Delft University of Technology
Faculty of Civil Engineering & Geosciences
Department of Structural Mechanics

Automatic Buckling Checks on Stiffened Panels Based on Finite Element Results

Master Thesis Report

October 2011

Student:
Ottar Hillers

Graduation Committee:
Prof. ir. F.S.K. Bijlaard
Dr. ir. P.C.J. Hoogenboom
Ir. W. van den Bos

TU Delft Delft University of Technology

femto engineering

Abstract

In this thesis a post processing tool for finite element analysis was developed to perform buckling checks on stiffened steel panels. The tool can perform buckling checks on rectangular, orthogonal stiffened plates including different panel sizes and openings. The procedure is completely automatic and is consequently conducive to reduction of engineering time. The tool detects geometrical and material properties from a finite element model and determines design loads based on stress results of a finite element analysis. The approach is in accordance with guidelines from design codes and therefore results can be considered to be verified according to the design code in question. The tool has been adapted to the American Bureau of Shipping guide for buckling and ultimate strength assessment for offshore structures. The tool is compared to the ABS plate buckling tool of the commercial software SDC Verifier. Results show that the developed tool does not need as fine finite element mesh as the ABS plate buckling tool of the SDC Verifier to predict accurate buckling factors. Furthermore for general cases up to 25% reduction of buckling factors can be obtained with the developed tool compared to the SDC Verifier.

Acknowledgment

I would like to show my gratitude to my thesis supervisors, Prof. ir. Frans Bijlaard, Dr. ir. Pierre Hoogenboom and Ir. Wouter van den Bos, for their guidance and support throughout the process of my project.

I would also like to thank Femto Engineering for the collaboration. It has truly been a pleasure to carry out my work at their office and I highly appreciate the assistance from the staff at Femto Engineering. I would like to give special thanks to Alexander Naatje for his help with the Femap programming.

Lastly, I offer my regards to all my friends and family who supported me during the completion of the project.

Ottar Hillers

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 General

Stiffened panels are common structural elements used in various fields of engineering. The high strength-to-weight ratio is important for structures where self weight has to be limited. Examples of application include aircrafts, ships, steel girder bridges and flood barriers (see figure 1.1 [1], [2], [3], [4]).

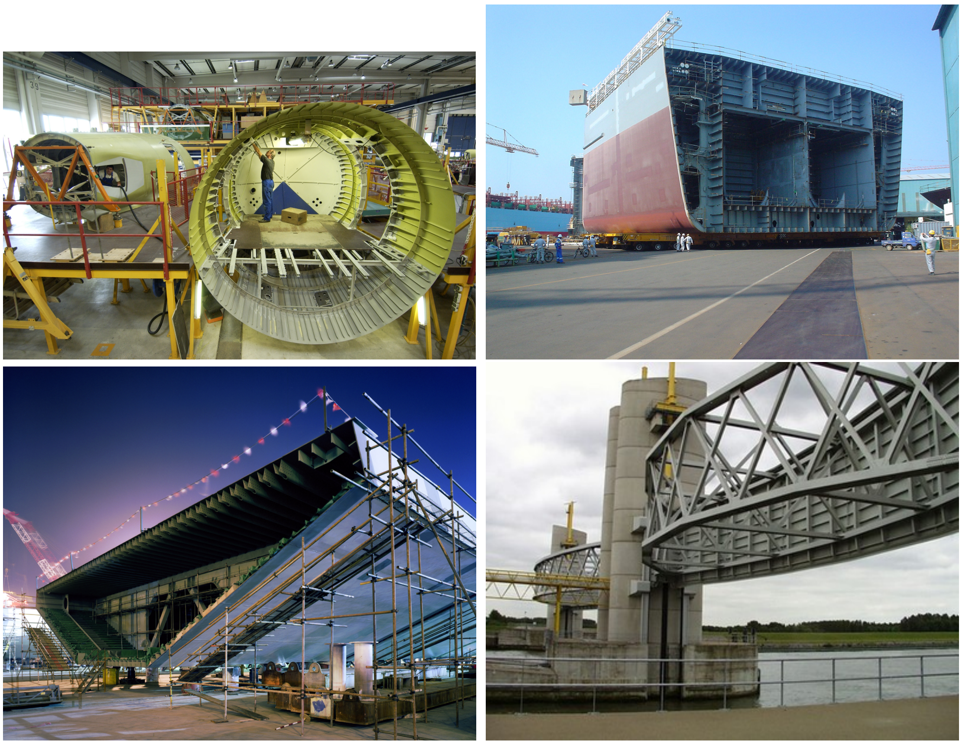


Figure 1.1: Examples of stiffened panels

The characteristics of a typical stiffened panel are shown in figure 1.2 [5]. These characteristics are flat rectangular plate with equally spaced longitudinal stiffeners, supported by larger and more widely spaced girders in both transverse and longitudinal direction [6]. The stiffeners and the girders usually have either angle shaped or T-shaped cross section although other shapes exists.

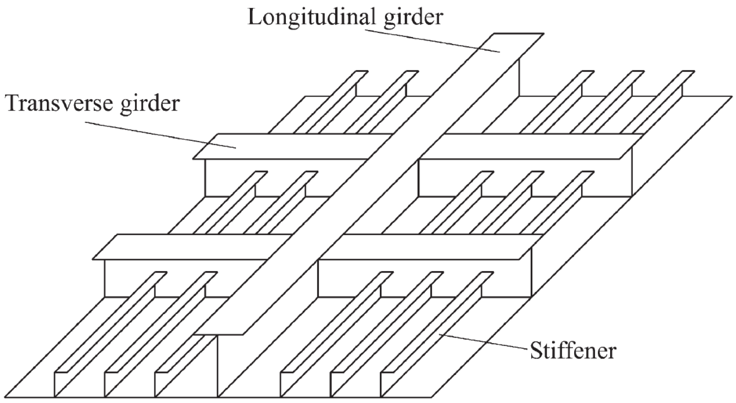


Figure 1.2: Schematic view of stiffened panel

General loading cases for the panels consist of combination of longitudinal, transverse and shear stresses along with lateral pressure. Due to the loading the panel can buckle in several ways, which can be categorised into local- or global buckling modes. The local buckling modes are the following:

1. **Plate buckling** between the stiffeners before the failure of the stiffeners.
2. **Torsional buckling** of the stiffeners.
3. **Local buckling** of the stiffeners web and flanges.

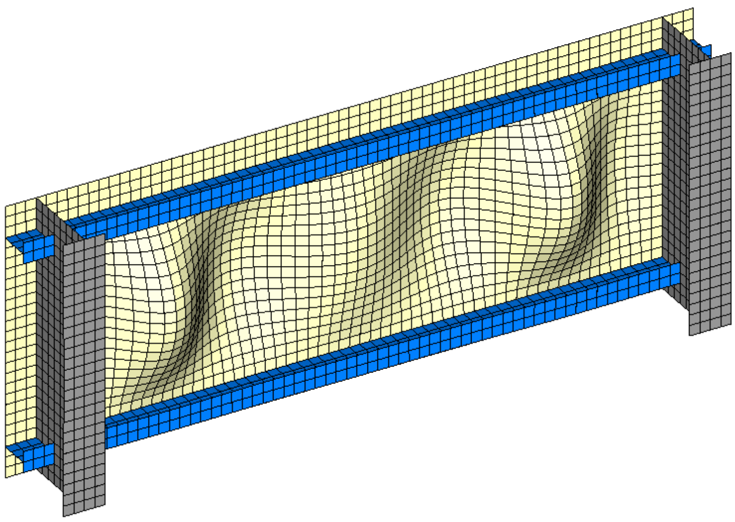


Figure 1.3: Plate buckling

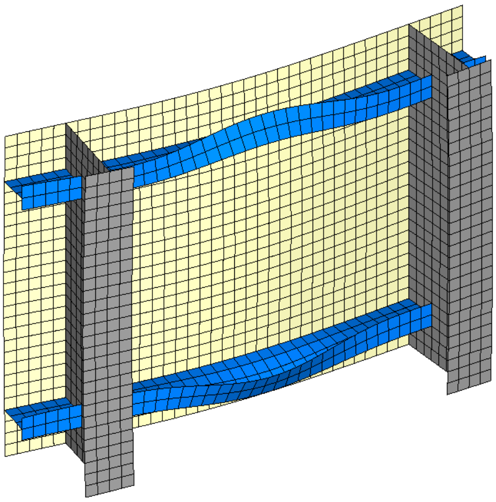


Figure 1.4: Torsional buckling

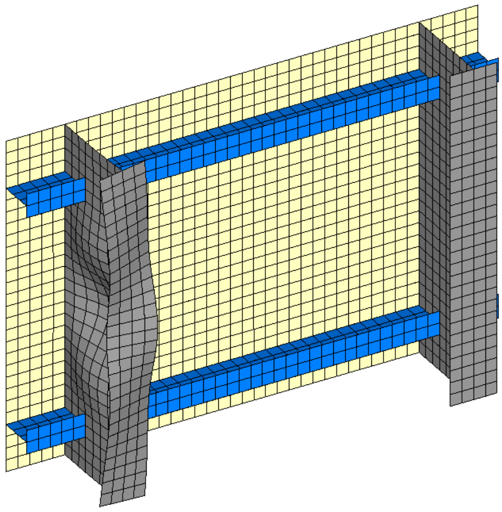


Figure 1.5: Local buckling

The global buckling modes are the following:

4. **Beam-column buckling** of the stiffener and the plate in a combined mode. The composite panel can either buckle toward the stiffener or toward the plate.

5. **Grillage buckling** where the supporting girders loses their stability.

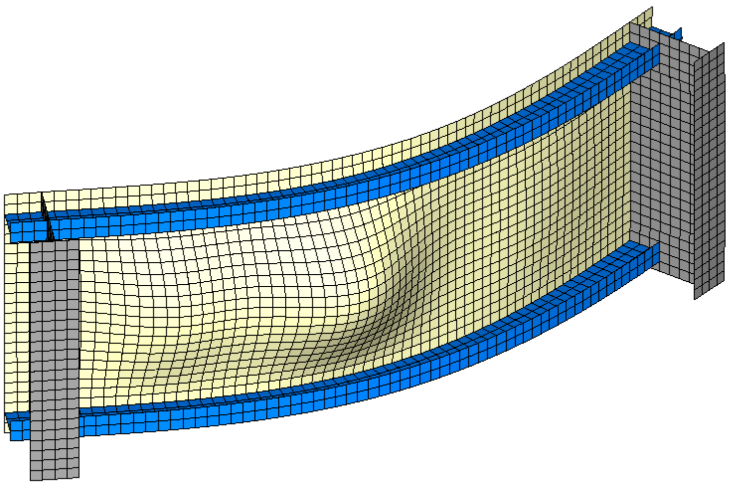


Figure 1.6: Beam-column buckling

Figure 1.6 shows a beam-column buckling mode where the composite panel buckles toward the stiffener. It can also been seen that the plated panel has buckled locally.

1.2 Problem Statement

Depending on the application of the stiffened panel, numerous load combinations have to be taken into account when designing and analysing the structure. Buckling analysis needs to be performed for each load combination and for a large structure, it can be a tremendously time consuming procedure. There are several methods used to perform buckling analysis and all of them have their advantages and disadvantages. These methods include:

a) Stress checks according to design codes.

b) Linear buckling analysis (eigenvalue analysis).

c) Non-linear analysis.

d) Physical tests

1.2.1 Stress Checks

The goal of stress checks according to design codes is to simplify the calculation procedure in a conservative way. Linear stress analysis is usually performed on the structure with the aid of computer software and stresses are compared to the rules of the design codes. The stress checking procedure can be semi-automatic, meaning that results from the computer software are manually imported to a calculation sheet like Excel or Mathcad. This method works well when the size of the structure and number of load combinations are limited.

1.2.2 Linear Buckling Analysis

Computer software can be used to perform linear buckling analysis of an ideal linear elastic structure. Generally the lowest buckling factor determines the fraction between the theoretical buckling load (the bifurcation point) and the applied load. Due to imperfections and non-linearities in real structures the theoretical buckling load normally yields unconservative results [6].

Since the buckling factor is related to all applied load this method can be troublesome if there are more than one type of loading and the relation between the loadings is not linear. For example if one type of the loads is constant (gravity load, water pressure) and another type of load is a live load the results indicates that the constant loads should also be multiplied by the buckling factor. Iterative procedure is therefore needed to determine the real buckling factor [7]. However for such load combination the buckling factor can be used as an indicator of safety, without determining the theoretical buckling load.

For stiffened panels, local buckling of the plates is sometimes allowed if the adjacent structural members can withstand the load. It is therefore likely when performing linear buckling analysis on a large structure that the lowest buckling factors belong to the local buckling modes and it might be difficult to detect the global buckling failure mode. Linear buckling analysis is also unable to predict the post buckling behaviour.

1.2.3 Non-Linear Analysis

Non-linear analysis can predict accurately the buckling load and the post buckling behaviour of structures. The non-linearities can be either geometrical (e.g. non-linear relation based on small strains) or physical (e.g. non-linear elastic or plastic material behaviour) [8]. Furthermore non-linear analysis has the potential to take into account geometrical imperfections and residual stresses. However to analyse a large model, excessive computational resources are required. This method is therefore not practical to analyse a large model but is instead useful when accurate behaviour of a critical area is required. However, computer capacity is continuously increasing so geometrical non-linear analysis can be expected to be more common approach in the future for large models.

1.2.4 Physical Tests

Physical tests can be performed on structures to investigate their performances. The tests can be performed on a scaled down model, full scaled model or part of the structure. The benefits with physical tests is that the actual behaviour of the structure can be analysed. However, for many cases, physical testing might not be economical feasible and due to construction deviations results from one test might differ from results from another test on the same type of structure and loading.

1.3 Objective and Approach

The primary objective of this research is to propose a standard calculation procedure to perform buckling analysis on stiffened panels which drastically reduces engineering time without using excessive computer resources.

Preferable solution for practise is to implement procedure into finite element software which compares linear stress results to design codes. The benefit from such approach is that results are verified by the applied design codes. Although the equations for the buckling checks are already known, the greatest challenge is to make the software recognise the input parameters for the equations.

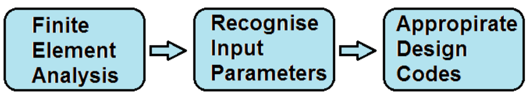


Figure 1.7: Flow chart for automatic buckling stress check

Since this thesis is done in collaboration with Femto Engineering, the procedure will be compared to their existing buckling tool. Furthermore results will also be verified by comparing them to examples from design code. Finally results of the proposed method will be compared to linear and and non-linear buckling analysis.

1.4 State of the Art

1.4.1 Design Codes

There are various design codes which give guidance for designing plated structures. *Eurocode 3, part 1.5 - Plated structural elements* gives design requirements of stiffened and unstiffened plates which are subjected to in-plane forces [9]. The main focuses are I-section girders and box girders.

American Bureau of Shipping (ABS) *Guide for buckling and ultimate strength assessment for offshore structures* provides formulation to assess buckling criteria of plates and stiffened panels [6].

Det Norske Veritas (DNV) *Recommended practise DNV-RP-C201: Buckling strength of plated structures* is a buckling code for stiffened and unstiffened panels of steel [10].

A rough distinction can be made between the Eurocode 3 guide in one hand and the ABS and DNV guides on the other hand. Eurocode 3 part 1.5 focuses on structural elements where the primary functionality is beam behaviour but due to geometrical aspects the elements have to be analysed as plates as well. The ABS and DNV guides focus however on offshore structures where the primary functionality of the structural elements is plate behaviour.

1.4.2 Computer Software

There are numerous computer programs available which offer plate buckling checks according to design codes. Often these programs are associated with certain finite element software. Licence for these software are usually expensive so therefore limited number of software were investigated for this thesis.

SDC (Structural Design Code) Verifier is a post processing software associated with the finite element software Femap (Finite Element Modelling And Postprocessing). It can perform plate buckling checks according to the ABS design code on unstiffened panels. The user manually determines the panel sizes and selects the finite elements which belong to the panels. Results are based on comparing the stress results of every finite element as if they had the same size as the panels.

Plate Buckling is a tool associated with the finite element software RSTAB which was developed by the company Dlubal. It offers plate buckling checks according to DIN 18800-3:1990-11. In the demo version of RSTAB the user has to define manually the material data, panel dimensions and boundary loading.

Platework is a program developed by DNV to perform code checking of plane stiffened steel plate structures [11]. It is associated with the software Genie. Properties and loads of specified plate area are automatically extracting from finite element analysis.

PULS (Panel Ultimate Limit State) is a computerised semi-analytical model for buckling assessment of plated structures developed by DNV [10]. It is officially part of the DNV buckling guide.

1.5 Contents

The main body of this report is divided as follows. Chapter 2 describes how a post processing tool (Stress Check Model) for finite element analysis was developed to perform buckling checks on stiffened panels. Chapter 3 describes how the Stress Check Model is adapted to the ABS design guide. In chapter 4 the functionality of the Stress Check Model is demonstrated. Finally in chapter 5 there is a discussion about the Stress Check Model and recommendations for further work.

Chapter 2

Methodology

This chapter describes how the Stress Check Model for stiffened panels is assembled using information from finite element model. The first section explains the difference between finite element model and the Stress Check Model. Limitations regarding element types and geometry are discussed. The second section describes in details how geometrical parameters for the Stress Check Model are determined automatically. Finally the third section explains how loads are applied automatically on the Stress Check Model based on stress results from finite element analysis.

2.1 General

2.1.1 Finite Element Model and Stress Check Model

Figure 2.1 shows a finite element model of a stiffened panel and typical stress results. Table 2.1 shows the parameters which are needed from the finite element model to perform buckling stress checks on the structure according to design codes. These parameters can be categorised into four groups, geometry of the panels, geometry of the stiffeners, material properties and loading.

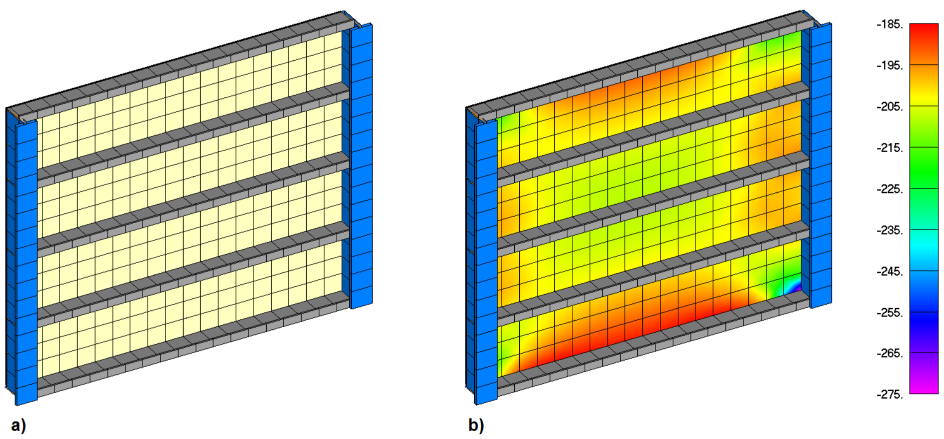


Figure 2.1: a) Typical finite element model, b) Typical stress results

Table 2.1: Buckling stress check parameters

Geometry of Panels	Geometry of Stiffeners	Material Properties	Loading
Length Width Thickness	Length Web Height Web Thickness Flange Width Flange Thickness	Young's Modulus Poisson's Ration Yield Stress	Max Longitudinal Stresses Min Longitudinal Stresses Max Transverse Stresses Min Transverse Stresses Shear Stresses Lateral Pressure

For this thesis a post processing tool, Stress Check Model, was developed to obtain the parameters described in table 2.1. Figure 2.2 shows how the Stress Check Model defines and numerates the panels, stiffeners and girders of the finite element model showed in figure 2.1.

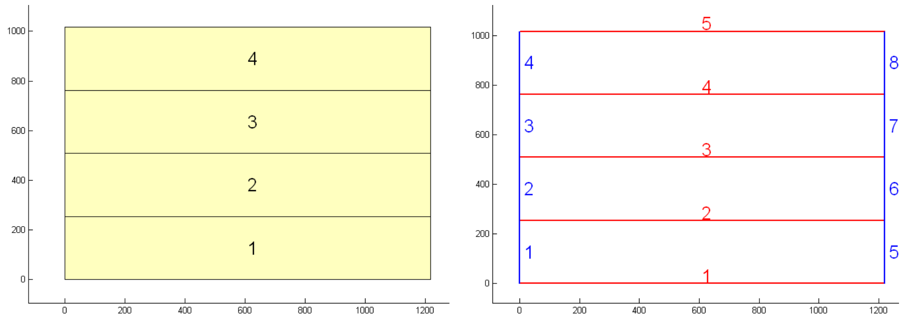


Figure 2.2: Numbering of panels, stiffeners and girders according to the Stress Check Model

When the finite element model in figure 2.1 is compared with the Stress Check Model in figure 2.2 it can be seen that both models are built up with several types of elements. Table 2.2 shows an overview of different elements and their meaning which is used in this thesis.

Table 2.2: Element definition

Finite Element Model	
Element name	**Description**
Plate element	Plate element from a finite element model
Beam element	Line element from a finite element model
Stress Check Model	
Element name	**Description**
Panel element	Panel which boundaries are defined by stiffener and girder elements
Stiffener element	Longitudinal element
Girder element	Transverse element

Notice that for the Stress Check Model longitudinal elements (red elements in figure 2.2) are called stiffeners and transverse elements (blue elements in figure 2.2) are called girders. This definition is only used for the sake of simplicity but in reality the orientation of stiffeners can be either longitudinal or transverse and the same holds for girders.

2.1.2 Limitations

The functionality of the Stress Check Model has certain limitations. These limitations are related to the element types used in the finite element model and the geometry of the model.

Element Types

The Stress Check Model can only recognise four noded plate elements and two noded beam elements (see figure 2.3).

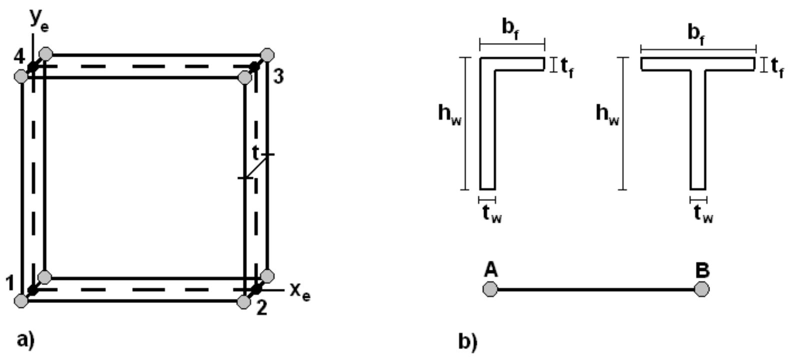


Figure 2.3: a) Four noded flat shell element with thickness t in element coordinate system, b) Beam element and cross section properties

The plate elements are indeed flat shell elements meaning they can resist bending. Stress results are therefore determined at both top- and bottom fibre of the elements and the results are extrapolated to the corner position (see the gray nodes in figure 2.3a). The beam elements can have either T-shaped or L- shaped cross section.

When stiffeners and girders are modelled with beam elements it is possible to determine the cross section properties automatically. However when beams are modelled with plate elements as showed in figure 2.4b the cross section properties cannot be determined. See further discussion in sections 2.2.1 and 5.3.

Geometry

The geometry of the Stress Check Model is limited to flat, rectangular stiffened plate field, oriented in the in the x-y plane. Longitudinal stiffeners and/or girders are oriented in the x-direction and transverse stiffeners and/or girders are oriented in the y-direction. Figure 2.5 shows in a schematic way the geometrical features which the Stress Check Model can comprehend.

The Stress Check Model is limited to single plate field oriented in the x-y plane with reference value of the z-coordinates set to zero. As an example

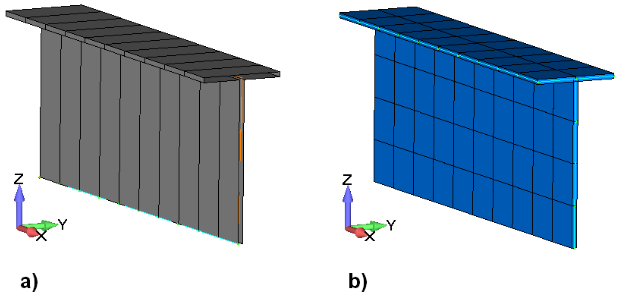


Figure 2.4: a) Beam modelled with beam elements, b) Beam modelled with plate elements

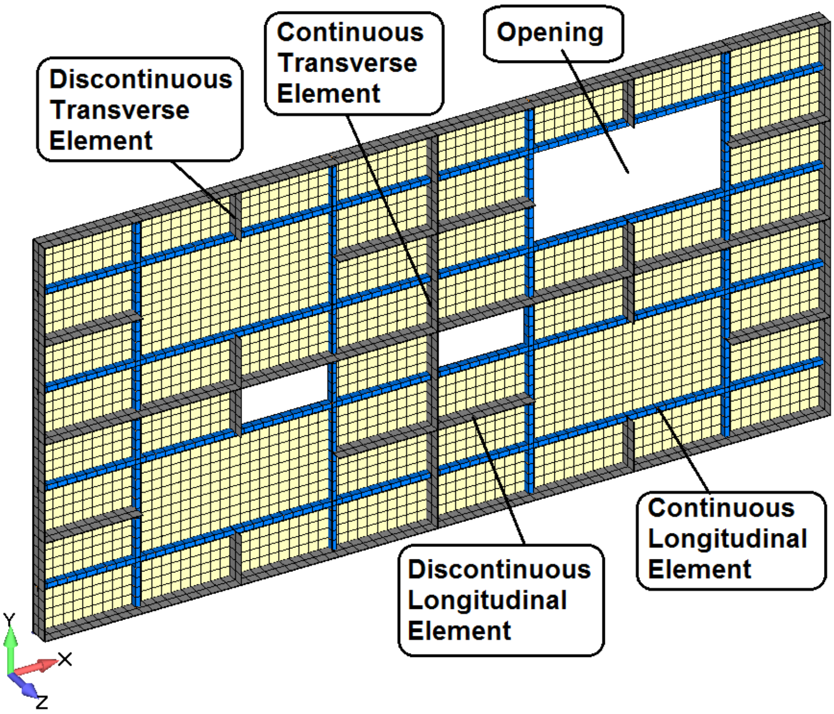


Figure 2.5: Allowable geometrical characteristics for the Stress Check Model

the plate field located in the reference plane z_0 in figure 2.6 can be analysed but the other two plate fields located in the reference systems z_1 and z_2 will be ignored. Further discussion about the reference planes can be found in section 5.3.

The Stress Check Model does not allow unsupported edges of panels, herewith called free edges (see figure 2.7). Further discussion about free edges can be found in section 5.3.

2.1.3 Femap

The programming of the Stress Check Model is adjusted to the finite element software Femap. The programming is done with MATLAB where results

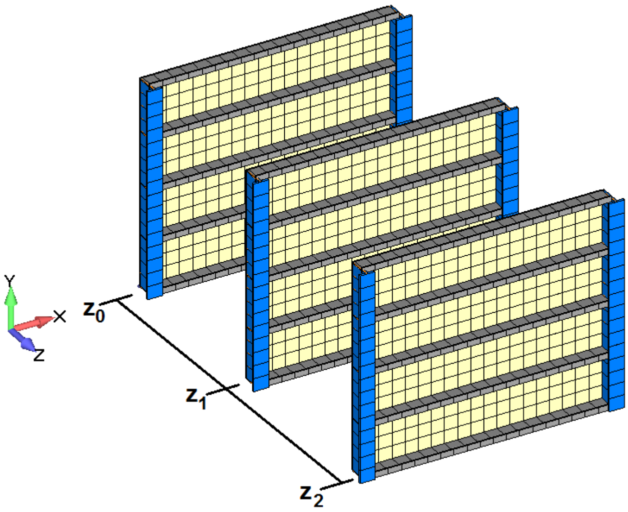


Figure 2.6: Stiffened panels in more than one plane

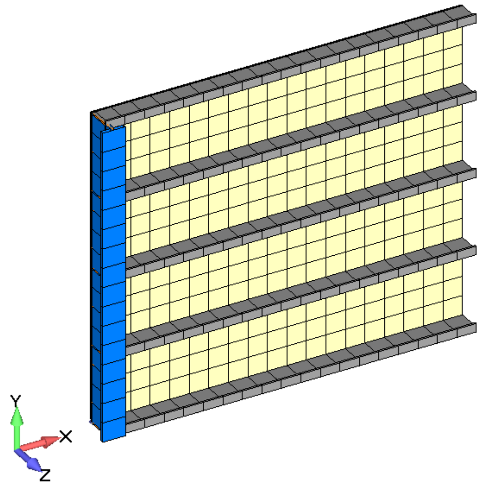


Figure 2.7: Stiffened panel with one edge unsupported

from Femap are imported as text files to MATLAB. Appendix C describes in details the procedure of importing results to MATLAB. The outputs are matrices containing the following information.

- Coordinates of plate elements (x_1, y_1, z_1, x_2, y_2, z_2, x_3, y_3, z_3, x_4, y_4 z_4)
- Coordinates of beam elements (x_A, y_A, z_A, x_B, y_B, z_B)
- Material and cross section properties of plate elements (E, ν, σ_{yield}[1], t)
- Material and cross section properties of beam elements (E, ν, σ_{yield}, h_w, b_f, t_w, t_f)

[1]The yield stress is not defined for linear static analysis in Femap so users input is required

- Stresses at corner nodes of plate element and lateral pressure acting on plate element ($C_{1,x}$, $C_{1,y}$, $C_{1,xy}$, $C_{2,x}$, $C_{2,y}$, $C_{2,xy}$, $C_{3,x}$, $C_{3,y}$, $C_{3,xy}$, $C_{4,x}$, $C_{4,y}$, $C_{4,xy}$, q)

No stress results are obtained for the beam elements. For each node and each stress component of the plate elements results are given at the top and bottom fibre. The user is given two choices to determine the stresses in the elements based on either the mean values of the cross section or the maximum compressive value of the top and bottom fibre. In case of mean stresses linear distribution is assumed. If maximum compressive value is chosen but the whole cross section is in tension the minimum tensile value is chosen. Figure 2.8 shows stresses at one corner point and the two options to determine the representing stress value for the point.

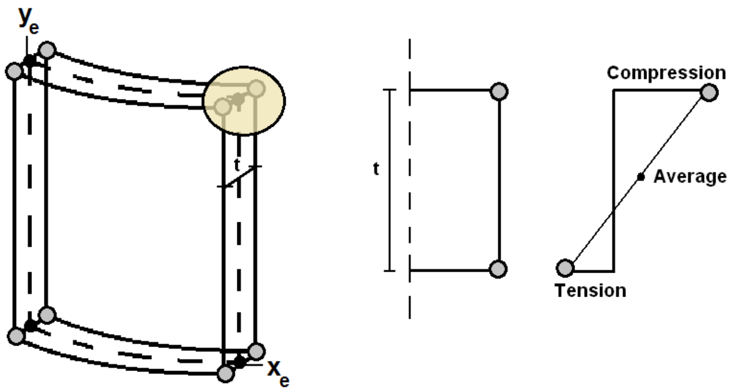


Figure 2.8: Bending action within plate element

2.2 Geometrical and Material Parameters

This section describes in details the strategy to determine the width and lengths of the panels and how to obtain all necessary geometrical and material properties used for buckling stress check. The output matrices from this section are the following:

- $\text{Panel}_{\text{information}}$(Panel id, Length, Width, E, ν, t)
- $\text{Stiffener}_{\text{information}}$(Stiffener id, Length, E, ν, h_w, b_f, t_w, t_f)
- $\text{Girder}_{\text{information}}$(Girder id, Length, E, ν, h_w, b_f, t_w, t_f)

2.2.1 Nodes Attached to Stiffeners and Girders

The first step is to define whether the beam elements from the finite element model are part of stiffener element or girder element. It is done by measuring the distances in x- and y directions between the two nodes which define the beam elements. If the x-distance is equal to zero (or relatively small number) the element can be sorted as a part of a girder element and if the y-distance is equal to zero (or relatively small number) the element can be sorted as a part of a stiffener element (see figure 2.9).

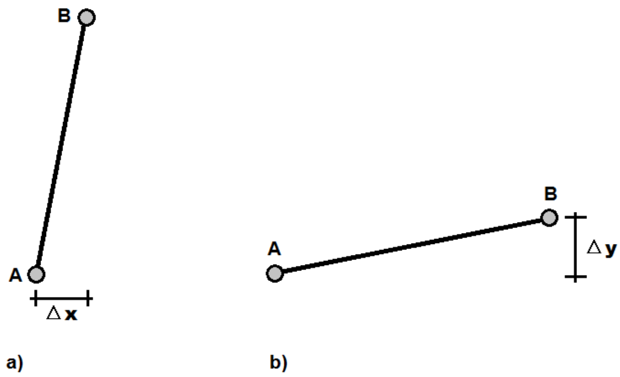


Figure 2.9: a) Transverse finite element, b) Longitudinal finite element

In the cases where beams are modelled with plate elements the plate elements which have only two nodes located in the reference plane are found. Figure 2.10 shows part of a stiffened plate panel where a girder is modelled with plate elements (blue elements). The plate elements of the girder which have two nodes in the reference plane (the red nodes) are detected. These elements are then sorted into either a part of a girder element or a part of a stiffener element, depending on the coordinates of the red nodes. The rest of the plate elements of the beam are ignored (those which are only attached to grey nodes in figure 2.10).

At this moment two groups of elements have been defined:

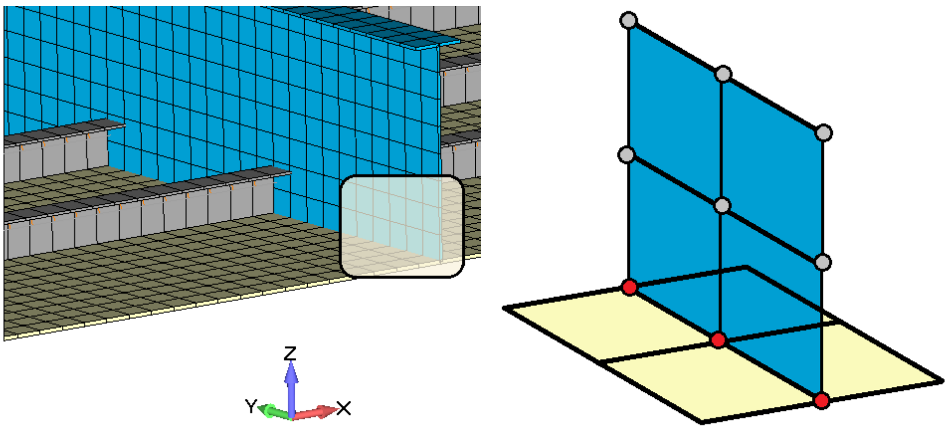


Figure 2.10: Girder modelled with plate elements

- **Group A** - Beam or plate elements which are part of a stiffener element (the red elements in figure 2.11)
- **Group B** - Beam or plate elements which are part of a girder element (the blue elements in figure 2.11)

To determine the boundaries of the girder- and stiffener elements a third group is defined as:

- **Group C** - Nodes which belong to both group A and group B (the grey nodes in figure 2.11)

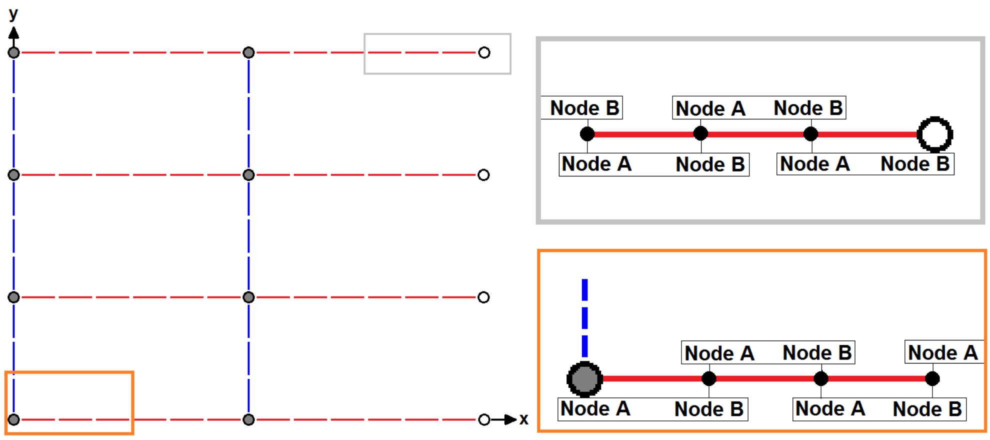
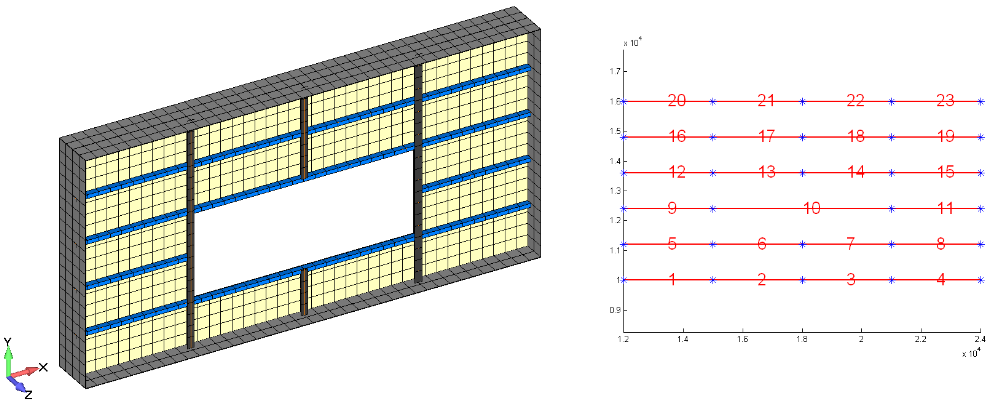


Figure 2.11: Beam elements and end-nodes of longitudinal stiffeners

For further purpose it is important to know the nodes which are at the end of the stiffeners and girders. To explain the procedure to detect the end nodes of stiffeners figure 2.11 is used as an example. Consider **group A** (the red elements in figure 2.11). *Node B* of one beam element is the same as *node A* of the adjacent beam element so every node is counted twice, except the nodes at the end. Furthermore if the end nodes are not attached to a member of **group B** (the blue elements in figure 2.11) the end nodes belong to a panel which has a free edge. Further discussion about free edges can be found in section 5.3. The end nodes for **group B** can be determined in a similar way.

2.2.2 Define Stiffener Elements

Before the stiffener elements are defined it is important for programming reason to sort the information of the coordinates of the nodes in **group C** in a systematic order. The Stress Check Model sorts the nodes primary according to ascending y-coordinates and secondary according to ascending x-coordinates. The starting point of determining the stiffener is the first node of the sorted information. The node is defined as *Node A* and the next node is defined as *Node B*. If *Node A* and *Node B* have the same y-coordinate a stiffener element is defined between the nodes. If *Node A* and *Node B* do not have the same y-coordinate no element is defined. *Node B* is redefined as *Node A* and the next node is defined as *Node B* and the procedure is repeated.



a) b)

Figure 2.12: a) Finite element model, b) Longitudinal stiffeners

Consider figure 2.12 as an example. The starting point is the node to the left of stiffener element number 1 (*Node A*). Since the next node (*Node B*) has the same y-coordinate stiffener element 1 is defined and the procedure is repeated until element number 4 has been defined. Then *Node A* becomes the node to the right of element number 4 and *Node B* becomes the node to the left of element 5. These nodes do not have the same y-coordinate and no element is defined.

Notice that in figure 2.12a there is an opening but in figure 2.12b element 10 overlaps the opening. The reason why element 10 is defined is because the Stress Check Model does not recognise the opening at this stage. When the node to the right of element 9 is defined as *Node A* the Stress Check Model will define the node to the left of element 11 as *Node B*. These nodes have the same y-coordinates and therefore element 10 is defined. In the next section a method to deal with openings and discontinuous stiffeners is explained.

2.2.3 Connect Element Properties to Stiffener Elements

The first step of recognising the geometrical and material properties of the stiffener elements is to find the centre coordinate of the elements in **group A**.

$$\text{Centre} = \frac{\text{Node B}_\text{x} + \text{Node A}_\text{x}}{2} \tag{2.1}$$

The centre coordinate of the elements in **group A** (see the yellow dots in figure 2.13a) are compared to the minimum and maximum x-coordinates of the stiffeners element (see the blue nodes in figure 2.13b). If the centre of the elements in **group A** falls in between the minimum and maximum coordinates of the stiffener element the stiffeners element is given the material and cross section properties of the corresponding elements in **group A**.

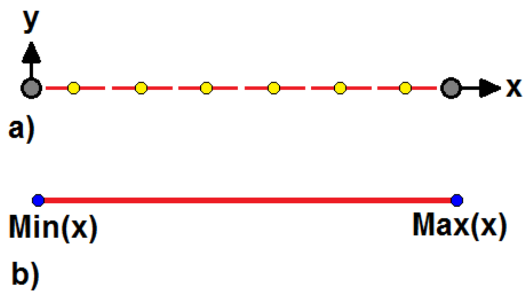


Figure 2.13: a) Centre coordinates of beam elements (yellow dots), b) Minimum and maximum x-coordinates of stiffener element (blue nodes)

Consider stiffener element number 10 from figure 2.12b. No elements from **group A** correspond to stiffener element number 10 so it gets eliminated. For programming reason all stiffener elements are renumbered to maintain continues numbering sequence of the stiffener elements (see figure 2.14).

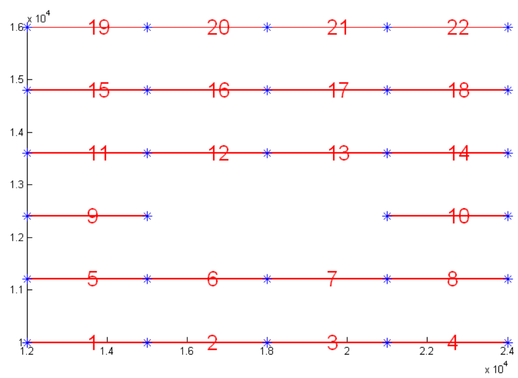


Figure 2.14: Stiffener elements for the Stress Check Model

The length of the stiffener elements can be determined from the minimum and maximum x-coordinates of every element.

$$\text{Length} = \text{Node B}_\text{x} - \text{Node A}_\text{x} \tag{2.2}$$

As mentioned in section 2.1.2 cross section properties of beams modelled with plate elements cannot be determined. It means that stiffeners elements 1-4 and 19-22 in figure 2.14 do not recognise their cross section properties.

2.2.4 Girder Elements

To determine the girder elements the coordinates of the nodes in **group C** are sorted in a systematic order. The nodes are sorted primary according to ascending x-coordinates and secondary according to ascending y-coordinates. The procedure becomes exactly the same as for the stiffeners elements except y-coordinates are replaced with x-coordinates.

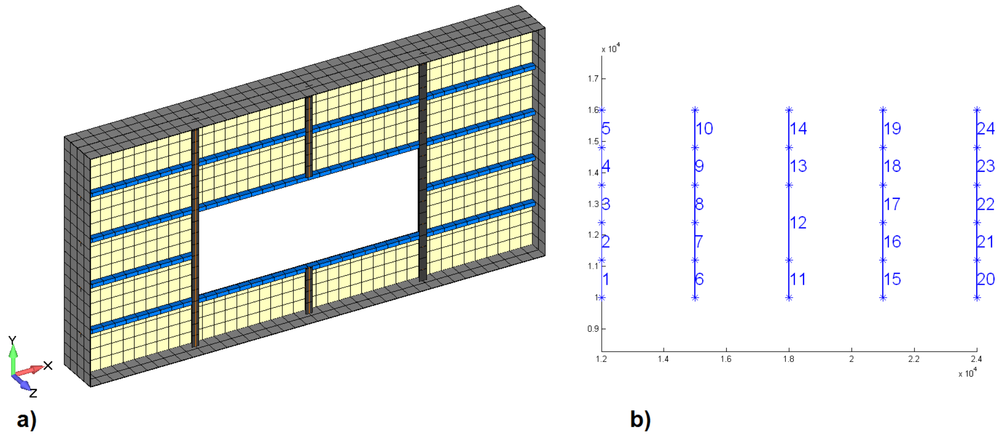


Figure 2.15: a) Finite element model, b) Transverse girders

Figure 2.15b shows that girder element number 12 overlaps the opening. This element is eliminated when the geometrical and material properties of the girder elements are determined. First the centre of the elements in **group B** are found.

$$\text{Centre} = \frac{\text{Node B}_\text{y} + \text{Node A}_\text{y}}{2} \tag{2.3}$$

The elements in **group B** which fall in between the minimum and maximum y-coordinates of the girder elements determine the properties of the girder elements. Girder element number 12 in figure 2.15b is eliminated because no elements in **group B** match the boundaries of element number 12. Since girder elements 1-5 and 19-23 in figure 2.16 are modelled with plate elements the cross section properties cannot be determined. The length of the girder elements is finally determined.

$$\text{Length} = \text{Node B}_\text{y} - \text{Node A}_\text{y} \tag{2.4}$$

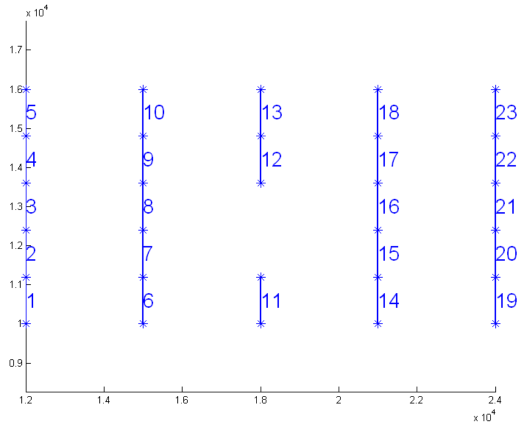


Figure 2.16: Girder elements for the Stress Check Model

2.2.5 Define Panel Elements Between Stiffeners

A key feature of the Stress Check Model is the algorithm to detect the panel sizes. The algorithm is explained in details in this section and is demonstrated with examples. The greatest challenge is to determine panel sizes when there are discontinuous stiffeners and girders. Discontinuous stiffeners and girders are caused by openings and difference in panel sizes. Figure 2.17 shows four possible configurations of discontinuity.

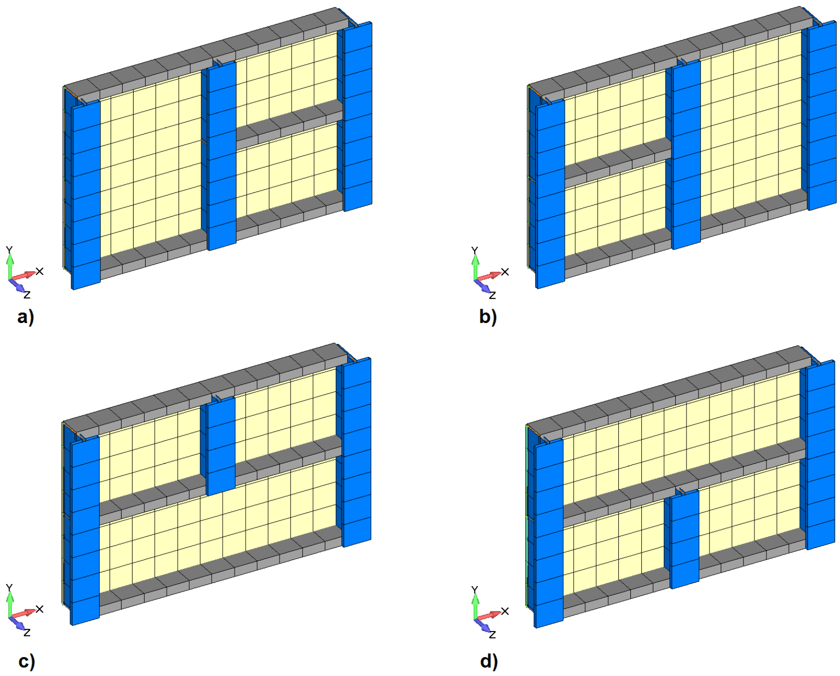


Figure 2.17: a) Discontinuity at left half, b) Discontinuity at right half, c) Discontinuity at bottom half, d) Discontinuity at top half

Larger models can be considered as being a combination of the different configurations shown in figure 2.17. The algorithm numbers the corner nodes of the panels according to figure 2.18.

Figure 2.18: Corner numbering of panels

The procedure is run for every stiffener element. The algorithm is explained here with examples referring to the every step. First corner node 1 is determined.

1. C_1 = Node A of the first stiffener element
2. If there are girder elements with y-coordinate of node B larger than the y-coordinate of C_1

   $C_{test,2}$ = Node B of the stiffener element

   Else go to step 10

   ⋮

10 C_1 = Node A of the next stiffener element

   Go to step 2

Figure 2.19 shows the coordinate system and element numbering for the case of figure 2.17a. When the algorithm runs for stiffener elements 1, 2 and 3 there are girder elements with y-coordinate greater than the stiffener elements.

1. C_1 = Node A of stiffener element 1 = (0,0)
2. All girder elements have y-coordinate of node B greater than the y-coordinate of C_1

   $C_{test,2}$ = Node B of stiffener element 1 = (300,0)

However when the algorithm runs for stiffener elements 4 and 5 there are no girder elements with y-coordinate greater than the stiffener elements.

10 C_1 = Node A of stiffener element 5 = (300,400)

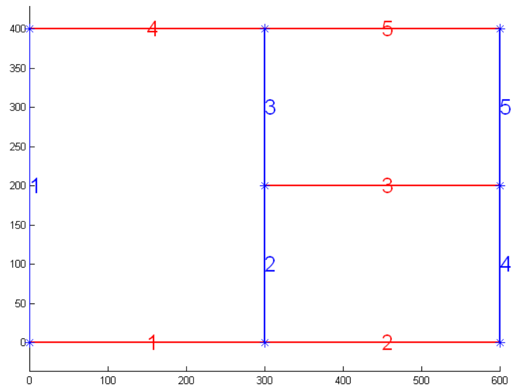


Figure 2.19: Longitudinal and transverse element numbering for case a

Go to step 2

2 There are no girder elements with y-coordinate of node B greater than the y-coordinate of C_1

Go to step 10

There are no more stiffener elements

end

The next steps of the algorithm determine the actual coordinate of corner node 2.

3 Index_x = Find girder elements with x-coordinate of node A the same as x-coordinate of $C_{test,2}$

Index_y = Find girder elements with y-coordinate of node A the same as y-coordinate of $C_{test,2}$

Index_2 = Intersect Index_x and Index_y

4 If the number of elements in Index_2 is not equal to 1

$C_{test,2}$ = Node B of the next stiffener element

Go back to step number 3

Else $C_2 = C_{test,2}$

Continuing with the case of figure 2.19 the procedure for determining corner node 2 of the first panel runs without iteration.

3 Index_x = 2,3

Index_y = 1,2,4

Index_2 = 2

4 Number of elements in Index_2 is equal to 1

$C_2 = C_{test,2}$ = (300,0)

Now, consider the configuration of figure 2.17d. Figure 2.20 shows the coordinate system and element numbering for that case. To determine corner node 2 of the large panel a iterative procedure is needed. First corner node 1 is determined as node A of stiffener element 3.

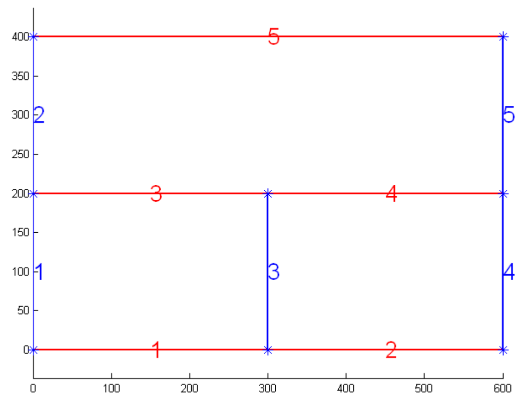


Figure 2.20: Longitudinal and transverse element numbering for case d

10 C_1 = Node A of stiffener element 3 = (0,200)

Go to step 2

2 There are girder elements with y-coordinate of node B greater than the y-coordinate of C_1

$C_{test,2}$ = Node B of stiffener element 3 = (300,200)

3 Index_x = 3

Index_y = 2,5

Index_2 = No value

4 Number of elements in Index_2 is not equal to 1

$C_{test,2}$ = Node B of girder element 4 = (600,200)

Go back to step number 3

3 Index_x = 4,5

Index_y = 2,5

Index_2 = 5

4 Number of elements in Index_2 is equal to 1

$C_2 = C_{test,2}$ = (600,200)

The algorithm continues to determine the corner node 3.

5 $C_{test,3}$ = Node B of the girder element corresponding to Index_2

6 Index_x = Find stiffener elements with x-coordinate of node B the same as x-coordinate of $C_{test,3}$

Index_y = Find stiffener elements with y-coordinate of node B the same as y-coordinate of $C_{test,3}$

Index_3 = Intersect Index_x and Index_y

7 If the number of elements in Index_3 is not equal to 1

$C_{test,3}$ = Node B of the next girder element

Go back to step number 6

Else $C_3 = C_{test,3}$

Continuing with the case of figure 2.19 the procedure for determining corner node 3 of the first panel runs with iteration.

5 $C_{test,3}$ = Node B of girder element 2 = (300,200)

6 Index_x = 1,4

Index_y = No value

Index_3 = No value

7 Number of elements in Index_3 is not equal to 1

$C_{test,3}$ = Node B of girder element 3 = (300,400)

6 Index_x = 1,4

Index_y = 4,5

Index_3 = 4

7 Number of elements in Index_3 is equal to 1

$C_3 = C_{test,3} = (300{,}400)$

The procedure to determine corner node 3 of the large panel in figure 2.20 runs without iteration.

5 $C_{test,3}$ = Node B of girder element 5 = (600,400)

6 $Index_x = 2{,}4{,}5$

$Index_y = 5$

$Index_3 = 5$

7 Number of elements in $Index_3$ is equal to 1

$C_3 = C_{test,3} = (600{,}400)$

The procedure to determine corner node 4 and the coordinates of the panels is non-iterative.

8 x-coordinate of C_4 = x-coordinate of C_1

y-coordinate of C_4 = y-coordinate of C_3

9 Panel is defined with corners located at C_1, C_2, C_3 and C_4

Continuing with the case of figure 2.20 the procedure for determining corner node 4 and the coordinates of the large panel is the following.

8 x-coordinate of C_4 = x-coordinate of C_1 = 0

y-coordinate of C_4 = y-coordinate of C_3 = 400

9 Panel 3 has the following coordinate, C_1 = (0,200), C_2 = (600,200), C_3 = (600,400) and C_4 = (0,400)

Remark

The bottom edge of the large panel in figure 2.20 consist of two stiffener elements. What will happen is that the algorithm will define additional panel with corner node 1 as node A of stiffener element 4. Figure 2.21 shows how two panels, 3 and 4, have been defined and overlap each other.

