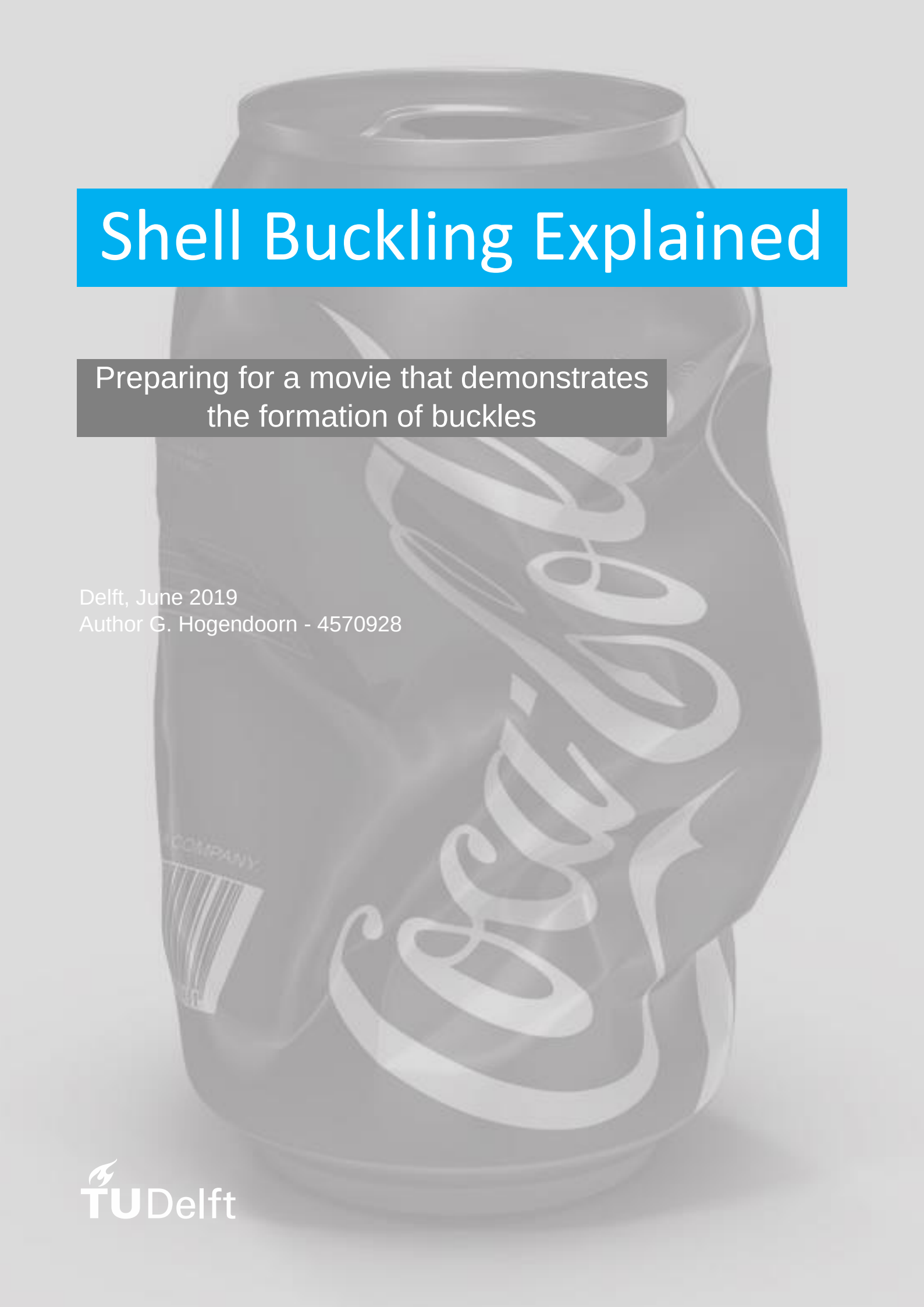
Shell Buckling Explained

Preparing for a movie that demonstrates the formation of buckles



Delft, June 2019
Author G. Hogendoorn - 4570928

TU Delft

Shell Buckling Explained

Preparing for a movie that demonstrates the formation of buckles

By

G. (Guido) Hogendoorn

in partial fulfilment of the requirements for the degree of

Bachelor of Science
in Civil Engineering

at the Delft University of Technology

Author:	G. (Guido) Hogendoorn Student number: 4570928
Supervisors:	Dr. ir. P. C. J. Hoogenboom Dr. ir. R. Abspoel
Date:	20 June 2019

TU Delft

Preface

In order to complete the bachelor's degree in Civil Engineering at the Delft University of Technology (TU Delft, the Netherlands), it is needed to carry out a small research. The course code of this so-called "bachelor eindwerk" is CTB3000 and after completion the student earns 10 ECTS of the in total 180 ECTS. This thesis is the end-product of the research I carried out over the period April-June 2019. The subject of the research I've done is, shell buckling and demonstrating the buckling process until just before buckling. It has been investigated whether the process of the formation of buckles can be demonstrated with a short movie.

The target audience of this thesis are fellow students and students who want to go further with investigating shell buckling or want to make a short movie in which the process of buckling is demonstrated. The reader is expected to have the knowledge associated with a completed scientific bachelor's degree in Civil Engineering.

During the research, I am guided by my supervisors dr. ir. P. C. J. Hoogenboom and dr. ir. R. Abspoel, who both work for TU Delft. I want to thank them both for their support, suggestions and transferring their knowledge throughout the whole research period.

Delft, June 2019
G. (Guido) Hogendoorn,
student number 4570928

Contents

Abstract

By putting a shell into a pressure bench, one can derive the actual failure load due to local buckling and the buckled shape. However, performing a pressure test doesn't give any information about how these buckles originate during increasing the compressive loading until the critical buckling load.

From earlier theoretical research at the TU Delft, it became clear that the normal membrane force and the curvature of the shell plays an important role in the formation of buckles. From this observation, a hypothesis about the formation of buckles was formulated by dr. ir. P. C. J. Hoogenboom.

The behaviour of the curvature and the membrane force before fatal buckling can be monitored by making contour plots of these two quantities. By making a short movie in which the applied load increases over time, the formation of buckles can be demonstrated and the hypothesis can be checked. In this research, it has been investigated whether it is possible to make the desired contour plots with Ansys and whether it is possible to make a movie of these plots. Based on this goal, the main research question reads as follows:

Is it possible to demonstrate the formation of buckles in a short movie, with the help of contour plots of the mean membrane force and the mean curvature? And can the hypothesis of dr. ir. P. C. J. Hoogenboom be approved?

First, it was tried to use Ansys workbench. However, it turned out that this software wasn't very handy to use in this case, so it was decided to switch to Ansys mechanical APDL as the supervisor was more familiar with this software. During the analysis, a cylindrical shell with arbitrary chosen dimensions has been modelled. The ratio a/t (radius / membrane thickness) was set to a/t = 500.

During the analysis in mechanical APDL, a shell has been created using nodes and elements. These square elements has been created by assigning each corner to a previously created node. Next, the boundary conditions of the top and bottom edge of the shell were set. At both edges, the rotations as well as the displacements were fixed, except the displacement in axial direction (z-direction) at the top edge. After this, a distributed load along the top edge was added.

First, the shell was analysed in a linear calculation, using the outcomes of this calculation, the first mode shape has been determined in a buckling analysis. The first buckling mode has been scaled and added to the perfect shell. By doing this, a shell with imperfections was obtained. The shell including the shape imperfections has then been analysed in a non-linear analysis. In this analysis, the applied load was increased over time. The non-linear analysis was stopped, just before divergence.

The contour plots were created, based on the results of the non-linear analysis. This was done using a lot of APDL code and calculations who were manly based upon geometry. The complete code used for the analyses and creating the plots can be found in appendix 1.

It isn't possible to create an animation of the contour plots in which the plots are shown for every sub-step, yet. Only the contour plots of the last sub-step were shown. However the contour plots of every single sub-step can be obtained by specifying a sub-step number in the APDL script. By editing each plot of every sub-step into a movie maker program, a kind of "stop-motion movie" can be obtained. This means that the first part of the research question can be answered with: "yes". Yes, it is possible to make a movie of the changing membrane force and curvature to demonstrate the formation of buckles.

The (enlarged) deformed shape of the last sub-step, shows some deviations from what is expected from theory. The buckling length is much larger compared to the theoretical buckling length and also the buckled shape is much different. Comparing the curvature based upon the deformations and the curvature plot, one can conclude that the curvature shown in the plot is much bigger than what is expected from the deformations (based upon Sagitta). This observations raises the question, whether the contour plot for the curvature is correct.

By looking at the plot of the mean membrane force, it can be concluded that the non-linear calculation started to divergence too early. This can be concluded from the fact that the applied force in the last sub-step is much smaller than the critical buckling load (more than 10%), also after taking into account the effect of the shape imperfections. An important aspect to investigate in further research, is to find out what the divergence of the non-linear calculation has caused, as it can't be caused due to buckling.

From the contour-plot of the mean membrane force, it can be seen that the normal membrane force becomes up to 10% larger than the applied load, just before divergence. The mean curvature becomes up to 20% smaller, than the original curvature, just before divergence. Putting these observations into the formula for the critical buckling load, the ultimate buckling load (obtained during experiments) can be up to 1.32 times smaller. This factor is much smaller than experimental research shows. The most important cause for this, is the fact that buckling wasn't reached yet. So, the membrane force and the curvature may change further and the factor may tend to the one obtained from research, which is 4.00 for $a/t < 500$.

The most important recommendations for further research are:

- Determine the cause of the divergence of the non-linear analysis before buckling is reached.
- Determine whether the obtained contour plots of the curvature are correct or find an explanation why they doesn't comply with the Sagitta-calculation.
- Determine whether it is possible to make an automatically generated animation of the contour-plots of all the different sub-steps in Ansys, rather than only display the contour plot of the last sub-step or making print-screens of every single sub-step.
- Determine whether the APDL-code used in this research (see appendix 1) can be simplified or made more efficient.
- Determine why the buckling shape doesn't comply with the theory and whether the change of some parameters (i.e.: dimensions, magnitude of imperfections or material parameters) will result in a deformed shape which complies with the theory.
- Making the desired movie in which the formation of buckles in shells is demonstrated.

1 Introduction

In order to get insight into the buckling behaviour of shells, one can perform a pressure test. By putting a shell into a pressure bench, it can be investigated whether the theoretical buckling load corresponds to the actual failure load. Experiments from the past show significantly lower critical buckling loads compared to the theoretical values. It is assumed that for shell structures, shape imperfections have a major impact on the critical buckling load. This is due to the fact that the shape imperfections are relatively large compared to the membrane thickness. Besides, experiments also show the shape of the failed shell.

However, in a pressure test, it is very difficult to get understanding about the process before the first buckles occurs. This is mainly due to the fact that the displacements are extremely small and invisible to the naked eye before buckling. At a certain moment, the buckles originates very suddenly throughout the shells membrane. When this buckling failure occurs, the deformations can be very big and are certainly visible.

From theoretical models, it can be shown that the changing membrane force and the changing curvature of the membrane play an important role during the formation of buckles. This is shown earlier by a small research at the TU Delft (Hoogenboom). From this research, dr. ir. P. C. J. Hoogenboom came up with the following hypothesis about the formation of buckles in shells:

> "When a small load is applied the shell deforms in a buckling mode. The buckling mode increases the shape imperfections of the shell. The deformation is very small and invisible to the naked eye. Nonetheless, the deformation changes the curvature, in some locations the curvature has become larger and in other locations the curvature has become smaller. It also changes the membrane forces. Inward buckles have extra compression and outward buckles have extra tension. When the load is increased the curvatures and membrane forces change further. At some location the curvature becomes small and the compression membrane force becomes large. At this location a local buckle starts. It has a larger length than the earlier buckling mode. This local buckle grows quickly and other buckles occur next to it and this spreads through the shell in a second. The shell collapses."

For educational reasons, there is a need to demonstrate what happens between the moment of starting the compressive axial loading and failure due to buckling and whether the above stated hypothesis can be correct or not. By using a computer simulation, the membrane forces and the curvature can be made visible with the help of contour plots. These contour plots can be displayed while applying an increasing compressive loading along the top circumference of the shell. In this way, the time between non-loading an fatal buckling is simulated. As the contour plots are time-dependant, it is needed to show both plots into a short movie. The goal of this move is to demonstrate what happens during the formation of buckles and to verify whether the above stated hypothesis can be correct. In this research, it has been investigated whether it is possible to make such a movie or not.

The main research question which will be answered in this thesis, reads as follows:

Is it possible to demonstrate the formation of buckles in a short movie, with the help of contour plots of the mean membrane force and the mean curvature? And can the hypothesis of dr. ir. P. C. J. Hoogenboom be approved?

