Steel sandwich structures

For steel sandwich structures the top plate, the bottom plate and the core are made of steel (Säynajäkangas & Taulavouri, 2004). The development of steel sandwich structures has been ongoing in different industrial branches since the 1950's. However, the welding techniques at that time created difficulties for the production of these structures. Laser welding enables the possibility to connect the internal core with the face sheets from the outside. It yields a continuous connection between the face plates, produced in an industrialized manner.

The development continued and in 1980's the United States Navy was the leading actor. The first strength test was made and some of the first applications were performed. Since then the development has continued in Europe, especially for marine applications, through research projects that joined forces between European main actors (Kujala & Klanac, 2005).

The core of a steel sandwich plate can have various configurations giving the plate different properties, see Figure 2.1. Lok and Cheng (2000) compared stiffness constants between sandwich plates with the same weight but different core configurations. The core configurations analysed were the web-core, the Z-core and the corrugated core. The results of their comparison shows that the web-core has the highest bending stiffness $D_x$ and $D_y$, twisting stiffness $D_{xy}$ and transverse shear stiffness in longitudinal direction to the core, $D_{Qx}$. However, the transverse shear stiffness perpendicular to the direction of the core $D_{Qy}$ is low for the web-core plate. The Z-core has behaviour similar to the web-core but with consistently lower stiffness. For the corrugated-core the stiffness's $D_x$, $D_y$, $D_{xy}$ and $D_{Qx}$ are a bit lower than for the web-core and are decreasing with decreasing

internal angle of the corrugation, see Figure 1.1. Nevertheless, the corrugated core has a substantially higher $D_{Qy}$, which instead increases with decreasing internal angle of the corrugation (Lok & Cheng, 2000). The increase of $D_{Qy}$ increases the overall stiffness of the plate as it reduces the level of orthotropy.

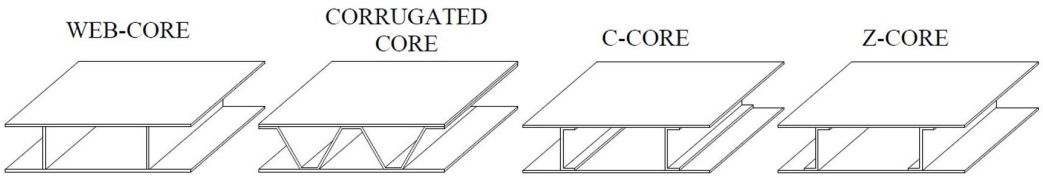



*Figure 2.1 Some different core configurations (Romanoff, 2007).*

Alwan and Järve (2012) concluded that the configuration most promising for bridge applications is the corrugated-core. The cross-section of the corrugated-core configuration can be seen in Figure 2.2.

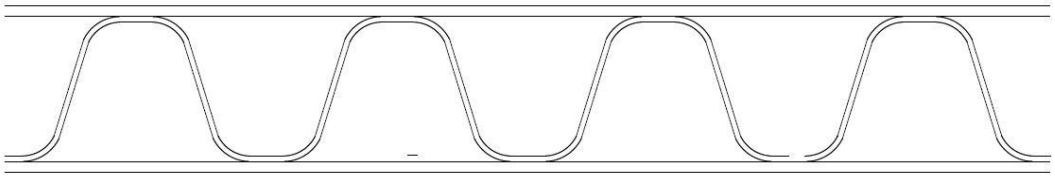


*Figure 2.2 Cross-section of corrugated-core configuration (Nilsson, 2015).*

### 2.2.1 Production of steel sandwich plates

The production methods of steel sandwich structures presented under this chapter will primary focus on the joining of the face-sheets to the core. The joining of the face-sheets to the core can be done by three different joining techniques, thermal joining (welding), adhesive bonding and mechanical joining. Thermal joining is only applicable for all-metal sandwich structures while the other two techniques has the benefit that they are also applicable for hybrid metal or composite sandwich structures (SANDCORE, u.d.).

The welding methods most suitable for joining the face plates and the core to a steel sandwich element are laser welding (LW) and hybrid laser-arc welding (HLAW). Laser welding joins the face plates to the core through the use of a laser beam and is a suitable method for small plate thicknesses. However, larger thicknesses may demand the use of HLAW, which is an automated laser technology that combines advantageous properties from laser welding and gas metal arc welding (GMAW). Advantages like deep weld penetration and small heat affected zones (HAZ) from laser welding combined with advantages like wide tolerance of joint gaps and a high control of flaws from GMAW (Abbot, et al., 2008). The execution of HLAW can be seen in Figure 2.3.

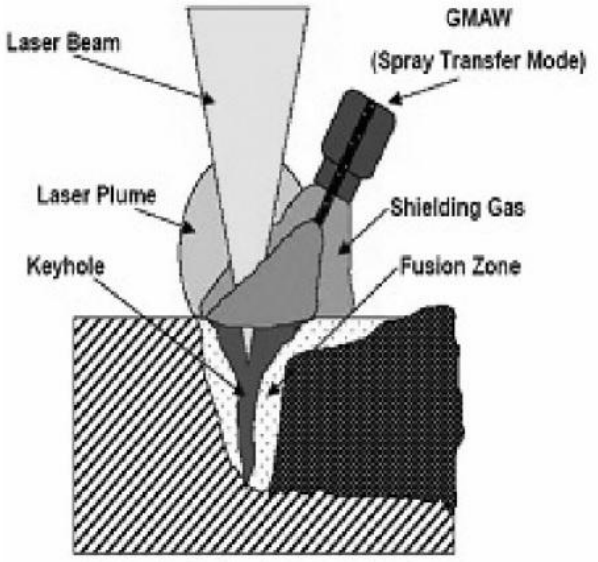



*Figure 2.3 Hybrid Laser Arc Welding (Abbot, et al., 2008).*

The many advantages of HLAW and the availability on the market should be reflected by a broadly spread practise of this method, however this is not the situation today. The investment cost and the conservatism when changing from an established method to a new can be two contributing causes (Beneus & Koc, 2014).

## 2.3 Plate theory

Plate theory is an engineering approximation that takes advantage of the plate's small thickness in relation to planar dimensions to reduce the original three-dimensional problem to a simpler two-dimensional problem (Ottosen & Petersson, 1992). The most common plate theories are called Kirchhoff theory for thin plates and Mindlin-Reissner theory for thick plates.

In the Kirchhoff theory it is assumed that plane sections normal to the mid-plane remain plane and normal to the mid-plane during deformation (Ottosen & Petersson, 1992). This assumption means that the transverse shear deformations do not contribute to the out-of-plane displacement. Since the sandwich panel has a low-stiffness core it will in general experience deflection due to shear and therefore the Kirchhoff plate theory cannot be used for this structure (Libove & Batdorf, 1948).

In the Mindlin-Reissner theory it is assumed that plane sections normal to the mid-plane remain plane but not necessarily normal to the mid-plane during deformation (Bhaskar & Varadan, 2013). This assumption accounts for transversal shear deformations and is therefore more suitable for the sandwich plate.

## 2.4 Sandwich plate theory

Libove and Batdorf (1948) presented a small deflection theory for sandwich plates that is based on the Mindlin-Reissner assumptions but extended to account for the orthotropic behaviour of a sandwich plate. The fundamental out-of-plane behaviour of the sandwich plate is described by seven elastic constants: the flexural stiffness's $D_x$ and $D_y$, the twisting stiffness $D_{xy}$, the transverse stiffness's $D_{Qx}$ and $D_{Qy}$, and the Poissons ratios $\nu_x$ and $\nu_y$. The elastic constants idealize the 3D corrugated-core sandwich element into a 2D homogenous orthotropic thick plate (Libove & Batdorf, 1948). An illustration of the idealization can be seen in Figure 2.4.

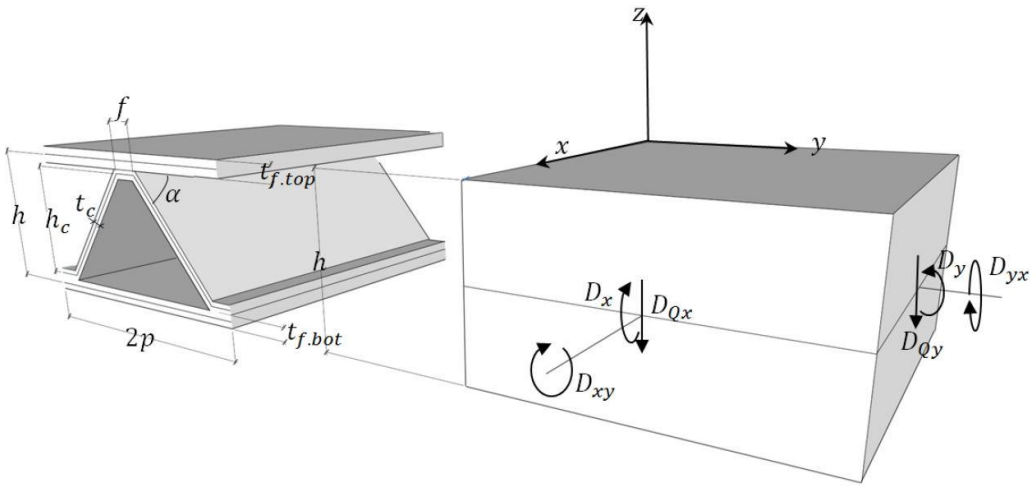



*Figure 2.4 3D corrugated-core sandwich element (left) and idealized 2D homogenous orthotropic thick plate with equivalent elastic constants (right) (Dackman & Ek, 2015).*

The constitutive equations, equilibrium equations and boundary conditions for an orthotropic sandwich plate expressed by Libove and Blatdorf (1948) are presented below. Furthermore, the closed form solution for a corrugated-core sandwich plate in bending, see e.g. Chang et al. (2004), is also presented below.

### 2.4.1 Constitutive equations

The plate equations are derived by assuming that only one load is acting on the plate at one time. The forces and moments acting on the element dxdy can be seen in Figure 2.5. For instance, the effect from letting only the bending moment $M_x$ act on two opposite faces is a primary curvature $\partial^2 w/\partial x^2$ in the middle surface together with a secondary curvature $\partial^2 w/\partial y^2$ which is the Poisson effect. In a similar way this is done for the bending moment $M_y$ and twisting moment $M_{xy}$ which gives:

$$M_x = -D_x \frac{\partial^2 w}{\partial x^2}, \text{ and } \frac{\partial^2 w}{\partial y^2} = -\nu_x \frac{\partial^2 w}{\partial x^2} \tag{2.1a}$$

$$M_x = -D_y \frac{\partial^2 w}{\partial y^2}, \text{ and } \frac{\partial^2 w}{\partial x^2} = -\nu_y \frac{\partial^2 w}{\partial y^2} \tag{2.1b}$$

$$M_{xy} = -D_{xy} \frac{\partial^2 w}{\partial x \partial y} \tag{2.1c}$$

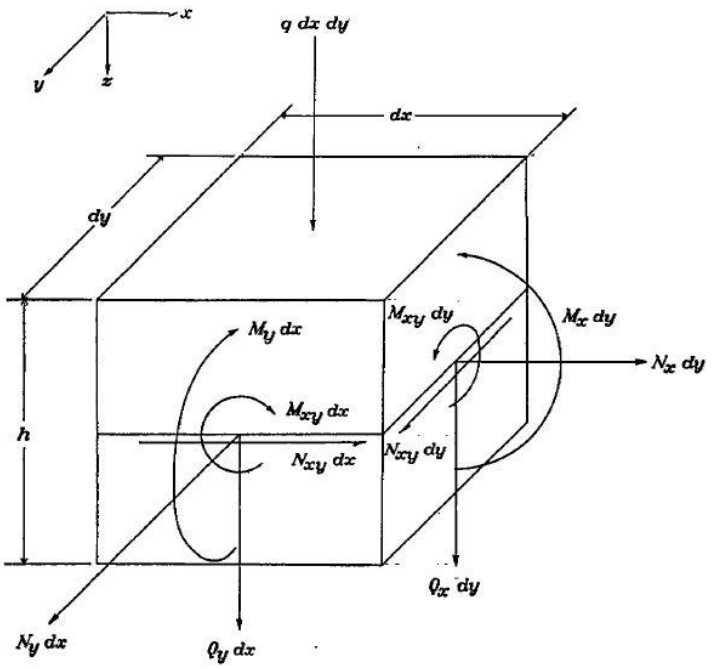



*Figure 2.5 Forces and moments acting on the element dxdy (Libove & Batdorf, 1948).*

The transverse shear responses are obtained when only the transverse shear forces $Q_x$ and $Q_y$ are acting on the plate.

$$\frac{\partial Q_x}{\partial x} = D_{Qx} \frac{\partial^2 w}{\partial x^2} \tag{2.2a}$$

$$\frac{\partial Q_y}{\partial y} = D_{Qy} \frac{\partial^2 w}{\partial y^2} \tag{2.2b}$$

The total curvature is obtained by superpositioning.

$$\frac{\partial^2 w}{\partial x^2} = -\frac{M_x}{D_x} + \nu_y \frac{M_y}{D_y} + \frac{1}{D_{Qx}} \frac{\partial Q_x}{\partial x} \tag{2.3a}$$

$$\frac{\partial^2 w}{\partial y^2} = -\frac{M_y}{D_y} + \nu_x \frac{M_x}{D_x} + \frac{1}{D_{Qy}} \frac{\partial Q_y}{\partial y} \tag{2.3b}$$

To obtain the third curvature a geometrical study is made.

$$\frac{\partial^2 w}{\partial x \partial y} = -\frac{M_{xy}}{D_{xy}} + \frac{1}{2D_{Qx}}\frac{\partial Q_x}{\partial x} + \frac{1}{2D_{Qy}}\frac{\partial Q_y}{\partial y} \tag{2.4}$$

Rearranging the equations (2.3) and (2.4) and inserting the introduced curvatures gives:

$$M_x = \frac{D_x}{1-\nu_x\nu_y}\left(\frac{\partial}{\partial x}\left(\frac{\partial w}{\partial x} - \frac{Q_x}{D_{Qx}}\right) + \nu_y \frac{\partial}{\partial y}\left(\frac{\partial w}{\partial y} - \frac{Q_y}{D_{Qy}}\right)\right) \tag{2.5a}$$

$$M_y = \frac{D_y}{1-\nu_x\nu_y}\left(\frac{\partial}{\partial y}\left(\frac{\partial w}{\partial y} - \frac{Q_y}{D_{Qy}}\right) + \nu_x \frac{\partial}{\partial x}\left(\frac{\partial w}{\partial x} - \frac{Q_x}{D_{Qx}}\right)\right) \tag{2.5b}$$

$$M_{xy} = \frac{D_{xy}}{2}\left(\frac{\partial}{\partial x}\left(\frac{\partial w}{\partial y} - \frac{Q_y}{D_{Qy}}\right) + \frac{\partial}{\partial y}\left(\frac{\partial w}{\partial x} - \frac{Q_x}{D_{Qx}}\right)\right) \tag{2.5c}$$

Integration over the thickness using equation (2.2) and inserting the shear angles gives the shear forces $Q_x$ and $Q_y$ per unit length.

$$Q_x = D_{Qx}\gamma_{xz} \tag{2.6a}$$

$$Q_y = D_{Qy}\gamma_{yz} \tag{2.6b}$$

### 2.4.2 Equilibrium equations

The equilibrium equations for forces in x-, y- and z-direction and for moments around x-, and y-axes for the element dxdy are gives as:

$$\frac{\partial N_x}{\partial x} + \frac{\partial N_{xy}}{\partial y} = 0 \tag{2.7a}$$

$$\frac{\partial N_y}{\partial y} + \frac{\partial N_{xy}}{\partial x} = 0 \tag{2.7b}$$

$$\frac{\partial Q_x}{\partial x} + \frac{\partial Q_y}{\partial y} = -\left(q + N_x\frac{\partial^2 w}{\partial x^2} + N_y\frac{\partial^2 w}{\partial y^2} + 2N_{xy}\frac{\partial^2 w}{\partial x \partial y}\right) \tag{2.7c}$$

$$\frac{\partial M_x}{\partial x} + \frac{\partial M_{xy}}{\partial y} - Q_x = 0 \tag{2.7d}$$

$$\frac{\partial M_{xy}}{\partial x} + \frac{\partial M_y}{\partial y} - Q_y = 0 \tag{2.7e}$$

In a small-deflection theory the normal forces N are assumed to be constant through the plate and to be unchanged during deflection of the plate. According to this equation (2.7a) and (2.7b) are automatically fulfilled.

### 2.4.3 Boundary conditions

Three boundary conditions should be prescribed at any point of the sandwich plate boundary. For a simply supported edge, two boundary conditions prescribe the bending moment in x-direction to zero and the displacement in z-direction to zero.

$$\begin{aligned} w &= 0 \\ M_x &= 0 \end{aligned} \tag{2.8}$$

The third boundary condition depends on the two different types of simple support that might emerge. For a simple support in which all points in the boundary are prevented from moving parallel to the edge the hard boundary condition is:

$$\gamma_{xz} = 0 \tag{2.9}$$

For a simple support in which all points in the boundary except those in the middle surface are free to move parallel to the edge the soft boundary condition is:

$$M_{xy} = 0 \tag{2.10}$$

Considering the two types of simple support, the first, equation (2.9), is more likely to occur in practise. The definition of the boundary conditions can be seen in Figure 2.6.

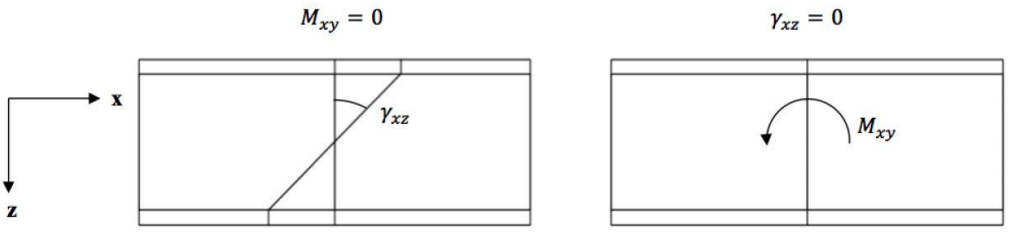



*Figure 2.6 Definition of soft and hard boundary conditions (Dackman & Ek, 2015).*

### 2.4.4 Closed form solution

A closed-form solution for a corrugated-core sandwich plate in bending is presented below. The solution is based on the Mindlin-Reissner plate theory and can be found in various literatures, see e.g. Chang (2008) or Zenkert (1995). The equations below correspond to a plate with simply supported edges.

#### 2.4.4.1 Governing equations

Governing equations for a corrugated-core sandwich element in bending are given as:

$$D_{xx}\frac{\partial^2\varphi_x}{\partial x^2}+\frac{D_{xy}}{2}\frac{\partial^2\varphi_x}{\partial y^2}-D_{Qx}+\left(\frac{D_{xy}}{2}+D_{xx}\nu_y\right)\frac{\partial^2\varphi_y}{\partial x\partial y}-D_{Qx}\frac{\partial w}{\partial x}=0 \qquad (2.11a)$$

$$\left(\frac{D_{xy}}{2}+D_{xx}\nu_y\right)\frac{\partial^2\varphi_x}{\partial x\partial y}+D_{yy}\frac{\partial^2\varphi_y}{\partial y^2}+\frac{D_{xy}}{2}\frac{\partial^2\varphi_y}{\partial x^2}-D_{Qy}-D_{Qy}\frac{\partial w}{\partial x}=0 \qquad (2.11b)$$

$$D_{Qx}\frac{\partial\varphi_x}{\partial x}+D_{Qy}\frac{\partial\varphi_y}{\partial x}+D_{Qx}\frac{\partial^2 w}{\partial x^2}+D_{Qy}\frac{\partial^2 w}{\partial y^2}-q=0 \qquad (2.11c)$$

For a rectangular plate, with length a and width b, with all edges simply-supported the solution to the governing equations (2.11) can be solved in the form of double Fourier series:

$$w=\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}w_{mn}\sin\left(\frac{m\pi x}{a}\right)\sin\left(\frac{n\pi y}{b}\right) \qquad (2.12a)$$

$$\varphi_x=\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}A_{mn}\cos\left(\frac{m\pi x}{a}\right)\sin\left(\frac{n\pi y}{b}\right) \qquad (2.12b)$$

$$\varphi_y=\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}B_{mn}\sin\left(\frac{m\pi x}{a}\right)\cos\left(\frac{n\pi y}{b}\right) \qquad (2.12c)$$

$$q=\sum_{m=1}^{\infty}\sum_{n=1}^{\infty}q_{mn}\sin\left(\frac{m\pi x}{a}\right)\sin\left(\frac{n\pi y}{b}\right) \qquad (2.12d)$$

Where $w_{mn}$, $A_{mn}$, $B_{mn}$ and $q_{mn}$ are coefficients to be determined. Inserting equations (2.12) into equations (2.11) these unknown coefficients can be solved.

## 2.5 Elastic constants for sandwich plates

Elastic constants for sandwich plates have been derived for different core configurations. Libove and Hubka (1951) derived the elastic constants for corrugated-cores based on straight beam theory. Nordstrand et al. (1994) derived two elastic shear modulus based on curved beam theory for corrugated-cores with various shape. Moreover, Fung et al. (1994) derived the transverse shear stiffness perpendicular to the core for Z-core sandwich panels. In 1996 the same writers also derived the transverse shear stiffness perpendicular to the core for C-core sandwich panels. The elastic constants derived by Libove and Hubka and Nordstrand et al. are presented further below.

### 2.5.1 Elastic constants according to Libove and Hubka

Libove and Hubka (1951) derived elastic constants for a corrugated-core sandwich element. These constants describe the deformations related to simple

loading conditions. They include two bending stiffness's $D_x$ and $D_y$, two transverse shear stiffness's $D_{Qx}$ and $D_{Qy}$, a twisting stiffness $D_{xy}$, the horizontal shear stiffness $G_{xy}$, two extensional stiffness's $E_x$ and $E_y$, and two Poisson's ratios $\nu_x$ and $\nu_y$.

For the derivation of the elastic constants the corrugation is assumed to be connected to the face plates by a single rigid joint at each crest and trough. The deformation figure and the connection positions for the shear loaded corrugated-core sandwich element can be seen in Figure 2.7.

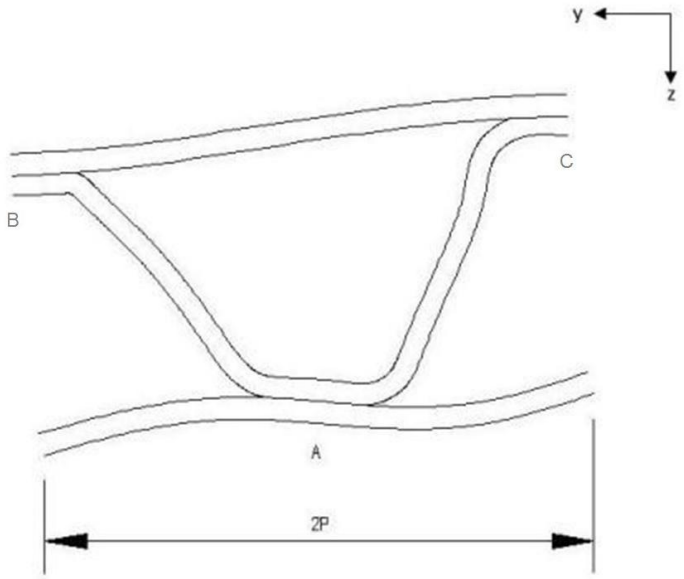



*Figure 2.7 Deformation figure for shear loaded corrugated-core sandwich element used in Libove and Hubkas derivation of elastic constants. The crest and trough is connected with a single rigid joint in point A, B and C.*

#### 2.5.1.1 Bending stiffness's

The bending stiffness's $D_x$ and $D_y$ for a corrugated-core sandwich element are given in equations (2.13) and (2.14).

$$D_x = E_c I_c + \frac{1}{2} E_f t_f h^2 \tag{2.13}$$

$$D_y = \frac{\frac{1}{2} E_f t_f h^2}{1 - \nu_f^2 \left( \frac{\frac{1}{2} E_f t_f h^2}{E_c I_c + \frac{1}{2} E_f t_f h^2} \right)} \tag{2.14}$$

Where:

$\nu_f$ Poisson's ratio of face plate material [-]

$E_f$ Modulus of elasticity of face plate material [MPa]

$E_c$ Modulus of elasticity of core material [MPa]

$I_c$ Moment of inertia, per unit width, of corrugation cross-sectional area about middle plane [m$^3$]

$t_f$ Thickness of each face plate [m]

$h$ Distance between middle surfaces of face plates [m]

#### 2.5.1.2 Poisson's ratios

The Poisson's ratios associated with bending $\nu_x$ and $\nu_y$ for a corrugated sandwich element are given in equations (2.15) and (2.16).

$$\nu_x = \nu_f \tag{2.15}$$

$$\nu_y = \nu_f \frac{D_y}{D_x} \tag{2.16}$$

#### 2.5.1.3 Extensional stiffness's

The extensional stiffness's $E_x$ and $E_y$ for a corrugated-core sandwich element are given in equations (2.17) and (2.18).

$$E_x = E_c A_c + 2E_f t_f \tag{2.17}$$

$$E_y = \frac{2E_f t_f}{1-\nu_f^2\left(1-\frac{2E_f t_f}{E_c A_c + 2E_f t_f}\right)} \tag{2.18}$$

Where:

$A_c$ Area, per unit width, of corrugation cross section perpendicular to corrugation axis [m]

#### 2.5.1.4 Twisting stiffness

The twisting stiffness $D_{xy}$ for a corrugated-core sandwich element is given in equation (2.19).

$$D_{xy} = 2\left(\frac{1}{2}G_f t_f h^2\right) \tag{2.19}$$

Where:

$G_f$ Shear modulus of elasticity of face plate material [MPa]

#### 2.5.1.5 Horizontal shear stiffness

The horizontal shear stiffness $G_{xy}$ for a corrugated-core sandwich element is given in equation (2.20).

$$G_{xy} = \frac{G_c t_c^2}{A_c} + G_f t_f \tag{2.20}$$

Where:

| | |
|---|---|
| $G_c$ | Shear modulus of elasticity of core material [MPa] |
| $t_c$ | Thickness of corrugated-core plate [m] |

#### 2.5.1.6 Transverse shear stiffnesses

The transverse shear stiffness's $D_{Qx}$ and $D_{Qy}$ for a corrugated-core sandwich element are given in equations (2.21) and (2.23).

$$D_{Qx} = \frac{G_c I t_c h}{p \int_0^l Q ds} \tag{2.21}$$

An approximate formula can be obtained for $D_{Qx}$ if the bending moment $M_x$ is assumed to be resisted only by the face-sheets. This assumption leads to constant shear flow in the corrugation. The approximation is given in equation (2.33).

$$D_{Qx} \approx \frac{G_c t_c^2}{A_c} \left( \frac{h}{p} \right)^2 \tag{2.22}$$

$$D_{Qy} = Sh \left( \frac{E_c}{1 - \nu_c^2} \right) \left( \frac{t_c}{h_c} \right)^2 \tag{2.23}$$

Where:

| | |
|---|---|
| $I$ | Moment of inertia of width 2p of cross section parallel to yz-plane, taken about centroidal axis parallel to y-axis [m] |
| $p$ | Half of the corrugation pitch [m] |
| $l$ | Length of one corrugation leg measured along the centre line [m] |
| $s$ | Coordinate measured along centre line of corrugation leg [m] |
| $h_c$ | Depth of corrugation, measured vertically from centre line at crest to centre line at trough [m] |
| $\nu_c$ | Poisson's ratio of core material [-] |
| $S$ | Non-dimensional coefficient depending upon shape of corrugation, relative proportions of sandwich cross section, and the material properties of the component parts [-] |

### 2.5.2 Transverse shear stiffness according to Nordstrand et al.

Nordstrand et al. (1994) derived two transverse shear stiffness's $D_{Qx}$ and $D_{Qy}$, using curved beam theory and based on the deformation of the core, for a general core shape.

For the derivations of the transverse shear stiffness's some assumptions was made. Only sandwiches with identical facings are considered and they are assumed to have a symmetrical pitch. The core was also assumed to be connected to the face plates by a single rigid joint at each crest and trough. Considering shear deformations the element with unit width is supported at two points, at the edges of the bottom face. The deformation figure and support position for the shear loaded corrugated-core sandwich element can be seen in Figure 2.8.

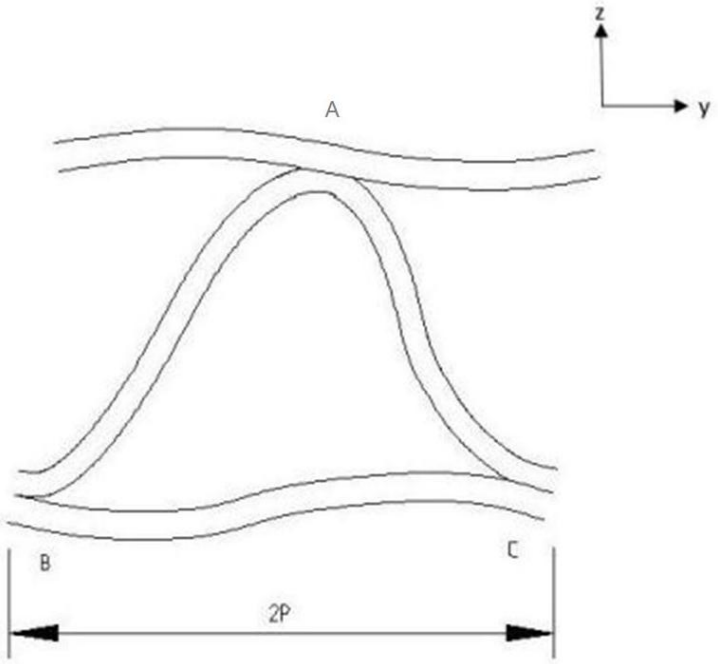



*Figure 2.8 Deformation figure for shear loaded corrugated-core sandwich element used in Nordstrand et al. derivation of elastic constants. The sandwich element is supported at the bottom face plate in point B and C and loaded with a horizontal shear force with magnitude of two unit loads in point A.*

#### 2.5.2.1 Transverse shear stiffness's

The transverse shear stiffness's $D_{Qx}$ and $D_{Qy}$ according to Nordstrand et al. (1994) are given in equations (2.24) and (2.25).

$$D_{Qx} = \frac{ht_c}{pS} G_{xy} \tag{2.24}$$

$$D_{Qy} = \frac{H/bp}{v/h} \tag{2.25}$$

Where:

$H$ Horizontal shear force acting on the element in the global system [N/m]
$b$ Width of sandwich plate in x-direction [m]
$v$ Displacement in y-direction [m]

## 2.6 Effective notch stress method

The effective notch stress method is a method independent of detail categories and the nominal stress, which in most cases has to be calculated numerically using computers. The ENS-method is instead based on the local stress at the point of crack initiation in a welded detail and a single fatigue strength curve.