Problem of this kind are dealt with in the following way. First the location of the centre of each panel is found. Next the boundaries of each panel are determined. The centre of each panel element is compared to the boundaries of all the other panel elements. If the centre of one panel element is located within the boundaries of another panel element the same panel is eliminated. In this case the centre of panel element 4 is within the boundaries of panel element 3 so panel element 4 is eliminated.

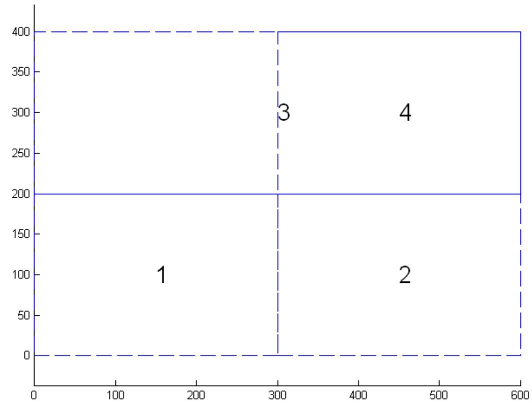


Figure 2.21: Panel numbering for case d

2.2.6 Connect Element Properties to Panel Elements

The procedure to recognise the geometrical and material properties of the panel elements is quite similar to the procedure for the stiffener and girder elements. First the centre coordinate of all the plate elements from the finite element model are found.

$$\text{Centre(x,y)} = \left(\frac{x_1 + x_2 + x_3 + x_4}{4}, \frac{y_1 + y_2 + y_3 + y_4}{4} \right) \tag{2.5}$$

The centre coordinates of the plate elements are compared to the minimum and maximum x and y coordinates of the panel elements. If the centre of the plate elements falls in between the boundaries of the panel element the panel element is given the material properties and thickness of the corresponding plate elements. Figure 2.22 shows the boundaries of a panel element (blue lines) and the centre of the plate elements (yellow dots) which fall in between the boundaries.

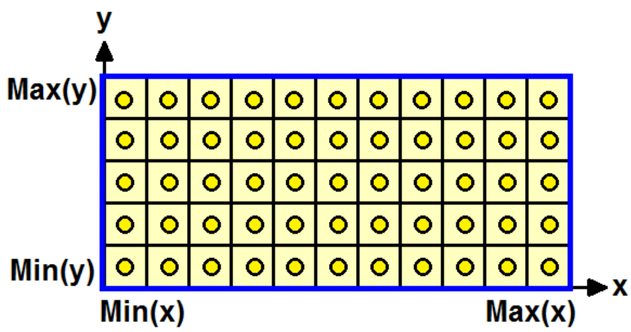


Figure 2.22: Plate elements within boundaries of panel element

Openings are dealt with in the same way as for the stiffener and girder elements. Figure 2.23b shows that the Stress Check Model first defines the opening as panel element 6 and 7. Later panel element 7 is eliminated as explained in case D in section 2.2.5.

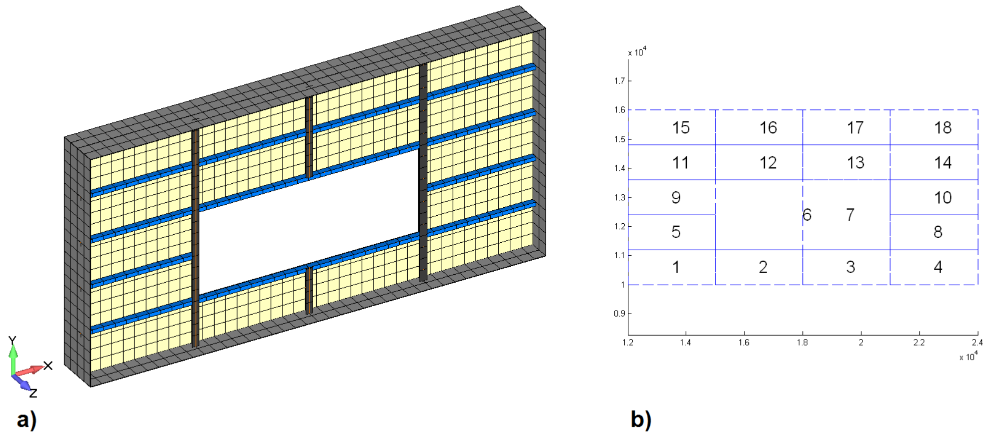


Figure 2.23: a) Finite element model, b) Panel elements

When the plate elements from the finite element model are matched with the panel elements from the Stress Check Model no plate elements will match panel element number 6 so it gets eliminated. Finally all panel elements are renumbered to maintain continuous numbering sequence. The result is showed in figure 2.24.

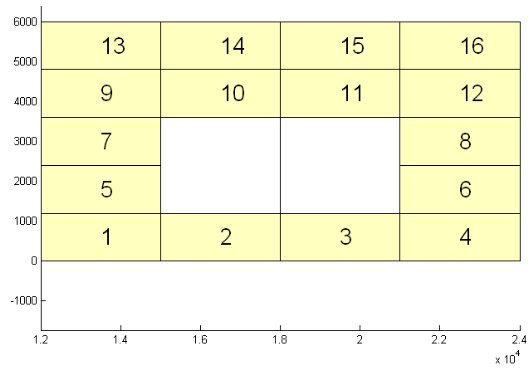


Figure 2.24: Panel elements for the Stress Check Model

The length and width of the panel elements can be determined from the coordinates of the boundaries.

$$\text{Length} = \text{Max(x)} - \text{Min(x)} \tag{2.6}$$

$$\text{Width} = \text{Max(y)} - \text{Min(y)} \tag{2.7}$$

2.3 Stress Parameters

This section describes in details the strategy to determine the design stresses acting on the panels. The output matrix from this section is the following:

- Stresses(Panel id, $\sigma_{\text{x,max}}$, $\sigma_{\text{x,min}}$, $\sigma_{\text{y,max}}$, $\sigma_{\text{y,min}}$, τ, q)

2.3.1 Stresses Acting on the Edge of Panels

The design load acting on panels is based on the stresses along the edges of the panels. Figure 2.25a shows a panel element modelled with 4x8 plate elements. The locations of known stresses are marked with x.

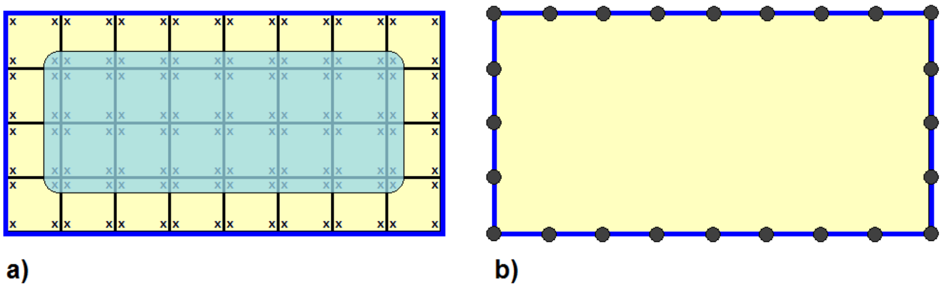


Figure 2.25: a) Location of stress values from finite element analysis, b) Location of stress values for the Stress Check Model

To determine the stresses along the edges of the panel the Stress Check Model ignores the results of the shaded area of figure 2.25a. The intermediate stress values along the edges are taken as the mean value of two adjacent plate elements. Figure 2.25b shows the corresponding stresses for the Stress Check Model and the location of the stresses.

Shear stress results of the plate elements in figure 2.3a are the same at every corner location, meaning that only one shear stress value is obtained for every plate element. Figure 2.26a shows the shear stress values marked with x.

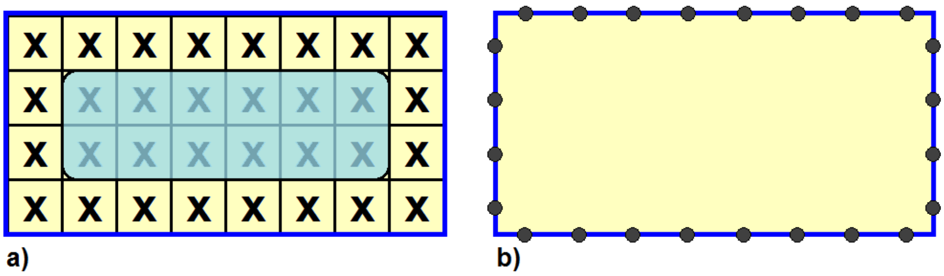


Figure 2.26: a) Shear stress values from finite element analysis, b) Location of shear stress values for the Stress Check Model

The Stress Check Model ignores the results of the shaded area of figure 2.26a. The shear stresses and their location is showed in figure 2.26b.

2.3.2 Linearization of Stress Results

The in-plane design stresses σ_x and σ_y have to vary linearly at the edges of the panels for the Stress Check Model. However that is seldom the case from a finite element analysis. Figure 2.27 shows a panel which is modelled with 4x8 plate elements. The location of the stress values are showed with black dots.

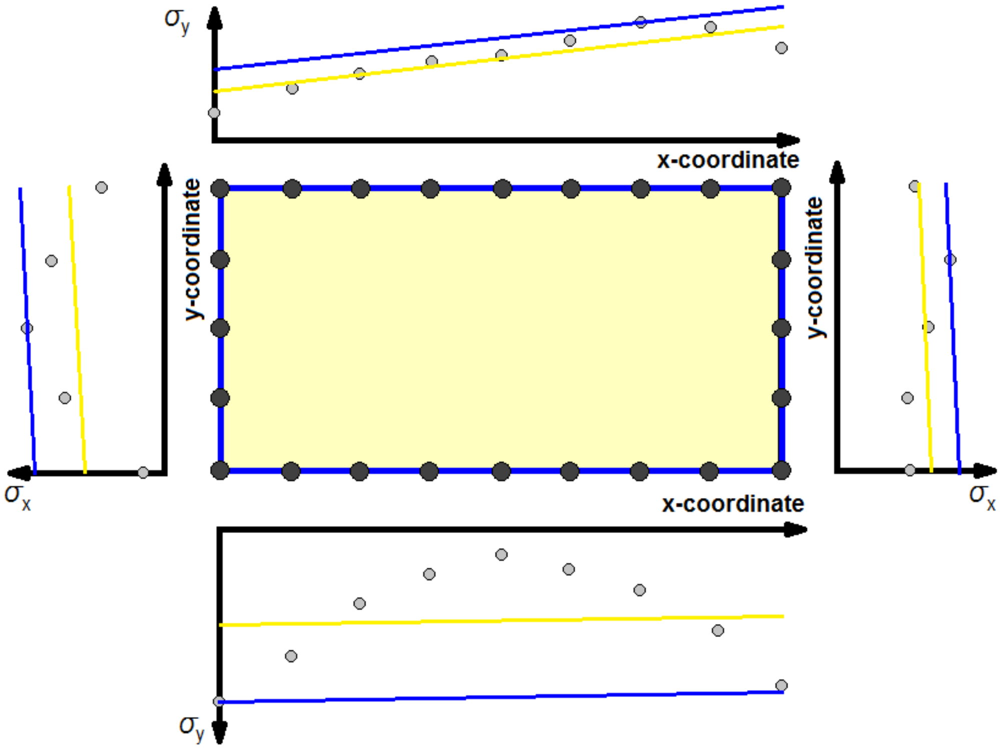


Figure 2.27: Linearization of finite element stress results

A typical finite element stress result for σ_x are plotted at the left and right edge of the panel with grey dots. Likewise the results for σ_y are plotted with grey dots at the top and bottom edge. Since both the stress values and the stress locations are known the stress distribution can be linearised using linear regression. The regression is done according to the method of least squares [12].

$$Y = \beta_0 + \beta_1 X \tag{2.8}$$

Where Y represent the stress value of the regression line and X represent the location of the stress value (x or y coordinate). The parameters β_0 and β_1 are determined according to:

$$\beta_1 = \frac{SS_{xy}}{SS_{xx}} \tag{2.9}$$

$$\beta_0 = \frac{\sum Y_i}{n} - \beta_1 \frac{\sum X_i}{n} \tag{2.10}$$

Where n is the number of sample points, Y_i are the stress results at the known locations X_i and SS_{xy} and SS_{xx} are determined according to:

$$SS_{xy} = \sum X_i Y_i - \frac{\sum X_i \sum Y_i}{n} \tag{2.11}$$

$$SS_{xx} = \sum X_i^2 - \frac{(\sum X_i)^2}{n} \tag{2.12}$$

To represent the stress distribution only with linear regression is unconservative approach since the peak stresses will be reduced. Therefore the regression line is shifted according to the difference between the peak stress value and corresponding linearised stress value. The blue lines in figure 2.27 show the outcome.

2.3.3 Design Load

This section describes how the design load is determined for the Stress Check Model. The design load can be categorised into the in-plane stresses σ_{x} and σ_{x}, edge shear load τ and lateral pressure q.

In-plane Stresses

The in plane stresses σ_{x} and σ_{y} have to be symmetric. This means that the stress distribution at opposite edges has to be the same. The user of the Stress Check Model is given two options to determine the in plane design stresses. The first option follows clause 4.6(3) in Eurocode 3, part 1.5 [9].

> The plate buckling verification of the panel should be carried out for the stress resultants at a distance 0,4a or 0,5b, whichever is the smallest, from the panel end where the stresses are the greater. In this case the gross sectional resistance needs to be checked at the end of the panel.

Figure 2.28 shows graphically how design in-plane stresses at left and right edge of a panel are determined according to the method of Eurocode 3. The stresses located at left and right top corners are plotted along the top edge. The same is done with the stresses located at the left and right bottom corners. The design value is then determined at a distance 0,4a or 0,5b from the greater corner stress. Same procedure is done for the stress distribution for the top and bottom edges.

Since design codes might contradict each other the user of the Stress Check Model is also given the option to use the greatest stress value at edges as the design value. One might expect this approach to be far more conservative compared to the method of Eurocode 3. Further discussion about this topic is given in section 5.3.

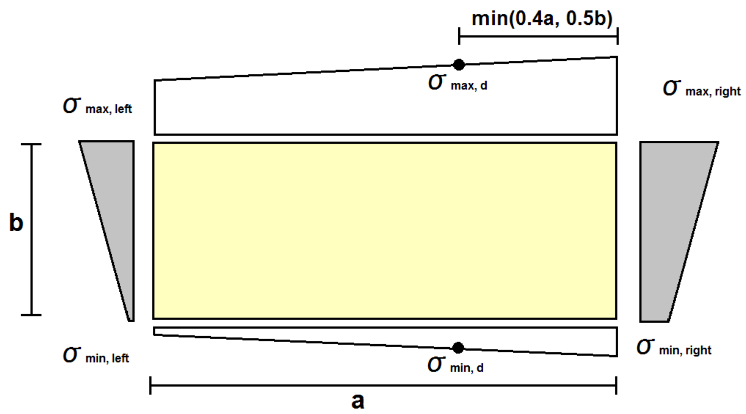


Figure 2.28: Design stresses acting on panels

Shear Stresses

The design value for shear stresses has to be constant for each panel and is based on the shear stresses acting on the edge of panels. The user of the Stress Check Model is given three options to determine the design load; to use the maximum shear stress value, use weighted average value or use weighted average value based on absolute shear stress values. Figure 2.29 shows graphically the difference between the three methods, for simplicity only one edge of a panel is considered but the procedure is performed on all edges of the panel.

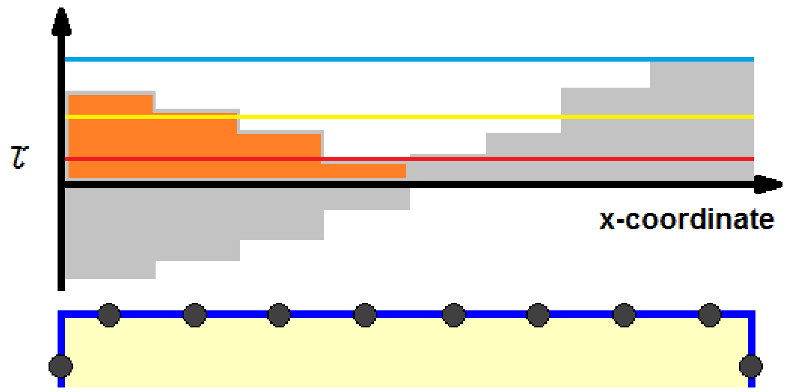


Figure 2.29: Shear stress at panel edges

The grey area represents a shear stress distribution along the panel edge from a finite element analysis. The orange area represents the absolute value of the negative shear stresses. The blue line is a result of maximum shear stresses, the yellow line represents the weighted average value based on absolute stress values and the red line is the weighted average value where distinction is made between positive and negative shear stress values.

If the shear stresses acting on the edges of the panels are all defined either negative or positive the two weighted average methods will give the same results. However if both negative and positive shear stresses are acting on the edges the weighted average value based absolute stress values will give higher design value.

Lateral Pressure

The design value for lateral load has to be constant for each panel. In case of lateral pressure is acting on the structure the user is given two options to determine the design load; either to use the maximum pressure value applied to each panel as the design load or the average value. Figure 2.30 demonstrates the options in case of triangular lateral pressure.

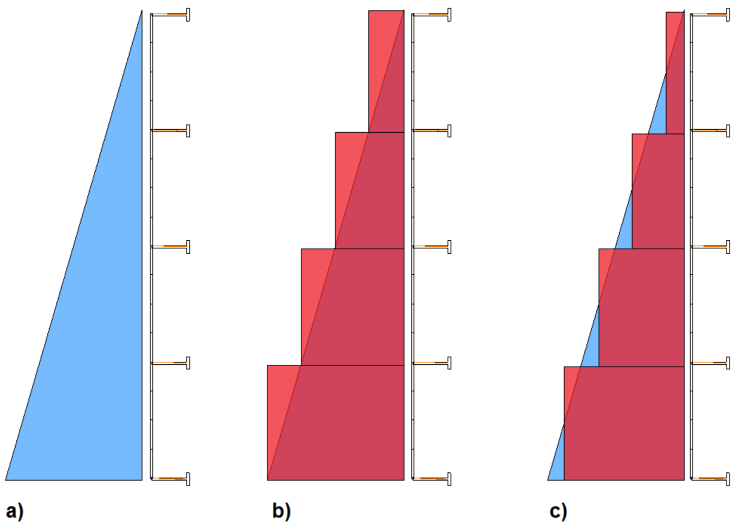


Figure 2.30: a) Lateral pressure acting on panel, b) Maximum value, c) Average value

Chapter 3

ABS Package

The *ABS: Guide for buckling and ultimate strength assessment for offshore structures* is chosen to demonstrate the functionality of the Stress Check Model. There are two main reasons for choosing the ABS guide. First of all to compare results with the SDC verifier which is also based on the ABS guide. Secondly the ABS guide has a relatively simple structure compared to the *DNV: Buckling strength of plated structures* design code, which makes the programming more straightforward.

Five buckling checks have been incorporated into the Stress Check Model. **Buckling state limit, ultimate strength** and **uniform lateral pressure** are related to unstiffened panels and are dealt with in section 3.1. **Beam-column buckling state limit** and **Flexural-torsional buckling state limit** are related to stiffened panels and are dealt with in section 3.2. According to the ABS guide panels are allowed to fail the buckling state limit as long as the ultimate strength criterion is satisfied. However beam-column buckling state of stiffened panels is affected when panels fail the buckling state limit.

Table 2.1 lists the parameters which are needed for the buckling stress checks. For the following sections it is convenient to use symbols for these parameters.

l	-	Length of panel
s	-	Width of panel
t	-	Thickness of panel
d_w	-	Height of the stiffener web
t_w	-	Thickness of the stiffener web
b_f	-	Width of the stiffener flange
t_f	-	Thickness of the stiffener flange
E	-	Young's modulus
ν	-	Poisson's ratio
σ_0	-	Specified minimum yield point of the material
$\sigma_{x,max}$	-	Maximum compressive stress in x-direction
$\sigma_{x,min}$	-	Minimum stress in x-direction
$\sigma_{y,max}$	-	Maximum compressive stress in y-direction
$\sigma_{y,min}$	-	Minimum stress in y-direction

τ	-	Edge shear stress
q	-	Lateral pressure

3.1 Plate Panels

This section describes the fundamentals of the buckling checks on unstiffened panel according to the ABS guide. Figure 3.1 [6] shows how stiffened panels are modelled according to the ABS guide.

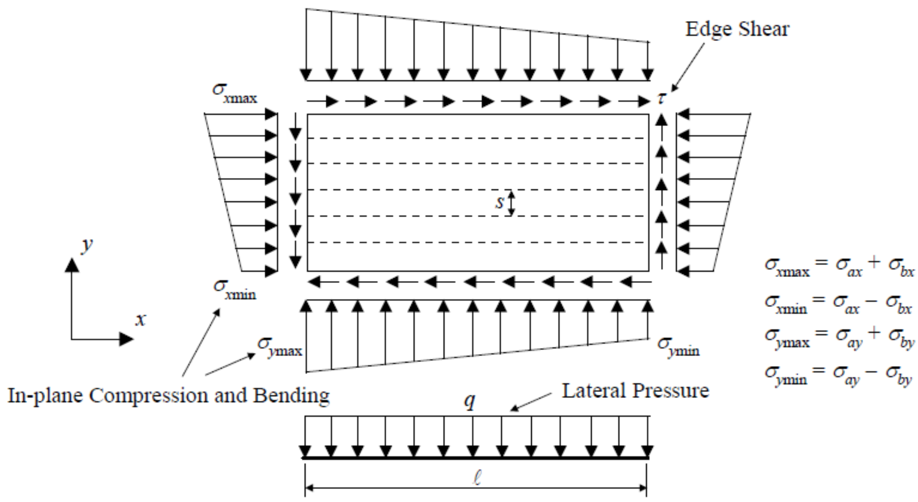


Figure 3.1: Primary loads and loads effects on plate and stiffened panel

3.1.1 Buckling State Limit

The buckling state limit of panels subjected to in-plane loading is expressed with the following interaction formula:

$$\left(\frac{\sigma_{x,max}}{\eta\sigma_{Cx}}\right)^2 + \left(\frac{\sigma_{y,max}}{\eta\sigma_{Cy}}\right)^2 + \left(\frac{\tau}{\eta\tau_C}\right)^2 \leq 1 \tag{3.1}$$

The denominators on the left hand side are the loads acting the panels (see figure 3.1). The Stress Check Model determines the load automatically for each panel. If an edge is loaded in tension, the corresponding maximum compressive stress is set to zero. The factor η is the maximum allowable strength utilization factor and should be determined by the user depending on the application of the structure (by default η is set equal to 1.0).

The critical buckling stresses σ_{Cx}, σ_{Cy} and τ_{C} are defined as:

$$\sigma_{Ci} = \begin{cases} \sigma_{Ei} & \text{for } \sigma_{Ei} \leq P_r\sigma_0 \\ \sigma_0\left[1 - P_r\left(1 - P_r\right)\frac{\sigma_0}{\sigma_{Ei}}\right] & \text{for } \sigma_{Ei} > P_r\sigma_0 \end{cases} \tag{3.2}$$

$$\tau_C = \begin{cases} \tau_E & \text{for } \tau_E \leq P_r \tau_0 \\ \tau_0 \left[1 - P_r \left(1 - P_r\right) \dfrac{\tau_0}{\tau_E} \right] & \text{for } \tau_E > P_r \tau_0 \end{cases} \tag{3.3}$$

Where i stands for x or y, P_r is the proportional linear elastic limit of the structure (recommended value for steel is 0.6) and the shear strength τ_0 is equal to:

$$\tau_0 = \frac{\sigma_0}{\sqrt{3}} \tag{3.4}$$

The elastic buckling stresses σ_{Ex}, σ_{Ey} and τ_{E} are defined as:

$$[\sigma_{Ei}, \tau_E] = k_s \frac{\pi^2 E}{12\left(1 - \nu^2\right)} \left(\frac{t}{s}\right)^2 \tag{3.5}$$

Where k_s is a boundary dependent constant, depending on the aspect ratio α and the ratio of edge stresses κ.

$$\alpha = \frac{l}{s} \tag{3.6}$$

$$\kappa = \frac{\sigma_{i,min}}{\sigma_{i,max}} \tag{3.7}$$

The boundary constant k_s for σ_{Ex} is equal to:

$$k_s = C_1 \begin{cases} \dfrac{8.4}{\kappa + 1.1} & \text{for } 0 \leq \kappa \leq 1 \\ 7.6 - 6.4\kappa + 10\kappa^2 & \text{for } -1 \leq \kappa < 1 \end{cases} \tag{3.8}$$

The boundary constant k_s for σ_{Ey} is equal to:

$$k_s = C_2 \begin{cases} \left[1.0875\left(1 + \dfrac{1}{\alpha^2}\right)^2 - 18\dfrac{1}{\alpha^2}\right](1 + \kappa) + 24\dfrac{1}{\alpha^2} \\ \qquad \text{for } \kappa < \dfrac{1}{3} \text{and } 1 \leq \alpha \leq 2 \\ \left[1.0875\left(1 + \dfrac{1}{\alpha^2}\right)^2 - 9\dfrac{1}{\alpha}\right](1 + \kappa) + 12\dfrac{1}{\alpha} \\ \qquad \text{for } \kappa < \dfrac{1}{3} \text{and } \alpha > 2 \\ \left(1 + \dfrac{1}{\alpha^2}\right)^2 (1.675 - 0.675\kappa) \\ \qquad \text{for } \kappa \geq \dfrac{1}{3} \end{cases} \tag{3.9}$$

The boundary constant k_s for τ_{E} is equal to:

$$k_s = C_1 \left[4\left(\frac{1}{\alpha^2}\right) + 5.34\right] \tag{3.10}$$

The factors C_1 and C_2 depend on the type of stiffener which define the boundaries of the panel. For T and angle shaped stiffeners C_1 and C_2 are equal to 1.1 and 1.2 respectively.

3.1.2 Ultimate Strength under Combined In-Plane Stresses

The ultimate strength of panels between stiffeners subjected to in-plane loading is expressed with the following interaction formula:

$$\left(\frac{\sigma_{x,max}}{\eta\sigma_{Ux}}\right)^2 + \left(\frac{\sigma_{y,max}}{\eta\sigma_{Uy}}\right)^2 + \left(\frac{\tau}{\eta\tau_U}\right)^2 - \varphi\left(\frac{\sigma_{x,max}}{\eta\sigma_{Ux}}\right)\left(\frac{\sigma_{y,max}}{\eta\sigma_{Uy}}\right) \leq 1 \quad (3.11)$$

The factor φ is a coefficient to reflect interaction between longitudinal and transverse stresses and is defined as:

$$\varphi = 1 - \frac{\beta}{2} \quad (3.12)$$

Where β is the slenderness ratio of the panel which is defined as:

$$\beta = \frac{s}{t}\sqrt{\frac{\sigma_{yield}}{E}} \quad (3.13)$$

The ultimate buckling stresses σ_{Ux}, σ_{Uy} and τ_{U} are defined as:

$$\sigma_{Ux} = C_x\sigma_0 \geq \sigma_{Cx} \quad (3.14)$$

$$\sigma_{Uy} = C_y\sigma_0 \geq \sigma_{Cy} \quad (3.15)$$

$$\tau_U = \tau_c + 0.5\frac{\sigma_0 - \sqrt{3}\tau_c}{\sqrt{1+\alpha+\alpha^2}} \geq \tau_c \quad (3.16)$$

Where σ_{Cx}, σ_{Cy}, τ_{C} and α are as defined in section 3.1.1. The ultimate buckling stresses should not be lower than the critical buckling stresses. The factors C_x and C_y are defined as:

$$C_x = \begin{cases} \frac{2}{\beta} - \frac{1}{\beta^2} & \text{for } \beta > 1 \\ 1 & \text{for } \beta \leq 1 \end{cases} \quad (3.17)$$

$$C_y = C_x\frac{1}{\alpha} + 0.1\left(1 - \frac{1}{\alpha}\right)\left(1 + \frac{1}{\beta^2}\right)^2 \quad (3.18)$$

3.1.3 Uniform Lateral Pressure

When panels are subjected to lateral pressure alone or combined with in-plane stresses the following interaction formula has to be satisfied.

$$\frac{q}{\eta 4.0\sigma_0\left(\frac{t}{s}\right)^2\left(1+\frac{1}{\alpha^2}\right)\sqrt{1-\left(\frac{\sigma_e}{\sigma_0}\right)^2}} \leq 1 \quad (3.19)$$

All parameters are as defined before except the equivalent stress according to von Mises:

$$\sigma_e = \sqrt{(\sigma_{x,max})^2 + (\sigma_{y,max})^2 - \sigma_{x,max}\sigma_{y,max} + 3\tau^2} \quad (3.20)$$

3.2 Stiffened Panels

This section describes the fundamentals of buckling checks on stiffened panels according to the ABS guide. The load which is applied to the stiffeners depends on the load acting on the associated panels. Furthermore the beam-column buckling check takes into account the geometry of both the panels and the attached stiffeners. Therefore an algorithm which detects the properties of the associated panels to stiffeners is required.

3.2.1 Stiffeners and Associated Panels

The algorithm runs for every stiffener element which is modelled with beam elements in the finite element model. Stiffeners which are only attached to one panel will be discarded.

1. x_A = x-coordinate of node A of the stiffener element

   x_B = x-coordinate of node B of the stiffener element

   y = y-coordinate of the stiffener element

2. Group 1 = Find all panel elements which have the minimum x-coordinate same as x_A

   Group 2 = Find all panel elements which have the maximum x-coordinate same as x_B

   Group 3 = Find all panel elements which have the maximum y-coordinate same as y

   Group 4 = Find all panel elements which have the minimum y-coordinate same as y

3. Group A = Union Group 3 and Group 4

   Group B = Union Group 1 and Group 2

   Group C = Find panel elements in Group B which are not in Group A

4. **1st check**

   If Group C contains any panel elements

   Remove panel elements of Group C from Group B

5. If the number of remaining panel elements in Group B is equal to two

   The stiffener element is attached to the remaining panel elements of Group B

The first check of the algorithm covers most general cases. Figure 3.2 shows in schematic way the different groups for the algorithm where the solid red stiffener is the target.

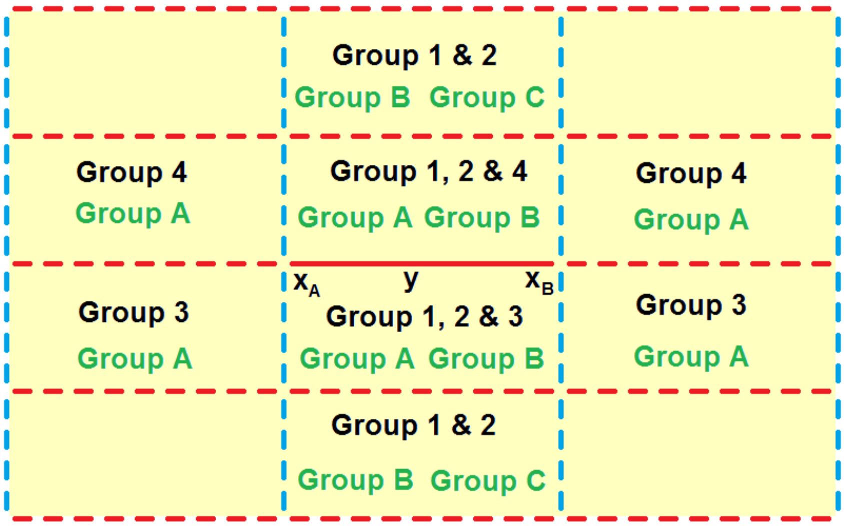


Figure 3.2: Finding associated panels to stiffeners for general cases

6 **2nd check**

If the number of remaining panel elements in Group B is equal to one

Group D = Intersect Group B and Group 3

Group E = Intersect Group B and Group 4

7 If there are any panel elements in Group D

Index = Find the element in Group 4 which have minimum x-coordinate smaller than x_A and maximum x-coordinate greater than x_B

The stiffener element is attached to the panel element of Group B and Index

8 If there are any panel elements in Group E

Index = Find the element in Group 3 which have minimum x-coordinate smaller than x_A and maximum x-coordinate greater than x_B

The stiffener element is attached to the panel element of Group B and Index

The second check covers the cases where the lengths of the associated panels differ. Figure 3.3 shows in schematic way the different groups for the algorithm where the solid red stiffener is the target.

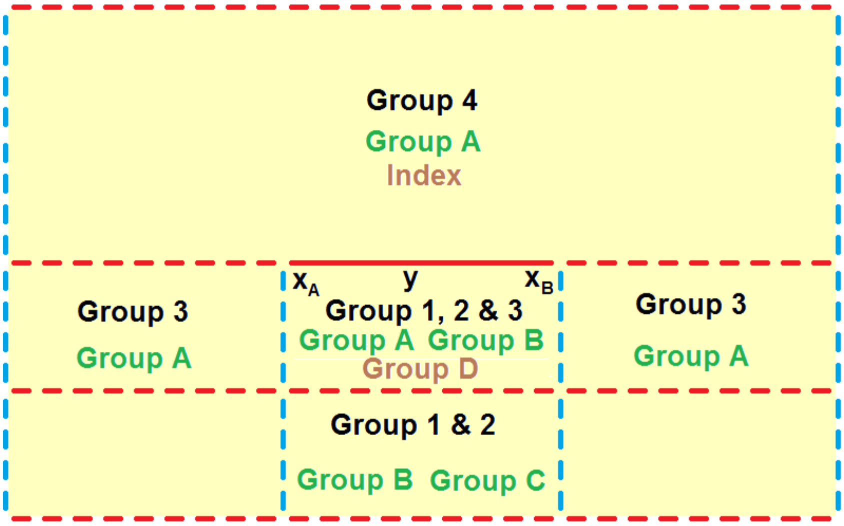


Figure 3.3: Finding associated panels to stiffeners for severe cases

The stresses applied to the stiffeners are derived from the stresses applied to the panels. Due to numerical deviation from both the finite element analysis and the determination of design stress from the Stress Check Model, the stress results at the location of a stiffener might differ between two panels. The design stress applied to stiffeners is taken as the greater compressive value of σ_1 and σ_2 showed in figure 3.4 where the solid red stiffener is the target.

$$\sigma_a = \max\left(\sigma_1, \sigma_2\right) \tag{3.21}$$

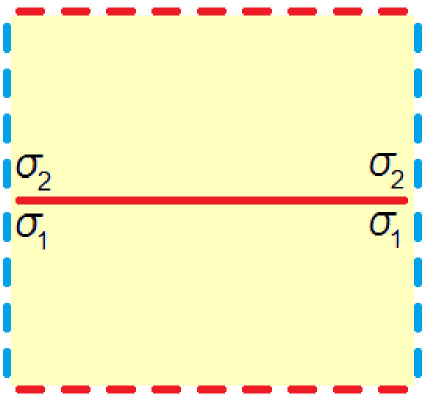


Figure 3.4: Stresses applied to stiffeners

The formulation of the ABS guide assumes that associated panels have the same geometrical and material properties. For the cases where the properties of the associated panels of stiffeners differ the Stress Check Model warns the user. The buckling checks are computed for both property types and the result is based on the worst outcome.

3.2.2 Beam-Column Buckling State Limit

The beam-column buckling state limit is to satisfy the following expression:

$$\frac{\sigma_a}{\eta\sigma_{CA}\left(A_e/A\right)} + \frac{C_m\sigma_b}{\eta\sigma_0\left[1-\sigma_a/\left(\eta\sigma_{E(C)}\right)\right]} \leq 1 \tag{3.22}$$

The first term at the left hand side of expression 3.22 is associated with axial forces and the second term is associated with bending action. The maximum allowable strength utilization factor η is as defined in section 3.1.1 and the parameter C_m is a moment adjustment coefficient (recommended value is 0.75). The axial stress, σ_a acting on the stiffener is determined according to equation 3.21 and figure 3.4. The total sectional area and the effective sectional area are defined as:

$$A = A_s + st = d_w t_w + b_f t_f + st \tag{3.23}$$

$$A_e = A_s + s_e t = d_w t_w + b_f t_f + s_e t \tag{3.24}$$

Where the effective width s_e is defined as:

$$s_e = \begin{cases} s & \text{if the buckling state limit is satisfied} \\ C_x C_y C_{xy} s & \text{if the buckling state limit is not satisfied} \end{cases} \tag{3.25}$$

The factor C_x is as defined in section 3.1.2. The factors C_y and C_{xy} are defined as:

$$C_y = 0.5\varphi\left(\frac{\sigma_{y,max}}{\sigma_{Uy}}\right) + \sqrt{1-\left(1-0.25\varphi^2\right)\left(\frac{\sigma_{y,max}}{\sigma_{Uy}}\right)^2} \tag{3.26}$$

$$C_x y = \sqrt{1-\left(\frac{\tau}{\tau_0}\right)^2} \tag{3.27}$$

Where φ, σ_Uy and τ_0 are as defined before. The critical buckling stress σ_CA is defined as:

$$\sigma_{CA} = \begin{cases} \sigma_{E(C)} & \text{for } \sigma_{E(C)} \leq P_r\sigma_0 \\ \sigma_0\left[1-P_r\left(1-P_r\right)\dfrac{\sigma_0}{\sigma_{E(C)}}\right] & \text{for } \sigma_{E(C)} > P_r\sigma_0 \end{cases} \tag{3.28}$$

Where the proportional linear elastic limit of the structure P_r is defined as in section 3.1.1 and the Euler's buckling stress $\sigma_\mathrm{E(C)}$ is defined as:

$$\sigma_{E(C)} = \frac{\pi^2 E r_e^2}{l^2} \tag{3.29}$$

Where r_e is the radius of gyration of the effective sectional area A_e:

$$r_e = \sqrt{\frac{I_e}{A_e}} \tag{3.30}$$

The moment of inertia of the effective sectional area is based on the sectional area of the stiffener and the effective width of the associating plate.

$$\begin{aligned} I_e = \frac{1}{12}\left(t^3 s_e + d_w^3 t_w t_f^3 b_f\right) + 0.25\left(t + d_w\right)^2 d_w t_w \\ + \left(0.5t + d_w + 0.5t_f\right)^2 b_f t_f - A_e z_{ep}^2 \end{aligned} \tag{3.31}$$

Where z_{ep} is the distance from the the centre of the plate to the centroid of the effective sectional area, determined as:

$$z_{ep} = \frac{0.5\left(t + d_w\right) d_w t_w + \left(0.5t + d_w + 0.5t_f\right) b_f t_f}{A_e} \tag{3.32}$$

The bending stress, σ_{b}, is determined according to:

$$\sigma_b = \frac{M}{SM_w} \tag{3.33}$$

Where M is the maximum bending moment induced by lateral pressure and SM_w is the effective section modulus of the stiffener.

$$M = \frac{qsl^2}{12} \tag{3.34}$$

$$SM_w = \frac{I_w}{0.5t + d_w + t_f - z_{wp}} \tag{3.35}$$

Where:

$$\begin{aligned} I_w = \frac{1}{12}\left(t^3 s_e + d_w^3 t_w t_f^3 b_f\right) + 0.25\left(t + d_w\right)^2 d_w t_w \\ + \left(0.5t + d_w + 0.5t_f\right)^2 b_f t_f - A_w z_{wp}^2 \end{aligned} \tag{3.36}$$

$$A_w = A_s + s_w t = d_w t_w + b_f t_f + s_w t \tag{3.37}$$

$$z_{wp} = \frac{0.5\left(t + d_w\right) d_w t_w + \left(0.5t + d_w + 0.5t_f\right) b_f t_f}{A_w} \tag{3.38}$$

The effective breadth s_w is related to the distance between zero bending moments along the stiffener. The Stress Check Model uses a conservative value for the effective breadth equal to 58% of the total width s. Further discussion about the effective breadth can be found in section 5.3.

The yield stress σ_0 is determined from weighted areas of the stiffener and its associated panels.

$$\sigma_0 = \frac{\left(A_e\text{-}A_{\mathit{stiffener}}\right) \sigma_{0,\mathrm{panel}} + A_{\mathit{stiffener}} \sigma_{0,\mathrm{stiffener}}}{A_e} \tag{3.39}$$

3.2.3 Flexural-Torsional Buckling State Limit

The flexural-torsional buckling state limit is to satisfy the following expression:

$$\frac{\sigma_a}{\eta \sigma_{CT}} \leq 1 \tag{3.40}$$

Where the axial stress σ_{a} is as defined in section 3.2.2 and the maximum allowable strength utilization factor η is as defined in section 3.1.1. The critical torsional-flexural buckling stress is defined as:

$$\sigma_{CT} = \begin{cases} \sigma_{ET} & \text{for } \sigma_{ET} \leq P_r \sigma_0 \\ \sigma_0 \left[1 - P_r \left(1 - P_r\right) \dfrac{\sigma_0}{\sigma_{ET}} \right] & \text{for } \sigma_{ET} > P_r \sigma_0 \end{cases} \tag{3.41}$$

Where σ_0 is determined as in section 3.2.2 and σ_{ET} is the elastic flexural-torsional buckling stress with respect to the axial compression of a stiffener including its associated panel, defined as:

$$\sigma_{ET} = \frac{\dfrac{K}{2.6} + \left(\dfrac{n\pi}{l}\right)^2 \Gamma + \dfrac{C_0}{E} \left(\dfrac{l}{n\pi}\right)^2}{I_0 + \dfrac{C_0}{\sigma_{cL}} \left(\dfrac{l}{n\pi}\right)^2} E \tag{3.42}$$

Where K is the St. Venant torsion constant for the stiffener cross section, Γ is a warping constant and I_0 is the polar moment of inertia of the stiffener.

$$K = \frac{b_f t_f^3 + d_w t_w^3}{3} \tag{3.43}$$

$$\Gamma = m I_{zf} d_w^2 + \frac{d_w^3 t_w^3}{36} \tag{3.44}$$

$$I_{zf} = \frac{t_f b_f^3}{12} \left(1 + 3 \frac{u^2 d_w t_w}{A_s}\right) \tag{3.45}$$

$$I_0 = I_y + m I_z + A_s \left(y_o^2 + z_0^2\right) \tag{3.46}$$

I_y and I_z are the moment of inertia of the stiffener about the y-and z-axis, y_0 is the horizontal distance between the centroid of the stiffener and the centerline of the plate and z_0 is the vertical distance between the centroid of the stiffener and its flange toe.

$$I_y = \frac{1}{12} \left(d_w^3 t_w + t_f^3 b_f\right) + 0.25 d_w^3 t_w + b_f t_f \left(d_w + 0.5 t_f\right)^2 - A_s z_0^2 \tag{3.47}$$

$$I_z = \frac{1}{12} \left(t_w^3 d_w + b_f^3 t_f\right) + b_f t_f \left(b_1 - 0.5 b_f\right)^2 - A_s y_0^2 \tag{3.48}$$

$$y_0 = \frac{(b_1 - 0.5b_f)\, b_f t_f}{A_s} \tag{3.49}$$

$$z_0 = \frac{0.5d_w^2 t_w + (d_w + 0.5t_f)\, b_f t_f}{A_s} \tag{3.50}$$

The parameter b_1 is the smaller outstand dimension of flange with respect to the web's centreline. The factor m accounts for unsymmetry of the stiffener cross section.

$$m = 1 - u\left(0.7 - 0.1\frac{d_w}{b_f}\right) \tag{3.51}$$

$$u = 1 - 2\frac{b_1}{b_f} \tag{3.52}$$

The factor C_0 is defined as:

$$C_0 = \frac{Et^3}{3s} \tag{3.53}$$

The critical buckling stress σ_{cL} for associated plating corresponding to n-half waves is defined as:

$$\sigma_{cL} = \frac{\pi^2 E\left(\frac{n}{\alpha} + \frac{\alpha}{n}\right)^2 \left(\frac{t}{s}\right)^2}{12\,(1 - \nu^2)} \tag{3.54}$$

Where α is the aspect ratio of the associated panels and n is the number of half waves which yield the lowest value for the elastic flexural-torsional buckling stress σ_{ET}. By default the Stress Check Model determines σ_{ET} for the first ten half waves and the lowest value is used to carry out the buckling check. Figure 3.5 show how the number of half waves influence the elastic flexural-torsional buckling stress for three different examples which are dealt with in section 4.1.

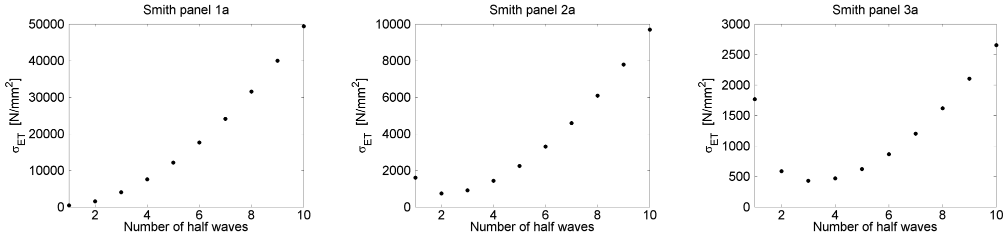


Figure 3.5: Elastic flexural-torsional buckling stress for various number of half waves

Chapter 4

Verification, Examples and Comparison

The functionality of the Stress Check Model is explained in this Chapter. In the first section the Stress Check Model is verified by comparing it to examples from a design code. The second section shows the usage of the Stress Check Model, starting from a finite element model. In the third section the Stress Check Model is compared to commercial buckling check software and in the fourth section the Stress Check Model is compared to classical plate buckling examples.

4.1 Verification

The ABS package of the Stress Check Model is verified using examples from *Commentary on the guide for buckling and ultimate strength assessment for offshore structure* on Smith's panels [13] (hereby called the *Commentary on the ABS guide*). The properties of the Smith's panels can be found in table 4.1. The Young's modulus and the Poisson's ratio are the same for each panel, equal to 206,000N/mm^2 and 0.3.

Table 4.1: Properties of Smith's Panels

Panel	**1a**	**1b**	**2a**	**2b**	**3a**	**3b**	**4a**	**4b**	**5**	**6**	**7**
Panel Properties											
L	6096.0	6096.0	6096.0	6096.0	6096.0	6096.0	1219.2	1219.2	6096.0	6096.0	6096.0
l	1219.2	1219.2	1524.0	1524.0	1524.0	1524.0	1219.2	1219.2	1524.0	1219.2	1524.0
B	3048	3048	3048	3048	3048	3048	1016	1016	3048	3048	3048
s	609.6	609.6	304.8	304.8	304.8	304.8	254.0	254.0	609.6	609.6	609.6
t	8.00	7.87	7.72	7.37	6.38	6.40	6.43	6.40	6.43	6.32	6.30
$\sigma_{\mathbf{yield}}$	249.1	252.2	261.3	259.7	250.6	252.2	259.7	264.3	247.6	256.7	290.1
Stiffener Properties											
$\mathbf{d_w}$	153.7	152.4	115.6	114.3	77.7	77.2	76.7	77.0	116.1	76.2	115.1
$\mathbf{t_w}$	7.21	7.11	5.44	5.38	4.52	4.65	4.85	4.55	5.33	4.55	5.16
$\mathbf{b_f}$	78.99	76.20	45.97	44.70	25.91	27.94	27.69	26.16	46.23	27.43	45.21
$\mathbf{t_f}$	14.22	14.22	9.53	9.53	6.35	6.35	6.35	6.35	9.53	6.35	9.53
$\sigma_{\mathbf{yield}}$	253.7	252.3	253.1	263.3	246.8	247.3	252.5	257.3	244.9	255.2	303.3
Loading											
$\sigma_{\mathbf{x}}$	190.3	184.2	239.4	218.5	170.3	150.9	207.1	213.6	176.3	125.0	197.1
q	0	0.103	0.048	0	0.021	0	0	0.055	0	0	0

Where:

L	-	Total length of the plate field [mm]
l	-	Length between transverse frames [mm]
B	-	Total width of the plate field [mm]
s	-	Width between longitudinals [mm]
t	-	Plate thickness [mm]
σ_{yield}	-	Specific minimum yield point [N/mm^2]
d_w	-	Web height [mm]
t_w	-	Web thickness [mm]
b_f	-	Flange width [mm]
t_f	-	Flange thickness [mm]
σ_x	-	Stress in the longitudinal direction [N/mm^2]
q	-	Lateral pressure [N/mm^2]

There are in total eleven different models with four different panel configuration, meaning that the total length and width of the plate field and the length and width of the sub-panels are the same. Figure 4.1 shows the four panel configurations. However the cross section properties of the stiffeners, the plate thickness and the yield stress vary between all the models.

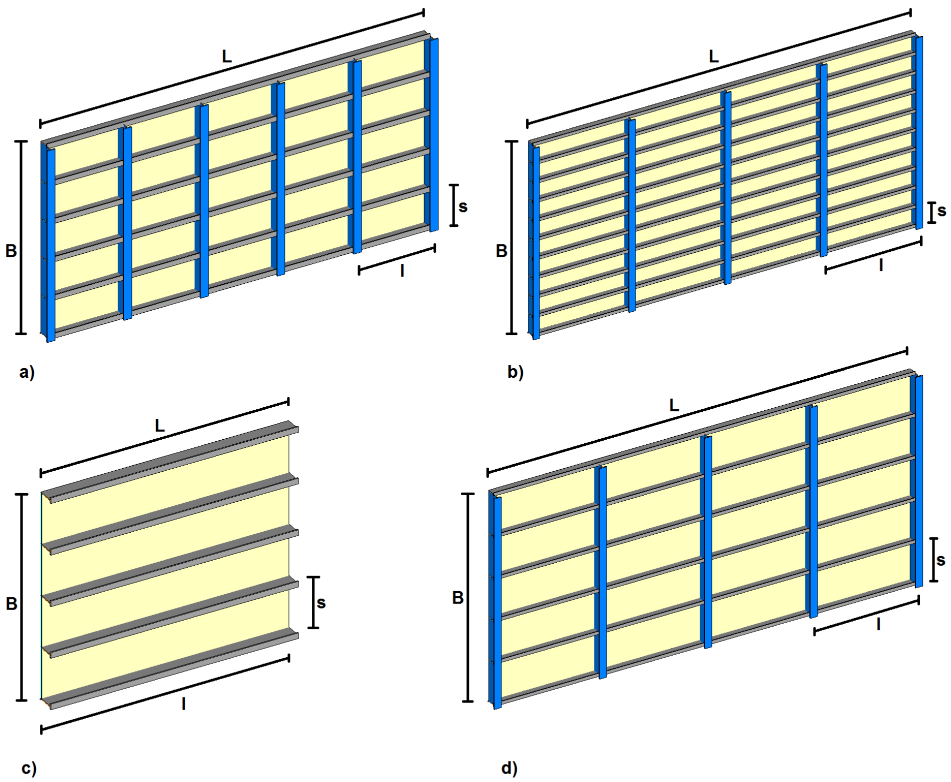


Figure 4.1: Smith panels, a) models 1a, 1b and 6, b) models 2a, 2b, 3a and 3b, c) models 4a and 4b, d) models 5 and 7

According to the Commentary on the ABS guide the design load which is applied to each sub-panel is the same as the load which is applied to the boundary of the plate field. Figure 4.2 shows a schematic view of the loading for model 1a.

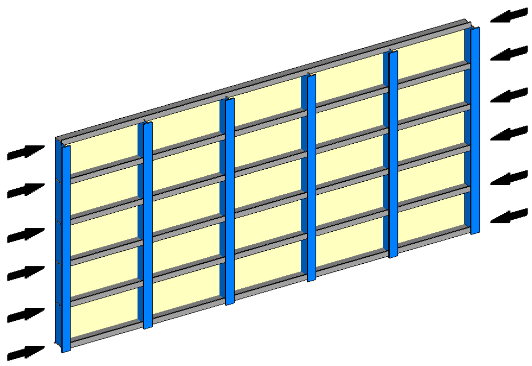


Figure 4.2: Uniformly distributed load applied to model 1a

All the Smith's panels are modelled with the finite element software Femap. Figure 4.3 shows how the boundary conditions and load are applied to model 1a. At the left edge, displacements in all directions and rotation about the x- and z-axis are constrained. At the right edge, displacements in y- and z-directions and rotation about the x- and z-axis are constrained. The model is loaded with uniformly distributed load at the right edge of the plate field. The load is tuned until the stress results in the x-direction of the sub-panel marked with dotted red square from the Stress Check Model match the examples from the Commentary on the ABS guide.

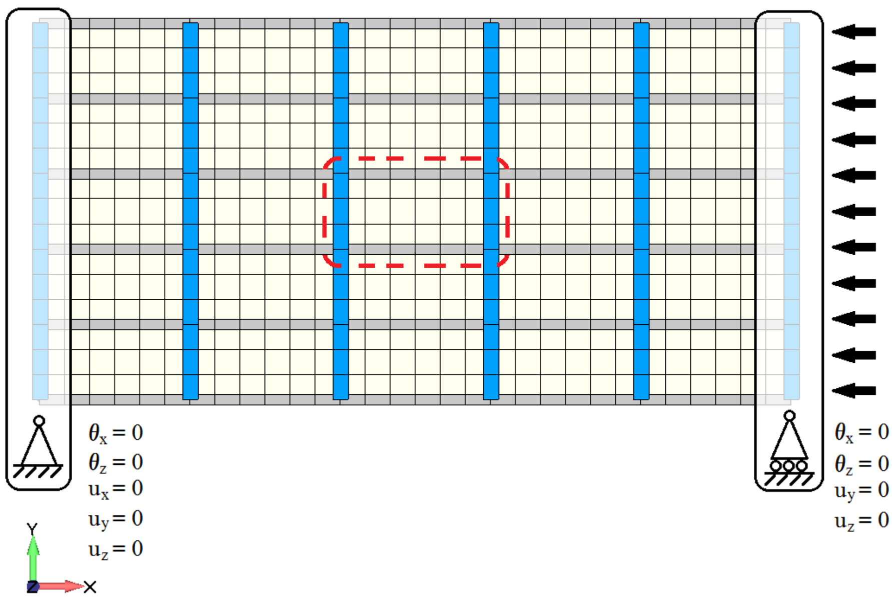


Figure 4.3: Boundary conditions and applied load to model 1a for finite element analysis

Figure 4.4 shows the stress results for model 1a from the finite element analysis. It is noticeable that the stress results are close to being uniformly distributed around the sub-panel located in the middle of the plate field. However close to the boundaries of the plate field the stress results are far from being uniformly distributed. Therefore the results from the Commentary on the ABS guide are compared to the sub-panel of the finite element model located in the middle of the plate field.