The simulation is done by using the computer program Ansys. With this software, several kinds of analyses can be done. For example, linear elastic, linear buckling and non-linear. As it is not possible to plot the desired quantities for the contour-plots directly, it is needed to use APDL script to create the desired contour plots of the membrane force and the curvature.

In the second chapter of this thesis, a very brief explanation of shells, the buckling of shells and experimental results is given. The next chapter (chapter 3) is about the steps taken for the simulation in Ansys. The first simulation was done in Ansys workbench. However, it became clear that doing the analyses in Ansys workbench wasn't the best idea. So, halfway the research time it was decided to switch to another program within the Ansys software package. This was Ansys mechanical APDL. The performed steps in Ansys mechanical APDL can be found in chapter 4. The created contour-plots, as well as a discussion and conclusion on the contour plots, is shown in chapter 5. Finally, an overall conclusion of this thesis, as well as recommendations for further research are written down in chapter 6.

2 Buckling of shells

In this chapter, the definition of a shell is given (§2.1). Also, the theory of buckling can be found in this chapter (§2.2). Next, some experimental results in relation to shape imperfections are discussed in §2.3. Finally, the importance of the membrane force and the curvature are discussed in §2.4.

2.1 What is a shell?

A shell is a thin, curved rigid structure. The thickness of a shell must be small compared to its other dimensions. The curved shape of a shell is related to the way the loads are transformed to the supports. In this case, a, cylindrical shaped shell has been analysed. Practical examples of these kinds of shells are soda cans, steel barrels (see figure 2.1), etc.

Figure 2.1 – Steel barrel: an example of a cylindrical shell (Distillers)

As already said, the ratio between the thickness and the main dimensions of the structure are very important to determine whether a structure can be called a shell or not. In the case of a perfect cylinder, with a radius a and a thickness t, this ratio is $\frac{a}{t}$ (also called slenderness).

The boundaries for this ratio are assumed to be $30 \leq \frac{a}{t} \leq 4000$. This assumption was made by the supervisor, dr. ir. P. C. J. Hoogenboom. The meaning of the lower and upper boundary are as follows: when the ratio a/t becomes smaller than 30, the structure doesn't act like a shell because the shell membrane is too thick compared to the radius a. On the other hand, when the ratio becomes larger than 4000 the structure is too weak, which means that it isn't barely able to carry its dead weight.

2.2 Buckling: critical load, buckling length and shape

Buckling can occur in structures or parts of structures which are loaded under compression. In case of thin walled structures (like shells) one often talks about local buckling. This is due to the fact that the buckling is not related to Euler's buckling criterion (in which the so-called buckling length plays an important role). In case of local buckling, the buckles occur as a result of big compressive forces in a relatively thin material.

According to theory, local buckling of cylindrical shells occurs when the normal compressive force exceeds n_{cr}. This normal compressive force is defined per unit of length along the circumference of a cross-section in the cylinder.

$n_{cr} = -\frac{1}{\sqrt{3(1-\nu^2)}} \cdot \frac{E \cdot t^2}{a}$ (Hoogenboom)

In this relation, ν is the Poisson's ratio, E is the Young's modulus, t the shell thickness and a the radius of the cylinder.

For practical and realistic values of the Poisson's ratio ($\nu \approx 0.35$) the expression for the maximum compressive force per unit of length becomes:

$$n_{cr} = -0.6 \cdot \frac{E \cdot t^2}{a}$$

The buckling length of shells, in the axial direction, is described by:
$l_{buc} = 1.7 \cdot \sqrt{at}$ (Hoogenboom)

If the normal force exceeds the critical value n_{cr}, the shell fails due to buckling and can have a buckled shape like the shell of figure 2.2.

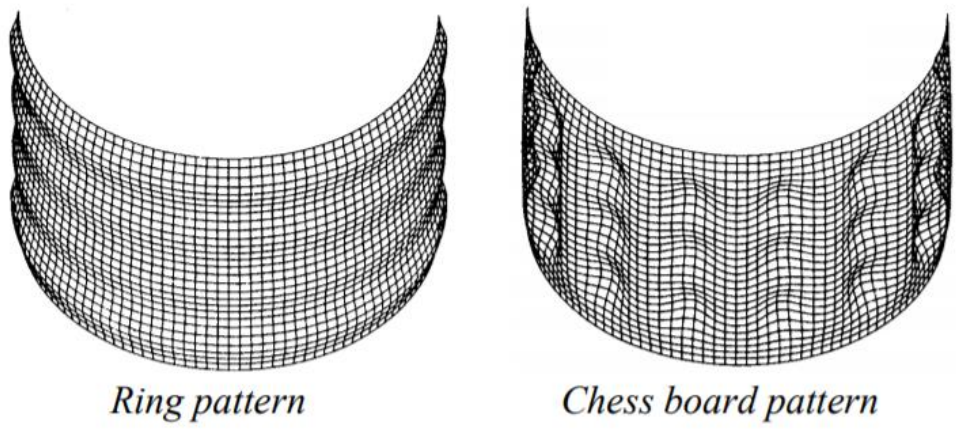


Figure 2.2 – theoretical buckling shape (Hoogenboom)

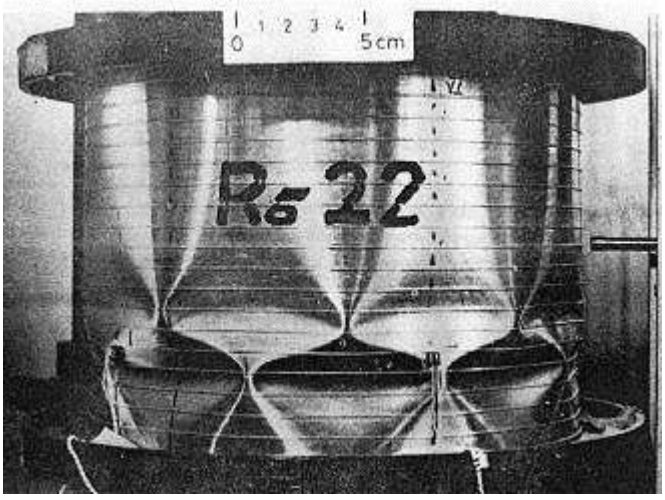


Figure 2.3 – "Yoshimura pattern" of a buckled shell (Wright, 2005)

When the buckling goes further, the pattern of figure 2.3 can be obtained. The buckled shape in this figure, is called the "Yoshimura pattern" (Hoogenboom).

2.3 Shape imperfections

As already said in the introduction, the experimental critical buckling load of shells is much lower than what might be expect from theory. Dependant on the ratio a/t, the experimental "ultimate load" can be 6 times smaller than the theoretical "critical load". The graph of figure 2.4 shows the outcome of 172 experiments in which shells with different a/t were axially loaded. As the factor between n_{ult} (ultimate load) and the n_{cr} (critical load) is less than 1.0, it can be concluded that every shell started to buckle before the theoretical critical buckling load was exceeded.

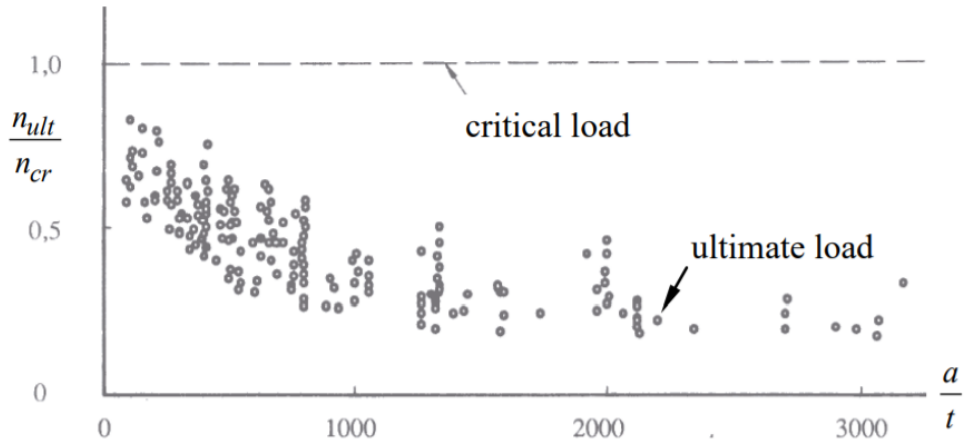


Figure 2.4 – Experimental ultimate loads of 172 axially loaded aluminium cylinders (Hoogenboom)

The much lower ultimate load which is obtained during experiments, can be explained by the fact that a shell is very sensitive to shape imperfections. These shape imperfections are relatively large compared to the membrane thickness t. As the shape imperfections are very random and not known on beforehand, the factor between the ultimate load and the critical load can have a very wide spread for constant a/t.

Shells contain always shape imperfections, as every surface isn't completely perfect. Furthermore, due to the relatively low thickness of the shell, a shell is also very sensitive to shape imperfections being induced during its lifetime, by for example (small) impacts.

2.4 Importance of membrane force and curvature

By setting up a so-called stick model and deriving some applicable formulas from this model, dr. ir. P. C. J. Hoogenboom came up with a mathematical model which describes the relationship between the membrane force and the displacements and the curvature and the displacements. These relationships are shown in the graphs of figure 2.5.

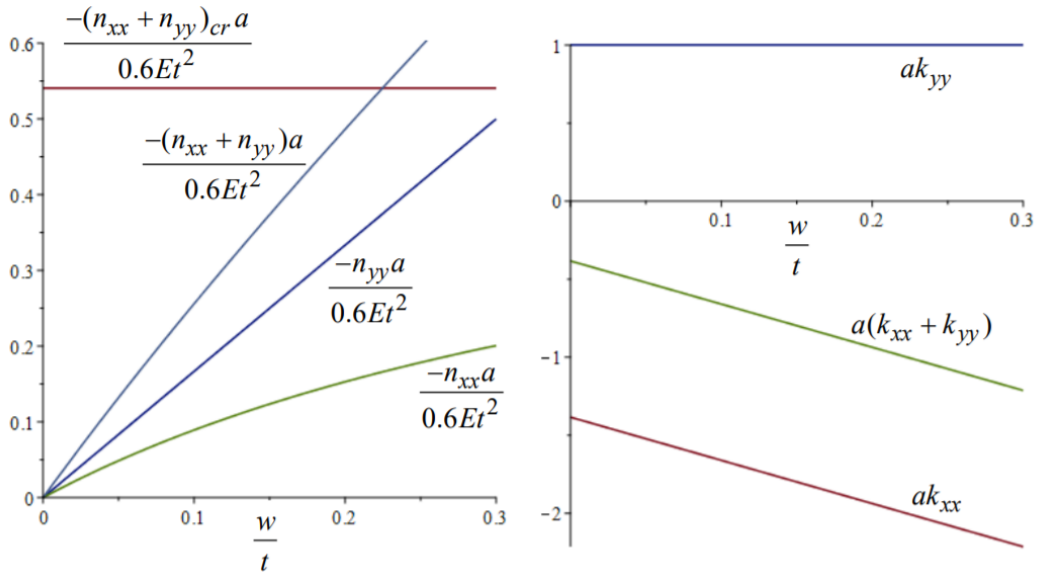


Figure 2.5 – relationship between normal force and displacement and between curvature and displacement. (Hoogenboom)

From the graphs of figure 2.5 it can be concluded the expression $\frac{-(n_{xx}+n_{yy})a}{0.6Et^2}$ becomes extremely bigger when the displacement gets bigger. In the right graph, one can see that the expression ak_{xx} gets smaller and ak_{yy} stays constant for increasing displacement.

3 Analysis in Ansys Workbench

In order to obtain the desired contour plots, a computer simulation was done using a finite element analysis in Ansys and its programming language APDL. In this chapter the procedure of this analysis will be discussed.

The analysis was done by using Ansys workbench. In order to model the perfect shell, add the imperfections, performing a non-linear analysis and making the plots, several chronological steps were needed. These steps can be found in paragraph §3.1. All the steps are discussed in the paragraphs following §3.1.

3.1 Workflow of the analysis

The analysis in Ansys workbench will consist of several chronological steps. These steps will be discussed separately in the following paragraphs (§3.2 till §3.5). A workflow of all the steps involved can be seen in figure 3.1.