The fatigue life in welded structures is to a high extend affected by stress concentrations caused by geometrical discontinuities such as holes, sharp corners and cracks. These kinds of stress raisers are also rather hard to avoid in welded structures. The total local stress at such stress concentrations is called the "notch stress". The notch stress can reach high values that tend to infinity if the considered notch is very sharp. This causes problems in fatigue assessment and is solved by introducing the effective notch stress which is the average stress over a certain distance (Al-Emrani & Aygül, 2014).

For fatigue assessment with the effective notch method the local stress at a critical point is calculated assuming an effective notch radius. The effective notch radius avoids stress singularity and considers the influence of varying weld shape and non-linear material behaviour at the notch root. The fatigue strength is then evaluated using a single fatigue strength curve. The use of a single curve simplifies a comparison between different weld geometries (Hobbacher, 2013). Another advantage of the method is that it can be applied for both root cracking and toe cracking of fillet welds.

The effective notch stress method presented by Radaj (2006) accounts for stress averaging in Neuber's micro-support theory by the effective notch radius. The effective notch radius is calculated according to equation (2.26) and illustrated in Figure 2.9. Even though the effective notch radius could be calculated for any real notch radius it is mostly applied for the worst case considering a sharp notch (ρ=0). Taking the micro-support length ($\rho^*$) as 0.4 and the constraint factor (s) as 2.5 the effective notch radius for structural steel becomes 1mm for plates with a thickness larger than 5mm (Radaj, et al., 2006).

$$\rho_f = \rho + s\rho^* \tag{2.26}$$

Where:
$\rho$ Real notch radius
$s$ Constraint factor
$\rho^*$ Micro-support length

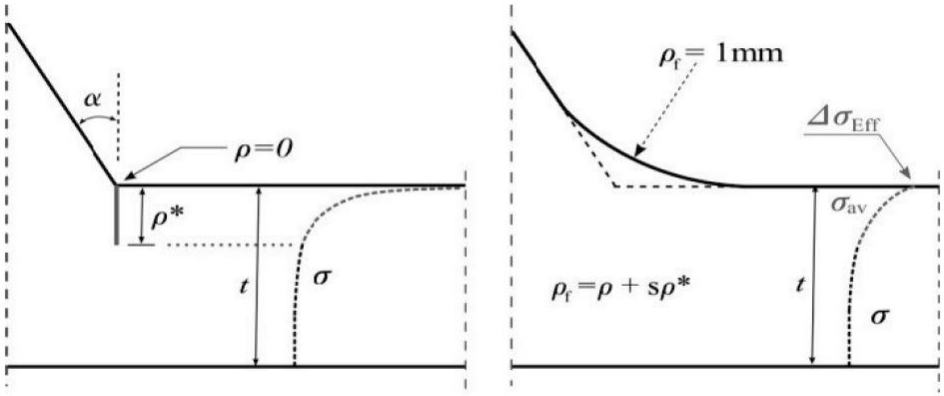



*Figure 2.9 Neuber's micro-support theory for calculation of effective notch radius for welded joint (Al-Emrani & Aygül, 2014).*

### 2.6.1 Determination of effective notch stress

The effective notch stress is most commonly determined by FE-analysis. The stress concentration and the surrounding geometry are generally modelled with 3D models with solid elements but can for simpler cases be modelled with 2D plane stress or strain elements. The element size is of importance since adequate mesh density is needed to capture the maximum stress at the point of stress concentration (Al-Emrani & Aygül, 2014). The international institute of welding (IIW) has provided recommendations for the element size that can be seen in Table 2.1.

*Table 2.1 IIW recommended element size using effective notch stress method (Al-Emrani & Aygül, 2014).*

| *Element type* | | *Element size* | | |
|---|---|---|---|---|
| | | *Relative size* | *r = 1 mm (t ≥ 5 mm)* | *r = 0.05 mm (t < 5 mm)* |
| **Hexahedral** | Quadratic | ≤ r/4 | 0.25 mm | 0.012 mm |
| | Linear | ≤ r/6 | 0.15 mm | 0.008 mm |
| **Tetrahedral** | Quadratic | ≤ r/6 | 0.15 mm | 0.008 mm |

In the FE-model the sharp edges is smoothed with an effective notch radius to avoid stress singularity and ensure that a convergent stress value is obtained. Rounding shapes of sharp edges at weld toes and weld roots can be seen in Figure 2.10. For weld roots IIW recommends the modelling to be made with either a keyhole or U-shape root configuration, see Figure 2.10. The U-shape should be used for non-load carrying welds while both configurations can be used for load carrying welds.

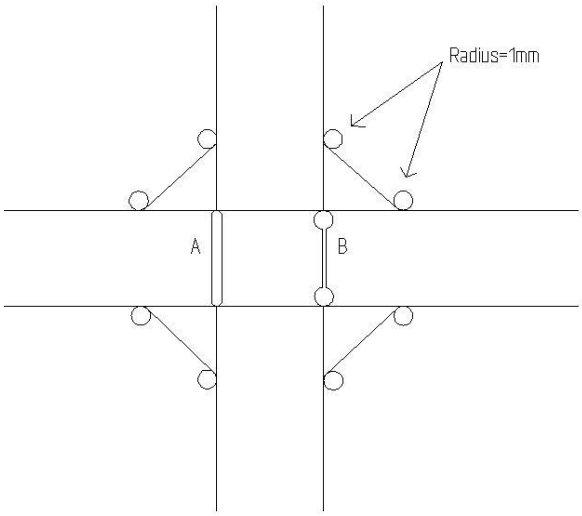



*Figure 2.10 Rounding of weld toes and weld roots with a radius of 1 mm. Two different modelling configurations can be used for weld roots: A) U-shape, B) keyhole.*

### 2.6.2 Assessment of fatigue life

The fatigue strength of a detail is evaluated using a single universal S-N curve since the effective notch method considers both the global stress concentration effects and the effect of local geometry. IIW recommends an S-N curve with FAT 225 for plates thicker than 5 mm. The recommended S-N curve can be seen in Figure 2.11.

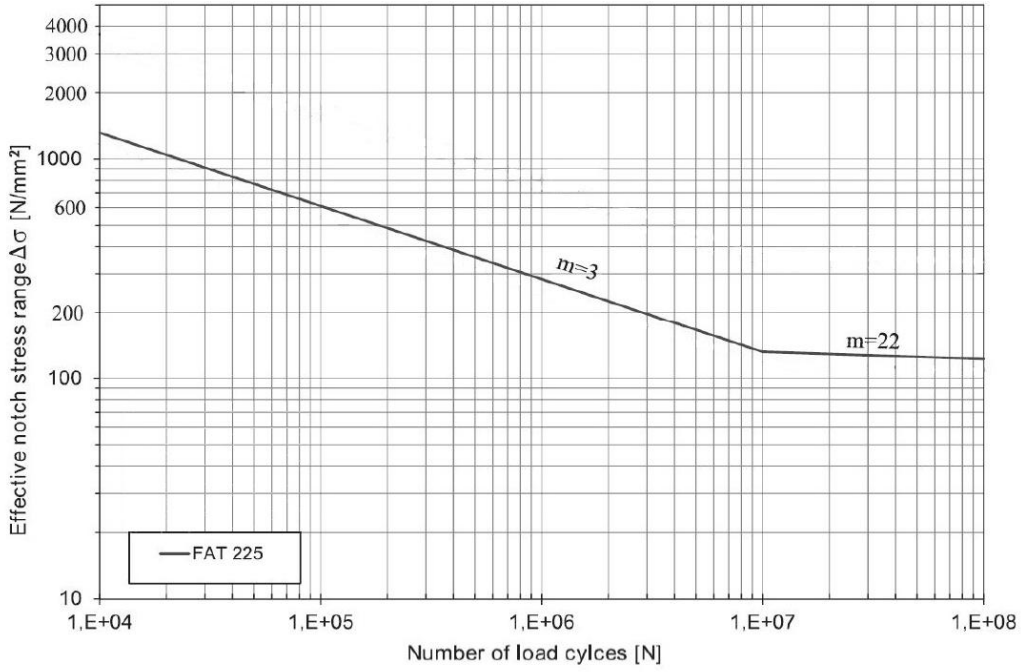



*Figure 2.11 Recommended S-N curve for effective notch stress method. The lowest solid line corresponds to FAT 225.*

## 2.7 Factorial design

Factorial design or full factorial design is a statistical approach to design of experiments where two or more factors are studied. The design is performed by selecting the number of levels for each of the factors of interest, and then to execute experiments for all possible combinations. The factors can be either quantitative and/or qualitative and the design is used to capture the main effects of each factor and interaction between factors. However, the number of runs grows exponentially with the number of factors which rapidly could make the experiments tedious and time-consuming (Box, et al., 2005).

### 2.7.1 Fractional factorial design

Fractional factorial design is also a statistical approach to design of experiments but in this case only a fraction of the experiments in the full factorial design is executed. This allows one to reduce the number of experiments significant but still get a good result when screening for the most important factors. For the fractional factorial design its resolution has to be chosen, which is the ability to separate main effects and low-order interactions from each other. The most commonly used resolutions are III, IV, and V (Box, et al., 2005). If resolution IV is chosen, three-factor interaction and higher order interaction will affect the main effects. Nevertheless, this noise to the result is commonly of low magnitude.

## 2.8 Regression analysis

Regression analysis is a statistical method for analysis of the relation between one dependent response variable and one or several independent predictor variables. The purpose of this analysis is to describe the observed variation of the response variable in terms of corresponding knowledge about the predictor variables. It is often of interest if and how much the predictor variables affects the response variable. The relation is expressed as a function of the variables and since it is hard to avoid some randomness between the variables the relation is seldom perfect, but for most cases a suitable approximation (Sykes, 1993).

The quantity of the variance in the dependent variable that is derived from the independent variables is described by the coefficient of determination, $R^2$ (Blomqvist, 2010). The coefficient of determination describes the fit of the data to the model, i.e. the regression equation and in the case of an FE-analysis, the analysis is deterministic i.e. no randomness is involved.

# 3 Parametric study

In this chapter the scripts used to create the FE-models needed in the parametric study are described. Also modelling choices for the FE-models are displayed and the input data for de different analyses are presented.

## 3.1 Work flow

The parametric study was carried out using the software BRIGADE/Plus 6.1. BRIGADE/Plus 6.1 is a tool-box for analysis and design of bridges and civil structures and includes an integrated Abaqus FEA solver from SIMULIA. To generate the numerous models of a unit cell, Python scripts was used. All the analyses are based on the same Python-script with small changes, made to fit the required analysis. After the analysis in BRIGADE/Plus the wanted results were saved in a text files and then post processed using Mathcad or Matlab.

The python scripts used consist of the main script MASTER.py and several other sub-scripts, executed within the main script. The script INPUTS.py contained the parameters and their values varied within this parametric study. For the analyses made with factorial design the script also considers the level and resolution chosen and creates the right amount of models with corresponding values for the parameters according to the factorial design matrix. The script PART.py sketches the cross-section of the unit cell and creates the geometry for the part while PROPERTY.py assigns the material properties and material orientation to the cross-section. ASSEMBLY.py defines the geometry of the assembly for the model by creating an instance for the part. In STEP.py the output requested from the analysis are defined. The script INTERACTION.py restrains the behaviour of nodes on the edges of the cross-section with regard to defined reference points. In LOAD.py the load and boundary conditions are applied. MESH.py generates the mesh for the instance and JOB.py creates the input-file.

The main script MASTER.py sends the input-files to the BRIGADE solver and retrieves the result files. MASTER.py then calls for OUTPUT.py to extract the relevant results from the analysis. The results for each model are saved in OUTPUT.txt and the corresponding values for the parameters are saved in INDATA.txt. A flow chart of showing the structure of the python scripts can be seen in Figure 3.1, and all scripts are shown in Appendix A.

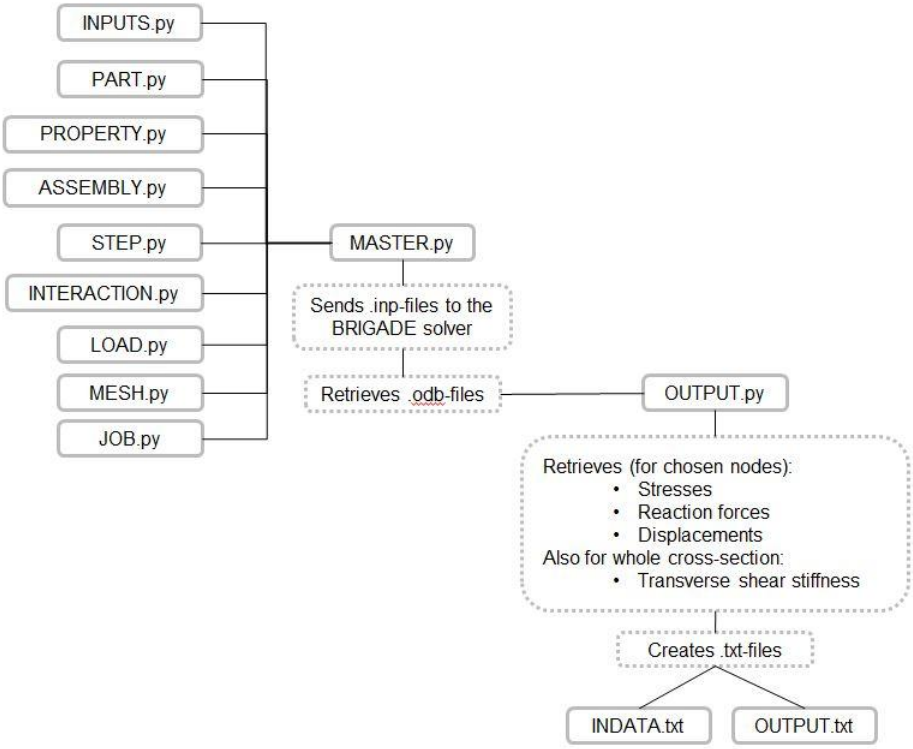



*Figure 3.1 Flow chart showing the structure of the python scrips used in the parametric study.*

The total parametric study of this thesis can be divided into 3 separate sub studies. The first study was a comparison of how the choice of the weld joint configuration, one weld or two welds between the face plate and corrugation, affects the response. The second study evaluated the main effects for each parameter using fractional factorial design. The intention was to screen for the most important parameters and identify the influence of each parameter on the effective notch stress and the transverse shear stiffness. In the last study a full factorial design matrix was executed generating a large number of analyses. The response was then evaluated with regression analysis to create an expression predicting the effective notch stress under a unit shear force and the transverse shear stiffness.

## 3.2 FE-model

The modelling choices such as element type, material data, boundary conditions and loads for the FE-models are displayed below. Also the input data for de different analyses are presented.

### 3.2.1 Element types and global coordinate system

The unit cell was modelled with four-node 2D continuum elements. The global coordinate system for the unit cell can be seen in Figure 3.2.

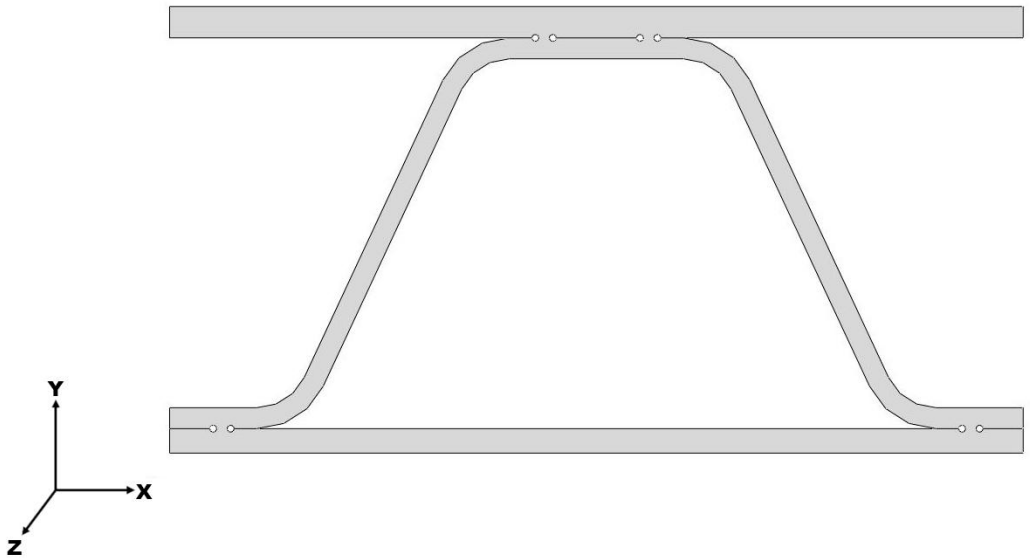



*Figure 3.2 Global coordinate system. The x-axis was orientated in horizontal direction, the y-axis was orientated in vertical direction and the z-axis was orientated out of the plane.*

### 3.2.2 Material data

The unit cell was modelled with elastic isotropic material properties. The choice for Young's modulus, E, and Poisson's ratio, ν, can be seen in Table 3.1. The Young's modulus was calculated with equation (3.1) to consider plain strain conditions. E=210 GPa was used in the calculation.

$$E_{plain.strain} = \frac{E}{\left(1-\nu^2\right)} \tag{3.1}$$

*Table 3.1 Material parameters for steel used in the FE-model*

| **Material** | **$E_{plain.strain}$ [GPa]** | **ν [-]** |
|---|---|---|
| Steel | 230 | 0.3 |

### 3.2.3 Boundary conditions

To obtain the same behaviour in all the nodes on the short edges of the face plates and the corrugation, interaction conditions was used. Reference points were created a small distance outside the corresponding edge and were chosen as the control points. The edges were chosen as the surface to be controlled, where the constrained degrees of freedom was translation in x- and y-direction. The reference points outside the corresponding edge can be seen in Figure 3.3.

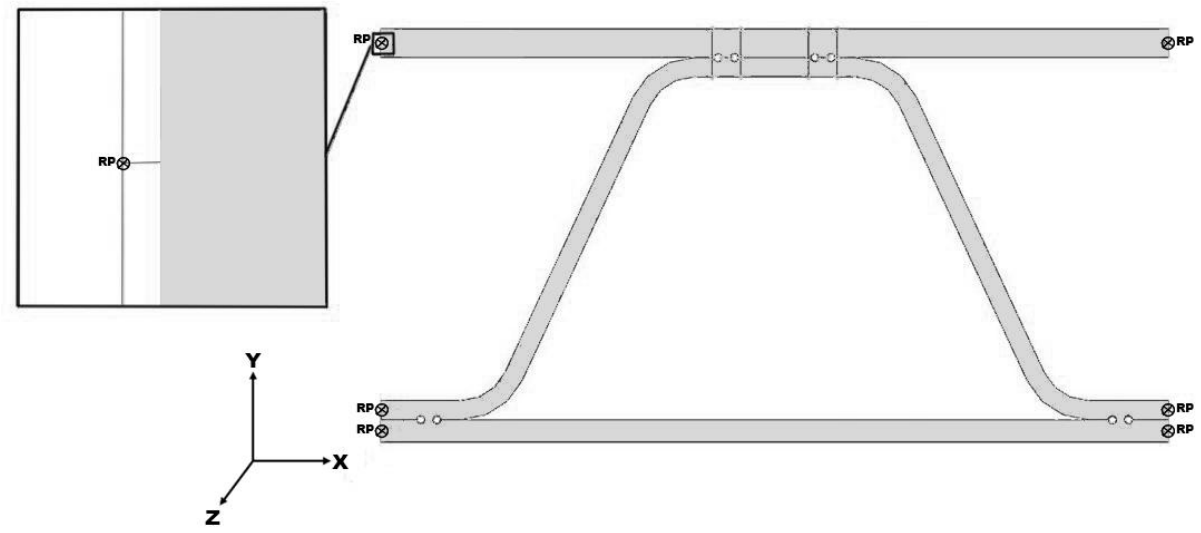



*Figure 3.3 Interaction between the reference points and the edges of the unit cell (edges of the face plates and corrugation). Each reference point was placed a small distance outside the corresponding edge.*

The boundary conditions for the model were applied in the reference points. For the model with one weld the bottom face plate was prevented from translation in x- and y-direction. Moreover, the top face plate was prevented from translation in y-direction. The boundary conditions for the model with one weld can be seen in Figure 3.4.

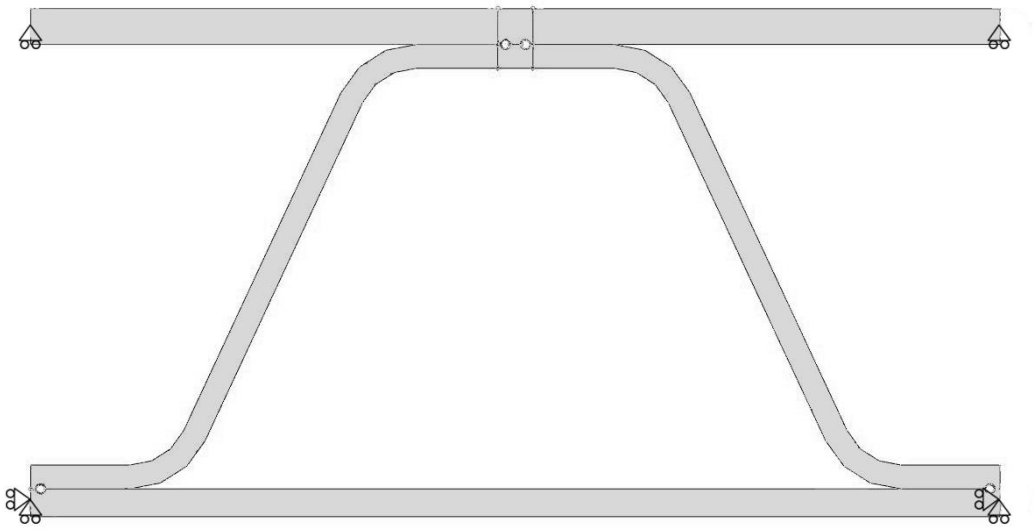


*Figure 3.4 Boundary conditions for model with one weld.*

For the model with two welds the bottom face plate was prevented from translation in x- and y-direction. Furthermore, the corrugation and the top plate are prevented from translation in y-direction. The boundary conditions for the model with two welds can be seen in Figure 3.5.

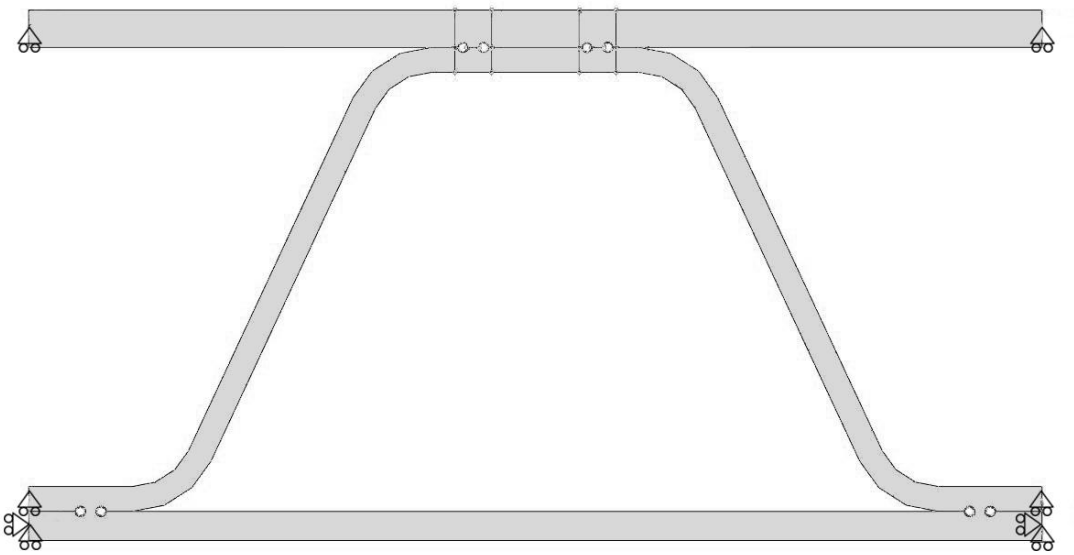


*Figure 3.5 Boundary conditions for model with two welds*

The boundary conditions for the corrugation differ between the two models. For the model with one weld the corrugation is prevented from movement by the weld at its ends. For the model with two welds the corrugation ends was free to move and therefore needs a boundary condition to reflect the situation where several cells are connected to each other. This was done by preventing the translation in y-direction for the core.

The choice of boundary condition for the corrugation of the two-weld model was analysed for verification. The displacement for one cell, with the boundary condition for the corrugation, was compared with the displacement for the middle cell in a model with 10 cells connected in a row. The verification was made for three different models, with different cross-sectional geometry, and the difference in displacement can be seen in Table 3.2.

*Table 3.2 Displacement in y-direction of the top face plate for 1 cell and for 10 cells for three models with different values of the parameters.*

| | **Displacement [m]** |
|---|---|
| **Model 1** | |
| 1 cell | 1.81128E-08 |
| 10 cells | 1.81106E-08 |
| **Model 2** | |
| 1 cell | 4.33712E-09 |
| 10 cells | 4.33697E-09 |
| **Model 3** | |
| 1 cell | 3.53014E-09 |
| 10 cells | 3.52856E-09 |

The difference in displacement between 1 cell and 10 cells was very small and it is therefore considered that preventing the translation of the corrugation in y-direction for one cell is a good assumption.

The stress distribution through the top face plate was also compared for the different models. The distribution for one cell and 10 cells was almost identical for all three models which further prove that the proposed boundary condition was a good approximation.

### 3.2.4 Loads

The load effect considered in this study is shear deformation. The model was loaded with a unit load in the y-direction at both edges of the top face plate. The load was applied in the reference point as a concentrated force and the loads can be seen in Figure 3.6.

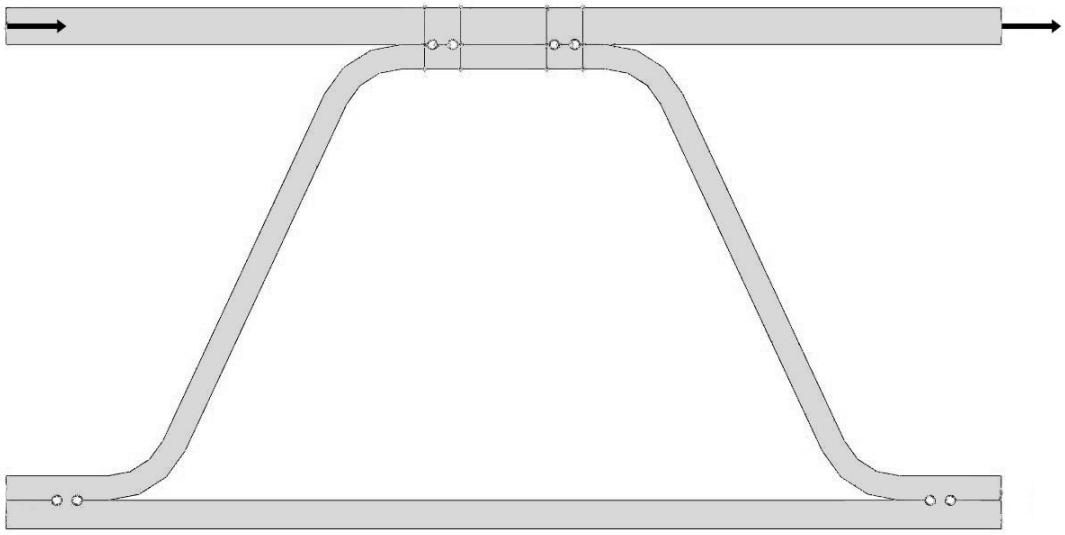


*Figure 3.6 The model is loaded in y-direction with two loads with the magnitude 1. The loads were applied at each edge of the top face plate in each corresponding reference point.*

### 3.2.5 Modelling of welds

The welds are modelled with a notch radius of 1 mm according to the ENS-method. However, the solution will be approximate since the notch will influence the transverse shear stiffness of the cross-section. This effect was investigated and the results can be seen in Table 3.3. It was concluded that for the smallest plate thicknesses used within this thesis there was an influence from the notch of less than 4 %.

*Table 3.3 Effects of the notch size on the transverse shear stiffness. The transverse shear stiffness was calculated for two cases, the first with r=1mm and the second with r=0.1mm.*

| Notch size [mm] | Transverse shear stiffness [Pa] | Difference |
|---|---|---|
| 1 | 4.489E+06 | 96.3% |
| 0.1 | 4.66E+06 | |

To link this study to previous research, the transverse shear stiffness of the cross-section was calculated using the displacement obtained from the FE-model together with the definition of shear strain, see e.g. Nordstrand et al. The stiffness from the FE-model was then compared to the transverse shear stiffness calculated using the equations derived by Libove and Hubka (1951). In addition a beam model was also created to further develop the comparison. The calculated stiffness's can be seen in Table 3.4 and the geometrical properties of the cross-section used in the comparison can be seen in Table 3.5.

*Table 3.4 Transverse shear stiffness calculated using three different methods. The first using the displacement obtained from model developed in this thesis and the equations derived by Nordstrand et al. The second using the equations derived by Libove and Hubkas. At last, the third using the displacement obtained from a beam model and equations derived by Nordstrand et al.*

| Calculations made using: | Transvers shear stiffness [Pa] |
|---|---|
| Plain strain model (the FE-model used in this thesis) | 7.578E+06 |
| Libove and Hubka theory | 1.005E+07 |
| Beam model | 1.039E+07 |

*Table 3.5 Geometrical properties for the cross-section used in the comparison of transverse shear stiffness between the plain strain model, the Libove and Hubka theory and the beam model.*

| Parameter | Value |
|---|---|
| $t_1$ | 10 mm |
| $t_c$ | 6 mm |
| $t_2$ | 10 mm |
| $\theta$ | 80 ° |
| $f_1$ | 38.5 mm |
| $f_2$ | 38.5 mm |
| h | 136 mm |
| $t_w$ | 1.5 mm |
| R | 12 mm |

The transverse shear stiffness for the plain strain model was about 30 % less than the stiffness calculated using the theory developed by Libove and Hubka for the same cross-section. The stiffness for the beam model on the other hand corresponds well with the stiffness from Libove and Hubka. The reason for the differences and similarities in stiffness is related to the modelling of the weld. Libove and Hubka base their theory on the assumption that the weld is a rigid joint. The beam model was modelled with the weld being a very stiff beam element which is equivalent to the assumption made by Libove and Hubka and therefore a good correlation between the stiffness's are obtained. For the FE-model used in this thesis the contact between the face plate and the core is excluded and the weld was modelled with continuum elements giving it the possibility to deform and a small angle is created in the connection. This way to model the welds is closer to the real behaviour and is considered to be the reason why the stiffness is lower for the plain strain model. The deformation figures for the plain strain model and the beam model can be seen in Figure 3.7.