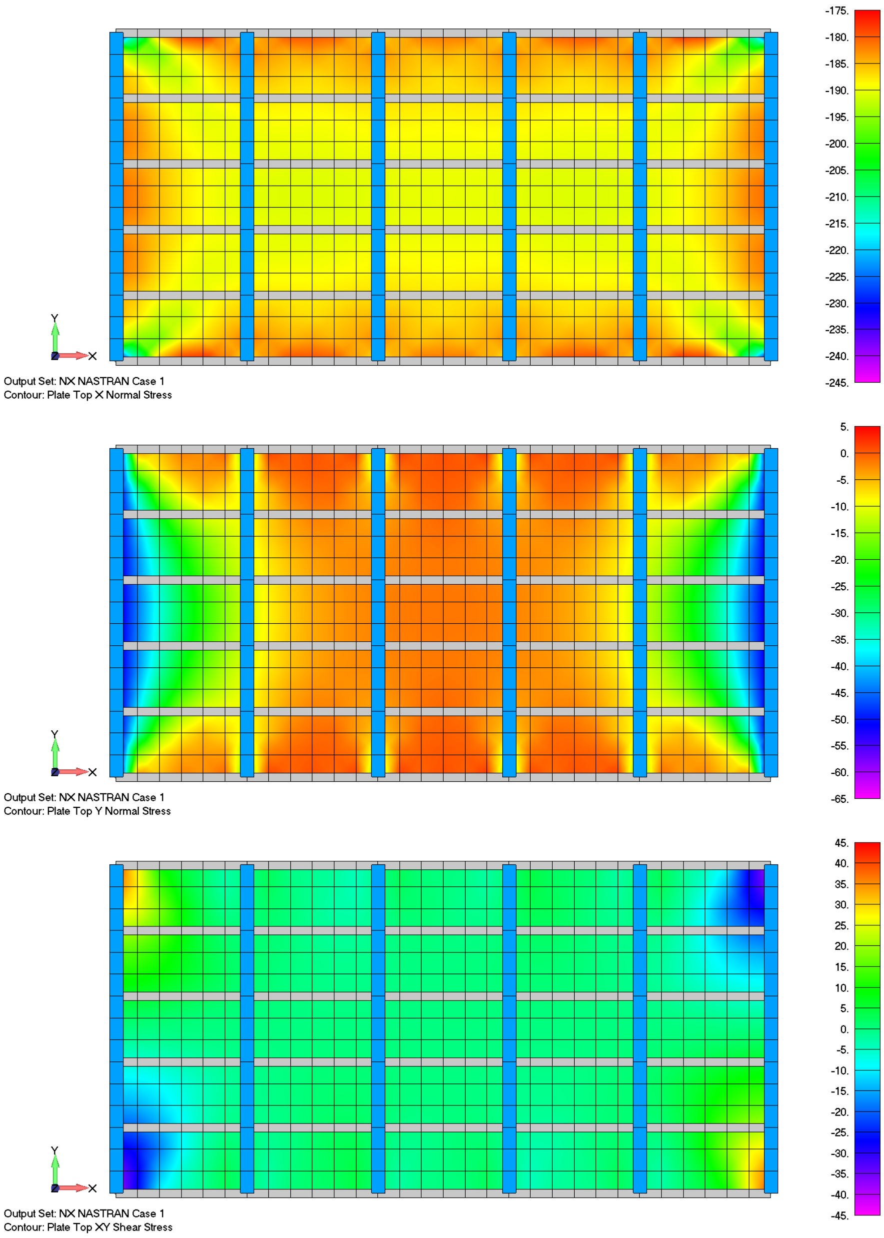


Figure 4.4: Finite element results for model 1a

The same procedure was carried out for the rest of the models. Table 4.2 shows a comparison between the load applied to the sub-panels according to the Commentary on the ABS guide and the load applied on the sub-panel located in the middle of the plate fields according to the Stress Check Model. Results show that stresses in the x-direction and lateral pressure are almost identical. However stresses in the y-direction and shear stresses are introduced in the Stress Check Model.

Table 4.2: Load applied to panels [N/mm^2]

	ABS				Stress Check Model (SCM)			
Model	σ_x	σ_y	τ	**q**	σ_x	σ_y	τ	**q**
1a	190	0	0	-	190	3	0	-
1b	184	0	0	0.103	184	17	0	0.103
2a	239	0	0	0.048	239	17	2	0.048
2b	219	0	0	-	218	17	1	-
3a	170	0	0	0.021	169	9	1	0.021
3b	151	0	0	-	151	9	0	-
4a	207	0	0	-	207	61	5	-
4b	214	0	0	0.055	213	62	7	0.055
5	176	0	0	-	176	10	1	-
6	125	0	0	-	125	3	0	-
7	197	0	0	-	197	11	1	-

These additional stresses can be traced to the modelling of the finite element models. First of all the boundary conditions highly affect the stress results of the finite element model. As an example the shorter edges of the sub-panels of models 4a and 4b define the boundaries of the plate field and are therefore highly affected by the boundary conditions (see figure 4.1c). Secondly the load applied in the x-direction is applied at the level of the plate elements. This causes eccentric loading on the beam elements. Consequently the results of the buckling checks are affected.

The difference between the ABS guide and the Stress Check Model is defined as:

$$\% = \frac{ABS_{value} - SCM_{value}}{ABS_{value}} \times 100 \tag{4.1}$$

Table 4.3 and figure 4.5 show the results for the buckling checks according to section 3.1. Due to stress results in the y-direction and shear stresses from the Stress Check Model more conservative outcome of the buckling checks is expected. Great mismatch is between the results of the example from the ABS guide and the Stress Check Model for models 4a and 4b.

The remaining results for the buckling state limits match within 5% difference and the Stress Check Model yields more conservative results except for model 3a. The explanation is that the σ_x result from the Stress Check Model is slightly less for panel 3 than from the ABS example (see table 4.2).

The results for the ultimate state limit state match within 10%. For models 2a, 2b, and 3a the Stress Check Model yields less conservative results. The explanation is that the factor φ in equation 3.11 is defined positvie in these cases and has positive influences.

The buckling checks for lateral pressure are equal or less conservative for the Stress Check Model in all cases. The interaction between different stress

Table 4.3: Verification of results for unstiffened panels

	Buckling Limit State			Ultimate Limit State			Lateral Pressure		
Model	**ABS**	**SCM**	**%**	**ABS**	**SCM**	**%**	**ABS**	**SCM**	**%**
1a	1.82	1.82	**0.0**	1.56	1.57	**-0.6**	-	-	**-**
1b	1.82	1.90	**-4.4**	1.46	1.59	**-8.9**	0.72	0.69	**4.2**
2a	1.08	1.09	**-0.9**	1.00	0.98	**2.0**	0.17	0.15	**11.8**
2b	0.94	0.95	**-1.1**	0.88	0.86	**2.3**	-	-	**-**
3a	0.67	0.66	**1.5**	0.66	0.64	**3.0**	0.06	0.06	**0.0**
3b	0.52	0.52	**0.0**	0.50	0.50	**0.0**	-	-	**-**
4a	0.81	0.98	**-21.0**	0.76	0.81	**-6.6**	-	-	**-**
4b	0.85	1.01	**-18.8**	0.79	0.85	**-7.6**	0.13	0.11	**15.4**
5	3.76	3.81	**-1.3**	1.90	2.05	**-7.9**	-	-	**-**
6	2.02	2.02	**0.0**	0.94	0.97	**-3.2**	-	-	**-**
7	5.11	5.17	**-1.2**	2.05	2.24	**-9.3**	-	-	**-**

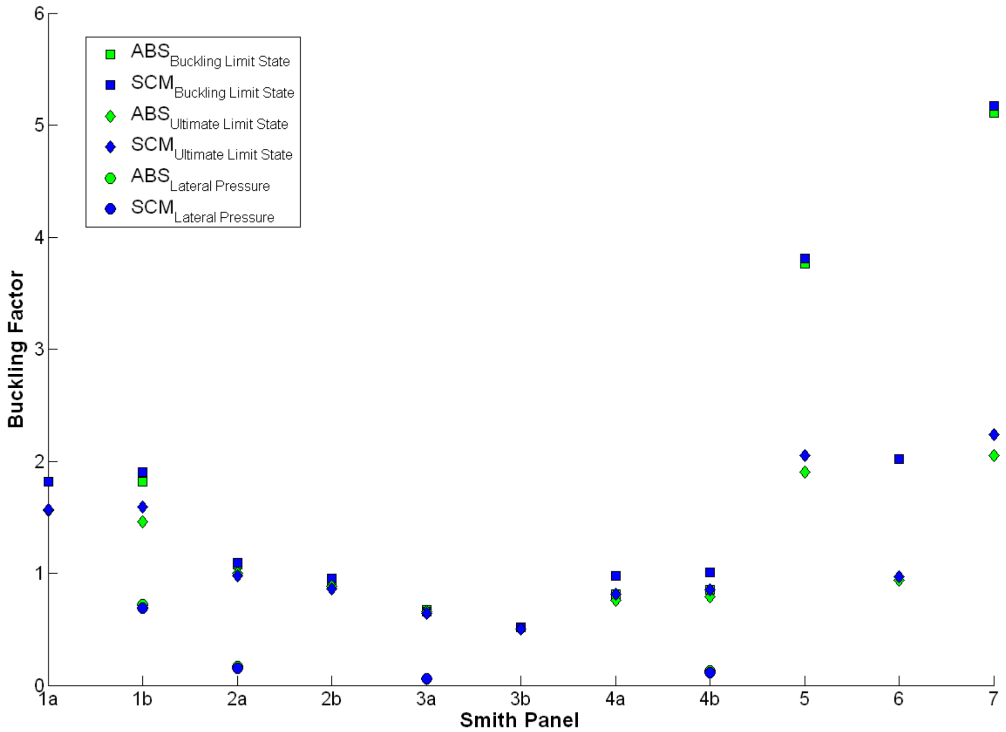


Figure 4.5: Results for unstiffened panels

components yield lower value for the equivalent stress according to von Mises in equation 3.19.

Table 4.4 and figure 4.6 show the results for the buckling checks according to section 3.2. Results for beam-column buckling check match within 5% limit and the Stress Check Model yields more conservative results except for model 3a. The reason for the difference can be traced to different result obtained between the approaches for the effective section modulus from equation 3.35. There are three possible explanations for the difference.

- There could be an error in the Stress Check Model programming code

- The formula in ABS guide could be wrong which consequently leads to wrong results for the Stress Check Model
- The value in the ABS example is wrong

Table 4.4: Verification of results for stiffened panels

	Beam Column Buckling			**Flexural-Torsional Buckling**		
Model	**ABS**	**SCM**	**%**	**ABS**	**SCM**	**%**
1a	1.04	1.05	**-1.0**	0.88	0.88	**0.0**
1b	1.14	1.16	**-1.8**	0.86	0.86	**0.0**
2a	1.15	1.15	**0.0**	1.01	1.01	**0.0**
2b	0.87	0.87	**0.0**	0.92	0.92	**0.0**
3a	1.02	1.00	**2.0**	0.79	0.79	**0.0**
3b	0.67	0.67	**0.0**	0.69	0.69	**0.0**
4a	0.85	0.85	**0.0**	0.89	0.88	**1.1**
4b	1.18	1.23	**-4.2**	0.91	0.90	**1.1**
5	1.19	1.23	**-3.4**	1.04	1.04	**0.0**
6	0.93	0.94	**-1.1**	1.05	1.05	**0.0**
7	1.19	1.24	**-4.2**	1.09	1.09	**0.0**

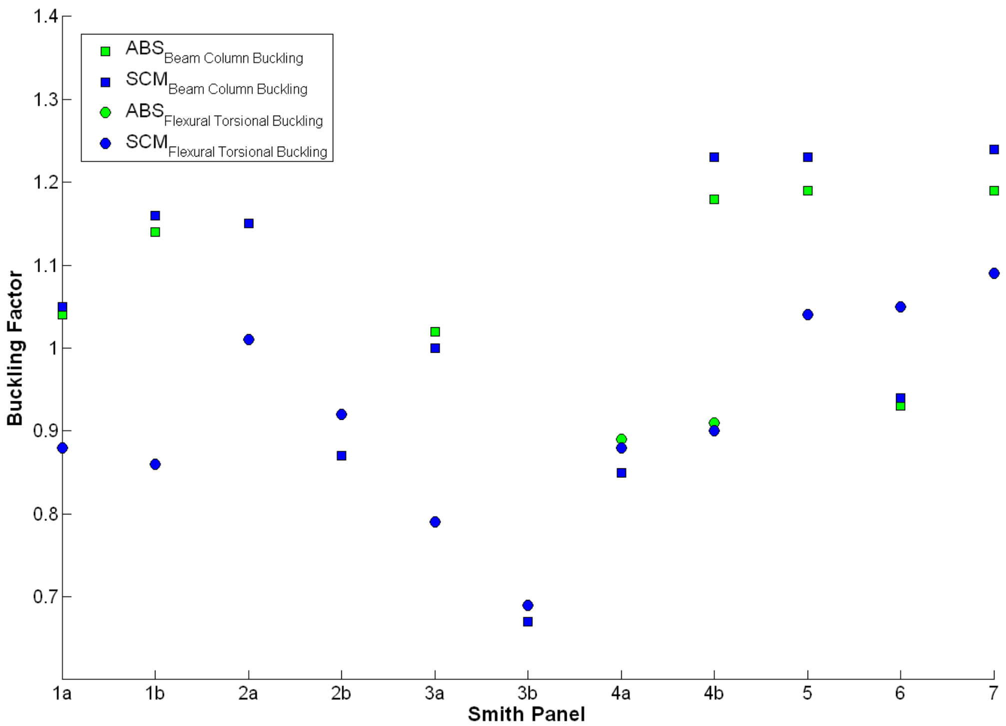


Figure 4.6: Results for stiffened panels

Excellent correspondence is between results for the torsional-flexural buckling state limit which was expected since the buckling check only depends on the stresses in the x-direction.

4.2 Example

The following example demonstrates the functionality of the Stress Check Model and the ABS package. Consider the finite element model showed in figure 4.7. It is part of a larger model which is loaded in compression in both in-plane directions and with constant lateral pressure.

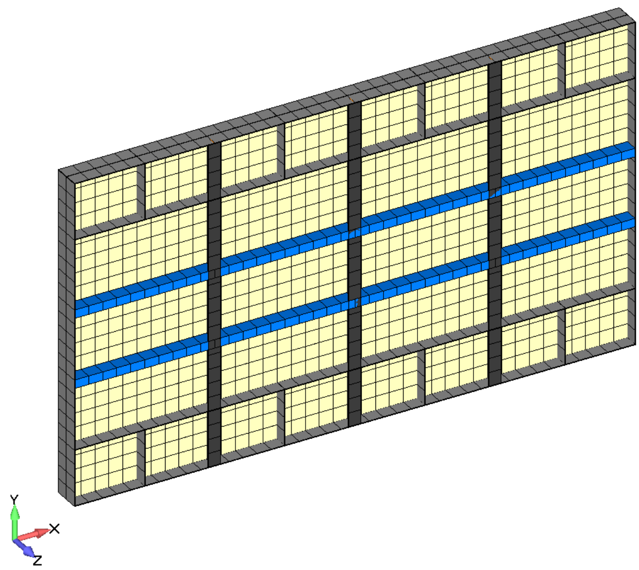


Figure 4.7: Finite element model of the example

Two different panel sizes can be recognised from the model. The outer boundaries of the model and the boundaries of the smaller panels are modelled with plate elements. The longitudinal angle stiffeners (blue) and the transverse stiffeners (dark grey) are modelled with beam elements. Figures 4.8 - 4.10 show the stress results for the finite element analysis.

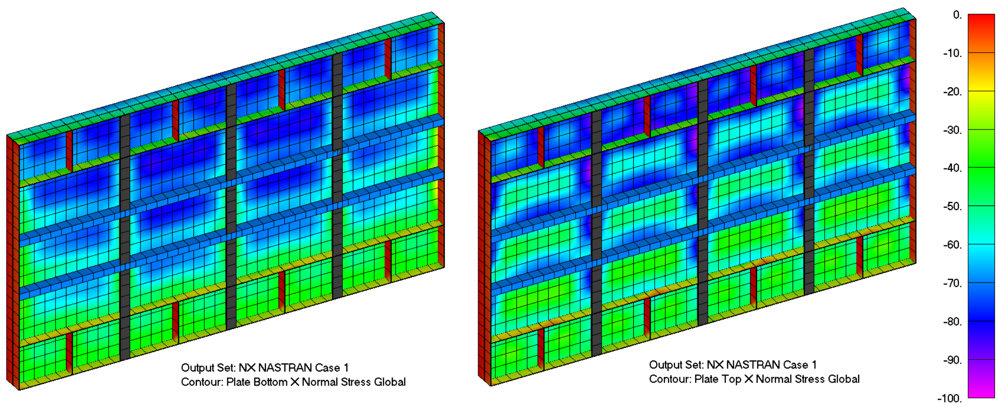


Figure 4.8: Stress results in x-direction at bottom and top fibre [N/mm^2]

The geometrical and material properties of the model along with the stress results are exported to the Stress Check Model. The Stress Check Model numbers the stiffeners and the panels according to figure 4.11.

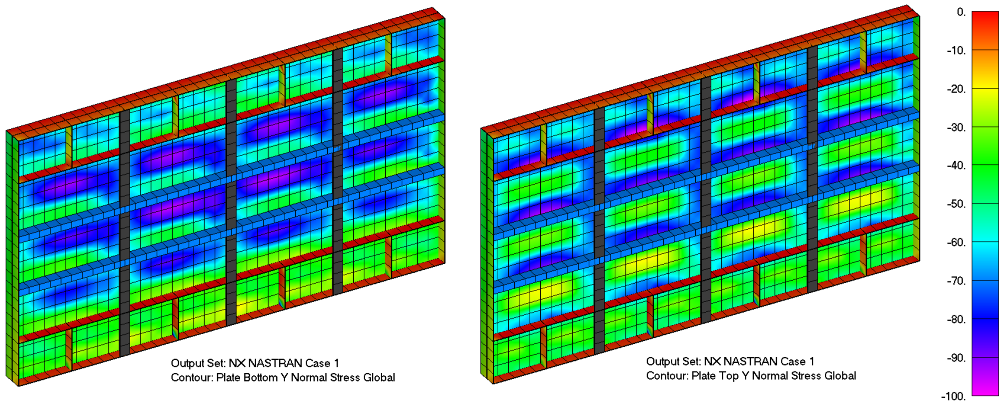


Figure 4.9: Stress results in y-direction at bottom and top fibre [N/mm^2]

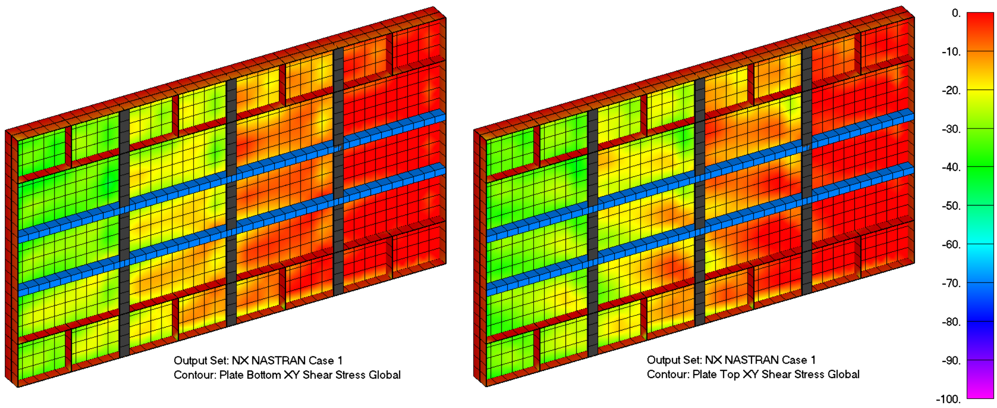


Figure 4.10: Shear stress results at bottom and top fibre [N/mm^2]

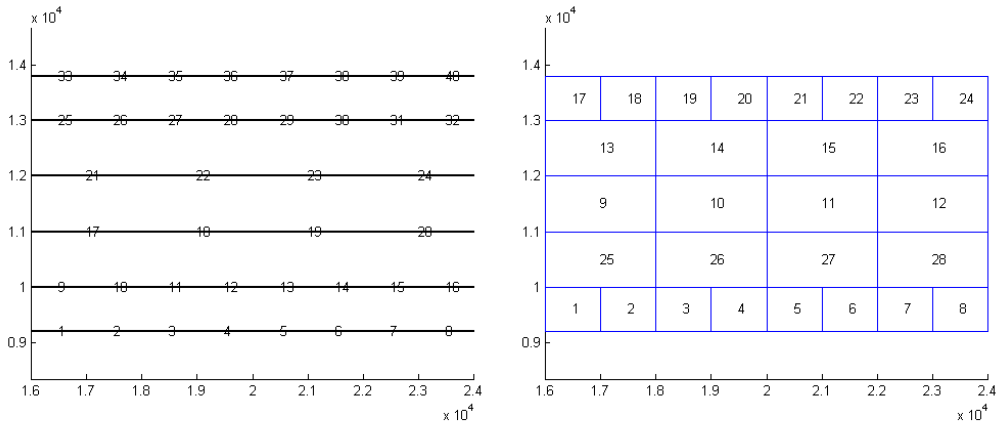


Figure 4.11: Stiffener and panel numbering from Stress Check Model (coordinates in [mm])

The users settings for the Stress Check Model are the following:

- The mean value of stress results at top and bottom fibre of plate elements is used

- The design in-plane stresses σ_x and σ_y are determined according to clause 4.6.(3) in Eurocode 3 part 1.5
- The design shear stress is based on the weighted average shear results
- The design lateral pressure is based on maximum value (not relevant for this example since constant lateral pressure is applied to the model)

The Stress Check Model gives a warning that buckling check will not be performed on stiffeners 1-16 and 25-40 since they are modelled with plate elements. Next the Stress Check Model warns the user which panels and stiffeners fail the buckling checks. In this case panels 10 and 11 and 13-16 failed the buckling state limit from section 3.1.1. The results for every buckling check from sections 3.1 and 3.2 are displayed in tables. The Stress Check Model also displays all intermediate results. The complete report can be found in appendix B.

ABS Plate buckling stress check

Panel Nr.	Critical check	Ultimate check	Lateral check
1	0.15	0.14	0.07
2	0.13	0.13	0.07
3	0.13	0.12	0.07
4	0.12	0.12	0.07
5	0.12	0.12	0.07
6	0.13	0.12	0.07
7	0.13	0.13	0.07
8	0.15	0.14	0.07
9	0.82	0.72	0.15
10	1.11	0.85	0.15
11	1.11	0.85	0.15
12	0.82	0.72	0.15
13	1.08	0.82	0.15
14	1.19	0.85	0.15
15	1.19	0.85	0.15
16	1.08	0.82	0.15
17	0.31	0.29	0.07
18	0.30	0.29	0.07
19	0.29	0.28	0.07
20	0.29	0.28	0.07
21	0.29	0.28	0.07
22	0.29	0.28	0.07
23	0.30	0.29	0.07
24	0.30	0.29	0.07
25	0.68	0.54	0.14
26	0.81	0.59	0.14
27	0.81	0.59	0.14
28	0.68	0.54	0.14

ABS Stiffened panels buckling stress check

Stiffener Nr.	Beam/ Column check	Flexural/ Torsional check
17	0.41	0.32
18	0.74	0.33
19	0.67	0.33
20	0.41	0.32
21	0.65	0.32
22	0.74	0.33
23	0.70	0.33
24	0.67	0.32

Due to the shifted linear regression of the finite element results the Stress Check Model determines the in-plane design loads always in a conservative way. Therefore if a panel or a stiffener fails a buckling check it can be useful to look closer at how the Stress Check Model determines the stress distribution. The Stress Check Model can plot the design in-plane stresses and compare them graphically to the stress results from the finite element analysis. Figure 4.12 shows the in-plane stress results for panel 16, which failed the buckling state limit check.

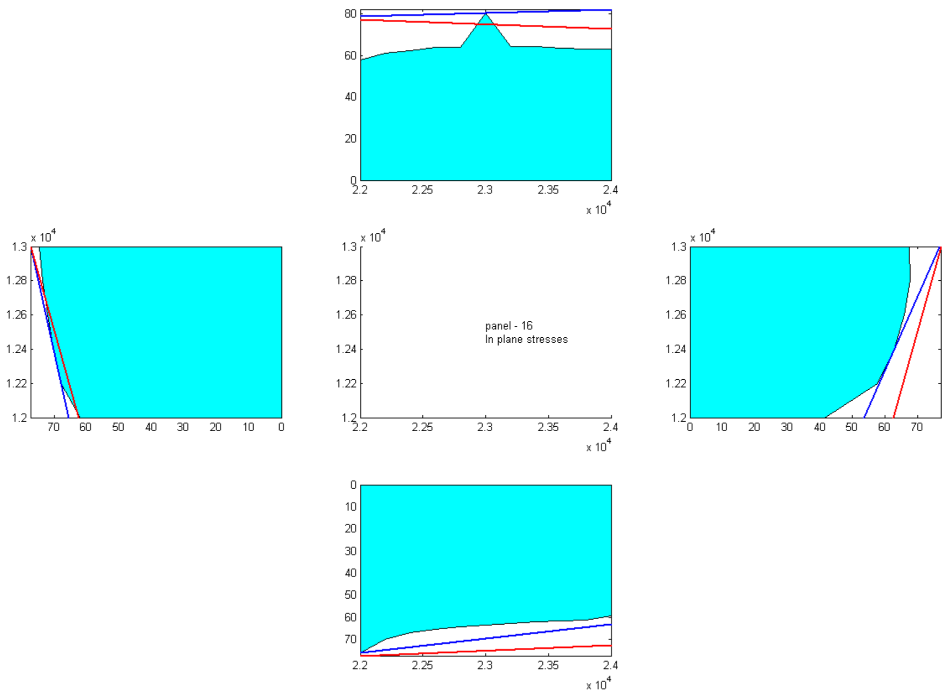


Figure 4.12: Stress results from the Stress Check Model

The middle plot of figure 4.12 represents the panel number and its coordinates. The plots to the left and right represent the σ_x distribution and the plots at the top and bottom represent the σ_y distribution. The solid cyan coloured plot represents the stress results from the finite element analysis. The solid blue line is the results of linear regression through the finite element results which has been shifted to include the maximum compressive stress values. The red line represents the symmetric design load.

The σ_y distribution for panel 16 has a maximum design value equal to 77 N/mm^2. It is up to the user to determine if this really is a representative value for the panel since it is highly affected by a stress concentration due to transverse stiffener located at the middle of the top edge (see figure 4.11). If one would use the design value for σ_y equal to 60 N/mm^2 the panel would pass the buckling state limit with a result of 0.80 instead of 1.09.

4.3 Comparison to the SDC Verifier

In this section the Stress Check Model is compared to the buckling tool of the SDC Verifier. The comparison is demonstrated using a simple model governed by in-plane bending action. Figure 4.13 shows a finite element model of a plate field. The length of the plate field is 19m and the height is 1.8m. The plate field is modelled as simply supported beam loaded with uniformly distributed load in the downward direction equal to 40.8kN/m. The length, width and thickness of the panels are 1000mm, 600mm and 12mm respectively. Only part of the model is analysed for plate buckling (see the shaded area of figure 4.13). The influences of the mesh size is taken into account in the analysis. Table 4.5 shows an overview of the mesh sizes.

Table 4.5: Mesh sizes

Mesh	**Elements per panel**
1x1	1
3x5	15
6x10	60
9x15	135
12x20	240
15x25	375
18x30	540

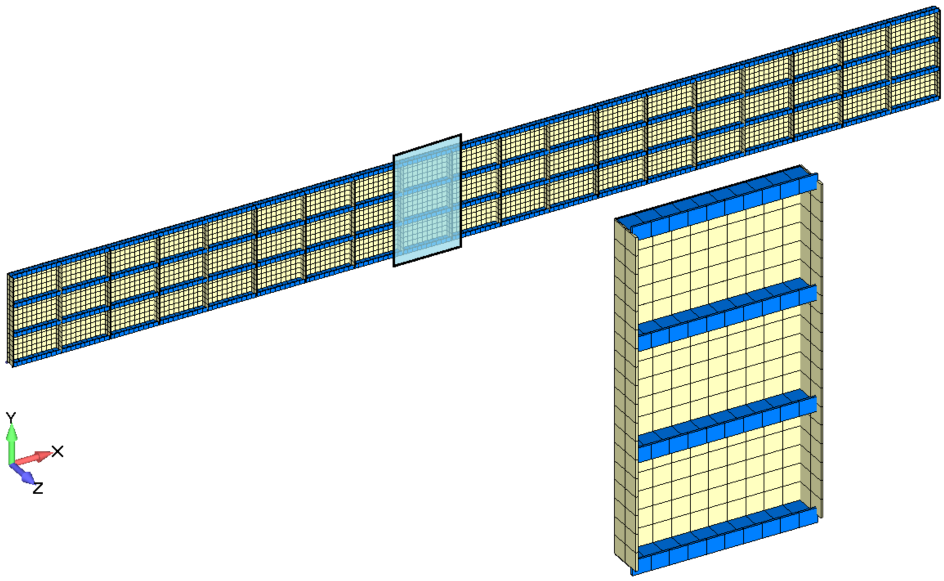


Figure 4.13: Finite element model of simple plate field using 6x10 elements for each panel

The SDC Verifier can perform buckling check according to the buckling limit state and the ultimate limit state as described in sections 3.1.1 and 3.1.2. The user has to manually define the panel sizes and select the finite elements which belong to every panel. The buckling checks are performed on every finite element as if they had the length and width from the panel they are linked to. Uniform stress distribution is assumed for every element. For the comparison to the Stress Check Model two different settings are considered for the SDC verifier

- **Settings 1:** Average buckling factor of all the finite elements selected to a panel of interest
- **Settings 2:** The result of finite element selected to a panel of interest which yields the highest buckling factor

The buckling check equations used for the SDC Verifier are in nature the same as the one used for the Stress Check Model [14]. However they are derived from a ABS guide for assessment of ship structures and some intermediate calculations differ. In order to compare the Stress Check Model with the SDC Verifier the ABS package of the Stress Check Model is adjusted to the equations used for the SDC Verifier. The coefficient of interaction between longitudinal and transverse stresses becomes (see equation 3.12):

$$\varphi = 1.5 - \frac{\beta}{2} \tag{4.2}$$

The factors C_x and C_y from equations 3.17 and 3.18 change to:

$$C_x = \begin{cases} \dfrac{2.25}{\beta} - \dfrac{1.25}{\beta^2} & \text{for } \beta > 0 \\ 1 & \text{for } \beta \leq 1.25 \end{cases} \tag{4.3}$$

$$C_y = C_x \frac{s}{l} + 0.115 \left(1 - \frac{s}{l}\right) \left(1 + \frac{1}{\beta^2}\right)^2 \leq 1 \tag{4.4}$$

Two different settings are analysed for the Stress Check Model.

- **Settings 1:** Average stress value in the cross section of the plate elements, weighted average shear stresses acting on the edge of the panels and stress distribution according to clause 4.6(3) in Eurocode 3, part 1.5
- **Settings 2:** Maximum stress value in the cross section of the plate elements, maximum shear stresses acting on the edge of the panels and in-plane stress distribution according based on maximum stress value acting on the panel

Figure 4.14 shows an overview of the different settings for the SDC Verifier and the Stress Check Model and in figure 4.15 the numbering of the panels is showed. Panel 1 is under tension with respect to the stresses in the x direction which is the governing factor in the unity checks for this example, therefore the unity checks are close to zero in all cases and the mesh size has negligible influences.

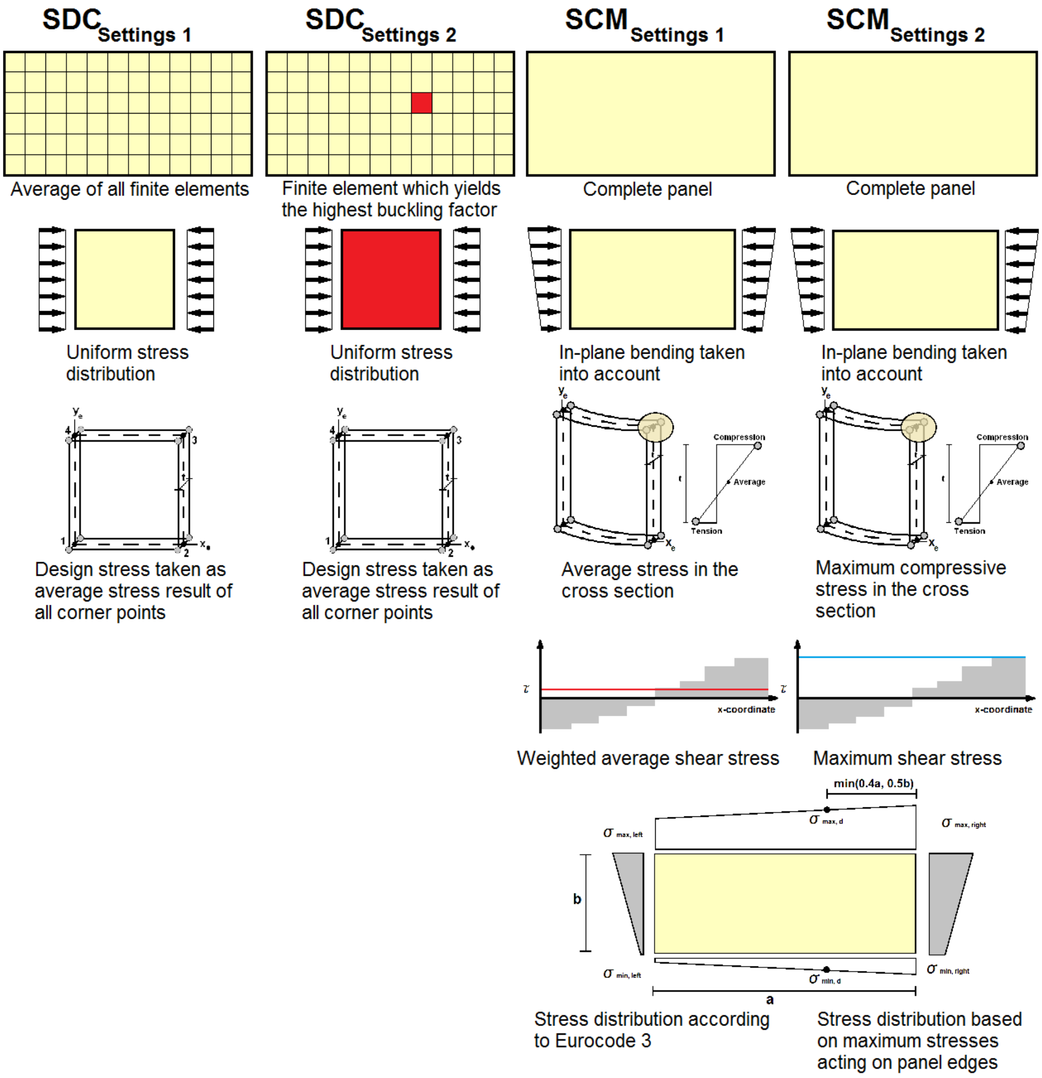


Figure 4.14: Overview of settings for buckling analysis

Figure 4.16 shows the maximum design stress in the x-direction for panel 2 according to both the SDC verifier and the Stress Check Model for all settings. The mesh size has little influence on **settings 1** and **2** of the Stress Check Model and acceptable design load is obtained using only one finite element for the panel. The difference between the two settings is around 3.5%.

Panel 2 can be considered being under pure in-plane bending with respect to σ_x. Therefore **settings 1** of the SDC Verifier, which are based on average stress results of all the finite elements belonging to the panel, yields zero loading, independent on the mesh size. The results for **Settings 2** of the SDC Verifier are highly influenced by the mesh size. In appendix A an expression is derived to estimate the deviation of the design load of the SDC Verifier. The expression is valid for panels loaded with linear distributed in-plane stresses in one direction. The deviation (ϵ) depends on the ratio (ψ) between the maximum and minimum stresses applied to the panel and the number of

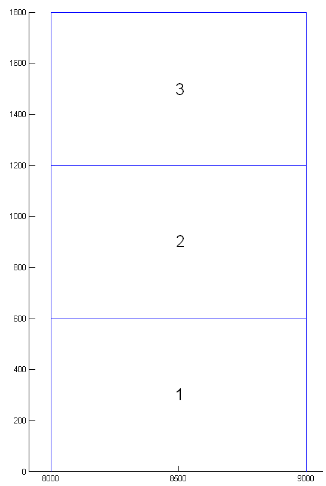


Figure 4.15: Panel numbering for buckling analysis

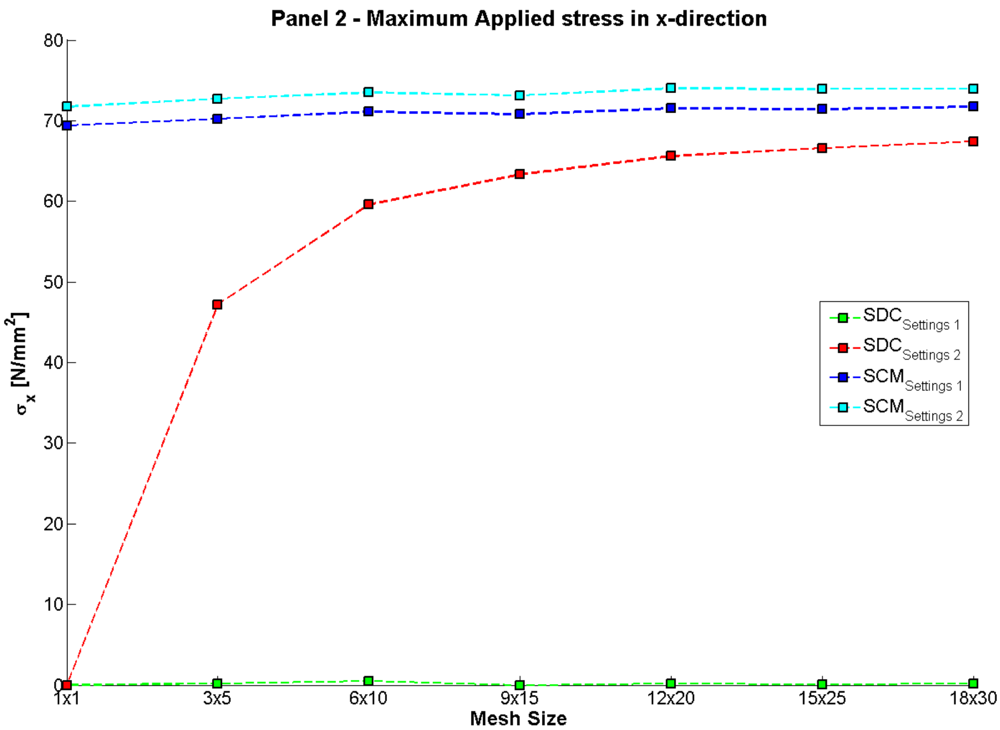


Figure 4.16: Maximum compressive stress in the x-direction for panel 2

finite elements (N_{el}) located on the edge of the panel.

$$\epsilon = \frac{1-\psi}{2\,(N_{el}-1)} \tag{4.5}$$

For pure in-plane bending ψ is equal to -1. For the 3x5 mesh, three finite elements are located along the panel edge which is loaded and the deviation is equal to 50%, that is the load should be increased by 50%. The value of the load for **settings 2** of the SDC Verifier corresponding to the 3x5 mesh is equal to 47.2N/mm^2 and increasing that value by 50% gives 70.8N/mm^2. For the 18x30 mesh, 18 finite elements are located along the panel edge which is loaded and the deviation is equal to 5.9%. The value corresponding to the 18x30 mesh is equal to 67.4N/mm^2 and increasing that value by 5.9% gives 71.4N/mm^2.

Figures 4.17 and 4.18 show the buckling check results for panel 2. As for the design load the Stress Check Model shows rather stable behaviour with regard to the mesh size. The difference between the two settings is approximately 20% for the buckling limit state and 15% for the ultimate limit state.

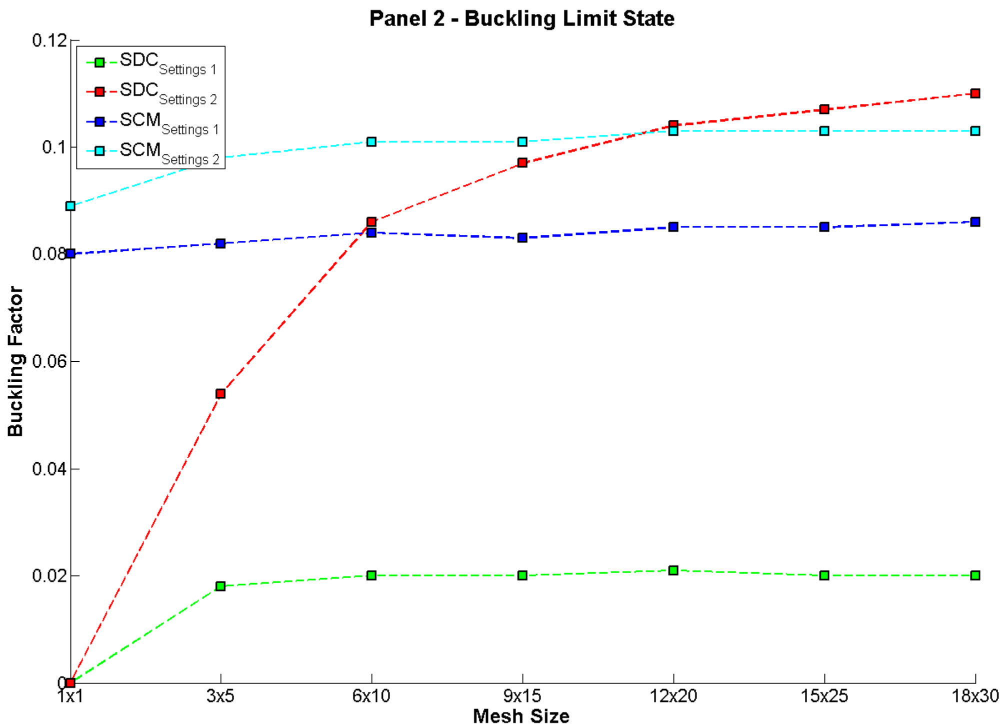


Figure 4.17: Buckling limit state results for panel 2

Since the mesh size highly influence the design load for **settings 2** of the SDC Verifier the results of the buckling checks are also highly affected and the buckling factor increases with each mesh refinement. The SDC Verifier assumes uniformly distributed load on the edges which is the most unfavourable condition. However the Stress Check Model takes into account the positive influence of the in-plane bending effects leading to much greater resistance compared to the SDC Verifier. Therefore the SDC Verifier yields higher buckling factor even though the design load is less.

For the buckling limit state the results becomes greater than both the settings from the Stress Check Model. For the 18x30 mesh the difference is 25% compared to **settings 1** of the Stress Check Model and 6% compared to **settings 2**. Nevertheless the results for the **settings 1** of the SDC Verifier have not converged so even greater difference is expected.

For the ultimate limit state **settings 2** of the SDC Verifier gives 9% higher buckling factor compared to **settings 1** of the Stress Check Model for the

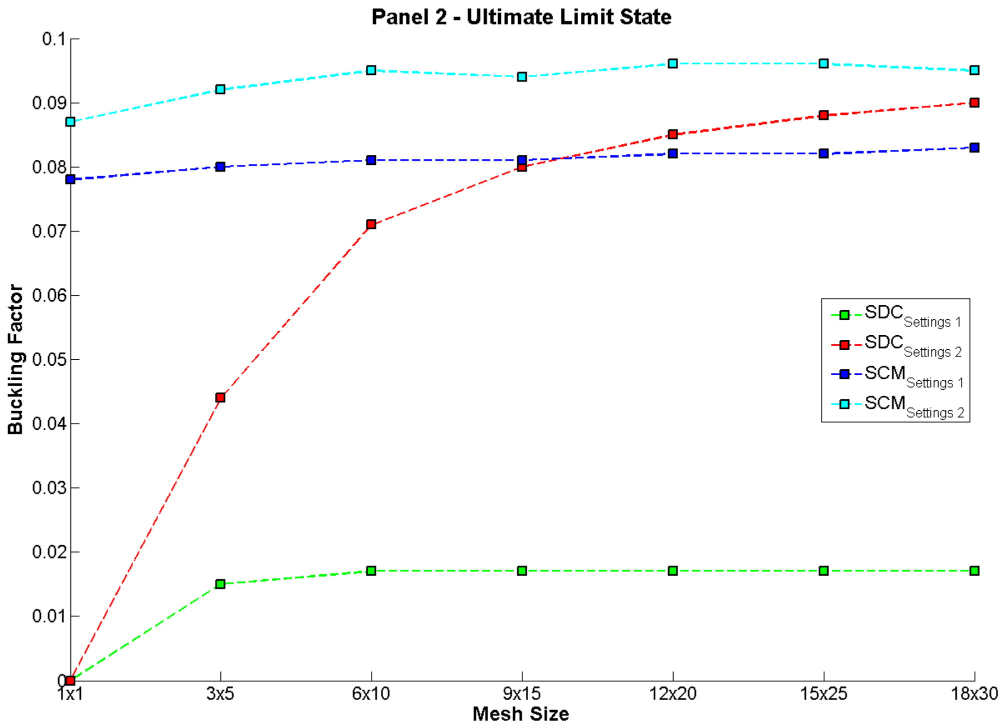


Figure 4.18: Ultimate limit state results for panel 2

18x30 mesh but 6% lower factor compared to **settings 2** of the Stress Check Model. However the results have not converged so the difference could be greater.

Although the σ_x design stress is close to being zero for **settings 2** of the SDC Verifier the σ_y and τ stresses have little influence on the buckling results. Still the results are significantly lower compared to the other settings.

Figure 4.19 shows the maximum design stress in the x-direction for panel 3 according to both the SDC verifier and the Stress Check Model for all settings. The mesh size slightly influences **settings 1** and **2** of the Stress Check Model and the difference between the two settings is around 4.5%. As for panel 2, **settings 2** of the SDC Verifier gives less design load for both **settings 1** and **2**. Panel 3 is under compression with respect to the stresses in the x direction with ratio of maximum and minimum stresses equal to 0.33. The deviation according to equation 4.5 for mesh 18x30 is 2.0% and increasing the corresponding load gives 219N/mm^2 which match well with load from **settings 1** of the Stress Check Model.

Figures 4.20 and 4.21 show the buckling check results for panel 3. The Stress Check Model starts to show rather stable behaviour with regards to the mesh size after mesh refinement 6x10. The difference in results between the two settings is approximately 13% for the buckling limit state and 6% for the ultimate limit state.

As for panel 2 the mesh size highly influence the buckling checks for **settings 2** of the SDC Verifier. For the buckling limit state **settings 2** of the SDC Verifier gives 10% higher buckling factor compared to **settings 1** of the

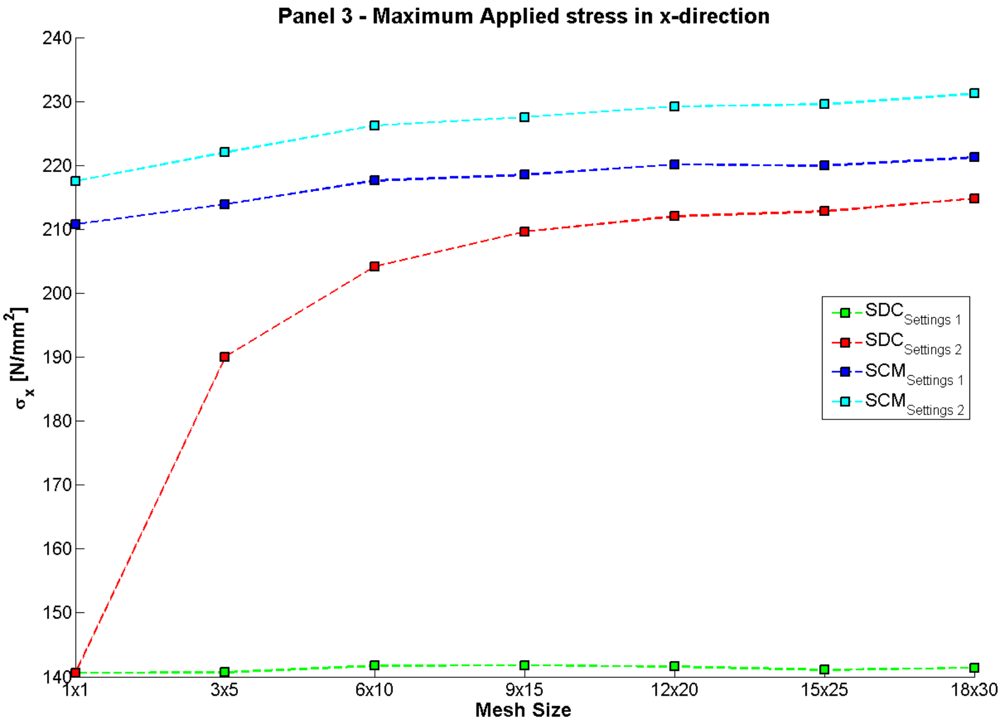


Figure 4.19: Maximum compressive stress in the x-direction for panel 3

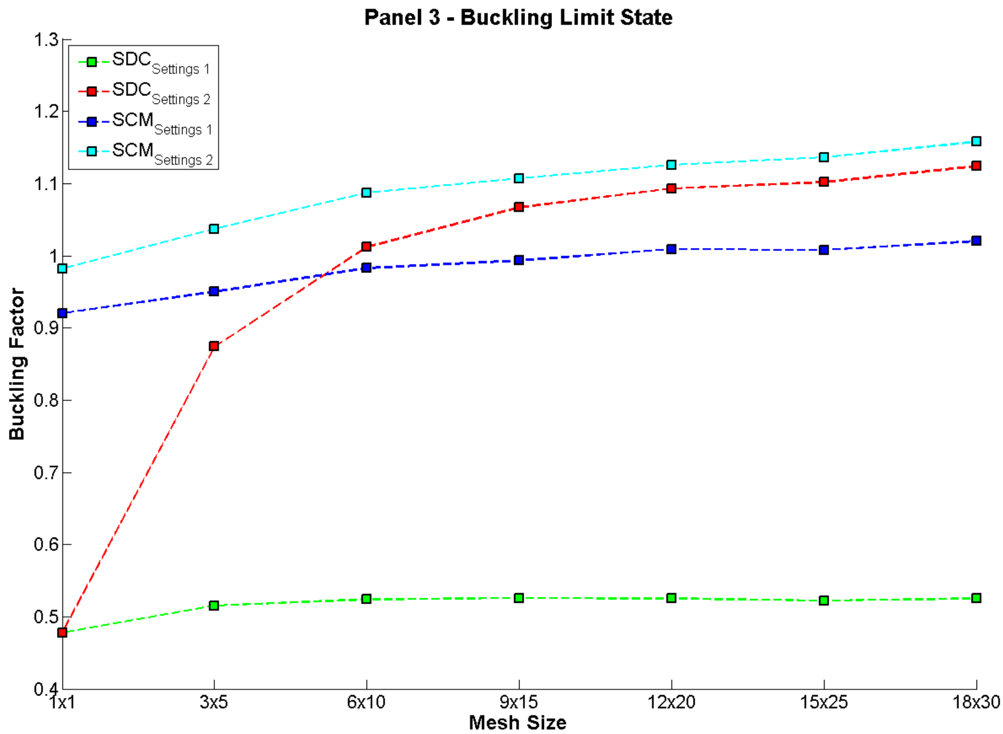


Figure 4.20: Buckling limit state results for panel 3

Stress Check Model for the 18x30 mesh but 3% lower factor compared to **settings 2** of the Stress Check Model. For the ultimate limit state **settings 1**

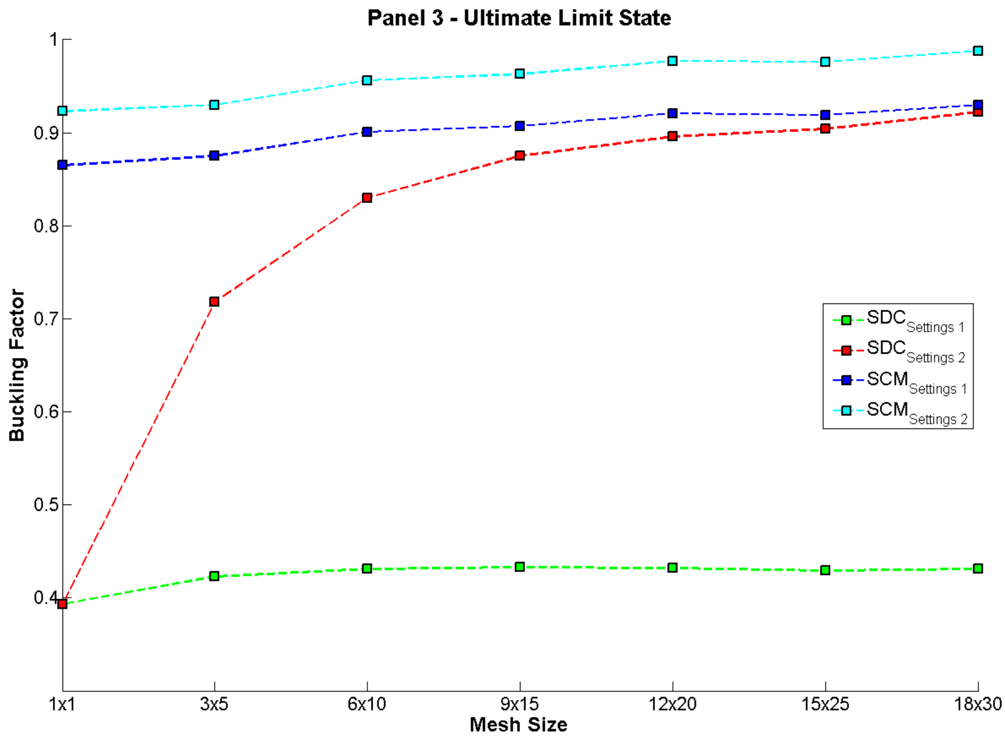


Figure 4.21: Ultimate limit state results for panel 3

of the SDC Verifier gives 1% lower buckling factor compared to **settings 1** of the Stress Check Model and 7% compared to **settings 2**.

4.4 Classical Examples

In this section the Stress Check Model is compared to analytical solutions of classic unstiffened plate buckling examples. Results are also compared to linear buckling analysis and geometrical non-linear analysis where geometrical imperfections and non-linear material behaviour are taken into account.

The length, width and thickness of the panel are 2000mm, 1000mm and 10mm respectively and the Young's modulus and the Poisson's ratio are taken as 210000N/mm^2 and 0.3.

The analysis is performed for three different load cases; **case 1** - uniform compression acting in the x-Direction, **case 2** - uniform compression acting in both x- and y-direction and **case 3** - in-plane bending.

4.4.1 Case 1 - Uniform Compression Acting in x-Direction

Figure 4.22 shows schematic view of the load case.

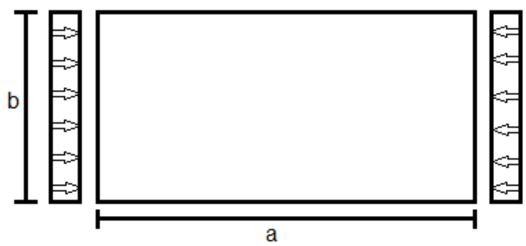


Figure 4.22: Schematic view of load case 1

Analytical solution

The analytical solution for simply supported plate under pure compression in one direction is the following [15]:

$$\sigma_{cr} = k\frac{\pi^2 E}{12\left(1-\nu^2\right)}\left(\frac{t}{b}\right)^2 = k \cdot 18.98\text{N/mm}^2 \tag{4.6}$$

Where σ_{cr} is the critical load the plate buckling coefficient k can be determined from figure 4.23 [15].