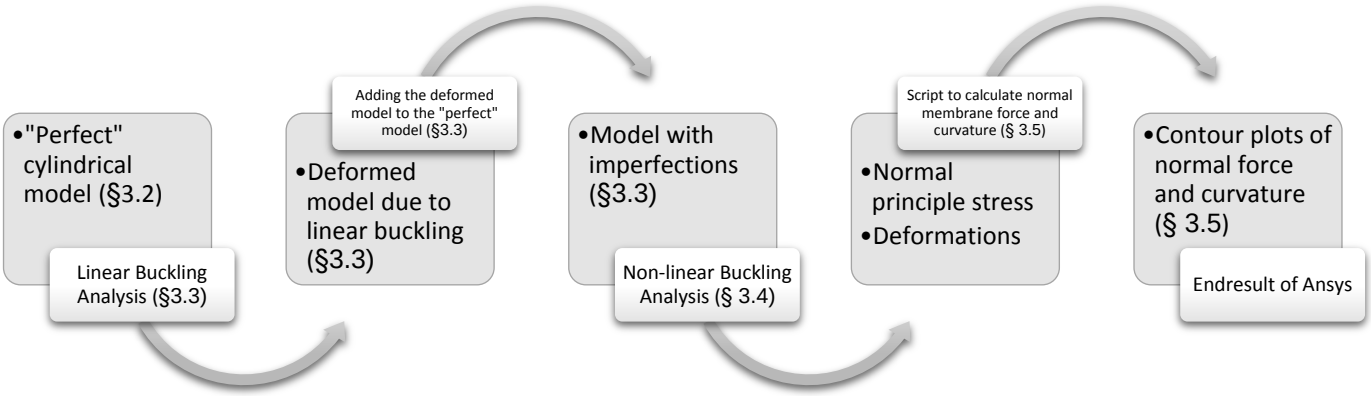


Figure 3.1 – *workflow of finite element analysis in Ansys*

In the Ansys workbench, above stated workflow looks a bit different. The workflow in Ansys workbench can be seen in figure 3.2.

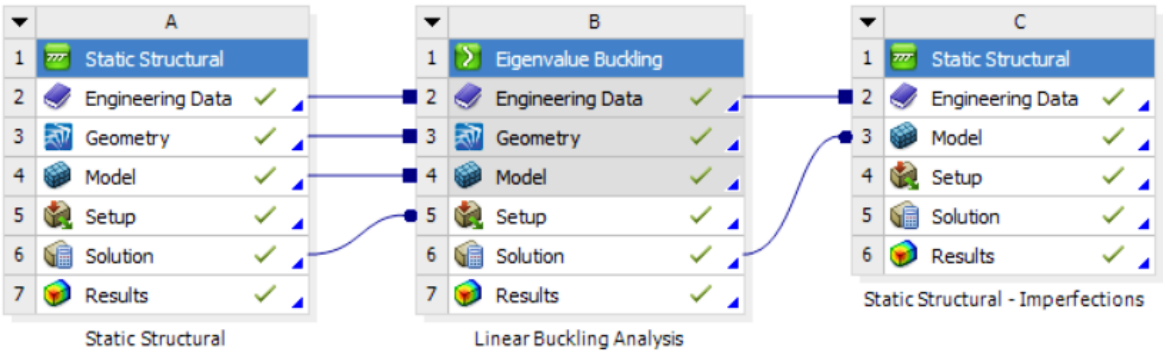


Figure 3.2 – workflow Ansys workbench

3.2 Model of cylindrical shell

The model which is used in the simulation is a cylinder made out of structural steel, which has an E-modulus of $2.1 \cdot 10^5$ N/mm^2 and a Poisson's ratio of $\nu = 0.35$. The dimensions of the cylinder are defined by the radius a, shell thickness t and an imperfection amplitude d. This imperfections will be added later.

In the finite element analysis, a cylindrical shell with the following dimensions has been modelled in the Geometry tab (see figure 3.2, cell A3):

radius $a = 60$ mm

The shell thickness t must be chosen in correspondence to the condition $30 \leq \frac{a}{t} \leq 4000$. For this analysis $\frac{a}{t} = 500$ (which is chosen arbitrary and is between the boundaries for a shell). So, the shell thickness t must be equal to:

$t = \frac{60}{150} = 0.12$ mm

However, the shell is first modelled without a thickness. Otherwise it is not possible to create a proper mesh that is needed for analysing a shell structure.

As the analysis focusses on local buckling, the length of the cylinder doesn't matter. The length of the cylinder will only have an effect on the global buckling of the structure as a whole. This means that the length can be chosen arbitrary, and is chosen to be equal to $l = 250$ mm in this particular case.

After double-clicking on the "model"-tab in Ansys workbench (see figure 3.2, cell A4) a new window opens. The new opened window is called "Ansys mechanical". In this window, most of the next steps are carried out.

First, the last needed dimension parameter can be added to the model. This is the shell thickness t, which is equal to 0.12 mm.

The next step is to set up the mesh of the cylindrical model. This can be done under the mesh tab in *Ansys mechanical*. In order to get a 2D meshing for shells, the method *"MultiZone Quad / Tri Method"* is chosen. This results in the meshed model as shown in figure 3.3.

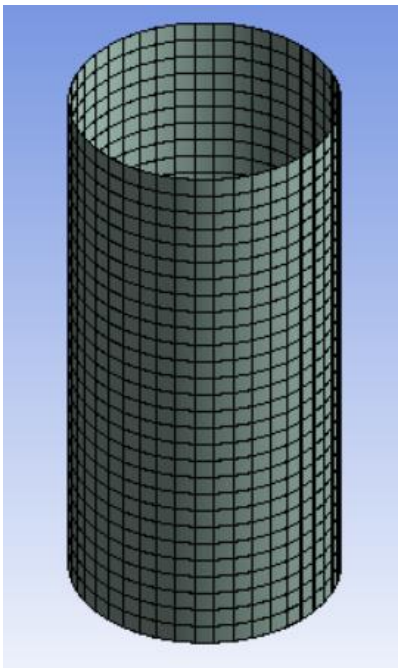


Figure 3.3 – meshed cylindrical shell

The meshing method used, results in nodes and elements who are lying in the middle of the shell membrane thickness t. All the material parameters, stresses and forces over the whole thickness were captured in these 2D elements (the third dimension would be the thickness). This means for example that the stress distribution over the thickness t is represented by its resultant. See figure 3.4.

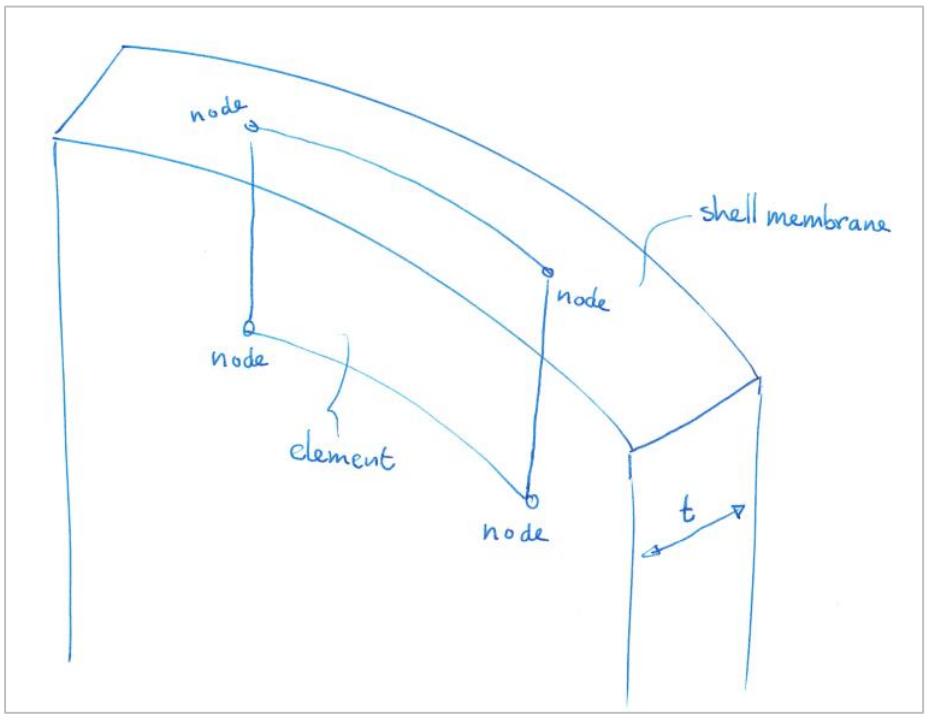


Figure 3.4 – elements and nodes in the middle of the shell membrane

Next, the boundary conditions and the loading has been set. The model has two edges on which boundary conditions can be applied. These edges are the top and the bottom circumference. At the top circumference, the shell is not allowed to displace in x- and y-

direction. Only displacements in z-direction and rotations in any direction are allowed at the top.

For the circumference at the bottom any displacement isn't allowed, however rotations at the bottom are allowed.

The loading of the model is applied along the top circumference of the cylinder. During the analysis in which the buckling modes are calculated, the magnitude of the loading isn't important. So, a unit load of 1 N/m was applied along the full circumference of the cylinder in negative y-direction.

After this, the "static structural" analysis of column A (figure 3.2) can be solved. With this solution, a "linear buckling analysis" can be done (see column B, figure 3.2). This analysis will be described in the following paragraph.

3.3 Linear buckling and imperfections

The model with imperfections has been obtained by determine the deformations due to linear buckling. This is done by putting the perfect shell into the linear buckling analysis which is shown in column B of figure 3.2.

In figure 3.2 it can be seen that for this analysis the "Engineering Data", "Geometry" and "Model" are directly coupled to the "static structural" analysis (see the blue lines). Furthermore, the "Solution" of the "static structural" analysis is used as the "Setup" for the "linear buckling analysis".

Finally, the linear buckling analysis results into different buckling modes. The number of to be considered buckling modes can be set, but in this case only the solution to the first buckling mode is enough. Ansys provides the deformed shape of the buckling mode and a factor called the "load multiplier". This is the factor between the applied load (§3.2) and the critical load at which the fatal buckling occurs. In this case, this factor is equal to the fatal buckling load as the applied load was a unit load. Figure 3.5. shows the shape of buckled cylinder in the first buckling mode. The load multiplier for the first mode is equal to 18,002, so the critical load for this buckling mode is 18,002 N/m.

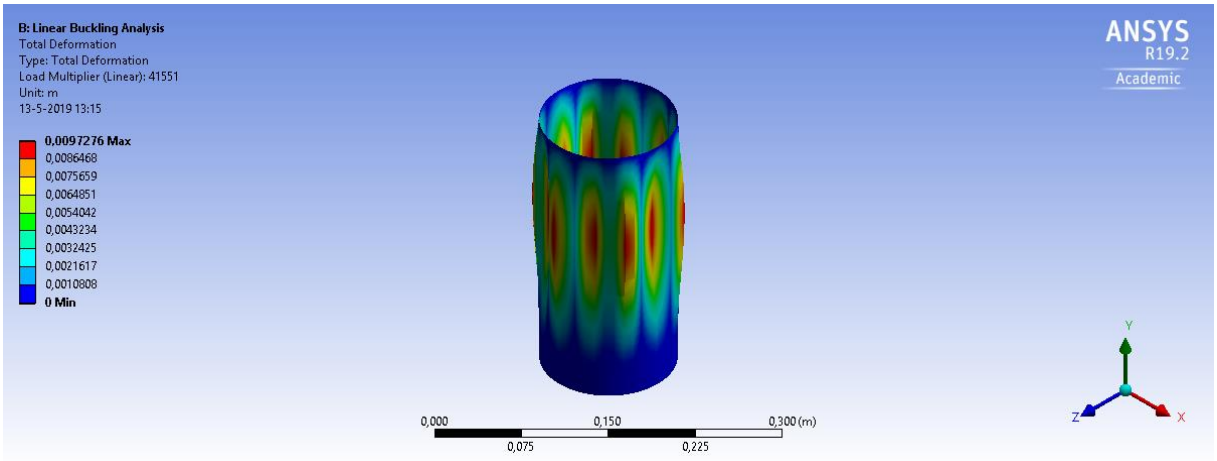


Figure 3.5 – buckled shape of the first buckling mode

To obtain the amplitude d of the imperfections, the deformations are multiplied by approximately $0.5\,t$. This results in a amplitude of the imperfections of $d = 0.5 \cdot t = 0.5 \cdot 0.12 = 0.06$ mm. The maximum amplitude of the deformed shape of the first buckling mode is equal to 0.0079972. This means that the factor between the desired magnitude of the imperfections and the amplitude of the buckled shape is equal to $\frac{6 \cdot 10^{-5}}{0.0097276} \approx 0.00617$.

The model with imperfections can finally be obtained by scaling the buckled shape by 0.0075 and linking it to the "Model" cell of a third analysis (see cell C3, figure 3.2). This scaling can be done by a right click on the "Solution" cell of the buckling analysis (cell B6, figure 3.2), clicking on properties and then setting the "Scale Factor" to 0.00617.

Now, a model with imperfections with a maximum amplitude of approximately 0.06 mm is obtained. The maximum amplitude isn't exactly 0.06, but that doesn't matter as the amplitude of the imperfection is chosen arbitrary and only the order of the magnitude is important.

3.4 Non-linear buckling analysis

Now, the model has its imperfections, the actual intended non-linear buckling analysis can be performed. To do this, a mesh has to be created, and also the load case and the boundary conditions has to be determined. The meshing and boundary conditions of this model are the same as for the "perfect" cylinder.