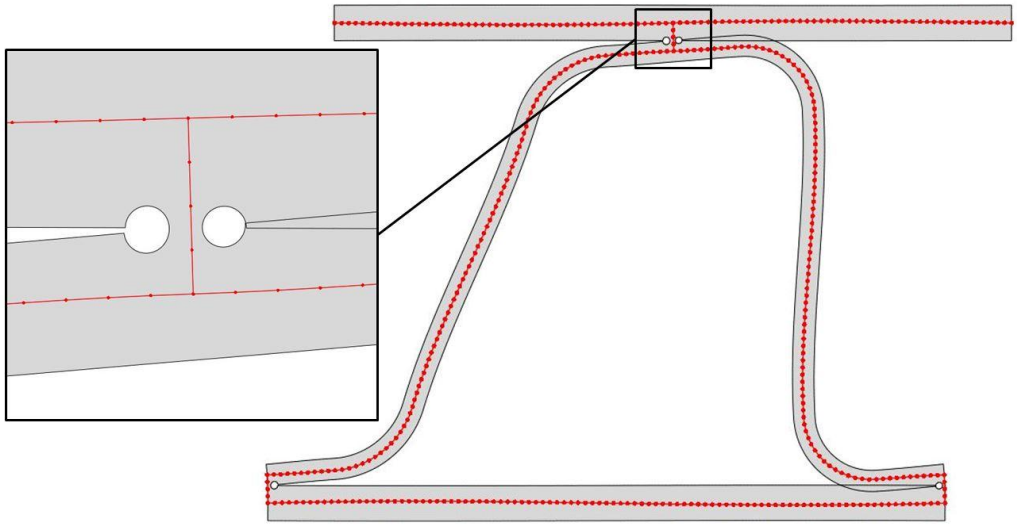


*Figure 3.7* *Deformation figures for the plain strain model and the beam model. The plain strain model is displayed in black and white while the beam model is displayed in red. In the zoomed box the small angle created for the plain strain model is visualized.*

### 3.2.6 Mesh

The model was meshed with tree different sizes. Around the notches the mesh is finest and the edges of the notch were seeded with a size of 0.15 mm according to IIW recommendations. The rest of the model was globally seeded with a size of 1 mm.

### 3.2.7 Extraction of results

From the model different results are extracted from different nodes. Around the notches stresses are extracted, both the von Mises stress and the principal stress. The maximum magnitude of each stress and its position in the notch is documented. In all reference points the reaction forces are extracted and for the two reference points corresponding to the top face plate the displacement in y-direction is extracted.

## 3.3 Input data

The input data used in the three different analyses; the comparison between one weld and two welds, the fractional factorial design and the full factorial design is presented below.

### 3.3.1 Comparison between one weld and two welds

The comparison was done for 5 models. The input data for the models were chosen randomly within reasonable intervals for each parameter to capture both the individual behaviour in different parts of the parameter range and the general behaviour. The values for the parameters for the different models can be seen in Table 3.6.

*Table 3.6 Values chosen for the 5 models evaluated with regard to the comparison between one weld and two welds. The parameter $d_w$ is only used in the case with two welds.*

| Parameter | Model 1 | Model 2 | Model 3 | Model 4 | Model 5 |
|---|---|---|---|---|---|
| $t_1$ | 8 mm | 6 mm | 4 mm | 5 mm | 7 mm |
| $t_c$ | 4 mm | 7 mm | 8 mm | 5 mm | 6 mm |
| $t_2$ | 8 mm | 6 mm | 10 mm | 7 mm | 9 mm |
| $\theta$ | 50 ° | 60 ° | 70 ° | 80 ° | 65 ° |
| $f_1$ | 50 mm | 50 mm | 50 mm | 50 mm | 50 mm |
| $f_2$ | 50 mm | 50 mm | 50 mm | 50 mm | 50 mm |
| h | 150 mm | 130 mm | 100 mm | 160 mm | 120 mm |
| $t_w$ | 3 mm | 3 mm | 3 mm | 3 mm | 3 mm |
| $d_w$ | 30 mm | 30 mm | 30 mm | 30 mm | 30 mm |
| R | 12 mm | 21 mm | 24 mm | 15 mm | 18 mm |

### 3.3.2 Fractional Factorial Design

The fractional factorial design was also done for 5 models to capture both the general as well as the individual behaviour in the reasonable range for the parameters. In the factorial design all the models were modelled with two welds. The values in Table 3.7 are the original values which then were varied with ±10% in the fractional factorial design matrix. Resolution IV was chosen and the number of runs in the fractional factorial design matrix for each of the 5 models was 64.

*Table 3.7 Original values chosen for the 5 models evaluated with Fractional Factorial Design.*

| Parameter | Model 1 | Model 2 | Model 3 | Model 4 | Model 5 |
|---|---|---|---|---|---|
| $t_1$ | 8 mm | 6 mm | 4 mm | 5 mm | 7 mm |
| $t_c$ | 4 mm | 7 mm | 8 mm | 5 mm | 6 mm |
| $t_2$ | 8 mm | 6 mm | 10 mm | 7 mm | 9 mm |
| θ | 50 ° | 60 ° | 70 ° | 80 ° | 65 ° |
| $f_1$ | 50 mm | 50 mm | 50 mm | 50 mm | 50 mm |
| $f_2$ | 50 mm | 50 mm | 50 mm | 50 mm | 50 mm |
| h | 150 mm | 130 mm | 100 mm | 160 mm | 120 mm |
| $t_w$ | 3 mm | 3 mm | 3 mm | 3 mm | 3 mm |
| $d_w$ | 30 mm | 30 mm | 30 mm | 30 mm | 30 mm |
| R | 12 mm | 21 mm | 24 mm | 15 mm | 18 mm |

### 3.3.3 Regression Analysis

The selection of values for the studied parameters in the regression analysis was limited by a reasonable range with respect to production, bridge application and run-time. For each parameter an individual range was determined in consultation with the supervisor. This result in a different amount of values for different parameters and the values for each parameter can be seen in Table 3.8. The total number of runs in the full factorial design matrix was 28 800.

*Table 3.8 Values for the parameters chosen for the full factorial design. To avoid the risk of the parameter $d_w$ being larger than $f_1$ or $f_2$, $\eta_1$ and $\eta_2$ are introduced. $\eta_1$ is the distance from the edge of the lower horizontal part of the core to the weld. $\eta_2$ is the distance from the edge of the upper horizontal part of the core to the weld. $f_1$ and $f_2$ are then calculated from $d_w$, $\eta_1$, and $\eta_2$.*

| Parameter | Values |
|---|---|
| $t_1$ | 5, 6, 7, 8, 9, 10, 11, 12 mm |
| $t_c$ | 4, 5, 6, 7, 8, mm |
| $t_2$ | 5, 6, 7, 8, 9, 10, 11, 12 mm |
| θ | 40, 50, 60, 70, 80 ° |
| $\eta_1$ | 20 mm |
| $\eta_2$ | 20 mm |
| h | 120, 140, 160, 180, 200, 220 mm |
| $t_w$ | 3 mm |
| $d_w$ | 30, 45, 60 mm |
| R | $3 \cdot t_c$ |

# 4 Results

The results obtained from the three different analyses; the comparison between one weld and two welds, the fractional factorial design and the regression analysis is presented below.

## 4.1 Comparison between one weld and two welds

A comparison of the stresses and the transverse shear stiffness between 1 weld and 2 welds was made for 5 models. The obtained maximum principal stress and the calculated shear stiffness for the cross-sections can be seen in Table 4.1. The calculation can be seen in Appendix C.

*Table 4.1 Maximum stress and transverse shear stiffness for 5 models both for the case with 1 weld and 2 welds.. The table also show calculated values of the transverse shear stiffness according to Libove & Hubka for all the 5 models.*

| | | **Model 1** | **Model 2** | **Model 3** | **Model 4** | **Model 5** |
|---|---|---|---|---|---|---|
| **1 weld** | Max. stress [Pa/(N·m)] | 2,25E+04 | 7,96E+03 | 5,61E+03 | 1,24E+04 | 1,01E+04 |
| | Normalized position of max.stress along path [-] | 0,54 | 0,52 | 0,55 | 0,51 | 0,52 |
| **2 welds** | Max. stress [Pa/(N·m)] | 7,84E+03 | 2,87E+03 | 1,64E+03 | 3,96E+03 | 2,58E+03 |
| | Normalized position of max.stress along path [-] | 0,71 | 0,29 | 0,33 | 0,26 | 0,26 |
| | **Decrease** | 65 % | 64 % | 71 % | 68 % | 74 % |
| **1 weld** | Transverse shear stiffness [Pa] | 6,04E+06 | 1,62E+07 | 2,23E+07 | 3,38E+06 | 1,05E+07 |
| **2 welds** | Transverse shear stiffness [Pa] | 1,53E+07 | 1,91E+07 | 2,45E+07 | 4,01E+06 | 1,55E+07 |
| | **Increase** | 154 % | 18 % | 10 % | 19 % | 47 % |
| | Transverse shear stiffness calculated according to Libove & Hubka [Pa] | 9,71E+06 | 1,95E+07 | 2,57E+07 | 3,87E+06 | 1,39E+07 |

The table shows a consistent reduction of the effective notch stress of 64-74% in the case of two welds compared with one weld. It can also be seen that the position of the maximum stress changes from the case with one weld compared with two welds. For one weld the maximum stress is located in the root of the weld while the maximum stress is located in the toe of the weld for the case with two welds. The result for the transverse shear stiffness isn't as clear as for the stress. The transverse shear stiffness increases with 10-154%, when adding an additional weld, depending on the cross-sectional geometry.

In addition, deformations figures for the model with the highest increase in transverse shear stiffness, model 1, and for the model with the smallest increase in transverse shear stiffness, model 3, was created and are presented in Figure 4.1 and Figure 4.2. It can be noted from the figures that for model 1 an additional weld causes a higher interaction between core and top face plate. This effect can also be seen for model 3 but not with the same magnitude.

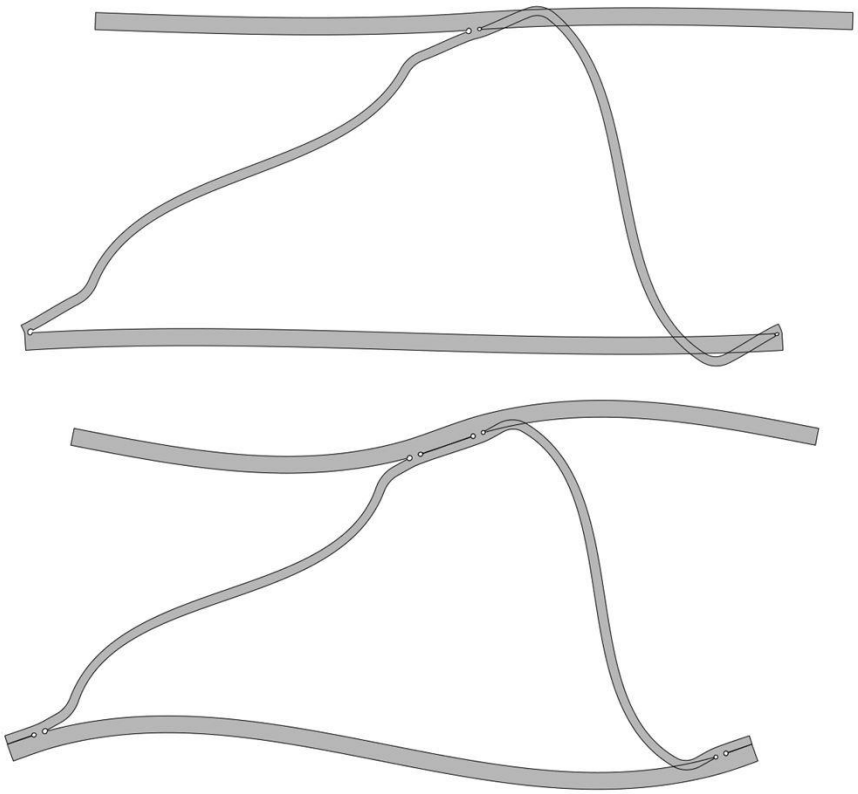


Figure 4.1 *Deformation figures for model 1. The top figure was modelled with 1 weld and the bottom figure was modelled with 2 welds.*

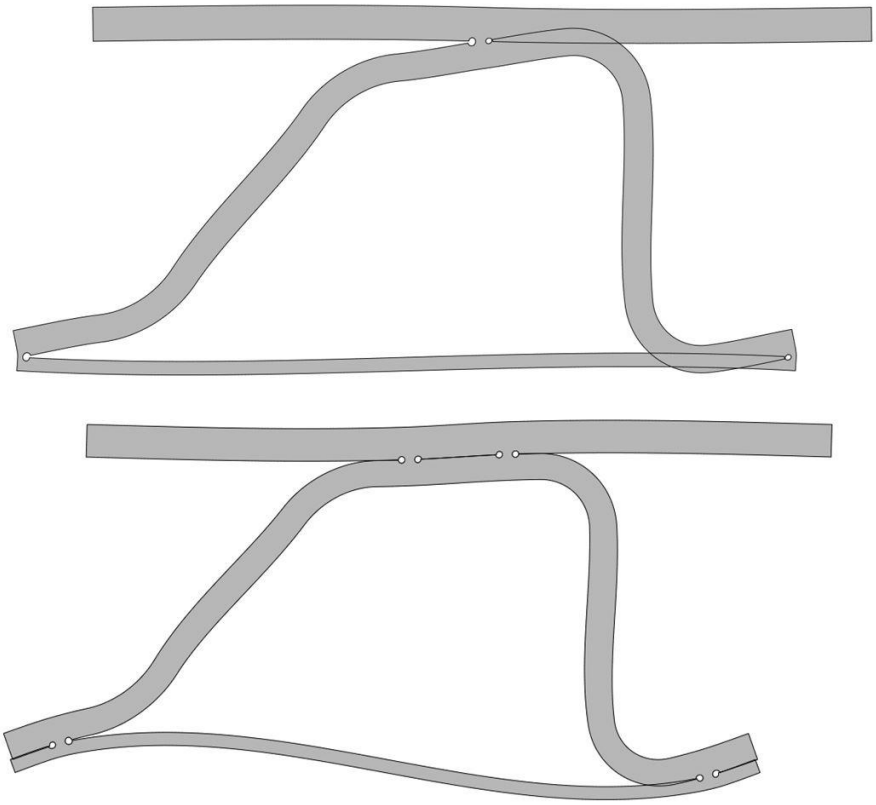


*Figure 4.2 Deformation figure for model 3. The top figure was modelled with 1 weld and the bottom figure was modelled with 2 welds.*

## 4.1 Fractional Factorial Design

From the stresses and transverse shear stiffness's obtained from the fractional factorial design-models, originating from each of the 5 models, the main effects are calculated using Mathcad and then plotted for visualization. The calculation can be seen in Appendix B. On the horizontal axis of the plot the low (-) and high (+) level average of each parameters is displayed. The low level average is the mean value when the parameter takes the -10%-value. The high level average is the mean value when the parameter takes the +10%-value. The normalized main effects for the effective notch stress can be seen in Figure 4.3. The main effects are normalized with respect to the highest positive value for each model.

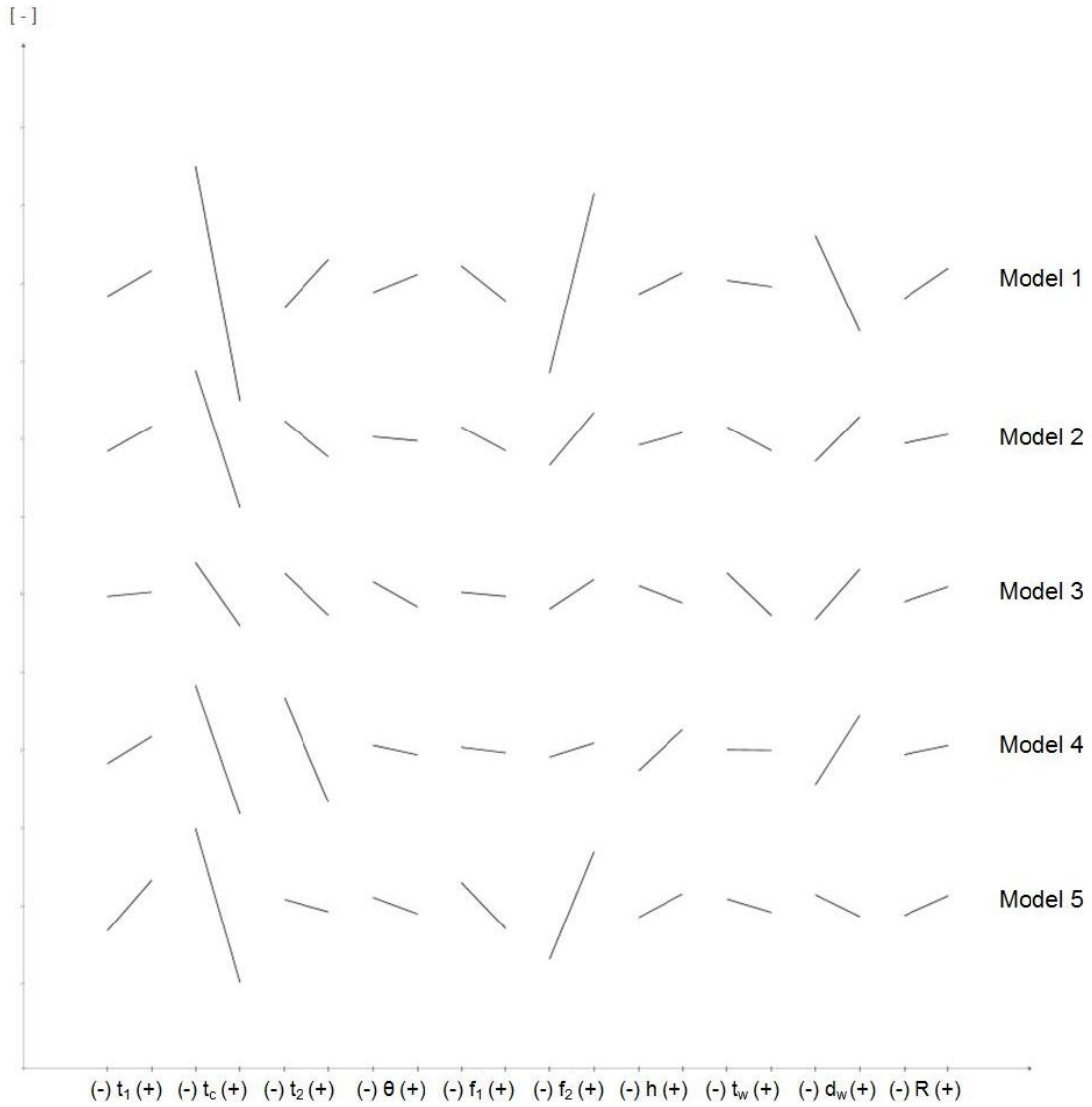



*Figure 4.3* *Main effects for the stress close to the weld for the 5 models. The length of the line shows the influence of the corresponding parameter, the longer the line the stronger the influence. The slope of the line shows how the parameter influences; a positive slope means that the stress is increased if the parameter is increased and a negative slope means that the stress is increased if the parameter is decreased.*

From Figure 4.3 it can be noted that one of the parameters that shows the greatest influence on the notch is $t_c$. The slope for $t_c$ is negative for all the 5 models, thus increasing the core thickness decreases the notch stress. Furthermore, $f_2$, $d_w$ and $t_2$ shows a large influence. The slope for $f_2$ is positive and the slope for $d_w$ indicates an irregular behaviour being both negative and positive. The parameter $t_2$ has a positive slope for model 1 but a negative slope for model 2-5.

The remaining parameters show a less important influence but the slopes for each remaining parameter is still interesting to note. The parameter $t_1$ has a positive slope for all the models. For θ the slope is positive for model 1 but negative for model 2-5. The parameter $f_1$ behaves similar for all the models and

has a negative slope. The slope for the height h is positive for model 1,2,4,5 but negative for model 3. At last $t_w$ has a negative slope for all the models and R a positive slope for all models.

The normalized main effects for the transverse shear stiffness can be seen in Figure 4.4. The main effects for the transverse shear stiffness are normalized with respect to the highest positive value for each model.

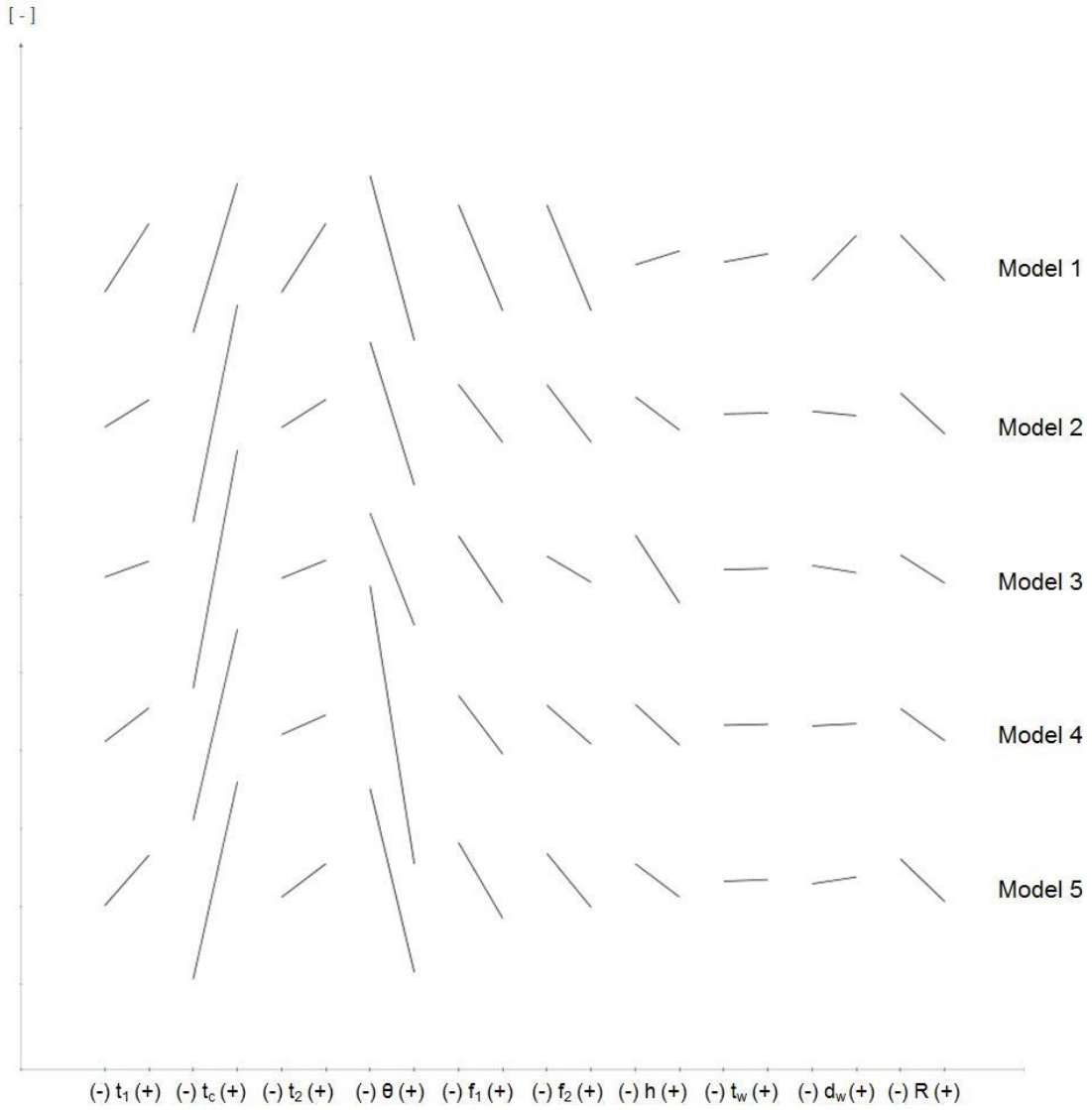



*Figure 4.4 Main effects for the transverse shear stiffness of the cross-section for the 5 models. The length of the line shows the influence of the corresponding parameter; the longer the line, the stronger the influence. The slope of the line shows how the parameter influences; a positive slope means that the transverse shear stiffness is increased if the parameter is increased and a negative slope means that the transverse shear stiffness is increased if the parameter is decreased.*

In Figure 4.4 it can be seen that the most important and influencing parameters on the transverse shear stiffness is $t_c$ and $\theta$. The slope for $t_c$ is positive for all models and the slope for $\theta$ is negative for all models.

For the remaining parameters with less influence the slopes are noted. $t_1$ together with $t_2$ both has a positive slope for all models. The parameters $f_1$ and $f_2$ has a negative slope for all models. Furthermore, h has a negative slope for all models except model 1 where it has a positive slope. $t_w$ has positive slopes for all models while $d_w$ is irregular with positive slopes for model 1,4,5 and negative for model 2 and 3. At last R has a negative slope for all models.

The main effects as a function of the mass of the cross-section was also calculated and plotted for both the stress and the transverse shear stiffness. The main effects for the stress including the effects of the mass can be seen in Figure 4.5.

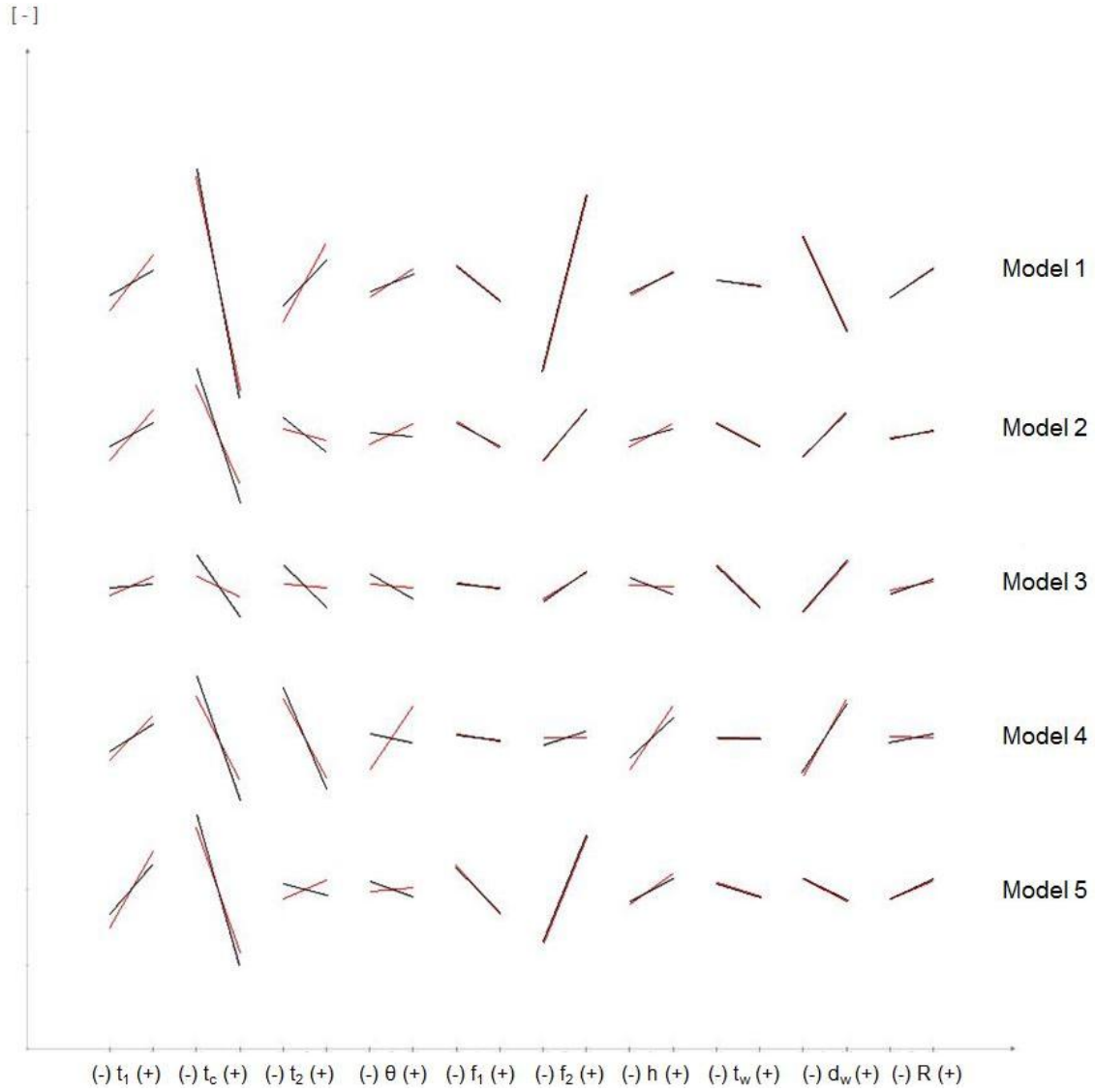



*Figure 4.5* *Main effects including the effect of the mass for the stress close to the weld for the 5 models. The black lines are the original main effects only considering the stress. The red lines are the main effects as a function of the mass.*

From Figure 4.5 it can be noted that the influence from $t_c$ is reduced when considering the weight of the cross-section. It can also be noted that the influence from $\theta$, $t_1$ and $t_2$ are the parameters with the largest increase in influence when also considering the weight of the cross-section.

The main effect for the transverse shear stiffness as a function of the mass can be seen in Figure 4.6.

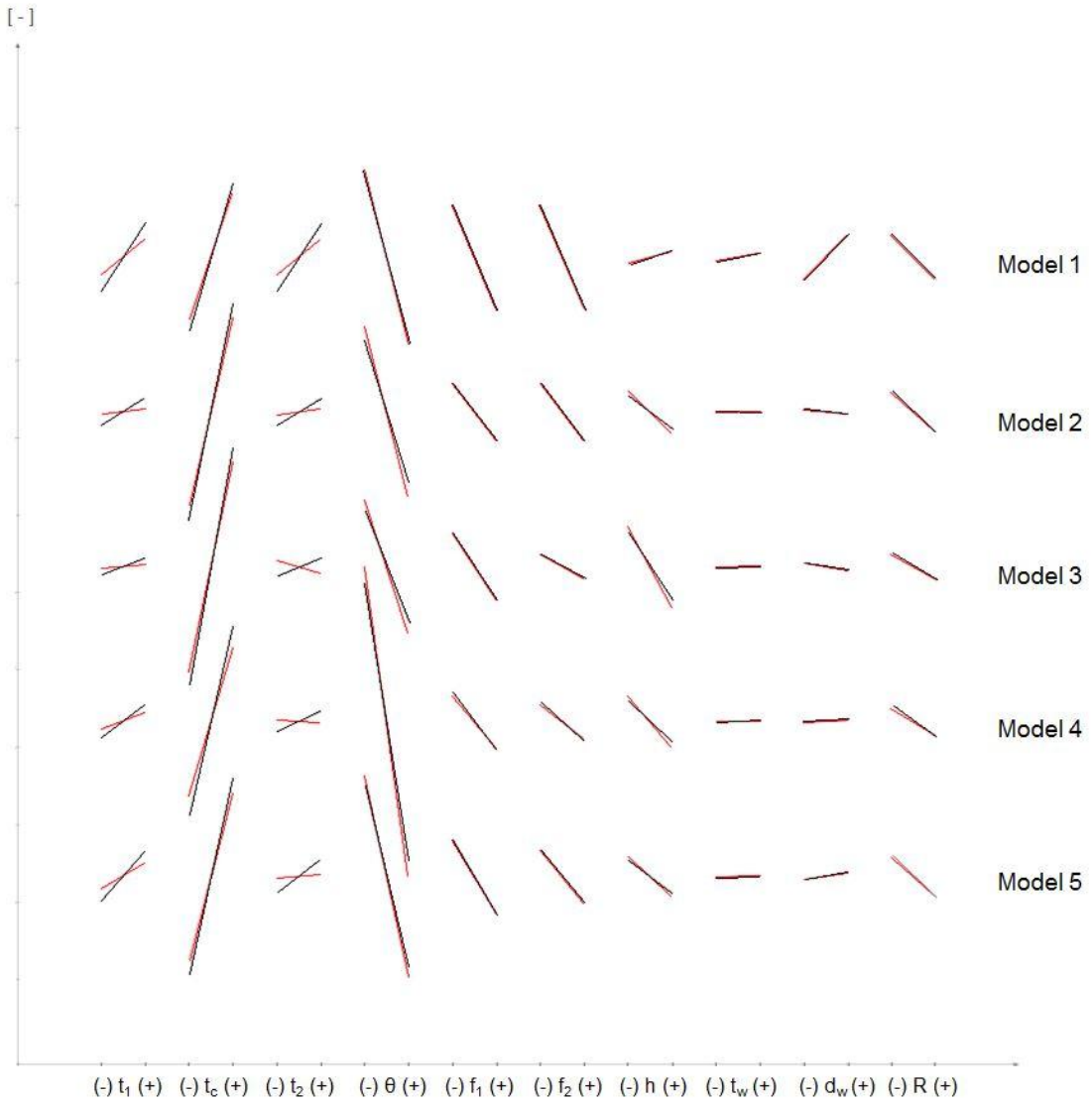



*Figure 4.6* *Main effects including the effect of the mass for the transverse shear stiffness for the 5 models. The black lines are the original main effects only considering the transverse shear stiffness. The red lines are the main effects as a function of the mass.*

From Figure 4.6 it can be noted that the influence from θ is considerably increased when considering the weight of the cross-section. It can also be noted that the influence from $t_c$ is reduced.