The aspect ratio of the panel is equal to 2 so the the plate buckling coefficient becomes equal to 4 and number of half waves for the buckling mode are equal to 2. The resulting critical stress is $\sigma_{cr} = \mathbf{75.9N/mm^2}$.

Linear Buckling Analysis

Figure 4.24 shows the boundary conditions and the load applied to the finite element model. Displacements along the edges in the z-direction are constrained (not showed in the figure). Displacements in both x- and y-direction are constrained in middle of the upper edge and displacements in x–direction

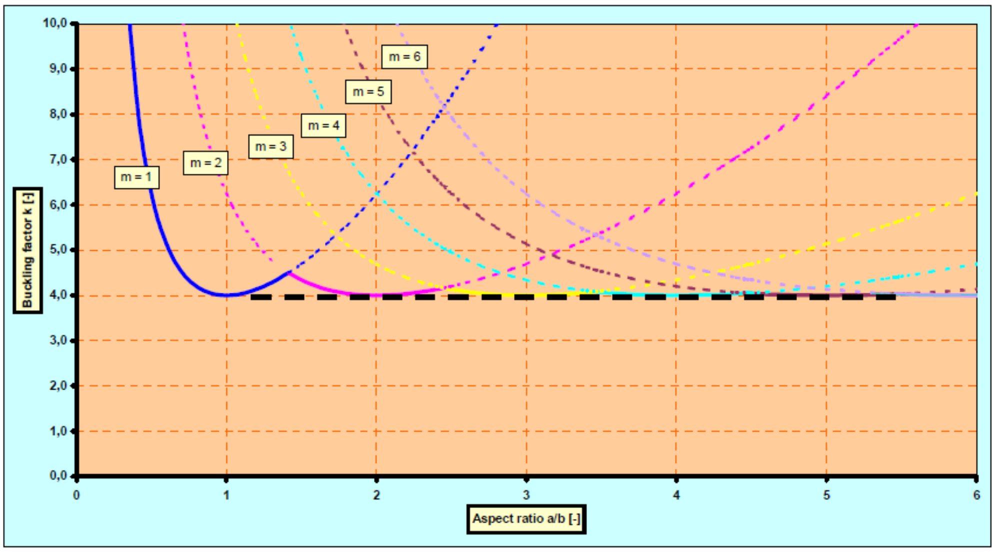
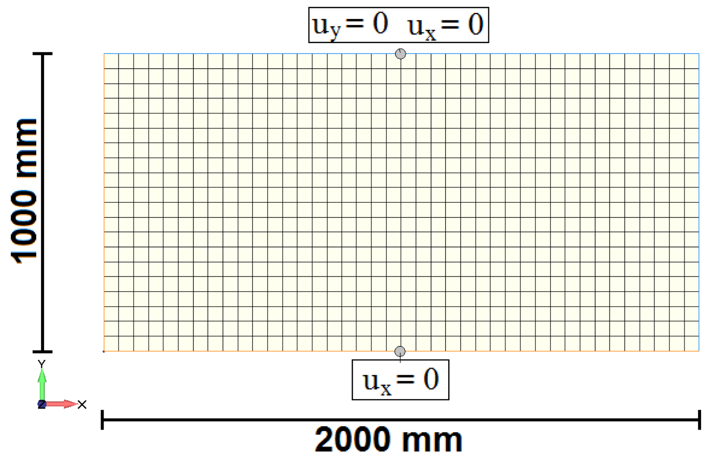


Figure 4.23: Plate buckling coefficient k for a simple supported plate under pure compression in one direction

is constrained in middle of the lower edge. The load applied to the panel is tuned until the lowest eigenvalue from linear buckling analysis becomes equal to 1.00. The mesh size is 50mmx50mm.



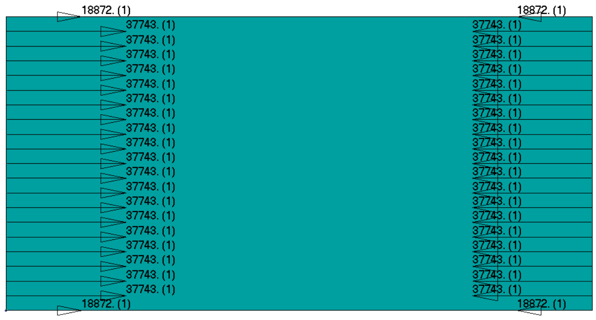


Figure 4.24: Boundary conditions for finite element analysis (above), applied load to finite element model (below)

The panel is free to deform in both x- and y-direction causing neither stresses in the y-direction nor shear stresses as showed in figure 4.25.

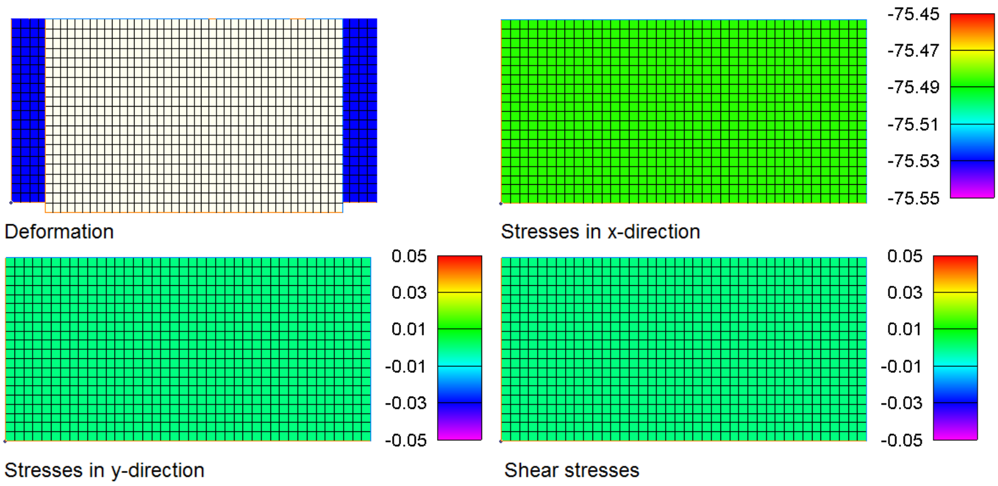


Figure 4.25: Linear elastic deformation of the panel and stress results

Figure 4.26 shows the buckling mode corresponding to the lowest eigenvalue. Two half waves can be seen in the figure which correspond to the analytical solution. The corresponding buckling load is equal to $\mathbf{75.5N/mm^2}$ which is slightly less than the analytical solution.

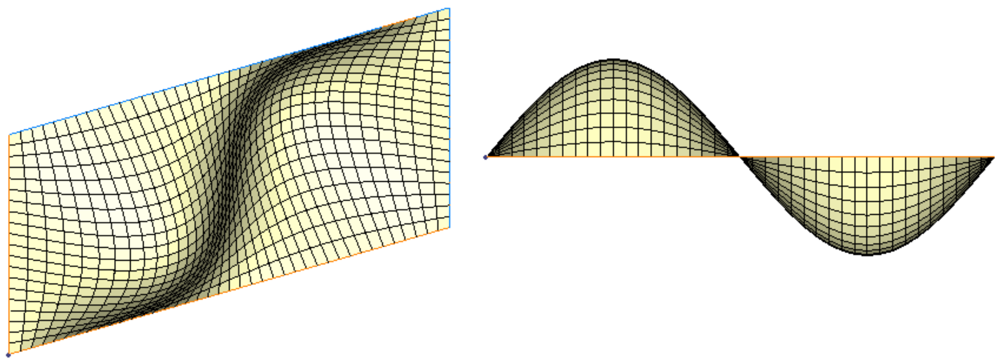


Figure 4.26: Buckling mode corresponding to the lowest eigenvalue

Stress Check Model - Buckling Limit State

The Stress Check Model uses the stress results from the linear elastic finite element analysis on the panel to perform the buckling limit state stress check. The yield stress is taken as 255N/mm^2.

$$\left(\frac{\sigma_{x,max}}{\sigma_{Cx}}\right)^2 + \left(\frac{\sigma_{y,max}}{\sigma_{Cy}}\right)^2 + \left(\frac{\tau}{\tau_C}\right)^2 = \left(\frac{75,5}{75,9}\right)^2 + 0 + 0 = 0.99 \tag{4.7}$$

Equivalent load factor for the unity check is 1.01 which means that the applied load of the finite element analysis can be increased by 1% for the Stress

Check Model. According to the ABS guide for this example, the critical buckling stress σ_{Cx} corresponds to the critical buckling stress from the analytical solution for yield stress greater than 127N/mm^2 (see equation 3.2).

Non-Linear Analysis

Geometrical imperfections are introduced to the model by using the deformation from the linear buckling analysis corresponding to the lowest eigenvalue. The magnitude of the imperfections is equal to 5mm (1/200 of the width of the panel) which is in accordance with recommended value from Eurocode 3, part 1.5 [9].

Geometrical non-linear analysis is performed using both linear elastic and elastic-plastic material behaviour where the Von Mises yield criterion is used for yield stress equal to 255N/mm^2. The load control method is used for the analysis with load increment equal to 0.1 and maximum 15 iterations per load step.

Figure 4.27 shows a load-displacement graph of the node corresponding to the maximum out of plane deformation from the linear buckling mode. For the linear elastic material behaviour a load factor equal to 5.77 is obtained. The deformation for several load steps is showed in figure 4.27. The analysis stops shortly after the panel starts to yield for the case of elastic-plastic material behaviour, giving load factor equal to 1.43.

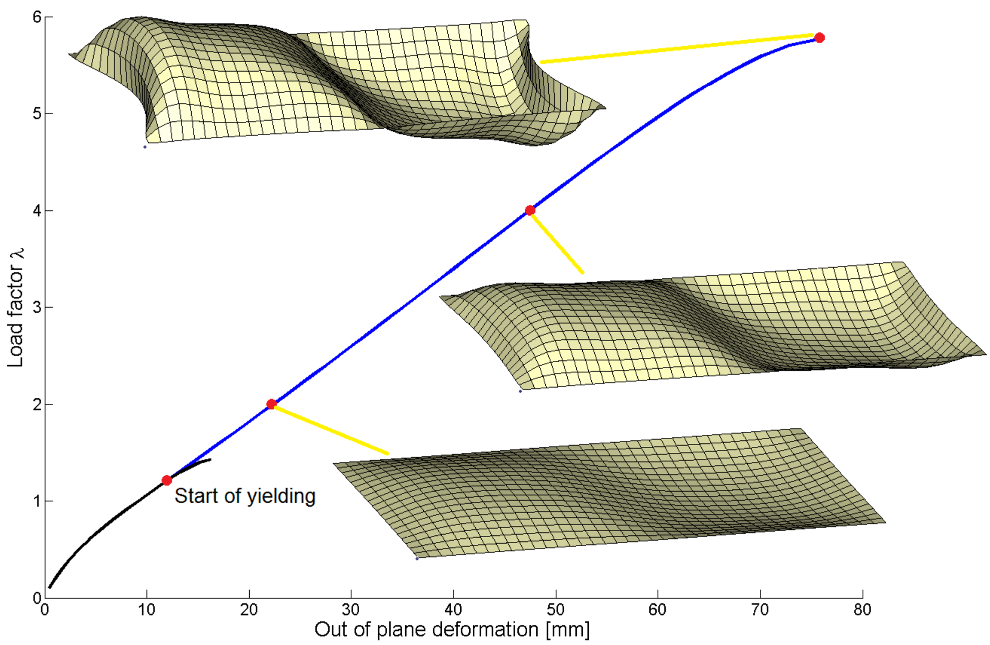


Figure 4.27: Load-displacement graph for non linear analysis for both linear elastic (blue line) and elastic-plastic (black line) material

Stress Check Model - Ultimate Limit State

The Stress Check Model uses the stress results from the linear elastic finite element analysis on the panel to perform the ultimate limit state stress check.

$$\begin{aligned}\left(\frac{\sigma_{x,max}}{\sigma_{Ux}}\right)^2+\left(\frac{\sigma_{y,max}}{\sigma_{Uy}}\right)^2+\left(\frac{\tau}{\tau_U}\right)^2-\varphi\left(\frac{\sigma_{x,max}}{\sigma_{Ux}}\right)\left(\frac{\sigma_{y,max}}{\sigma_{Uy}}\right)\\ =\left(\frac{75.5}{125.4}\right)^2+0+0+0=0.36\end{aligned} \tag{4.8}$$

Equivalent load factor for the unity check is 1,67 which is larger than the load factor from the geometrical non-linear analysis with elastic-plastic material behaviour.

Summary

The result from the Stress Check Model for the buckling limit state match well with the analytical solution and the linear buckling analysis. However when the outcome of the ultimate limit state check is compared to the geometrical non-linear analysis using elastic-plastic material behaviour the Stress Check Model gives a higher load capacity.

4.4.2 Case 2 - Uniform Compression Acting in both x- and y-Direction

Figure 4.28 shows schematic view of the load case.

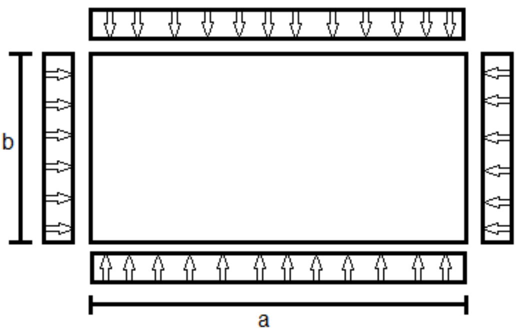


Figure 4.28: Schematic view of load case 2

Analytical solution

The analytical solution for simply supported plate under uniform compression in both x- and y-direction with ratio between the stress components equal to;

$$\psi = \frac{\sigma_y}{\sigma_x} < 0.5 \tag{4.9}$$

is the following with respect to stresses in the x-direction [16]:

$$\sigma_{cr,x} = k_x \frac{\pi^2 E}{12\left(1-\nu^2\right)} \left(\frac{t}{b}\right)^2 = k_x \cdot 18.98\text{N/mm}^2 \tag{4.10}$$

The plate buckling coefficient k_x can be determined as:

$$k_x = \frac{\left(\frac{m^2}{\alpha^2} + n^2\right)^2}{\frac{m^2}{\alpha^2} +^2} \tag{4.11}$$

Where m and n are the number of half waves in longitudinal and transverse directions. For aspect ratio α equal to 2 and stress ratio ψ equal 0.3 (same as the Poisson's ratio of the material) the buckling coefficient has a minimum value for m = n = 1 resulting in k_x equal to 2.84. The resulting critical stress is $\sigma_{cr,x}$ = **56.8N/mm²**.

Linear Buckling Analysis

Figure 4.29 shows the boundary conditions and the load applied to the finite element model. Displacements along the edges in the z-direction are constrained (not showed in the figure). Displacements in y-direction are constrained at the upper and lower edge and displacements in x–direction are constrained in middle of the panel. The load applied to the panel is tuned until the lowest eigenvalue from linear buckling analysis becomes equal to 1.00. The mesh size is 50mmx50mm.

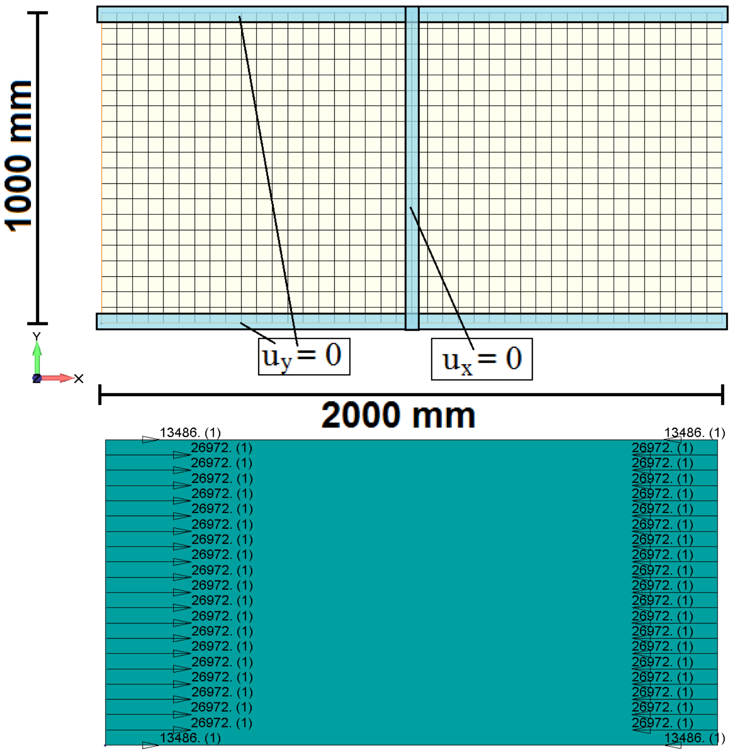


Figure 4.29: Boundary conditions for finite element analysis (above), applied load to finite element model (below)

The panel is free to deform only in the x-direction causing constant stresses in the x- and y-direction with ratio between the stress components equal to 0.3 (same as the Poisson's ratio). The shear stresses are zero as showed in figure 4.30.

Figure 4.31 shows the buckling mode corresponding to the lowest eigenvalue. One half wave in both transverse and longitudinal directions can be seen in the figure which correspond to the analytical solution. The corresponding buckling load is equal to $\mathbf{53.9N/mm^2}$ which is 5% less than the analytical solution.

Stress Check Model - Buckling Limit State

The Stress Check Model uses the stress results from the linear elastic finite element analysis on the panel to perform the buckling limit state stress check. The yield stress is taken as 255N/mm^2.

$$\left(\frac{\sigma_{x,max}}{\sigma_{Cx}}\right)^2 + \left(\frac{\sigma_{y,max}}{\sigma_{Cy}}\right)^2 + \left(\frac{\tau}{\tau_C}\right)^2 = \left(\frac{53.9}{75.9}\right)^2 + \left(\frac{16.2}{29.7}\right)^2 + 0 = 0.80 \tag{4.12}$$

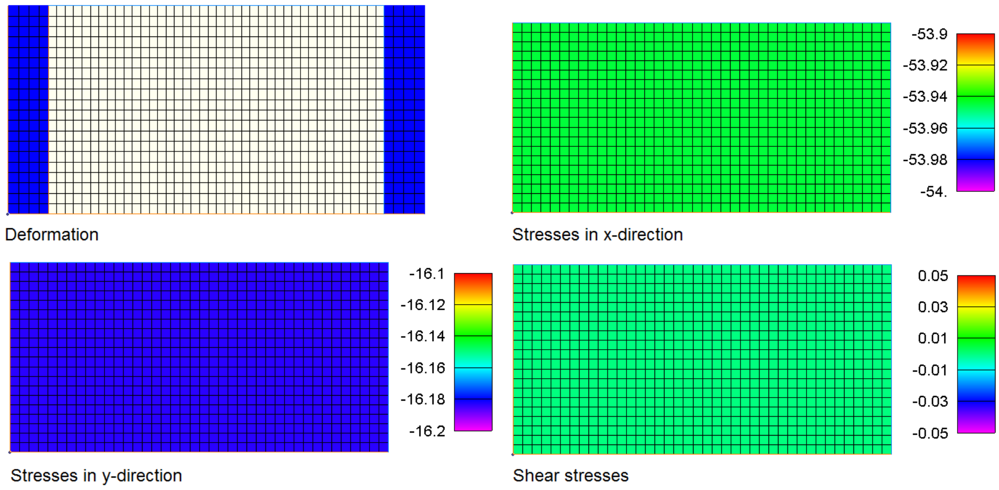


Figure 4.30: Linear elastic deformation of the panel and stress results

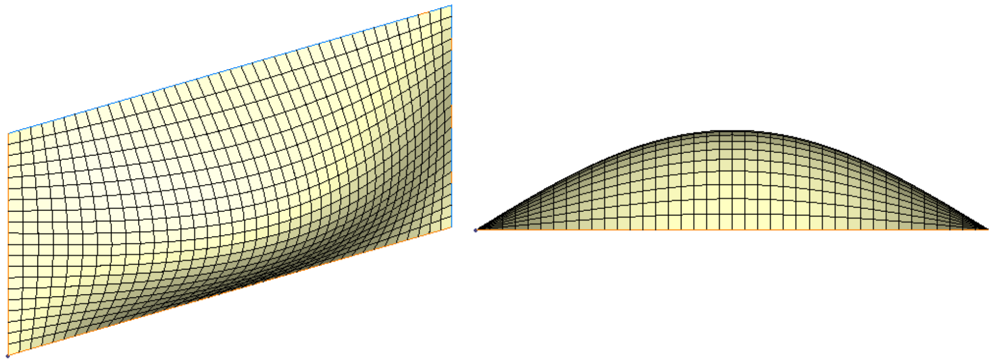


Figure 4.31: Buckling mode corresponding to the lowest eigenvalue

Equivalent load factor for the unity check is 1.12 which means that the applied load of the finite element analysis can be increased by 12% for the Stress Check Model leading to greater value than the analytical solution.

Non-Linear Analysis

Geometrical imperfections are introduced to the model by using the deformation from the linear buckling analysis corresponding to the lowest eigenvalue. The magnitude of the imperfections is equal to 5mm (1/200 of the width of the panel) which is in accordance with recommended value from Eurocode 3, part 1.5 [9].

Geometrical non-linear analysis is performed using both linear elastic and elastic-plastic material behaviour where the Von Mises yield criterion is used for yield stress equal to 255N/mm^2. The load control method is used for the analysis with load increment equal to 0.1 and maximum 15 iterations per load step.

Figure 4.32 shows a load-displacement graph of the node corresponding to the maximum out of plane deformation from the linear buckling mode, located at the middle of the panel.

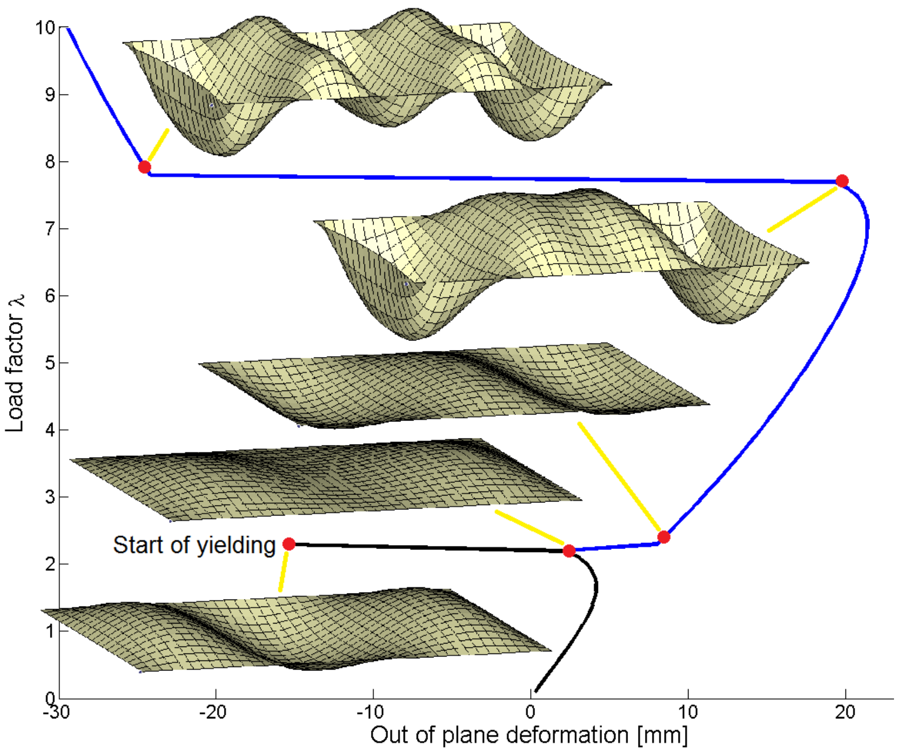


Figure 4.32: Load-displacement graph for non linear analysis for both linear elastic (blue line) and elastic-plastic (black line) material

For the linear elastic material behaviour a load factor equal to 31.2 is obtained. When the load factor is equal to 2.3 the buckling mode changes from one half wave into three half waves and again for load factor equal to 7.8 the buckling mode changes into five half waves mode (see figure 4.32). The graph in figure 4.32 only show results for load factor up to 10.0. If a displacement control or arc-length control method would have been used the snap-back behaviour showed in figure 4.32 could probably have been analysed further.

When the analysis is performed with elastic-plastic material behaviour an opposite buckling shape occurs after load factor equal to 2.3 compared to the linear elastic material behaviour. The analysis stops immediately due to yielding of the panel resulting in load factor equal to 2.2.

Stress Check Model - Ultimate Limit State

The Stress Check Model uses the stress results from the linear elastic finite element analysis on the panel to perform the ultimate limit state stress check.

$$\begin{aligned} &\left(\frac{\sigma_{x,max}}{\sigma_{Ux}}\right)^2 + \left(\frac{\sigma_{y,max}}{\sigma_{Uy}}\right)^2 + \left(\frac{\tau}{\tau_U}\right)^2 - \varphi\left(\frac{\sigma_{x,max}}{\sigma_{Ux}}\right)\left(\frac{\sigma_{y,max}}{\sigma_{Uy}}\right) \\ &= \left(\frac{75.5}{125.4}\right)^2 + \left(\frac{16.2}{77.6}\right)^2 + 0 + 0.74\left(\frac{75.5}{125.4}\right)\left(\frac{16.2}{77.6}\right) = 0.30 \end{aligned} \tag{4.13}$$

Equivalent load factor for the unity check is 1,82 which is lower than the load factor from the geometrical non-linear analysis with elastic-plastic material behaviour.

Summary

The result from the Stress Check Model for the buckling limit state gives higher load capacity compared to the analytical solution and the linear buckling analysis. When the outcome of the ultimate limit state check is compared to the geometrical non-linear analysis using elastic-plastic material behaviour the Stress Check Model gives a lower load capacity.

4.4.3 Case 3 - In-Plane Bending

Figure 4.33 shows schematic view of the load case.

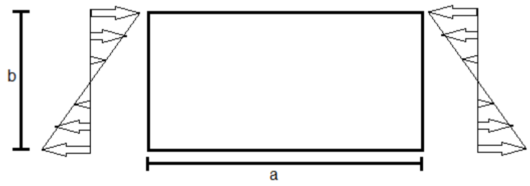


Figure 4.33: Schematic view of load case 1

Analytical solution

The analytical solution for simply supported plate under in-plane bending is the following [15]:

$$\sigma_{cr} = k\frac{\pi^2 E}{12\left(1-\nu^2\right)}\left(\frac{t}{b}\right)^2 = k \cdot 18.98\text{N/mm}^2 \tag{4.14}$$

Where σ_{cr} is the critical load the plate buckling coefficient k can be determined from figure 4.34 [15].

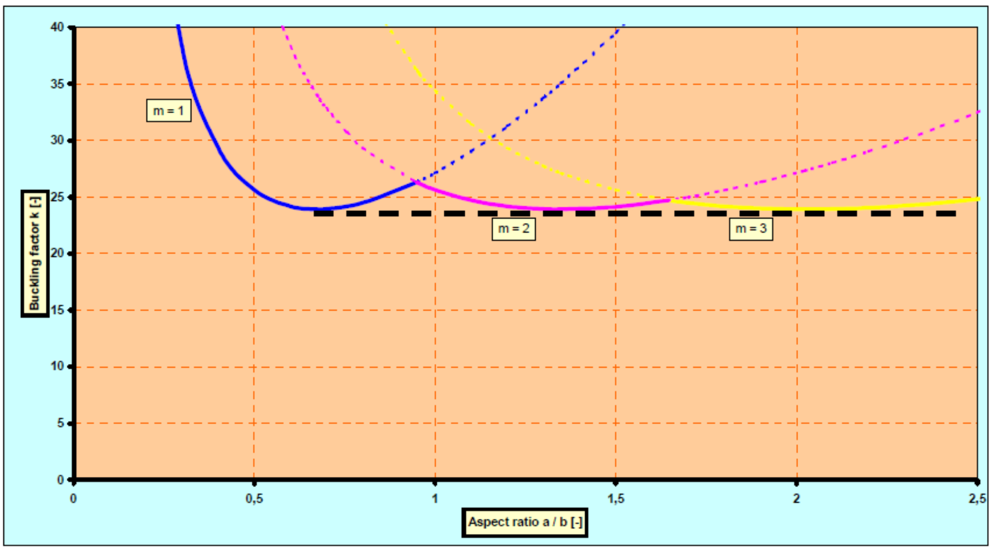


Figure 4.34: Plate buckling coefficient k for a simple supported plate under bending

The aspect ratio of the panel is equal to 2 so the the plate buckling coefficient becomes equal to 23.92 and number of half waves for the buckling mode are equal to 3. The resulting critical stress is σ_{cr} = $\mathbf{454.0N/mm^2}$.

Linear Buckling Analysis

Figure 4.35 shows the boundary conditions and the load applied to the finite element model. Displacements along the edges in the z-direction are constrained (not showed in the figure). Displacements in x-direction are constrained in the middle of the panel and for the node located at the middle of the panel, displacement in y-direction is constrained. The load applied to the panel is tuned until the lowest eigenvalue from linear buckling analysis becomes equal to 1.00. The mesh size is 50mmx50mm.

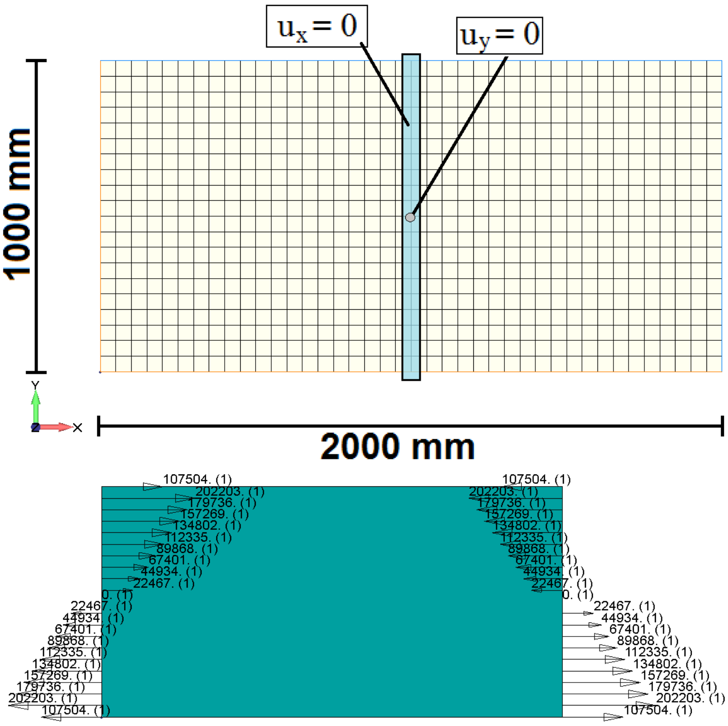


Figure 4.35: Boundary conditions for finite element analysis (above), applied load to finite element model (below)

The panel is free to deform in both x- and y-direction causing relatively small stresses in the y-direction and shear stresses compared to the stresses in the x-direction as showed in figure 4.25.

Figure 4.37 shows the buckling mode corresponding to the lowest eigenvalue. Three half waves can be seen in the figure which correspond to the analytical solution. The corresponding buckling load is equal to **452.4N/mm^2** which is slightly less than the analytical solution.

Stress Check Model - Buckling Limit State

The Stress Check Model uses the stress results from the linear elastic finite element analysis on the panel to perform the buckling limit state stress check.

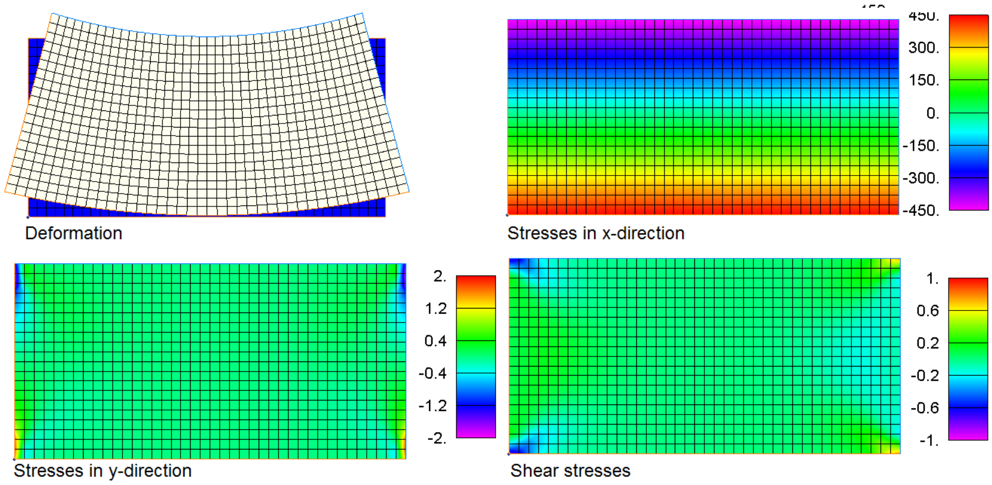


Figure 4.36: Linear elastic deformation of the panel and stress results

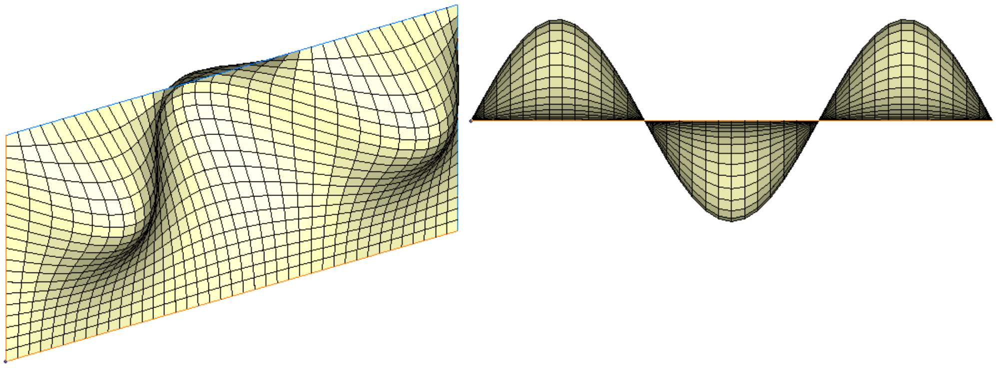


Figure 4.37: Buckling mode corresponding to the lowest eigenvalue

The yield stress is taken as 255N/mm^2.

$$\left(\frac{\sigma_{x,max}}{\sigma_{Cx}}\right)^2 + \left(\frac{\sigma_{y,max}}{\sigma_{Cy}}\right)^2 + \left(\frac{\tau}{\tau_C}\right)^2 = \left(\frac{452.4}{220.0}\right)^2 + 0 + 0 = 4.23 \qquad (4.15)$$

Equivalent load factor for the unity check is 0.49 which is significantly lower than the analytical and linear buckling solution. The reason for the difference is that for the Stress Check Model the resistance of the panel depends on the yield stress (see equation 3.2). Figure 4.38 shows a graph where the outcome of the unity check is plotted against the yield stress. To obtain the same results for the Stress Check Model as for the analytical solution the material would have a yield stress equal to 750N/mm^2 which is a unrealistic solution for general structures.

Non-Linear Analysis

Geometrical imperfections are introduced to the model by using the deformation from the linear buckling analysis corresponding to the lowest eigenvalue. The magnitude of the imperfections is equal to 5mm (1/200 of the width of

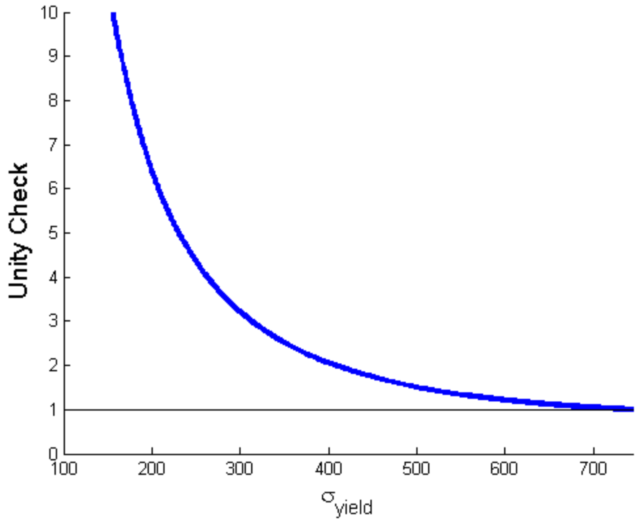


Figure 4.38: Unity check as a function of the yield stress

the panel) which is in accordance with recommended value from Eurocode 3, part 1.5 [9].

Geometrical non-linear analysis is performed using both linear elastic and elastic-plastic material behaviour where the Von Mises yield criterion is used for yield stress equal to 255N/mm^2. The load control method is used for the analysis with load increment equal to 0.05 and maximum 15 iterations per load step.

Figure 4.39 shows a load-displacement graph of the node corresponding to the maximum out of plane deformation from the linear buckling mode. For the linear elastic material behaviour a load factor equal to 4.43 is obtained. The deformation for several load steps is showed in figure 4.27. The analysis stops shortly after the panel starts to yield for the case of elastic-plastic material behaviour, giving load factor equal to 0.59.

Stress Check Model - Ultimate Limit State

The Stress Check Model uses the stress results from the linear elastic finite element analysis on the panel to perform the ultimate limit state stress check.

$$\begin{aligned}\left(\frac{\sigma_{x,max}}{\sigma_{Ux}}\right)^2 + \left(\frac{\sigma_{y,max}}{\sigma_{Uy}}\right)^2 + \left(\frac{\tau}{\tau_U}\right)^2 - \varphi\left(\frac{\sigma_{x,max}}{\sigma_{Ux}}\right)\left(\frac{\sigma_{y,max}}{\sigma_{Uy}}\right) \\ = \left(\frac{452.4}{220.0}\right)^2 + 0 + 0 + 0 = 4.23\end{aligned} \tag{4.16}$$

The ultimate limit state yields the same results as the buckling limit state for this case. Equivalent load factor for the unity check is 0.49 which is lower than the load factor from the geometrical non-linear analysis with elastic-plastic material behaviour.

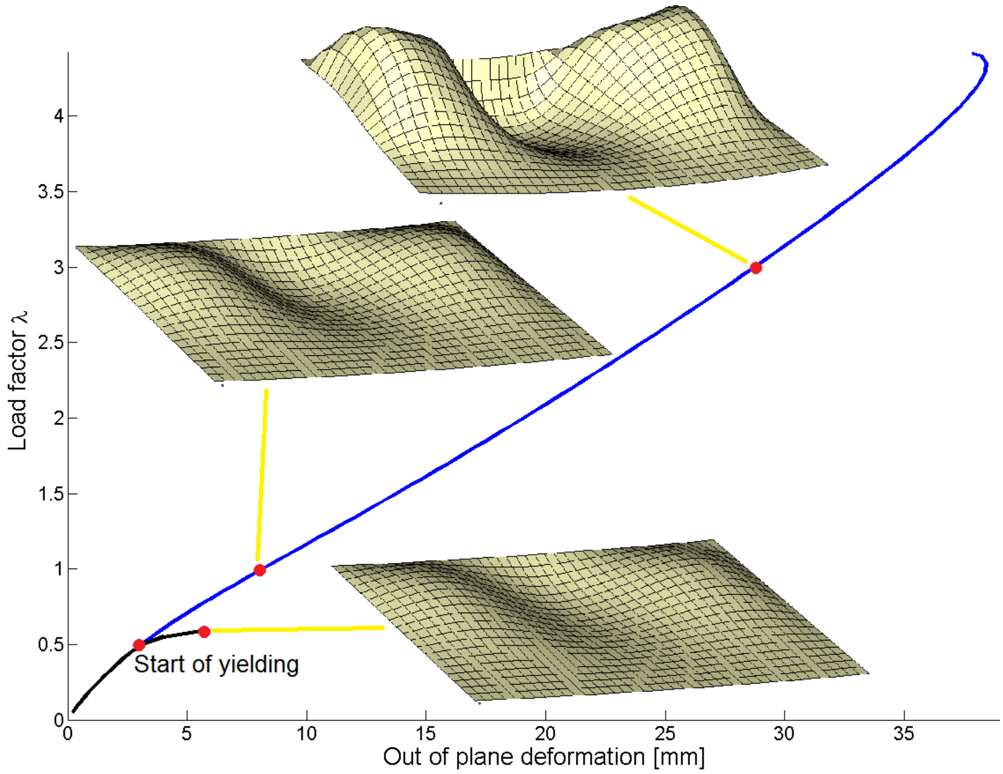


Figure 4.39: Load-displacement graph for non linear analysis for both linear elastic (blue line) and elastic-plastic (black line) material

Summary

The result from the Stress Check Model show significantly lower load factors compared to the analytical solution and the linear buckling analysis. The reason is that in-plane bending is according to the theory a favourable case resulting in a resistance much greater than normal yield stresses. This will cause the panel to yield long before the critical stresses are reached. When the outcome of the ultimate limit state check is compared to the geometrical non-linear analysis using elastic-plastic material behaviour the Stress Check Model gives a lower load capacity.

Chapter 5

Conclusions and Recommendations

5.1 Conclusion

In this thesis a post processing tool for finite element analysis, Stress Check Model, has been developed to perform buckling checks on stiffened panels based on design codes. The procedure is completely automatic apart from user's initial settings related to what type of method should be used to determine stresses acting on panels. It is possible to use the tool to analyse flat rectangular plate fields with stiffeners and girder oriented in the longitudinal and transverse directions. Both geometrical and material properties of panels, stiffeners and girders may vary.

The design load which is applied to panels is based on stress results from finite element analysis located along the edges of the panels. The approach is conservative leading to design load which is equivalent or greater than the finite element results. The user is given the option to verify the results by comparing graphically the finite element stress results to the design load as shown in figure 4.12.

In section 4.1 the Stress Check Model is used to perform buckling checks on examples from the ABS's *Commentary on the guide for buckling and ultimate strength assessment for offshore structure*. Identical results were expected between the Stress Check Model and the examples but results show some variation. In most cases the Stress Check Model yielded more conservative results. The difference in results was mainly traced to the finite element models which provoked stresses in the Stress Check Model which were not included in the examples.

The example from section 4.2 demonstrates the functionality of the Stress Check Model on a small finite element model. The results of the buckling checks are compared to each panel and the stiffeners which are modelled with beam elements. A panel which failed the buckling state limit check is investigated further to give clearer vision on whether the applied load is too conservative.

The Stress Check Model is compared to the SDC Verifier in section 4.3. The following conclusions can be drawn from the comparison.

- The Stress Check Model gives a representative value for the whole panel instead of every finite element as the buckling tool of the SDC Verifier does.
- The Stress Check Model does not require as fine finite element mesh as the buckling tool of the SDC Verifier to obtain safe design load.
- The Stress Check Model takes into account the in-plane bending influences of the loading leading to higher buckling resistance and consequently lower buckling factors.

In section 4.4 the Stress Check Model is compared to three different plate buckling examples where an analytical solution exists. Results are also compared to linear buckling analysis and geometrical non-linear analysis using non-linear material behavior.

Major conclusion from the comparison indicates that the analytical solution and the linear buckling analysis leads to great overestimation of the buckling load for in-plane bending loading. The Stress Check Model includes the influences of the yield stress leading to more accurate results for this type of cases. Furthermore the Stress Check Model saves engineering time on post analysing the results for this type of cases where the theoretical buckling load significantly exceeds the yield stress of the material.

5.2 Improvements for the Stress Check Model

The practical usefulness of the Stress Check Model can be increased significantly by implementing into the program a more extensive algorithm to detect geometrical properties, reduce the user's settings options and add alternative stress checks according to different design codes.

5.2.1 Determining Settings for the Stress Check Model

A parametric research could be done on the user's settings for the Stress Check Model to determine appropriate settings which yield acceptable level of conservatism. The questions which need to be answered are:

- Should stress values in the cross section of a plate element be taken as the maximum compressive value at the outer fibres or as the average value?

- In order to obtain symmetric design stress distribution at the edges of panels should the method describes in Clause 4.6(3) in Eurocode 3 part 1.5 be used or maximum stress values at edges be used?
- Should the design shear stresses be based on average shear stress results acting on the edges of the panels or use the maximum value? Alternative approach could be using the root of the sum of the squares of the shear stress results as a design value.
- Should the design lateral pressure be based on average or maximum pressure acting on the panel?

For practise it is recommended to perform first analysis using the most conservative settings. If all the buckling checks pass the analysis the the results can be considered being verified. If a buckling checks fails using the conservative setting the analysis can be run again using less conservative setting.

If the buckling checks fails again the user should compare the design load with the finite element stress results using the plot option of the Stress Check Model. If the design load match the finite element stress results the design of the structure must be improved. If the design load does not match the finite element stress results more accurate analysis should be performed, e.g. using geometrical non-linear analysis.

If the buckling checks which failed the analysis using more conservative settings pass the analysis using less conservative settings a more accurate analysis is required.

5.2.2 Detecting Geometry

Element Types

Section 2.1.2 explained that the Stress Check Model recognises four noded flat shell elements and two noded beam elements with either angle shaped or T-shaped cross sections. The database of finite element types which the Stress Check Model can detect should be increased in order to read more complicated finite element models. The starting point could be implementing three noded plate elements and beam elements with closed cross sections.

Stiffeners Modelled with Plate Elements

The buckling checks on stiffeners and girders can be improved if the cross sectional properties of stiffeners and girders which are modelled with plate elements can be detected. In section 2.2.1 a method is described to detect the orientation of stiffeners and girders which are modelled with plate elements. The same algorithm which is used to detect panel sizes can be used to detect the height and thickness of the webs since the plate elements of the stiffeners and girders which are attached to the reference plane of the panels are known. Only the reference plane of the webs has to determined. To detect the properties of the flanges free edges have to be detected.

Inclined Stiffeners and Girders

The Stress Check Model can only recognise rectangular plate fields. However it often happens in practise that some panels have non rectangular form. Those panels can be analysed using the greater dimensions as the length and width as showed in figure 5.1. Therefore the algorithm needs to be extended to detect panels which have inclined stiffeners and girders.

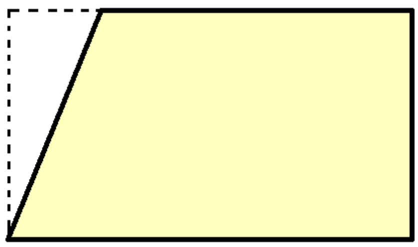


Figure 5.1: Non rectangular panel

Reference Plane

The reference plane in the Stress Check Model is by default set to z equal to zero, meaning that only plate elements in the x-y plane with the z-coordinate equal to zero will be analysed. It would be optimal to have an algorithm which would detect the reference plane automatically, allowing any kind of orientation of the finite element model. Furthermore it would be advantageous to have an algorithm which could analyse more than one reference planes at once.

Free Edges

In order to detect flange properties of stiffeners or perform local buckling check on flanges modelled with plate elements, free edges (unsupported edges) need to be detected. During the development of the Stress Check Model an algorithm was written to detect plate fields with simple geometry including free edges. However the algorithm would not run for plate fields including openings or different panel sizes. It would be possible to combine the old algorithm with the current algorithm with some modification to solve the problem.

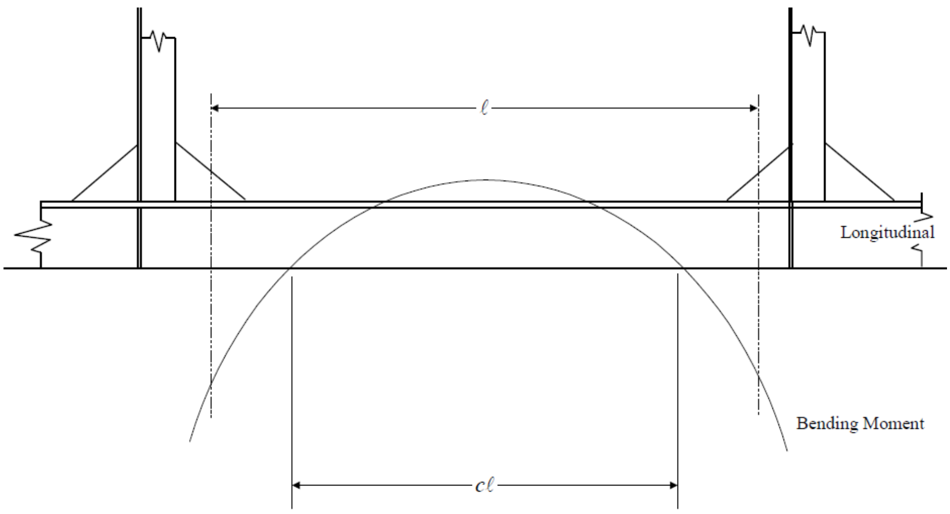
5.2.3 Alternative Buckling Checks

For various reasons the user of the Stress Check Model should have an option to perform buckling checks according to different design codes, for example according to Eurocode 3 or the DNV guide. Furthermore the ABS package is not completed. Additional checks to the ones described in Chapter 3 have to be carried out to perform sufficient buckling check on stiffened panels according to the ABS guide. These checks are:

- Stiffness of stiffeners to avoid local buckling of stiffeners prior to buckling of the plates
- Proportions of stiffeners flange, web and face plate to avoid local buckling of the stiffeners
- Buckling of tripping brackets
- Buckling checks on girders

5.2.4 Determining the Effective Breadth for the ABS Package

Figure 5.2 shows in a schematic way how the effective breadth from section 3.2.2 can be determined.



$c\ell/s$	1.5	2	2.5	3	3.5	4	4.5 and greater
s_w/s	0.58	0.73	0.83	0.90	0.95	0.98	1.0

Figure 5.2: Effective breadth of plating s_w

The effective breadth depends on the distance between zero bending moments. Equation 3.34 determines the maximum moment assuming uniformly distributed lateral load and clamped boundary conditions. The moment distribution for such case can be expressed as:

$$M = \frac{qs\left(6lx - l^2 - 6x^2\right)}{12} \tag{5.1}$$

The locations of the zero bending moments can be determined by solving the quadratic equation in the brackets.

$$6lx - l^2 - 6x^2 = 0 \Rightarrow x = [0.211L, 0.789L] \tag{5.2}$$

The distance between the zero bending moments is therefore equal to 0.578 times the length of the stiffener. By replacing this value for the parameter c in the table in figure 5.2 makes it possible to determine for which aspect ratios l/s the effective breadth can be increased. The result is that for aspect ratios greater than 2.6 the effective breadth can be increased.

Therefore it could improve the ABS package of the Stress Check Model to use interpolation according to the table in figure 5.2 to determine the effective breadth instead of using the current conservative value of 0.58 times the width of the panel.

5.3 Recommendation for the SDC Verifier

Results in section 4.3 indicate that for coarse finite element mesh the buckling tool of the SDC Verifier might lead to unsafe results. It is therefore recommended to use maximum stress results within each finite element instead of of average stress results when performing buckling checks using the SDC Verifier.

Bibliography

[1] RUAG Holding. Aerostructures in Oberpfaffenhofen, Germany. http://www.ruag.com/Group/Media/Photos (accessed September 6, 2011).

[2] gcaptain.com. Cross-Section of Drillship Hull. http://www.flickr.com/-photos/gcaptain/991744606/sizes/l/in/photostream/ (accessed September 6, 2011).

[3] Tom Paiva. Orthotropic Box Girder-Crossbeam Connection. http://www.mtc.ca.gov/images/paiva/8.jpg (accessed September 6, 2011).

[4] Marhi. http://www.marhi.ru/AMIT/2009/4kvart09/Lunina/Article.php (accessed September 6, 2011).

[5] E. Byklum, E. Steen, J. Amdahl. A semi-analytical model for global buckling and postbuckling analysis of stiffened panels. Thin-Walled Structures 42 (2004) 701-717;

[6] American Bureau of Shipping. Buckling and ultimate strength assessment for offshore structure. Houston 2004.

[7] Autodesk, Inc. Critical Buckling Load. http://download.autodesk.com/-us/algor/userguides/mergedProjects/Analysis_Types/Linear/-Linear_Critical_Buckling_Load.htm (accessed July 13, 2011).

[8] Prof. ir. A.C.W.M. Vrouwenvelder. Structural stability. Technical University Delft, Faculty of Civil Engineering and Geosciences. 2003.

[9] Eurocode 3 - Design of steel structures - Part 1-5: Plated structural elements. 2006.

[10] Det Norske Veritas. Recommended practise, DNV-RP-C201, Buckling strength of plated structures. 2008.

[11] Det Norske Veritas. Platework - plate analysis postprocessor. http://www.dnv.com/services/software/products/sesam/-sesamfloatingstructures/platework.asp (accessed April 26, 2011).

[12] Prof. Dr. Amany Mousa. Statisitcal Data Analysis. Center for Advancement of Postgraduate Studies and Research in Engineering Science. Faculty of Engineering - Cairo University. Cairo 2005.

[13] American Bureau of Shipping. Commentary on the guide for buckling and ultimate strength assessment for offshore structure. Houston 2005.

[14] Femto Engineering. SDC Verifier, User's Guide, version 1.2.

[15] Ir. R. Abspoel and Prof. ir. F.S.K. Bijlaard. Steel Structures 3, Part: Stability of Steel Plate Structures.Technical University Delft, Faculty of Civil Engineering and Geosciences. 2005.

[16] Frederick Bloom and Douglas Coffin. Handbook of Thin Plate Buckling and Postbuckling. Chapman and Hall/CRC. 2001

Appendix A

Estimation of the Accuracy of the Design Load

The SDC Verifier determines the design load for buckling checks based on average stress results within each finite element. This approach is righteous for uniformly distributed load but for varying stress distribution the peak stresses are reduced leading to less conservative results. Below an expression is derived to estimate the required number of finite elements ($\mathrm{N_{el}}$) to establish acceptable value for the design load. The expression is valid for panels loaded with linear distributed in-plane stresses in one direction and depends on the deviation (ϵ) of the peak stress and the ratio (ψ) between the maximum and minimum stresses applied to the panel.

Figure A.1 shows a strip of a panel loaded with linear distributed stress in one direction with the greater stress value equal to σ_1 and the lower stress value equal to σ_2. The length of the strip is equal to L and the length of each element is equal to $L/\mathrm{N_{el}}$. The purpose is to estimate the accuracy of the design stress σ_d acting on element number $\mathrm{N_{el}}$.