The analysis is done with an increasing load up to buckling. From theory it may be expected that the buckles will occur when a load of $n_{cr} = -30.24$ N / mm along the circumference is reached. However, the applied imperfections (§ 3.3) will lower this critical load, so the buckles will show up earlier.

As said, the load increased during the analysis. At the beginning the load was equal to $n = 0$ N/mm, after 45 seconds the load is increased linearly to an end-value of $n = -35$ N/mm. The time span of 45 seconds gives the viewer enough time to see what happens during loading. At the end of the loading, the fatal buckles surely must have occurred as the critical loading – for a cylindrical shell without imperfections – is exceeded.

3.5 Contour plots of mean membrane force and mean curvature

The next – and also the most critical – step is to create the desired contour plots. The difficulty of this step lies in the fact that the desired contour plots cannot be generated directly.

The plot of the mean membrane force $\left(\frac{n_{xx}+n_{yy}}{2}\right)$ at the middle of the thickness can be generated by using the normal principle stresses at the exterior and the interior (outer fiber). At the exterior and interior surface of the shell, the shear stress is 0. From this, it follows that one of the principle stresses always points perpendicular to the surface of the shell. In addition, from the load case it follows that at every location any of the three principle normal stresses (n_1, n_2 or n_3) is equal to 0.

In order to calculate the membrane stress at the middle of the thickness, the following formulas can be used:

$$n_{xx} + n_{yy} = \sigma_{xx} \cdot t + \sigma_{yy} \cdot t$$

$$n_{xx} + n_{yy} = (\sigma_1 + \sigma_2 + \sigma_3) \cdot t$$

where one of the principle normal stresses is always equal to 0.

$$n_{xx;m} + n_{yy;m} = (\sigma_{1;b} + \sigma_{2;b} + \sigma_{3;b} + \sigma_{1;t} + \sigma_{2;t} + \sigma_{3;t}) \cdot \frac{t}{2}$$

where subscript m stands for *middle*, b for *bottom* and t for *top*, see figure 3.6.

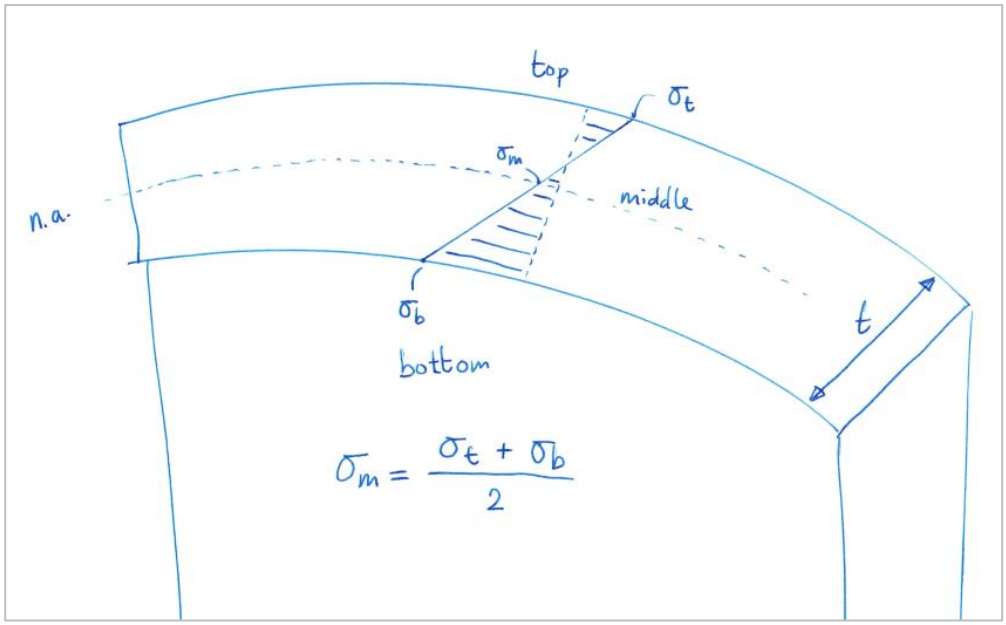


Figure 3.6 – membrane stress: middle, bottom and top

So the desired result can be calculated by dividing by 2:

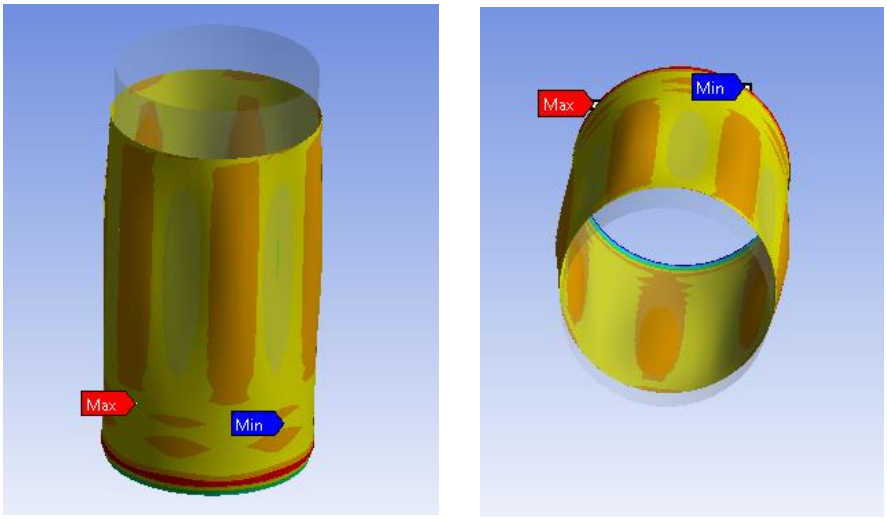
$$(n_{xx;m} + n_{yy;m}) \cdot \frac{1}{2} = (\sigma_{1;b} + \sigma_{2;b} + \sigma_{3;b} + \sigma_{1;t} + \sigma_{2;t} + \sigma_{3;t}) \cdot \frac{t}{4}$$

Contour plots based on formulas and solution parameters (like σ_1) can be generated by inserting a “user defined result”. This user defined result can be defined with an expression. The symbol Ansys uses for σ_1 is **S1**. However, it is not possible to distinguish between an outer fibre at the exterior or an outer fibre at the interior of the shell. So, it is not possible to calculate the normal membrane force at the middle of the membrane.

Using the expression:

```
(S1 + S2 + S3) * 0,00012 / 2
```

Results in a plot, with different patterns at the exterior and interior. On a convex surface, the membrane force is bigger than on a hollow surface. At the exterior and interior of the shell, the location of a convex or hollow surface are opposite. This means that also the pattern of the contour plot are opposite. In other words, there is not one averaged value given for the middle of the shell. See figure 3.7.



a) exterior *b) interior / exterior*

Figure 3.7 – contour plot of membrane force, not averaged over the thickness of the shell.

Another way to make user defined contour plots is by using APDL commands. However, using this doesn't give the desired results.

As the first supervisor is more familiar with using the Ansys mechanical APDL software, it was decided to switch to this software rather than continue trying to solve the problem with Ansys workbench.

3.6 Conclusions about Ansys workbench

Ansys workbench is good software to analyse shells, buckling of shells and making contour plots. However, it turned out that Ansys workbench wasn't able to create a contour plot of the membrane stress at the middle of the thickness. This is mainly the result of the fact that it isn't possible to distinguish between the principle force (**S1**, etc) at the middle, top or bottom of the membrane thickness (see figure 3.6). It may be possible to create the desired plots with APDL commands in Ansys workbench, however this did not work yet.

4 Analysis in Ansys mechanical APDL

As it didn't seem to be easy to create the desired contour plots in Ansys workbench, it was decided to switch to Ansys mechanical APDL. This software allows the user to create a command file (in APDL) and run the file in one go. This command file was created by using *notepad*. Instead of saving the notepad-file as a *.txt*, the file was saved as a *.mac* in order to run it in Ansys mechanical APDL.

The file used for the analyses and creating the desired contour plots can be found in appendix 1. In the subsequent paragraphs, each part of the script will be explained. For every part, the explanation is accompanied with the related APDL script, which is a snippet of the whole script.

Most of the used commands were found on the website: www.sharcnet.ca

4.1 Input parameters

The analysis in Ansys mechanical APDL was performed by using the same input parameters as used in the Ansys workbench. These parameters are as follows:

```
E   = 2.1e5          ! N/mm2     Young's modulus
nu  = 0.35           ! -         Poisson's ratio
h   = 250            ! mm        length
a   = 60             ! mm        radius of curvature
t   = 0.12           ! mm        thickness
q1  = 1              ! N/mm      load along top edge
rho = 7000e-9        ! kg/mm3    mass density
n   = 50             ! -    number of elements in the x or y direction
d   = t/2            ! mm        amplitude of imperfection
pi  = acos(-1)       ! -         pi-value
```

A new parameter in this list is **n**., The number of elements in both the direction along the circumference and the height of the cylinder was determined with **n**. This number of elements is an important parameter to create the mesh. Both the nodes and the elements between these nodes were created by using parameter **n**, see § 4.2 and § 4.3

4.2 Creating nodes

In the previous paragraph, the parameter **n** was defined. This parameter is equal to the number of elements along the circumference of the cylinder and the number of elements in z-direction (the height of the cylinder).

First a loop is created to determine the elevation along the z-axis. In this loop, the z-coordinate is equal to:

$$z_i = i \cdot \frac{h}{n}$$

Where h is the height of the cylinder, n the number of elements in the direction of z and i an integer, with $\boldsymbol{i} \in [\boldsymbol{0}, \boldsymbol{n}]$

Within the loop to determine the elevation along the z-axis, a second loop is created to divide each circular circumference into n elements. For each elevation along the z-axis, the circular pattern of nodes can be created as follows:

$$\alpha_j = j \cdot \frac{2\pi}{n}$$

Where $\frac{2\pi}{n}$ is equal to the angle between each node with respect to the centre of the circle in radians. The definition of the nodes and the angle α_j, can be seen in figure 4.1.

$$x_j = a \cdot \cos(\alpha_j)$$
$$y_j = a \cdot \sin(\alpha_j)$$

With $\boldsymbol{j} \in [\boldsymbol{0}, \boldsymbol{n-1}]$

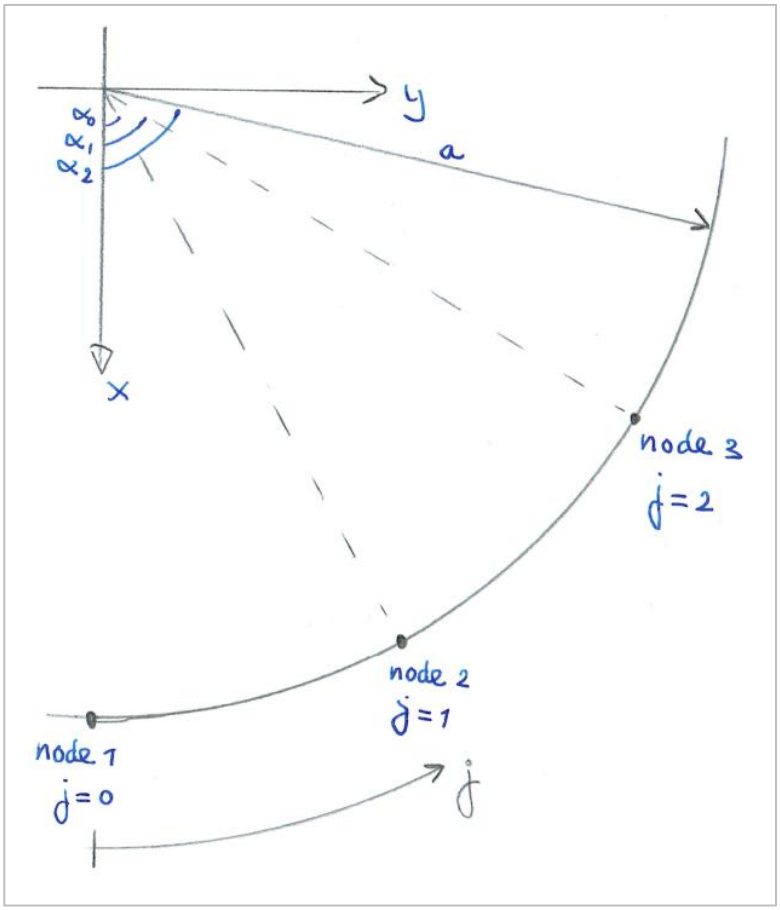


Figure 4.1 – definition of the angle and the nodes along the circle with radius a

Above stated formulas eventually result into the following APDL code:

```
*DO,i,0,n                          ! insert nodes
 *DO,j,0,n-1
   x=a*cos(j*2*pi/n)
   y=a*sin(j*2*pi/n)
   z=i*h/n
```

```
   N,,x,y,z,,,
 *ENDDO
*ENDDO
```

This code results into the node numbering shown in figure 4.2. For every arbitrary node, the node number can be calculated as follows: $\boldsymbol{nnode = j + i \cdot n + 1}$.