## 4.2 Regression Analysis

The stresses and transverse shear stiffness from executing the full factorial design matrix with 28 800 models was used as data for the regression analysis. In the regression analysis only the parameters with more than one value are included. The parameters $f_1$ and $f_2$ are replaced with f since they always adopt the same value. Moreover, $d_w$ is excluded since its linearly dependent of f and R is excluded sine its linearly dependent of $t_c$. The regression analysis was performed with both a linear regression model and a quadratic regression model. However,

the functions and corresponding β-values presented in this chapter are determined using the quadratic model since this model has a significantly higher coefficient of determination.

The function for the effective notch stress under a unit shear can be seen in equation (4.1). The corresponding β-values can be seen in Table 4.2 and the coefficient of determination can be seen in Table 4.3.

$$\begin{aligned}\sigma_{ENS}(t_1,t_c,t_2,\theta,f,h) = {} & \beta_0 + \beta_1 t_1 + \beta_2 t_c + \beta_3 t_2 + \beta_4 \theta + \beta_5 f + \beta_6 h + \\ & \beta_7 t_1 t_c + \beta_8 t_1 t_2 + \beta_9 t_1 \theta + \beta_{10} t_1 f + \beta_{11} t_1 h + \beta_{12} t_c t_2 + \beta_{13} t_c \theta + \beta_{14} t_c f + \\ & \beta_{15} t_c h + \beta_{16} t_2 \theta + \beta_{17} t_2 f + \beta_{18} t_2 h + \beta_{19} \theta f + \beta_{20} \theta h + \beta_{21} f h + \beta_{22} t_1^2 + \\ & \beta_{23} t_c^2 + \beta_{24} t_2^2 + \beta_{25} \theta^2 + \beta_{26} f^2 + \beta_{27} h^2\end{aligned} \tag{4.1}$$

*Table 4.2 β-values corresponding to the function for the effective notch stress under a unit shear, equation (4.1).*

| | **Value [-]** |
|---|---|
| $\beta_0$ | 0.0002E+08 |
| $\beta_1$ | 0.0109E+08 |
| $\beta_2$ | -0.0835E+08 |
| $\beta_3$ | 0.0090E+08 |
| $\beta_4$ | 0.0000E+08 |
| $\beta_5$ | -0.0003E+08 |
| $\beta_6$ | 0.0003E+08 |
| $\beta_7$ | -1.6347E+08 |
| $\beta_8$ | 0.5239E+08 |
| $\beta_9$ | 0.0024E+08 |
| $\beta_{10}$ | 0.0290E+08 |
| $\beta_{11}$ | 0.0130E+08 |
| $\beta_{12}$ | -1.1205E+08 |
| $\beta_{13}$ | -0.0047E+08 |
| $B_{14}$ | 0.0694E+08 |
| $\beta_{15}$ | -0.0830E+08 |
| $\beta_{16}$ | -0.0046E+08 |
| $\beta_{17}$ | -0.1942E+08 |
| $\beta_{18}$ | 0.0043E+08 |
| $\beta_{19}$ | 0.0003E+08 |
| $\beta_{20}$ | 0.0003E+08 |
| $\beta_{21}$ | -0.0004E+08 |
| $\beta_{22}$ | -0.5302E+08 |
| $\beta_{23}$ | 8.0072E+08 |
| $\beta_{24}$ | 0.7678E+08 |
| $\beta_{25}$ | -0.0000E+08 |
| $\beta_{26}$ | 0.0075E+08 |
| $\beta_{27}$ | -0.0001E+08 |

*Table 4.3 The coefficient of determination, $R^2$, for the function for the effective notch stress under a unit shear. $R^2$ is presented for both the linear model and the quadratic model. The coefficient of determination for the quadratic model is the one corresponding to the function in equation (4.1).*

| **Regression model** | **Coefficient of determination, R² [-]** |
|---|---|
| Linear | 0.7503 |
| Quadratic | 0.9375 |

The function for the transverse shear stiffness can be seen in equation (4.2). The corresponding β-values can be seen in Table 4.4 and the coefficient of determination can be seen in Table 4.5.

$$\begin{aligned}\sigma_{DQy}(t_1,t_c,t_2,\theta,f,h) = \beta_0 + \beta_1 t_1 + \beta_2 t_c + \beta_3 t_2 + \beta_4 \theta + \beta_5 f + \beta_6 h + \\ \beta_7 t_1 t_c + \beta_8 t_1 t_2 + \beta_9 t_1 \theta + \beta_{10} t_1 f + \beta_{11} t_1 h + \beta_{12} t_c t_2 + \beta_{13} t_c \theta + \beta_{14} t_c f + \\ \beta_{15} t_c h + \beta_{16} t_2 \theta + \beta_{17} t_2 f + \beta_{18} t_2 h + \beta_{19} \theta f + \beta_{20} \theta h + \beta_{21} f h + \beta_{22} t_1^2 + \\ \beta_{23} t_c^2 + \beta_{24} t_2^2 + \beta_{25} \theta^2 + \beta_{26} f^2 + \beta_{27} h^2\end{aligned} \quad (4.2)$$

*Table 4.4 β-values corresponding to the function for the transverse shear stiffness, equation (4.2).*

| | **Value [-]** |
|---|---|
| $\beta_0$ | 0.0001E+11 |
| $\beta_1$ | 0.0155E+11 |
| $\beta_2$ | 0.0746E+11 |
| $\beta_3$ | 0.0162E+11 |
| $\beta_4$ | -0.0002E+11 |
| $\beta_5$ | -0.0050E+11 |
| $\beta_6$ | 0.0000E+11 |
| $\beta_7$ | 1.1062E+11 |
| $\beta_8$ | 1.0222E+11 |
| $\beta_9$ | -0.0137E+11 |
| $\beta_{10}$ | -0.0812E+11 |
| $\beta_{11}$ | 0.0037E+11 |
| $\beta_{12}$ | 1.1648E+11 |
| $\beta_{13}$ | -0.0390E+11 |
| $B_{14}$ | -0.5608E+11 |
| $\beta_{15}$ | -0.0834E+11 |
| $\beta_{16}$ | -0.0142E+11 |
| $\beta_{17}$ | -0.0853E+11 |
| $\beta_{18}$ | 0.0031E+11 |

| | |
|---|---|
| $\beta_{19}$ | 0.0038E+11 |
| $\beta_{20}$ | -0.0002E+11 |
| $\beta_{21}$ | 0.0019E+11 |
| $\beta_{22}$ | -0.2681E+11 |
| $\beta_{23}$ | 3.0590E+11 |
| $\beta_{24}$ | -0.2687E+11 |
| $\beta_{25}$ | 0.0001E+11 |
| $\beta_{26}$ | 0.0222E+11 |
| $\beta_{27}$ | 0.0011E+11 |

*Table 4.5 The coefficient of determination, $R^2$, for the function for the transverse shear stiffness. $R^2$ is presented for both the linear model and the quadratic model. The coefficient of determination for the quadratic model is the one corresponding to the function in equation (4.2).*

| **Regression model** | **Coefficient of determination, $R^2$ [-]** |
|---|---|
| Linear | 0.8298 |
| Quadratic | 0.9804 |

# 5 Discussion

## 5.1 Comparison between one weld and two welds

The comparison between one weld and two welds gives a clear result especially with regard to the effective notch stress. The stress is reduced with 64-74% for the case with two welds mostly dependent of the ability of the two welds to handle the moment created in the weld region with an axial force couple. Something that is not possible in the case with one weld where the force is transferred from the face plate to the core by moment action.

The transverse shear stiffness doesn't show as a general behaviour for the different models as the stress-comparison did. However, it can be stated that the stiffness of the cross-section always increases with two welds but the magnitude of the increase differs from 10-154%. This differ is partly an effect of how much the cross-section would benefit from a higher degree of interaction between the face plate and the core. In the case with one weld the interaction between the top face plate and the corrugation is rather low and the face plate therefore has a low contribution to the transverse shear stiffness. When a second weld is added the interaction is increased, as could be seen in Figure 4.1 and Figure 4.2. The improved interaction enables for a larger contribution from the face plate to the transverse shear stiffness. If the cross-section has a low thickness of the corrugation, resulting in a weak core, the addition of a second weld will make a big influence on the transverse shear stiffness, like for model 1. If the core on the other hand is rather stiff from the beginning the additional stiffness from the face plate won't have as a big impact, like for model 3.

It can also be established that for one weld and two welds the point of maximum stress was changed. For one weld the position of the maximum stress is located in the root and for two welds the position of the maximum stress is located in the toe. The position of the maximum stress in the toe, for the two-weld case can be located both in the top face plate or the corrugation depending on the geometrical properties of the cross-section.

## 5.2 Fractional Factorial Design

The results obtained from the fractional factorial design are discussed below. The discussion leads to general conclusions for the studied parameters but it should be stated that the fractional factorial design was performed for 5 models and that all possible geometries has not been investigated.

### 5.2.1 Effective notch stress

From the main effect plots interesting results are obtained. For effective notch stress the four most influencing parameters are $t_c$, $f_2$, $d_w$ and $t_2$. The main effects also show that the stress will decrease with increasing $t_c$, as could be expected. However, an increase of $t_c$ might not always be the most efficient way to reduce the stress if the mass is taken into account. An increase of $t_c$ gives a high weight increase that should be considered in design, especially since a big increase of

weight would diminish the favourable high strength-to-weight ratio of the sandwich element.

For the parameter $f_2$, the stress will increase with increasing $f_2$. The reason behind this is that a larger $f_2$ will create a larger distance, from the normal force in the corrugation to the weld. This leads to larger load effects on the welds and therefore higher stresses. From the main effects plots it can also be seen that the influence of changing $f_2$ is larger for cross-sections with small θ.

For the parameter $d_w$, it is obvious that it has an impact on the stress but it shows a rather unpredictable behaviour. The stress is increased for both increasing and decreasing $d_w$ depending on the cross-section. An increase of the distance between the welds results in an increase of the moment capacity of the welds. This was expected to reduce the notch stress but the main effect plots show that this is not always the case. A possible explanation for the irregular behaviour and the phenomenon behind it has not been found in this thesis.

The parameter $t_2$ had both positive and negative slopes in the effect plots. $t_2$ influences the degree of constraint for the core. Increasing $t_2$ reduces the rotation at the joint between the bottom face and the core. Thus, the stresses at the joint between the top face and the core will increase. But since $t_2$ had both positive and negative slopes in the effects plots, the behaviour of $t_2$ is affected by something more. This parameter is also dependent on where the crack is located, if the crack is located in the face plate or in the corrugation. If the crack is located in the face plate, the stress is decreased with increased $t_2$. This is because an increased $t_2$, reduce the bending stress at the top of the notch. If the crack is located in the corrugation, the stress on the other hand is increased with increased $t_2$. This since the top plate provides a stronger constraint to the core. Moreover, it should also be noted that the difference between $t_2$ and $t_c$ influence of the position of the crack.

Other parameters with less influence on the notch stress has also been analysed to determine the causes behind their influence. From the main effect plots it is seen that an increase of the parameter $t_1$ increases the stress. This is because $t_1$ has the same influence on the restrain degree; an increasing $t_1$ increases the restrain degree for the core.

The angle, θ, has a rather small influence on the notch stress. If θ is large the core provides a stiff support to the top plate giving a increased stress in the top plate and decreased stress in the core. So the effects are also here dependent on the position of the crack for the cross-section. An important finding when adding the mass to the consideration is that θ increases its influence, most noticeably of all the parameters. The angle θ possesses the favourable capacity to be changed with a smaller effect on the total weight compared to e.g. changing a thickness.

The thickness of the weld, $t_w$, was seen to have a small influence on the notch stress for the 5 models tested. A possible explanation for this is that all the models crack in the top or the bottom of the notch, resulting in toe cracks. This parameter might show a different behaviour if the models would have cracked in

the root instead, then the thickness of the weld would have a more important influence of the stress.

For the height, h, the notch stress is increased if h is increased and the influence on the stress is small. Furthermore, if the radius of the corrugation, R, is increased the stress is increased. The same argumentation about the distance, from the normal force in the corrugation to the weld, as for $f_2$ applies for R. Last, the parameter $f_1$ reduces the notch stress if $f_1$ increased.

### 5.2.2 Transverse shear stiffness

The main effect plots for the transverse shear stiffness shows a more general behaviour and shows clearly which parameters that has the largest influence. However, it can be noted that model 1 shows a slight diverging influence of the parameters compared to the other models. This is most likely from the relatively big difference in thickness between top face plate and core for this model. Despite this fact the main effects for this model still follows the same overall behaviour.

For all the models it can be concluded that two parameters are important and has a large influence on the transverse shear stiffness. For the remaining parameters the influence is weak in comparison and therefore only the two important parameters will be discussed further.

The thickness of the corrugation, $t_c$, shows a strong influence. The transverse shear stiffness is increased when $t_c$ is increased. However as discussed above, an increase in $t_c$ results in an unfavourable increase of the weight of the cross-section.

The second parameter that shows a strong influence on the transverse shear stiffness is the angle θ. If the angle is increased the transverse shear stiffness is decreased. When adding the effect on the mass, the influence gets even stronger which makes the choice of θ crucial in design with regard to shear deformation.

## 5.3 Regression Analysis

The regression analysis was performed with both a linear model and a quadratic model. For the linear model the coefficient of determination, $R^2$, was determined to 75 % for the effective notch stress and 83% for the transverse shear stiffness. When the quadratic model was used the coefficient of determination was increased to 94% for the effective notch stress and 98% for the transverse shear stiffness. It is therefore considered that the quadratic model gives a better fit.

From the β-values it can be seen which parameters that has a large impact on the derived functions. For both the effective notch stress and the transverse shear stiffness, it is clear that $t_c$ has a large impact.

# 6 Conclusions

The influence of different geometrical properties on the fatigue life of a corrugated core sandwich element using Effective Notch Stress (ENS) method has been evaluated. The influence of the different parameters on the transverse shear stiffness has also been investigated. From the results of the analyses the following conclusions can be drawn:

- Using two welds instead of one reduces the effective notch stress with approximately 70%.
- For one weld the point of the maximum stress is located in the root and for two welds the point of the maximum stress is located in the toe.
- Using two welds instead of one increases the transverse shear stiffness of the cross-section. The stiffness increases with approximately 10-150% depending on the thickness of the core.
- The most important parameters influencing the effective notch stress and thereby the fatigue life, are $t_c$, $f_2$, $d_w$ and $t_2$. The stress is reduced, and therefore the fatigue life extended, by an increase of $t_c$ and an decrease of $f_2$. However, $d_w$ showed an unpredictable behaviour and no conclusions regarding its effect on the stress could be drawn.
- The most important parameters influencing the transverse shear stiffness of the cross-section are $\theta$ and $t_c$. The transverse shear stiffness is increased by an increase of $\theta$ and an increase of $t_c$. Since an increase of $t_c$ has a large impact on the mass off the cross-section an increase of $\theta$ is a more favourable choice for obtaining a higher stiffness.

## 6.1 Further studies within the field

There exist several topics that need to be discussed in further studies within this field. Some are listed below:

- This thesis discusses the influence of different parameters on the fatigue life and the transverse shear stiffness considering shear deformation as the load effect. Similar studies for all load effects needs to be conducted to get a complete picture of the state of stress in the weld region.
- Develop the parametric study further by investigating the parameter $d_w$ more deeply. Considering that $d_w$ is an important parameter influencing the effective notch stress, it would be of interest to try identifying the reasons behind its irregular behaviour.
- If more than two welds should be found necessary to use for the sandwich element in bridge applications a similar parametric study with additional welds would be of interest.
- Performing the regression analysis with a cubic model to might obtain a better fit. However, one should be aware of the risk with overfitting.

# 7 References


Abbot, S. P. et al., 2008. *Automated Laser Welded High Performance Steel Sandwich Bridge Deck Development.* Washington DC, Transportation Research Board.

Al-Emrani, M. & Aygül, M., 2014. *Fatigue Design of Steel and Composite Bridges,* Göteborg: Chalmers Univeristy of Technology.

Alwan, U. & Järve, D., 2012. *New Concept for Industrial Bridge Construction,* Göteborg: Chalmers University of Technology.

Beneus, E. & Koc, I., 2014. *Innovative road bridges with steel sandwich decks,* Göteborg: Chalmers Univeristy of Technology.

Bhaskar, K. & Varadan, T., 2013. *Plates Theories and Applications.* 1st ed. New Delhi: Ane Books Pvt. Ltd..

Blomqvist, U., 2010. *Matematisk statistik.* Göteborg: HB Matematiklitteratur.

Box, G. E., Hunter, J. S. & Hunter, W. G., 2005. *Statistics for Experimenters: Design, Innovation and Discovery.* 2nd ed. Hoboken, N.J.: Wiley-Interscience.

Bright, S. & Smith, J., 2007. A new design for steel bridge decks using laser fabrication. *The Structural Engineer*, 6 November, pp. 49-57.

Dackman, D. & Ek, W., 2015. *Steel sandwich decks in medium span bridges,* Göteborg: Chalmers University of Technology.

Fung, T.-C., Tan, K.-H. & Lok, T.-S., 1994. Elastic Constants for Z-core Sandwich Panels. *Journal of Structural Engineering,* 120(10), pp. 3046-3055.

Fung, T., Tan, K. & Lok, T., 1996. Shear Stiffness DQy For C-core Sandwich Panels. *Journal of Structural Engineering,* 122(8), pp. 958-966.

Hobbacher, A., 2013. *Recommendations for Fatigue Design of Welded Joints and Components,* s.l.: International Institute of Welding.

Kujala, P. & Klanac, A., 2005. Steel Sandwich Panels in Marine Applications. *Brodogradnja,* 56(4), pp. 305-314.

Libove, C. & Batdorf, S., 1948. *A General Small-Deflection Theory for Flat Sandwich Slabs,* Washington D.C.: National Advisory Committee for Aeronautics.

Libove, C. & Hubka, R. E., 1951. *Elastic Constants for Corrugated-Core Sandwich Plates,* Washington D.C: National Advisory Committee for Aeronautics.

Lok, T.-S. & Cheng, Q.-H., 2000. Elastic Stiffness Properties and behaviour of Truss-Core Sandwich Panel. *Journal of Structural Engineering,* Issue 126, pp. 552-559.

Nilsson, P., 2015. *Steel-sandwich elements in bridge applications,* Göteborg: Chalmers University of Technology.

Nordstrand, T., Carlsson, L. A. & Allen, H. G., 1994. Transverse shear stiffness of structural core sandwich. *Composite Structures,* Issue 27, pp. 317-329.

Ottosen, N. & Petersson, H., 1992. *Introduction to the Finite Element Method.* 1st ed. s.l.:Prentice Hall.

Palmkvist, A. & Sandberg, L., 2015. *Fatigue Analysis of Hybrid Laser Welds in Steel Sandwich Bridge Decks,* Göteborg: Chalmers Univeristy of Technology.

Radaj, D., Sonsino, C. M. & W, F., 2006. *Fatigue assessment of welded joints by local approaches.* 2nd ed. Cambridge: Woodhead Publishing Limited.

Romanoff, J., 2007. *Bending Response of Laser-Welded Web-Core Sandwich Plates,* Helsinki: Helsinki University of Technology.

SANDCORE, n.d. *Best Practice Guide for Sandwich Structures in Marine Applications,* s.l.: Prepared by the SAND.CORe Co-ordination Action on Advanced Sandwich Structures in the Transport Industries Under European Commission Contract No. FP6-506330.

Säynajäkangas, J. & Taulavouri, T., 2004. A review in design and manufacturing of stainless steel sandwich panels. *Stainless Steel World*, October, pp. 21-24.

Sykes, A. O., 1993. An Introduction to Regression Analysis. *Law & Economics Working Papers,* Volume 20, pp. 1-33.

Zenkert, D., 1995. *An Introduction to Sandwich Construction.* 2nd ed. Stockholm: Engineering Materials Advisory Services.

# Appendix A

## Python source code

Appendix A comprises the Python source code of the following scripts:

- MASTER.py
- INPUTS.py
- PART.py
- PROPERTY.py
- ASSEMBLY.py
- STEP.py
- INTERACTION.py
- LOAD.py
- MESH.py
- JOB.py
- OUTPUT.py

```
#-------------------------------------------------------------------------------
#   MASTER.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
from abaqus import *
from abaqusConstants import *
from caeModules import *

# CREATING GEOMETRY OF SANDWICH ELEMENT
session.journalOptions.setValues(replayGeometry=COORDINATE)

# OBTAINING INDATA FROM INPUTS.PY
execfile('C:\Parametric_study_TwoWelds_Fullfact\INPUTS.py')


ii=1

j=0
k=0
l=0
m=0
n=0
o=0
p=0
q=0
s=0

os.remove('C:/Parametric_study_TwoWelds_Fullfact/Area.txt')
Area=range(len(t1))
Modelname=range(len(t1))
for i in range(len(t1)):
    Modelname[i]=str(ii)
    ii=ii+1

    # CREATES A MODEL
    mdb.Model(name=Modelname[i], modelType=STANDARD_EXPLICIT)
    MyModel = mdb.models[Modelname[i]]

    # EXECUTE PART.PY
    execfile('C:\Parametric_study_TwoWelds_Fullfact\PART.py')

    # CREATRE.TXT-FILE CONTAINING THE AREA AND P FOR EACH MODEL
    Area[i]=range(2)
    Area[i][0]=MyPart.getMassProperties()['area']
    Area[i][1]=p_
    AreaFile=open('C:/Parametric_study_TwoWelds_Fullfact/Area.txt','a+')
    AreaFile.write('%10.20E\t%10.20E\n' %(Area[i][0],Area[i][1]))
    AreaFile.close()
```

```
    # EXECUTE PROPERTY.PY, ASSEMBLY.PY, STEP.PY, INTERACTION.PY, LOAD.PY, MESH.PY, JOB.PY
    execfile('C:\Parametric_study_TwoWelds_Fullfact\PROPERTY.py')
    execfile('C:\Parametric_study_TwoWelds_Fullfact\ASSEMBLY.py')
    execfile('C:\Parametric_study_TwoWelds_Fullfact\STEP.py')
    execfile('C:\Parametric_study_TwoWelds_Fullfact\INTERACTION.py')
    execfile('C:\Parametric_study_TwoWelds_Fullfact\LOAD.py')
    execfile('C:\Parametric_study_TwoWelds_Fullfact\MESH.py')
    execfile('C:\Parametric_study_TwoWelds_Fullfact\JOB.py')

    # SUBMITS AND RUN THE JOBS
    mdb.Job(name=Modelname[i],model=Modelname[i])
    mdb.jobs[Modelname[i]].submit()
    mdb.jobs[Modelname[i]].waitForCompletion()


    j=j+1
    k=k+1
    l=l+1
    m=m+1
    n=n+1
    o=o+1
    p=p+1
    q=q+1
    s=s+1
# EXECUTE OUTPUT.PY
execfile('C:\Parametric_study_TwoWelds_Fullfact\OUTPUT.py')
```

```
#-------------------------------------------------------------------------------
#   INPUTS.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE VALUES FOR THE PARAMETERS ARE DEFINED AND THE RIGHT
#   AMOUNT OF MODELS ARE CREATED WITH REGARD TO THE FACTORIAL DESIGN MATRIX.

ModelName = 'Sandwich Element'
PartName = 'Sandwich'
MaterialName ='Steel'
AssemblyName='Sandwich Assembly'

# DEFINE INPUTS
t1_=[0.005, 0.006, 0.007, 0.008, 0.009, 0.01, 0.011, 0.012]
tc_=[0.004, 0.005, 0.006, 0.007, 0.008]
t2_=[0.005, 0.006, 0.007, 0.008, 0.009, 0.01, 0.011, 0.012]
theta_=[40, 50, 60, 70, 80]
eta1_=[0.02]
eta2_=[0.02]
h_=[0.120, 0.140, 0.160, 0.180, 0.200, 0.220]
tw_=[0.003]
dw_=[0.03, 0.045, 0.060]

r=0.001

th=0.00001

E=210000000000.0
ny=0.3

E_plainstrain=E/(1.0-ny**2)

elementsize=0.00015
rf=0.00001

# READING FACTORIAL DESIGN MATRIX
fh = open( 'C:/Parametric_study_TwoWelds_Fullfact/fullfact.txt' )
factfrac = []
for line in fh.readlines():
    y = [value for value in line.split()]
    factfrac.append( y )
fh.close()
D_=[list(aa) for aa in zip(*factfrac)]

D=range(len(D_))
for I in range(len(D_)):
    D[I]=range(len(D_[0]))
    for J in range(len(D_[0])):
        D[I][J]=float(D_[I][J])
```

```
# CREATING VECTORS, FOR EACH PARAMETERS, CONTAINING VALUES FOR ALL MODELS
t1=range(len(D[0]))
tc=range(len(D[0]))
t2=range(len(D[0]))
theta=range(len(D[0]))
eta1=range(len(D[0]))
eta2=range(len(D[0]))
h=range(len(D[0]))
tw=range(len(D[0]))
dw=range(len(D[0]))
f1=range(len(D[0]))
f2=range(len(D[0]))

for jj in range(len(D[0])):
    t1[jj]=t1_[int(D[0][jj])]
    tc[jj]=tc_[int(D[1][jj])]
    t2[jj]=t2_[int(D[2][jj])]
    theta[jj]=theta_[int(D[3][jj])]
    eta1[jj]=eta1_[int(D[4][jj])]
    eta2[jj]=eta2_[int(D[5][jj])]
    h[jj]=h_[int(D[6][jj])]
    tw[jj]=tw_[int(D[7][jj])]
    dw[jj]=dw_[int(D[8][jj])]

    f1[jj]=dw[jj]+2*eta1[jj]
    f2[jj]=dw[jj]+2*eta2[jj]
```

```
#-------------------------------------------------------------------------------
#   PART.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE CROSS-SECTION OF THE UNIT CELL IS SKETCHED AND THE
#   GEOMETRY OF THE PART IS CREATED.

# CREATE A PART
MyPart= MyModel.Part(name=PartName, dimensionality=TWO_D_PLANAR, type=DEFORMABLE_BODY)

# CALCULATING MEASURES NEEDED TO SKETCH THE CROSS-SECTION
theta_r=theta[l]*pi/180
hc=h[o]-t1[i]/2-t2[k]/2-tc[j]

R=range(len(tc))
for jj in range(len(tc)):
    R[jj]=3*tc[jj]

R1=R[s]-tc[j]/2
R2=R[s]+tc[j]/2

a1=R1*sin(theta_r)
a=R[s]*sin(theta_r)
a2=R2*sin(theta_r)

b1=R1*(1-cos(theta_r))
b=R[s]*(1-cos(theta_r))
b2=R2*(1-cos(theta_r))

c1=(h[o]-t1[i]/2-t2[k]/2-tc[j]-b1-b2)
c=(h[o]-t1[i]/2-t2[k]/2-tc[j]-2*b)
c2=(h[o]-t1[i]/2-t2[k]/2-b2-tc[j]-b1)

d1=c1/tan(theta_r)
d=c/tan(theta_r)
d2=c2/tan(theta_r)

f=(r**2-(th/2)**2)**(0.5)

e1=f1[m]/2-dw[q]/2-tw[p]/2-r-f
e2=f2[n]/2-dw[q]/2-tw[p]/2-r-f

p_=(f1[m]+a+d+a+f2[n]+a+d+a)/2

# SKETCHING THE FACE PLATES
Specimen = MyModel.ConstrainedSketch(name='MySketch', sheetSize=200.0)

Specimen.Line(point1=(0.0, t1[i]/2+h[o]-t2[k]/2), point2=(f1[m]/2+a2+d2+a1+e2, t1[i]/2+h[o]-
t2[k]/2))
Specimen.Line(point1=(f1[m]/2+a2+d2+a1+f2[n]-e2, t1[i]/2+h[o]-t2[k]/2), point2=(f1[m]+a+d+a+
f2[n]+a+d+a, t1[i]/2+h[o]-t2[k]/2))
Specimen.Line(point1=(0.0, t1[i]/2+h[o]+t2[k]/2), point2=(f1[m]+a+d+a+f2[n]+a+d+a, t1[i]/2+h[
o]+t2[k]/2))
```

```
Specimen.Line(point1=(0.0, 0.0), point2=(f1[m]+a+d+a+f2[n]+a+d+a, 0.0))
Specimen.Line(point1=(f1[m]/2-e1, t1[i]), point2=(f1[m]/2+a+d+a+f2[n]+a+d+a+e1, t1[i]))

Specimen.Line(point1=(0.0, h[o]+t1[i]/2-t2[k]/2), point2=(0.0, h[o]+t1[i]/2+t2[k]/2))
Specimen.Line(point1=(f1[m]+a+d+a+f2[n]+a+d+a, h[o]+t1[i]/2-t2[k]/2), point2=(f1[m]+a+d+a+f2[
n]+a+d+a, h[o]+t1[i]/2+t2[k]/2))

# SKETCHING THE CORRUGATED-CORE
Specimen.Line(point1=(0.0, t1[i]+tc[j]+th), point2=(f1[m]/2, t1[i]+tc[j]+th))
Specimen.Line(point1=(f1[m]/2-e1, t1[i]+th), point2=(f1[m]/2, t1[i]+th))

# NOTCHES AT LEFT END
Specimen.ArcByCenterEnds(center=(f1[m]/2-e1-f, t1[i]+th/2), point1=(f1[m]/2-e1, t1[i]+th),
point2=(f1[m]/2-e1, t1[i]), direction=COUNTERCLOCKWISE)
Specimen.ArcByCenterEnds(center=(f1[m]/2-e1-f-r-tw[p]-r, t1[i]+th/2), point1=(f1[m]/2-e1-2*f-
tw[p]-2*f, t1[i]+th), point2=(f1[m]/2-e1-2*f-tw[p]-2*f, t1[i]), direction=CLOCKWISE)

# LINES AT LEFT END
Specimen.Line(point1=(0.0, t1[i]), point2=(f1[m]/2-e1-2*f-tw[p]-2*f, t1[i]))
Specimen.Line(point1=(0.0, t1[i]+th), point2=(f1[m]/2-e1-2*f-tw[p]-2*f, t1[i]+th))
Specimen.Line(point1=(0.0, 0.0), point2=(0.0, t1[i]))
Specimen.Line(point1=(0.0, t1[i]+th), point2=(0.0, t1[i]+th+tc[j]))

# CORRUGATION LEFT BOTTOM
Specimen.ArcByCenterEnds(center=(f1[m]/2, t1[i]+tc[j]+R1+th), point1=(f1[m]/2, t1[i]+tc[j]+th
), point2=(f1[m]/2+a1, t1[i]+tc[j]+b1+th), direction=COUNTERCLOCKWISE)
Specimen.ArcByCenterEnds(center=(f1[m]/2, t1[i]+R2+th), point1=(f1[m]/2, t1[i]+th), point2=(
f1[m]/2+a2, t1[i]+b2+th), direction=COUNTERCLOCKWISE)

# INCLINED LINES LEFT
Specimen.Line(point1=(f1[m]/2+a1, t1[i]+tc[j]+b1+th), point2=(f1[m]/2+a1+d1, t1[i]+tc[j]+b1+
c1-th))
Specimen.Line(point1=(f1[m]/2+a2, t1[i]+b2+th), point2=(f1[m]/2+a2+d2, t1[i]+b2+c2-th))

# CORRUGATION LEFT TOP
Specimen.ArcByCenterEnds(center=(f1[m]/2+a1+d1+a2, t1[i]+tc[j]+b1+c1+b2-R2-th), point1=(f1[m
]/2+a1+d1, t1[i]+tc[j]+b1+c1-th), point2=(f1[m]/2+a1+d1+a2, t1[i]+tc[j]+b1+c1+b2-th),
direction=CLOCKWISE)
Specimen.ArcByCenterEnds(center=(f1[m]/2+a2+d2+a1, t1[i]+b2+c2+b1-R1-th), point1=(f1[m]/2+a2+
d2, t1[i]+b2+c2-th), point2=(f1[m]/2+a2+d2+a1, t1[i]+b2+c2+b1-th), direction=CLOCKWISE)

# LINES FROM CORRUGATION TO OUTER NOTCHES IN MIDDLE (LEFT AND RIGHT RESPECTIVELY) AND LINE
FOR BOTTOM EDGE OF CORRUGATION
Specimen.Line(point1=(f1[m]/2+a2+d2+a1, t1[i]+b2+c2+b1+tc[j]-th), point2=(f1[m]/2+a2+d2+a1+e2
, t1[i]+b2+c2+b1+tc[j]-th))
Specimen.Line(point1=(f1[m]/2+a2+d2+a1+f2[n]-e2, t1[i]+b2+c2+b1+tc[j]-th), point2=(f1[m]/2+a2
+d2+a1+f2[n], t1[i]+b2+c2+b1+tc[j]-th))
Specimen.Line(point1=(f1[m]/2+a2+d2+a1, t1[i]+b2+c2+b1-th), point2=(f1[m]/2+a2+d2+a1+f2[n],
t1[i]+b2+c2+b1-th))

# OUTER TOP NOTCHES, LEFT AND RIGHT RESPECTIVELY
Specimen.ArcByCenterEnds(center=(f1[m]/2+a2+d2+a1+e2+f, t1[i]+b2+c2+b1+tc[j]-th/2), point1=(
f1[m]/2+a2+d2+a1+e2, t1[i]+b2+c2+b1+tc[j]-th), point2=(f1[m]/2+a2+d2+a1+e2, t1[i]+b2+c2+b1+tc
[j]), direction=COUNTERCLOCKWISE)
Specimen.ArcByCenterEnds(center=(f1[m]/2+a2+d2+a1+f2[n]-e2-f, t1[i]+b2+c2+b1+tc[j]-th/2),
point1=(f1[m]/2+a2+d2+a1+f2[n]-e2, t1[i]+b2+c2+b1+tc[j]-th), point2=(f1[m]/2+a2+d2+a1+f2[n]-
e2, t1[i]+b2+c2+b1+tc[j]), direction=CLOCKWISE)
```

```
# INNTER TOP NOTCHES, LEFT AND RIGHT RESPECTIVELY
Specimen.ArcByCenterEnds(center=(f1[m]/2+a2+d2+a1+e2+f+r+tw[p]+r, t1[i]+b2+c2+b1+tc[j]-th/2),
 point1=(f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f, t1[i]+b2+c2+b1+tc[j]-th), point2=(f1[m]/2+a2+d2+
a1+e2+2*f+tw[p]+2*f, t1[i]+b2+c2+b1+tc[j]), direction=CLOCKWISE)
Specimen.ArcByCenterEnds(center=(f1[m]/2+a2+d2+a1+f2[n]-e2-f-r-tw[p]-r, t1[i]+b2+c2+b1+tc[j]-
th/2), point1=(f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f, t1[i]+b2+c2+b1+tc[j]-th), point2=(f1[
m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f, t1[i]+b2+c2+b1+tc[j]), direction=COUNTERCLOCKWISE)

# LINES TO CLOSE INNER TOP NOTCHES
Specimen.Line(point1=(f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f, t1[i]+b2+c2+b1+tc[j]-th), point2=(f1
[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f, t1[i]+b2+c2+b1+tc[j]-th))
Specimen.Line(point1=(f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f, t1[i]+b2+c2+b1+tc[j]), point2=(f1[m
]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f, t1[i]+b2+c2+b1+tc[j]))

# CORRUGATION RIGHT TOP
Specimen.ArcByCenterEnds(center=(f1[m]/2+a1+d1+a2+f2[n], t1[i]+tc[j]+b1+c1+b2-R2-th), point1
=(f1[m]/2+a1+d1+a2+f2[n], t1[i]+tc[j]+b1+c1+b2-th), point2=(f1[m]/2+a1+d1+a2+f2[n]+a2, t1[i]+
tc[j]+b1+c1-th), direction=CLOCKWISE)
Specimen.ArcByCenterEnds(center=(f1[m]/2+a2+d2+a1+f2[n], t1[i]+b2+c2+b1-R1-th), point1=(f1[m
]/2+a2+d2+a1+f2[n], t1[i]+b2+c2+b1-th), point2=(f1[m]/2+a2+d2+a1+f2[n]+a1, t1[i]+b2+c2-th),
direction=CLOCKWISE)

# INCLINED LINES RIGHT
Specimen.Line(point1=(f1[m]/2+a1+d1+a2+f2[n]+a2, t1[i]+tc[j]+b1+c1-th), point2=(f1[m]/2+a1+d1
+a2+f2[n]+a2+d1, t1[i]+tc[j]+b1+th))
Specimen.Line(point1=(f1[m]/2+a2+d2+a1+f2[n]+a1, t1[i]+b2+c2-th), point2=(f1[m]/2+a2+d2+a1+f2
[n]+a1+d2, t1[i]+b2+th))

# CORRUGATION RIGHT BOTTOM
Specimen.ArcByCenterEnds(center=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1, t1[i]+tc[j]+R1+th), point1
=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1, t1[i]+tc[j]+b1+th), point2=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1,
t1[i]+tc[j]+th), direction=COUNTERCLOCKWISE)
Specimen.ArcByCenterEnds(center=(f1[m]/2+a2+d2+a1+f2[n]+a1+d2+a2, t1[i]+R2+th), point1=(f1[m
]/2+a2+d2+a1+f2[n]+a1+d2, t1[i]+b2+th), point2=(f1[m]/2+a2+d2+a1+f2[n]+a1+d2+a2, t1[i]+th),
direction=COUNTERCLOCKWISE)

# LINES AT RIGHT END
Specimen.Line(point1=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1, t1[i]+tc[j]+th), point2=(f1[m]+a1+d1+
a2+f2[n]+a2+d1+a1, t1[i]+tc[j]+th))
Specimen.Line(point1=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1, t1[i]+th), point2=(f1[m]/2+a1+d1+a2+f2
[n]+a2+d1+a1+e1, t1[i]+th))
Specimen.Line(point1=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+2*f+tw[p]+2*f, t1[i]), point2=(f1[m
]+a1+d1+a2+f2[n]+a2+d1+a1, t1[i]))
Specimen.Line(point1=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+2*f+tw[p]+2*f, t1[i]+th), point2=(f1
[m]+a1+d1+a2+f2[n]+a2+d1+a1, t1[i]+th))
Specimen.Line(point1=(f1[m]+a1+d1+a2+f2[n]+a2+d1+a1, 0.0), point2=(f1[m]+a1+d1+a2+f2[n]+a2+d1
+a1, t1[i]))
Specimen.Line(point1=(f1[m]+a1+d1+a2+f2[n]+a2+d1+a1, t1[i]+th), point2=(f1[m]+a1+d1+a2+f2[n]+
a2+d1+a1, t1[i]+th+tc[j]))
```

```
# NOTCHES AT RIGHT END
Specimen.ArcByCenterEnds(center=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f, t1[i]+th/2), point1=(
f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1, t1[i]+th), point2=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1, t1
[i]), direction=CLOCKWISE)
Specimen.ArcByCenterEnds(center=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f+r+tw[p]+r, t1[i]+th/2),
 point1=(f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+2*f+tw[p]+2*f, t1[i]+th), point2=(f1[m]/2+a1+d1+
a2+f2[n]+a2+d1+a1+e1+2*f+tw[p]+2*f, t1[i]), direction=COUNTERCLOCKWISE)

# CREATE THE PS-GEOMETRY
MyPart.BaseShell(sketch=Specimen)

# OBTAINING THE COORDINATES FOR THE CENTRE OF THE MASS
MassP=MyPart.getMassProperties()
MassX=MassP['areaCentroid'][0]
MassY=MassP['areaCentroid'][1]
```

```
#-------------------------------------------------------------------------------
#   PROPERTY.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE MATERIAL PROPERTIES AND MATERIAL ORIENTATION ARE
#   ASSIGNED TO THE CROSS-SECTION.

# MATERIAL PROPERTIES
MyMaterial=MyModel.Material(name=MaterialName)
MyModel.materials[MaterialName].Elastic(table=((E_plainstrain, ny), ))

# CREATING SET
faces = MyPart.faces.findAt(((f1[m]/4, t1[i]/4, 0.0), ))
MyPart.Set(faces=faces, name='Cross-section')

# CREATING SECTION
MyModel.HomogeneousSolidSection(name='Cross-section', material=MaterialName, thickness=None)

# ASSIGNING MATERIAL PROPERTIES TO CROSS-SECTION
region = MyPart.sets['Cross-section']
MyPart.SectionAssignment(region=region, sectionName='Cross-section', offset=0.0, offsetType=
MIDDLE_SURFACE, offsetField='', thicknessAssignment=FROM_SECTION)

# ASSIGNING MATERIAL ORIENTATION TO CROSS-SECTION
orientation=None
MyPart.MaterialOrientation(region=region, orientationType=GLOBAL, axis=AXIS_3,
additionalRotationType=ROTATION_NONE, localCsys=None, fieldName='', stackDirection=STACK_3)
```

```
#-------------------------------------------------------------------------------
#   ASSEMBLY.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE GEOMETRY OF THE ASSEMBLY FOR THE MODEL IS CREATED
#   BY CREATING AN INSTANCE FOR THE PART.

# MEASURES
r1=r+elementsize
r2=r+elementsize*2

f_1=(r1**2-(th/2)**2)**0.5
f_2=(r2**2-(th/2)**2)**0.5

# CREATING INDEPENDENT INSTANCE
MyAssembly = MyModel.rootAssembly
MyAssembly.Instance(name=AssemblyName, part=MyPart, dependent=OFF)

# PARTITION AROUND NOTCHES
Face = MyAssembly.instances[AssemblyName].faces
Transform = MyAssembly.MakeSketchTransform(sketchPlane=Face[0], sketchPlaneSide=SIDE1, origin
=(MassX, MassY, 0.0))

PartitionSketch = MyModel.ConstrainedSketch(name='Partition', sheetSize=485.37, gridSpacing=
12.13, transform=Transform)

MyAssembly.projectReferencesOntoSketch(sketch=PartitionSketch, filter=COPLANAR_EDGES)
PartitionSketch.sketchOptions.setValues(gridOrigin=(-MassX, -MassY))

# FIRST CIRCLE (NOTCHES FROM LEFT TO RIGHT)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2-e1-f, -MassY+t1[i]+th/2), point1=(-
MassX+f1[m]/2-e1+f_1-f, -MassY+t1[i]+th), point2=(-MassX+f1[m]/2-e1+f_1-f, -MassY+t1[i]),
direction=COUNTERCLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2-e1-f-r-tw[p]-r, -MassY+t1[i]+th/2),
point1=(-MassX+f1[m]/2-e1-2*f-tw[p]-f-f_1, -MassY+t1[i]+th), point2=(-MassX+f1[m]/2-e1-2*f-tw
[p]-f-f_1, -MassY+t1[i]), direction=CLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a2+d2+a1+e2+f+r+tw[p]+r, -MassY+t1[i]+
b2+c2+b1+tc[j]-th/2), point1=(-MassX+f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+f+f_1, -MassY+t1[i]+b2+c2+
b1+tc[j]-th), point2=(-MassX+f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+f+f_1, -MassY+t1[i]+b2+c2+b1+tc[j
]), direction=CLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2-f-r-tw[p]-r, -MassY+
t1[i]+b2+c2+b1+tc[j]-th/2), point1=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-f-f_1, -MassY+
t1[i]+b2+c2+b1+tc[j]-th), point2=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-f-f_1, -MassY+t1
[i]+b2+c2+b1+tc[j]), direction=COUNTERCLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a2+d2+a1+e2+f, -MassY+t1[i]+b2+c2+b1+
tc[j]-th/2), point1=(-MassX+f1[m]/2+a2+d2+a1+e2+f-f_1, -MassY+t1[i]+b2+c2+b1+tc[j]-th),
point2=(-MassX+f1[m]/2+a2+d2+a1+e2+f-f_1, -MassY+t1[i]+b2+c2+b1+tc[j]), direction=
COUNTERCLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2-f, -MassY+t1[i]+b2+
c2+b1+tc[j]-th/2), point1=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2+f_1-f, -MassY+t1[i]+b2+c2+b1+tc[j
]-th), point2=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2+f_1-f, -MassY+t1[i]+b2+c2+b1+tc[j]),
direction=CLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f, -MassY+
t1[i]+th/2), point1=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f-f_1, -MassY+t1[i]+th),
point2=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f-f_1, -MassY+t1[i]), direction=CLOCKWISE)
```

```
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f+r+tw[p]+r
, -MassY+t1[i]+th/2), point1=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+2*f+tw[p]+f+f_1, -
MassY+t1[i]+th), point2=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+2*f+tw[p]+f+f_1, -MassY+t1
[i]), direction=COUNTERCLOCKWISE)

# SECOND CIRCLE (NOTCHES FROM LEFT TO RIGHT)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2-e1-f, -MassY+t1[i]+th/2), point1=(-
MassX+f1[m]/2-e1+f_2-f, -MassY+t1[i]+th), point2=(-MassX+f1[m]/2-e1+f_2-f, -MassY+t1[i]),
direction=COUNTERCLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2-e1-f-r-tw[p]-r, -MassY+t1[i]+th/2),
point1=(-MassX+f1[m]/2-e1-2*f-tw[p]-f-f_2, -MassY+t1[i]+th), point2=(-MassX+f1[m]/2-e1-2*f-tw
[p]-f-f_2, -MassY+t1[i]), direction=CLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a2+d2+a1+e2+f+r+tw[p]+r, -MassY+t1[i]+
b2+c2+b1+tc[j]-th/2), point1=(-MassX+f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+f+f_2, -MassY+t1[i]+b2+c2+
b1+tc[j]-th), point2=(-MassX+f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+f+f_2, -MassY+t1[i]+b2+c2+b1+tc[j
]), direction=CLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2-f-r-tw[p]-r, -MassY+
t1[i]+b2+c2+b1+tc[j]-th/2), point1=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-f-f_2, -MassY+
t1[i]+b2+c2+b1+tc[j]-th), point2=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-f-f_2, -MassY+t1
[i]+b2+c2+b1+tc[j]), direction=COUNTERCLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a2+d2+a1+e2+f, -MassY+t1[i]+b2+c2+b1+
tc[j]-th/2), point1=(-MassX+f1[m]/2+a2+d2+a1+e2+f-f_2, -MassY+t1[i]+b2+c2+b1+tc[j]-th),
point2=(-MassX+f1[m]/2+a2+d2+a1+e2+f-f_2, -MassY+t1[i]+b2+c2+b1+tc[j]), direction=
COUNTERCLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2-f, -MassY+t1[i]+b2+
c2+b1+tc[j]-th/2), point1=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2+f_2-f, -MassY+t1[i]+b2+c2+b1+tc[j
]-th), point2=(-MassX+f1[m]/2+a2+d2+a1+f2[n]-e2+f_2-f, -MassY+t1[i]+b2+c2+b1+tc[j]),
direction=CLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f, -MassY+
t1[i]+th/2), point1=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f-f_2, -MassY+t1[i]+th),
point2=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f-f_2, -MassY+t1[i]), direction=CLOCKWISE)
PartitionSketch.ArcByCenterEnds(center=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f+r+tw[p]+r
, -MassY+t1[i]+th/2), point1=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+2*f+tw[p]+f+f_2, -
MassY+t1[i]+th), point2=(-MassX+f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+2*f+tw[p]+f+f_2, -MassY+t1
[i]), direction=COUNTERCLOCKWISE)

pickedFaces = Face.findAt(((f1[m]/2, t1[i]/4, 0.0), ))
MyAssembly.PartitionFaceBySketch(faces=pickedFaces, sketch=PartitionSketch)
```

```
#-------------------------------------------------------------------------------
#   STEP.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE OUTPUT REQUESTED FROM ANALYSIS ARE DEFINED.

# CREATING STEP
MyModel.StaticLinearPerturbationStep(name='Step-1', previous='Initial')

# SET FOR STRESSES
edge = MyAssembly.instances[AssemblyName].edges
datum = MyAssembly.datums

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2+f+r, t1[i]+b2+c2+b1+tc[j]-th/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Stresses_1')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2+f+r+tw[p], t1[i]+b2+c2+b1+tc[j]-th/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Stresses_2')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+f2[n]-e2-f-r-tw[p], t1[i]+b2+c2+b1+tc[j]-th/2, 0.0),
))
MyAssembly.Set(edges=edges1, name='Stresses_3')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+f2[n]-e2-f-r, t1[i]+b2+c2+b1+tc[j]-th/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Stresses_4')

# CREATING REFERENCE POINT FOR CONSTRAINTS
MyAssembly.ReferencePoint(point=(-rf, t1[i]/2, 0.0))
MyAssembly.ReferencePoint(point=(-rf, t1[i]+th+tc[j]/2, 0.0))
MyAssembly.ReferencePoint(point=(-rf, t1[i]/2+h[o], 0.0))
MyAssembly.ReferencePoint(point=(f1[m]+a+d+a+f2[n]+a+d+a+rf, t1[i]/2, 0.0))
MyAssembly.ReferencePoint(point=(f1[m]+a+d+a+f2[n]+a+d+a+rf, t1[i]+th+tc[j]/2, 0.0))
MyAssembly.ReferencePoint(point=(f1[m]+a+d+a+f2[n]+a+d+a+rf, t1[i]/2+h[o], 0.0))

# SET FOR REACTION FORCES
ref = MyAssembly.referencePoints
refPoints1=(ref[8], )
MyAssembly.Set(referencePoints=refPoints1, name='Reaction_Forces_Bottom_Left')
refPoints1=(ref[9], )
MyAssembly.Set(referencePoints=refPoints1, name='Reaction_Forces_Bottom_Left_Core')
refPoints1=(ref[10], )
MyAssembly.Set(referencePoints=refPoints1, name='Reaction_Forces_Top_Left')
refPoints1=(ref[11], )
MyAssembly.Set(referencePoints=refPoints1, name='Reaction_Forces_Bottom_Right')
refPoints1=(ref[12], )
MyAssembly.Set(referencePoints=refPoints1, name='Reaction_Forces_Bottom_Right_Core')
refPoints1=(ref[13], )
MyAssembly.Set(referencePoints=refPoints1, name='Reaction_Forces_Top_Right')

# SET FOR DISPLACEMENTS
refPoints1=(ref[10], )
MyAssembly.Set(referencePoints=refPoints1, name='Displacements_Left')
refPoints1=(ref[13], )
MyAssembly.Set(referencePoints=refPoints1, name='Displacements_Right')
```

```
# DATUM POINT FOR FREE BODY CUT
size=0.0015

MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i
]/2+t2[k]/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i
]/2-t2[k]/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i
]/2-t2[k]/2-th, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i
]/2-t2[k]/2-th-tc[j], 0.0))

MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+
size/3, h[o]+t1[i]/2+t2[k]/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+
size/3, h[o]+t1[i]/2-t2[k]/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+
size/3, h[o]+t1[i]/2-t2[k]/2-th, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+
size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j], 0.0))

MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*
elementsize-size/3, h[o]+t1[i]/2+t2[k]/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*
elementsize-size/3, h[o]+t1[i]/2-t2[k]/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*
elementsize-size/3, h[o]+t1[i]/2-t2[k]/2-th, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*
elementsize-size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j], 0.0))

MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o
]+t1[i]/2+t2[k]/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o
]+t1[i]/2-t2[k]/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o
]+t1[i]/2-t2[k]/2-th, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o
]+t1[i]/2-t2[k]/2-th-tc[j], 0.0))

MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2+f+r, t1[i]+b+c+b+tc[j]-th/2,
0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2+f+r+tw[p], t1[i]+b+c+b+tc[j]-th
/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2+f+r+dw[q], t1[i]+b+c+b+tc[j]-th
/2, 0.0))
MyAssembly.DatumPointByCoordinate(coords=(f1[m]/2+a2+d2+a1+e2+f+r+dw[q]+tw[p], t1[i]+b+c+b+tc
[j]-th/2, 0.0))

# PARTITION FOR FREE BODY CUT
pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+f2[n]/2, h[o]+t1[i]/2-t2[k]/4, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[22], point2=datum[23], faces=pickedFaces)

pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+f2[n]/2, h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0),
))
MyAssembly.PartitionFaceByShortestPath(point1=datum[24], point2=datum[25], faces=pickedFaces)

pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+f2[n], h[o]+t1[i]/2-t2[k]/4, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[26], point2=datum[27], faces=pickedFaces)
```

```
pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+f2[n], h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[28], point2=datum[29], faces=pickedFaces)

pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+f2[n], h[o]+t1[i]/2-t2[k]/4, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[30], point2=datum[31], faces=pickedFaces)

pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+f2[n], h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[32], point2=datum[33], faces=pickedFaces)

pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+f2[n], h[o]+t1[i]/2-t2[k]/4, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[34], point2=datum[35], faces=pickedFaces)

pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+f2[n], h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[36], point2=datum[37], faces=pickedFaces)

pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+elementsize/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+(elementsize+elementsize/2), t1[i]+b+c+b+tc[j]-th/2, 0.0), ), ((f1
 [m]/2+a2+d2+a1+e2+f+r+tw[p]-elementsize/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]-(elementsize+elementsize/2), t1[i]+b+c+b+tc[j]-th/2, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[38], point2=datum[39], faces=pickedFaces)

pickedFaces = Face.findAt(((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q], t1[i]+b+c+b+tc[j]-th/2,
0.0), ), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]-tw[p]/2+elementsize/2, t1[i]+b+c+b+tc[j]-th/
2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]-tw[p]/2+(elementsize+elementsize/2), t1[i]+b+c+b+tc[
 j]-th/2, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]+tw[p]/2-elementsize/2, t1[i]+b+c+b
 +tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]+tw[p]/2-(elementsize+elementsize/2), t1[i]+b+c+b+tc[
 j]-th/2, 0.0), ))
MyAssembly.PartitionFaceByShortestPath(point1=datum[40], point2=datum[41], faces=pickedFaces)

# CREATING SET FOR FREE BODY CUT
edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i]/2-t2[k]/4, 0.0),
), ((f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/4, 0.0), ),  ((
 f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0
 ), ),
 ((f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*elementsize-size/3, h[o]+t1[i]/2-t2[k]/4, 0.0),
 ),  ((f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*elementsize-size/3, h[o]+t1[i]/2-t2[k]/2-th-
 tc[j]/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/4, 0.0), ), ((f1[m]/2+
 a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0), ), ((f1[m]/2+
 a2+d2+a1+e2+f+r+tw[p]/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f+r+
 elementsize/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+(elementsize+elementsize/2), t1[i]+b+c+b+tc[j]-th/2, 0.0), ), ((f1
 [m]/2+a2+d2+a1+e2+f+r+tw[p]-elementsize/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]-(elementsize+elementsize/2), t1[i]+b+c+b+tc[j]-th/2, 0.0),
 ), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q], t1[i]+b+c+b+tc[j]-th/2, 0.0), ), ((f1[m]/2+a2+d2
 +a1+e2+f+r+tw[p]/2+dw[q]-tw[p]/2+elementsize/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]-tw[p]/2+(elementsize+elementsize/2), t1[i]+b+c+b+tc[
 j]-th/2, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]+tw[p]/2-elementsize/2, t1[i]+b+c+b
 +tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]+tw[p]/2-(elementsize+elementsize/2), t1[i]+b+c+b+tc[
 j]-th/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Free Body Cut')
```

```
edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i]/2-t2[k]/4, 0.0),
))
MyAssembly.Set(edges=edges1, name='FBC_1_Plate')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j
]/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='FBC_1_Core')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+size/3, h[o]+t1[i]/2-
t2[k]/4, 0.0), ))
MyAssembly.Set(edges=edges1, name='FBC_2_Plate')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+size/3, h[o]+t1[i]/2-
t2[k]/2-th-tc[j]/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='FBC_2_Core')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*elementsize-size/3, h[o]+t1[
i]/2-t2[k]/4, 0.0), ))
MyAssembly.Set(edges=edges1, name='FBC_3_Plate')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*elementsize-size/3, h[o]+t1[
i]/2-t2[k]/2-th-tc[j]/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='FBC_3_Core')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/4,
0.0), ))
MyAssembly.Set(edges=edges1, name='FBC_4_Plate')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/2-
th-tc[j]/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='FBC_4_Core')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ), ((f1
[m]/2+a2+d2+a1+e2+f+r+elementsize/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+(elementsize+elementsize/2), t1[i]+b+c+b+tc[j]-th/2, 0.0), ), ((f1
 [m]/2+a2+d2+a1+e2+f+r+tw[p]-elementsize/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]-(elementsize+elementsize/2), t1[i]+b+c+b+tc[j]-th/2, 0.0),
 ))
MyAssembly.Set(edges=edges1, name='FBC_1_Weld')

edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q], t1[i]+b+c+b+tc[j]-th/2, 0.0),
), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]-tw[p]/2+elementsize/2, t1[i]+b+c+b+tc[j]-th/2, 0.0
), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]-tw[p]/2+(elementsize+elementsize/2), t1[i]+b+c+b+tc[
 j]-th/2, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]+tw[p]/2-elementsize/2, t1[i]+b+c+b
 +tc[j]-th/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q]+tw[p]/2-(elementsize+elementsize/2), t1[i]+b+c+b+tc[
 j]-th/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='FBC_2_Weld')
```

```
# DEFINING FIELD OUTPUT
MyModel.fieldOutputRequests['F-Output-1'].suppress()

regionDef=MyModel.rootAssembly.sets['Stresses_1']
MyModel.FieldOutputRequest(name='Stresses_1', createStepName='Step-1', variables=('S', ),
region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Stresses_2']
MyModel.FieldOutputRequest(name='Stresses_2', createStepName='Step-1', variables=('S', ),
region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Stresses_3']
MyModel.FieldOutputRequest(name='Stresses_3', createStepName='Step-1', variables=('S', ),
region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Stresses_4']
MyModel.FieldOutputRequest(name='Stresses_4', createStepName='Step-1', variables=('S', ),
region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)

regionDef=MyModel.rootAssembly.sets['Reaction_Forces_Bottom_Left']
MyModel.FieldOutputRequest(name='Reaction_Forces_Bottom_Left', createStepName='Step-1',
variables=('RF', ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Reaction_Forces_Bottom_Left_Core']
MyModel.FieldOutputRequest(name='Reaction_Forces_Bottom_Left_Core', createStepName='Step-1',
variables=('RF', ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Reaction_Forces_Top_Left']
MyModel.FieldOutputRequest(name='Reaction_Forces_Top_Left', createStepName='Step-1',
variables=('RF', ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Reaction_Forces_Bottom_Right']
MyModel.FieldOutputRequest(name='Reaction_Forces_Bottom_Right', createStepName='Step-1',
variables=('RF', ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Reaction_Forces_Bottom_Right_Core']
MyModel.FieldOutputRequest(name='Reaction_Forces_Bottom_Right_Core', createStepName='Step-1',
 variables=('RF', ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Reaction_Forces_Top_Right']
MyModel.FieldOutputRequest(name='Reaction_Forces_Top_Right', createStepName='Step-1',
variables=('RF', ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)

regionDef=MyModel.rootAssembly.sets['Displacements_Left']
MyModel.FieldOutputRequest(name='Displacements_Left', createStepName='Step-1', variables=('U'
, ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
regionDef=MyModel.rootAssembly.sets['Displacements_Right']
MyModel.FieldOutputRequest(name='Displacements_Right', createStepName='Step-1', variables=(
'U', ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)

regionDef=MyModel.rootAssembly.sets['Free Body Cut']
MyModel.FieldOutputRequest(name='Free Body Cut', createStepName='Step-1', variables=('NFORC',