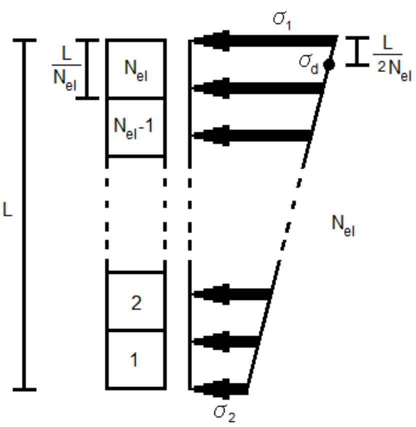


Figure A.1: Linear stress distribution acting on panel

Simple trigonometry leads to the following relation:

$$\frac{\sigma_1 - \sigma_2}{L} = \frac{\sigma_1 - \sigma_d}{\frac{L}{2N_{el}}} \tag{A.1}$$

Introducing the ratio between the maximum and minimum stresses and the deviation of the design stress from the peak stress:

$$\psi = \frac{\sigma_2}{\sigma_1} \rightarrow \sigma_2 = \psi \sigma_1 \tag{A.2}$$

$$\sigma_d = \sigma_1 \left(1 - \epsilon\right) \tag{A.3}$$

Inserting equations A.2 and A.3 into equation A.1 gives:

$$\frac{\sigma_1 - \psi \sigma_1}{L} = \frac{2N_{el} \left(\sigma_1 - \sigma_1 \left(1 - \epsilon\right)\right)}{L} \tag{A.4}$$

Eliminating L and σ_1 from equation A.4 and solve for the number of elements leads to:

$$N_{el} = \frac{1 - \psi}{2\epsilon} \tag{A.5}$$

For uniformly distributed stress pattern ($\psi = 1$) equation A.5 becomes equal to zero. Therefore the equation is adjusted to minimum number of elements equal to one.

$$N_{el} = \frac{1 - \psi}{2\epsilon} + 1 \tag{A.6}$$

Figure A.2 shows the number of required finite elements for 10%, 5% and 2% deviation of the design stress from the peak stress for ratio between the maximum and minimum stresses varying from -1 to 1.

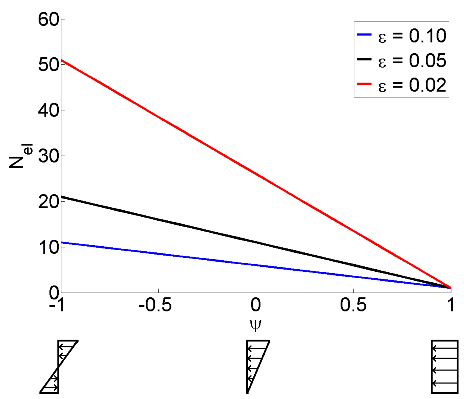


Figure A.2: Required number of finite elements

Equation A.6 can be rearranged to determine the deviation of the design stress from the peak stress for known number of elements and stress pattern.

$$\epsilon = \frac{1 - \psi}{2\left(N_{el} - 1\right)} \tag{A.7}$$

Appendix B

Report For Example

```
-------------------------------------------------------------------
It is not possible to perform beam-column buckling check on the
following stiffeners because they are either modelled with plate
elements or the stiffeners are only attached to one panel, buckling
check will not be proceded on these stiffeners
-------------------------------------------------------------------
Stiffener
  Nr.      Reason
       1   Modelled with plate elements
       2   Modelled with plate elements
       3   Modelled with plate elements
       4   Modelled with plate elements
       5   Modelled with plate elements
       6   Modelled with plate elements
       7   Modelled with plate elements
       8   Modelled with plate elements
       9   Modelled with plate elements
      10   Modelled with plate elements
      11   Modelled with plate elements
      12   Modelled with plate elements
      13   Modelled with plate elements
      14   Modelled with plate elements
      15   Modelled with plate elements
      16   Modelled with plate elements
      25   Modelled with plate elements
      26   Modelled with plate elements
      27   Modelled with plate elements
      28   Modelled with plate elements
      29   Modelled with plate elements
      30   Modelled with plate elements
      31   Modelled with plate elements
      32   Modelled with plate elements
      33   Modelled with plate elements
      34   Modelled with plate elements
      35   Modelled with plate elements
      36   Modelled with plate elements
      37   Modelled with plate elements
      38   Modelled with plate elements
      39   Modelled with plate elements
      40   Modelled with plate elements

----------------------------------------------------------
The following panels failed the Buckling state limit check
----------------------------------------------------------
    10
    11
    13
    14
    15
    16
```

ABS Plate buckling stress check

Panel Nr.	Critical check	Ultimate check	Lateral check
1	0.15	0.14	0.07
2	0.13	0.13	0.07
3	0.13	0.12	0.07
4	0.12	0.12	0.07
5	0.12	0.12	0.07
6	0.13	0.12	0.07
7	0.13	0.13	0.07
8	0.15	0.14	0.07
9	0.82	0.72	0.15
10	1.11	0.85	0.15
11	1.11	0.85	0.15
12	0.82	0.72	0.15
13	1.08	0.82	0.15
14	1.19	0.85	0.15
15	1.19	0.85	0.15
16	1.08	0.82	0.15
17	0.31	0.29	0.07
18	0.30	0.29	0.07
19	0.29	0.28	0.07
20	0.29	0.28	0.07
21	0.29	0.28	0.07
22	0.29	0.28	0.07
23	0.30	0.29	0.07
24	0.30	0.29	0.07
25	0.68	0.54	0.14
26	0.81	0.59	0.14
27	0.81	0.59	0.14
28	0.68	0.54	0.14

ABS Stiffened panels buckling stress check

Stiffener Nr.	Beam/ Column check	Flexural/ Torsional check
17	0.41	0.32
18	0.74	0.33
19	0.67	0.33
20	0.41	0.32
21	0.65	0.32
22	0.74	0.33
23	0.70	0.33
24	0.67	0.32

Geometry and material properties of panels

Panel Nr.	Length	Width	Aspect ratio	Thickness	Youngs modulus	Poissons ratio	Yield Stress
1	1000	800	1.25	15.0	210000	0.3	255.0
2	1000	800	1.25	15.0	210000	0.3	255.0
3	1000	800	1.25	15.0	210000	0.3	255.0
4	1000	800	1.25	15.0	210000	0.3	255.0
5	1000	800	1.25	15.0	210000	0.3	255.0
6	1000	800	1.25	15.0	210000	0.3	255.0
7	1000	800	1.25	15.0	210000	0.3	255.0
8	1000	800	1.25	15.0	210000	0.3	255.0
9	2000	1000	2.00	15.0	210000	0.3	255.0
10	2000	1000	2.00	15.0	210000	0.3	255.0
11	2000	1000	2.00	15.0	210000	0.3	255.0
12	2000	1000	2.00	15.0	210000	0.3	255.0
13	2000	1000	2.00	15.0	210000	0.3	255.0
14	2000	1000	2.00	15.0	210000	0.3	255.0
15	2000	1000	2.00	15.0	210000	0.3	255.0
16	2000	1000	2.00	15.0	210000	0.3	255.0

17	1000	800	1.25	15.0	210000	0.3	255.0
18	1000	800	1.25	15.0	210000	0.3	255.0
19	1000	800	1.25	15.0	210000	0.3	255.0
20	1000	800	1.25	15.0	210000	0.3	255.0
21	1000	800	1.25	15.0	210000	0.3	255.0
22	1000	800	1.25	15.0	210000	0.3	255.0
23	1000	800	1.25	15.0	210000	0.3	255.0
24	1000	800	1.25	15.0	210000	0.3	255.0
25	2000	1000	2.00	15.0	210000	0.3	255.0
26	2000	1000	2.00	15.0	210000	0.3	255.0
27	2000	1000	2.00	15.0	210000	0.3	255.0
28	2000	1000	2.00	15.0	210000	0.3	255.0

Geometry and material properties of stiffeners

Stiffener Nr.	Length	Web height	Web thickness	Flange width	Flange thickness	Youngs modulus	Poissons ratio	Yield Stress
17	2000	192	8.0	120	8.0	210000	0.3	255.0
18	2000	192	8.0	120	8.0	210000	0.3	255.0
19	2000	192	8.0	120	8.0	210000	0.3	255.0
20	2000	192	8.0	120	8.0	210000	0.3	255.0
21	2000	192	8.0	120	8.0	210000	0.3	255.0
22	2000	192	8.0	120	8.0	210000	0.3	255.0
23	2000	192	8.0	120	8.0	210000	0.3	255.0
24	2000	192	8.0	120	8.0	210000	0.3	255.0

Modelled stress distribution

Panel Nr.	Maximum x-stress	Minimum x-stress	Ratio x-stress	Maximum y-stress	Minimum y-stress	Ratio y-stress	Shear stress	Lateral pressure
1	47.6	42.4	0.89	45.7	41.8	0.92	-24.6	0.0400
2	49.5	43.6	0.88	46.1	35.9	0.78	-16.6	0.0400
3	50.1	43.2	0.86	46.4	36.3	0.78	-12.8	0.0400
4	50.7	44.0	0.87	47.4	31.4	0.66	-0.9	0.0400
5	50.9	43.9	0.86	47.5	31.4	0.66	1.0	0.0400
6	49.9	43.2	0.86	46.4	36.3	0.78	12.9	0.0400
7	49.7	43.5	0.87	46.1	35.9	0.78	16.7	0.0400
8	47.6	42.4	0.89	45.7	41.9	0.92	24.6	0.0400
9	65.0	56.9	0.88	74.6	58.4	0.78	-21.2	0.0400
10	67.0	59.7	0.89	84.2	74.8	0.89	-7.4	0.0400
11	67.0	59.7	0.89	84.3	74.9	0.89	7.4	0.0400
12	65.1	57.0	0.88	74.7	58.5	0.78	21.2	0.0400
13	77.2	62.4	0.81	77.5	72.8	0.94	-23.4	0.0400
14	81.6	65.2	0.80	79.9	79.0	0.99	-8.0	0.0400
15	81.5	65.2	0.80	79.9	79.0	0.99	7.9	0.0400
16	77.1	62.5	0.81	77.4	72.8	0.94	23.3	0.0400
17	77.0	69.7	0.91	69.4	63.5	0.91	-21.6	0.0400
18	80.6	73.4	0.91	68.0	55.4	0.81	-15.9	0.0400
19	81.6	74.7	0.92	66.8	55.9	0.84	-6.8	0.0400
20	83.8	76.2	0.91	68.0	47.0	0.69	-5.5	0.0400
21	83.7	76.4	0.91	68.1	47.0	0.69	5.4	0.0400
22	81.5	74.5	0.91	66.7	56.0	0.84	6.7	0.0400
23	80.5	73.6	0.91	68.0	55.4	0.81	15.9	0.0400
24	77.0	69.7	0.91	69.4	63.5	0.92	21.6	0.0400
25	55.3	54.3	0.98	64.7	56.5	0.87	-17.9	0.0400
26	56.9	56.0	0.98	69.1	65.8	0.95	-6.2	0.0400
27	57.0	55.9	0.98	69.2	65.7	0.95	6.3	0.0400
28	55.3	54.3	0.98	64.7	56.5	0.87	18.0	0.0400

Applied stresses on panels

Panel Nr.	Maximum x-stress	Maximum y-stress	Shear stress	Lateral load	Von Mises stress	Bending x-stress	Bending y-stress
1	47.6	45.7	-24.6	0.0	63.2	0.89	0.92
2	49.5	46.1	-16.6	0.0	55.9	0.88	0.78
3	50.1	46.4	-12.8	0.0	53.2	0.86	0.78
4	50.7	47.4	-0.9	0.0	49.2	0.87	0.66
5	50.9	47.5	1.0	0.0	49.3	0.86	0.66
6	49.9	46.4	12.9	0.0	53.2	0.86	0.78

7	49.7	46.1	16.7	0.0	56.1	0.87	0.78
8	47.6	45.7	24.6	0.0	63.2	0.89	0.92
9	65.0	74.6	-21.2	0.0	79.3	0.88	0.78
10	67.0	84.2	-7.4	0.0	78.1	0.89	0.89
11	67.0	84.3	7.4	0.0	78.2	0.89	0.89
12	65.1	74.7	21.2	0.0	79.4	0.88	0.78
13	77.2	77.5	-23.4	0.0	87.3	0.81	0.94
14	81.6	79.9	-8.0	0.0	81.9	0.80	0.99
15	81.5	79.9	7.9	0.0	81.9	0.80	0.99
16	77.1	77.4	23.3	0.0	87.2	0.81	0.94
17	77.0	69.4	-21.6	0.0	82.5	0.91	0.91
18	80.6	68.0	-15.9	0.0	80.0	0.91	0.81
19	81.6	66.8	-6.8	0.0	76.2	0.92	0.84
20	83.8	68.0	-5.5	0.0	77.7	0.91	0.69
21	83.7	68.1	5.4	0.0	77.7	0.91	0.69
22	81.5	66.7	6.7	0.0	76.1	0.91	0.84
23	80.5	68.0	15.9	0.0	80.0	0.91	0.81
24	77.0	69.4	21.6	0.0	82.4	0.91	0.92
25	55.3	64.7	-17.9	0.0	68.1	0.98	0.87
26	56.9	69.1	-6.2	0.0	64.8	0.98	0.95
27	57.0	69.2	6.3	0.0	64.9	0.98	0.95
28	55.3	64.7	18.0	0.0	68.1	0.98	0.87

Stress on stiffeners

Stiffener Nr.	Modelled x-Stress	Modelled y-stress	Modelled shear	Modelled pressure
17	65.0	74.6	21.2	0.040
18	67.0	84.2	7.4	0.040
19	67.0	74.9	7.4	0.040
20	65.1	64.7	21.2	0.040
21	65.0	74.6	23.4	0.040
22	67.0	84.2	8.0	0.040
23	67.0	79.9	7.9	0.040
24	65.1	77.4	23.3	0.040

Resistance of panels panels 1/2

Panel Nr.	Euler stress	Slenderness ratio	Coefficient of interaction
1	66.7	1.86	0.07
2	66.7	1.86	0.07
3	66.7	1.86	0.07
4	66.7	1.86	0.07
5	66.7	1.86	0.07
6	66.7	1.86	0.07
7	66.7	1.86	0.07
8	66.7	1.86	0.07
9	42.7	2.32	-0.16
10	42.7	2.32	-0.16
11	42.7	2.32	-0.16
12	42.7	2.32	-0.16
13	42.7	2.32	-0.16
14	42.7	2.32	-0.16
15	42.7	2.32	-0.16
16	42.7	2.32	-0.16
17	66.7	1.86	0.07
18	66.7	1.86	0.07
19	66.7	1.86	0.07
20	66.7	1.86	0.07
21	66.7	1.86	0.07
22	66.7	1.86	0.07
23	66.7	1.86	0.07
24	66.7	1.86	0.07
25	42.7	2.32	-0.16
26	42.7	2.32	-0.16
27	42.7	2.32	-0.16
28	42.7	2.32	-0.16

Resistance of panels panels 2/2

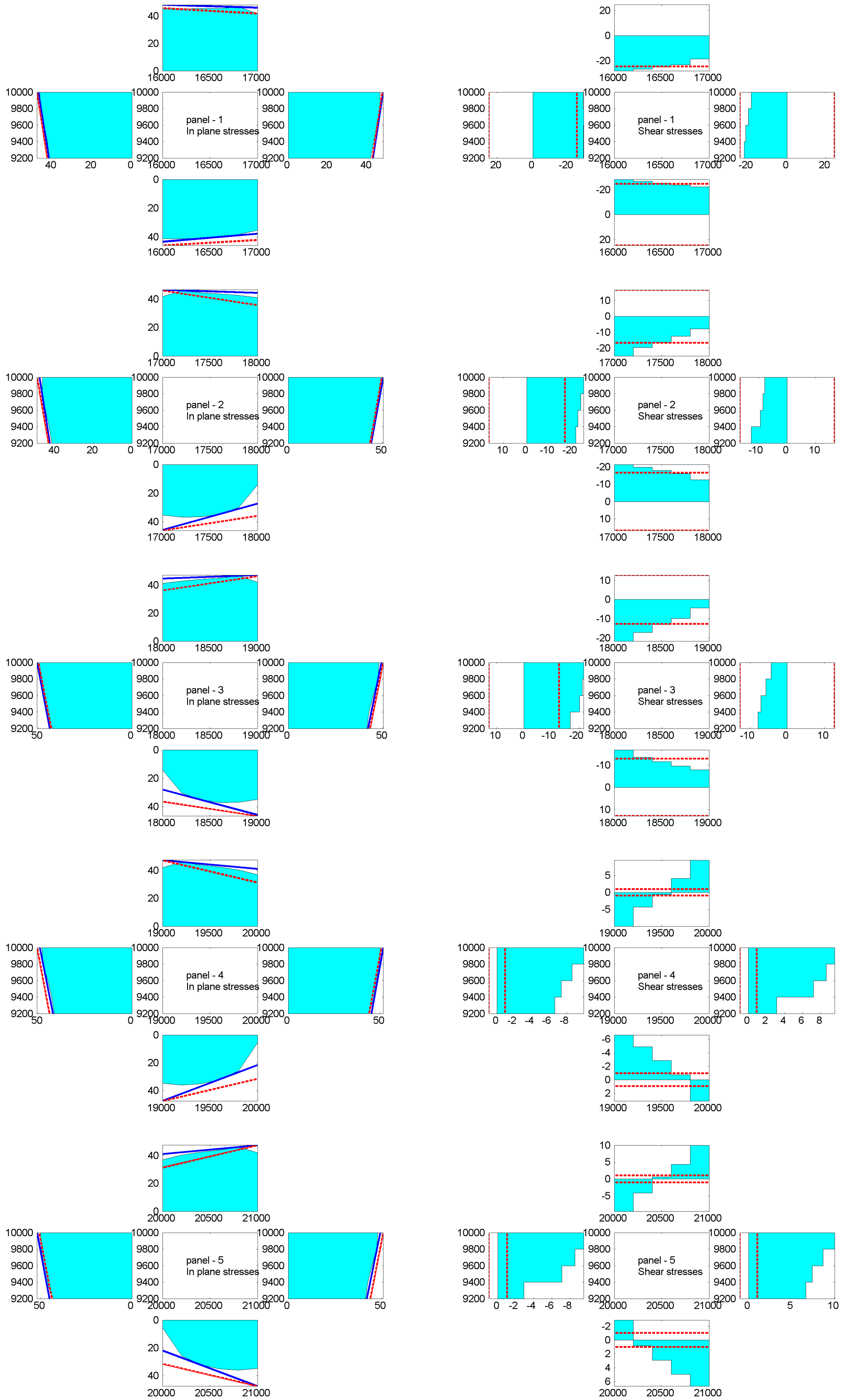
Panel Nr.	Critical x-stress	Ultimate x-stress	Critical y-stress	Ultimate y-stress	Critical shear stress	Ultimate shear stress
1	204.6	204.6	186.4	186.4	138.25	142.23
2	204.9	204.9	192.0	192.0	138.25	142.23
3	205.4	205.4	191.8	191.8	138.25	142.23
4	205.2	205.2	196.0	196.0	138.25	142.23
5	205.3	205.3	196.0	196.0	138.25	142.23
6	205.3	205.3	191.8	191.8	138.25	142.23
7	205.0	205.0	191.9	191.9	138.25	142.23
8	204.6	204.6	186.4	186.4	138.25	142.23
9	176.9	176.9	91.8	104.1	129.76	135.47
10	176.3	176.3	86.1	104.1	129.76	135.47
11	176.3	176.3	86.1	104.1	129.76	135.47
12	176.9	176.9	91.8	104.1	129.76	135.47
13	179.5	179.5	83.3	104.1	129.76	135.47
14	179.9	179.9	80.6	104.1	129.76	135.47
15	179.8	179.8	80.7	104.1	129.76	135.47
16	179.4	179.4	83.3	104.1	129.76	135.47
17	204.2	204.2	186.5	186.5	138.25	142.23
18	204.1	204.1	190.6	190.6	138.25	142.23
19	204.0	204.0	189.7	189.7	138.25	142.23
20	204.1	204.1	195.1	195.1	138.25	142.23
21	204.1	204.1	195.1	195.1	138.25	142.23
22	204.0	204.0	189.7	189.7	138.25	142.23
23	204.0	204.0	190.6	190.6	138.25	142.23
24	204.2	204.2	186.5	186.5	138.25	142.23
25	172.7	172.7	86.9	104.1	129.76	135.47
26	172.6	172.6	82.7	104.1	129.76	135.47
27	172.7	172.7	82.8	104.1	129.76	135.47
28	172.6	172.6	86.9	104.1	129.76	135.47

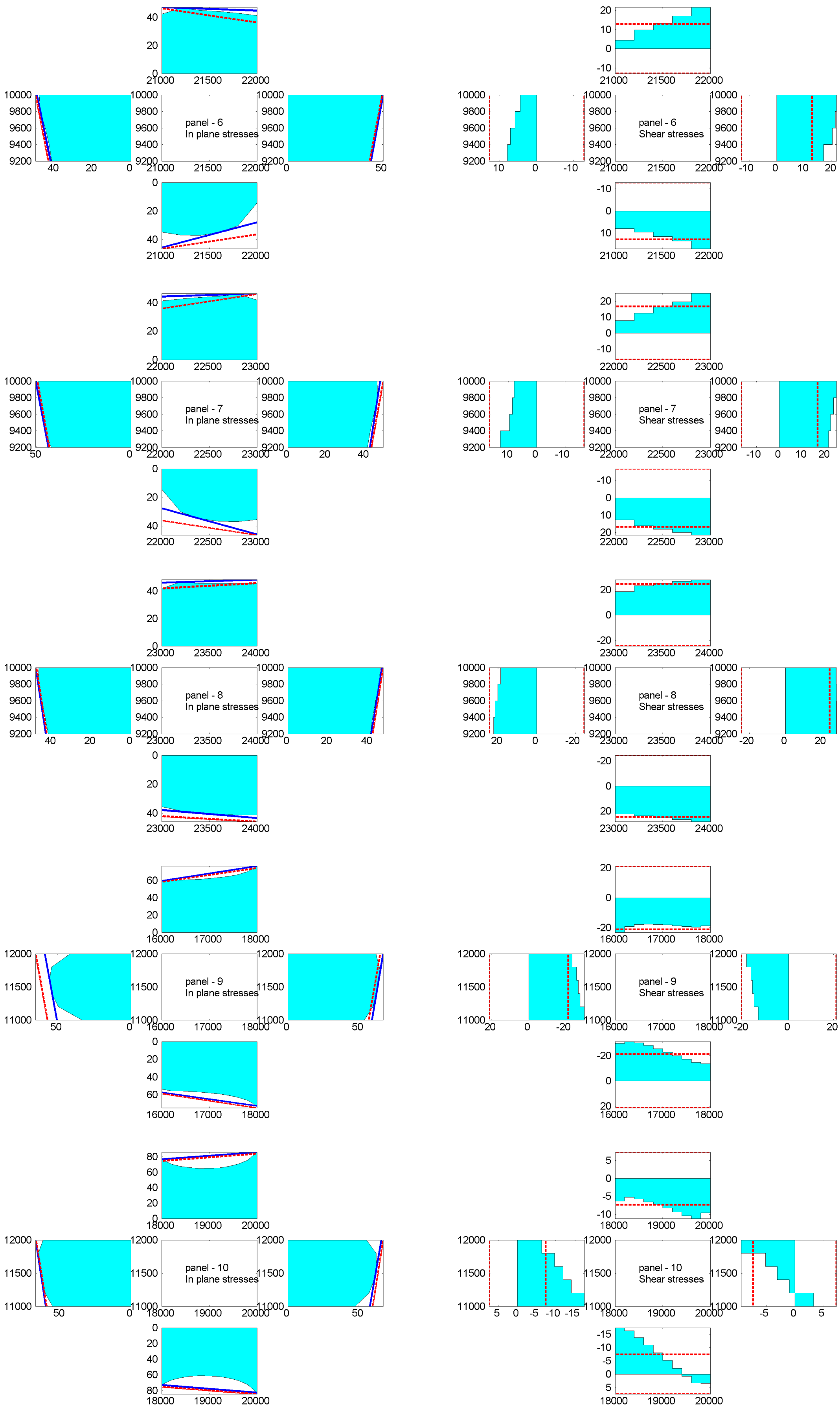
Resistance of stiffeners with identical associated panels

Stiffener Nr.	Effective yield stress	Euler Buckling stress	Critical Buckling stress	Bending stress	Elastic torsional stress	Critical torsional stress
17	255.0	1600.4	245.2	46.9	308.9	204.5
18	255.0	2982.0	249.8	47.1	308.9	204.5
19	255.0	2719.7	249.3	47.0	308.9	204.5
20	255.0	1600.4	245.2	46.9	308.9	204.5
21	255.0	2729.5	249.3	47.0	308.9	204.5
22	255.0	2982.3	249.8	47.1	308.9	204.5
23	255.0	2848.1	249.5	47.1	308.9	204.5
24	255.0	2797.5	249.4	47.1	308.9	204.5

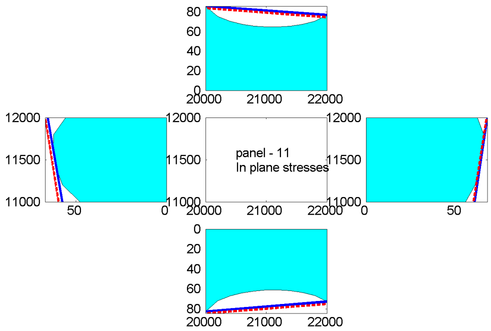
Stiffeners with identical associated panels

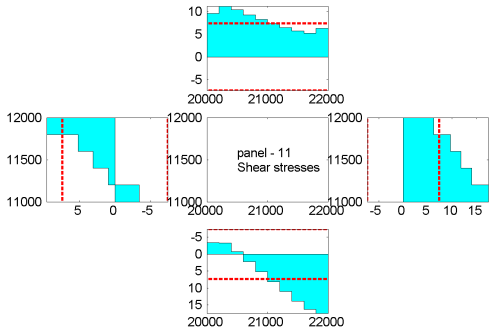
Stiffener Nr.	Stiffener area	Total area	Effective area	Section modulus	St. Venant constant	Polar moment of inertia	Warping constant
17	2496	17496	17496	284334	53248	58600081	5.57e+010
18	2496	17496	7813	283301	53248	58600081	5.57e+010
19	2496	17496	8963	283424	53248	58600081	5.57e+010
20	2496	17496	17496	284334	53248	58600081	5.57e+010
21	2496	17496	8917	283419	53248	58600081	5.57e+010
22	2496	17496	7811	283301	53248	58600081	5.57e+010
23	2496	17496	8380	283362	53248	58600081	5.57e+010
24	2496	17496	8605	283386	53248	58600081	5.57e+010

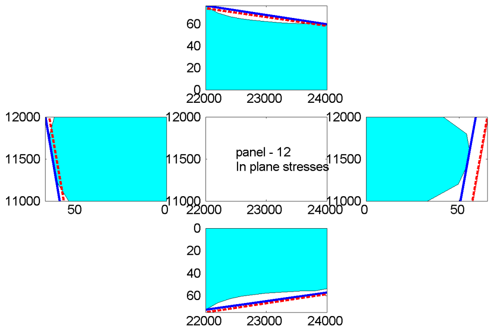

panel - 1
In plane stresses
panel - 1
Shear stresses
panel - 2
In plane stresses
panel - 2
Shear stresses
panel - 3
In plane stresses
panel - 3
Shear stresses
panel - 4
In plane stresses
panel - 4
Shear stresses
panel - 5
In plane stresses
panel - 5
Shear stresses

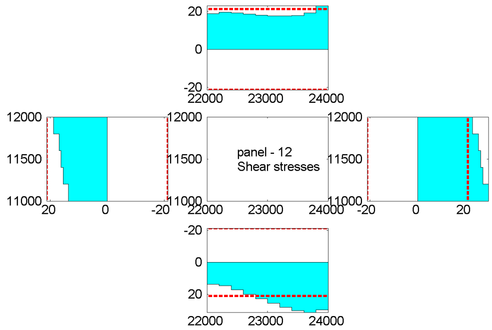


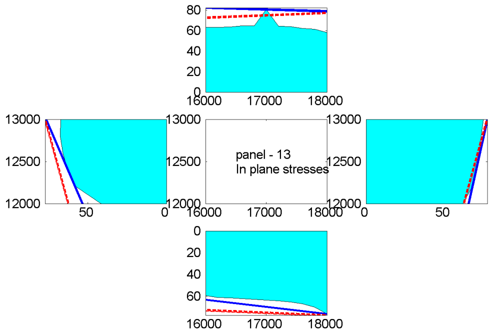
panel - 6
In plane stresses
panel - 6
Shear stresses
panel - 7
In plane stresses
panel - 7
Shear stresses
panel - 8
In plane stresses
panel - 8
Shear stresses
panel - 9
In plane stresses
panel - 9
Shear stresses
panel - 10
In plane stresses
panel - 10
Shear stresses

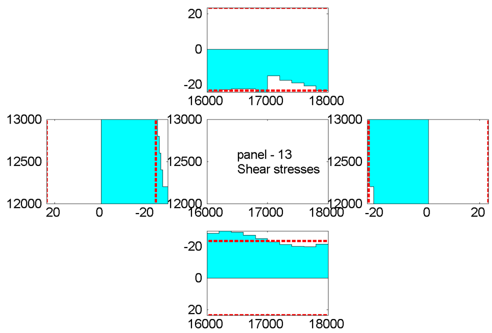

panel - 11
In plane stresses

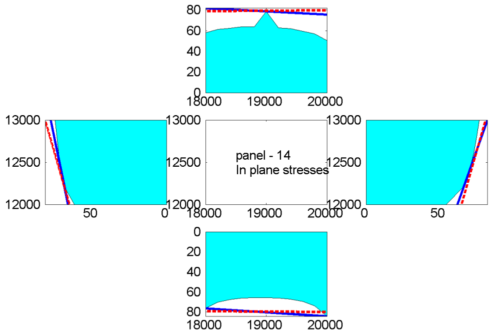

panel - 11
Shear stresses

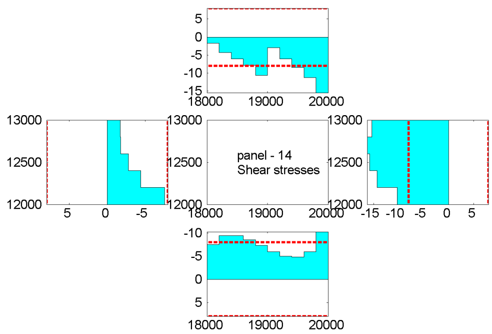

panel - 12
In plane stresses

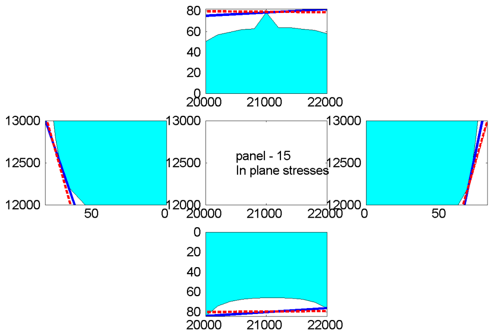

panel - 12
Shear stresses

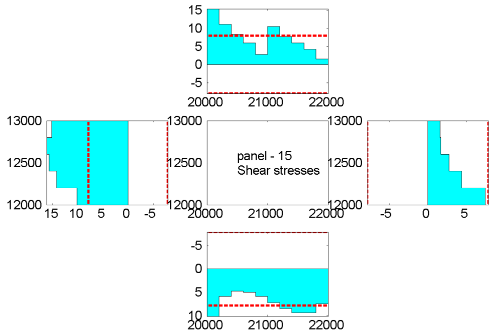

panel - 13
In plane stresses

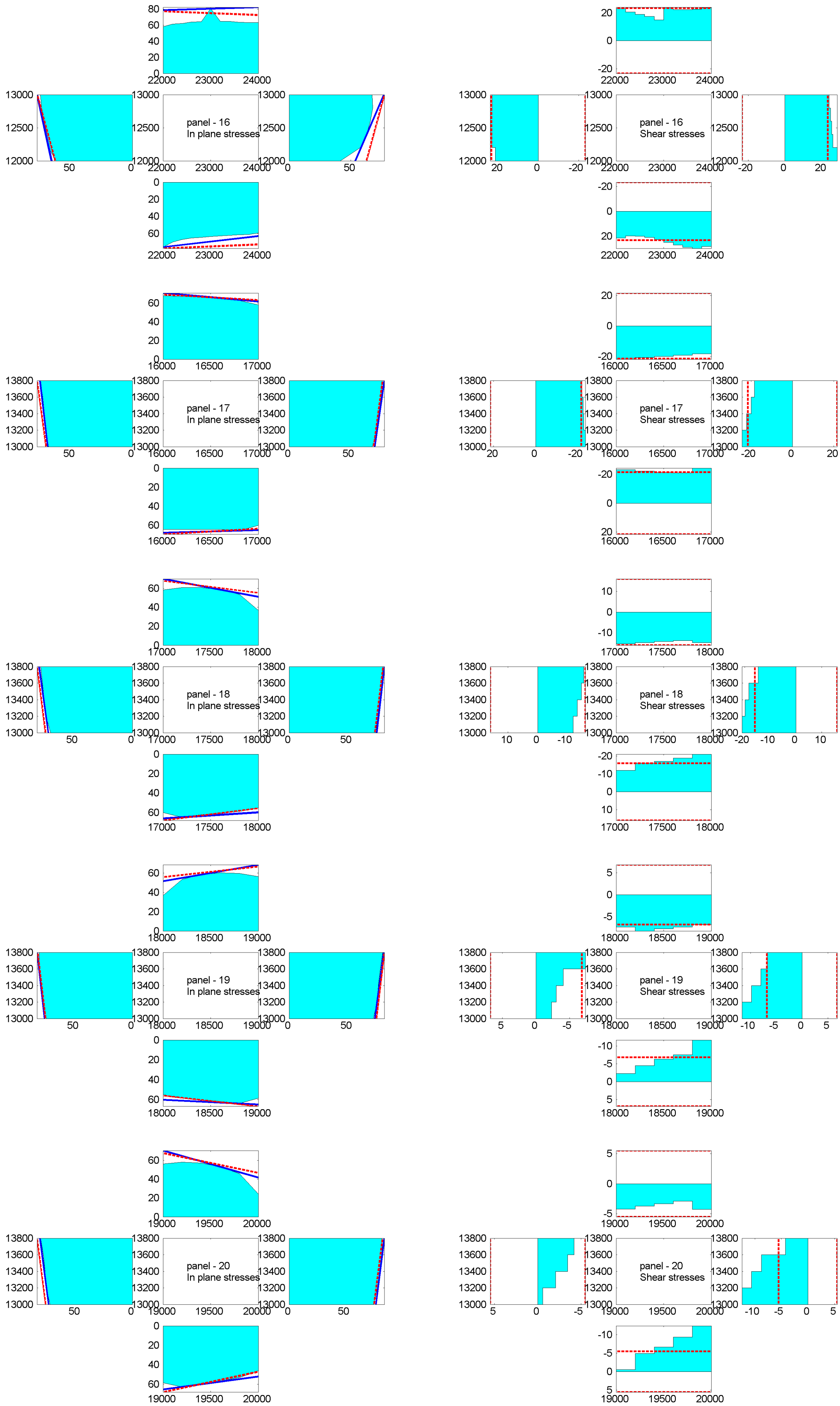

panel - 13
Shear stresses

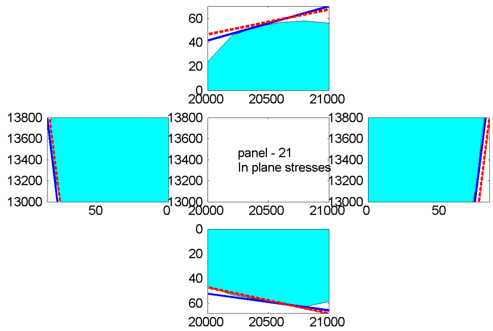
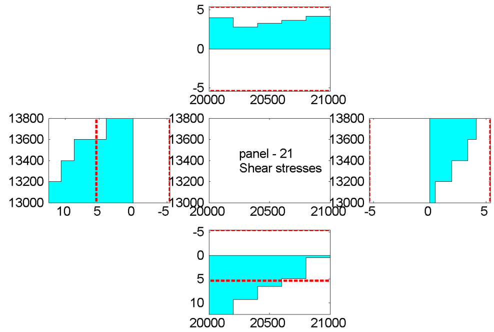
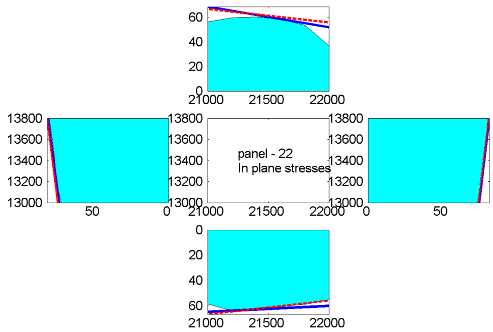
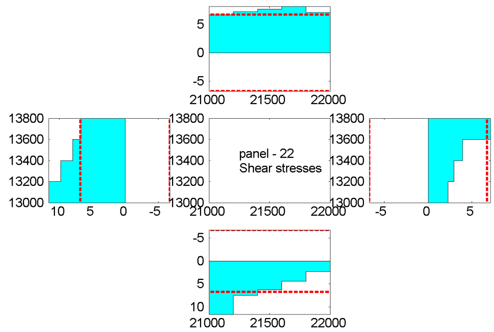
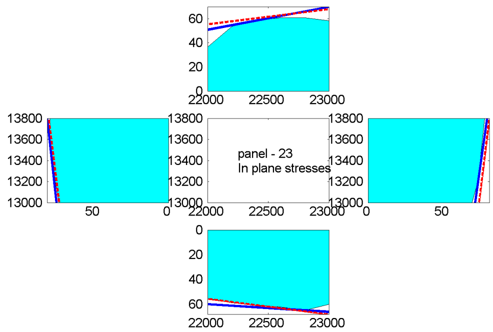
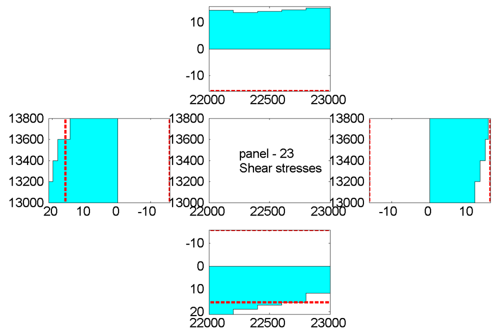
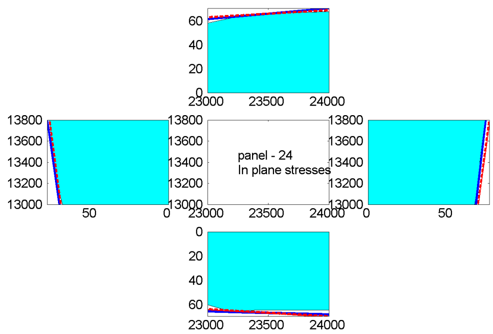
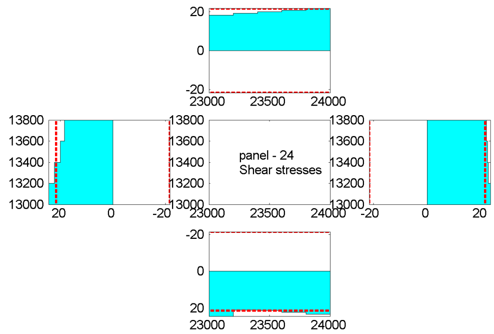
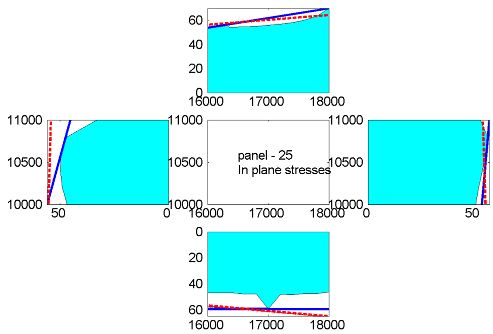
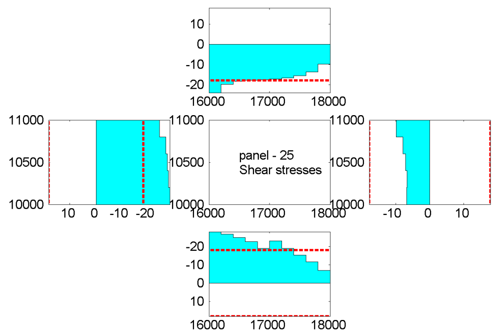

panel - 14
In plane stresses

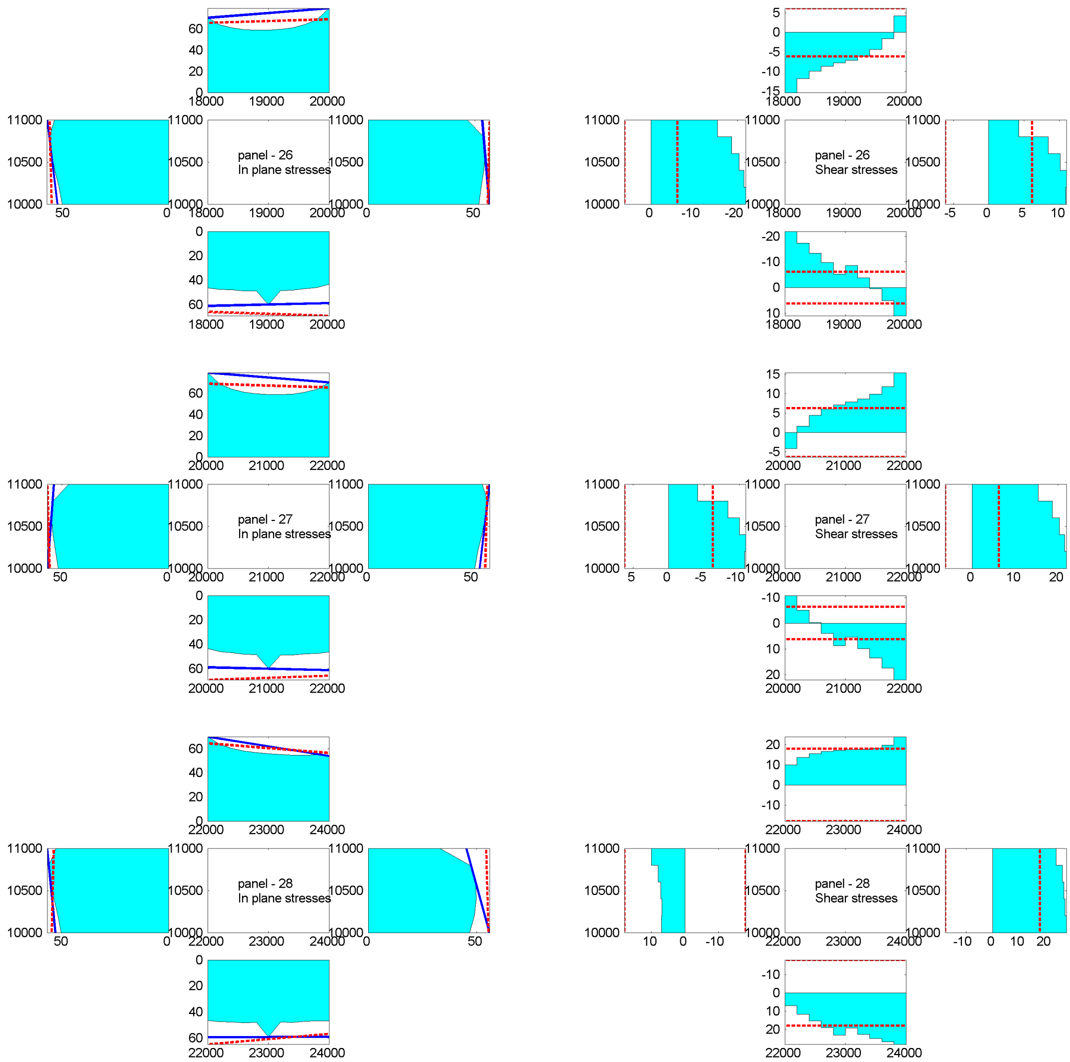

panel - 14
Shear stresses


panel - 15
In plane stresses


panel - 15
Shear stresses


panel - 16
In plane stresses
panel - 16
Shear stresses
panel - 17
In plane stresses
panel - 17
Shear stresses
panel - 18
In plane stresses
panel - 18
Shear stresses
panel - 19
In plane stresses
panel - 19
Shear stresses
panel - 20
In plane stresses
panel - 20
Shear stresses


panel - 21
In plane stresses

panel - 21
Shear stresses

panel - 22
In plane stresses

panel - 22
Shear stresses

panel - 23
In plane stresses

panel - 23
Shear stresses

panel - 24
In plane stresses

panel - 24
Shear stresses

panel - 25
In plane stresses

panel - 25
Shear stresses


panel - 26
In plane stresses
panel - 26
Shear stresses
panel - 27
In plane stresses
panel - 27
Shear stresses
panel - 28
In plane stresses
panel - 28
Shear stresses

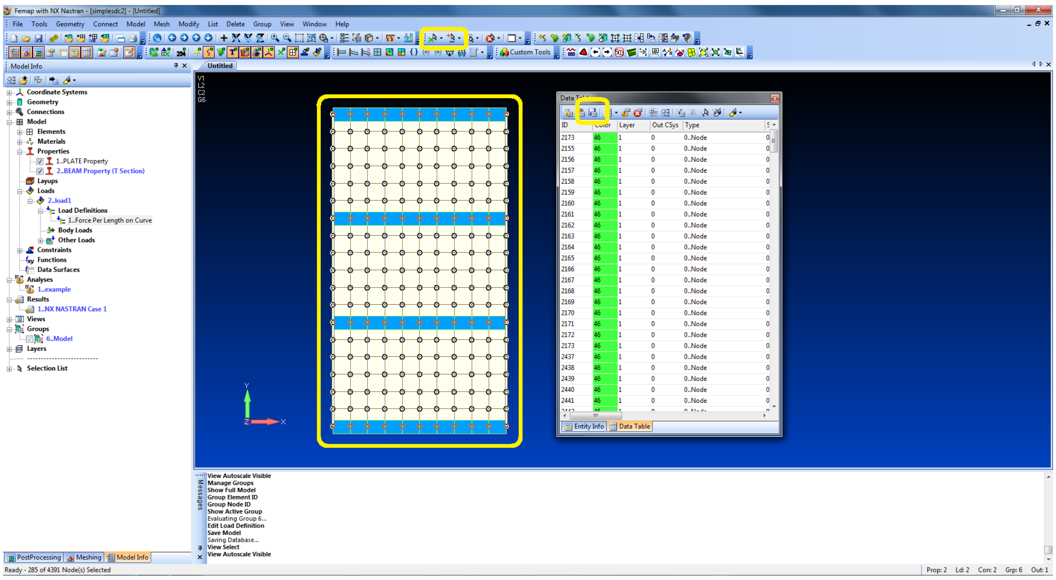
Appendix C

Reading Information from Femap

The programming of the Stress Check Model is done with MATLAB where results from Femap are exported out of Femap as text files. The "data table" feature of Femap is used to collect the information and save it as a text files. The text files are given a name and are imported into MATLAB as matrices with the same names. The following sections describe how the necessary information are gathered from Femap.

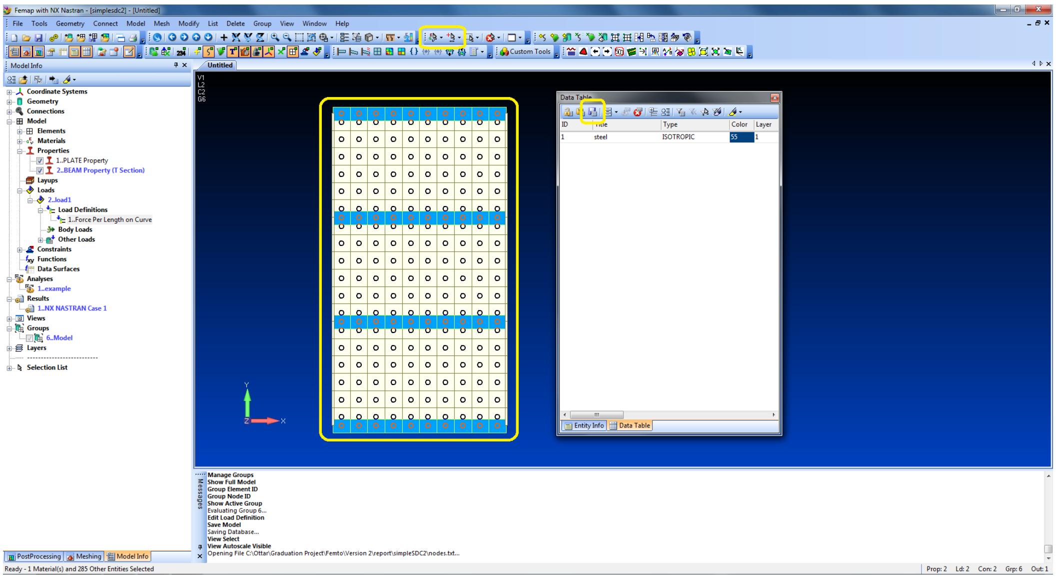
C.1 Nodes

The Entity Selector is set to *Node* and the Mode Selector is set to select *Multiple*. All the nodes of the model are selected and the data table is saved under the name *nodes.txt*.



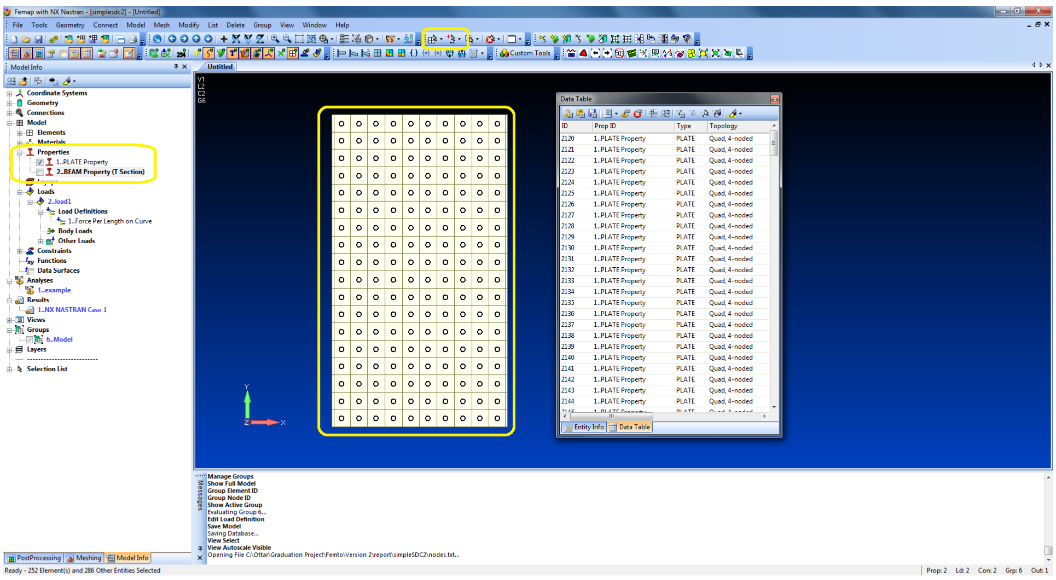
C.2 Material

The Entity Selector is set to *Material* and the Mode Selector is set to select *Multiple*. All the elements of the model are selected and the data table is saved under the name *material.txt*.

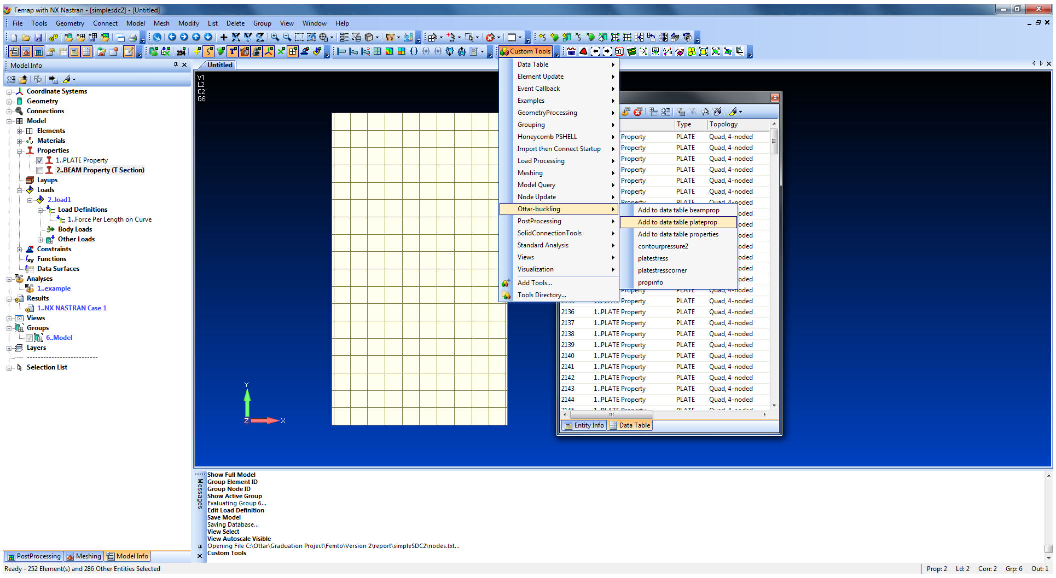


C.3 Plate Elements

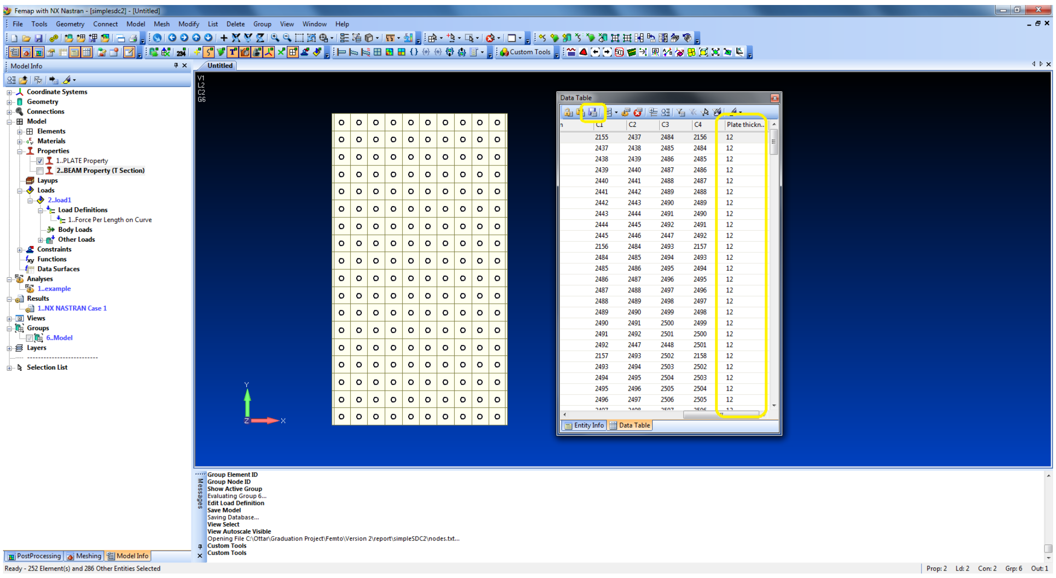
The Entity Selector is set to *Elements* and the Mode Selector is set to select *Multiple*. Only element properties which belong to plates are checked in the Properties Branch of the Model Info Tree. All plate elements of the model are selected.



A small application was written with the VB programming in Femap to add the cross sectional properties of plate elements to the data table. This application is not part of the standard Femap applications.