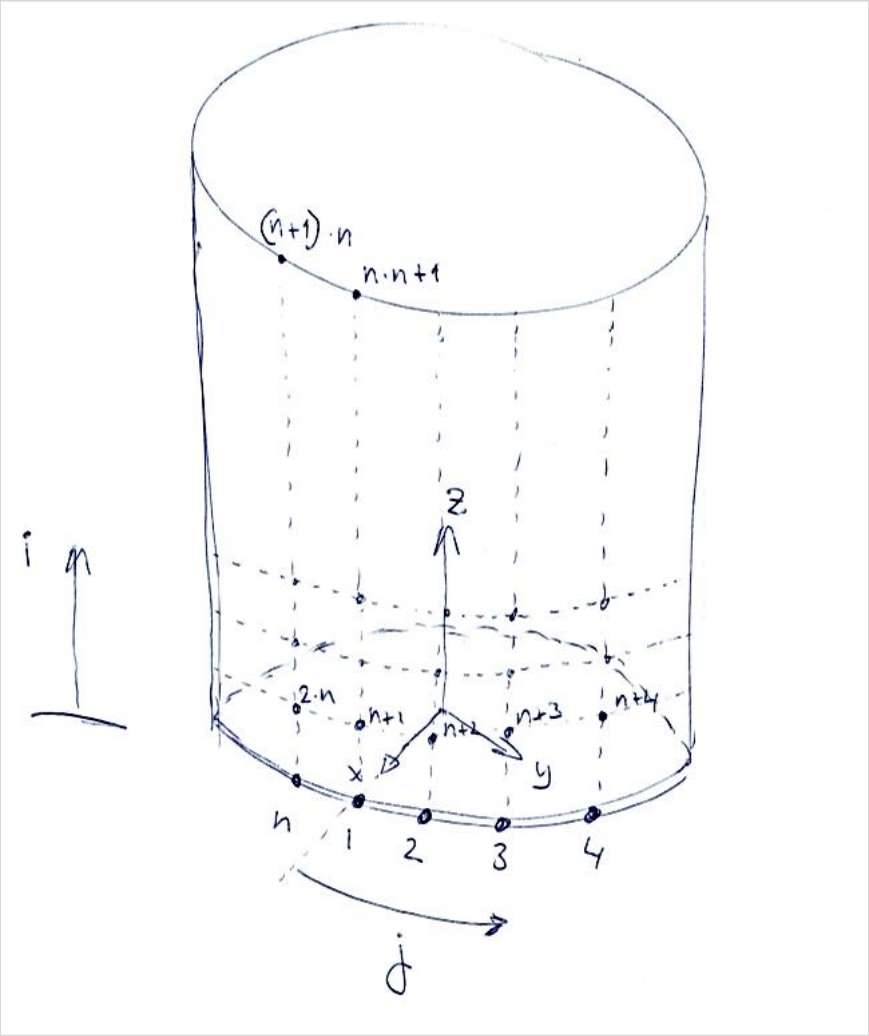


Figure 4.2 – numbering of the nodes

4.3 Creating elements

The next step is to create elements between the previous created nodes. The local orientation of the axis is defined as shown in figure 4.3.

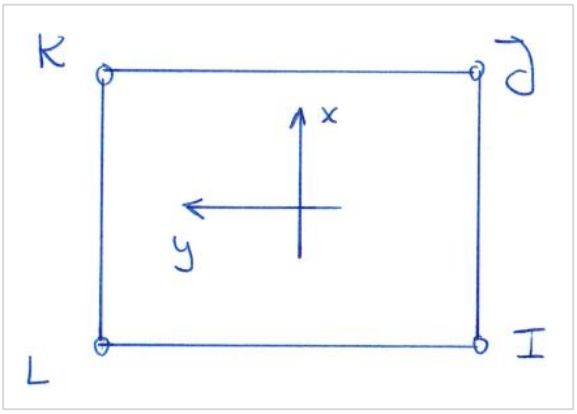


Figure 4.3 – definition of the local axis and the default naming of the nodes

The elements can be created using the APDL code: **`E,I,J,K,L`** where **`E`** stands for "element" and **`I`**, **`J`**, **`K`** and **`L`** represent the number of the node, corresponding to figure 4.3.

For each element, the node number of the lower left node (node L) is determined, being parameter **`L`**. The node number of the 3 other nodes can be determined from the node number of node L. In figure 4.4, the relation between the node number of node L and I, J and K can be seen.

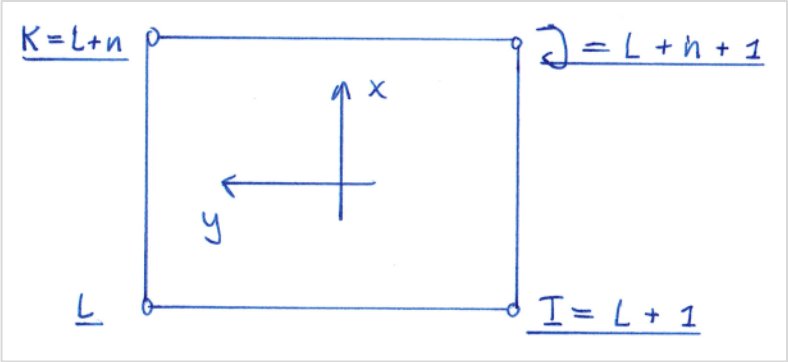


Figure 4.4 – relation between node L and I, J and K.

This means that each element (except the last one of each circumference) can be created using the following script: **`E,L+1,L+n+1,L+n,L`**

By using a double loop, each element along the circumference and at each elevation in z-direction can be created. This is done by looping through the elevation with $\boldsymbol{i} \in [\mathbf{1}, \boldsymbol{n}]$ and a second loop to create $n-1$ elements along the circumference at each elevation. For this loop, parameter j with $\boldsymbol{j} \in [\mathbf{1}, \boldsymbol{n}-\mathbf{1}]$ is used. The n-th element along the circumference needs to be created with a different code, because the last nodes needs to be connected to the first nodes, see figure 4.5.

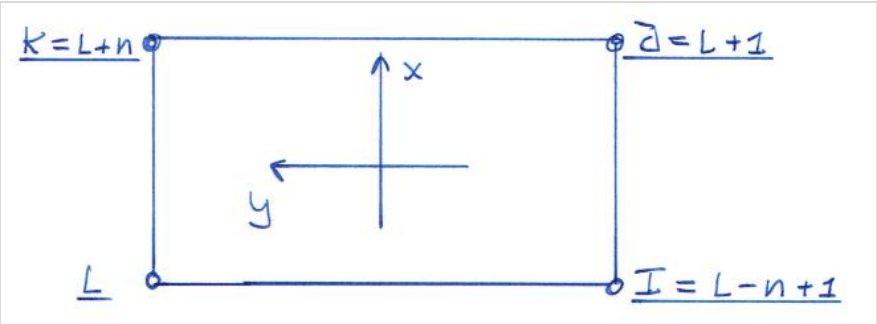


Figure 4.5 – relation between node L and I, J and K for the last element at each level of i.

From figure 4.5, it can be concluded that the script to create the last element at each circumference reads: **`E,L-n+1,L+1,L+n,L`**

From the definition of i and j, it follows that he node number of node L can be calculated as follows:

$L = (i-1) \cdot n + j$ for each 1st till 19th element along the circumference
$L = i \cdot n$ for each 20th element along the circumference

This eventually results in the following APDL code:

```
*DO,i,1,n                       ! insert elements
 *DO,j,1,n-1
  L=(i-1)*n+j
  E,L+1,L+n+1,L+n,L
 *ENDDO
 L=i*n
 E,L-n+1,L+1,L+n,L
*ENDDO
```

4.4 Boundary conditions and loading

Along the top and bottom edge several boundary conditions must be set. The boundary conditions are assigned to the nodes. This means that for each node with one or more restrictions these restrictions must be specified. The easiest way to do this, is with a loop through the nodes along the bottom edge and another loop for the top edge.

For the bottom edge, it holds that rotations and displacements in all three directions are not allowed. The node number of the bottom edge can be called i with $\boldsymbol{i} \in [\boldsymbol{1}, \boldsymbol{n}]$. At the top edge only displacements in the z-direction are allowed, all the other degrees of freedom are again restricted. Also the node number at the top edge can be called i, however this time with $\boldsymbol{i} \in \left[\boldsymbol{n^2+1}, \boldsymbol{n \cdot (n+1)}\right]$

The boundary conditions can be set by using the following APDL code:

```
*DO,i,1,n                     ! boundary conditions bottom edge
 D,i,UX,0
 D,i,UY,0
 D,i,UZ,0
```

```
 D,i,ROTX,0
 D,i,ROTy,0
 D,i,ROTZ,0
*ENDDO

*DO,i,n*n+1,n*(n+1) ! boundary conditions top edge
 D,i,UX,0
 D,i,UY,0
 D,i,ROTX,0
 D,i,ROTy,0
 D,i,ROTZ,0
*ENDDO
```

In a similar way, the loading along the top edge can be added. As an input parameter, the distributed load **q** (N/mm) is defined. The length of the circumference of the circle is equal to $2\pi \cdot a$, where **a** is the radius of the circle. Multiplying the length of the circumference with the distributed load gives the total load acting on **n** nodes along the top edge. The load per node can thus be calculated as follows:

$$F = q \cdot 2\pi \cdot a \cdot \frac{1}{n} \quad \text{(N)}$$

The load acts in the top edge (so the same node numbers are called as for the boundary conditions on the top edge) and in the negative z-direction. Running the following APDL code, gives the desired loading of the cylindrical shell:

```
*DO,i,n*n+1,n*(n+1)       ! insert load on top edge
  F,i,FZ,-q*2*pi*a/n
*ENDDO
```

4.5 Adding imperfections

Again, the imperfections are added by adding a scaled buckling mode to the original perfect cylinder. To do this, it is needed to do a linear analysis followed by a buckling analysis. In this buckling analysis, the first buckling mode was calculated. In order to perform both analyses, the following script is used:

```
/SOLU                ! compute linear elastic
SOLVE
FINISH

/SOLU                ! compute buckling modes
ANTYPE, MODAL
MODOPT, LANB, 1,,,
MXPAND,20
PSTRES,ON
SOLVE
FINISH
```

After the first buckling mode was known, the shape of this buckling mode was scaled and added to the original shape of the cylinder. The deflection or amplitude of the buckled shape

is stored unity less, with a maximum deflection of 1. In order to obtain a maximum deflection or amplitude of the imperfection (d) of $d = 0.5 \cdot t$. The scaling factor for the deflection or amplitude must be equal to d. This results into an imperfection amplitude of $1 \cdot d = d = 0.5 \cdot 0.12\ mm = 0.06\ mm$. With 1 being the unity less, maximum deflection of the buckled shape and d the scaling factor as well as the desired maximum amplitude of imperfection.

In order to scale the buckled shape and add it to the original cylinder, the following script is used:

```
/PREP7                    ! inserting imperfections
UPGEOM,d,1,1,'file',rst
/SOLU
```

4.6 Non-linear analysis

The next step, was to perform a non-linear analysis. Based on the results of this analysis, the contour plots were made (see §4.7).

In the non-linear analysis, the compressive loading at the top circumference of the cylinder, has been increased in several load steps. Performing the non-linear analysis in this way (increasing the loading in several steps) can be called “force controlled”. The loading-displacement curve of such an analysis can be seen in figure 4.6.

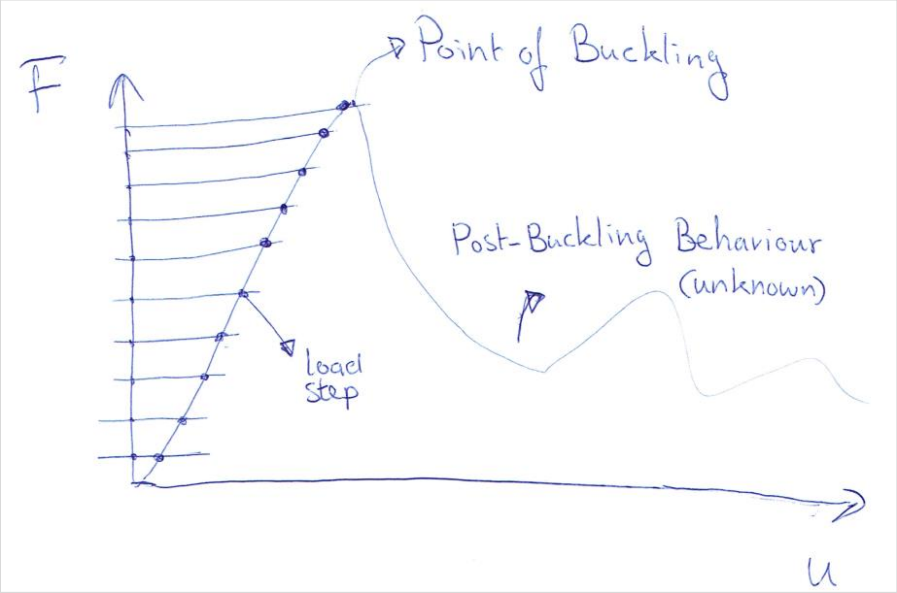


Figure 4.6 – Loading-displacement curve non-linear analysis

In figure 4.6, every horizontal line through the curve represents a load step at which the displacements and other results have been calculated. By zooming-in, the iterative process of the Newton-Raphson method can also be visualized. The basic idea of the Newton-Raphson iteration used during the non-linear analysis is shown in figure 4.7.