 ), region=regionDef, sectionPoints=DEFAULT, rebar=EXCLUDE)
```

```
#-------------------------------------------------------------------------------
#   INTERACTION.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE INTERACTION BETWEEN THE NODES AND THE REFERENCE-
#   POINTS ARE CREATED.

# SET FOR SLAVE NODES
edges1 = edge.findAt(((0.0, t1[i]/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Slave Nodes1')

edges1 = edge.findAt(((0.0, t1[i]+th+tc[j]/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Slave Nodes2')

edges1 = edge.findAt(((0.0, t1[i]/2+h[o], 0.0), ))
MyAssembly.Set(edges=edges1, name='Slave Nodes3')

edges1 = edge.findAt(((f1[m]+a+d+a+f2[n]+a+d+a, t1[i]/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Slave Nodes4')

edges1 = edge.findAt(((f1[m]+a+d+a+f2[n]+a+d+a, t1[i]+th+tc[j]/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Slave Nodes5')

edges1 = edge.findAt(((f1[m]+a+d+a+f2[n]+a+d+a, t1[i]/2+h[o], 0.0), ))
MyAssembly.Set(edges=edges1, name='Slave Nodes6')

# SET FOR MASTER NODES
refPoints1=(MyAssembly.referencePoints[8], )
MyAssembly.Set(referencePoints=refPoints1, name='Master Node1')
refPoints2=(MyAssembly.referencePoints[9], )
MyAssembly.Set(referencePoints=refPoints2, name='Master Node2')
refPoints3=(MyAssembly.referencePoints[10], )
MyAssembly.Set(referencePoints=refPoints3, name='Master Node3')
refPoints4=(MyAssembly.referencePoints[11], )
MyAssembly.Set(referencePoints=refPoints4, name='Master Node4')
refPoints4=(MyAssembly.referencePoints[12], )
MyAssembly.Set(referencePoints=refPoints4, name='Master Node5')
refPoints4=(MyAssembly.referencePoints[13], )
MyAssembly.Set(referencePoints=refPoints4, name='Master Node6')

# CREATING COUPLING BETWEEN MASTER NODES AND SLAVE NODES
MyModel.Coupling(name='Left_Bottom_Face', controlPoint=MyAssembly.sets['Master Node1'],
surface=MyAssembly.sets['Slave Nodes1'], influenceRadius=WHOLE_SURFACE, couplingType=
KINEMATIC, localCsys=None, u1=ON, u2=ON, ur3=ON)
MyModel.Coupling(name='Left_Bottom_Core', controlPoint=MyAssembly.sets['Master Node2'],
surface=MyAssembly.sets['Slave Nodes2'], influenceRadius=WHOLE_SURFACE, couplingType=
KINEMATIC, localCsys=None, u1=ON, u2=ON, ur3=ON)
MyModel.Coupling(name='Left_Top_Face', controlPoint=MyAssembly.sets['Master Node3'], surface=
MyAssembly.sets['Slave Nodes3'], influenceRadius=WHOLE_SURFACE, couplingType=KINEMATIC,
localCsys=None, u1=ON, u2=ON, ur3=ON)
MyModel.Coupling(name='Right_Bottom_Face', controlPoint=MyAssembly.sets['Master Node4'],
surface=MyAssembly.sets['Slave Nodes4'], influenceRadius=WHOLE_SURFACE, couplingType=
KINEMATIC, localCsys=None, u1=ON, u2=ON, ur3=ON)
MyModel.Coupling(name='Right_Bottom_Core', controlPoint=MyAssembly.sets['Master Node5'],
surface=MyAssembly.sets['Slave Nodes5'], influenceRadius=WHOLE_SURFACE, couplingType=
KINEMATIC, localCsys=None, u1=ON, u2=ON, ur3=ON)
```

```
MyModel.Coupling(name='Right_Top_Face', controlPoint=MyAssembly.sets['Master Node6'], surface
=MyAssembly.sets['Slave Nodes6'], influenceRadius=WHOLE_SURFACE, couplingType=KINEMATIC,
localCsys=None, u1=ON, u2=ON, ur3=ON)
```

```
#-----------------------------------------------------------------------------
#   LOAD.PY
#   By: Lovisa Persson
#-----------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE LOAD AND BOUNDARY CONDITIONS ARE APPLIED.

# LOAD
MyModel.ConcentratedForce(name='Unit Load - Left', createStepName='Step-1', region=MyAssembly
.sets['Master Node3'], cf1=1.0, distributionType=UNIFORM, field='', localCsys=None)
MyModel.ConcentratedForce(name='Unit Load - Right', createStepName='Step-1', region=
MyAssembly.sets['Master Node6'], cf1=1.0, distributionType=UNIFORM, field='', localCsys=None)

# BOUNDARY CONDITIONS
MyModel.DisplacementBC(name='Left Bottom RP Face', createStepName='Initial', region=
MyAssembly.sets['Master Node1'], u1=SET, u2=SET, ur3=UNSET, amplitude=UNSET, distributionType
=UNIFORM, fieldName='', localCsys=None)
MyModel.DisplacementBC(name='Left Bottom RP Core', createStepName='Initial', region=
MyAssembly.sets['Master Node2'], u1=UNSET, u2=SET, ur3=UNSET, amplitude=UNSET,
distributionType=UNIFORM, fieldName='', localCsys=None)
MyModel.DisplacementBC(name='Left Top RP Face', createStepName='Initial', region=MyAssembly.
sets['Master Node3'], u1=UNSET, u2=SET, ur3=UNSET, amplitude=UNSET, distributionType=UNIFORM,
 fieldName='', localCsys=None)
MyModel.DisplacementBC(name='Right Bottom RP Face', createStepName='Initial', region=
MyAssembly.sets['Master Node4'], u1=SET, u2=SET, ur3=UNSET, amplitude=UNSET, distributionType
=UNIFORM, fieldName='', localCsys=None)
MyModel.DisplacementBC(name='Right Bottom RP Core', createStepName='Initial', region=
MyAssembly.sets['Master Node5'], u1=UNSET, u2=SET, ur3=UNSET, amplitude=UNSET,
distributionType=UNIFORM, fieldName='', localCsys=None)
MyModel.DisplacementBC(name='Right Top RP Face', createStepName='Initial', region=MyAssembly.
sets['Master Node6'], u1=UNSET, u2=SET, ur3=UNSET, amplitude=UNSET, distributionType=UNIFORM,
 fieldName='', localCsys=None)
```

```
#-------------------------------------------------------------------------------
#   MESH.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE MESH FOR THE INSTANCE ARE GENERATED.

# SEED PART INSTANCE
partInstances =(MyAssembly.instances[AssemblyName], )
MyAssembly.seedPartInstance(regions=partInstances, size=0.001, deviationFactor=0.1,
minSizeFactor=0.1)

# CREATING SET FOR CIRCLES AROUND NOTCH
edge = MyAssembly.instances[AssemblyName].edges
edges1 = edge.findAt(((f1[m]/2-e1-f-r, t1[i]+th/2, 0.0), ), ((f1[m]/2-e1-f-r1, t1[i]+th/2,
0.0), ), ((f1[m]/2-e1-f-r2, t1[i]+th/2, 0.0), ),
    ((f1[m]/2-e1-f-r-tw[p], t1[i]+th/2, 0.0), ), ((f1[m]/2-e1-f-r-tw[p]-r+r1, t1[i]+th/2, 0.0
    ), ), ((f1[m]/2-e1-f-r-tw[p]-r+r2, t1[i]+th/2, 0.0), ),
    ((f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f+r, t1[i]+th/2, 0.0), ), ((f1[m]/2+a1+d1+a2+f2[n]+
    a2+d1+a1+e1+f+r1, t1[i]+th/2, 0.0), ), ((f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f+r2, t1[i]+
    th/2, 0.0), ),
    ((f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+e1+f+r+tw[p], t1[i]+th/2, 0.0), ), ((f1[m]/2+a1+d1+a2+
    f2[n]+a2+d1+a1+e1+f+r+tw[p]+r-r1, t1[i]+th/2, 0.0), ), ((f1[m]/2+a1+d1+a2+f2[n]+a2+d1+a1+
    e1+f+r+tw[p]+r-r2, t1[i]+th/2, 0.0), ),
    ((f1[m]/2+a2+d2+a1+e2+f, t1[i]/2+h[o]-t2[k]/2-th/2+r, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f,
    t1[i]/2+h[o]-t2[k]/2-th/2-r, 0.0), ),((f1[m]/2+a2+d2+a1+e2+f, t1[i]/2+h[o]-t2[k]/2-th/2+
    r1, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f, t1[i]/2+h[o]-t2[k]/2-th/2-r1, 0.0), ),((f1[m]/2+a2+
    d2+a1+e2+f, t1[i]/2+h[o]-t2[k]/2-th/2+r2, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f, t1[i]/2+h[o]-
    t2[k]/2-th/2-r2, 0.0), ),
    ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]+r, t1[i]/2+h[o]-t2[k]/2-th/2+r, 0.0), ), ((f1[m]/2+a2+d2+
    a1+e2+f+r+tw[p]+r, t1[i]/2+h[o]-t2[k]/2-th/2-r, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]+
    r, t1[i]/2+h[o]-t2[k]/2-th/2+r1, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]+r, t1[i]/2+h[o
    ]-t2[k]/2-th/2-r1, 0.0), ),((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]+r, t1[i]/2+h[o]-t2[k]/2-th/2+
    r2, 0.0), ), ((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]+r, t1[i]/2+h[o]-t2[k]/2-th/2-r2, 0.0), ),
    ((f1[m]/2+a2+d2+a1+f2[n]-e2-f-r-tw[p]-r, t1[i]/2+h[o]-t2[k]/2-th/2+r, 0.0), ), ((f1[m]/2+
    a2+d2+a1+f2[n]-e2-f-r-tw[p]-r, t1[i]/2+h[o]-t2[k]/2-th/2-r, 0.0), ), ((f1[m]/2+a2+d2+a1+
    f2[n]-e2-f-r-tw[p]-r, t1[i]/2+h[o]-t2[k]/2-th/2+r1, 0.0), ), ((f1[m]/2+a2+d2+a1+f2[n]-e2-
    f-r-tw[p]-r, t1[i]/2+h[o]-t2[k]/2-th/2-r1, 0.0), ), ((f1[m]/2+a2+d2+a1+f2[n]-e2-f-r-tw[p
    ]-r, t1[i]/2+h[o]-t2[k]/2-th/2+r2, 0.0), ), ((f1[m]/2+a2+d2+a1+f2[n]-e2-f-r-tw[p]-r, t1[i
    ]/2+h[o]-t2[k]/2-th/2-r2, 0.0), ),
    ((f1[m]/2+a2+d2+a1+f2[n]-e2-f, t1[i]/2+h[o]-t2[k]/2-th/2+r, 0.0), ),((f1[m]/2+a2+d2+a1+f2
    [n]-e2-f, t1[i]/2+h[o]-t2[k]/2-th/2-r, 0.0), ), ((f1[m]/2+a2+d2+a1+f2[n]-e2-f, t1[i]/2+h[
    o]-t2[k]/2-th/2+r1, 0.0), ), ((f1[m]/2+a2+d2+a1+f2[n]-e2-f, t1[i]/2+h[o]-t2[k]/2-th/2-r1,
     0.0), ), ((f1[m]/2+a2+d2+a1+f2[n]-e2-f, t1[i]/2+h[o]-t2[k]/2-th/2+r2, 0.0), ), ((f1[m]/2
    +a2+d2+a1+f2[n]-e2-f, t1[i]/2+h[o]-t2[k]/2-th/2-r2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Circles')

# SEED THE EDGES OF THE CIRCLES AROUND NOTCH
MyAssembly.seedEdgeBySize(edges=edges1, size=elementsize, deviationFactor=0.1, minSizeFactor=
0.1, constraint=FINER)
```

```
# CREATING SET FOR INTERMEDIATE MESH FOR "BOX-PARTITION" AS A CAUSE FROM PARTITION FOR FBC
edges1 = edge.findAt(((f1[m]/2+a2+d2+a1+e2-3*elementsize-size/3, h[o]+t1[i]/2-t2[k]/4, 0.0),
), ((f1[m]/2+a2+d2+a1+e2-3*elementsize-size/6, h[o]+t1[i]/2-t2[k]/2, 0.0), ), ((f1[m]/2+a2+d2
+a1+e2-3*elementsize-size/6, h[o]+t1[i]/2-t2[k]/2-th, 0.0), ),((f1[m]/2+a2+d2+a1+e2-3*
elementsize-size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/4, 0.0), ), ((
 f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+size/6, h[o]+t1[i]/2-t2[k]/2, 0.0), ), ((f1[
 m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+size/6, h[o]+t1[i]/2-t2[k]/2-th, 0.0), ),  ((f1
 [m]/2+a2+d2+a1+e2+2*f+tw[p]+2*f+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0),
  ),
 ((f1[m]/2+a2+d2+a1+e2+2*f+tw[p]/2, h[o]+t1[i]/2+t2[k]/2, 0.0), ),  ((f1[m]/2+a2+d2+a1+e2+2*f
 +tw[p]/2, h[o]+t1[i]/2-t2[k]/2-th-tc[j], 0.0), ),
 ((f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*elementsize-size/3, h[o]+t1[i]/2-t2[k]/4, 0.0),
 ),((f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*elementsize-size/6, h[o]+t1[i]/2-t2[k]/2, 0.0
 ), ),  ((f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*elementsize-size/6, h[o]+t1[i]/2-t2[k]/2-
 th, 0.0), ),((f1[m]/2+a2+d2+a1+f2[n]-e2-2*f-tw[p]-2*f-3*elementsize-size/3, h[o]+t1[i]/2-t2[
 k]/2-th-tc[j]/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+f2[n]-e2+3*elementsize+size/3, h[o]+t1[i]/2-t2[k]/4, 0.0), ), ((f1[m]/2+
 a2+d2+a1+f2[n]-e2+3*elementsize+size/6, h[o]+t1[i]/2-t2[k]/2, 0.0), ), ((f1[m]/2+a2+d2+a1+f2
 [n]-e2+3*elementsize+size/6, h[o]+t1[i]/2-t2[k]/2-th, 0.0), ), ((f1[m]/2+a2+d2+a1+f2[n]-e2+3
 *elementsize+size/3, h[o]+t1[i]/2-t2[k]/2-th-tc[j]/2, 0.0), ),
 ((f1[m]/2+a2+d2+a1+e2+2*f+tw[p]+dw[q]+tw[p]/2, h[o]+t1[i]/2+t2[k]/2, 0.0), ),  ((f1[m]/2+a2+
 d2+a1+e2+2*f+tw[p]+dw[q]+tw[p]/2, h[o]+t1[i]/2-t2[k]/2-th-tc[j], 0.0), ), ((f1[m]/2+a2+d2+a1
 +e2+f+r+tw[p]/2, t1[i]+b+c+b+tc[j]-th/2, 0.0), ),((f1[m]/2+a2+d2+a1+e2+f+r+tw[p]/2+dw[q], t1
 [i]+b+c+b+tc[j]-th/2, 0.0), ))
MyAssembly.Set(edges=edges1, name='Mesh')

# SEED EDGES OF "BOX-PARTITION"
MyAssembly.seedEdgeBySize(edges=edges1, size=0.00055, deviationFactor=0.1, minSizeFactor=0.1,
 constraint=FINER)

# GENERATES MESH
partInstances =(MyAssembly.instances[AssemblyName], )
MyAssembly.generateMesh(regions=partInstances)
```

```
#-------------------------------------------------------------------------------
#   JOB.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE JOB IS CREATED.

# CREATING JOB
mdb.Job(name=Modelname[i], model=Modelname[i])
mdb.jobs[Modelname[i]].setValues(description='', memoryUnits=PERCENTAGE, memory=50,
getMemoryFromAnalysis=True)

mdb.customData.jobName = Modelname[i]
mdb.customData.jobModel = Modelname[i]
mdb.jobs[Modelname[i]].writeInput()
```

```
#-------------------------------------------------------------------------------
#   OUTPUT.PY
#   By: Lovisa Persson
#-------------------------------------------------------------------------------
#   THE SCRIPT IS CONNECTED TO "MASTER.PY" AND CANNOT BE RUN BY IT SELF.
#   IN THIS SCRIPT THE RELEVANT RESULTS ARE EXTRACTED AND SAVED IN .TXT-FILES.

from abaqus import *
from abaqusConstants import *
from caeModules import *
session.journalOptions.setValues(replayGeometry=COORDINATE)

# OBTAINING INDATA FROM INPUTS.PY
execfile('C:\Parametric_study_TwoWelds_Fullfact\INPUTS.py')

ii=1

j=0
k=0
l=0
m=0
n=0
o=0
p=0
q=0
s=0

# ANGLE (HALF OF THE TOTAL ANGLE) FOR PART OF NOTCH THAT IS LOCATED IN THE OPENING BETWEEN
THE FACEPLATE AND THE CORE
angle1=asin((th/2)/r)

# NUMBER OF POINTS FOR PATH AROUND THE NOTCH
numb_points=int(round((2*pi*r-angle1*2*r)/elementsize))

# ANGLE BETWEEN EACH POINT ON THE PATH
angle2=(2*pi-2*angle1)/numb_points

Modelname=range(len(t1))
Output=range(len(t1))
Indata=range(len(t1))
for i in range(len(t1)):

    # MODELNAME
    Modelname[i]=str(ii)
    ii=ii+1

    # OPEN THE ODB-FILE
    odb = session.openOdb('C:/BRIGADE Plus Work Directory/'+Modelname[i]+'.odb')
    odb = session.odbs['C:/BRIGADE Plus Work Directory/'+Modelname[i]+'.odb']

    session.viewports['Viewport: 1'].odbDisplay.basicOptions.setValues(useRegionBoundaries=
    False, includeFeatureBoundaries=False, averagingThreshold=100)
```

```
# MEASURES
theta_r=theta[l]*pi/180
hc=h[o]-t1[i]/2-t2[k]/2-tc[j]

R=range(len(tc))
for jj in range(len(tc)):
    R[jj]=3*tc[jj]

a=R[s]*sin(theta_r)
b=R[s]*(1-cos(theta_r))
c=(h[o]-t1[i]/2-t2[k]/2-tc[j]-2*b)
d=c/tan(theta_r)
f=(r**2-(th/2)**2)**(0.5)
e1=f1[m]/2-dw[q]/2-tw[p]/2-r-f
e2=f2[n]/2-dw[q]/2-tw[p]/2-r-f

# CREATING PATHS TO EVALUATE THE STRESS
point_x_1=range(numb_points+1)
point_y_1=range(numb_points+1)
coord_1=range(numb_points+1)

for u in range(numb_points+1):
    point_x_1[u]=f1[m]/2+a+d+a+e2+f-r*cos(u*angle2+angle1)
    point_y_1[u]=t1[i]+b+c+b+tc[j]-th/2+r*sin(u*angle2+angle1)
    coord_1[u]=point_x_1[u], point_y_1[u], 0.0

session.Path(name='Stresses_1_'+Modelname[i], type=POINT_LIST, expression=(coord_1))

point_x_2=range(numb_points+1)
point_y_2=range(numb_points+1)
coord_2=range(numb_points+1)

for u in range(numb_points+1):
    point_x_2[u]=f1[m]/2+a+d+a+e2+f+r+tw[p]+r+r*cos(u*angle2+angle1)
    point_y_2[u]=t1[i]+b+c+b+tc[j]-th/2+r*sin(u*angle2+angle1)
    coord_2[u]=point_x_2[u], point_y_2[u], 0.0

session.Path(name='Stresses_2_'+Modelname[i], type=POINT_LIST, expression=(coord_2))

point_x_3=range(numb_points+1)
point_y_3=range(numb_points+1)
coord_3=range(numb_points+1)

for u in range(numb_points+1):
    point_x_3[u]=f1[m]/2+a+d+a+f2[n]-e2-f-r-tw[p]-r-r*cos(u*angle2+angle1)
    point_y_3[u]=t1[i]+b+c+b+tc[j]-th/2+r*sin(u*angle2+angle1)
    coord_3[u]=point_x_3[u], point_y_3[u], 0.0

session.Path(name='Stresses_3_'+Modelname[i], type=POINT_LIST, expression=(coord_3))

point_x_4=range(numb_points+1)
point_y_4=range(numb_points+1)
coord_4=range(numb_points+1)

for u in range(numb_points+1):
    point_x_4[u]=f1[m]/2+a+d+a+f2[n]-e2-f+r*cos(u*angle2+angle1)
    point_y_4[u]=t1[i]+b+c+b+tc[j]-th/2+r*sin(u*angle2+angle1)
    coord_4[u]=point_x_4[u], point_y_4[u], 0.0

session.Path(name='Stresses_4_'+Modelname[i], type=POINT_LIST, expression=(coord_4))
```

```
# OBTAINING STRESSES AT NOTCH 1
session.viewports['Viewport: 1'].setValues(displayedObject=odb)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Mises'))
S_M=session.XYDataFromPath(name='Mises_1_'+Modelname[i], path=session.paths['Stresses_1_'
+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS, numIntervals=10, shape
=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_M) > numb_points+1:
    B=[w[0] for w in S_M]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_M).pop(C)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Min. Principal'))
S_PSMIN=session.XYDataFromPath(name='Min.Principal_1_'+Modelname[i], path=session.paths[
'Stresses_1_'+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS,
numIntervals=10, shape=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_PSMIN) > numb_points+1:
    B=[w[0] for w in S_PSMIN]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_PSMIN).pop(C)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Max. Principal'))
S_PSMAX=session.XYDataFromPath(name='Max.Principal_1_'+Modelname[i], path=session.paths[
'Stresses_1_'+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS,
numIntervals=10, shape=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_PSMAX) > numb_points+1:
    B=[w[0] for w in S_PSMAX]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_PSMAX).pop(C)

S_1=range(numb_points+1)
for v in range(numb_points+1):
    session.viewports['Viewport: 1'].setValues(displayedObject=odb)
    S_1[v]=range(6)

    S_1[v][0]=S_M[v][0]
    S_1[v][1]=S_M[v][1]
    S_1[v][2]=S_PSMIN[v][1]
    S_1[v][3]=S_PSMAX[v][1]
    S_1[v][4]=abs(S_1[v][2])

    if S_1[v][2]<= 0:
        S_1[v][5]=1

    else:
        S_1[v][5]=0



Output[i]=range(61)

vec=range(numb_points+1)
```

```
for v in range(numb_points+1):
    vec[v]=S_1[v][1]
    max_VM=max(vec)

Output[i][1]=max_VM

for v in range(numb_points+1):
    if max_VM==S_1[v][1]:
        Output[i][0]=S_1[v][0]/S_M[numb_points][0]

vec=range((numb_points+1)*2)
vvv=0
for vv in range((numb_points+1)*2):
    if vv < len(range((numb_points+1))):
        vec[vv]=S_1[vvv][3]
        vvv=vvv+1
    else:
        if vvv == len(range((numb_points+1))):
            vvv=0
        vec[vv]=S_1[vvv][4]
        vvv=vvv+1

max_PS=max(vec)

Output[i][3]=max_PS

for v in range(numb_points+1):
    if max_PS==S_1[v][3]:
        Output[i][2]=S_1[v][0]/S_PSMAX[numb_points][0]
        Output[i][4]=S_1[v][5]
    if max_PS==S_1[v][4]:
        Output[i][2]=S_1[v][0]/S_PSMAX[numb_points][0]
        Output[i][4]=S_1[v][5]

# OBTAINING STRESSES AT NOTCH 2
session.viewports['Viewport: 1'].setValues(displayedObject=odb)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Mises'))
S_M=session.XYDataFromPath(name='Mises_2_'+Modelname[i], path=session.paths['Stresses_2_'
+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS, numIntervals=10, shape
=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_M) > numb_points+1:
    B=[w[0] for w in S_M]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_M).pop(C)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Min. Principal'))
S_PSMIN=session.XYDataFromPath(name='Min.Principal_2_'+Modelname[i], path=session.paths[
'Stresses_2_'+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS,
numIntervals=10, shape=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_PSMIN) > numb_points+1:
    B=[w[0] for w in S_PSMIN]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_PSMIN).pop(C)
```

```
session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Max. Principal'))
S_PSMAX=session.XYDataFromPath(name='Max.Principal_2_'+Modelname[i], path=session.paths[
'Stresses_2_'+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS,
numIntervals=10, shape=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_PSMAX) > numb_points+1:
    B=[w[0] for w in S_PSMAX]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_PSMAX).pop(C)

S_2=range(numb_points+1)
for v in range(numb_points+1):
    session.viewports['Viewport: 1'].setValues(displayedObject=odb)
    S_2[v]=range(6)

    S_2[v][0]=S_M[v][0]
    S_2[v][1]=S_M[v][1]
    S_2[v][2]=S_PSMIN[v][1]
    S_2[v][3]=S_PSMAX[v][1]
    S_2[v][4]=abs(S_2[v][2])

    if S_2[v][2]<=0:
        S_2[v][5]=1

    else:
        S_2[v][5]=0


vec=range(numb_points+1)
for v in range(numb_points+1):
    vec[v]=S_2[v][1]
    max_VM=max(vec)

Output[i][6]=max_VM

for v in range(numb_points+1):
    if max_VM==S_2[v][1]:
        Output[i][5]=S_2[v][0]/S_M[numb_points][0]

vec=range((numb_points+1)*2)
vvv=0
for vv in range((numb_points+1)*2):
    if vv < len(range((numb_points+1))):
        vec[vv]=S_2[vvv][3]
        vvv=vvv+1
    else:
        if vvv == len(range((numb_points+1))):
            vvv=0
        vec[vv]=S_2[vvv][4]
        vvv=vvv+1

max_PS=max(vec)

Output[i][8]=max_PS
```

```
for v in range(numb_points+1):
    if max_PS==S_2[v][3]:
        Output[i][7]=S_2[v][0]/S_PSMAX[numb_points][0]
        Output[i][9]=S_2[v][5]
    if max_PS==S_2[v][4]:
        Output[i][7]=S_2[v][0]/S_PSMAX[numb_points][0]
        Output[i][9]=S_2[v][5]

# OBTAINING STRESSES AT NOTCH 3
session.viewports['Viewport: 1'].setValues(displayedObject=odb)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Mises'))
S_M=session.XYDataFromPath(name='Mises_3_'+Modelname[i], path=session.paths['Stresses_3_'
+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS, numIntervals=10, shape
=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_M) > numb_points+1:
    B=[w[0] for w in S_M]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_M).pop(C)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Min. Principal'))
S_PSMIN=session.XYDataFromPath(name='Min.Principal_3_'+Modelname[i], path=session.paths[
'Stresses_3_'+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS,
numIntervals=10, shape=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_PSMIN) > numb_points+1:
    B=[w[0] for w in S_PSMIN]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_PSMIN).pop(C)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Max. Principal'))
S_PSMAX=session.XYDataFromPath(name='Max.Principal_3_'+Modelname[i], path=session.paths[
'Stresses_3_'+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS,
numIntervals=10, shape=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_PSMAX) > numb_points+1:
    B=[w[0] for w in S_PSMAX]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_PSMAX).pop(C)

S_3=range(numb_points+1)
for v in range(numb_points+1):
    session.viewports['Viewport: 1'].setValues(displayedObject=odb)
    S_3[v]=range(6)

    S_3[v][0]=S_M[v][0]
    S_3[v][1]=S_M[v][1]
    S_3[v][2]=S_PSMIN[v][1]
    S_3[v][3]=S_PSMAX[v][1]
    S_3[v][4]=abs(S_3[v][2])

    if S_3[v][2]<= 0:
        S_3[v][5]=1

    else:
        S_3[v][5]=0

vec=range(numb_points+1)
```

```
for v in range(numb_points+1):
    vec[v]=S_3[v][1]
max_VM=max(vec)

Output[i][11]=max_VM

for v in range(numb_points+1):
    if max_VM==S_3[v][1]:
        Output[i][10]=S_3[v][0]/S_M[numb_points][0]

vec=range((numb_points+1)*2)
vvv=0
for vv in range((numb_points+1)*2):
    if vv < len(range((numb_points+1))):
        vec[vv]=S_3[vvv][3]
        vvv=vvv+1
    else:
        if vvv == len(range((numb_points+1))):
            vvv=0
        vec[vv]=S_3[vvv][4]
        vvv=vvv+1

max_PS=max(vec)

Output[i][13]=max_PS

for v in range(numb_points+1):
    if max_PS==S_3[v][3]:
        Output[i][12]=S_3[v][0]/S_PSMAX[numb_points][0]
        Output[i][14]=S_3[v][5]
    if max_PS==S_3[v][4]:
        Output[i][12]=S_3[v][0]/S_PSMAX[numb_points][0]
        Output[i][14]=S_3[v][5]

# OBTAINING STRESSES AT NOTCH 4
session.viewports['Viewport: 1'].setValues(displayedObject=odb)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Mises'))
S_M=session.XYDataFromPath(name='Mises_4_'+Modelname[i], path=session.paths['Stresses_4_'
+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS, numIntervals=10, shape
=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_M) > numb_points+1:
    B=[w[0] for w in S_M]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_M).pop(C)

session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Min. Principal'))
S_PSMIN=session.XYDataFromPath(name='Min.Principal_4_'+Modelname[i], path=session.paths[
'Stresses_4_'+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS,
numIntervals=10, shape=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_PSMIN) > numb_points+1:
    B=[w[0] for w in S_PSMIN]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_PSMIN).pop(C)
```

```
session.viewports['Viewport: 1'].odbDisplay.setPrimaryVariable(variableLabel='S',
outputPosition=INTEGRATION_POINT, refinement=(INVARIANT, 'Max. Principal'))
S_PSMAX=session.XYDataFromPath(name='Max.Principal_4_'+Modelname[i], path=session.paths[
'Stresses_4_'+Modelname[i]], includeIntersections=False, pathStyle=PATH_POINTS,
numIntervals=10, shape=UNDEFORMED, labelType=TRUE_DISTANCE)
if len(S_PSMAX) > numb_points+1:
    B=[w[0] for w in S_PSMAX]
    C=B.index([x for x in B if B.count(x) > 1][0])
    list(S_PSMAX).pop(C)

S_4=range(numb_points+1)
for v in range(numb_points+1):
    session.viewports['Viewport: 1'].setValues(displayedObject=odb)
    S_4[v]=range(6)

    S_4[v][0]=S_M[v][0]
    S_4[v][1]=S_M[v][1]
    S_4[v][2]=S_PSMIN[v][1]
    S_4[v][3]=S_PSMAX[v][1]
    S_4[v][4]=abs(S_4[v][2])

    if S_4[v][2]<= 0:
        S_4[v][5]=1

    else:
        S_4[v][5]=0

vec=range(numb_points+1)
for v in range(numb_points+1):
    vec[v]=S_4[v][1]
max_VM=max(vec)

Output[i][16]=max_VM

for v in range(numb_points+1):
    if max_VM==S_4[v][1]:
        Output[i][15]=S_4[v][0]/S_M[numb_points][0]

vec=range((numb_points+1)*2)
vvv=0
for vv in range((numb_points+1)*2):
    if vv < len(range((numb_points+1))):
        vec[vv]=S_4[vvv][3]
        vvv=vvv+1
    else:
        if vvv == len(range((numb_points+1))):
            vvv=0
        vec[vv]=S_4[vvv][4]
        vvv=vvv+1

max_PS=max(vec)

Output[i][18]=max_PS
```

```
for v in range(numb_points+1):
    if max_PS==S_4[v][3]:
        Output[i][17]=S_4[v][0]/S_PSMAX[numb_points][0]
        Output[i][19]=S_4[v][5]
    if max_PS==S_4[v][4]:
        Output[i][17]=S_4[v][0]/S_PSMAX[numb_points][0]
        Output[i][19]=S_4[v][5]

# OBTAINING REACTION FORCES
Output[i][20]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
 NODAL, ((COMPONENT, 'RF1'), )), ), nodeSets=('REACTION_FORCES_BOTTOM_LEFT', ))[0][0][1]
Output[i][21]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
 NODAL, ((COMPONENT, 'RF2'), )), ), nodeSets=('REACTION_FORCES_BOTTOM_LEFT_CORE', ))[0][0
][1]
Output[i][22]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
 NODAL, ((COMPONENT, 'RF1'), )), ), nodeSets=('REACTION_FORCES_BOTTOM_RIGHT', ))[0][0][1]
Output[i][23]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
 NODAL, ((COMPONENT, 'RF2'), )), ), nodeSets=('REACTION_FORCES_BOTTOM_RIGHT_CORE', ))[0][
0][1]
Output[i][24]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
 NODAL, ((COMPONENT, 'RF2'), )), ), nodeSets=('REACTION_FORCES_BOTTOM_LEFT', ))[0][0][1]
Output[i][25]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
 NODAL, ((COMPONENT, 'RF2'), )), ), nodeSets=('REACTION_FORCES_BOTTOM_RIGHT', ))[0][0][1]
Output[i][26]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
 NODAL, ((COMPONENT, 'RF2'), )), ), nodeSets=('REACTION_FORCES_TOP_LEFT', ))[0][0][1]
Output[i][27]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('RF',
 NODAL, ((COMPONENT, 'RF2'), )), ), nodeSets=('REACTION_FORCES_TOP_RIGHT', ))[0][0][1]

# OBTAINING DISPLACEMENTS
Output[i][28]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('U',
NODAL, ((COMPONENT, 'U1'), )), ), nodeSets=('DISPLACEMENTS_LEFT', ))[0][0][1]
Output[i][29]=session.xyDataListFromField(odb=odb, outputPosition=NODAL, variable=(('U',
NODAL, ((COMPONENT, 'U1'), )), ), nodeSets=('DISPLACEMENTS_RIGHT', ))[0][0][1]

# CALCULATING TRANSVERSE SHEAR STIFFNESS
p_=(f1[m]+a+d+a+f2[n]+a+d+a)/2
Output[i][30]=(h[o]**2)/((session.xyDataListFromField(odb=odb, outputPosition=NODAL,
variable=(('U', NODAL, ((COMPONENT, 'U1'), )), ), nodeSets=('DISPLACEMENTS_RIGHT', ))[0][
0][1])*p_)

# OBTAINING MOMENT, SHEAR FORCE AND NORMAL FORCE FROM FREE BODY CUT
W=range(4)
U=['CORE', 'PLATE']

for w in W:
    w_=w+1
    for u in U:
        if u=='CORE':
            eLeaf = dgo.LeafFromElementSets(elementSets=('FBC_'+str(w_)+'_'+u, ))
            nLeaf = dgo.LeafFromNodeSets(nodeSets=('FBC_'+str(w_)+'_'+u, ))
            session.FreeBodyFromNodesElements(name='FreeBody-1', elements=eLeaf, nodes=
            nLeaf, summationLoc=NODAL_AVERAGE, componentResolution=CSYS, csysName=GLOBAL)
```

```
            M_C=session.XYDataFromFreeBody(odb=odb, force=OFF, moment=ON, resultant=OFF,
            comp1=OFF, comp2=OFF, comp3=ON)[0][0][1]
            V_C=session.XYDataFromFreeBody(odb=odb, force=ON, moment=OFF, resultant=OFF,
            comp1=OFF, comp2=ON, comp3=OFF)[0][0][1]
            N_C=session.XYDataFromFreeBody(odb=odb, force=ON, moment=OFF, resultant=OFF,
            comp1=ON, comp2=OFF, comp3=OFF)[0][0][1]

            Output[i][31+w*6]=M_C
            Output[i][32+w*6]=V_C
            Output[i][33+w*6]=N_C

            del session.freeBodies['FreeBody-1']


        if u=='PLATE':
            eLeaf = dgo.LeafFromElementSets(elementSets=('FBC_'+str(w_)+'_'+u, ))
            nLeaf = dgo.LeafFromNodeSets(nodeSets=('FBC_'+str(w_)+'_'+u, ))
            session.FreeBodyFromNodesElements(name='FreeBody-1', elements=eLeaf, nodes=
            nLeaf, summationLoc=NODAL_AVERAGE, componentResolution=CSYS, csysName=GLOBAL)

            M_P=session.XYDataFromFreeBody(odb=odb, force=OFF, moment=ON, resultant=OFF,
            comp1=OFF, comp2=OFF, comp3=ON)[0][0][1]
            V_P=session.XYDataFromFreeBody(odb=odb, force=ON, moment=OFF, resultant=OFF,
            comp1=OFF, comp2=ON, comp3=OFF)[0][0][1]
            N_P=session.XYDataFromFreeBody(odb=odb, force=ON, moment=OFF, resultant=OFF,
            comp1=ON, comp2=OFF, comp3=OFF)[0][0][1]

            Output[i][34+w*6]=M_P
            Output[i][35+w*6]=V_P
            Output[i][36+w*6]=N_P

            del session.freeBodies['FreeBody-1']

W=range(2)
U=['WELD']

for w in W:
    w_=w+1
    for u in U:
            eLeaf = dgo.LeafFromElementSets(elementSets=('FBC_'+str(w_)+'_'+u, ))
            nLeaf = dgo.LeafFromNodeSets(nodeSets=('FBC_'+str(w_)+'_'+u, ))
            session.FreeBodyFromNodesElements(name='FreeBody-1', elements=eLeaf, nodes=
            nLeaf, summationLoc=NODAL_AVERAGE, componentResolution=CSYS, csysName=GLOBAL)

            M_W=session.XYDataFromFreeBody(odb=odb, force=OFF, moment=ON, resultant=OFF,
            comp1=OFF, comp2=OFF, comp3=ON)[0][0][1]
            V_W=session.XYDataFromFreeBody(odb=odb, force=ON, moment=OFF, resultant=OFF,
            comp1=OFF, comp2=ON, comp3=OFF)[0][0][1]
            N_W=session.XYDataFromFreeBody(odb=odb, force=ON, moment=OFF, resultant=OFF,
            comp1=ON, comp2=OFF, comp3=OFF)[0][0][1]
```

```
                Output[i][55+w*3]=M_W
                Output[i][56+w*3]=V_W
                Output[i][57+w*3]=N_W

                del session.freeBodies['FreeBody-1']

    # OBTAINING VALUES FOR PARAMETERS FOR INDATA-FILE
    Indata[i]=range(12)

    Indata[i][0]=int(float(Modelname[i]))
    Indata[i][1]=t1[i]
    Indata[i][2]=tc[j]
    Indata[i][3]=t2[k]
    Indata[i][4]=theta[l]
    Indata[i][5]=f1[m]
    Indata[i][6]=f2[n]
    Indata[i][7]=h[o]
    Indata[i][8]=tw[p]
    Indata[i][9]=dw[q]
    Indata[i][10]=R[s]

    j=j+1
    k=k+1
    l=l+1
    m=m+1
    n=n+1
    o=o+1
    p=p+1
    q=q+1
    s=s+1

# SAVING RESULTS IN OUTPUT-FILE
os.remove('C:/Parametric_study_TwoWelds_Fullfact/Output.txt')
for i in range(len(t1)):
    OutputFile=open('C:/Parametric_study_TwoWelds_Fullfact/Output.txt','a+')
    OutputFile.write(
    '%10.5E\t%10.5E\t%10.5E\t%10.5E\t%g\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%g\t%10.5E\t%10.5E\t%
    10.5E\t%10.5E\t%g\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%g\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10
    .5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5
    E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\
    t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%10.5E\t%
    10.5E\t%10.5E\t%10.5E\n' %(Output[i][0],
    Output[i][1],Output[i][2],Output[i][3],Output[i][4],Output[i][5],
    Output[i][6],Output[i][7],Output[i][8],Output[i][9],Output[i][10],Output[i][11],Output[i
    ][12],Output[i][13],
    Output[i][14],Output[i][15],Output[i][16],Output[i][17],Output[i][18],Output[i][19],
    Output[i][20],Output[i][21]
    , Output[i][22],Output[i][23],Output[i][24],Output[i][25],Output[i][26],Output[i][27],
    Output[i][28], Output[i][29],Output[i][30], Output[i][31], Output[i][32], Output[i][33],
    Output[i][34]
    , Output[i][35], Output[i][36], Output[i][37], Output[i][38], Output[i][39], Output[i][40
    ], Output[i][41], Output[i][42], Output[i][43], Output[i][44], Output[i][45], Output[i][
    46]
    , Output[i][47], Output[i][48], Output[i][49], Output[i][50], Output[i][51], Output[i][52
    ], Output[i][53], Output[i][54],  Output[i][55], Output[i][56], Output[i][57], Output[i][
    58], Output[i][59], Output[i][60]))
    OutputFile.close()
```

```
# SAVING MODELNUMBER CORRESPONDING VALUES OF PARAMETERS IN INDATA-FILE
os.remove('C:/Parametric_study_TwoWelds_Fullfact/Indata.txt')
for i in range(len(t1)):
    IndataFile=open('C:/Parametric_study_TwoWelds_Fullfact/Indata.txt','a+')
    IndataFile.write('%g\t\t%g\t%g\t%g\t%g\t%g\t%g\t%g\t%g\t%g\n' %(Indata[i][0],Indata[i
    ][1],Indata[i][2],Indata[i][3],Indata[i][4],Indata[i][5],
    Indata[i][6],Indata[i][7],Indata[i][8],Indata[i][9],Indata[i][10]))
    IndataFile.close()
```