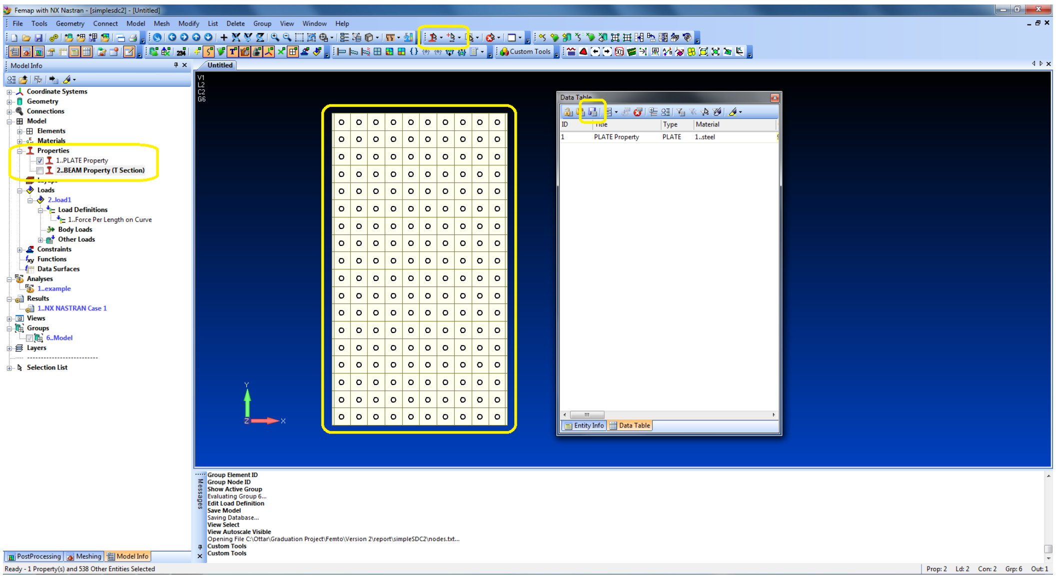


A new column is added to the data table containing the thickness of the plate elements. The data table is saved under the name *plates.txt*.



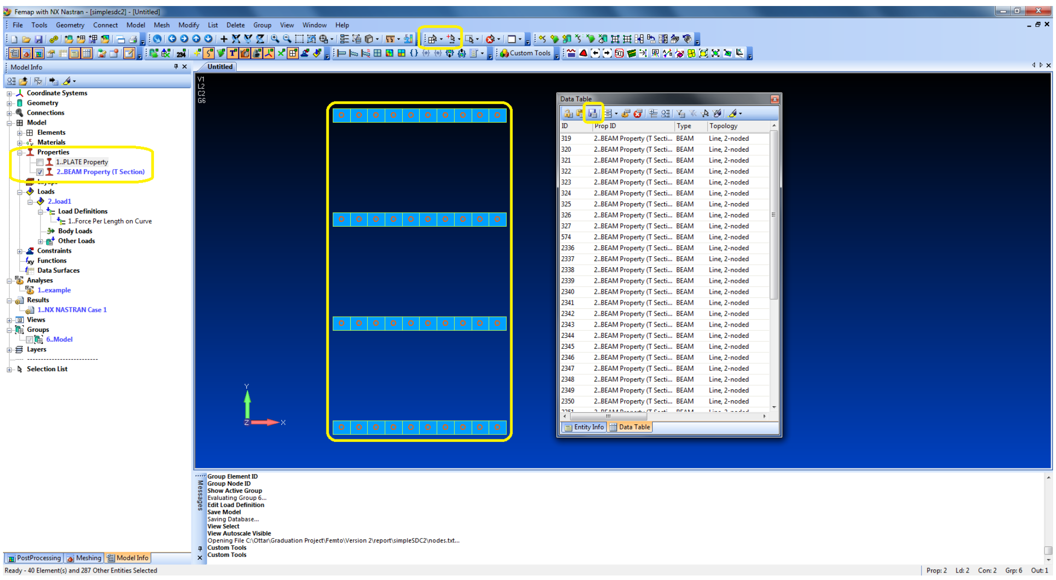
C.4 Plate Element Properties

The Entity Selector is set to *Property* and the Mode Selector is set to select *Multiple*. All plate elements of the model are selected and the data table is saved under the name *plateprop.txt*.

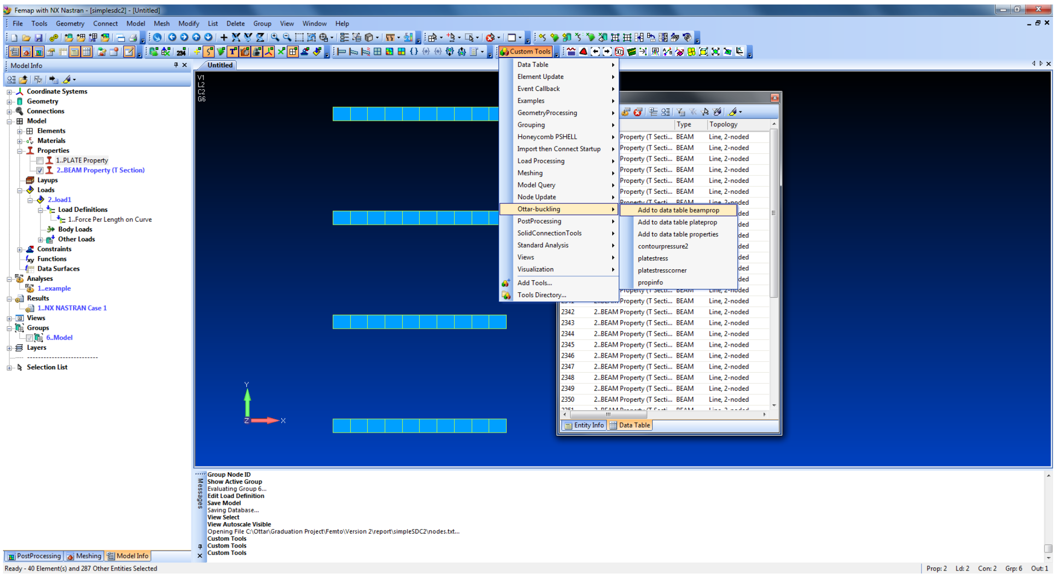


C.5 Beam Elements

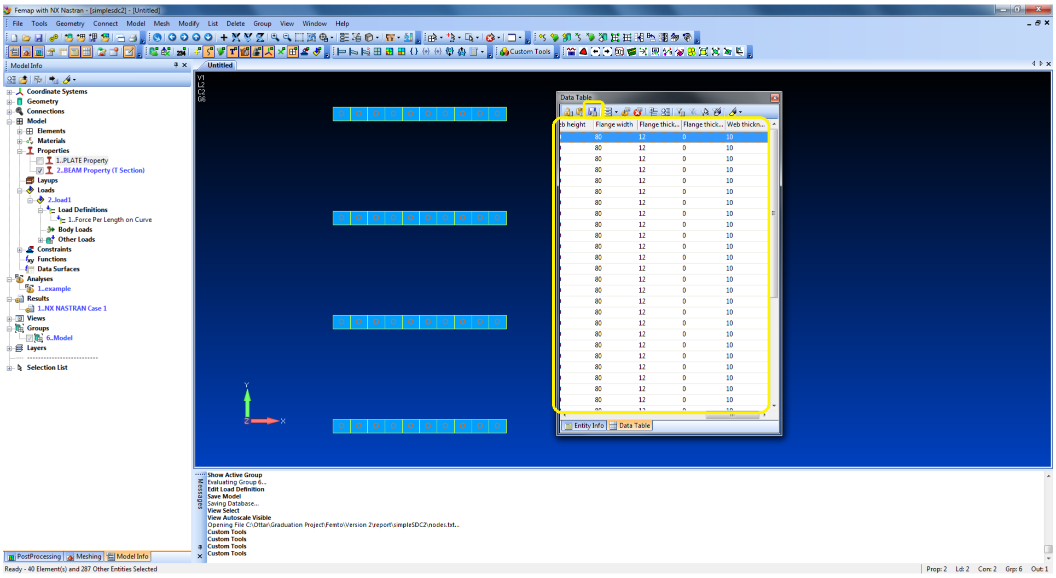
The Entity Selector is set to *Elements* and the Mode Selector is set to select *Multiple*. Only element properties which belong to beams are checked in the Properties Branch of the Model Info Tree. All beam elements of the model are selected.



A small application was written with the VB programming in Femap to add the cross sectional properties of beam elements to the data table. This application is not part of the standard Femap applications.

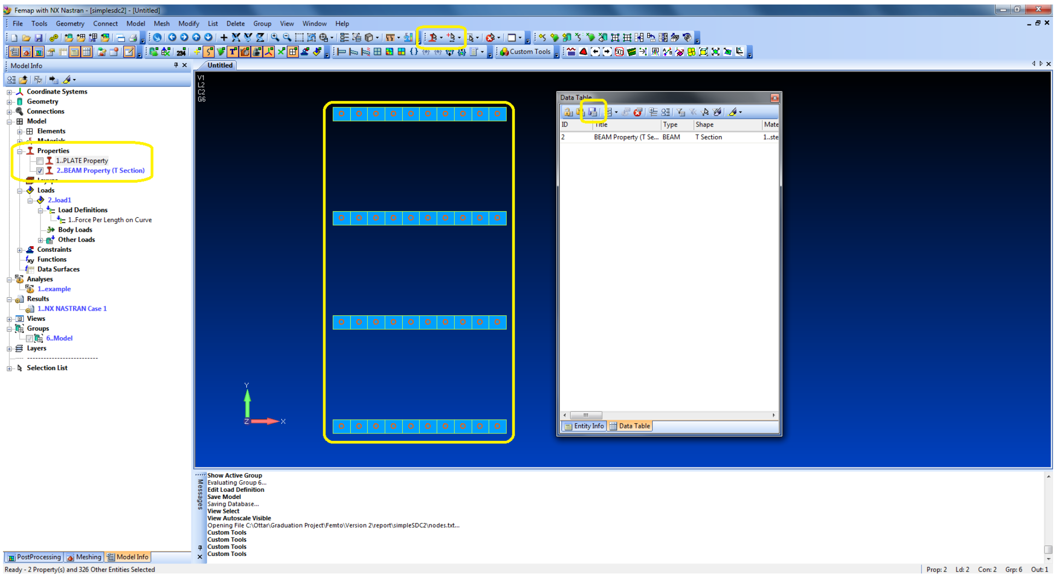


New columns are added to the data table containing the cross section properties of the beam elements. The data table is saved under the name *beams.txt*.



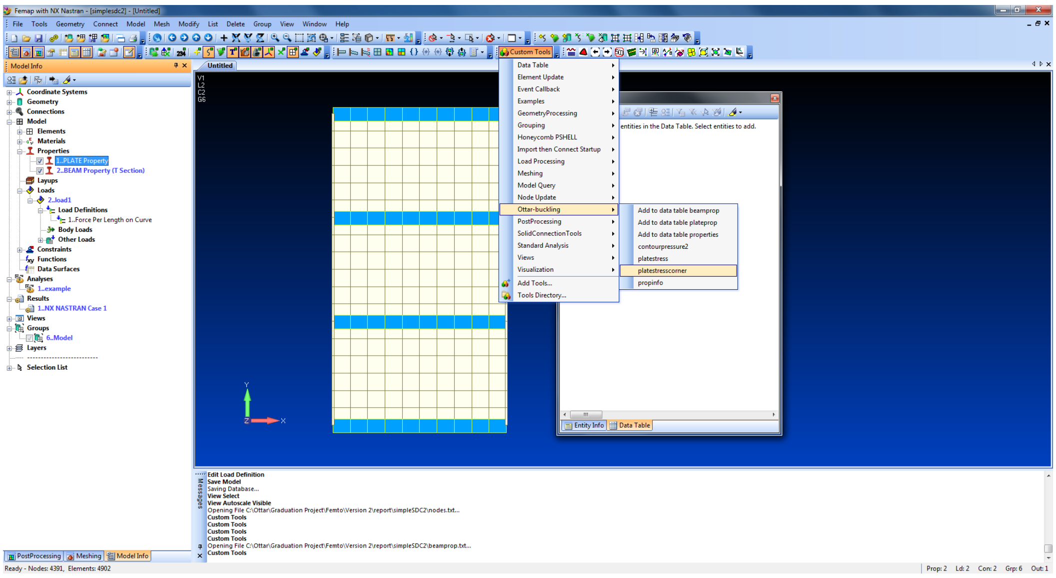
C.6 Beam Element Properties

The Entity Selector is set to *Property* and the Mode Selector is set to select *Multiple*. All beam elements of the model are selected and the data table is saved under the name *beamprop.txt*.

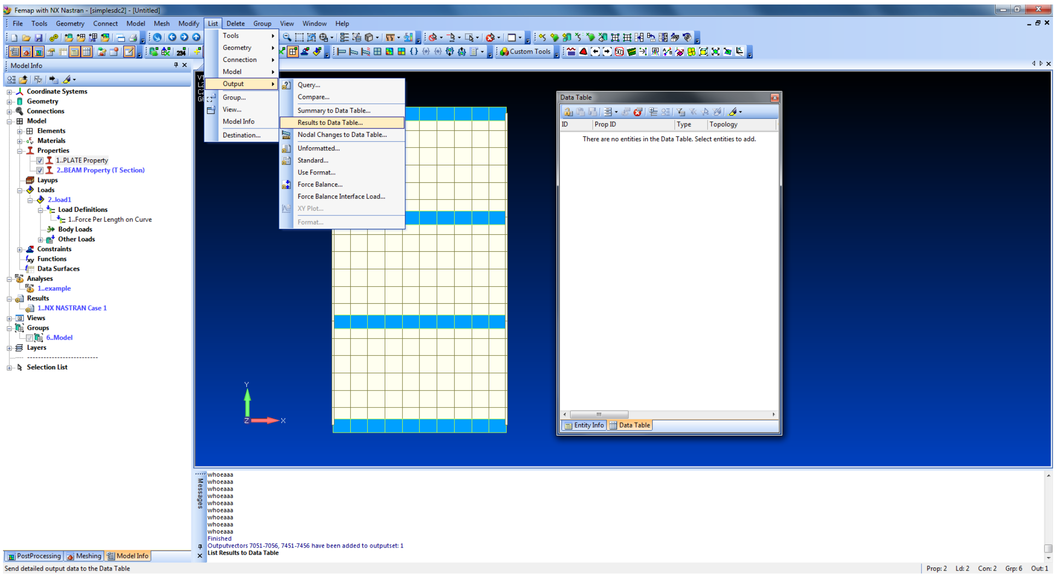


C.7 Stresses

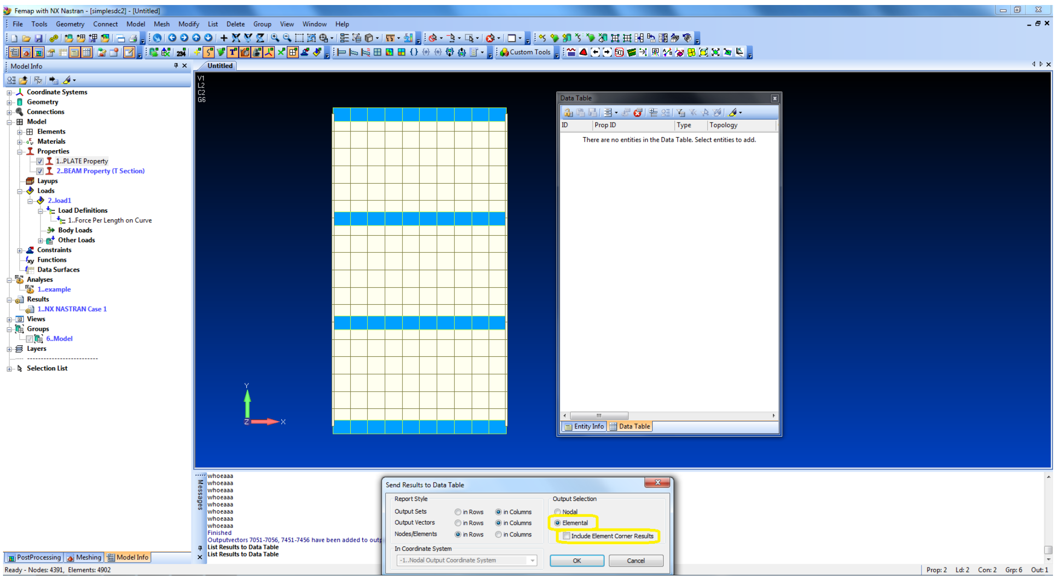
A small application was written with the VB programming in Femap to add stress results in global directions at plate elements nodes as an output vector. This application is not part of the standard Femap applications.



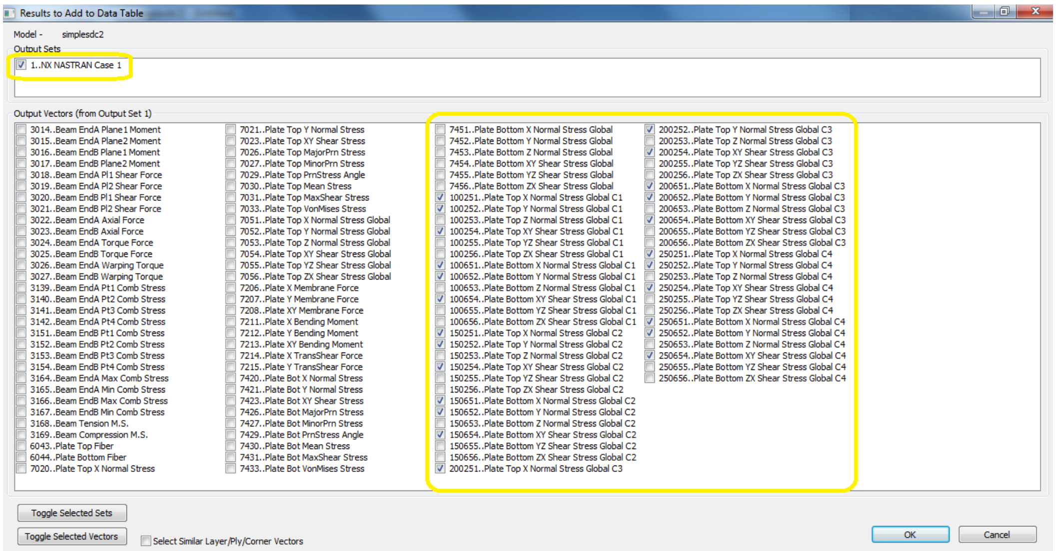
The results can be accessed by selecting the *List* drop down menu and then *Output* and *Results to Data Table*.



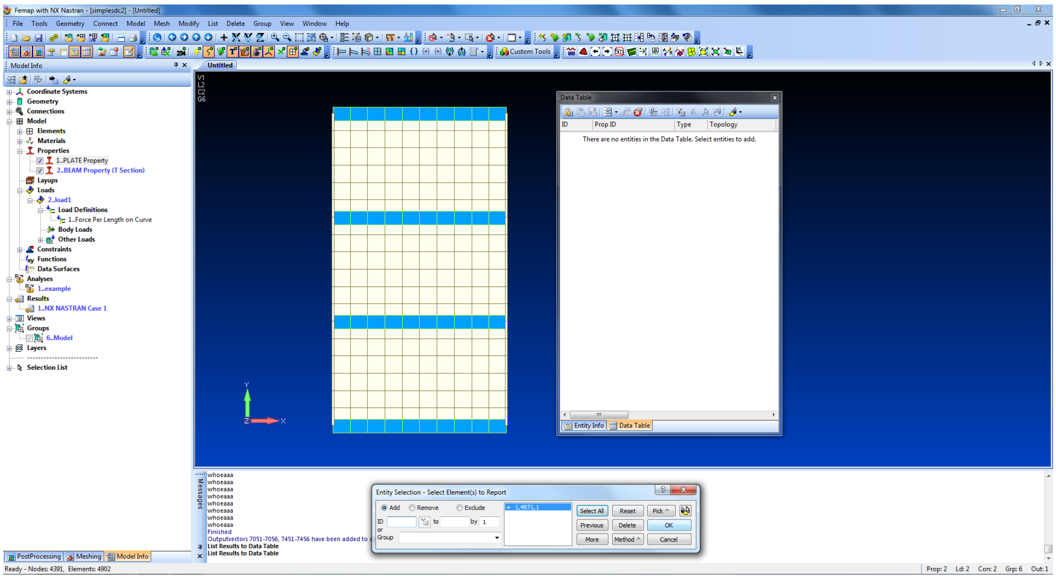
The output selection should be *Elemental* and for conveniance the *Include Element Corner Result* box is unchecked.



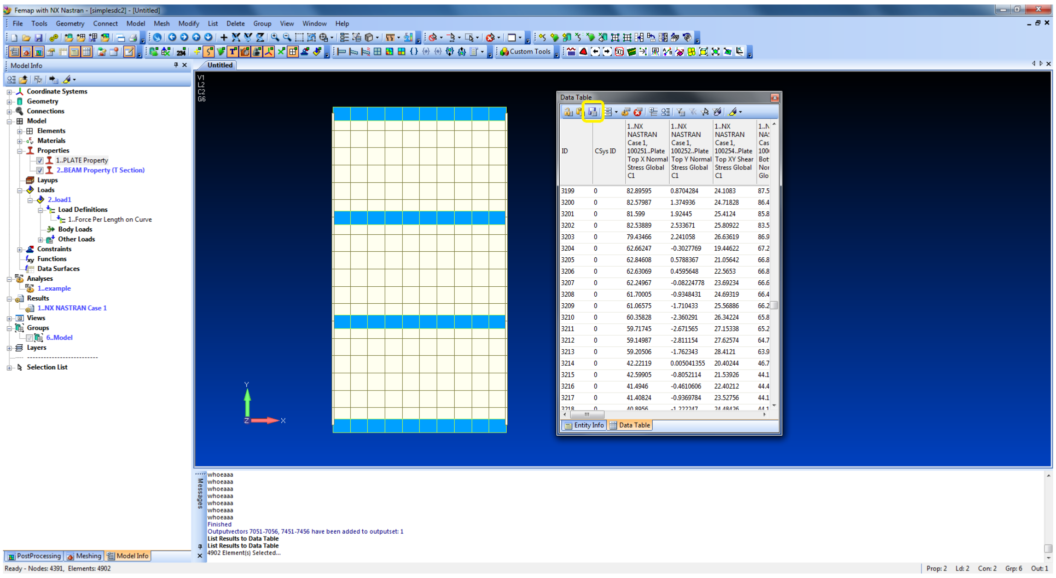
The stresses at top and bottom fibres at every node are selected for x- and y-normal stresses and xy-shear stresses.



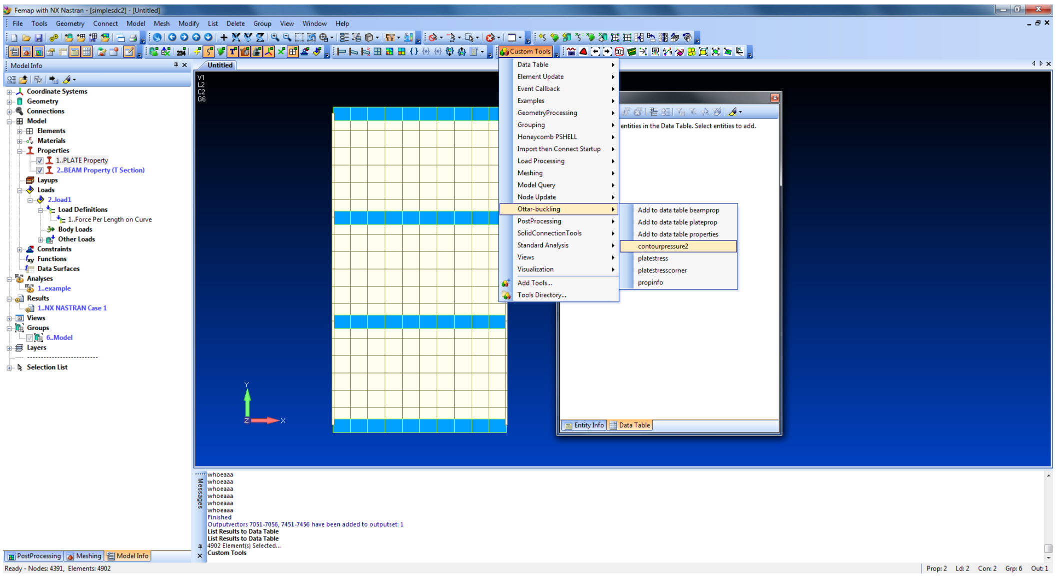
All the elements of the model are selected using the *Select All* option in the *Entity Selection*.



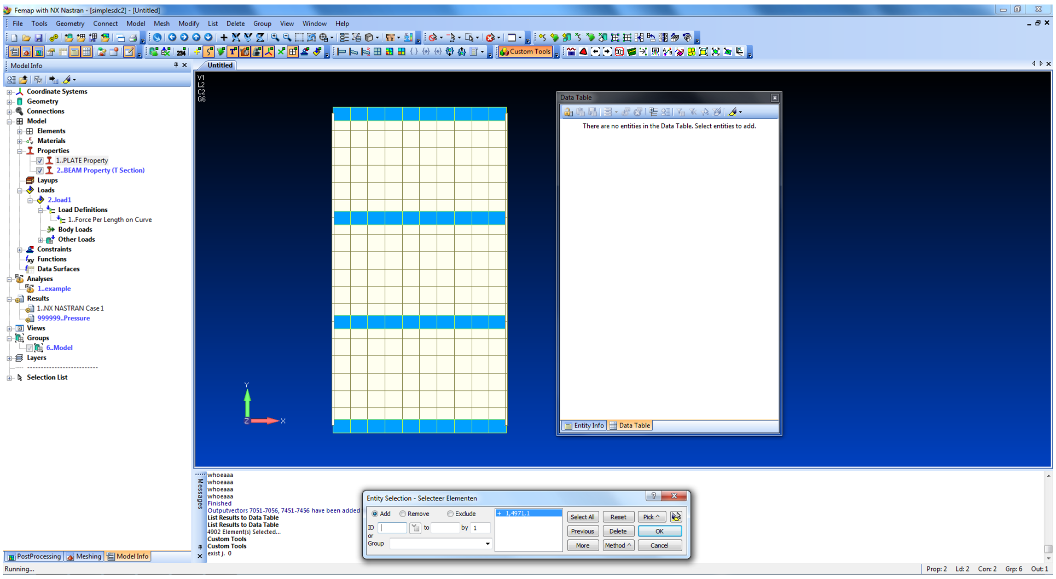
The final step is to save the data table under the name *stresses.txt*.



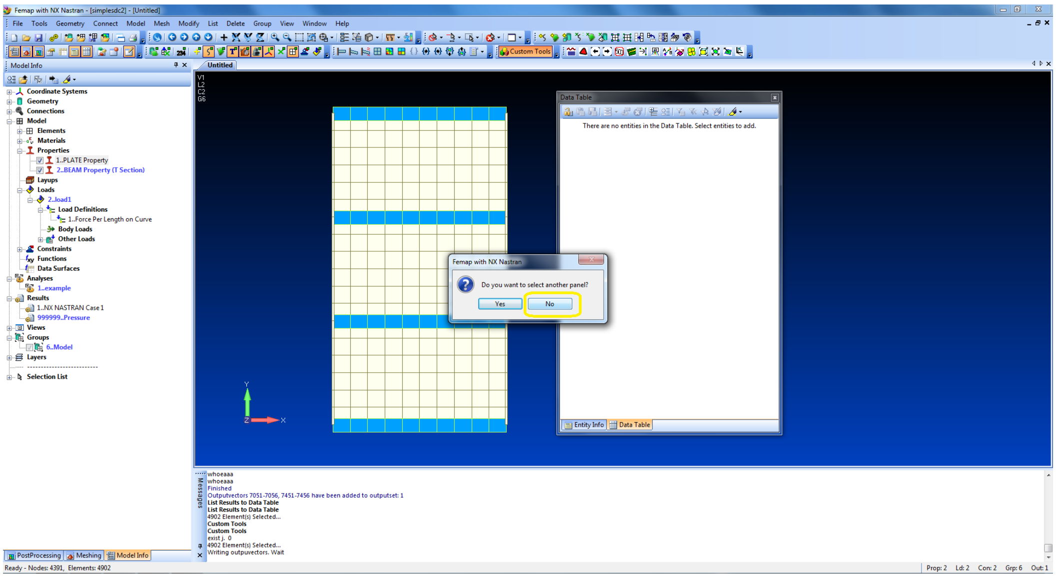
A small application was written with the VB programming in Femap to add lateral load acting at plate elements surfaces as an output vector. This application is not part of the standard Femap applications.



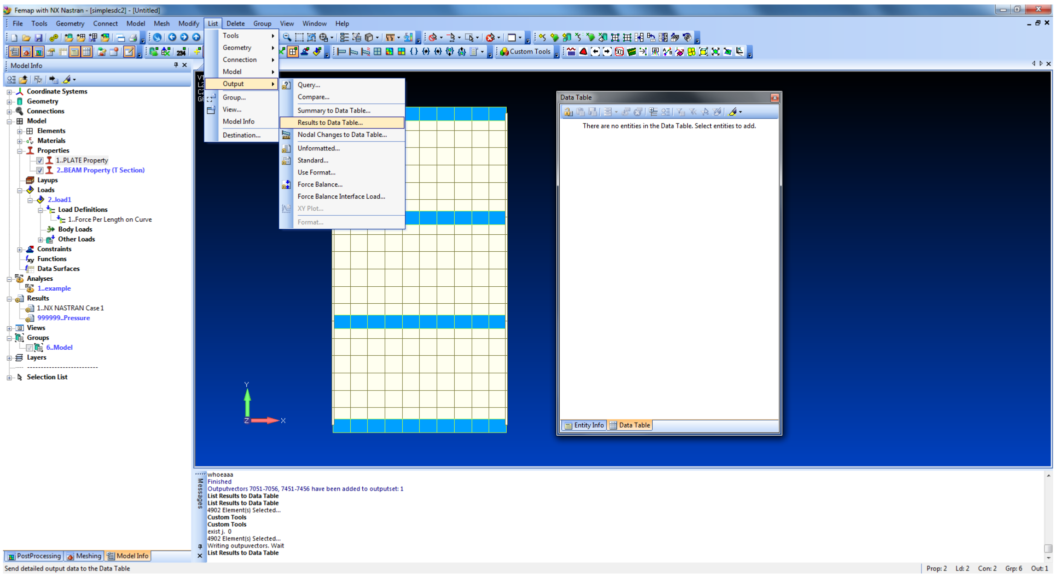
All the elements of the model are selected using the *Select All* option in the *Entity Selection*.



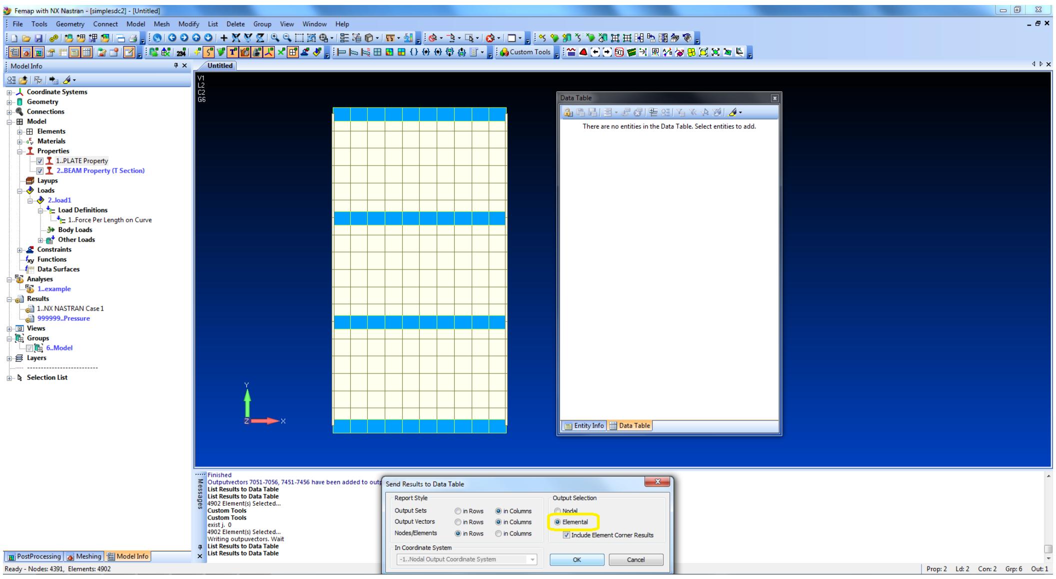
When the program ask "'Do you want to select another panel?"' cklick *no*.



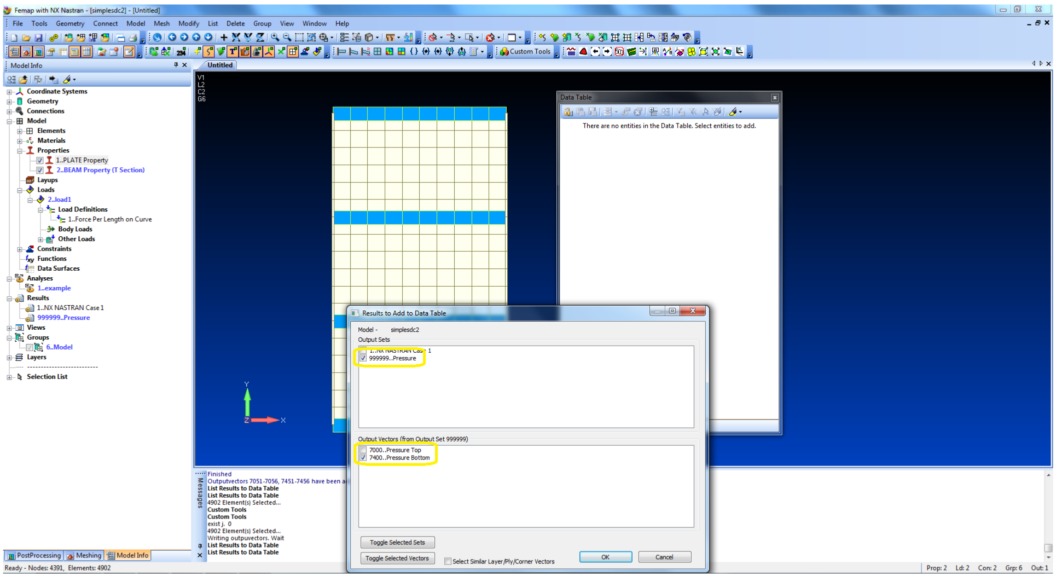
The results can be accessed by selecting the *List* drop down menu and then *Output* and *Results to Data Table*.



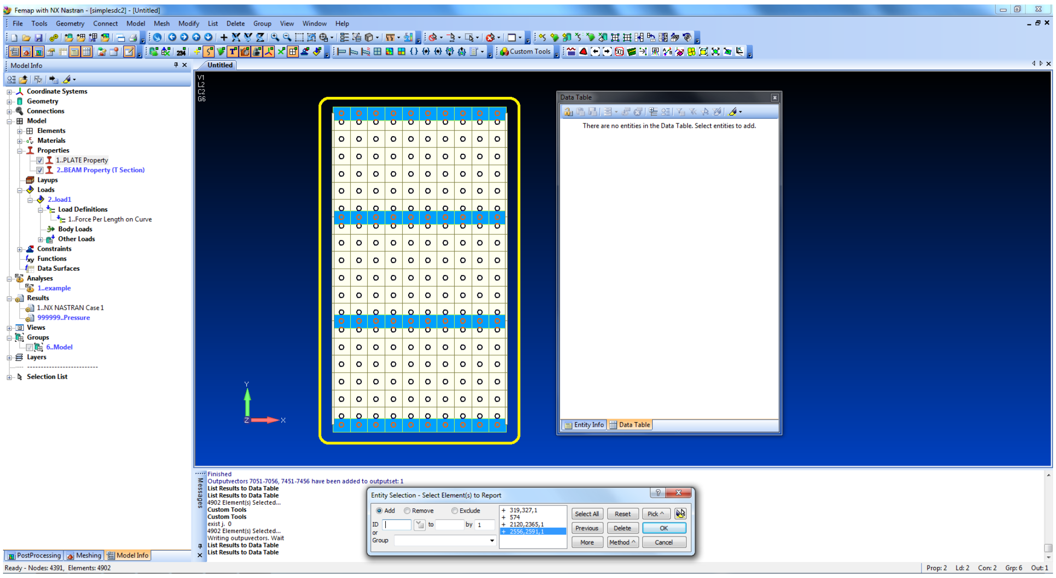
The output selection should be *Elemental*.



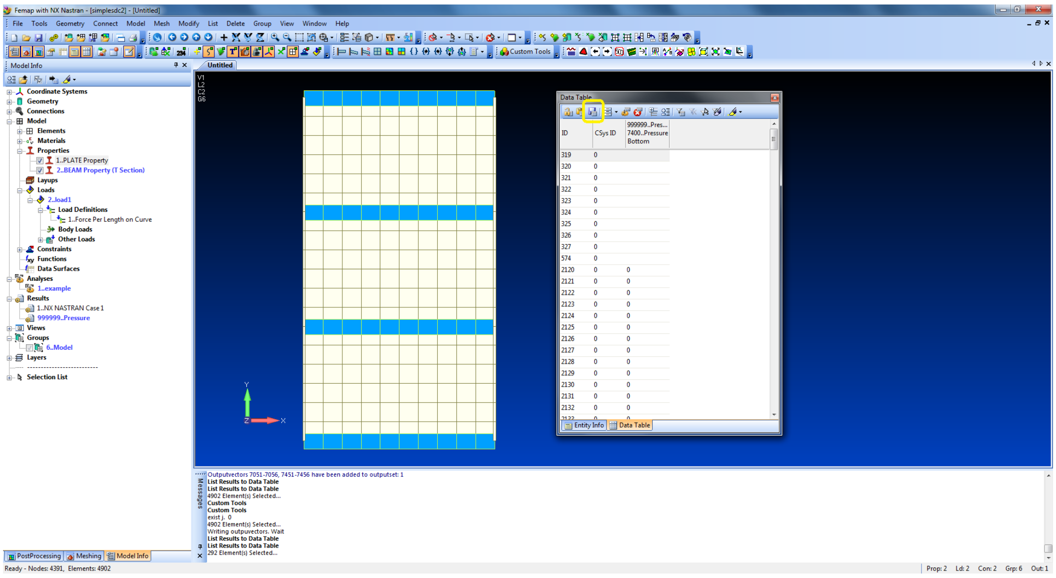
The output set *Pressure* and the output vector *Pressure Bottom* are selected.



All the elements of the model are selected using the *Select All* option in the *Entity Selection.*



The final step is to save the data table under the name *stresses.txt*.



Appendix D

MATLAB Programs

D.1 Assemble.m

```
% Function name: Assemble.m
% Written by: Ottar Hillers, September 2011
% Purpose: To assemble sub−programs
% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
clear
close all
clc

% Sub−programs
Settings
FEMread
Geometry
Stresses
ABSpackage_panels
ABSpackage_stiffened_panels
Results
```

D.2 Settings.m

```
% Function name: Settings.m
% Written by: Ottar Hillers, September 2011
% Purpose: To set plot and stress determination settings for the Stress
% Check Model
% −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
%% Plotting settings [0−off, 1−on]
% Plot plate FEM elements and their ID to compare to element numbering from
% the FEM model
plot_plates = 0;

% Plot beam FEM elements and their ID to compare to element numbering from
% the FEM model
plot_beams = 0;

% Plot normal stress distribution within every panel
plot_planestresses = 0;

% Plot shear stress distribution within every panel
plot_shearstresses = 0;

```

```
%% Calculation preferences
% Determine whether to use the mean value of the top and bottom stresses at
% nodes or use the higher value (compressive)
% [0-max values, 1-average values]
stress_settings = 1;

% Determine whether to use clause 4.6.(3) in Eurocode 3 part 1-5 to
% determine the in-plane design stresses or use maximum values
% [0 = Maximum values, 1 = clause 4.6.(3)]
inplane_settings = 1;

% Determine constant shear stress from actual shear stress distribution.
% The maximum shear stress value along the edges of the panel or
% average absolute value or average value along the edges of the panel
% [0-max absolute value, 1 = average absolute value, 2 = average value]
shear_settings = 2;

% Determine constant lateral pressure from actual lateral pressure
% distribution. The maximum lateral pressure applied to each panel or
% average value appiled to the panel
% [0-max values, 1-average values]
lateral_settings = 1;
```

D.3 FEMread.m

```
% Function name: FEMread.m
% Written by: Ottar Hillers, May 2011
% Purpose: To read and organise necessary parameters for the Stress Check
% Model analysis of stiffened panels from Femap
% -----------------------------------------------------------------------
% INPUTS
% nodes.txt
% material.txt
% plateprop.txt
% beamprop.txt
% plates.txt
% beams.txt
% stresses.txt
% lateral.txt
% -----------------------------------------------------------------------
% OUTPUTS
% plate_coordinate(ID,ID_fem,x1,y1,z1,x2,y2,z2,x3,y3,z3,x4,y4,z4)
% plate_information(ID,ID_fem,E,v,t)
% beam_coordinate(ID,ID_fem,x1,y1,z1,x2,y2,z2)
% beam_information(ID,ID_fem,E,v,hw,bf,tf,tw,type(1=T-shape,2=L-shape))
% stress_information (ID,IDfem,C1x,C1y,C1xy,C2x,C2y,C2xy,C3x,C3y,C3xy,
% C4x,C4y,C4xy,q)
% -----------------------------------------------------------------------
%% Read nodes
node = importdata('nodes.txt');
% Sort the nodes information (ID,x,y,z)
node_info = str2double(node.textdata(2:end,[1 12:14]));
ez = 1e-10;
node_info(find(abs(node_info(:,4))<ez),4) = 0;

%% Read material properties
material = importdata('material.txt');
% Recognise different types of material to compare later to plate and beam
% materials
material_type = material.textdata(2:end,2);
% Sort the material information (E,v)
material_info = material.data(:,[6 8]);

%% Read plate nodes
```

```
plate = importdata('plates.txt');
% Sort the plate nodes information
plate_info = [str2double(plate.textdata(2:end,1)) plate.data];

% Relate coordinates of plates to nodes (ID,ID_fem,x1,y1,z1,x2,y2,z2,x3,y3,
% z3,x4,y4,z4)
n = size(plate_info);
plate_coordinate = zeros(n(1),14);
% New ID
plate_coordinate(:,1) = 1:n(1);
% Femap ID
plate_coordinate(:,2) = plate_info(:,1);
for i = 1:n(1)
    index1 = find(node_info(:,1)==plate_info(i,2));
    index2 = find(node_info(:,1)==plate_info(i,3));
    index3 = find(node_info(:,1)==plate_info(i,4));
    index4 = find(node_info(:,1)==plate_info(i,5));
    plate_coordinate(i,3:5) = node_info(index1,2:4);
    plate_coordinate(i,6:8) = node_info(index2,2:4);
    plate_coordinate(i,9:11) = node_info(index3,2:4);
    plate_coordinate(i,12:14) = node_info(index4,2:4);
end

% Plot the plate elements
if plot_plates == true
    figure
    hold on
    for i = 1:n(1)
        plot3(plate_coordinate(i,[3 6 9 12 3]), ...
            plate_coordinate(i,[4 7 10 13 4]), ...
            plate_coordinate(i,[5 8 11 14 5]))
        text((plate_coordinate(i,3)+plate_coordinate(i,9))/2, ...
            (plate_coordinate(i,4)+plate_coordinate(i,10))/2, ...
            (plate_coordinate(i,5)+plate_coordinate(i,11))/2, ...
            num2str(plate_coordinate(i,2)))
    end
    axis equal
end

%% Read plate properties
plate_prop = importdata('plateprop.txt');
% Recognise different plate types to compare later to every plate element
% of the mesh
plate_type = plate_prop.textdata(2:end,2);
% Recognise plate material
v = char(plate_prop.textdata(2:end,4));
v = v(1:end,4:end);
plate_material = cellstr(v);
% Sort plate properties with material properties (E,v,t)
n = size(plate_material);
plate_properties = zeros(n(1),2);
% Match plate material with material type
for i = 1:n(1)
    index = find(ismember(material_type, plate_material(i))==1);
    plate_properties(i,1:2) = material_info(index,1:2);
end
% plate_properties(:,3) = str2double(plate_prop.textdata(2:end,7));

%% Recognise plate element properties
v = char(plate.textdata(2:end,2));
v = v(1:end,4:end);
b = find(v==char(46));
if isempty(b) == false
    c = v(b,2:end);
    c(:,end+1) = char(32);
    v(b,:) = c;
end
plate_char = cellstr(v);
```

```
% Sort plate properties and materials with plate elements (ID,ID_fem,E,v,t)
n = size(plate_char);
plate_information = zeros(n(1),5);
plate_information(1:end,1:2) = plate_coordinate(1:end,1:2);
for i = 1:n(1)
    index = find(ismember(plate_type, plate_char(i))==1);
    plate_information(i,3:4) = plate_properties(index,1:2);
    plate_information(i,5) = plate_info(i,end);
end

%% Read beam nodes
beam = importdata('beams.txt');
% Sort the beam nodes information
beam_info = [str2double(beam.textdata(2:end,1)) beam.data];

% Relate coordinates of beams to nodes
n = size(beam_info);
beam_coordinate = zeros(n(1),8);
% New ID
beam_coordinate(:,1) = 1:n(1);
% Femap ID
beam_coordinate(:,2) = beam_info(:,1);
for i = 1:n(1)
    index1 = find(node_info(:,1)==beam_info(i,2));
    index2 = find(node_info(:,1)==beam_info(i,3));
    beam_coordinate(i,3:5) = node_info(index1,2:4);
    beam_coordinate(i,6:8) = node_info(index2,2:4);
end

% Plot the beam elements
if plot_beams == true
    figure
    hold on
    for i = 1:n(1)
        plot3(beam_coordinate(i,[3 6 3]), ...
            beam_coordinate(i,[4 7 4]), ...
            beam_coordinate(i,[5 8 5]))
        text((beam_coordinate(i,3)+beam_coordinate(i,6))/2, ...
            (beam_coordinate(i,4)+beam_coordinate(i,7))/2, ...
            (beam_coordinate(i,5)+beam_coordinate(i,8))/2, ...
            num2str(beam_coordinate(i,2)))
    end
    axis equal
end

%% Read beam properties
beam_prop = importdata('beamprop.txt');
% Recognise different beam types to compare later to every beam element
% of the mesh
beam_type = beam_prop.textdata(2:end,2);
% Recognise beam material
v = char(beam_prop.textdata(2:end,5));
v = v(1:end,4:end);
beam_material = cellstr(v);
% Sort beam properties with material properties (E,v)
n = size(beam_material);
beam_properties = zeros(n(1),2);
% Match plate material with material type
for i = 1:n(1)
    index = find(ismember(material_type, beam_material(i))==1);
    beam_properties(i,1:2) = material_info(index,1:2);
end

%% Recognise beam element properties
v = char(beam.textdata(2:end,2));
v = v(1:end,4:end);
b = find(v==char(46));
if isempty(b) == false
    c = v(b,2:end);
```

```
    c(:,end+1) = char(32);
    v(b,:) = c;
end
beam_char = cellstr(v);

n = size(beam_char);
beam_information = zeros(n(1),9);
beam_information(1:end,1:2) = beam_coordinate(1:end,1:2);

for i = 1:n(1)
    index = find(ismember(beam_type, beam_char(i))==1);
    tf = max(beam_info(i,6),beam_info(i,7));
    type = find([beam_info(i,6),beam_info(i,7)]== ...
        max(beam_info(i,6),beam_info(i,7)));
    beam_information(i,3:4) = beam_properties(index,1:2);
    beam_information(i,7) = tf;
    beam_information(i,9) = type;
end

beam_information(:,5:6)= beam_info(:,4:5);
beam_information(:,8)= beam_info(:,8);

%% Read stress results and lateral pressure
stresses = importdata('stresses.txt');
lateral = importdata('lateral.txt');

A = stresses.data;
[R C] = find(isnan(A));
A(unique(R),:) = [];

B = lateral.data;
B(unique(R),:) = [];
q = A(:,[1 3]);

sigma = A(:,3:end);
n = size(A);
% stress_information (ID,IDfem,C1x,C1y,C1xy,C2x,C2y,C2xy,C3x,C3y,C3xy,
% C4x,C4y,C4xy,q)
stress_information = zeros(n(1),15);
stress_information(:,1) = 1:n(1);
stress_information(:,2) = A(:,1);
stress_information(:,15) = B(:,3);
if stress_settings == 0
    stress_information(:,3) = min(sigma(:,[1 4]),[],2);
    stress_information(:,4) = min(sigma(:,[2 5]),[],2);
    stress_information(:,5) = min(sigma(:,[3 6]),[],2);
    stress_information(:,6) = min(sigma(:,[7 10]),[],2);
    stress_information(:,7) = min(sigma(:,[8 11]),[],2);
    stress_information(:,8) = min(sigma(:,[9 12]),[],2);
    stress_information(:,9) = min(sigma(:,[13 16]),[],2);
    stress_information(:,10) = min(sigma(:,[14 17]),[],2);
    stress_information(:,11) = min(sigma(:,[15 18]),[],2);
    stress_information(:,12) = min(sigma(:,[19 22]),[],2);
    stress_information(:,13) = min(sigma(:,[20 23]),[],2);
    stress_information(:,14) = min(sigma(:,[21 24]),[],2);
else
    stress_information(:,3) = mean(sigma(:,[1 4]),2);
    stress_information(:,4) = mean(sigma(:,[2 5]),2);
    stress_information(:,5) = mean(sigma(:,[3 6]),2);
    stress_information(:,6) = mean(sigma(:,[7 10]),2);
    stress_information(:,7) = mean(sigma(:,[8 11]),2);
    stress_information(:,8) = mean(sigma(:,[9 12]),2);
    stress_information(:,9) = mean(sigma(:,[13 16]),2);
    stress_information(:,10) = mean(sigma(:,[14 17]),2);
    stress_information(:,11) = mean(sigma(:,[15 18]),2);
    stress_information(:,12) = mean(sigma(:,[19 22]),2);
    stress_information(:,13) = mean(sigma(:,[20 23]),2);
    stress_information(:,14) = mean(sigma(:,[21 24]),2);
end
```