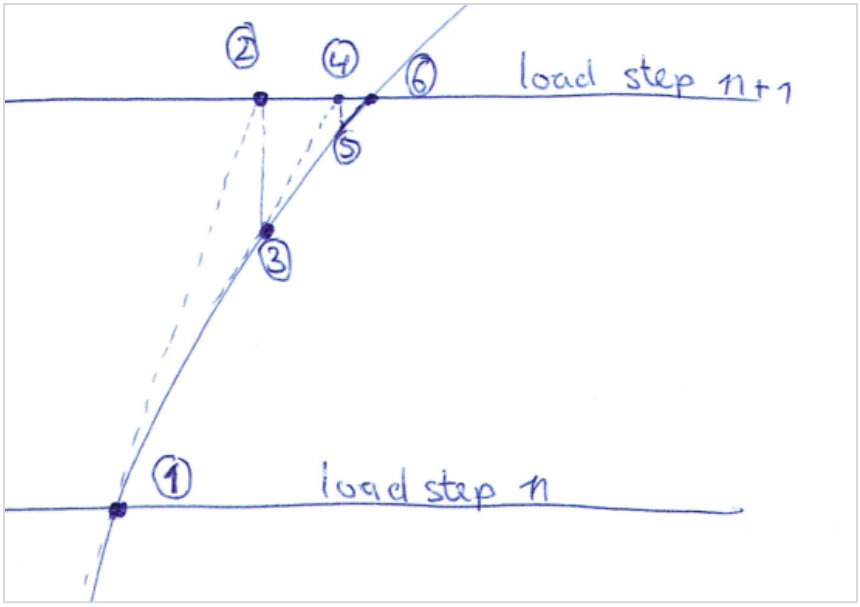


Figure 4.7 – Iterative process of the Newton-Raphson method between load step n and n+1.

Explanation of the points in figure 4.7:
1 *– Solution of load step n, initial guess for load step n+1*
2 *– Intersection of F at load step n+1 and the tangent of point 1*
3 *– Straight line downwards from point 2, until F,u-diagram*
4 *– Intersection of F at load step n+1 and the tangent of point 3*
5 *– Straight line downwards from point 4, until F,u-diagram*
6 *– Intersection of F at load step n+1 and the tangent of point 5 → Solution of load-step n+1 (within pre-set deviation tolerances)*

After each load step, the results were calculated iteratively using the Newton Raphson method. After several iterations, the results were converged at one particular load step. At this moment, the results of the next load step where determined iteratively, etcetera. The maximum number of iterations per load step was set to 100, this means that the analysis was quitted when the result didn't converge after 100 iterations.

The analysis was done until the shell started to buckle. When the shell starts to buckle, the number of iterations needed becomes infinity. As the analysis quitted when the number of iterations exceeds 100, the analysis also stopped when the shell started to buckle.

During the non-linear analysis, the initial loading was set to be equal to 0. During 20 different steps, the load was increased until a value of 30 N/mm. This value is bigger than the critical buckling value. By setting the maximum number of iterations to 100, the non-linear analysis must have been quitted just before buckling. Because, it is not possible to convergence beyond the critical buckling load and thus the maximum number of iterations must have been exceeded.

The above described method can be implemented in Ansys mechanical APDL with the following script:

```
/SOLU
NCNV,0                    ! Do not terminate program if not-converged
ANTYPE, STATIC            ! Static analysis
```

```
NLGEOM, ON                  ! Nonlinear geometry
TIME, 1                     ! Time at the end of load step
nsubstep = 20               ! # of substeps
NSUBST,nsubstep,,           ! Number of substeps
NEQIT,100,                  ! Max. number of iterations
CNVTOL,STAT                 ! Convergence tolerance (default)
RESCONTROL,DEFINE,ALL,1,    ! Write new files at every substep
OUTRES,NSOL,ALL,,,,         ! Write nodal results at every substep
OUTRES,ESOL,ALL,,,,         ! Write element results at every substep
SOLVE
FINISH
```

4.7 Contour plot of the mean membrane force

In order to create the desired contour plots of the mean membrane force $\left(\frac{n_{xx}+n_{yy}}{2}\right)$ and the mean curvature $\left(\frac{k_{xx}+k_{yy}}{2}\right)$, it is needed to write a script, as it is not possible to plot these quantities automatically. In §4.7 the calculation of the mean membrane force is shown. The calculation of the mean curvature is a lot more complex, the calculation methods and assumptions used are showed and written down in §4.8.

As earlier said, the values of nxx and nyy who were used, were the values at the middle of the shells thickness t (see figure 3.4). The values of nxx and nyy can be calculated by multiplication of the stresses with the shell thickness. This results into the following formula:

$$n_{xx} + n_{yy} = (\sigma_{xx} + \sigma_{yy}) \cdot t$$

As σ_{xx} and σ_{yy} uses the global coordinate system, it's not possible to use this formula directly. Instead of using σ_{xx} and σ_{yy}, the principal stresses can be used. The sum of the non-zero principal stresses is equal to the sum of the normal stresses with any other orientation of the coordinate system. This follows from the following formula:

$$\sigma_{1,2} = \frac{\sigma_{xx} + \sigma_{yy}}{2} \pm \sqrt{\left(\frac{\sigma_{xx} + \sigma_{yy}}{2}\right)^2 + \tau_{xy}^2}$$

By performing $\sigma_1 + \sigma_2$, the square-root-term cancels out and the first term becomes $2 \cdot \frac{\sigma_{xx}+\sigma_{yy}}{2}$, which proves that $\sigma_1 + \sigma_2 = \sigma_{xx} + \sigma_{yy}$.

However, in a 3D-situation there is also a third principal stress. This means that one also has to consider σ_3. Taking into account that for this load case at every point any of the principal stresses is equal to zero, the following expressions are valid:

(1): $\sigma_1 + \sigma_2 = \sigma_1 + \sigma_2 + \sigma_3$ (cause at every location any of the principal stresses is zero)
(2): $\sigma_1 + \sigma_2 = \sigma_{xx} + \sigma_{yy}$ (as proven before)
(3): $\sigma_{xx} + \sigma_{yy} = \sigma_1 + \sigma_2 + \sigma_3$ (combination of 1 and 2)

At each node, the value for $\left(\frac{n_{xx}+n_{yy}}{2}\right)$ can be calculated using:

$$\frac{n_{xx} + n_{yy}}{2} = \frac{(\sigma_1 + \sigma_2 + \sigma_3) \cdot t}{2}$$

In APDL code, this becomes: **nxxyy = (s1+s2+s3)*t/2**

By using the following APDL-script, the desired values can be calculated and stored.

```
*DO,i,0,n                          ! calculating (nxx + nyy) / 2
 *DO,j,0,n-1
   nnode = j + i*n + 1
   SHELL, MID                      ! Averged value of TOP and BOTTOM
   *GET,s1,NODE,nnode,S,1
   *GET,s2,NODE,nnode,S,2
   *GET,s3,NODE,nnode,S,3
   nxxyy = (s1+s2+s3)*t/2
   DNSOL,nnode,U,X,nxxyy     ! the desired value will be stored into ux
 *ENDDO
*ENDDO
```

With **nnode = j + i*n + 1** the number of the node is calculated. Next, the principal stresses at the middle of the shell thickness are requested, the desired value for $\left(\frac{n_{xx}+n_{yy}}{2}\right)$ is calculated and eventually stored by using the code **DNSOL,nnode,U,X,nxxyy**. This code replaces at each node the value of a random chosen parameter (in this case u_x) by the value for $\left(\frac{n_{xx}+n_{yy}}{2}\right)$. By showing the contour-plot of u_x, actually a contour-plot of $\left(\frac{n_{xx}+n_{yy}}{2}\right)$ is made.

4.8 Contour plot of the mean curvature

The curvature in each direction can be calculated based upon the location of three points. One point is the point at which the curvature is calculated, the other two points are adjacent points. When to coordinates of three points are known, it is possible to fit a circle through these three points. Then, the curvature can be calculated very simple as the curvature is equal to the reciprocal of the circles radius R.

In order to calculate the circles radius based on the coordinates of three points, the following formulas can be used:

$$a = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$
$$b = \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2}$$
$$c = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}$$

$$q = \frac{a^2 + b^2 - c^2}{2ab}$$

$$R = \frac{c}{2 \cdot \sqrt{1 - q^2}}$$

The parameters a, b and c are the distances between the considered points, see the sketch of figure 4.8. q Represents a dimensionless parameter and R is the radius of the fitted circle. Based upon this radius, the curvature at point 2 can be calculated, using:

$$k = \frac{1}{R}$$

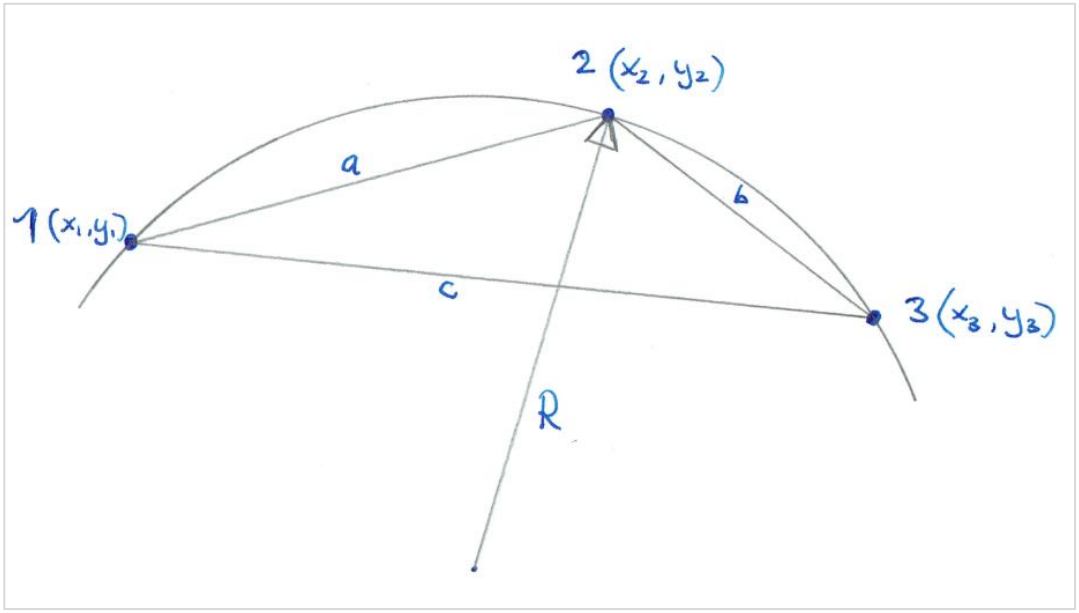


Figure 4.8 – calculating the radius at point 2, based upon two adjacent points (1 and 3)

The calculation of kyy and kxx was done in one loop. In order to do this, in total 5 nodes has to be considered during each loop. These nodes were numbered according to the sketch of figure 4.9. Node 2 is defined as the node at which the curvature was calculated. In order to calculate kyy, node 1, 2 and 3 were needed. For the calculation of kxx, node 2, 4 and 5 where needed. The numbers 1 till 5 doesn't refer to the number of the node, they only refer to the numbering for the curvature calculation.