# Appendix B

## Mathcad calculations of main effect

### Area of cross-section

Reading data from files

$Data := \text{READPRN}(\text{“Area\_1.txt”})$

$Indata := \text{READPRN}(\text{“Indata\_1.txt”})$

Extracting area, distance p, and height for each model

$A_{tot} := Data^{\langle 0 \rangle} \cdot m^2$ Area of cross-sections

$p := Data^{\langle 1 \rangle} \cdot m$ Half of the with of cross-sections

$h := Indata^{\langle 7 \rangle} \cdot m$ Height of the cross-sections

Density of steel

$$\rho_{steel} := 7800 \cdot \frac{kg}{m^2}$$

Area per unit width

$$A := \frac{A_{tot}}{2 \cdot p}$$

Calculating mass for each model

$$mass := \rho_{steel} \cdot A = \begin{bmatrix} 148.32 \\ 160.89 \\ 155.803 \\ 168.118 \\ 160.277 \\ 172.624 \\ 168.868 \\ 181.46 \\ 151.778 \\ 164.497 \\ 162.422 \\ 174.551 \\ \vdots \end{bmatrix} \frac{kg}{m}$$

**Design matrix**

Generating Fractional Factorial Design matrix (10 parameters, resolution 4)

$A_0 := \text{fractfact}(10, 4)$

**Main effects**

Reading data from output-file created analysing the models in Brigade 6.1

$C := \text{READPRN}(\text{“Output\_1.txt”})$

Extracting, from the output-data the principal stresses around the four notches in the models

$PS1 := C^{\langle 3 \rangle}$ Principal stress around notch 1

$PS2 := C^{\langle 8 \rangle}$ Principal stress around notch 2

$PS3 := C^{\langle 13 \rangle}$ Principal stress around notch 3

$PS4 := C^{\langle 18 \rangle}$ Principal stress around notch 4

Creating a matrix containing all the principal stresses

$PS := \text{augment}(PS1, PS2, PS3, PS4)$

Creating a vector containing the maximum principal stress, from the four notches, for each model

$$S := \left\| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS^{\langle 0 \rangle}\right) \\ \quad \left\| \begin{array}{l} S_i \leftarrow \max(\text{submatrix}(PS, i, i, 0, 3)) \\ \end{array} \right. \\ S \end{array} \right.$$

Dividing the principal stress with h/p to obtain the principal stress caused by one unit shear

$$S := \frac{S}{\left(\frac{h}{p}\right)}$$

Extracting, from the output-data the transverse shear stiffness for the models

$$D_{Qy} := \frac{C^{\langle 30 \rangle}}{p^2}$$

Calculating the effect of each factor level, principal stress.

The effects of the factor level are calculated about the overall mean of the results. The effect returned for the low level of factor "A" is the difference between its low avrage response and the overall mean of the results. The effect returned for the high level of factor "A" is the difference between its high avrage response and the overall mean of the results.

$$\text{effects}\langle A_0, S\rangle = \begin{bmatrix} \text{“Factor”} & \text{“Effects”} \\ \text{“A”} & [3 \times 2] \\ \text{“B”} & [3 \times 2] \\ \text{“C”} & [3 \times 2] \\ \text{“D”} & [3 \times 2] \\ \text{“E”} & [3 \times 2] \\ \text{“F”} & [3 \times 2] \\ \text{“G”} & [3 \times 2] \\ \text{“H”} & [3 \times 2] \\ \text{“I”} & [3 \times 2] \\ \text{“J”} & [3 \times 2] \end{bmatrix}$$

Calculating the effect if each factor, principal stress

$$G := \text{quickscreen}\langle A_0, S\rangle = \begin{bmatrix} \text{“Factor”} & \text{“(-) Avg”} & \text{“(+) Avg”} & \text{“(+) - (-)”} \\ \text{“A”} & 8.006 \cdot 10^3 & 8.544 \cdot 10^3 & 537.919 \\ \text{“B”} & 1.07 \cdot 10^4 & 5.847 \cdot 10^3 & -4.857 \cdot 10^3 \\ \text{“C”} & 7.778 \cdot 10^3 & 8.773 \cdot 10^3 & 994.853 \\ \text{“D”} & 8.089 \cdot 10^3 & 8.462 \cdot 10^3 & 373.034 \\ \text{“E”} & 8.64 \cdot 10^3 & 7.911 \cdot 10^3 & -728.406 \\ \text{“F”} & 6.423 \cdot 10^3 & 1.013 \cdot 10^4 & 3.705 \cdot 10^3 \\ \text{“G=BCDF”} & 8.055 \cdot 10^3 & 8.496 \cdot 10^3 & 440.622 \\ \text{“H=ACDF”} & 8.339 \cdot 10^3 & 8.212 \cdot 10^3 & -127.509 \\ \text{“I=ABDE”} & 9.265 \cdot 10^3 & 7.285 \cdot 10^3 & -1.98 \cdot 10^3 \\ \text{“J=ABCE”} & 7.962 \cdot 10^3 & 8.588 \cdot 10^3 & 626.151 \end{bmatrix}$$

Extracting the principal stress effect, the difference between the high and the low avrage responses

$$G_3 := \text{submatrix}(G, 1, \text{rows}(G) - 1, 3, 3)$$

$$G_3 := \frac{G_3}{\max(G_3)}$$

$$T1 := \text{WRITEPRN}(\text{“G31.txt”}, G_3)$$

Creating a vector containg the parameters

$$G_0 := \begin{bmatrix} \text{“t1”} \\ \text{“tc”} \\ \text{“t2”} \\ \text{“θ”} \\ \text{“f1”} \\ \text{“f2”} \\ \text{“h”} \\ \text{“tw”} \\ \text{“dw”} \\ \text{“R”} \end{bmatrix}$$

Creating a vector of two elements that is used to generate an effects plot

$$P := \text{effectsgraph}(G)$$

$$P_1 := \frac{P_1}{\max(P_1)}$$

$$T2 := \text{WRITEPRN}(\text{“P11.txt”}, P_1)$$

Calculating the effect of each factor level, transverse shear stiffness

The effects of the factor level are calculated about the overall mean of the results. The effect returned for the low level of factor "A" is the difference between its low avrage response and the overall mean of the results. The effect returned for the high level of factor "A" is the difference between its high avrage response and the overall mean of the results.

$$\text{effects}(A_0, D_{Qv}) = \begin{bmatrix} \text{“Factor”} & \text{“Effects”} \\ \text{“A”} & [3 \times 2] \\ \text{“B”} & [3 \times 2] \\ \text{“C”} & [3 \times 2] \\ \text{“D”} & [3 \times 2] \\ \text{“E”} & [3 \times 2] \\ \text{“F”} & [3 \times 2] \\ \text{“G”} & [3 \times 2] \\ \text{“H”} & [3 \times 2] \\ \text{“I”} & [3 \times 2] \\ \text{“J”} & [3 \times 2] \end{bmatrix}$$

Calculating the effect if each factor, transverse shear stiffness

$$H := \text{quickscreen}\left(A_0, D_{Qy}\right) = \begin{bmatrix} \text{"Factor"} & \text{"(-) Avg"} & \text{"(+) Avg"} & \text{"(+) - (-)"} \\ \text{"A"} & \left(1.428 \cdot 10^7\right) \frac{1}{m^2} & \left(1.673 \cdot 10^7\right) \frac{1}{m^2} & \left(2.443 \cdot 10^6\right) \frac{1}{m^2} \\ \text{"B"} & \left(1.285 \cdot 10^7\right) \frac{1}{m^2} & \left(1.816 \cdot 10^7\right) \frac{1}{m^2} & \left(5.315 \cdot 10^6\right) \frac{1}{m^2} \\ \text{"C"} & \left(1.428 \cdot 10^7\right) \frac{1}{m^2} & \left(1.673 \cdot 10^7\right) \frac{1}{m^2} & \left(2.45 \cdot 10^6\right) \frac{1}{m^2} \\ \text{"D"} & \left(1.844 \cdot 10^7\right) \frac{1}{m^2} & \left(1.257 \cdot 10^7\right) \frac{1}{m^2} & -5.862 \cdot 10^6 \frac{1}{m^2} \\ \text{"E"} & \left(1.739 \cdot 10^7\right) \frac{1}{m^2} & \left(1.362 \cdot 10^7\right) \frac{1}{m^2} & -3.767 \cdot 10^6 \frac{1}{m^2} \\ \text{"F"} & \left(1.738 \cdot 10^7\right) \frac{1}{m^2} & \left(1.363 \cdot 10^7\right) \frac{1}{m^2} & -3.754 \cdot 10^6 \frac{1}{m^2} \\ \text{"G=BCDF"} & \left(1.526 \cdot 10^7\right) \frac{1}{m^2} & \left(1.575 \cdot 10^7\right) \frac{1}{m^2} & \left(4.945 \cdot 10^5\right) \frac{1}{m^2} \\ \text{"H=ACDF"} & \left(1.536 \cdot 10^7\right) \frac{1}{m^2} & \left(1.565 \cdot 10^7\right) \frac{1}{m^2} & \left(2.912 \cdot 10^5\right) \frac{1}{m^2} \\ \text{"I=ABDE"} & \left(1.471 \cdot 10^7\right) \frac{1}{m^2} & \left(1.63 \cdot 10^7\right) \frac{1}{m^2} & \left(1.599 \cdot 10^6\right) \frac{1}{m^2} \\ \text{"J=ABCE"} & \left(1.632 \cdot 10^7\right) \frac{1}{m^2} & \left(1.469 \cdot 10^7\right) \frac{1}{m^2} & -1.622 \cdot 10^6 \frac{1}{m^2} \end{bmatrix}$$

Extracting the transverse shear stifness,
the difference between the high and the low avrage responses

$$H_3 := \text{submatrix}\left(H, 1, \text{rows}(H) - 1, 3, 3\right)$$

$$H_3 := \frac{H_3}{\max\left(H_3\right)}$$

$$T3 := \text{WRITEPRN}\left(\text{"H31.txt"}, H_3\right)$$

Creating a vector containg the parameters

$$H_0 := \begin{bmatrix} \text{"t1"} \\ \text{"tc"} \\ \text{"t2"} \\ \text{"θ"} \\ \text{"f1"} \\ \text{"f2"} \\ \text{"h"} \\ \text{"tw"} \\ \text{"dw"} \\ \text{"R"} \end{bmatrix}$$

Creating a vector of two elements that is used to generate an effects plot

$$Q := \text{effectsgraph}(H)$$

$$Q_1 := \frac{Q_1}{\max\left(Q_1\right)}$$

$$T4 := \text{WRITEPRN}\left(\text{"Q11.txt"}, Q_1\right)$$

**Mass effectivity**

Dividing the principal stress with the mass of the cross-section

$$\sigma_{eff} := \overrightarrow{S \cdot mass} = \begin{bmatrix} 9.828 \cdot 10^5 \\ 1.51 \cdot 10^6 \\ 6.823 \cdot 10^5 \\ 7.08 \cdot 10^5 \\ 1.104 \cdot 10^6 \\ 1.844 \cdot 10^6 \\ 1.02 \cdot 10^6 \\ 7.621 \cdot 10^5 \\ 1.479 \cdot 10^6 \\ 1.105 \cdot 10^6 \\ 6.91 \cdot 10^5 \\ 9.751 \cdot 10^5 \\ \vdots \end{bmatrix} \frac{kg^2}{m^2} \cdot \frac{m}{kg}$$

Calculating the effect of each factor level, principal stress with regard to mass effectivity

The effects of the factor level are calculated about the overall mean of the results. The effect returned for the low level of factor "A" is the difference between its low avrage response and the overall mean of the results. The effect returned for the high level of factor "A" is the difference between its high avrage response and the overall mean of the results.

$$\text{effects}\left(\langle A_0, \sigma_{eff}\rangle\right) = \begin{bmatrix} \text{"Factor"} & \text{"Effects"} \\ \text{"A"} & [3 \times 2] \\ \text{"B"} & [3 \times 2] \\ \text{"C"} & [3 \times 2] \\ \text{"D"} & [3 \times 2] \\ \text{"E"} & [3 \times 2] \\ \text{"F"} & [3 \times 2] \\ \text{"G"} & [3 \times 2] \\ \text{"H"} & [3 \times 2] \\ \text{"I"} & [3 \times 2] \\ \text{"J"} & [3 \times 2] \end{bmatrix}$$

Calculating the effect if each factor, principal stress with regard to mass effectivity

$$G_{eff} := \text{quickscreen}\left(A_0, \sigma_{eff}, \frac{kg}{m}\right) = \begin{bmatrix} \text{"Factor"} & \text{"(-) Avg"} & \text{"(+) Avg"} & \text{"(+) - (-)"} \\ \text{"A"} & (1.278 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.471 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.93 \cdot 10^5)\ \frac{kg^2}{m^2} \\ \text{"B"} & (1.745 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.004 \cdot 10^6)\ \frac{kg^2}{m^2} & -7.414 \cdot 10^5\ \frac{kg^2}{m^2} \\ \text{"C"} & (1.24 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.508 \cdot 10^6)\ \frac{kg^2}{m^2} & (2.684 \cdot 10^5)\ \frac{kg^2}{m^2} \\ \text{"D"} & (1.324 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.424 \cdot 10^6)\ \frac{kg^2}{m^2} & (9.978 \cdot 10^4)\ \frac{kg^2}{m^2} \\ \text{"E"} & (1.436 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.313 \cdot 10^6)\ \frac{kg^2}{m^2} & -1.234 \cdot 10^5\ \frac{kg^2}{m^2} \\ \text{"F"} & (1.068 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.68 \cdot 10^6)\ \frac{kg^2}{m^2} & (6.117 \cdot 10^5)\ \frac{kg^2}{m^2} \\ \text{"G=BCDF"} & (1.333 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.415 \cdot 10^6)\ \frac{kg^2}{m^2} & (8.206 \cdot 10^4)\ \frac{kg^2}{m^2} \\ \text{"H=ACDF"} & (1.385 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.363 \cdot 10^6)\ \frac{kg^2}{m^2} & -2.173 \cdot 10^4\ \frac{kg^2}{m^2} \\ \text{"I=ABDE"} & (1.538 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.211 \cdot 10^6)\ \frac{kg^2}{m^2} & -3.273 \cdot 10^5\ \frac{kg^2}{m^2} \\ \text{"J=ABCE"} & (1.323 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.425 \cdot 10^6)\ \frac{kg^2}{m^2} & (1.019 \cdot 10^5)\ \frac{kg^2}{m^2} \end{bmatrix}$$

Extracting the principal stress effect, the difference between the high and the low avrage responses

$$G_{eff:3} := \text{submatrix}\left(G_{eff}, 1, \text{rows}\left(G_{eff}\right) - 1, 3, 3\right)$$

$$G_{eff:3} := \frac{G_{eff:3}}{\max\left(G_{eff:3}\right)}$$

$$T5 := \text{WRITEPRN}\left(\text{"G3e1.txt"}, G_{eff:3}\right)$$

Creating a vector containg the parameters

$$G_{eff:0} := \begin{bmatrix} \text{"t1"} \\ \text{"tc"} \\ \text{"t2"} \\ \text{"θ"} \\ \text{"f1"} \\ \text{"f2"} \\ \text{"h"} \\ \text{"tw"} \\ \text{"dw"} \\ \text{"R"} \end{bmatrix}$$

Creating a vector of two elements that is used to generate an effects plot

$$P_{eff} := \text{effectsgraph}\left(G_{eff}\right)$$

$$P_{eff_1} := \frac{P_{eff_1}}{\max\left(P_{eff_1}\right)} \qquad T6 := \text{WRITEPRN}\left(\text{"P1e1.txt"}, P_{eff_1}\right)$$

Dividing the transverse shear stiffness with the mass of the cross-section

$$D_{Qy.eff} := \frac{D_{Qy}}{mass} = \begin{bmatrix} 1.153 \cdot 10^5 \\ 1.216 \cdot 10^5 \\ 1.493 \cdot 10^5 \\ 1.537 \cdot 10^5 \\ 1.335 \cdot 10^5 \\ 1.171 \cdot 10^5 \\ 1.458 \cdot 10^5 \\ 2.021 \cdot 10^5 \\ 6.506 \cdot 10^4 \\ 8.949 \cdot 10^4 \\ 1.051 \cdot 10^5 \\ 9.666 \cdot 10^4 \\ \vdots \end{bmatrix} \frac{1}{kg \cdot m}$$

Calculating the effect of each factor level, transverse shear stiffness with regard to mass effectivity

The effects of the factor level are calculated about the overall mean of the results. The effect returned for the low level of factor "A" is the difference between its low avrage response and the overall mean of the results. The effect returned for the high level of factor "A" is the difference between its high avrage response and the overall mean of the results.

$$\text{effects}\left(A_0, D_{Qy.eff}\right) = \begin{bmatrix} \text{"Factor"} & \text{"Effects"} \\ \text{"A"} & [3 \times 2] \\ \text{"B"} & [3 \times 2] \\ \text{"C"} & [3 \times 2] \\ \text{"D"} & [3 \times 2] \\ \text{"E"} & [3 \times 2] \\ \text{"F"} & [3 \times 2] \\ \text{"G"} & [3 \times 2] \\ \text{"H"} & [3 \times 2] \\ \text{"I"} & [3 \times 2] \\ \text{"J"} & [3 \times 2] \end{bmatrix}$$

Calculating the effect if each factor, transverse shear stiffness with reagard to mass effectivity

$$H_{eff} := \text{quickscreen}\left(A_0, D_{Qy.eff} \cdot \frac{kg}{m}\right) =$$

| "Factor" | "(-) Avg" | "(+) Avg" | "(+) - (-)" |
|---|---|---|---|
| "A" | $(8.882 \cdot 10^4)\ \frac{1}{m^2}$ | $(9.644 \cdot 10^4)\ \frac{1}{m^2}$ | $(7.621 \cdot 10^3)\ \frac{1}{m^2}$ |
| "B" | $(7.905 \cdot 10^4)\ \frac{1}{m^2}$ | $(1.062 \cdot 10^5)\ \frac{1}{m^2}$ | $(2.715 \cdot 10^4)\ \frac{1}{m^2}$ |
| "C" | $(8.879 \cdot 10^4)\ \frac{1}{m^2}$ | $(9.647 \cdot 10^4)\ \frac{1}{m^2}$ | $(7.685 \cdot 10^3)\ \frac{1}{m^2}$ |
| "D" | $(1.114 \cdot 10^5)\ \frac{1}{m^2}$ | $(7.386 \cdot 10^4)\ \frac{1}{m^2}$ | $-3.754 \cdot 10^4\ \frac{1}{m^2}$ |
| "E" | $(1.038 \cdot 10^5)\ \frac{1}{m^2}$ | $(8.146 \cdot 10^4)\ \frac{1}{m^2}$ | $-2.233 \cdot 10^4\ \frac{1}{m^2}$ |
| "F" | $(1.038 \cdot 10^5)\ \frac{1}{m^2}$ | $(8.15 \cdot 10^4)\ \frac{1}{m^2}$ | $-2.225 \cdot 10^4\ \frac{1}{m^2}$ |
| "G=BCDF" | $(9.143 \cdot 10^4)\ \frac{1}{m^2}$ | $(9.382 \cdot 10^4)\ \frac{1}{m^2}$ | $(2.39 \cdot 10^3)\ \frac{1}{m^2}$ |
| "H=ACDF" | $(9.177 \cdot 10^4)\ \frac{1}{m^2}$ | $(9.349 \cdot 10^4)\ \frac{1}{m^2}$ | $(1.713 \cdot 10^3)\ \frac{1}{m^2}$ |
| "I=ABDE" | $(8.79 \cdot 10^4)\ \frac{1}{m^2}$ | $(9.736 \cdot 10^4)\ \frac{1}{m^2}$ | $(9.455 \cdot 10^3)\ \frac{1}{m^2}$ |
| "J=ABCE" | $(9.739 \cdot 10^4)\ \frac{1}{m^2}$ | $(8.787 \cdot 10^4)\ \frac{1}{m^2}$ | $-9.512 \cdot 10^3\ \frac{1}{m^2}$ |

Extracting the transverse shear stiffness effect,
the difference between the high and the low avrage responses

$$H_{eff.3} := \text{submatrix}\left(H_{eff}, 1, \text{rows}\left(H_{eff}\right) - 1, 3, 3\right)$$

$$H_{eff.3} := \frac{H_{eff.3}}{\max\left(H_{eff.3}\right)}$$

$$T7 := \text{WRITEPRN}\left("\text{H3e1.txt}", H_{eff.3}\right)$$

Creating a vector containg the parameters

$$H_{eff.0} := \begin{bmatrix} "t1" \\ "tc" \\ "t2" \\ "\theta" \\ "f1" \\ "f2" \\ "h" \\ "tw" \\ "dw" \\ "R" \end{bmatrix}$$