D.4 Geometry.m

```
% Function name: Geometry.m
% Written by: Ottar Hillers, August 2011
% Purpose: Gather the necessary input parameters for stress check model
% ---------------------------------------------------------------------------
% OUTPUTS
% panel_info(ID,Length,Width,E,v,t)
% transtiff_info(ID,Length,E,v,hw,bf,tf,tw)
% longstiff_info(ID,Length,E,v,hw,bf,tf,tw,type(1=T-shape,2=L-shape))
% ---------------------------------------------------------------------------
%% Nodes attached to stiffeners and girders
ex = 1e-10;     % Tolerances in x-direction
ey = 1e-10;     % Tolerances in y-direction
ez = 1e-10;     % Tolerances in z-direction

z_ref = 0;      % Reference z-coordinate

% Beam elements
% Length and width differences
delta_x = abs(beam_coordinate(:,3)-beam_coordinate(:,6));
delta_y = abs(beam_coordinate(:,4)-beam_coordinate(:,7));

% ID number of longitudinal and transverse stiffeners
longstiff_ID = find(delta_y<=ey);
transtiff_ID = find(delta_x<=ex);

% Coordinate informations of longitudinal and transverse stiffeners
longstiff_coordinate = beam_coordinate(longstiff_ID,:);
transtiff_coordinate = beam_coordinate(transtiff_ID,:);

% Plate elements ID which have two nodes in the x-y reference plane
N_xy = [plate_coordinate(:,5) == z_ref] + ...
    [plate_coordinate(:,8) == z_ref] + ...
    [plate_coordinate(:,11) == z_ref] + [plate_coordinate(:,14) == z_ref];
plate_id = find(N_xy==2);

% Initialise number of longitudinal and transverse plate elements which
% define girders and stiffeners and their coordinates
N_long = 0;
N_tran = 0;
long_plategirder = [];
tran_plategirder = [];
for i = 1:length(plate_id)
    % Find the nodes of the plate elements which are in the x-y refernece
    % plane
    index = find(abs(plate_coordinate(plate_id(i),[5 8 11 14]))<=ez);
    % Determine whether the element is longitudinal or transverse
    deltaplate_x = abs(plate_coordinate(plate_id(i),index(1)*3) - ...
        plate_coordinate(plate_id(i),index(2)*3));
    deltaplate_y = abs(plate_coordinate(plate_id(i),index(1)*3+1) - ...
        plate_coordinate(plate_id(i),index(2)*3+1));
    % Longitudinal element
    if deltaplate_y < deltaplate_x
        N_long = N_long+1;
        long_plategirder(N_long,1) = plate_id(i);
        long_plategirder(N_long,2:7) = plate_coordinate(plate_id(i), ...
            [index(1)*3 index(1)*3+1 index(1)*3+2 index(2)*3 ...
            index(2)*3+1 index(2)*3+2]);
    % Transverse element
    elseif deltaplate_y > deltaplate_x
        N_tran = N_tran+1;
        tran_plategirder(N_tran,1) = plate_id(i);
        tran_plategirder(N_tran,2:7) = plate_coordinate(plate_id(i), ...
            [index(1)*3 index(1)*3+1 index(1)*3+2 index(2)*3 ...
            index(2)*3+1 index(2)*3+2]);
```

```
    end
end

% Group A — Nodes attached to longitudinal stiffeners
node_Abeam = [];
node_Aplate = [];
% Find the nodes attached to longitudinal beam elements
if isempty(longstiff_coordinate) == 0
    node_1beam = [longstiff_coordinate(:,3) longstiff_coordinate(:,4) ...
        longstiff_coordinate(:,5)];
    node_2beam = [longstiff_coordinate(:,6) longstiff_coordinate(:,7) ...
        longstiff_coordinate(:,8)];
    node_Abeam = union(node_1beam,node_2beam,'rows');
end

% Find the nodes attached to longitudinal plate elements
if isempty(long_plategirder) == 0
    node_1plate = [long_plategirder(:,2) long_plategirder(:,3) ...
        long_plategirder(:,4)];
    node_2plate = [long_plategirder(:,5) long_plategirder(:,6) ...
        long_plategirder(:,7)];
    node_Aplate = union(node_1plate,node_2plate,'rows');
end

% Occurrence of both longitudinal beam and plate elements
if isempty(node_Abeam) == 0 && isempty(node_Aplate) == 0
    node_A = union(node_Abeam,node_Aplate,'rows');
    node_Adiffbeam = [setdiff(node_Abeam,node_1beam,'rows')
        setdiff(node_Abeam,node_2beam,'rows')];
    % Nodes at ends
    node_Adiffplate = [setdiff(node_Aplate,node_1plate,'rows')
        setdiff(node_Aplate,node_2plate,'rows')];
    node_Adiff = union(node_Adiffbeam,node_Adiffplate,'rows');
% Occurrence of only longitudinal beam elements
elseif isempty(node_Abeam) == 0 && isempty(node_Aplate) == 1
    node_A = node_Abeam;
    % Nodes at ends
    node_Adiff = [setdiff(node_Abeam,node_1beam,'rows')
        setdiff(node_Abeam,node_2beam,'rows')];
% Occurrence of only longitudinal plate elements
elseif isempty(node_Abeam) == 1 && isempty(node_Aplate) == 0
    node_A = node_Aplate;
    % Nodes at ends
    node_Adiff = [setdiff(node_Aplate,node_1plate,'rows')
        setdiff(node_Aplate,node_2plate,'rows')];
end

% Group B — Nodes attached to transverse stiffeners
node_Bbeam = [];
node_Bplate = [];
% Find the nodes attached to transverse beam elements
if isempty(transtiff_coordinate) == 0
    node_1beam = [transtiff_coordinate(:,3) transtiff_coordinate(:,4) ...
        transtiff_coordinate(:,5)];
    node_2beam = [transtiff_coordinate(:,6) transtiff_coordinate(:,7) ...
        transtiff_coordinate(:,8)];
    node_Bbeam = union(node_1beam,node_2beam,'rows');
end

% Find the nodes attached to transverse plate elements
if isempty(tran_plategirder) == 0
    node_1plate = [tran_plategirder(:,2) tran_plategirder(:,3) ...
        tran_plategirder(:,4)];
    node_2plate = [tran_plategirder(:,5) tran_plategirder(:,6) ...
        tran_plategirder(:,7)];
    node_Bplate = union(node_1plate,node_2plate,'rows');
end

% Occurrence of both transverse beam and plate elements
```

```
if isempty(node_Bbeam) == 0 && isempty(node_Bplate) == 0
    node_B = union(node_Bbeam,node_Bplate,'rows');
    % Nodes at ends
    node_Bdiffbeam = [setdiff(node_Bbeam,node_1beam,'rows')
        setdiff(node_Bbeam,node_2beam,'rows')];
    node_Bdiffplate = [setdiff(node_Bplate,node_1plate,'rows')
        setdiff(node_Bplate,node_2plate,'rows')];
    node_Bdiff = union(node_Bdiffbeam,node_Bdiffplate,'rows');
% Occurrence of only transverse beam elements
elseif isempty(node_Bbeam) == 0 && isempty(node_Bplate) == 1
    node_B = node_Bbeam;
    % Nodes at ends
    node_Bdiff = [setdiff(node_Bbeam,node_1beam,'rows')
        setdiff(node_Bbeam,node_2beam,'rows')];
% Occurrence of only transverse plate elements
elseif isempty(node_Bbeam) == 1 && isempty(node_Bplate) == 0
    node_B = node_Bplate;
    % Nodes at ends
    node_Bdiff = [setdiff(node_Bplate,node_1plate,'rows')
        setdiff(node_Bplate,node_2plate,'rows')];
end

% Group C - Nodes which belong to both Group A and Group B
node_AB = intersect(node_A,node_B,'rows');

% Free longitudinal and transeverse nodes
free_long = setdiff(node_Adiff,node_AB,'rows');
free_trans = setdiff(node_Bdiff,node_AB,'rows');
free_nodes = union(free_trans,free_long,'rows');

%% Define longitudinal stiffeners
% Nodes belonging to longitudinal stiffeners
long_nodes = sortrows(union(node_AB,free_long,'rows'),2);
n = size(long_nodes);

% Connect the longitudinal nodes
k = 0;
for i = 1:n(1)-1
    C1 = long_nodes(i,:);
    if long_nodes(i+1,2) == C1(2) % If same y-coordinate
        C2 = long_nodes(i+1,:);
        k = k+1;
        % long_stiffener(ID,x1,y1,z1,x2,y2,z2)
        long_stiffener(k,1) = k;
        long_stiffener(k,2:7) = [C1 C2];
    end
end

%% Connect beam element properties to longitudinal stiffeners
% Total number of FE longitudinal elements
n1 = size(longstiff_coordinate);
n2 = size(long_plategirder);
n = n1(1)+n2(1);

% element_centre(element_ID,xcentre,ycentre,zcentre,type(1=beam,2=plate))
element_centre = zeros(n,5);
element_centre(1:n1(1),1) = longstiff_coordinate(:,1);
element_centre(1:n1(1),5) = 1;
if isempty(long_plategirder) == 0
    element_centre(n1(1)+1:n,5) = 2;
    element_centre(n1(1)+1:n,1) = long_plategirder(:,1);
end

% Centre of longitudinal beam elements
for i = 1:n1(1)
    element_centre(i,2:4) = [sum(longstiff_coordinate(i,3:3:6))/2 ...
        sum(longstiff_coordinate(i,4:3:7))/2 ...
        sum(longstiff_coordinate(i,5:3:8))/2];
end
```

```
% Centre of longitudinal plate elements
for i = 1:n2(1)
    element_centre(i+n1(1),2:4) = [sum(long_plategirder(i,2:3:5))/2 ...
        sum(long_plategirder(i,3:3:6))/2 ...
        sum(long_plategirder(i,4:3:7))/2];
end

% Find the coordinates limits of the the longitudinal elements
n = size(long_stiffener);
N_longstiff = n(1);
% longstiff_limit(longstiff_ID,xmin,xmax,ymin,ymax,zmin,zmax)
longstiff_limit = zeros(N_longstiff,7);
longstiff_limit(:,1) = 1:sum(N_longstiff);
for i = 1:N_longstiff
    longstiff_limit(i,2) = min(long_stiffener(i,[2 5]));
    longstiff_limit(i,3) = max(long_stiffener(i,[2 5]));
    longstiff_limit(i,4) = min(long_stiffener(i,[3 6]));
    longstiff_limit(i,5) = max(long_stiffener(i,[3 6]));
    longstiff_limit(i,6) = min(long_stiffener(i,[4 7]));
    longstiff_limit(i,7) = max(long_stiffener(i,[4 7]));
end

% Connect the FE elements to the stiffener elements
% element_longstiff(element_ID,longstiff_ID,type)
n = n1(1)+n2(1);
element_longstiff = zeros(n(1),3);
element_longstiff(:,[1 3]) = element_centre(:,[1 5]);
for i = 1:n(1)
    for j = 1:sum(N_longstiff)
        a = element_centre(i,2)>=longstiff_limit(j,2) & ...
            element_centre(i,2)<=longstiff_limit(j,3);
        b = element_centre(i,3)>=longstiff_limit(j,4) & ...
            element_centre(i,3)<=longstiff_limit(j,5);
        if a == 1 && b == 1
            element_longstiff(i,2) = j;
        end
    end
end

% Find the ID's of the actual longitudinal stiffener elements, here
% previous defined longitudinal stiffener elements which no FE elements
% belong to are eliminated
Nr_longstiff = unique(sort(element_longstiff(:,2)));
real_longstiff(:,1) = 1:length(Nr_longstiff);
real_longstiff(:,2:7) = long_stiffener(Nr_longstiff,2:7);
n = size(real_longstiff);

% Plot the longitudinal elements
figure
hold on
for i = 1:n(1)
    plot3(real_longstiff(i,[2 5]), ...
        real_longstiff(i,[3 6]), ...
        real_longstiff(i,[4 7]), ...
        'k','LineWidth',2)
    text((real_longstiff(i,2)+real_longstiff(i,5))/2, ...
        (real_longstiff(i,3)+real_longstiff(i,6))/2, ...
        (real_longstiff(i,4)+real_longstiff(i,7))/2, ...
        num2str(real_longstiff(i,1)))
end
axis equal

% Determine the material and geometrical properties of the longitudinal
% stiffeners
longstiff_info = zeros(length(Nr_longstiff),9);
longstiff_info(:,1) = real_longstiff(:,1);
for j = 1:length(Nr_longstiff)
    i = Nr_longstiff(j,1);
```

```
    index = find(element_longstiff(:,2)==i);
    if element_longstiff(index,3) == 1
        t = mean(beam_information(element_longstiff(index),3:9));
    else
        % In case the stiffener is modelled with plate elements the cross
        % section properties cannot be obtained
        t = [mean(plate_information(element_longstiff(index),3:4)) ...
            NaN NaN NaN NaN NaN];
    end
    longstiff_info(j,3:9) = t;
end
longstiff_info(:,2) = real_longstiff(:,5)-real_longstiff(:,2);

%% Define transverse stiffeners
% Nodes belonging to transverse stiffeners
tran_nodes = union(node_AB,free_trans,'rows');
n = size(tran_nodes);

% Connect the longitudinal nodes
k = 0;
for i = 1:n(1)-1
    C1 = tran_nodes(i,:);
    if tran_nodes(i+1,1) == C1(1) % If same x-coordinate
        C2 = tran_nodes(i+1,:);
        k = k+1;
        % tran_stiffener(ID,x1,y1,z1,x2,y2,z2)
        tran_stiffener(k,1) = k;
        tran_stiffener(k,2:7) = [C1 C2];
    end
end

%% Connect beam element properties to transverse stiffeners
% Total number of FE transverse elements
n1 = size(transtiff_coordinate);
n2 = size(tran_plategirder);
n = n1(1)+n2(1);
% element_centre(element_ID,xcentre,ycentre,zcentre,type(1=beam,2=plate))
element_centre = zeros(n,5);
element_centre(1:n1(1),1) = transtiff_coordinate(:,1);
element_centre(1:n1(1),5) = 1;
if isempty(tran_plategirder) == 0
    element_centre(n1(1)+1:n,5) = 2;
    element_centre(n1(1)+1:n,1) = tran_plategirder(:,1);
end

% Centre of transverse beam elements
for i = 1:n1(1)
    element_centre(i,2:4) = [sum(transtiff_coordinate(i,3:3:6))/2 ...
        sum(transtiff_coordinate(i,4:3:7))/2 ...
        sum(transtiff_coordinate(i,5:3:8))/2];
end

% Centre of transverse plate elements
for i = 1:n2(1)
    element_centre(i+n1(1),2:4) = [sum(tran_plategirder(i,2:3:5))/2 ...
        sum(tran_plategirder(i,3:3:6))/2 ...
        sum(tran_plategirder(i,4:3:7))/2];
end

% Find the coordinates limits of the the transverse elements
n = size(tran_stiffener);
N_transtiff = n(1);
% transtiff_limit(transtiff_ID,xmin,xmax,ymin,ymax,zmin,zmax)
transtiff_limit = zeros(N_transtiff,7);
transtiff_limit(:,1) = 1:sum(N_transtiff);
for i = 1:N_transtiff
    transtiff_limit(i,2) = min(tran_stiffener(i,[2 5]));
    transtiff_limit(i,3) = max(tran_stiffener(i,[2 5]));
    transtiff_limit(i,4) = min(tran_stiffener(i,[3 6]));
```

```
    transtiff_limit(i,5) = max(tran_stiffener(i,[3 6]));
    transtiff_limit(i,6) = min(tran_stiffener(i,[4 7]));
    transtiff_limit(i,7) = max(tran_stiffener(i,[4 7]));
end

% Connect the FE elements to the stiffener/girder elements
% element_transtiff(element_ID,panel_ID,type)
n = n1(1)+n2(1);
element_transtiff = zeros(n(1),3);
element_transtiff(:,[1 3]) = element_centre(:,[1 5]);
for i = 1:n(1)
    for j = 1:sum(N_transtiff)
        a = element_centre(i,2)>=transtiff_limit(j,2) & ...
            element_centre(i,2)<=transtiff_limit(j,3);
        b = element_centre(i,3)>=transtiff_limit(j,4) & ...
            element_centre(i,3)<=transtiff_limit(j,5);
        if a == 1 && b == 1
            element_transtiff(i,2) = j;
        end
    end
end

% Find the ID's of the actual transverse stiffener elements, here the
% previous defined transverse stiffener/girder elements which no FE
% elements belong to are eliminated
Nr_transtiff = unique(sort(element_transtiff(:,2)));
real_transtiff(:,1) = 1:length(Nr_transtiff);
real_transtiff(:,2:7) = tran_stiffener(Nr_transtiff,2:7);
n = size(real_transtiff);

% Plot the transverse elements
figure
hold on
for i = 1:n(1)
    plot3(real_transtiff(i,[2 5]), ...
        real_transtiff(i,[3 6]), ...
        real_transtiff(i,[4 7]), ...
        'k','LineWidth',2)
    text((real_transtiff(i,2)+real_transtiff(i,5))/2, ...
        (real_transtiff(i,3)+real_transtiff(i,6))/2, ...
        (real_transtiff(i,4)+real_transtiff(i,7))/2, ...
        num2str(real_transtiff(i,1)))
end
axis equal

% Determine the material and geometrical properties of the transverse
% stiffeners
transtiff_info = zeros(length(Nr_transtiff),8);
transtiff_info(:,1) = real_transtiff(:,1);
for j = 1:length(Nr_transtiff)
    i = Nr_transtiff(j,1);
    index = find(element_transtiff(:,2)==i);
    if element_transtiff(index,3) == 1
        t = mean(beam_information(element_transtiff(index),3:8));
    else
        % In case the stiffener is modelled with plate elements the cross
        % section properties cannot be obtained
        t = [mean(plate_information(element_transtiff(index),3:4)) ...
            NaN NaN NaN NaN];
    end
    transtiff_info(j,3:8) = t;
end
transtiff_info(:,2) = real_transtiff(:,6)-real_transtiff(:,3);

%% Define plate panel between stiffeners
n = size(real_longstiff);
panel = zeros(n(1)-1,15);
for i = 1:n(1)-1
    C1 = real_longstiff(i,2:4);
```

```
    if any(real_transtiff(:,6)>C1(2))
        C2_test = real_longstiff(i,5:7);
        index3_x = find(real_transtiff(:,2)==C2_test(1));
        index3_y = find(real_transtiff(:,3)==C2_test(2));
        index3 = intersect(index3_x,index3_y);
        k = 0;
        while length(index3) ~= 1
            k = k+1;
            C2_test = real_longstiff(i+k,5:7);
            index3_x = find(real_transtiff(:,2)==C2_test(1));
            index3_y = find(real_transtiff(:,3)==C2_test(2));
            index3 = intersect(index3_x,index3_y);
        end
        C2 = C2_test;
        C3_test = real_transtiff(index3,5:7);
        index4_x = find(real_longstiff(:,5)==C3_test(1));
        index4_y = find(real_longstiff(:,6)==C3_test(2));
        index4 = intersect(index4_x,index4_y);
        k = 0;
        while length(index4) ~= 1
            k = k+1;
            C3_test = real_transtiff(index3+k,5:7);
            index4_x = find(real_longstiff(:,5)==C3_test(1));
            index4_y = find(real_longstiff(:,6)==C3_test(2));
            index4 = intersect(index4_x,index4_y);
        end
        C3 = C3_test;
        C4 = [C1(1) C3(2) 0];
    else
        C2 = [NaN NaN NaN];
        C3 = [NaN NaN NaN];
        C4 = [NaN NaN NaN];
    end
    panel(i,2:13) = [C1 C2 C3 C4];
end

[R C] = find(isnan(panel));
R = unique(R);
panel(R,:) = [];
n = size(panel);
panel(:,1) = 1:n(1);

panel_centre = zeros(n(1),1);
panel_centre(:,1) = panel(:,1);

% panel_limit(panel_ID,xmin,xmax,ymin,ymax,zmin,zmax)
panel_limit = zeros(n(1),7);
panel_limit(:,1) = panel(:,1);

for i = 1:n(1)
    panel_centre(i,2:4) = [sum(panel(i,2:3:11))/4 ...
        sum(panel(i,3:3:12))/4 ...
        sum(panel(i,4:3:13))/4];
    panel_limit(i,2) = min(panel(i,[2 5 8 11]));
    panel_limit(i,3) = max(panel(i,[2 5 8 11]));
    panel_limit(i,4) = min(panel(i,[3 6 9 12]));
    panel_limit(i,5) = max(panel(i,[3 6 9 12]));
    panel_limit(i,6) = min(panel(i,[4 7 10 13]));
    panel_limit(i,7) = max(panel(i,[4 7 10 13]));
end

k = 0;
for i = 1:n(1)
    for j = 1:n(1)
        a = panel_centre(i,2)>=panel_limit(j,2) & ...
            panel_centre(i,2)<=panel_limit(j,3);
        b = panel_centre(i,3)>=panel_limit(j,4) & ...
            panel_centre(i,3)<=panel_limit(j,5);
        if a == 1 && b == 1
```

```
            if j ~= i
                k = k+1;
                combine(k,1) = i;
                combine(k,2) = j;
            end
        end
    end
end

if isempty(combine) == 0
    combine = sortrows(sort(combine,2),2);

    vertical_panel = unique(combine(:,2));
    count1 = 0;
    k = panel(end,1);
    h = 0;
    d = [];
    if length(vertical_panel)>1
        for i =1:length(vertical_panel)-1
            count2 = length(find(combine(:,2)==vertical_panel(i)));
            if isempty(d)
                a = unique(combine(count1+1:count1+count2,:));
            else
                a = d;
            end
            count1 = count1+count2;
            count2 = length(find(combine==vertical_panel(i+1)));
            b = unique(combine(count1+1:count1+count2,:));
            c = intersect(a,b);
            if isempty(c)
                k = k+1;
                h = h+1;
                newpanel_limit(h,1) = k;
                newpanel_limit(h,2) = min(panel_limit(a,2));
                newpanel_limit(h,3) = max(panel_limit(a,3));
                newpanel_limit(h,4) = min(panel_limit(a,4));
                newpanel_limit(h,5) = max(panel_limit(a,5));
                newpanel_limit(h,6) = min(panel_limit(a,6));
                newpanel_limit(h,7) = max(panel_limit(a,7));
                d = [];
            else
                d = union(a,b);
            end
            if i == length(vertical_panel)-1
                if isempty(c) == 0
                    k = k+1;
                    h = h+1;
                    newpanel_limit(h,1) = k;
                    newpanel_limit(h,2) = min(panel_limit(d,2));
                    newpanel_limit(h,3) = max(panel_limit(d,3));
                    newpanel_limit(h,4) = min(panel_limit(d,4));
                    newpanel_limit(h,5) = max(panel_limit(d,5));
                    newpanel_limit(h,6) = min(panel_limit(d,6));
                    newpanel_limit(h,7) = max(panel_limit(d,7));
                else
                    k = k+1;
                    h = h+1;
                    newpanel_limit(h,1) = k;
                    newpanel_limit(h,2) = min(panel_limit(b,2));
                    newpanel_limit(h,3) = max(panel_limit(b,3));
                    newpanel_limit(h,4) = min(panel_limit(b,4));
                    newpanel_limit(h,5) = max(panel_limit(b,5));
                    newpanel_limit(h,6) = min(panel_limit(b,6));
                    newpanel_limit(h,7) = max(panel_limit(b,7));
                end
            end
        end
    else
        a = unique(combine);
```

```
        k = k+1;
        h = h+1;
        newpanel_limit(h,1) = k;
        newpanel_limit(h,2) = min(panel_limit(a,2));
        newpanel_limit(h,3) = max(panel_limit(a,3));
        newpanel_limit(h,4) = min(panel_limit(a,4));
        newpanel_limit(h,5) = max(panel_limit(a,5));
        newpanel_limit(h,6) = min(panel_limit(a,6));
        newpanel_limit(h,7) = max(panel_limit(a,7));
    end

    n = size(newpanel_limit);
    for i = 1:n(1)
        index = panel(end,1)+1;
        panel(index,1) = newpanel_limit(i,1);
        panel(index,[2 11]) = newpanel_limit(i,2);
        panel(index,[5 8]) = newpanel_limit(i,3);
        panel(index,[3 6]) = newpanel_limit(i,4);
        panel(index,[9 12]) = newpanel_limit(i,5);
        panel(index,[4 7]) = newpanel_limit(i,6);
        panel(index,[10 13]) = newpanel_limit(i,7);
    end

    panel(unique(combine),:) = [];
end
panel(:,14) = panel(:,5)-panel(:,2);
panel(:,15) = panel(:,9)-panel(:,6);

%% Connect plate element properties to plate panels
n = size(plate_coordinate);
nn = size(panel);
N_panels = nn(1);
% element_centre(element_ID,xcentre,ycentre,zcentre)
element_centre = zeros(n(1),4);
for i = 1:n(1)
    element_centre(i,:) = [i sum(plate_coordinate(i,3:3:12))/4 ...
        sum(plate_coordinate(i,4:3:13))/4 ...
        sum(plate_coordinate(i,5:3:14))/4];
end

[r c] = find(element_centre(:,end)==0);
test = plate_coordinate(r,2);
% panel_limit(panel_ID,xmin,xmax,ymin,ymax,zmin,zmax)
panel_limit = zeros(N_panels,7);
panel_limit(:,1) = 1:N_panels;
for i = 1:N_panels
    panel_limit(i,2) = min(panel(i,[2 5 8 11]));
    panel_limit(i,3) = max(panel(i,[2 5 8 11]));
    panel_limit(i,4) = min(panel(i,[3 6 9 12]));
    panel_limit(i,5) = max(panel(i,[3 6 9 12]));
    panel_limit(i,6) = min(panel(i,[4 7 10 13]));
    panel_limit(i,7) = max(panel(i,[4 7 10 13]));
end

% element_panel(element_ID,panel_ID)
panelelement_centre = element_centre(r,:);
n = size(panelelement_centre);
element_panel = zeros(n(1),2);
element_panel(:,1) = panelelement_centre(:,1);
for i = 1:n(1)
    for j = 1:N_panels
        a = panelelement_centre(i,2)>panel_limit(j,2) & ...
            panelelement_centre(i,2)<panel_limit(j,3);
        b = panelelement_centre(i,3)>panel_limit(j,4) & ...
            panelelement_centre(i,3)<panel_limit(j,5);
        if a == 1 && b == 1
            element_panel(i,2) = j;
        end
    end
```

```
end

Nr_panels = unique(sort(element_panel(:,2)));
real_panel = panel(Nr_panels,:);
n = size(real_panel);
N_panels = n(1);
real_panel(:,1) = 1:N_panels;

figure
hold on
for i = 1:N_panels
    plot(real_panel(i,[2 5 8 11 2]), ...
        real_panel(i,[3 6 9 12 3]))
    text((real_panel(i,2)+real_panel(i,8))/2, ...
        (real_panel(i,3)+real_panel(i,9))/2, ...
        num2str(real_panel(i,1)))
end
axis equal

panel_info = zeros(N_panels,6);
panel_info(:,1) = real_panel(:,1);
for j = 1:N_panels
    i = Nr_panels(j);
    index = find(element_panel(:,2)==i);
    value = element_panel(index,1);
    t = mean(plate_information(value,3:5));
    panel_info(j,4:6) = t;
end
panel_info(:,2:3) = real_panel(:,14:15);
```

D.5 Stresses.m

```
% Function name: Stresses.m
% Written by: Ottar Hillers, September 2011
% Purpose: To determine design stresses for the Stress Check Model using
% results from Femap
% ------------------------------------------------------------------------
% OUTPUTS:
% modelled_stress(panel_ID,s_x,max,s_x,min,Rx,s_y,max,s_y,min,Ry,s_t,max,q)
% ------------------------------------------------------------------------
%% Location of stresses within elements
% Relate the inital panel relation of the FE elements to the actual panel
% numbering
A = real_panel(:,1);
B = element_panel;
C = [A unique(B(:,2))];

for i = 1:length(A)
    a = C(i,2);
    b = find(B(:,2)==a);
    B(b,2)=C(i,1);
end

sort_panel = sortrows(B,2); % sortrows(element_panel,2);

% index - summation of plat elements belonging to the panels
%         [0 0+N_1 0+N_1+N_2 ... 0+N_1+N_2+...+N_i]
index = zeros(1,N_panels+1);
for i = 1:N_panels
    index(i+1) = find(sort_panel(:,2)==real_panel(i),1,'last');
end

% Stress boundaries, columns represent the panels
%                    1st row - stresses at left edge of the panel
```

```
%                    2nd row — stresses at right edge of the panel
%                    3nd row — stresses at bottom edge of the panel
%                    4th row — stresses at top edge of the panel
%                    5th — 8th row — shear stresses around the boundaries
stress_boundaries = cell(8,N_panels);
edge_coordinate = cell(4,N_panels);
lateral_pressure = cell(1,N_panels);
for i = 1:N_panels
    % Find the ID_fem of the plate elements of each panel within the
    % plate_coordinate
    element_number = plate_coordinate(sort_panel(index(i)+ ...
        1:index(i+1),1),2);

    n = size(element_number);
    R = zeros(n(1),1);
    for t = 1:n(1)
        R(t) = find(plate_coordinate(:,2)==element_number(t));
    end

    %% LEFT EDGE OF PANELS
    % Finds the index of ID_fem of the elements belonging to the left edge
    % of each panel within element_location
    minx = find(plate_coordinate(R,3)== ...
        min(plate_coordinate(R,3)));
    % Finds the actual FEM element ID
    minx_location = element_number(minx,1);
    n = length(minx_location);
    y_left = zeros(2*n,1);
    stresses_left = zeros(2*n,1);
    stresses_shearleft = zeros(2*n,1);
    % For every element on the left edge
    for j = 1:n
        % Real ID
        id = find(plate_coordinate(:,2)== minx_location(j));
        % x coordinates of the element nodes
        x_nodes = plate_coordinate(id,3:3:12);
        % sort the nodes x-coordinate in ascend order
        ascend_x = sort(x_nodes);
        % the two smalles values define the edge
        C_nodes = ascend_x(1:2);
        % Find the corresponding node (1,2,3 or 4)
        C = union(find(x_nodes == C_nodes(1)), ...
            find(x_nodes == C_nodes(2)));
        % y coordinates of the element nodes
        y_left(2*j-1:2*j) = plate_coordinate(id,C*3+1);
        % x stresses at the edge nodes
        stresses_left((2*j-1:2*j)) = -1*stress_information(id,C*3);
        % shear stresses at the edge nodes
        stresses_shearleft((2*j-1:2*j)) = stress_information(id,C*3+2);
    end
    % Assemble y coordinates and stresses at the left edge of each panel
    edge_coordinate{1,i} = y_left;
    stress_boundaries{1,i} = stresses_left;
    stress_boundaries{5,i} = stresses_shearleft;

    %% RIGHT EDGE OF PANELS
    % Finds the index of ID_fem of the elements belonging to the right edge
    % of each panel within element_location
    maxx = find(plate_coordinate(R,6)== ...
        max(plate_coordinate(R,6)));
    % Finds the actual FEM element ID
    maxx_location = element_number(maxx,1);
    n = length(maxx_location);
    y_right = zeros(2*n,1);
    stresses_right = zeros(2*n,1);
    stresses_shearright = zeros(2*n,1);
    % For every element on the right edge
    for j = 1:n
        % Real ID
```

```matlab
    id = find(plate_coordinate(:,2)== maxx_location(j));
    % x coordinates of the element nodes
    x_nodes = plate_coordinate(id,3:3:12);
    % sort the nodes x-coordinate in descend order
    descend_x = sort(x_nodes,'descend');
    % the two largest values define the edge
    C_nodes = descend_x(1:2);
    % Find the corresponding node (1,2,3 or 4)
    C = union(find(x_nodes == C_nodes(1)), ...
        find(x_nodes == C_nodes(2)));
    % y coordinates of the element nodes
    y_right(2*j-1:2*j) = plate_coordinate(id,C*3+1);
    % x stresses at the edge nodes
    stresses_right((2*j-1:2*j)) = -1*stress_information(id,C*3);
    % shear stresses at the edge nodes
    stresses_shearright((2*j-1:2*j)) = stress_information(id,C*3+2);
end
% Assemble y coordinates and stresses at the left edge of each panel
edge_coordinate{2,i} = y_right;
stress_boundaries{2,i} = stresses_right;
stress_boundaries{6,i} = stresses_shearright;

%% BOTTOM EDGE OF PANELS
% Finds the index of ID_fem of the elements belonging to the bottom
% edge of each panel within element_location
miny = find(plate_coordinate(R,4)== ...
    min(plate_coordinate(R,4)));
% Finds the actual FEM element ID
miny_location = element_number(miny,1);
n = length(miny_location);
x_bottom = zeros(2*n,1);
stresses_bottom = zeros(2*n,1);
stresses_shearbottom = zeros(2*n,1);
% For every element on the bottom edge
for j = 1:n
    % Real ID
    id = find(plate_coordinate(:,2)== miny_location(j));
    % y coordinates of the element nodes
    y_nodes = plate_coordinate(id,4:3:13);
    % sort the nodes y-coordinate in ascend order
    ascend_y = sort(y_nodes);
    % the two smalles values define the edge
    C_nodes = ascend_y(1:2);
    % Find the corresponding node (1,2,3 or 4)
    C = union(find(y_nodes == C_nodes(1)), ...
        find(y_nodes == C_nodes(2)));
    % x coordinates of the element nodes
    x_bottom(2*j-1:2*j) = plate_coordinate(id,C*3);
    % y stresses at the edge nodes
    stresses_bottom((2*j-1:2*j)) = -1*stress_information(id,C*3+1);
    % shear stresses at the edge nodes
    stresses_shearbottom((2*j-1:2*j)) = stress_information(id,C*3+2);
end
% Assemble y coordinates and stresses at the left edge of each panel
edge_coordinate{3,i} = x_bottom;
stress_boundaries{3,i} = stresses_bottom;
stress_boundaries{7,i} = stresses_shearbottom;

%% TOP EDGE OF PANELS
% Finds the index of ID_fem of the elements belonging to the top edge
% of each panel within element_location
maxy = find(plate_coordinate(R,10)== ...
    max(plate_coordinate(R,10)));
% Finds the actual FEM element ID
maxy_location = element_number(maxy,1);
n = length(maxy_location);
x_top = zeros(2*n,1);
stresses_top = zeros(2*n,1);
stresses_sheartop = zeros(2*n,1);
```

```
    % For every element on the right edge
    for j = 1:n
        % Real ID
        id = find(plate_coordinate(:,2)== maxy_location(j));
        % x coordinates of the element nodes
        y_nodes = plate_coordinate(id,4:3:13);
        % sort the nodes y-coordinate in descend order
        descend_y = sort(y_nodes,'descend');
        % the two largest values define the edge
        C_nodes = descend_y(1:2);
        % Find the corresponding node (1,2,3 or 4)
        C = union(find(y_nodes == C_nodes(1)), ...
            find(y_nodes == C_nodes(2)));
        % x coordinates of the element nodes
        x_top(2*j-1:2*j) = plate_coordinate(id,C*3);
        % y stresses at the edge nodes
        stresses_top((2*j-1:2*j)) = -1*stress_information(id,C*3+1);
        % shear stresses at the edge nodes
        stresses_sheartop((2*j-1:2*j)) = stress_information(id,C*3+2);
    end
    % Assemble y coordinates and stresses at the left edge of each panel
    edge_coordinate{4,i} = x_top;
    stress_boundaries{4,i} = stresses_top;
    stress_boundaries{8,i} = stresses_sheartop;

    % Lateral pressure
    lateral_pressure{1,i} = stress_information(R,end);

end

%% Actual stresses
actual_stresses = cell(4,N_panels);
max_leftsigma = zeros(N_panels,2);
min_leftsigma = zeros(N_panels,2);
max_rightsigma = zeros(N_panels,2);
min_rightsigma = zeros(N_panels,2);
max_bottomsigma = zeros(N_panels,2);
min_bottomsigma = zeros(N_panels,2);
max_topsigma = zeros(N_panels,2);
min_topsigma = zeros(N_panels,2);

for i = 1:N_panels
    %--------------------------LEFT EDGE-----------------------------------%
    % Sort the stresses at left edge according to ascend y - coordinates
    A = sortrows([edge_coordinate{1,i} stress_boundaries{1,i} ...
        stress_boundaries{5,i}]);
    % Group together the nodes which have the same coordinates
    coordinate = unique(A(:,1));
    % Check if there are nodes with the same coordinate
    diff = size(A)-size(coordinate);
    % If there are elements sharing the same node the average value at that
    % node will be used
    if diff(1) ~= 0
        averaging = zeros(diff(1)+2,2);
        averaging(1,1:2) = A(1,2:3);
        for j = 1:diff(1)
            averaging(j+1,1) = mean([A(2*j,2) A(2*j+1,2)]);
            averaging(j+1,2) = mean([A(2*j,3) A(2*j+1,3)]);
        end
        averaging(end,1:2) = A(end,2:3);
    else
        averaging = A(:,2:3);
    end
    leftedge_stresses = ([coordinate averaging]);
    actual_stresses{1,i} = leftedge_stresses;

    % Values for linear interpolation between corner values
    k = (coordinate(end)-coordinate(1))/(leftedge_stresses(end,2)- ...
        leftedge_stresses(1,2));
```

```
n = size(leftedge_stresses);
sigma_ed = zeros(n(1)-2,1);
for j = 1:n(1)-2
    sigma_ed(j,1) = (coordinate(j+1)-coordinate(1))/k+ ...
        leftedge_stresses(1,2);
end
% If the actual stresses form a concave pattern a linear interpolation
% between the corner values result in conservative values. Else a
% linear regression is made between the stress values and the results
% shifted corresponding to the difference of the maximum stress value
% and the corresponding value from the linera regression
diff = leftedge_stresses(2:end-1,2)-sigma_ed;
if any(diff >= 0)
    % Linear regression
    x = leftedge_stresses(:,1); % Coordinates
    y = leftedge_stresses(:,2); % Stress values
    % Factors
    Sx = sum(x);
    Sy = sum(y);
    Sxx = sum(x.^2);
    Syy = sum(y.^2);
    Sxy = sum(x.*y);
    % Slope and intersection point
    n = length(x);
    b = (n*Sxy-Sx*Sy)/(n*Sxx-Sx^2);
    a = 1/n*Sy-b/n*Sx;
    X = x;
    % Regression line
    Y = a+b*X;
    % Difference of the regression line and stress results
    K = Y-leftedge_stresses(:,2);
    % Updated conservative stress pattern
    new_stress = Y-min(K);
    leftsigma = [coordinate(1) new_stress(1);
        coordinate(end) new_stress(end)];
    % Max stress at left edge
    [Rmax,Cmax] = find(leftsigma == ...
        max([leftsigma(1,2) leftsigma(2,2)]));
    % Min stress at left edge
    [Rmin,Cmin] = find(leftsigma == ...
        min([leftsigma(1,2) leftsigma(2,2)]));
    max_leftsigma(i,:) = leftsigma(Rmax,:);
    min_leftsigma(i,:) = leftsigma(Rmin,:);
else
    % Max stress at left edge
    [Rmax,Cmax] = find(leftedge_stresses == ...
        max([leftedge_stresses(1,2) leftedge_stresses(end,2)]));
    max_leftsigma(i,:) = leftedge_stresses(Rmax,1:2);
    % Min stress at left edge
    [Rmin,Cmin] = find(leftedge_stresses == ...
        min([leftedge_stresses(1,2) leftedge_stresses(end,2)]));
    min_leftsigma(i,:) = leftedge_stresses(Rmin,1:2);
end

%-------------------------RIGHT EDGE----------------------------------%
% Sort the stresses at right edge according to ascend y - coordinates
A = sortrows([edge_coordinate{2,i} stress_boundaries{2,i}]);
% Group together the nodes which have the same coordinates
coordinate = unique(A(:,1));
% Check if there are nodes with the same coordinate
diff = size(A)-size(coordinate);
% If there are elements sharing the same node the average value at that
% node will be used
if diff(1) ~= 0
    averaging = zeros(diff(1)+2,1);
    averaging(1) = A(1,2);
    for j = 1:diff(1)
        averaging(j+1) = mean([A(2*j,2) A(2*j+1,2)]);
    end
```

```
    averaging(end) = A(end,2);
else
    averaging = A(:,2);
end
rightedge_stresses = ([coordinate averaging]);
actual_stresses{2,i} = rightedge_stresses;

% Values for linear interpolation between corner values
k = (coordinate(end)-coordinate(1))/(rightedge_stresses(end,2)- ...
    rightedge_stresses(1,2));
n = size(rightedge_stresses);
sigma_ed = zeros(n(1)-2,1);
for j = 1:n(1)-2
    sigma_ed(j,1) = (coordinate(j+1)-coordinate(1))/k+ ...
        rightedge_stresses(1,2);
end
% If the actual stresses form a concave pattern a linear interpolation
% between the corner values result in conservative values. Else a
% linear regression is made between the stress values and the results
% shifted corresponding to the difference of the maximum stress value
% and the corresponding value from the linera regression
diff = rightedge_stresses(2:end-1,2)-sigma_ed;
if any(diff >= 0)
    x = rightedge_stresses(:,1); % Coordinates
    y = rightedge_stresses(:,2); % Stress values
    % Factors
    Sx = sum(x);
    Sy = sum(y);
    Sxx = sum(x.^2);
    Syy = sum(y.^2);
    Sxy = sum(x.*y);
    % Slope and intersection point
    n = length(x);
    b = (n*Sxy-Sx*Sy)/(n*Sxx-Sx^2);
    a = 1/n*Sy-b/n*Sx;
    X = x;
    % Regression line
    Y = a+b*X;
    % Difference of the regression line and stress results
    K = Y-rightedge_stresses(:,2);
    % Updated conservative stress pattern
    new_stress = Y-min(K);
    rightsigma = [coordinate(1) new_stress(1);
        coordinate(end) new_stress(end)];
    % Max stress at left edge
    [Rmax,Cmax] = find(rightsigma == ...
        max([rightsigma(1,2) rightsigma(2,2)]));
    % Min stress at left edge
    [Rmin,Cmin] = find(rightsigma == ...
        min([rightsigma(1,2) rightsigma(2,2)]));
    max_rightsigma(i,:) = rightsigma(Rmax,:);
    min_rightsigma(i,:) = rightsigma(Rmin,:);
else
    % Max stress at right edge
    [Rmax,Cmax] = find(rightedge_stresses == ...
        max([rightedge_stresses(1,2) rightedge_stresses(end,2)]));
    max_rightsigma(i,:) = rightedge_stresses(Rmax,1:2);
    % Min stress at right edge
    [Rmin,Cmin] = find(rightedge_stresses == ...
        min([rightedge_stresses(1,2) rightedge_stresses(end,2)]));
    min_rightsigma(i,:) = rightedge_stresses(Rmin,1:2);
end

%--------------------------BOTTOM EDGE-----------------------------------%
% Sort the stresses at bottom edge according to ascend x - coordinates
A = sortrows([edge_coordinate{3,i} stress_boundaries{3,i}]);
% Group together the nodes which have the same coordinates
coordinate = unique(A(:,1));
% Check if there are nodes with the same coordinate
```

```
diff = size(A)-size(coordinate);
% If there are elements sharing the same node the average value at that
% node will be used
if diff(1) ~= 0
    averaging = zeros(diff(1)+2,1);
    averaging(1) = A(1,2);
    for j = 1:diff(1)
        averaging(j+1) = mean([A(2*j,2) A(2*j+1,2)]);
    end
    averaging(end) = A(end,2);
else
    averaging = A(:,2);
end
bottomedge_stresses = ([coordinate averaging]);
actual_stresses{3,i} = bottomedge_stresses;

% Values for linear interpolation between corner values
k = (coordinate(end)-coordinate(1))/(bottomedge_stresses(end,2)- ...
    bottomedge_stresses(1,2));
n = size(bottomedge_stresses);
sigma_ed = zeros(n(1)-2,1);
for j = 1:n(1)-2
    sigma_ed(j,1) = (coordinate(j+1)-coordinate(1))/k+ ...
        bottomedge_stresses(1,2);
end
% If the actual stresses form a concave pattern a linear interpolation
% between the corner values result in conservative values. Else a
% linear regression is made between the stress values and the results
% shifted corresponding to the difference of the maximum stress value
% and the corresponding value from the linera regression
diff = bottomedge_stresses(2:end-1,2)-sigma_ed;
if any(diff >= 0)
    x = bottomedge_stresses(:,1); % Coordinates
    y = bottomedge_stresses(:,2); % Stress values
    % Factors
    Sx = sum(x);
    Sy = sum(y);
    Sxx = sum(x.^2);
    Syy = sum(y.^2);
    Sxy = sum(x.*y);
    % Slope and intersection point
    n = length(x);
    b = (n*Sxy-Sx*Sy)/(n*Sxx-Sx^2);
    a = 1/n*Sy-b/n*Sx;
    X = x;
    % Regression line
    Y = a+b*X;
    % Difference of the regression line and stress results
    K = Y-bottomedge_stresses(:,2);
    % Updated conservative stress pattern
    new_stress = Y-min(K);
    bottomsigma = [coordinate(1) new_stress(1);
        coordinate(end) new_stress(end)];
    % Max stress at bottom edge
    [Rmax,Cmax] = find(bottomsigma == ...
        max([bottomsigma(1,2) bottomsigma(2,2)]));
    % Min stress at bottom edge
    [Rmin,Cmin] = find(bottomsigma == ...
        min([bottomsigma(1,2) bottomsigma(2,2)]));
    max_bottomsigma(i,:) = bottomsigma(Rmax,:);
    min_bottomsigma(i,:) = bottomsigma(Rmin,:);
else
    % Max stress at bottom edge
    [Rmax,Cmax] = find(bottomedge_stresses == ...
        max([bottomedge_stresses(1,2) bottomedge_stresses(end,2)]));
    max_bottomsigma(i,:) = bottomedge_stresses(Rmax,1:2);
    % Min stress at bottom edge
    [Rmin,Cmin] = find(bottomedge_stresses == ...
        min([bottomedge_stresses(1,2) bottomedge_stresses(end,2)]));
```

```
    min_bottomsigma(i,:) = bottomedge_stresses(Rmin,1:2);
end

%--------------------------TOP EDGE-----------------------------------%
% Sort the stresses at top edge according to ascend y - coordinates
A = sortrows([edge_coordinate{4,i} stress_boundaries{4,i}]);
% Group together the nodes which have the same coordinates
coordinate = unique(A(:,1));
% Check if there are nodes with the same coordinate
diff = size(A)-size(coordinate);
% If there are elements sharing the same node the averag value at that
% node will be used
if diff(1) ~= 0
    averaging = zeros(diff(1)+2,1);
    averaging(1) = A(1,2);
    for j = 1:diff(1)
        averaging(j+1) = mean([A(2*j,2) A(2*j+1,2)]);
    end
    averaging(end) = A(end,2);
else
    averaging = A(:,2);
end
topedge_stresses = ([coordinate averaging]);
actual_stresses{4,i} = topedge_stresses;

% Values for linear interpolation between corner values
k = (coordinate(end)-coordinate(1))/(topedge_stresses(end,2)- ...
    topedge_stresses(1,2));
n = size(topedge_stresses);
sigma_ed = zeros(n(1)-2,1);
for j = 1:n(1)-2
    sigma_ed(j,1) = (coordinate(j+1)-coordinate(1))/k+ ...
        topedge_stresses(1,2);
end
% If the actual stresses form a concave pattern a linear interpolation
% between the corner values result in conservative values. Else a
% linear regression is made between the stress values and the results
% shifted corresponding to the difference of the maximum stress value
% and the corresponding value from the linera regression
diff = topedge_stresses(2:end-1,2)-sigma_ed;
if any(diff >= 0)

    x = topedge_stresses(:,1); % Coordinates
    y = topedge_stresses(:,2); % Stress values
    % Factors
    Sx = sum(x);
    Sy = sum(y);
    Sxx = sum(x.^2);
    Syy = sum(y.^2);
    Sxy = sum(x.*y);
    % Slope and intersection point
    n = length(x);
    b = (n*Sxy-Sx*Sy)/(n*Sxx-Sx^2);
    a = 1/n*Sy-b/n*Sx;
    X = x;
    % Regression line
    Y = a+b*X;
    % Difference of the regression line and stress results
    K = Y-topedge_stresses(:,2);
    % Updated conservative stress pattern
    new_stress = Y-min(K);
    topsigma = [coordinate(1) new_stress(1);
        coordinate(end) new_stress(end)];
    newsigma_1 = zeros(n(1)-1,1);
    % Max stress at top edge
    [Rmax,Cmax] = find(topsigma == ...
        max([topsigma(1,2) topsigma(2,2)]));
    % Min stress at top edge
    [Rmin,Cmin] = find(topsigma == ...
```

```
            min([topsigma(1,2) topsigma(2,2)]));
        max_topsigma(i,:) = topsigma(Rmax,:);
        min_topsigma(i,:) = topsigma(Rmin,:);
    else
        % Max stress at top edge
        [Rmax,Cmax] = find(topedge_stresses == ...
            max([topedge_stresses(1,2) topedge_stresses(end,2)]));
        max_topsigma(i,:) = topedge_stresses(Rmax,1:2);
        % Min stress at top edge
        [Rmin,Cmin] = find(topedge_stresses == ...
            min([topedge_stresses(1,2) topedge_stresses(end,2)]));
        min_topsigma(i,:) = topedge_stresses(Rmin,1:2);
    end
end

%% Modelled stress distribution
% modelled_stress(panel_ID,s_x,max,s_x,min,Rx,s_y,max,s_y,min,Ry,s_t,max,q)
modelled_stress = zeros(N_panels,9);
modelled_stress(1:end,1) = panel_info(:,1);
modelled_location = zeros(N_panels,4);

% Sort the corner stress values into matricies
stress_left = [max_leftsigma min_leftsigma];
stress_right = [max_rightsigma min_rightsigma];
stress_bottom = [max_bottomsigma min_bottomsigma];
stress_top = [max_topsigma min_topsigma];

for i = 1:N_panels

    % a = length of the panel, b = width of the panel
    a = panel_info(i,2);
    b = panel_info(i,3);
    L_stress = min(0.4*a,0.5*b);

    % x direction - top stress
    [R_left C_left] = max(stress_left(i,1:2:end));
    [R_right C_right] = max(stress_right(i,1:2:end));

    sigma_max = max([stress_left(i,2*C_left) stress_right(i,2*C_right)]);
    sigma_min = min([stress_left(i,2*C_left) stress_right(i,2*C_right)]);
    if inplane_settings == true
        % Eurocode 3 part 1-5 clause 4.6.(3)
        sigma_top = sigma_max-L_stress/a*(sigma_max-sigma_min);
    else
        % Maximum stresses at edges
        sigma_top = sigma_max;
    end

    % x direction - bottom stress
    [R_left C_left] = min(stress_left(i,1:2:end));
    [R_right C_right] = min(stress_right(i,1:2:end));

    sigma_max = max([stress_left(i,2*C_left) stress_right(i,2*C_right)]);
    sigma_min = min([stress_left(i,2*C_left) stress_right(i,2*C_right)]);
    if inplane_settings == true
        % Eurocode 3 part 1-5 clause 4.6.(3)
        sigma_bot = sigma_max-L_stress/a*(sigma_max-sigma_min);
    else
        % Maximum stresses at edges
        sigma_bot = sigma_max;
    end

    % max and min x-stresses
    X_max = max([sigma_bot sigma_top]);
    X_min = min([sigma_bot sigma_top]);
    if X_max<0 && X_min<0
        Rx = NaN;
    else
        Rx = X_min/X_max;
```

```
end
modelled_stress(i,2:4) = [X_max X_min Rx];
modelled_location(i,1:2) = [sigma_top sigma_bot];

% y direction - left stress
[R_bottom C_bottom] = min(stress_bottom(i,1:2:end));
[R_top C_top] = min(stress_top(i,1:2:end));

sigma_max = max([stress_bottom(i,2*C_bottom) stress_top(i,2*C_top)]);
sigma_min = min([stress_bottom(i,2*C_bottom) stress_top(i,2*C_top)]);
if inplane_settings == true
    % Eurocode 3 part 1-5 clause 4.6.(3)
    sigma_left = sigma_max-L_stress/b*(sigma_max-sigma_min);
else
    % Maximum stresses at edges
    sigma_left = sigma_max;
end

% y direction - right stress
[R_bottom C_bottom] = max(stress_bottom(i,1:2:end));
[R_top C_top] = max(stress_top(i,1:2:end));

sigma_max = max([stress_bottom(i,2*C_bottom) stress_top(i,2*C_top)]);
sigma_min = min([stress_bottom(i,2*C_bottom) stress_top(i,2*C_top)]);
if inplane_settings == true
    % Eurocode 3 part 1-5 clause 4.6.(3)
    sigma_right = sigma_max-L_stress/b*(sigma_max-sigma_min);
else
    % Maximum stresses at edges
    sigma_right = sigma_max;
end

% max and min y-stresses
Y_max = max([sigma_left sigma_right]);
Y_min = min([sigma_left sigma_right]);
if Y_max<0 && Y_min<0
    Ry = NaN;
else
    Ry = Y_min/Y_max;
end
modelled_stress(i,5:7) = [Y_max Y_min Ry];
modelled_location(i,3:4) = [sigma_right sigma_left];

% Shear stresses
if shear_settings == 0
    % Maximum values
    % Max and min shear stress at left edge
    Min_L = min(stress_boundaries{5,i});
    Max_L = max(stress_boundaries{5,i});
    % Max and min shear stress at right edge
    Min_R = min(stress_boundaries{6,i});
    Max_R = max(stress_boundaries{6,i});
    % Max and min shear stress at bottom edge
    Min_B = min(stress_boundaries{7,i});
    Max_B = max(stress_boundaries{7,i});
    % Max and min shear stress at top edge
    Min_T = min(stress_boundaries{8,i});
    Max_T = max(stress_boundaries{8,i});
    % Max shear stress
    S = [Min_L Max_L Min_R Max_R Min_B Max_B Min_T Max_T];
    modelled_stress(i,8) = max(abs(S));
elseif shear_settings == 1
    % Average absolute values
    % Average shear stress at left edge
    Left_coordinate = edge_coordinate{1,i};
    Left_value = abs(stress_boundaries{5,i});
    n = size(Left_coordinate);
    weighted_stress = zeros(n(1)/2,1);
    for j = 1:n(1)/2
```

```
        weight = abs(Left_coordinate(2*j)-Left_coordinate(2*j-1))/ ...
            panel_info(i,3);
        value = mean(Left_value(2*j-1:2*j));
        weighted_stress(j) = weight*value;
    end
    av_L = sum(weighted_stress);

    % Average shear stress at right edge
    Right_coordinate = edge_coordinate{2,i};
    Right_value = abs(stress_boundaries{6,i});
    n = size(Right_coordinate);
    weighted_stress = zeros(n(1)/2,1);
    for j = 1:n(1)/2
        weight = abs(Right_coordinate(2*j)- ...
            Right_coordinate(2*j-1))/panel_info(i,3);
        value = mean(Right_value(2*j-1:2*j));
        weighted_stress(j) = weight*value;
    end
    av_R = sum(weighted_stress);

    % Average shear stress at bottom edge
    Bottom_coordinate = edge_coordinate{3,i};
    Bottom_value = abs(stress_boundaries{7,i});
    n = size(Bottom_coordinate);
    weighted_stress = zeros(n(1)/2,1);
    for j = 1:n(1)/2
        weight = abs(Bottom_coordinate(2*j)- ...
            Bottom_coordinate(2*j-1))/panel_info(i,2);
        value = mean(Bottom_value(2*j-1:2*j));
        weighted_stress(j) = weight*value;
    end
    av_B = sum(weighted_stress);

    % Average shear stress at top edge
    Top_coordinate = edge_coordinate{4,i};
    Top_value = abs(stress_boundaries{8,i});
    n = size(Top_coordinate);
    weighted_stress = zeros(n(1)/2,1);
    for j = 1:n(1)/2
        weight = abs(Top_coordinate(2*j)-Top_coordinate(2*j-1))/ ...
            panel_info(i,2);
        value = mean(Top_value(2*j-1:2*j));
        weighted_stress(j) = weight*value;
    end
    av_T = sum(weighted_stress);

    % Total circumference of the panel
    Circ = 2*panel_info(i,2)+2*panel_info(i,3);
    % Weighted shear values along the edges
    S = sum([av_L*panel_info(i,3) av_R*panel_info(i,3) ...
        av_B*panel_info(i,2) av_T*panel_info(i,2)]);
        modelled_stress(i,8) = S/Circ;
elseif shear_settings == 2
    % Average values
    % Average shear stress at left edge
    Left_coordinate = edge_coordinate{1,i};
    Left_value = stress_boundaries{5,i};
    n = size(Left_coordinate);
    weighted_stress = zeros(n(1)/2,1);
    for j = 1:n(1)/2
        weight = abs(Left_coordinate(2*j)-Left_coordinate(2*j-1))/ ...
            panel_info(i,3);
        value = mean(Left_value(2*j-1:2*j));
        weighted_stress(j) = weight*value;
    end
    av_L = sum(weighted_stress);

    % Average shear stress at right edge
    Right_coordinate = edge_coordinate{2,i};
```

```
        Right_value = stress_boundaries{6,i};
        n = size(Right_coordinate);
        weighted_stress = zeros(n(1)/2,1);
        for j = 1:n(1)/2
            weight = abs(Right_coordinate(2*j)- ...
                Right_coordinate(2*j-1))/panel_info(i,3);
            value = mean(Right_value(2*j-1:2*j));
            weighted_stress(j) = weight*value;
        end
        av_R = sum(weighted_stress);

        % Average shear stress at bottom edge
        Bottom_coordinate = edge_coordinate{3,i};
        Bottom_value = stress_boundaries{7,i};
        n = size(Bottom_coordinate);
        weighted_stress = zeros(n(1)/2,1);
        for j = 1:n(1)/2
            weight = abs(Bottom_coordinate(2*j)- ...
                Bottom_coordinate(2*j-1))/panel_info(i,2);
            value = mean(Bottom_value(2*j-1:2*j));
            weighted_stress(j) = weight*value;
        end
        av_B = sum(weighted_stress);

        % Average shear stress at top edge
        Top_coordinate = edge_coordinate{4,i};
        Top_value = stress_boundaries{8,i};
        n = size(Top_coordinate);
        weighted_stress = zeros(n(1)/2,1);
        for j = 1:n(1)/2
            weight = abs(Top_coordinate(2*j)- ...
                Top_coordinate(2*j-1))/panel_info(i,2);
            value = mean(Top_value(2*j-1:2*j));
            weighted_stress(j) = weight*value;
        end
        av_T = sum(weighted_stress);

        % Total circumference of the panel
        Circ = 2*panel_info(i,2)+2*panel_info(i,3);
        % Weighted shear values along the edges
        S = sum([av_L*panel_info(i,3) av_R*panel_info(i,3) ...
            av_B*panel_info(i,2) av_T*panel_info(i,2)]);
            modelled_stress(i,8) = S/Circ;
    end

    % Lateral pressure
    if lateral_settings == true
        pressure = lateral_pressure{:,i};
        modelled_stress(i,9) = mean(pressure);
    else
        pressure = lateral_pressure{:,i};
        modelled_stress(i,9) = max(pressure);
    end
end

%% Plots
for i = 1:N_panels
    if plot_planestresses == true
        figure
        subplot(3,3,2)
        actual_values = actual_stresses{4,i};
        area(actual_values(:,1),actual_values(:,2))
        hold on
        plot([max_topsigma(i,1) min_topsigma(i,1)], ...
            [max_topsigma(i,2) min_topsigma(i,2)],'LineWidth',2)
        plot(sort([max_topsigma(i,1) min_topsigma(i,1)],'descend'), ...
            modelled_location(i,3:4),'--r','LineWidth',2)
        colormap cool
        axis tight
```

```
        subplot(3,3,4)
        actual_values = actual_stresses{1,i};
        area(actual_values(:,1),actual_values(:,2))
        hold on
        plot([max_leftsigma(i,1) min_leftsigma(i,1)], ...
            [max_leftsigma(i,2) min_leftsigma(i,2)],'LineWidth',2)
        plot(sort([max_leftsigma(i,1) min_leftsigma(i,1)],'descend'), ...
            modelled_location(i,1:2),'--r','LineWidth',2)
        colormap cool
        view(-90,90)
        axis tight

        j = panel_info(i,1);
        subplot(3,3,5)
        textstr(1) = {['panel - ',num2str(j)]};
        textstr(2) = {'In plane stresses'};
        text((min(panel(i,2:3:11))+max(panel(i,2:3:11)))/2, ...
            (min(panel(i,3:3:12))+max(panel(i,3:3:12)))/2,textstr)
        axis([min(panel(i,2:3:11)) max(panel(i,2:3:11)) ...
            min(panel(i,3:3:12)) max(panel(i,3:3:12))])

        subplot(3,3,6)
        actual_values = actual_stresses{2,i};
        area(actual_values(:,1),actual_values(:,2))
        hold on
        plot([max_rightsigma(i,1) min_rightsigma(i,1)], ...
            [max_rightsigma(i,2) min_rightsigma(i,2)],'LineWidth',2)
        plot(sort([max_rightsigma(i,1) min_rightsigma(i,1)],'descend'), ...
            modelled_location(i,1:2),'--r','LineWidth',2)
        colormap cool
        view(90,-90)
        axis tight

        subplot(3,3,8)
        actual_values = actual_stresses{3,i};
        area(actual_values(:,1),actual_values(:,2))
        hold on
        plot([max_bottomsigma(i,1) min_bottomsigma(i,1)], ...
            [max_bottomsigma(i,2) min_bottomsigma(i,2)],'LineWidth',2)
        plot(sort([max_bottomsigma(i,1) ...
            min_bottomsigma(i,1)],'descend'),modelled_location(i,3:4), ...
            '--r','LineWidth',2)
        colormap cool
        view(0,-90)
        axis tight
    end
end


if plot_shearstresses == true
    for i = 1:N_panels
        figure
        subplot(3,3,2)
        area(sort(edge_coordinate{4,i}),stress_boundaries{8,i})
        hold on
        plot([min(panel(i,2:3:11)) max(panel(i,2:3:11))], ...
            [modelled_stress(i,8) modelled_stress(i,8)], ...
            '--r','LineWidth',2)
        plot([min(panel(i,2:3:11)) max(panel(i,2:3:11))], ...
            [-modelled_stress(i,8) -modelled_stress(i,8)], ...
            '--r','LineWidth',2)
        colormap cool

        subplot(3,3,4)
        area(edge_coordinate{1,i},stress_boundaries{5,i})
        hold on
        plot([min(panel(i,3:3:12)) max(panel(i,3:3:12))], ...
            [modelled_stress(i,8) modelled_stress(i,8)], ...
```

```
            '--r','LineWidth',2)
        plot([min(panel(i,3:3:12)) max(panel(i,3:3:12))], ...
            [-modelled_stress(i,8) -modelled_stress(i,8)], ...
            '--r','LineWidth',2)
        view(-90,90)
        colormap cool

        j = panel_info(i,1);
        subplot(3,3,5)
        textstr(1) = {['panel - ',num2str(j)]};
        textstr(2) = {'Shear stresses'};
        text((min(panel(i,2:3:11))+max(panel(i,2:3:11)))/2, ...
            (min(panel(i,3:3:12))+max(panel(i,3:3:12)))/2,textstr)
        axis([min(panel(i,2:3:11)) max(panel(i,2:3:11)) ...
            min(panel(i,3:3:12)) max(panel(i,3:3:12))])

        subplot(3,3,6)
        area(edge_coordinate{2,i},stress_boundaries{6,i})
        hold on
        plot([min(panel(i,3:3:12)) max(panel(i,3:3:12))], ...
            [modelled_stress(i,8) modelled_stress(i,8)], ...
            '--r','LineWidth',2)
        plot([min(panel(i,3:3:12)) max(panel(i,3:3:12))], ...
            [-modelled_stress(i,8) -modelled_stress(i,8)], ...
            '--r','LineWidth',2)
        view(90,-90)
        colormap cool

        subplot(3,3,8)
        area(sort(edge_coordinate{4,i}),stress_boundaries{7,i})
        hold on
        plot([min(panel(i,2:3:11)) max(panel(i,2:3:11))], ...
            [modelled_stress(i,8) modelled_stress(i,8)], ...
            '--r','LineWidth',2)
        plot([min(panel(i,2:3:11)) max(panel(i,2:3:11))], ...
            [-modelled_stress(i,8) -modelled_stress(i,8)], ...
            '--r','LineWidth',2)
        view(0,-90)
        colormap cool
    end
end
```