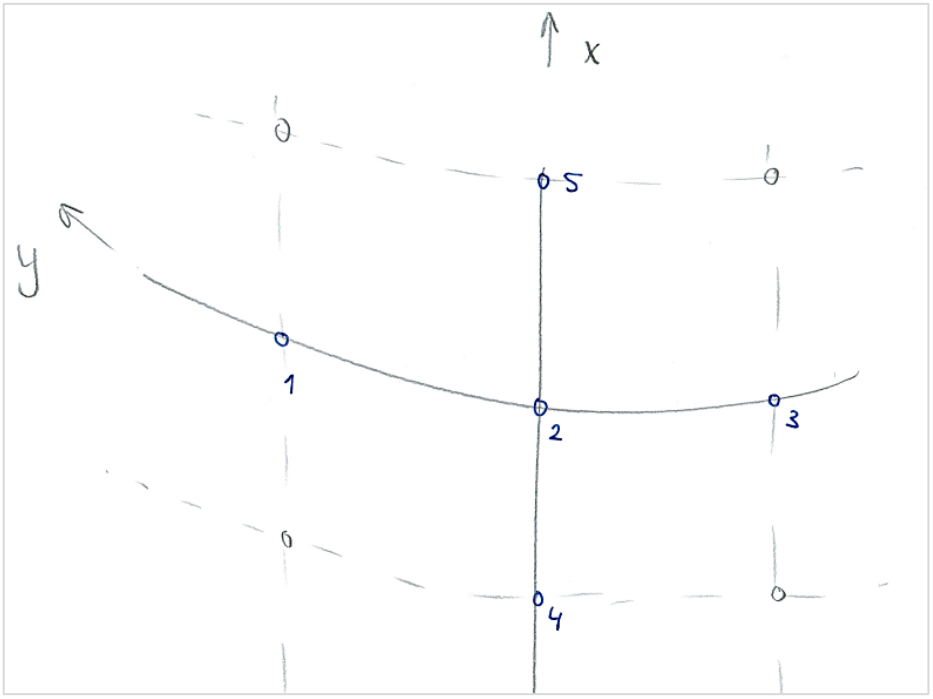


Figure 4.9 – nodes used to calculate the curvature at node 2

For each loop, the node number of node 2 is defined by: $nnode = j + i \cdot n + 1$. The node number of the other 4 adjacent nodes can be derived from the node number of node 2. In the APDL script, parameter **nnodeII** is used to determine the node number of node 2. In table 4.1, the expression for the adjacent node numbers is shown.

Table 4.1 – node number of each node in the curvature calculation, relative to the node number of node 2 (**nnodeII**)

node	node number (**nnode**)
1	**nnodeII-1**
2	**nnodeII**
3	**nnodeII+1**
4	**nnodeII-n**
5	**nnodeII+n**

The curvature kyy can be calculated directly when the x- and y-coordinates of three nodes (1, 2, and 3) are known. In order to calculate the curvature kxx, it is needed to calculate an extra parameter at each node. With this parameter, the 3D situation can be transformed into a 2D (line) situation. This parameter is the horizontal distance between the node and the original centre line of the cylinder and was determined based upon *Pythagoras*. See figure 4.10.

The parameter r at node i can be calculated as follows:
$r = \sqrt{x^2 + y^2}$

Next, the curvature has been calculated by using the coordinates r and z.

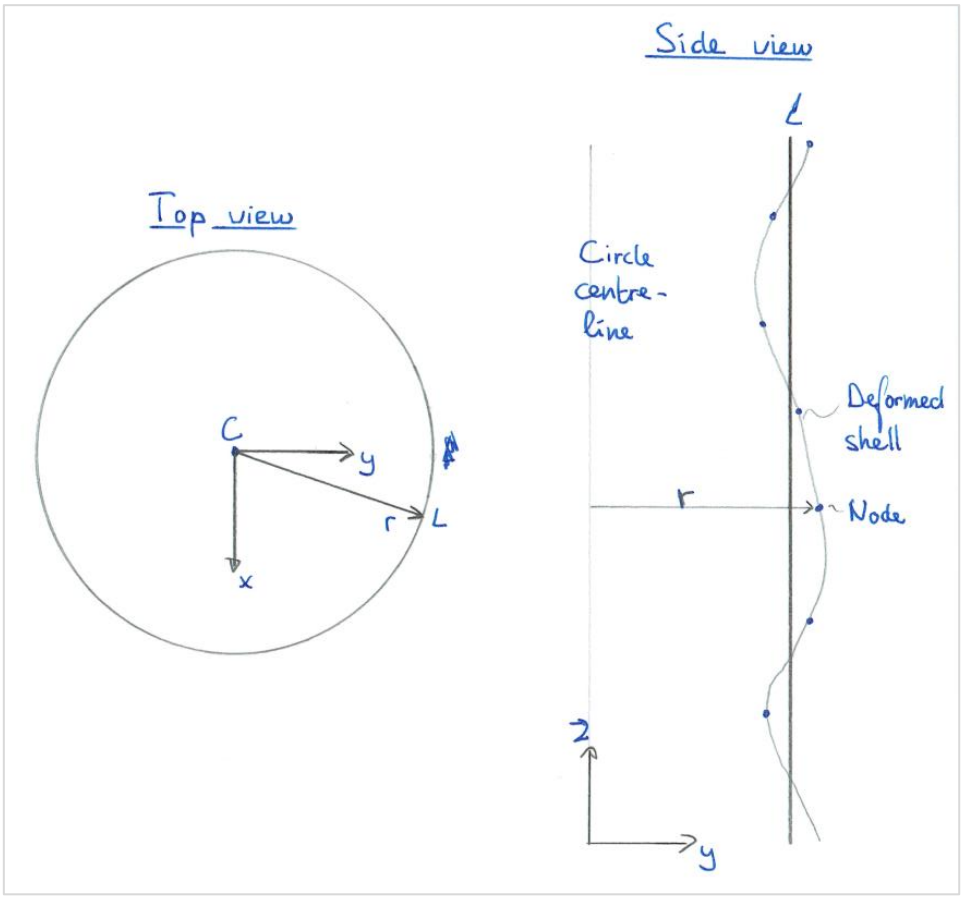


Figure 4.10 – definition of parameter r

As it is not possible to request for the nodal coordinates after deformation directly, the nodal coordinates before deformation (but with the imperfections included) and the amount of deformation were requested during the curvature calculation. The current location of each node has then been calculated by summing these two parameters.

As the curvature is determined by using the radius of a tangent circle, the outcome is always a positive value. However, it is needed to distinguish between positive and negative curvature. Before the buckling occurs, the kyy (lateral direction) will always be negative (convex). In the direction of kxx (axial direction), the sign of the curvature can be positive or negative. By connecting node 4 and 5 (see figure 4.11) with a straight line, and determine the r-coordinate (**`rcl`**) of this line at z is equal to the z of node 2 (**`z2`**), the direction (positive or negative) of the curvature can be determined. See figure 4.11 for an illustration of the mentioned nodes and parameters. When the r-coordinate (**`rcl`**) of this connection line is bigger than the r-coordinate of node 2 (**`r2`**), the curvature is negative (convex). When **`rcl`** is smaller than **`r2`**, the curvature is positive (concave).

The parameter **`rcl`** can be calculated using the following formula:

$$rcl = \frac{r_4 - r_5}{z_4 - z_5} \cdot (z_2 - z_4) + r_4$$

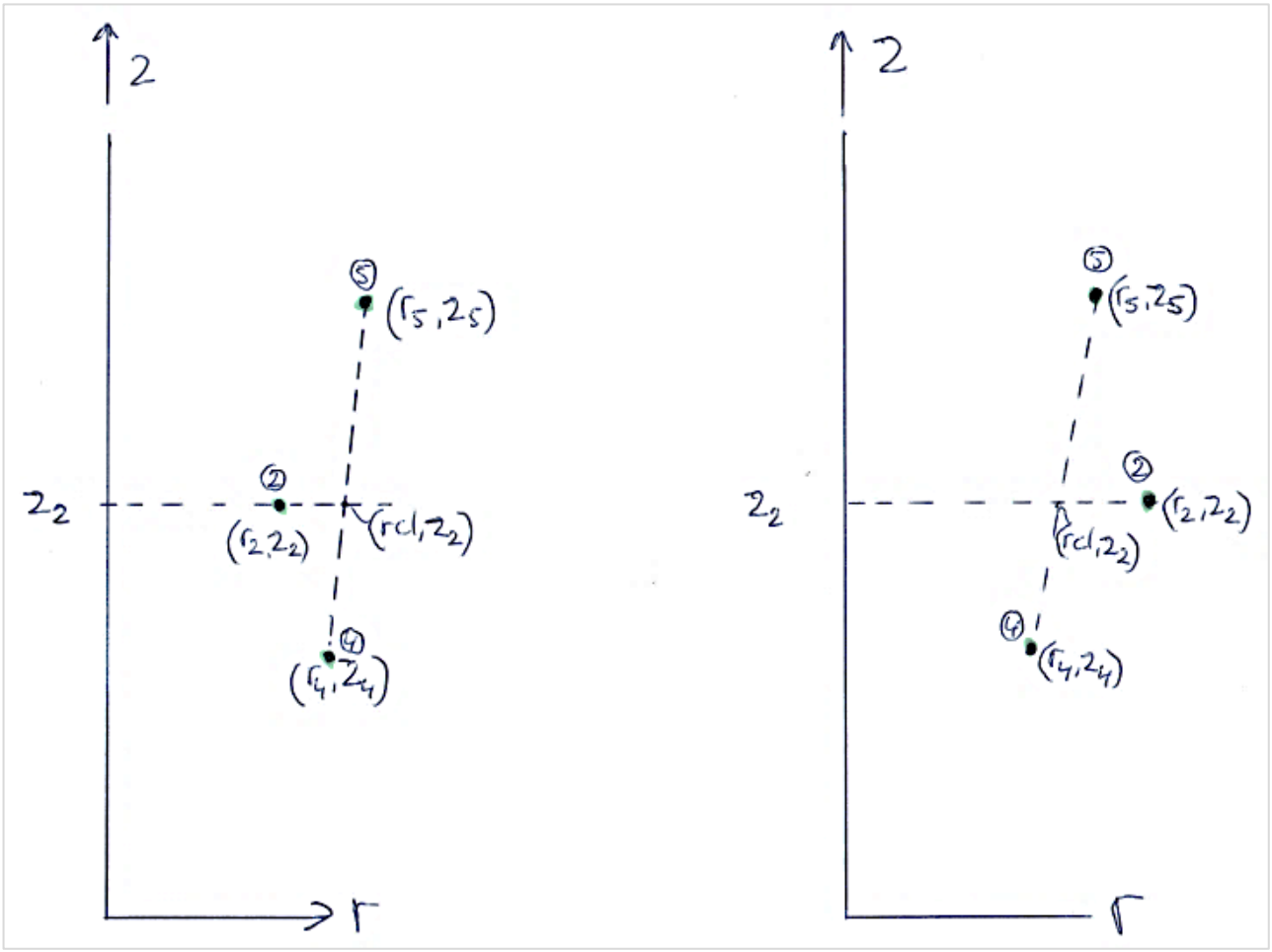


Figure 4.11 – determine the sign of the curvature. Nodes named like in figure 4.9
Left: $r_2 \leq r_{cl}$, positive curvature
Right: $r_2 > r_{cl}$, negative curvature

The APDL script shown below was used to calculate $\left(\frac{k_{xx}+k_{yy}}{2}\right)$ and to store it into the deflection in y-direction. By making a contour plot of the deflection in y-direction, in practice the $\left(\frac{k_{xx}+k_{yy}}{2}\right)$ is plotted.

Some parameters might need some explanation:
x01, ***x02***, *etc.* - *initial x-coordinate of node 1, 2, etc.*
ux1, ***ux2***, *etc.* - *deflection in x-direction of node 1, 2, etc.*

```
*DO,i,1,n-1                          ! calculating (kyy+kxx)/2
 *DO,j,1,n-2
   nnodeII = j + i*n + 1
   *GET,x01,NODE,nnodeII-1,LOC,x      ! original x-coordinates
   *GET,x02,NODE,nnodeII,LOC,x
   *GET,x03,NODE,nnodeII+1,LOC,x
   *GET,x04,NODE,nnodeII-n,LOC,x
   *GET,x05,NODE,nnodeII+n,LOC,x

   *GET,y01,NODE,nnodeII-1,LOC,y      ! original y-coordinates
   *GET,y02,NODE,nnodeII,LOC,y
```

```
*GET,y03,NODE,nnodeII+1,LOC,y
*GET,y04,NODE,nnodeII-n,LOC,y
*GET,y05,NODE,nnodeII+n,LOC,y

*GET,z02,NODE,nnodeII,LOC,z          ! original z-coordinates
*GET,z04,NODE,nnodeII-n,LOC,z
*GET,z05,NODE,nnodeII+n,LOC,z

*GET,ux1,NODE,nnodeII-1,U,x          ! displacements in x-direction
*GET,ux2,NODE,nnodeII,U,x
*GET,ux3,NODE,nnodeII+1,U,x
*GET,ux4,NODE,nnodeII-n,U,x
*GET,ux5,NODE,nnodeII+n,U,x

*GET,uy1,NODE,nnodeII-1,U,y         ! displacements in y-direction
*GET,uy2,NODE,nnodeII,U,y
*GET,uy3,NODE,nnodeII+1,U,y
*GET,uy4,NODE,nnodeII-n,U,y
*GET,uy5,NODE,nnodeII+n,U,y

*GET,uz2,NODE,nnodeII,U,z          ! displacements in z-direction
*GET,uz4,NODE,nnodeII-n,U,z
*GET,uz5,NODE,nnodeII+n,U,z

x1 = x01 + ux1                     ! new coordinates
x2 = x02 + ux2
x3 = x03 + ux3
y1 = y01 + uy1
y2 = y02 + uy2
y3 = y03 + uy3
z2 = z02 + uz2
z4 = z04 + uz4
z5 = z05 + uz5
r2 = SQRT((x02+ux2)**2+(y02+uy2)**2)
r4 = SQRT((x04+ux4)**2+(y04+uy4)**2)
r5 = SQRT((x05+ux5)**2+(y05+uy5)**2)

ay = SQRT((x2-x1)**2+(y2-y1)**2)  ! calculating kyy
by = SQRT((x3-x2)**2+(y3-y2)**2)
cy = SQRT((x3-x1)**2+(y3-y1)**2)
qy = (ay**2+by**2-cy**2)/(2*ay*by)
Ryy = cy / (2*SQRT(1-qy**2))
kyy = 1/Ryy

ax = SQRT((r2-r4)**2+(z2-z4)**2)  ! calculating kxx
bx = SQRT((r5-r2)**2+(z5-z2)**2)
cx = SQRT((r5-r4)**2+(z5-z4)**2)
qx = (ax**2+bx**2-cx**2)/(2*ax*bx)
Rxx = cx / (2*SQRT(1-qx**2))
```

```
   rcl = ((r4-r5)/(z4-z5))*(z2-z4)+r4  ! sign of kxx
   *IF,r2,LE,rcl,THEN
    kxx = 1/Rxx
   *ELSE
    kxx = -1/Rxx
   *ENDIF

   kxxyy = (kyy + kxx) / 2     ! calculating (kyy + kxx) / 2 and store
   DNSOL,nnodeII,U,Y,kxxyy
 *ENDDO
*ENDDO
```