Creating a vector of two elements that is used to generate an effects plot

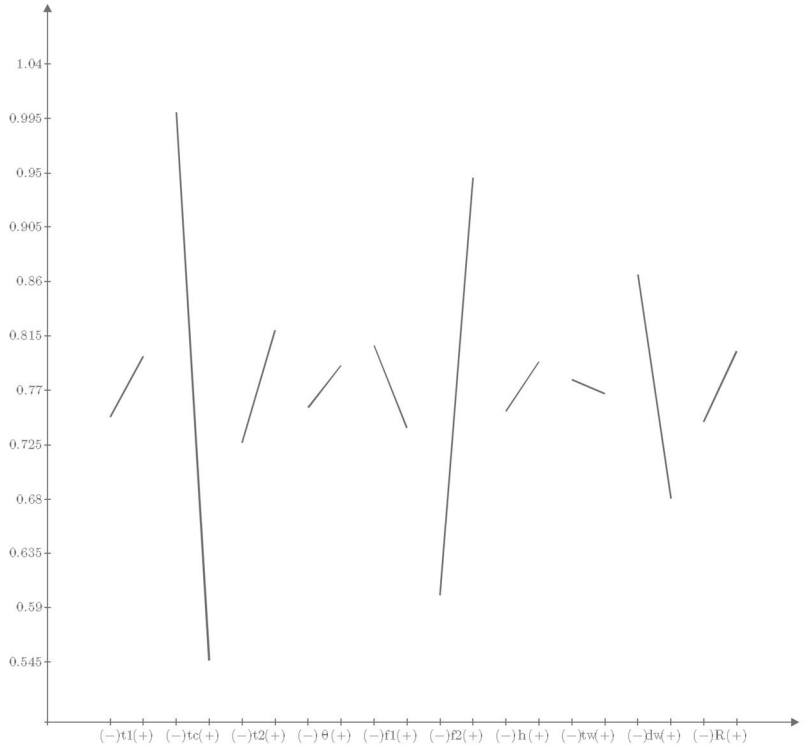
$$Q_{eff} := \text{effectsgraph}\left(H_{eff}\right)$$

$$Q_{eff_1} := \frac{Q_{eff_1}}{\max\left(Q_{eff_1}\right)}$$

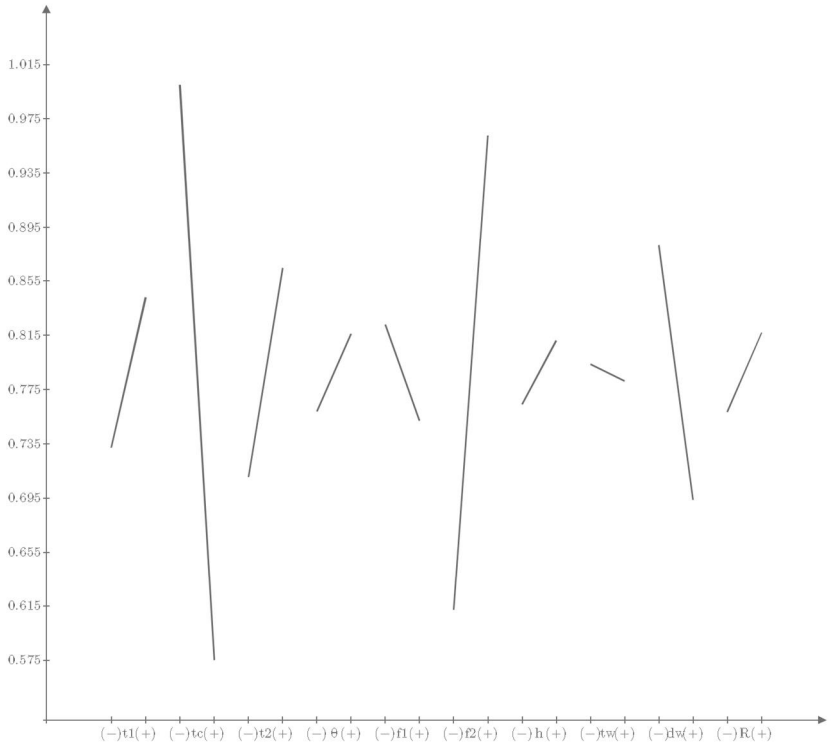
$$T8 := \text{WRITEPRN}\left(\text{"Q1e1.txt"}, Q_{eff_1}\right)$$

**Effect plots**

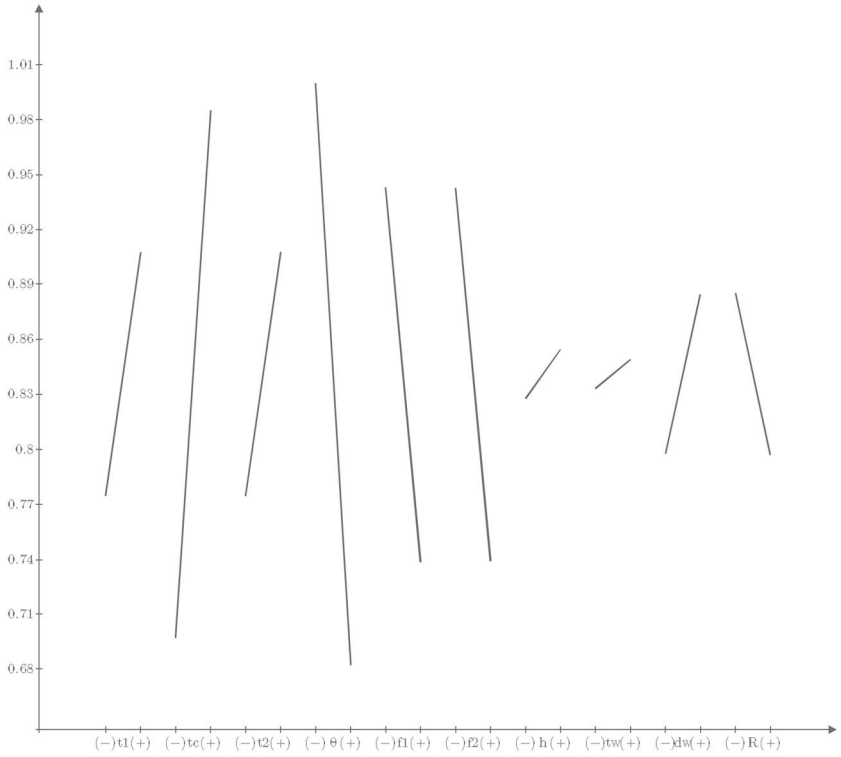
**Main effects, Principal Stress**



**Main effects, Principal Stress, including effect of mass**



**Main effects, Transverse Shear Stiffness**



**Main effects, Transverse Shear Stiffness, including effect of mass**

# Appendix C

## Mathcad calculations for comparison between one weld and two welds

**Area of cross-section (TWO WELDS)**

Reading data from files

$Data1 := \text{READPRN}(\text{“Area\_1\_2w.txt”})$

$Indata1 := \text{READPRN}(\text{“Indata\_1\_2w.txt”})$

$Data2 := \text{READPRN}(\text{“Area\_2\_2w.txt”})$

$Indata2 := \text{READPRN}(\text{“Indata\_2\_2w.txt”})$

$Data3 := \text{READPRN}(\text{“Area\_3\_2w.txt”})$

$Indata3 := \text{READPRN}(\text{“Indata\_3\_2w.txt”})$

$Data4 := \text{READPRN}(\text{“Area\_4\_2w.txt”})$

$Indata4 := \text{READPRN}(\text{“Indata\_4\_2w.txt”})$

$Data5 := \text{READPRN}(\text{“Area\_5\_2w.txt”})$

$Indata5 := \text{READPRN}(\text{“Indata\_5\_2w.txt”})$

Extracting area, distance p, and height for each model

| | |
|---|---|
| $A1 := Data1^{\langle 0 \rangle} \cdot m^2$ | Area of cross-sections |
| $p1 := Data1^{\langle 1 \rangle} \cdot m$ | Half of the with of cross-sections |
| $h1 := Indata1^{\langle 7 \rangle} \cdot m$ | Height of the cross-sections |
| $A2 := Data2^{\langle 0 \rangle} \cdot m^2$ | Area of cross-sections |
| $p2 := Data2^{\langle 1 \rangle} \cdot m$ | Half of the with of cross-sections |
| $h2 := Indata2^{\langle 7 \rangle} \cdot m$ | Height of the cross-sections |
| $A3 := Data3^{\langle 0 \rangle} \cdot m^2$ | Area of cross-sections |
| $p3 := Data3^{\langle 1 \rangle} \cdot m$ | Half of the with of cross-sections |
| $h3 := Indata3^{\langle 7 \rangle} \cdot m$ | Height of the cross-sections |
| $A4 := Data4^{\langle 0 \rangle} \cdot m^2$ | Area of cross-sections |

$p4 := Data4^{\langle 1 \rangle} \cdot \boldsymbol{m}$ Half of the with of cross-sections

$h4 := Indata4^{\langle 7 \rangle} \cdot \boldsymbol{m}$ Height of the cross-sections

$A5 := Data5^{\langle 0 \rangle} \cdot \boldsymbol{m}^2$ Area of cross-sections

$p5 := Data5^{\langle 1 \rangle} \cdot \boldsymbol{m}$ Half of the with of cross-sections

$h5 := Indata5^{\langle 7 \rangle} \cdot \boldsymbol{m}$ Height of the cross-sections

**Maximum Stress and Transverse Shear Stiffness (TWO WELDS)**

Reading data from output-file created analysing the models in Brigade 6.1

$C1 := \text{READPRN}(\text{“Output\_1\_2w.txt”})$

$C2 := \text{READPRN}(\text{“Output\_2\_2w.txt”})$

$C3 := \text{READPRN}(\text{“Output\_3\_2w.txt”})$

$C4 := \text{READPRN}(\text{“Output\_4\_2w.txt”})$

$C5 := \text{READPRN}(\text{“Output\_5\_2w.txt”})$

Extracting, from the output-data the principal stresses around the four notches in the models

$PS1\_1 := \text{augment}(C1^{\langle 3 \rangle}, C1^{\langle 2 \rangle})$ Principal stress around notch 1

$PS2\_1 := \text{augment}(C1^{\langle 8 \rangle}, C1^{\langle 7 \rangle})$ Principal stress around notch 2

$PS3\_1 := \text{augment}(C1^{\langle 13 \rangle}, C1^{\langle 12 \rangle})$ Principal stress around notch 3

$PS4\_1 := \text{augment}(C1^{\langle 18 \rangle}, C1^{\langle 17 \rangle})$ Principal stress around notch 4

$PS1\_2 := \text{augment}(C2^{\langle 3 \rangle}, C2^{\langle 2 \rangle})$ Principal stress around notch 1

$PS2\_2 := \text{augment}(C2^{\langle 8 \rangle}, C2^{\langle 7 \rangle})$ Principal stress around notch 2

$PS3\_2 := \text{augment}(C2^{\langle 13 \rangle}, C2^{\langle 12 \rangle})$ Principal stress around notch 3

$PS4\_2 := \text{augment}(C2^{\langle 18 \rangle}, C2^{\langle 17 \rangle})$ Principal stress around notch 4

$PS1\_3 := \text{augment}(C3^{\langle 3 \rangle}, C3^{\langle 2 \rangle})$ Principal stress around notch 1

$PS2\_3 := \text{augment}(C3^{\langle 8 \rangle}, C3^{\langle 7 \rangle})$ Principal stress around notch 2

$PS3\_3 := \text{augment}(C3^{\langle 13 \rangle}, C3^{\langle 12 \rangle})$ Principal stress around notch 3

$PS4\_3 := \text{augment}(C3^{\langle 18 \rangle}, C3^{\langle 17 \rangle})$ Principal stress around notch 4

$PS1\_4 := \text{augment}\left(C4^{\langle 3\rangle}, C4^{\langle 2\rangle}\right)$ Principal stress around notch 1

$PS2\_4 := \text{augment}\left(C4^{\langle 8\rangle}, C4^{\langle 7\rangle}\right)$ Principal stress around notch 2

$PS3\_4 := \text{augment}\left(C4^{\langle 13\rangle}, C4^{\langle 12\rangle}\right)$ Principal stress around notch 3

$PS4\_4 := \text{augment}\left(C4^{\langle 18\rangle}, C4^{\langle 17\rangle}\right)$ Principal stress around notch 4

$PS1\_5 := \text{augment}\left(C5^{\langle 3\rangle}, C5^{\langle 2\rangle}\right)$ Principal stress around notch 1

$PS2\_5 := \text{augment}\left(C5^{\langle 8\rangle}, C5^{\langle 7\rangle}\right)$ Principal stress around notch 2

$PS3\_5 := \text{augment}\left(C5^{\langle 13\rangle}, C5^{\langle 12\rangle}\right)$ Principal stress around notch 3

$PS4\_5 := \text{augment}\left(C5^{\langle 18\rangle}, C5^{\langle 17\rangle}\right)$ Principal stress around notch 4

Creating a matrix containing all the principal stresses

$$PS1 := \text{augment}\left(PS1\_1^{\langle 0\rangle}, PS2\_1^{\langle 0\rangle}, PS3\_1^{\langle 0\rangle}, PS4\_1^{\langle 0\rangle}\right)$$

$$PS2 := \text{augment}\left(PS1\_2^{\langle 0\rangle}, PS2\_2^{\langle 0\rangle}, PS3\_2^{\langle 0\rangle}, PS4\_2^{\langle 0\rangle}\right)$$

$$PS3 := \text{augment}\left(PS1\_3^{\langle 0\rangle}, PS2\_3^{\langle 0\rangle}, PS3\_3^{\langle 0\rangle}, PS4\_3^{\langle 0\rangle}\right)$$

$$PS4 := \text{augment}\left(PS1\_4^{\langle 0\rangle}, PS2\_4^{\langle 0\rangle}, PS3\_4^{\langle 0\rangle}, PS4\_4^{\langle 0\rangle}\right)$$

$$PS5 := \text{augment}\left(PS1\_5^{\langle 0\rangle}, PS2\_5^{\langle 0\rangle}, PS3\_5^{\langle 0\rangle}, PS4\_5^{\langle 0\rangle}\right)$$

Creating a vector containing the maximum principal stress, from the four notches, for each model

$$S1 := \left\| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS1^{\langle 0\rangle}\right) \\ \quad S_i \leftarrow \max\left(\text{submatrix}\left(PS1, i, i, 0, 3\right)\right) \\ S \end{array} \right\|$$

$$S2 := \left\| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS2^{\langle 0\rangle}\right) \\ \quad S_i \leftarrow \max\left(\text{submatrix}\left(PS2, i, i, 0, 3\right)\right) \\ S \end{array} \right\|$$

$$S3 := \left\| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS3^{\langle 0\rangle}\right) \\ \quad S_i \leftarrow \max\left(\text{submatrix}\left(PS3, i, i, 0, 3\right)\right) \\ S \end{array} \right\|$$

$$S4 := \left| \begin{array}{l} \text{for } i \in 0,1..\text{last}\left(PS4^{\langle 0 \rangle}\right) \\ \quad S_i \leftarrow \max(\text{submatrix}(PS4, i, i, 0, 3)) \\ S \end{array} \right.$$

$$S5 := \left| \begin{array}{l} \text{for } i \in 0,1..\text{last}\left(PS5^{\langle 0 \rangle}\right) \\ \quad S_i \leftarrow \max(\text{submatrix}(PS5, i, i, 0, 3)) \\ S \end{array} \right.$$

$$X1\_2w := \left| \begin{array}{l} \text{if } S1 = PS1\_1^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_1^{\langle 1 \rangle} \\ \text{else if } S1 = PS2\_1^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_1^{\langle 1 \rangle} \\ \text{else if } S1 = PS3\_1^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_1^{\langle 1 \rangle} \\ \text{else if } S1 = PS4\_1^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_1^{\langle 1 \rangle} \end{array} \right.$$

$$X2\_2w := \left| \begin{array}{l} \text{if } S2 = PS1\_2^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_2^{\langle 1 \rangle} \\ \text{else if } S2 = PS2\_2^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_2^{\langle 1 \rangle} \\ \text{else if } S2 = PS3\_2^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_2^{\langle 1 \rangle} \\ \text{else if } S2 = PS4\_2^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_2^{\langle 1 \rangle} \end{array} \right.$$

$$X3\_2w := \left| \begin{array}{l} \text{if } S3 = PS1\_3^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_3^{\langle 1 \rangle} \\ \text{else if } S3 = PS2\_3^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_3^{\langle 1 \rangle} \\ \text{else if } S3 = PS3\_3^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_3^{\langle 1 \rangle} \\ \text{else if } S3 = PS4\_3^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_3^{\langle 1 \rangle} \end{array} \right.$$

$$X4\_2w := \begin{array}{l} \text{if } S4 = PS1\_4^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_4^{\langle 1 \rangle} \\ \text{else if } S4 = PS2\_4^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_4^{\langle 1 \rangle} \\ \text{else if } S4 = PS3\_4^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_4^{\langle 1 \rangle} \\ \text{else if } S4 = PS4\_4^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_4^{\langle 1 \rangle} \end{array}$$

$$X5\_2w := \begin{array}{l} \text{if } S5 = PS1\_5^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_5^{\langle 1 \rangle} \\ \text{else if } S5 = PS2\_5^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_5^{\langle 1 \rangle} \\ \text{else if } S5 = PS3\_5^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_5^{\langle 1 \rangle} \\ \text{else if } S5 = PS4\_5^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_5^{\langle 1 \rangle} \end{array}$$

Dividing the principal stress with h/p to obtain the principal stress caused by one unit shear

$$S1\_2w := \frac{S1}{\left(\frac{h1}{p1}\right)} = \left[ 7.842 \cdot 10^{3} \right]$$

$$S2\_2w := \frac{S2}{\left(\frac{h2}{p2}\right)} = \left[ 2.874 \cdot 10^{3} \right]$$

$$S3\_2w := \frac{S3}{\left(\frac{h3}{p3}\right)} = \left[ 1.643 \cdot 10^{3} \right]$$

$$S4\_2w := \frac{S4}{\left(\frac{h4}{p4}\right)} = \left[ 3.955 \cdot 10^{3} \right]$$

$$S5\_2w := \frac{S5}{\left(\frac{h5}{p5}\right)} = \left[ 2.581 \cdot 10^{3} \right]$$

Extracting, from the output-data the transverse shear stiffness for the models

$D1\_2w_{Qy} := C1^{\langle 30 \rangle} = [\,1.531 \cdot 10^{7}\,]$

$D2\_2w_{Qy} := C2^{\langle 30 \rangle} = [\,1.912 \cdot 10^{7}\,]$

$D3\_2w_{Qy} := C3^{\langle 30 \rangle} = [\,2.45 \cdot 10^{7}\,]$

$D4\_2w_{Qy} := C4^{\langle 30 \rangle} = [\,4.008 \cdot 10^{6}\,]$

$D5\_2w_{Qy} := C5^{\langle 30 \rangle} = [\,1.546 \cdot 10^{7}\,]$

**Area of cross-section (ONE WELD)**

Reading data from files

$Data1 := \mathrm{READPRN}(\text{“Area\_1\_1w.txt”})$

$Indata1 := \mathrm{READPRN}(\text{“Indata\_1\_1w.txt”})$

$Data2 := \mathrm{READPRN}(\text{“Area\_2\_1w.txt”})$

$Indata2 := \mathrm{READPRN}(\text{“Indata\_2\_1w.txt”})$

$Data3 := \mathrm{READPRN}(\text{“Area\_3\_1w.txt”})$

$Indata3 := \mathrm{READPRN}(\text{“Indata\_3\_1w.txt”})$

$Data4 := \mathrm{READPRN}(\text{“Area\_4\_1w.txt”})$

$Indata4 := \mathrm{READPRN}(\text{“Indata\_4\_1w.txt”})$

$Data5 := \mathrm{READPRN}(\text{“Area\_5\_1w.txt”})$

$Indata5 := \mathrm{READPRN}(\text{“Indata\_5\_1w.txt”})$

Extracting area, distance p, and height for each model

| | |
|---|---|
| $A1 := Data1^{\langle 0 \rangle} \cdot \boldsymbol{m}^2$ | Area of cross-sections |
| $p1 := Data1^{\langle 1 \rangle} \cdot \boldsymbol{m}$ | Half of the with of cross-sections |
| $h1 := Indata1^{\langle 7 \rangle} \cdot \boldsymbol{m}$ | Height of the cross-sections |
| $A2 := Data2^{\langle 0 \rangle} \cdot \boldsymbol{m}^2$ | Area of cross-sections |
| $p2 := Data2^{\langle 1 \rangle} \cdot \boldsymbol{m}$ | Half of the with of cross-sections |
| $h2 := Indata2^{\langle 7 \rangle} \cdot \boldsymbol{m}$ | Height of the cross-sections |

$A3 := Data3^{\langle 0 \rangle} \cdot m^2$ Area of cross-sections

$p3 := Data3^{\langle 1 \rangle} \cdot m$ Half of the with of cross-sections

$h3 := Indata3^{\langle 7 \rangle} \cdot m$ Height of the cross-sections

$A4 := Data4^{\langle 0 \rangle} \cdot m^2$ Area of cross-sections

$p4 := Data4^{\langle 1 \rangle} \cdot m$ Half of the with of cross-sections

$h4 := Indata4^{\langle 7 \rangle} \cdot m$ Height of the cross-sections

$A5 := Data5^{\langle 0 \rangle} \cdot m^2$ Area of cross-sections

$p5 := Data5^{\langle 1 \rangle} \cdot m$ Half of the with of cross-sections

$h5 := Indata5^{\langle 7 \rangle} \cdot m$ Height of the cross-sections

**Maximum Stress and Transverse Shear Stiffness (ONE WELD)**

Reading data from output-file created analysing the models in Brigade 6.1

$C1 := \text{READPRN}(\text{“Output\_1\_1w.txt”})$

$C2 := \text{READPRN}(\text{“Output\_2\_1w.txt”})$

$C3 := \text{READPRN}(\text{“Output\_3\_1w.txt”})$

$C4 := \text{READPRN}(\text{“Output\_4\_1w.txt”})$

$C5 := \text{READPRN}(\text{“Output\_5\_1w.txt”})$

Extracting, from the output-data the principal stresses around the four notches in the models

$PS1\_1 := \text{augment}(C1^{\langle 3 \rangle}, C1^{\langle 2 \rangle})$ Principal stress around notch 1

$PS2\_1 := \text{augment}(C1^{\langle 8 \rangle}, C1^{\langle 7 \rangle})$ Principal stress around notch 2

$PS1\_2 := \text{augment}(C2^{\langle 3 \rangle}, C2^{\langle 2 \rangle})$ Principal stress around notch 1

$PS2\_2 := \text{augment}(C2^{\langle 8 \rangle}, C2^{\langle 7 \rangle})$ Principal stress around notch 2

$PS1\_3 := \text{augment}(C3^{\langle 3 \rangle}, C3^{\langle 2 \rangle})$ Principal stress around notch 1

$PS2\_3 := \text{augment}(C3^{\langle 8 \rangle}, C3^{\langle 7 \rangle})$ Principal stress around notch 2

$PS1\_4 := \text{augment}\left(C4^{\langle 3 \rangle}, C4^{\langle 2 \rangle}\right)$ Principal stress around notch 1

$PS2\_4 := \text{augment}\left(C4^{\langle 8 \rangle}, C4^{\langle 7 \rangle}\right)$ Principal stress around notch 2

$PS1\_5 := \text{augment}\left(C5^{\langle 3 \rangle}, C5^{\langle 2 \rangle}\right)$ Principal stress around notch 1

$PS2\_5 := \text{augment}\left(C5^{\langle 8 \rangle}, C5^{\langle 7 \rangle}\right)$ Principal stress around notch 2

Creating a matrix containing all the principal stresses

$$PS1 := \text{augment}\left(PS1\_1^{\langle 0 \rangle}, PS2\_1^{\langle 0 \rangle}\right)$$

$$PS2 := \text{augment}\left(PS1\_2^{\langle 0 \rangle}, PS2\_2^{\langle 0 \rangle}\right)$$

$$PS3 := \text{augment}\left(PS1\_3^{\langle 0 \rangle}, PS2\_3^{\langle 0 \rangle}\right)$$

$$PS4 := \text{augment}\left(PS1\_4^{\langle 0 \rangle}, PS2\_4^{\langle 0 \rangle}\right)$$

$$PS5 := \text{augment}\left(PS1\_5^{\langle 0 \rangle}, PS2\_5^{\langle 0 \rangle}\right)$$

Creating a vector containing the maximum principal stress, from the four notches, for each model

$$S1 := \left| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS1^{\langle 0 \rangle}\right) \\ \quad S_i \leftarrow \max\left(\text{submatrix}\left(PS1, i, i, 0, 1\right)\right) \\ S \end{array} \right.$$

$$S2 := \left| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS2^{\langle 0 \rangle}\right) \\ \quad S_i \leftarrow \max\left(\text{submatrix}\left(PS2, i, i, 0, 1\right)\right) \\ S \end{array} \right.$$

$$S3 := \left| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS3^{\langle 0 \rangle}\right) \\ \quad S_i \leftarrow \max\left(\text{submatrix}\left(PS3, i, i, 0, 1\right)\right) \\ S \end{array} \right.$$

$$S4 := \left| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS4^{\langle 0 \rangle}\right) \\ \quad S_i \leftarrow \max\left(\text{submatrix}\left(PS4, i, i, 0, 1\right)\right) \\ S \end{array} \right.$$

$$S5 := \left| \begin{array}{l} \text{for } i \in 0, 1 .. \text{last}\left(PS5^{\langle 0 \rangle}\right) \\ \quad S_i \leftarrow \max\left(\text{submatrix}\left(PS5, i, i, 0, 1\right)\right) \\ S \end{array} \right.$$

$$X1\_1w := \begin{array}{l} \text{if } S1 = PS1\_1^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_1^{\langle 1 \rangle} \\ \text{else if } S1 = PS2\_1^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_1^{\langle 1 \rangle} \\ \text{else if } S1 = PS3\_1^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_1^{\langle 1 \rangle} \\ \text{else if } S1 = PS4\_1^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_1^{\langle 1 \rangle} \end{array}$$

$$X2\_1w := \begin{array}{l} \text{if } S2 = PS1\_2^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_2^{\langle 1 \rangle} \\ \text{else if } S2 = PS2\_2^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_2^{\langle 1 \rangle} \\ \text{else if } S2 = PS3\_2^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_2^{\langle 1 \rangle} \\ \text{else if } S2 = PS4\_2^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_2^{\langle 1 \rangle} \end{array}$$

$$X3\_1w := \begin{array}{l} \text{if } S3 = PS1\_3^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_3^{\langle 1 \rangle} \\ \text{else if } S3 = PS2\_3^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_3^{\langle 1 \rangle} \\ \text{else if } S3 = PS3\_3^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_3^{\langle 1 \rangle} \\ \text{else if } S3 = PS4\_3^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_3^{\langle 1 \rangle} \end{array}$$

$$X4\_1w := \begin{array}{l} \text{if } S4 = PS1\_4^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_4^{\langle 1 \rangle} \\ \text{else if } S4 = PS2\_4^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_4^{\langle 1 \rangle} \\ \text{else if } S4 = PS3\_4^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_4^{\langle 1 \rangle} \\ \text{else if } S4 = PS4\_4^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_4^{\langle 1 \rangle} \end{array}$$

$$X5\_1w := \begin{array}{l} \text{if } S5 = PS1\_5^{\langle 0 \rangle} \\ \quad X \leftarrow PS1\_5^{\langle 1 \rangle} \\ \text{else if } S5 = PS2\_5^{\langle 0 \rangle} \\ \quad X \leftarrow PS2\_5^{\langle 1 \rangle} \\ \text{else if } S5 = PS3\_5^{\langle 0 \rangle} \\ \quad X \leftarrow PS3\_5^{\langle 1 \rangle} \\ \text{else if } S5 = PS4\_5^{\langle 0 \rangle} \\ \quad X \leftarrow PS4\_5^{\langle 1 \rangle} \end{array}$$

Dividing the principal stress with h/p to obtain the principal stress caused by one unit shear

$$S1\_1w := \frac{S1}{\left(\frac{h1}{p1}\right)} = [2.249 \cdot 10^4]$$

$$S2\_1w := \frac{S2}{\left(\frac{h2}{p2}\right)} = [7.956 \cdot 10^3]$$

$$S3\_1w := \frac{S3}{\left(\frac{h3}{p3}\right)} = [5.606 \cdot 10^3]$$

$$S4\_1w := \frac{S4}{\left(\frac{h4}{p4}\right)} = [1.235 \cdot 10^4]$$

$$S5\_1w := \frac{S5}{\left(\frac{h5}{p5}\right)} = [1.012 \cdot 10^4]$$

Extracting, from the output-data the transverse shear stiffness for the models

$$D1\_1w_{Qy} := C1^{\langle 18 \rangle} = [6.039 \cdot 10^6]$$

$$D2\_1w_{Qy} := C2^{\langle 18 \rangle} = [1.617 \cdot 10^7]$$

$$D3\_1w_{Qy} := C3^{\langle 18 \rangle} = [2.227 \cdot 10^7]$$

$$D4\_1w_{Qy} := C4^{\langle 18 \rangle} = [3.377 \cdot 10^6]$$

$$D5\_1w_{Qy} := C5^{\langle 18 \rangle} = [1.051 \cdot 10^7]$$

**Comparison**

Model 1:

$S1\_1w = [2.249 \cdot 10^4]$

$S1\_2w = [7.842 \cdot 10^3]$

$\frac{S1\_2w}{S1\_1w} = 0.349$

$X1\_1w = [0.537]$

$X1\_2w = [0.707]$

$D1\_1w_{Qy} = [6.039 \cdot 10^6]$

$D1\_2w_{Qy} = [1.531 \cdot 10^7]$

$\frac{D1\_2w_{Qy}}{D1\_1w_{Qy}} = 2.535$

Model 2:

$S2\_1w = [7.956 \cdot 10^3]$

$S2\_2w = [2.874 \cdot 10^3]$

$\frac{S2\_2w}{S2\_1w} = 0.361$

$X2\_1w = [0.524]$

$X2\_2w = [0.286]$

$D2\_1w_{Qy} = [1.617 \cdot 10^7]$

$D2\_2w_{Qy} = [1.912 \cdot 10^7]$

$\frac{D2\_2w_{Qy}}{D2\_1w_{Qy}} = 1.183$

Model 3:

$S3\_1w = [5.606 \cdot 10^3]$

$S3\_2w = [1.643 \cdot 10^3]$

$\frac{S3\_2w}{S3\_1w} = 0.293$

$X3\_1w = [0.548]$

$X3\_2w = [0.333]$

$D3\_1w_{Qy} = [2.227 \cdot 10^7]$

$D3\_2w_{Qy} = [2.45 \cdot 10^7]$

$\frac{D3\_2w_{Qy}}{D3\_1w_{Qy}} = 1.1$

Model 4:

$S4\_1w = [1.235 \cdot 10^4]$

$S4\_2w = [3.955 \cdot 10^3]$

$\frac{S4\_2w}{S4\_1w} = 0.32$

$X4\_1w = [0.512]$

$X4\_2w = [0.262]$

$D4\_1w_{Qy} = [3.377 \cdot 10^6]$

$D4\_2w_{Qy} = [4.008 \cdot 10^6]$

$\frac{D4\_2w_{Qy}}{D4\_1w_{Qy}} = 1.187$

Model 5:

$S5\_1w = [\,1.012 \cdot 10^{4}\,]$

$S5\_2w = [\,2.581 \cdot 10^{3}\,]$

$\frac{S5\_2w}{S5\_1w} = 0.255$

$X5\_1w = [\,0.524\,]$

$X5\_2w = [\,0.262\,]$

$D5\_1w_{Qy} = [\,1.051 \cdot 10^{7}\,]$

$D5\_2w_{Qy} = [\,1.546 \cdot 10^{7}\,]$

$\frac{D5\_2w_{Qy}}{D5\_1w_{Qy}} = 1.47$