D.6 ABSpackage_panels.m

```
% Function name: ABSpackage_panels.m
% Written by: Ottar Hillers, July 2011
% Purpose: Determine parameters for buckling checks according to the ABS
% buckling guide
% -------------------------------------------------------------------------
%% Buckling parameters
Pr = 0.6;                   % Proportional linear elastic limit
C1 = 1.1;                   %
C2 = 1.2;                   %
eta = 1.0;                  % Maximum allowable strength utilization factor
Syield_plate = 259.7;                   % Yield point of plate
Tauyield_plate = Syield_plate/sqrt(3);  % Shear strength
%% Geometry and material
% Length
l = panel_info(:,2);
% Width
s = panel_info(:,3);
% Thickness
t = panel_info(:,6);
% Aspect ratio
```

```
a = l./s;
% Young's modulus
E = panel_info(:,4);
% Poisson's ratio
v = panel_info(:,5);
% Yield stress
Syield(1:length(l),1) =  Syield_plate;

Panel_details = [panel_info(:,1) l s a t E v Syield];

%% Applied stresses
% Maximum x-stress
Smax_x = zeros(N_panels,1);
% Maximum y-stress
Smax_y = zeros(N_panels,1);
% Shear stress
Tau = zeros(N_panels,1);
% Laterl pressure
q = zeros(N_panels,1);
% Von Mises stress
S_vm = zeros(N_panels,1);
% Bending influence of x- and y stresses
kx = zeros(N_panels,1);
ky = zeros(N_panels,1);

for i = 1:N_panels
    Smax_x(i) = modelled_stress(i,2);
    if Smax_x(i) < 0
        Smax_x(i) = 0;
    end
    Smax_y(i) = modelled_stress(i,5);
    if Smax_y(i) < 0
        Smax_y(i) = 0;
    end
    Tau(i) = modelled_stress(i,8);
    q(i) = modelled_stress(i,9);
    S_vm(i) = sqrt(Smax_x(i)^2+Smax_y(i)^2-Smax_x(i)*Smax_y(i)+ ...
        3*Tau(i)^2);
    kx(i) = modelled_stress(i,4);
    ky(i) = modelled_stress(i,7);
end

Applied_stresses = [panel_info(:,1) Smax_x Smax_y Tau q S_vm kx ky];

%% Resistance
% Euler's buckling stress
S_Euler = zeros(N_panels,1);
% Slenderness ratio
beta = zeros(N_panels,1);
% Coefficient to reflect interaction between longitudinal and transverse
% stresses
phi = zeros(N_panels,1);
% Critical x-stress
Sc_x = zeros(N_panels,1);
% Ultimate x-stress
Su_x = zeros(N_panels,1);
% Critical y-stress
Sc_y = zeros(N_panels,1);
% Ultimate y-stress
Su_y = zeros(N_panels,1);
% Critical shear stress
Sc_tau = zeros(N_panels,1);
% Ultimate shear stress
Su_tau = zeros(N_panels,1);

for i = 1:N_panels
    % Euler's buckling stress
    S_Euler(i) = pi^2*E(i)*(t(i)/s(i))^2/(12*(1-v(i)^2));
```

```
    % Slenderness ratio
    beta(i) = s(i)/t(i)*sqrt(Syield_plate/E(i));
    phi(i) = 1-beta(i)/2;

    % Boundary constant for load applied along the short edge
    if kx(i) >= 0
        ks_x = C1*8.4/(kx(i)+1.1);
    elseif kx(i) >= -1
        ks_x = C1*(7.6-6.4*kx(i)+10*kx(i)^2);
    else
        ks_x = C1*24;
    end
    % Critical buckling stress in x-direction
    if ks_x*S_Euler(i) <= Pr*Syield_plate
        Sc_x(i) = ks_x*S_Euler(i);
    else
        Sc_x(i) = Syield_plate*(1-Pr*(1-Pr)*Syield_plate/ ...
            (ks_x*S_Euler(i)));
    end
    % Ultimate buckling stress in x-direction
    if beta(i) > 1
        Cx = 2.0/beta(i)-1/beta(i)^2;
    else
        Cx = 1;
    end
    Su_x(i) = max(Cx*Syield_plate,Sc_x(i));

    % Boundary constant for load applied along the long edge
    if ky(i) >= 1/3
        ks_y = C2*(1+1/a(i)^2)^2*(1.675-0.675*ky(i));
    elseif ky(i) >= -1 && a(i) >=2
        ks_y = C2*((1.0875*(1+1/a(i)^2)^2-9/a(i))*(1+ky(i))+12/a(i));
    elseif ky(i) <= -1 && a(i) >=2
        ks_y = C2*12/a(i)^2;
    elseif ky(i) >= -1 && a(i) >=1
        ks_y = C2*((1.0875*(1+1/a(i)^2)^2-18/a(i)^2)*(1+ky(i))+24/a(i)^2);
    elseif ky(i) <= -1 && a(i) >=1
        ks_y = C2*24/a(i)^2;
    end
    % Critical buckling stress in y-direction
    if ks_y*S_Euler(i) <= Pr*Syield_plate
        Sc_y(i) = ks_y*S_Euler(i);
    else
        Sc_y(i) = Syield_plate*(1-Pr*(1-Pr)*Syield_plate/ ...
            (ks_y*S_Euler(i)));
    end
    % Ultimate buckling stress in y-direction
    Cy = min(Cx/a(i)+0.1*(1-1/a(i))*(1+1/beta(i)^2)^2,1);
    Su_y(i) = max(Cy*Syield_plate,Sc_y(i));

    % Boundary constant for shear
    ks_tau = C1*(4/a(i)^2+5.34);
    % Critical shear stress
    if ks_tau*S_Euler(i) <= Pr*Tauyield_plate
        Sc_tau(i) = ks_tau*S_Euler(i);
    else
        Sc_tau(i) = Tauyield_plate*(1-Pr*(1-Pr)*Tauyield_plate/ ...
            (ks_tau*S_Euler(i)));
    end
    % Ultimate shear stress
    Su_tau(i) = max(Sc_tau(i)+0.5*(Syield_plate-sqrt(3)*Sc_tau(i))/ ...
        sqrt((1+a(i)+a(i)^2)),Sc_tau(i));
end

Resistance_panel1 = [panel_info(:,1) S_Euler beta phi];

%% Plate panels checks
% Buckling state limit
Critical_check = zeros(N_panels,1);
```

```
% Ultimate strength under combined in-plane stresses
Ultimate_check = zeros(N_panels,1);
% Uniformal lateral pressure
Lateral_check = zeros(N_panels,1);

for i = 1:N_panels
    Critical_check(i) = (Smax_x(i)/(eta*Sc_x(i)))^2+ ...
        (Smax_y(i)/(eta*Sc_y(i)))^2+(Tau(i)/(eta*Sc_tau(i)))^2;

    Ultimate_check(i) = (Smax_x(i)/(eta*Su_x(i)))^2+ ...
        (Smax_y(i)/(eta*Su_y(i)))^2+(Tau(i)/(eta*Su_tau(i)))^2- ...
        phi(i)*(Smax_x(i)/(eta*Su_x(i)))*(Smax_y(i)/(eta*Su_y(i)));

    Lateral_check(i) = q(i)/(eta*4*Syield_plate*(t(i)/s(i))^2* ...
        (1+1/a(i)^2)*sqrt(1-(S_vm(i)/Syield_plate)^2));
end

Panel_check = [panel_info(:,1) Critical_check ...
    Ultimate_check Lateral_check];

% Critical failure
[R C] = find(Panel_check(:,2)>1);
Critical_fail = Panel_check(R,1);

% Ultimate failure
[R C] = find(Panel_check(:,3)>1);
Ultimate_fail = Panel_check(R,1);

% Lateral failure
[R C] = find(Panel_check(:,4)>1);
Lateral_fail = Panel_check(R,1);
```

D.7 ABSpackage_stiffened_panels.m

```
% Function name: ABSpackage_stiffened_panels.m
% Written by: Ottar Hillers, July 2011
% Purpose: Determine parameters for buckling checks according to the ABS
% buckling guide
% ---------------------------------------------------------------------------
%% Buckling parameters
Pr = 0.6;                   % Proportional linear elastic limit
C1 = 1.1;                   %
C2 = 1.2;                   %
Csw = 0.58;                 % Factor for effective breadth of plating
Cm = 0.75;                  % Moment adjusment factor
eta = 1.0;                  % Maximum allowable strength utilization factor
N_waves = 10;               % Number of half sinus waves
Syield_stiff = 255;               % Yield point of plate
Tauyield_stiff = Syield_stiff/sqrt(3);  % Shear strength

%% Invalid stiffeners
[R C] = find(isnan(longstiff_info));
invalid = longstiff_info(unique(R),1);

%% Valid stiffeners
% Stiffeners modelled with beam elements
beam_stiffener = longstiff_limit(Nr_longstiff,:);
beam_stiffener(:,1) = real_longstiff(:,1);
beam_stiffener(invalid,:) = [];
n = size(beam_stiffener);

A = real_panel(:,1);
B = panel_limit(Nr_panels,:);
C = [A (B(:,1))];
```

```
for i = 1:length(A)
    a = C(i,2);
    b = find(B(:,1)==a);
    B(b,1)=C(i,1);
end

panel_limit = B;

% panel_stiffener(stiffener, panel below, panel above)
panel_stiffener = zeros(n(1),3);
panel_stiffener(:,1) = beam_stiffener(:,1);
for i = 1:n(1)
    % x and y coordinates for the stiffener
    x1 = beam_stiffener(i,2);
    x2 = beam_stiffener(i,3);
    y = beam_stiffener(i,4);
    % Find panels wich have edges that correspond to the stiffeners
    % coordinates
    px1 = find(panel_limit(:,2)==x1);
    px2 = find(panel_limit(:,3)==x2);
    py1 = find(panel_limit(:,5)==y);
    py2 = find(panel_limit(:,4)==y);
    % 1st check
    a = union(py1,py2);
    b = union(px1,px2);
    c = setdiff(b,a);
    if isempty(c) == 0
        for j = 1:length(c)
            d(j) = find(b==c(j));
        end
        b(d) = [];
        clear d
    end
    if length(b) == 2
        panel_stiffener(i,2:3) = b;
        % 2nd check
    elseif intersect(b,py1)
        for j = 1:length(py2)
            if panel_limit(py2(j),2) < panel_limit(b,2) ...
                    && panel_limit(py2(j),3) > panel_limit(b,2)
                e = py2(j);
                panel_stiffener(i,2:3) = [b e];
            end
        end
    elseif intersect(b,py2)
        for j = 1:length(py1)
            if panel_limit(py1(j),2) < panel_limit(b,2) ...
                    && panel_limit(py1(j),3) > panel_limit(b,2)
                e = py1(j);
                panel_stiffener(i,2:3) = [b e];
            end
        end
    end

end
[R C] = find(panel_stiffener==0);
exclude2 = panel_stiffener(unique(R),1);
panel_stiffener(unique(R),:) = [];

n = size(panel_stiffener);
Nr_stiffener = n(1);
k = 0;
different_panel = [];
for i = 1:Nr_stiffener
    if any(panel_info(panel_stiffener(i,2),2:6)- ...
            panel_info(panel_stiffener(i,3),2:6)>abs(1e-10))
        k = k+1;
        different_panel(k) = panel_stiffener(i,1);
```

```
    end
end

%% Associated panels
% Stiffeners with same properties of associated panels
valid = panel_stiffener(:,1);
stiff_same = setdiff(valid,different_panel);
N_same = length(stiff_same);

% Associated_samepanels(Panel ID, Length, Width, Thickness, Young's
% modulus, Poisson's)
Associated_samepanels = zeros(N_same,7);
Associated_samepanels(:,1) = stiff_same;
index_same = zeros(N_same,1);
for i = 1:N_same
    index_same(i) = find(panel_stiffener(:,1)==stiff_same(i));
    Associated_samepanels(i,2) = ...
        panel_info(panel_stiffener(index_same(i),2),2);
    Associated_samepanels(i,3) = ...
        panel_info(panel_stiffener(index_same(i),2),3);
    Associated_samepanels(i,4) = ...
        panel_info(panel_stiffener(index_same(i),2),6);
    Associated_samepanels(i,5) = ...
        panel_info(panel_stiffener(index_same(i),2),4);
    Associated_samepanels(i,6) = ...
        panel_info(panel_stiffener(index_same(i),2),5);
    Associated_samepanels(i,7) = ...
        Associated_samepanels(i,2)/ ...
        Associated_samepanels(i,3);
end

% Stiffeners with different properties of associated panels
N_diff = length(different_panel);
% Associated_samepanels(Panel ID, Length, Width, Thickness, Young's
% modulus, Poisson's)
Associated_diffpanels = zeros(2*N_diff,7);
Associated_diffpanels(1:2:end,1) = different_panel;
Associated_diffpanels(2:2:end,1) = different_panel;
index_diff = zeros(N_diff,1);
for i = 1:N_diff
    index_diff(i) = find(panel_stiffener(:,1)==different_panel(i));
    Associated_diffpanels(2*i-1,2) = ...
        panel_info(panel_stiffener(index_diff(i),2),2);
    Associated_diffpanels(2*i-1,3) = ...
        panel_info(panel_stiffener(index_diff(i),2),3);
    Associated_diffpanels(2*i-1,4) = ...
        panel_info(panel_stiffener(index_diff(i),2),6);
    Associated_diffpanels(2*i-1,5) = ...
        panel_info(panel_stiffener(index_diff(i),2),4);
    Associated_diffpanels(2*i-1,6) = ...
        panel_info(panel_stiffener(index_diff(i),2),5);
    Associated_diffpanels(2*i-1,7) = ...
        Associated_diffpanels(2*i-1,2)/ ...
        Associated_diffpanels(2*i-1,3);
    Associated_diffpanels(2*i,2) = ...
        panel_info(panel_stiffener(index_diff(i),3),2);
    Associated_diffpanels(2*i,3) = ...
        panel_info(panel_stiffener(index_diff(i),3),3);
    Associated_diffpanels(2*i,4) = ...
        panel_info(panel_stiffener(index_diff(i),3),6);
    Associated_diffpanels(2*i,5) = ...
        panel_info(panel_stiffener(index_diff(i),3),4);
    Associated_diffpanels(2*i,6) = ...
        panel_info(panel_stiffener(index_diff(i),3),5);
    Associated_diffpanels(2*i,7) = ...
        Associated_diffpanels(2*i,2)/ ...
        Associated_diffpanels(2*i,3);
end
```

```
%% Geometry and material
% Height of web (excluding flange thickness)
dw = longstiff_info(valid,5)-longstiff_info(valid,7);
% Thickness of web
tw = longstiff_info(valid,8);
% Width of flange
bf = longstiff_info(valid,6);
% Thickness of flange
tf = longstiff_info(valid,7);
% Length
l_stiff = longstiff_info(valid,2);
% Young's modulus
E_stiff = longstiff_info(valid,3);
% Poisson's ratio
v_stiff = longstiff_info(valid,4);
% Yield stress
Syield_stiffener(1:length(dw),1) =  Syield_stiff;

% Type (1=T-shape,2=L-shape)
type = longstiff_info(valid,9);

Stiffener_details = [longstiff_info(valid,1) l_stiff dw tw bf tf ...
    E_stiff v_stiff Syield_stiffener];

%% Applied stresses
% Panel_stress (Panel ID,Top stress,Bottom stress,Right stress,Left stress)
Panel_stress(:,1) = panel_info(:,1);
Panel_stress(:,2:5) = modelled_location;

% Stress acting on stiffener
Stiffener_stress = zeros(Nr_stiffener,5);
Stiffener_stress(:,1) = panel_stiffener(:,1);
for i = 1:Nr_stiffener
    Stiffener_stress(i,2) = max(Panel_stress(panel_stiffener(i,2),2), ...
        Panel_stress(panel_stiffener(i,3),3));
    if Stiffener_stress(i,2) < 0
        Stiffener_stress(i,2) = 0;
    end
    Stiffener_stress(i,3) = max(Panel_stress(panel_stiffener(i,2),4), ...
        Panel_stress(panel_stiffener(i,3),5));
    if Stiffener_stress(i,3) < 0
        Stiffener_stress(i,3) = 0;
    end
    Stiffener_stress(i,4) = ...
        max(abs(modelled_stress(panel_stiffener(i,2),8)), ...
        abs(modelled_stress(panel_stiffener(i,3),8)));
    Stiffener_stress(i,5) = ...
        max(modelled_stress(panel_stiffener(i,2),9), ...
        modelled_stress(panel_stiffener(i,3),9));
end

%% Area properties
% Stiffener area
area_stiffener(:,1) = valid;
area_stiffener(:,2) = dw.*tw+bf.*tf;

area_samestiffener = area_stiffener(index_same,:);
area_diffstiffener = zeros(2*N_diff,2);
area_diffstiffener(1:2:end,1) = panel_stiffener(index_diff,1);
area_diffstiffener(2:2:end,1) = panel_stiffener(index_diff,1);
area_diffstiffener(1:2:end,2) = area_stiffener(index_diff,2);
area_diffstiffener(2:2:end,2) = area_stiffener(index_diff,2);

% Total area of stiffener and associated panels
area_sametotal(:,1) = panel_stiffener(index_same,1);
area_sametotal(:,2) = area_stiffener(index_same,2)+ ...
    Associated_samepanels(:,3).*Associated_samepanels(:,4);

area_difftotal = zeros(2*N_diff,2);
```

```
area_difftotal(:,1) = area_diffstiffener(:,1);
area_difftotal(1:2:end,2) = area_stiffener(index_diff,2)+ ...
    Associated_diffpanels(1:2:end,3).*Associated_diffpanels(1:2:end,4);
area_difftotal(2:2:end,2) = area_stiffener(index_diff,2)+ ...
    Associated_diffpanels(2:2:end,3).*Associated_diffpanels(2:2:end,4);

% Effective area
beta_same(:,1) = panel_stiffener(index_same,1);
beta_same(:,2) = Associated_samepanels(:,3)./ ...
    Associated_samepanels(:,4).*sqrt(Syield_plate./ ...
    Associated_samepanels(:,5));

beta_diff = zeros(2*N_diff,2);
beta_diff(:,1) = area_difftotal(:,1);
beta_diff(1:2:end,2) = Associated_diffpanels(1:2:end,3)./ ...
    Associated_diffpanels(1:2:end,4).* ...
    sqrt(Syield_plate./Associated_diffpanels(1:2:end,5));
beta_diff(2:2:end,2) = Associated_diffpanels(2:2:end,3)./ ...
    Associated_diffpanels(2:2:end,4).* ...
    sqrt(Syield_plate./Associated_diffpanels(2:2:end,5));

Cx_same = zeros(N_same,2);
Cx_same(:,1) = panel_stiffener(index_same,1);
for i = 1:N_same
    if beta_same(i,2) > 1
        Cx_same(i,2) = 2/beta_same(i,2)-1/beta_same(i,2)^2;
    else
        Cx_same(i,2) = 1;
    end
end

Cx_diff = zeros(2*N_diff,2);
Cx_diff(:,1) = area_difftotal(:,1);
for i = 1:2*N_diff
    if beta_diff(i,2) > 1
        Cx_diff(i,2) = 2/beta_diff(i,2)-1/beta_diff(i,2)^2;
    else
        Cx_diff(i,2) = 1;
    end
end

phi_same(:,1) = beta_same(:,1);
phi_same(:,2) = 1-beta_same(:,2)/2;

phi_diff(:,1) = beta_diff(:,1);
phi_diff(:,2) = 1-beta_diff(:,2)/2;

Cy_same = zeros(N_same,2);
Cy_same(:,1) = panel_stiffener(index_same,1);
for i = 1:N_same
    crit_ultimate = Stiffener_stress(index_same(i),3)/ ...
        min(Su_y(panel_stiffener(index_same(i),2)), ...
        Su_y(panel_stiffener(index_same(i),3)));
    Cy_same(i,2) = 0.5*phi_same(i,2)*crit_ultimate+sqrt(1-(1-0.25* ...
        phi_same(i,2).^2)*crit_ultimate^2);
    if Cy_same(i,2) > 1
        Cy_same(i,2) = 1;
    end
end

Cy_diff = zeros(2*N_diff,2);
Cy_diff(:,1) = beta_diff(:,1);
for i = 1:N_diff
    crit_ultimate = Stiffener_stress(index_diff(i),3)/ ...
        Su_y(panel_stiffener(index_diff(i),2));
    Cy_diff(2*i-1,2) = 0.5*phi_diff(2*i-1,2)* crit_ultimate+ ...
        sqrt(1-(1-0.25*phi_diff(2*i-1,2).^2)*crit_ultimate^2);
    if Cy_diff(2*i-1,2) > 1
        Cy_diff(2*i-1,2) = 1;
```

```
    end

    crit_ultimate = Stiffener_stress(index_diff(i),3)/ ...
        Su_y(panel_stiffener(index_diff(i),3));
    Cy_diff(2*i,2) = 0.5*phi_diff(2*i,2)* crit_ultimate+sqrt(1-(1-0.25* ...
        phi_diff(2*i,2).^2)*crit_ultimate^2);
    if Cy_diff(2*i,2) > 1
        Cy_diff(2*i,2) = 1;
    end
end

Cxy_same(:,1) = panel_stiffener(index_same,1);
Cxy_same(:,2) = sqrt(1-(Stiffener_stress(index_same,4)/Tauyield_plate).^2);

Cxy_diff(:,1) =  beta_diff(:,1);
Cxy_diff(1:2:end,2) = sqrt(1-(Stiffener_stress(index_diff,4)/ ...
    Tauyield_plate).^2);
Cxy_diff(2:2:end,2) = sqrt(1-(Stiffener_stress(index_diff,4)/ ...
    Tauyield_plate).^2);

se_same = zeros(N_same,2);
se_same(:,1) = panel_stiffener(index_same,1);
for i = 1:N_same
    if any(ismember(Critical_fail,panel_stiffener(index_same(i),2))) || ...
            any(ismember(Critical_fail,panel_stiffener(index_same(i),3)))
        se_same(i,2) = Associated_samepanels(i,3).* ...
            Cx_same(i,2).*Cy_same(i,2).*Cxy_same(i,2);
    else
        se_same(i,2) = Associated_samepanels(i,3);
    end
end

se_diff = zeros(2*N_diff,2);
se_diff(:,1) = beta_diff(:,1);
for i = 1:N_diff
    if any(ismember(Critical_fail,panel_stiffener(index_diff(i),2))) || ...
            any(ismember(Critical_fail,panel_stiffener(index_diff(i),3)))
        se_diff(2*i-1,2) = Associated_diffpanels(2*i-1,3).* ...
            Cx_diff(2*i-1,2).*Cy_diff(2*i-1,2).*Cxy_diff(2*i-1,2);
        se_diff(2*i,2) = Associated_diffpanels(2*i,3).* ...
            Cx_diff(2*i,2).*Cy_diff(2*i,2).*Cxy_diff(2*i,2);
    else
        se_diff(2*i-1,2) = Associated_diffpanels(2*i-1,3);
        se_diff(2*i,2) = Associated_diffpanels(2*i,3);
    end
end

area_sameeffective(:,1) = panel_stiffener(index_same,1);
area_sameeffective(:,2) = area_samestiffener(:,2)+ ...
    se_same(:,2).*Associated_samepanels(:,4);

area_diffeffective(:,1) = beta_diff(:,1);
area_diffeffective(:,2) = area_diffstiffener(:,2)+ ...
    se_diff(:,2).*Associated_diffpanels(:,4);

Area_same = [area_samestiffener area_sametotal(:,2) ...
    area_sameeffective(:,2)];

Area_diff = [area_diffstiffener area_difftotal(:,2) ...
    area_diffeffective(:,2)];

%% Geometric properties
% St. Venant torsion constant
K = zeros(Nr_stiffener,2);
K(:,1) = panel_stiffener(:,1);

% Warping constant
Gamma = zeros(Nr_stiffener,2);
Gamma(:,1) = panel_stiffener(:,1);
```

```
% Polar moment of inertia
I_0 = zeros(Nr_stiffener,2);
I_0(:,1) = panel_stiffener(:,1);

% Effective moment of inertia
Ie_same = zeros(N_same,2);
Ie_same(:,1) = panel_stiffener(index_same,1);

Ie_diff = zeros(2*N_diff,2);
Ie_diff(:,1) = beta_diff(:,1);

% Effective radius of gyration
re_same = zeros(N_same,2);
re_same(:,1) = panel_stiffener(index_same,1);

re_diff = zeros(2*N_diff,2);
re_diff(:,1) = beta_diff(:,1);

% Effective section modulus
SMw_same = zeros(N_same,2);
SMw_same(:,1) = panel_stiffener(index_same,1);

SMw_diff = zeros(2*N_diff,2);
SMw_diff(:,1) = beta_diff(:,1);

% C_0 factor
C0_same = zeros(N_same,2);
C0_same(:,1) = panel_stiffener(index_same,1);

C0_diff = zeros(2*N_diff,2);
C0_diff(:,1) = beta_diff(:,1);

% Properties which are the same for all stiffeners
K(:,2) = (bf.*tf.^3+dw.*tw.^3)/3;
u = zeros(Nr_stiffener,1);
b1 = zeros(Nr_stiffener,1);
for i = 1:Nr_stiffener
    if type(i) == 1
        u(i) = 0;
        b1(i) = bf(i)/2;
    elseif type(i) == 2
        u(i) = 1;
        b1(i) = 0;
    end
end
m = 1-u.*(0.7-0.1*dw./bf);
As = area_stiffener(:,2);
y0 = (b1-0.5*bf).*bf.*tf./As;
z0 = (0.5*dw.^2.*tw+(dw+0.5*tf).*bf.*tf)./As;
Iy = (dw.^3.*tw+tf.^3.*bf)/12+0.25*dw.^3.*tw+bf.*tf.*(dw+0.5*tf).^2- ...
    As.*z0.^2;
Iz = (tw.^3.*dw+bf.^3.*tf)/12+bf.*tf.*(b1-0.5*bf).^2-As.*y0.^2;
I_0(:,2) = Iy+m.*Iz+As.*(y0.^2+z0.^2);
Gamma(:,2) = m.*tf.*bf.^3/12.*(1+3*u.^2.*dw.*tw./As).*dw.^2+ ...
    dw.^3.*tw.^3/36;

% Stiffeners with associated panels with same properties
Dw = dw(index_same);
Tw = tw(index_same);
Bf = bf(index_same);
Tf = tf(index_same);
t = Associated_samepanels(:,4);
Se = se_same(:,2);
Ae = area_sameeffective(:,2);
Aw = Csw*Associated_samepanels(:,3).*t+area_samestiffener(:,2);
zep = (0.5*(t+Dw).*Dw.*Tw+(0.5*t+Dw+0.5*Tf).*Bf.*Tf)./Ae;
zwp = (0.5*(t+Dw).*Dw.*Tw+(0.5*t+Dw+0.5*Tf).*Bf.*Tf)./Aw;
Ie_same(:,2) = 1/12*(t.^3.*Se+Dw.^3.*Tw+Tf.^3.*Bf)+ ...
```

```
    0.25*(t+Dw).^2.*Dw.*Tw+Bf.*Tf.*(0.5*t+Dw+0.5*Tf).^2-Ae.*zep.^2;
re_same(:,2) = sqrt(Ie_same(:,2)./Ae);
Iw = 1/12*(t.^3.*Se+Dw.^3.*Tw+Tf.^3.*Bf)+ ...
    0.25*(t+Dw).^2.*Dw.*Tw+Bf.*Tf.*(0.5*t+Dw+0.5*Tf).^2-Aw.*zwp.^2;
SMw_same(:,2) = Iw./((0.5*t+Dw+Tf)-zwp);
C0_same(:,2) = Associated_samepanels(:,5).* ...
    Associated_samepanels(:,4).^3./(3*Associated_samepanels(:,3));

clear Dw Tw Bf Tf t Se Ae Aw zep zwp
% Stiffeners with associated panels with different properties
n = size(index_diff);
Dw(1:2:2*n(1)-1,1) = dw(index_diff);
Dw(2:2:2*n(1),1) = dw(index_diff);
Tw(1:2:2*n(1)-1,1) = tw(index_diff);
Tw(2:2:2*n(1),1) = tw(index_diff);
Bf(1:2:2*n(1)-1,1) = bf(index_diff);
Bf(2:2:2*n(1),1) = bf(index_diff);
Tf(1:2:2*n(1)-1,1) = tf(index_diff);
Tf(2:2:2*n(1),1) = tf(index_diff);
t = Associated_diffpanels(:,4);
Se = se_diff(:,2);
Ae = area_diffeffective(:,2);
Aw = Csw*Associated_diffpanels(:,3).*t+area_diffstiffener(:,2);
zep = (0.5*(t+Dw).*Dw.*Tw+(0.5*t+Dw+0.5*Tf).*Bf.*Tf)./Ae;
zwp = (0.5*(t+Dw).*Dw.*Tw+(0.5*t+Dw+0.5*Tf).*Bf.*Tf)./Aw;
Ie_diff(:,2) = 1/12*(t.^3.*Se+Dw.^3.*Tw+Tf.^3.*Bf)+ ...
    0.25*(t+Dw).^2.*Dw.*Tw+Bf.*Tf.*(0.5*t+Dw+0.5*Tf).^2-Ae.*zep.^2;
re_diff(:,2) = sqrt(Ie_diff(:,2)./Ae);
Iw = 1/12*(t.^3.*Se+Dw.^3.*Tw+Tf.^3.*Bf)+ ...
    0.25*(t+Dw).^2.*Dw.*Tw+Bf.*Tf.*(0.5*t+Dw+0.5*Tf).^2-Aw.*zwp.^2;
SMw_diff(:,2) = Iw./((0.5*t+Dw+Tf)-zwp);
C0_diff(:,2) = Associated_diffpanels(:,5).* ...
    Associated_diffpanels(:,4).^3./(3*Associated_diffpanels(:,3));

%% Bending stress, critical buckling stress and compressive stress
% Maximum bending moment
M_same = zeros(N_same,2);
M_same(:,1) = panel_stiffener(index_same,1);
q_same = Stiffener_stress(index_same,5);
M_same(:,2) = q_same.*Associated_samepanels(:,3).* ...
    Stiffener_details(index_same,2).^2/12;

M_diff = zeros(2*N_diff,2);
M_diff(:,1) = beta_diff(:,1);
q_diff = zeros(2*N_diff,1);
q_diff(1:2:end) = Stiffener_stress(index_diff,5);
q_diff(2:2:end) = Stiffener_stress(index_diff,5);
stiff_length = zeros(2*N_diff,1);
stiff_length(1:2:end) = Stiffener_details(index_diff,2);
stiff_length(2:2:end) = Stiffener_details(index_diff,2);
M_diff(:,2) = q_diff.*Associated_diffpanels(:,3).* ...
    stiff_length.^2/12;

% Bending stress
Sbending_same = zeros(N_same,2);
Sbending_same(:,1) = panel_stiffener(index_same,1);
Sbending_same(:,2) = M_same(:,2)./SMw_same(:,2);

Sbending_diff = zeros(2*N_diff,2);
Sbending_diff(:,1) = beta_diff(:,1);
Sbending_diff(:,2) = M_diff(:,2)./SMw_diff(:,2);

% Euler buckling stress
Seuler_same = zeros(N_same,2);
Seuler_same(:,1) = panel_stiffener(index_same,1);
Seuler_same(:,2) = pi^2*Stiffener_details(index_same,7).* ...
    re_same(:,2).^2./Stiffener_details(index_same,2).^2;

Seuler_diff = zeros(2*N_diff,2);
```

```
Seuler_diff(:,1) = beta_diff(:,1);
stiff_youngs = zeros(2*N_diff,1);
stiff_youngs(1:2:end) = Stiffener_details(index_diff,7);
stiff_youngs(2:2:end) = Stiffener_details(index_diff,7);
Seuler_diff(:,2) = pi^2*stiff_youngs.* ...
    re_diff(:,2).^2./stiff_length.^2;

% Specific minimum yield point of composite stiffener and effective plate
Syield_same = zeros(N_same,2);
Syield_same(:,1) = panel_stiffener(index_same,1);
Syield_same(:,2) = ((area_sameeffective(:,2)-area_samestiffener(:,2))* ...
    Syield_plate+area_samestiffener(:,2)*Syield_stiff)./ ...
    area_sameeffective(:,2);

Syield_diff = zeros(2*N_diff,2);
Syield_diff(:,1) = beta_diff(:,1);
Syield_diff(:,2) = ((area_diffeffective(:,2)-area_diffstiffener(:,2))* ...
    Syield_plate+area_diffstiffener(:,2)*Syield_stiff)./ ...
    area_diffeffective(:,2);

% Critical buckling stress
Scritical_same = zeros(N_same,2);
Scritical_same(:,1) = panel_stiffener(index_same,1);
for i = 1:N_same
    if Seuler_same(i,2) <= Pr*Syield_same(i,2)
        Scritical_same(i,2) = Seuler_same(i,2);
    else
        Scritical_same(i,2) = Syield_same(i,2)*(1-Pr*(1-Pr)* ...
            Syield_same(i,2)/Seuler_same(i,2));
    end
end

Scritical_diff = zeros(2*N_diff,2);
Scritical_diff(:,1) = beta_diff(:,1);
for i = 1:N_diff
    if Seuler_diff(2*i-1,2) <= Pr*Syield_diff(2*i-1,2)
        Scritical_diff(2*i-1,2) = Seuler_diff(2*i-1,2);
    else
        Scritical_diff(2*i-1,2) = Syield_diff(2*i-1,2)*(1-Pr*(1-Pr)* ...
            Syield_diff(2*i-1,2)/Seuler_diff(2*i-1,2));
    end

    if Seuler_diff(2*i,2) <= Pr*Syield_diff(2*i,2)
        Scritical_diff(2*i,2) = Seuler_diff(2*i,2);
    else
        Scritical_diff(2*i,2) = Syield_diff(2*i,2)*(1-Pr*(1-Pr)* ...
            Syield_diff(2*i,2)/Seuler_diff(2*i,2));
    end
end

% Critical buckling stress of associted plates coresponding to N-half waves
Shalfwaves_same = zeros(N_same,N_waves+1);
Shalfwaves_same(:,1) = panel_stiffener(index_same,1);
for i = 1:N_waves
    Shalfwaves_same(:,i+1) = pi^2*Associated_samepanels(:,5).* ...
        (i./Associated_samepanels(:,7)+ ...
        Associated_samepanels(:,7)./i).^2.* ...
        (Associated_samepanels(:,4)./Associated_samepanels(:,3)).^2./ ...
        (12*(1-Associated_samepanels(:,6).^2));
end

Shalfwaves_diff = zeros(2*N_diff,N_waves+1);
Shalfwaves_diff(:,1) = beta_diff(:,1);
for i = 1:N_waves
    Shalfwaves_diff(:,i+1) = pi^2*Associated_diffpanels(:,5).* ...
        (i./Associated_diffpanels(:,7)+Associated_diffpanels(:,7)./i).* ...
        (Associated_diffpanels(:,4)./Associated_diffpanels(:,3)).^2./ ...
        (12*(1-Associated_diffpanels(:,6).^2));
end
```

```
% Elastic flexuaral torsional buckling stress
Storsional_same = zeros(N_same,N_waves+1);
Storsional_same(:,1) = panel_stiffener(index_same,1);
K_same = K(index_same,2);
Gamma_same = Gamma(index_same,2);
I0_same = I_0(index_same,2);
L_same = Stiffener_details(index_same,2);
for i = 1:N_waves
    Storsional_same(:,i+1) = (K_same/2.6+(i*pi./L_same).^2.*Gamma_same+ ...
        C0_same(:,2)./Associated_samepanels(:,5).* ...
        (L_same./(i*pi)).^2)./(I0_same+C0_same(:,2)./ ...
        Shalfwaves_same(:,i+1).*(L_same/(i*pi)).^2) ...
        .*Associated_samepanels(:,5);
end

Storsional_diff = zeros(2*N_diff,N_waves+1);
Storsional_diff(:,1) = beta_diff(:,1);
K_diff = zeros(2*N_diff,1);
K_diff(1:2:end) = K(index_diff,2);
K_diff(2:2:end) = K(index_diff,2);
Gamma_diff = zeros(2*N_diff,1);
Gamma_diff(1:2:end) = Gamma(index_diff,2);
Gamma_diff(2:2:end) = Gamma(index_diff,2);
I0_diff = zeros(2*N_diff,1);
I0_diff(1:2:end) = I_0(index_diff,2);
I0_diff(2:2:end) = I_0(index_diff,2);
L_diff = zeros(2*N_diff,1);
L_diff(1:2:end) = Stiffener_details(index_diff,2);
L_diff(2:2:end) = Stiffener_details(index_diff,2);
for i = 1:N_waves
    Storsional_diff(:,i+1) = (K_diff/2.6+(i*pi./L_diff).^2.*Gamma_diff+ ...
        C0_diff(:,2)./Associated_diffpanels(:,5).* ...
        (L_diff./(i*pi)).^2)./(I0_diff+C0_diff(:,2)./ ...
        Shalfwaves_diff(:,i+1).*(L_diff/(i*pi)).^2) ...
        .*Associated_diffpanels(:,5);
end

% Compressive stress
Scompressive_same(:,1) = panel_stiffener(index_same,1);
Scompressive_same(:,2) = Stiffener_stress(index_same,2);

Scompressive_diff = zeros(2*N_diff,2);
Scompressive_diff(:,1) = beta_diff(:,1);
Scompressive_diff(1:2:end,2) = Stiffener_stress(index_diff,2);
Scompressive_diff(2:2:end,2) = Stiffener_stress(index_diff,2);

Prop_same(:,1:4) = Area_same;
Prop_same(:,5:8) = [SMw_same(:,2) K_same I0_same Gamma_same];

Prop_diff(:,1:4) = Area_diff;
Prop_diff(:,5:8) = [SMw_diff(:,2) K_diff I0_diff Gamma_diff];

%% Stiffened plate panels checks
% Beam-Column Buckling State limit
BeamColumn_samecheck(:,1) = Stiffener_stress(index_same,1);
Sa = Scompressive_same(:,2);
Sca = Scritical_same(:,2);
Sb = Sbending_same(:,2);
Se = Seuler_same(:,2);
Ae = area_sameeffective(:,2);
A = area_sametotal(:,2);
BeamColumn_samecheck(:,2) = Sa./(eta*Sca.*(Ae./A))+ ...
    Cm*Sb./(eta*Syield_same(:,2).*(1-Sa./(eta*Se)));
Resistance_samestiffeners(:,1:5) = [Stiffener_stress(index_same,1) ...
    Syield_same(:,2) Se Sca Sb];

BeamColumn_diffcheck(:,1) = beta_diff(:,1);
Sa = Scompressive_diff(:,2);
```

```
Sca = Scritical_diff(:,2);
Sb = Sbending_diff(:,2);
Se = Seuler_diff(:,2);
Ae = area_diffeffective(:,2);
A = area_difftotal(:,2);
BeamColumn_diffcheck(:,2) = Sa./(eta*Sca.*(Ae./A))+ ...
    Cm*Sb./(eta*Syield_diff(:,2).*(1-Sa./(eta*Se)));
Resistance_diffstiffeners(:,1:5) = [beta_diff(:,1) Syield_diff(:,2) ...
    Se Sca Sb];

diff_check = zeros(N_diff,2);
diff_check(:,1) = Stiffener_stress(index_diff,1);
for i = 1:N_diff
    diff_check(i,2) = max(BeamColumn_diffcheck(2*i-1,2), ...
        BeamColumn_diffcheck(2*i,2));
end

BeamColumn_check = sortrows([BeamColumn_samecheck;diff_check]);

% Flexural-Torsional buckling state limit
Torsional_samecheck(:,1) = Stiffener_stress(index_same,1);
Sa = Scompressive_same(:,2);
Set = [Storsional_same(:,1) min(Storsional_same(:,2:end),[],2)];
Sct = zeros(N_same,2);
Sct(:,1) = Stiffener_stress(index_same,1);
for i = 1:N_same
    if Set(i,2) <= Pr*Syield_same(i,2)
        Sct(i,2) = Set(i,2);
    else
        Sct(i,2) = Syield_same(i,2)*(1-Pr*(1-Pr)* ...
            Syield_same(i,2)/Set(i,2));
    end
end
Torsional_samecheck(:,2) = Sa./(eta*Sct(:,2));
Resistance_samestiffeners(:,6:7) = [Set(:,2) Sct(:,2)];

Torsional_diffcheck(:,1) = beta_diff(:,1);
Sa = Scompressive_diff(:,2);
Set = [Storsional_diff(:,1) min(Storsional_diff(:,2:end),[],2)];
Sct = zeros(2*N_diff,2);
Sct(1:2:end,1) = Stiffener_stress(index_diff,1);
Sct(2:2:end,1) = Stiffener_stress(index_diff,1);
for i = 1:N_diff
    if Set(2*i-1,2) <= Pr*Syield_diff(2*i-1,2)
        Sct(2*i-1,2) = Set(2*i-1,2);
    else
        Sct(2*i-1,2) = Syield_diff(2*i-1,2)*(1-Pr*(1-Pr)* ...
            Syield_diff(2*i-1,2)/Set(2*i-1,2));
    end

    if Set(2*i,2) <= Pr*Syield_diff(2*i,2)
        Sct(2*i,2) = Set(2*i,2);
    else
        Sct(2*i,2) = Syield_diff(2*i,2)*(1-Pr*(1-Pr)* ...
            Syield_diff(2*i,2)/Set(2-1,2));
    end
end
Torsional_diffcheck(:,2) = Sa./(eta*Sct(:,2));
Resistance_diffstiffeners(:,6:7) = [Set(:,2) Sct(:,2)];

diff_check = zeros(N_diff,2);
diff_check(:,1) = Stiffener_stress(index_diff,1);
for i = 1:N_diff
    diff_check(i,2) = max(Torsional_diffcheck(2*i-1,2), ...
        Torsional_diffcheck(2*i,2));
end

Torsional_check = sortrows([Torsional_samecheck;diff_check]);
```

```
Stiffener_check = [Torsional_check(:,1) BeamColumn_check(:,2) ...
    Torsional_check(:,2)];

% Beam-Column Buckling failure
[R C] = find(BeamColumn_check(:,2)>1);
BeamColumn_fail = BeamColumn_check(R,1);

% Flexural-Torsional Buckling failure
[R C] = find(Torsional_check(:,2)>1);
Torsional_fail = Torsional_check(R,1);
```

D.8 Results.m

```
% Function name: Results.m
% Written by: Ottar Hillers, August 2011
% Purpose:
% ----------------------------------------------------------------------------
%% Stiffeners which are not valid for unity checks
disp('----------------------------------------------------------------------')
disp('It is not possible to perform beam-column buckling check on the')
disp('following stiffeners because they are either modelled with plate')
disp('elements or the stiffeners are only attached to one panel, buckling')
disp('check will not be proceded on these stiffeners')
disp('----------------------------------------------------------------------')
disp('Stiffener        ')
disp('  Nr.       Reason')
disp1 = sprintf('%8.0f   Modelled with plate elements\n',invalid);
disp2 = sprintf('%8.0f   Stiffener only attached to one panel\n',exclude2);
disp(disp1)
disp(disp2)

%% Stiffners which have associated panels with differnet properties
if isempty(different_panel) == 0
    disp('------------------------------------------------------------------')
    disp('The associated panels to the following stiffener have different')
    disp('properties. Buckling check for these stiffeners will be based on')
    disp('worst case scenario')
    disp('------------------------------------------------------------------')
    disp(' Stiffener Nr.')
    fprintf(1,'%6.0f\n',different_panel)
end

%% Unity checks which fail
% Buckling state limit check
if isempty(Critical_fail) == 0
    disp('------------------------------------------------------------')
    disp('The following panels failed the Buckling state limit check')
    disp('------------------------------------------------------------')
    disp(Critical_fail)
end

% Ultimate strength under combined in-plane stresses
if isempty(Ultimate_fail) == 0
    disp('---------------------------------------------------------')
    disp('The following panels failed the Ultimate strength under')
    disp('combined in-plane stresses check')
    disp('---------------------------------------------------------')
    disp(Ultimate_fail)

end

% Uniformal lateral pressure
if isempty(Lateral_fail) == 0
    disp('------------------------------------------------------------------')
```

```
    disp('The following panels failed the Uniformal lateral pressure check')
    disp('------------------------------------------------------------------')
    disp(Lateral_fail)
end

% Beam-Column Buckling State limit
if isempty(BeamColumn_fail) == 0
    disp('------------------------------------------------------------')
    disp('The following stiffeners failed the Beam-Column Buckling')
    disp('State limit check')
    disp('------------------------------------------------------------')
    disp(BeamColumn_fail)
end

% Flexural-Torsional Buckling State limit
if isempty(Torsional_fail) == 0
    disp('----------------------------------------------------------')
    disp('The following stiffeners failed the Flexural-Torsional')
    disp('Buckling State limit check')
    disp('----------------------------------------------------------')
    disp(Torsional_fail)
end

%% All unity checks
disp('------------------------------------')
disp('ABS Plate buckling stress check')
disp('------------------------------------')
disp('Panel  Critical  Ultimate  Lateral')
disp('  Nr.      check     check    check')
fprintf(1,'%4.0f %10.2f %9.2f %8.2f\n',Panel_check')

disp('-------------------------------------------')
disp('ABS Stiffened panels buckling stress check')
disp('-------------------------------------------')
disp('            Beam/  Flexural/')
disp('Stiffener  Column  Torsional')
disp('    Nr.     check      check')
fprintf(1,'%6.0f %10.2f %10.2f\n',Stiffener_check')

%% Geometry and material properties
disp('------------------------------------------------------')
disp('Geometry and material properties of panels')
disp('------------------------------------------------------')
disp('Panel                 Aspect                Youngs  Poissons   Yield')
disp('  Nr.  Length  Width   ratio  Thickness  modulus     ratio  Stress')
fprintf(1,'%4.0f %8.0f %6.0f %7.2f %10.1f  %7.0f %9.1f %7.1f\n',Panel_details')

disp('----------------------------------------------------------------------')
disp('Geometry and material properties of stiffeners')
disp('----------------------------------------------------------------------')
disp('Stiffener             Web        Web  Flange     Flange   Youngs  Poissons
Yield')
disp('    Nr.    Length  height  thickness    width  thickness  modulus     ratio  Stress')
fprintf(1,'%6.0f %10.0f %7.0f %10.1f %7.0f %10.1f  %7.0f %9.1f % 7.1f\n',Stiffener_details')

%% Modelled and applied stresses
disp('--------------------------------------------------------------------------------')
disp('Modelled stress distribution')
disp('--------------------------------------------------------------------------------')
disp('Panel   Maximum   Minimum     Ratio   Maximum   Minimum     Ratio   Shear  Lateral')
disp('  Nr.  x-stress  x-stress  x-stress  y-stress  y-stress  y-stress  stress pressure')
fprintf(1,'%4.0f %10.1f %9.1f %9.2f %9.1f %9.1f %9.2f %7.1f %8.4f\n',modelled_stress')

disp('-------------------------------------------------------------------------')
disp('Applied stresses on panels')
disp('-------------------------------------------------------------------------')
disp('Panel   Maximum   Maximum   Shear  Lateral  Von Mises   Bending   Bending')
disp('  Nr.  x-stress  y-stress  stress     load     stress  x-stress  y-stress')
fprintf(1,'%4.0f %10.1f %9.1f %7.1f %8.1f %10.1f  %8.2f %9.2f\n',Applied_stresses')
```

```
disp('---------------------------------------------------')
disp('Stress on stiffeners')
disp('---------------------------------------------------')
disp('Stiffener  Modelled  Modelled  Modelled  Modelled')
disp('    Nr.    x-Stress  y-stress     shear  pressure')
fprintf(1,'%6.0f %12.1f %9.1f %9.1f %9.3f\n',Stiffener_stress')

%% Resistance of panels
disp('--------------------------------------------------------')
disp('Resistance of panels panels 1/2')
disp('--------------------------------------------------------')
disp('Panel   Euler   Slenderness  Coefficient of')
disp('  Nr.  stress         ratio     interaction')
fprintf(1,'%4.0f %8.1f  %12.2f %15.2f\n',Resistance_panel1')

Resistance_panel2 = [panel_info(:,1) Sc_x Su_x Sc_y Su_y Sc_tau Su_tau];
disp('---------------------------------------------------------------------------')
disp('Resistance of panels panels 2/2')
disp('---------------------------------------------------------------------------')
disp('Panel  Critical  Ultimate  Critical  Ultimate      Critical      Ultimate')
disp('  Nr.  x-stress  x-stress  y-stress  y-stress  shear stress  shear stress')
fprintf(1,'%4.0f %10.1f %9.1f %9.1f %9.1f %13.2f %13.2f\n',Resistance_panel2')

%% Resistance of stiffeners
disp('-------------------------------------------------------------------------')
disp('Resistance of stiffeners with identical associated panels')
disp('-------------------------------------------------------------------------')
disp('           Effective     Euler  Critical             Elastic   Critical')
disp('Stiffener      yield  Buckling  Buckling  Bending  torsional  torsional')
disp('    Nr.       stress    stress    stress   stress     stress     stress')
fprintf(1,'%6.0f %13.1f %9.1f  %8.1f  %7.1f  %9.1f %10.1f\n',Resistance_samestiffeners')

if isempty(Resistance_diffstiffeners) == 0
    disp('-------------------------------------------------------------------------')
    disp('Resistance of stiffeners with different associated panels')
    disp('-------------------------------------------------------------------------')
    disp('           Effective     Euler  Critical             Elastic   Critical')
    disp('Stiffener      yield  Buckling  Buckling  Bending  torsional  torsional')
    disp('    Nr.       stress    stress    stress   stress     stress     stress')
    fprintf(1,'%6.0f %13.1f %9.1f  %8.1f  %7.1f  %9.1f %10.1f\n',Resistance_diffstiffeners')
end

%% Properties of stiffeners
disp('-------------------------------------------------------------------------------------')
disp('Stiffeners with identical associated panels')
disp('-------------------------------------------------------------------------------------')
disp('Stiffener  Stiffener  Total  Effective  Section  St. Venant  Polar moment   Warping')
disp('    Nr.         area   area       area  modulus    constant   of inertia  constant')
fprintf(1,'%6.0f  %12.0f %6.0f %10.0f %8.0f %11.0f %13.0f %4.2e\n',Prop_same')

if isempty(Area_diff) == 0
    disp('-------------------------------------------------------------------------------------')
    disp('Stiffeners with different associated panels')
    disp('-------------------------------------------------------------------------------------')
    disp('Stiffener  Stiffener  Total  Effective  Section  St. Venant  Polar moment
Warping')
    disp('    Nr.         area   area       area  modulus    constant   of inertia
constant')
    fprintf(1,'%6.0f  %12.0f %6.0f %10.0f %8.0f %11.0f %13.0f %4.2e\n',Prop_diff')
end
```