By running this (general) code, the curvature cannot be calculated at four different locations. This is at the top (1) and at the bottom edge (2) and at the begin (3) and end of each circumference (4). See the non-red areas in figure 4.12.

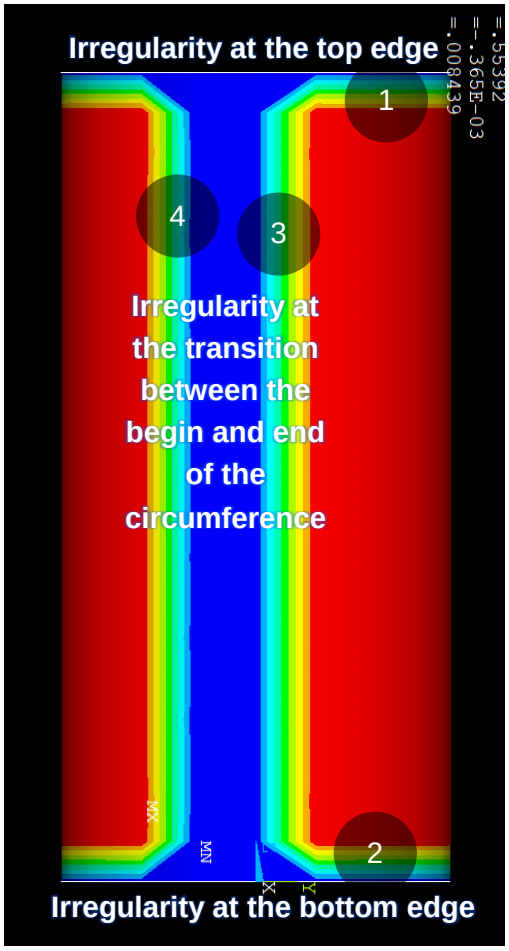


Figure 4.12 – irregularities in the curvature plot after running the general script to create the curvature plot (this contour plot is only to show the irregularities).

At the top and bottom edge of the shell (location 1 and 2 of figure 4.12), it is not possible to calculate the kxx as there is only one adjacent node in the local x-direction. This means that it is not possible to determine the radius of a circle through three points in the axial (local x-direction). To solve this problem, extra nodes were added at the top and the bottom of the cylinder. These nodes have the same r-coordinate (distance from the original circle centre) as node 2 (figure 4.9). However, the z-coordinates of these points have a slightly deviation to

the z-coordinate of the node at which the kxx must be calculated (node 2, figure 4.9). This deviation is defined with parameter **dz**. Parameter **dz** was set to 1/1000.

As the node at which the curvature has been calculated must lie in the middle (like point 2 of figure 4.9), the “dummy” nodes at the top edge lie **dz** above the top edge and the “dummy” nodes at the bottom edge lie **dz** under the bottom edge. In figure 4.13 the added “dummy” nodes for the top edge are shown (original nodes in blue, dummy nodes in grey). The dummy nodes were not literally added to the model, their (virtual) coordinates were only used in the calculations of the curvature.

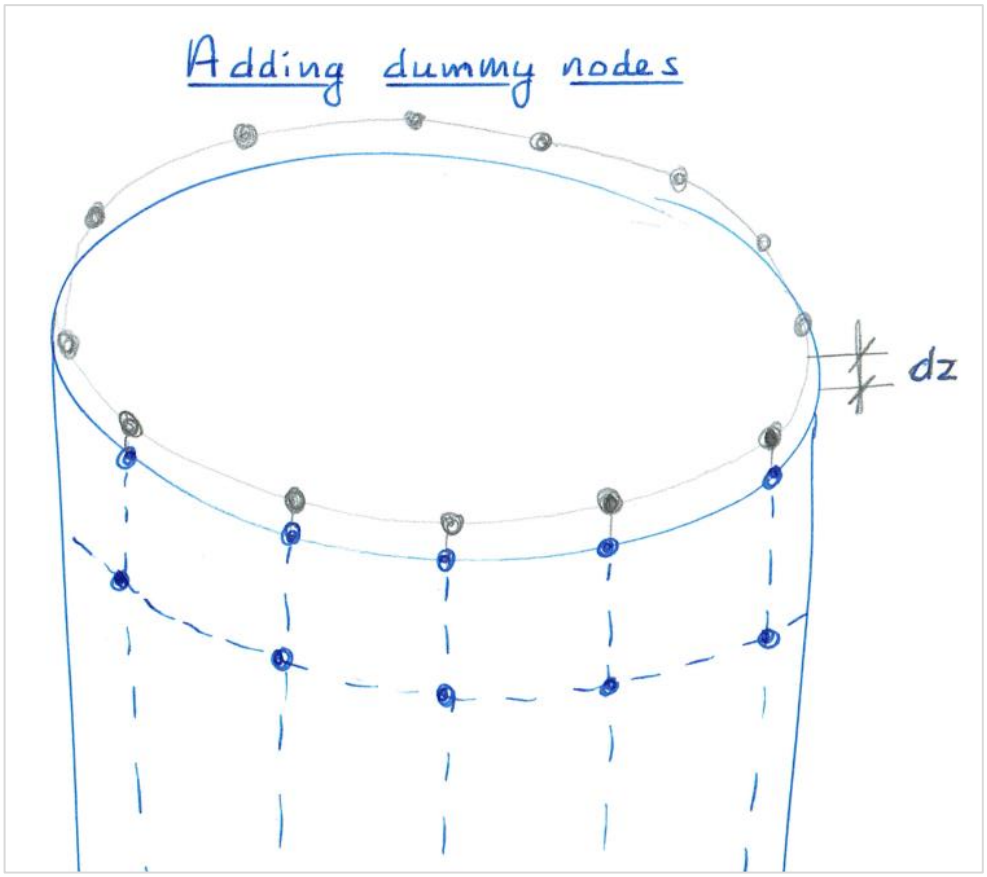


Figure 4.13 – Adding dummy nodes at the top and bottom edge

The APDL script to calculate the curvature at the top and bottom edge is very similar to the general script to calculate the curvature. For this reason, the script to calculate the curvature at the top and bottom edge is only shown in appendix 1.

Also at the begin and the end of each circumference, the curvature cannot be calculated using the general APDL script. This is due to the fact that the node numbers of the adjacent nodes cannot be defined within the general script. The lowest node number of each circumference lies next to the highest node number of the same circumference. This irregularity can be solved in a similar way as for the irregularities at the top and the bottom edge. Also in this case, the APDL code can be consulted into appendix 1.

After the irregularities were solved, there were still 4 points at which the curvature has not been calculated. These 4 points lie in the intersection of two “irregularity zones”, see the red crosses of figure 4.14.

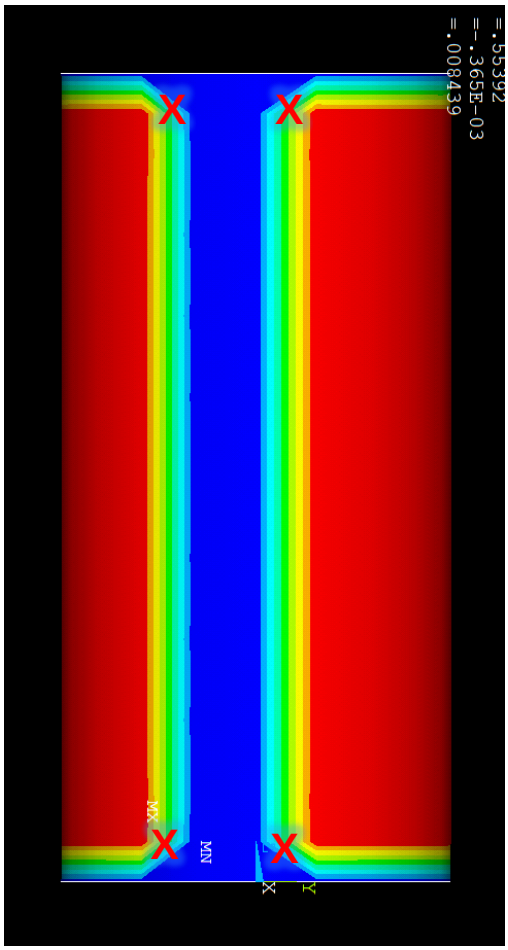


Figure 4.14 – 4 points (**X**) at the intersection of two “irregularity zones”.

As each node needs a different APDL script to resolve this problem and it are only four points (of n*(n+1) points in total), it was decided to assume that the curvature at these points is the same as their nearest neighbour in local y-direction. Assigning the curvature of the nearest neighbour to each of these 4 points can be done, by using the following APDL script:

```
*GET,kxxyybb,NODE,2,U,Y                    ! 4 points at the intersection
DNSOL,1,U,Y,kxxyybb                         of irregularity zones
*GET,kxxyybe,NODE,n-1,U,Y
DNSOL,n,U,Y,kxxyybe
*GET,kxxyytb,NODE,n*n+2,U,Y
DNSOL,n*n+1,U,Y,kxxyytb
*GET,kxxyyte,NODE,n*n+n-1,U,Y
DNSOL,n*n+n,U,Y,kxxyytb
```

4.9 Conclusions about Ansys mechanical APDL

Ansys mechanical APDL is much different to the earlier used program Ansys workbench. The benefits of mechanical APDL are the possibility to perform all the needed analysis in one sequence by running the complete code and the possibility the adjust some input parameters (§4.1) very easily. Furthermore, the user can have a big influence on how the mesh is created, which can have a lot of benefits while calculating the mean membrane force and the mean curvature.

The most prominent drawback of mechanical APDL is the need to learn a programming script which is specifically made for Ansys. This means that despite of being familiar with programming, it is needed to search for commands who are only used for Ansys. A second drawback of using mechanical APDL, is the fact that it is not easy the see what the program has exactly done after running the script, especially in the case of a sequence of different analyses.

By running the code of appendix 1, it is not possible to make an animation directly. For now, the only possibility to make a movie is to make screen-shots of each sub-step and to edit these screen-shots in a movie maker program. The contour plots of each sub-step can be obtained by inserting the script **`STEP,1,x`** into the APDL code, between the script for the non-linear analysis and the script to calculate the mean membrane force. In this script, **`x`** is the number of the sub-step.

5 Results and conclusions on the contour plots

In this chapter, the obtained contour plots of the mean membrane force and the mean curvature are shown and being discussed. After running the code of appendix 1, only the contour plots of the last sub-step were shown. These contour plots of the last sub-step are displayed in this chapter, as well as an enlarged plot of the deformed shape of the cylinder.

First, the cylinder was modelled with a thickness of 0.12 mm and fixed supports at the top and the bottom. As it is really easy to adjust the APDL code a bit and re-run it, it was decided to investigate the effect of the shell thickness and the boundary conditions on the outcome of the buckled shape, mean membrane force and the mean curvature. §5.1 shows the resulting contour plots of a/t = 500 with fixed edges. In §5.2 the effect of the ratio a/t is shown. The effect of the boundary conditions on the outcome is shown and being discussed in §5.3. In §5.4, some conclusions from the different contour-plots are drawn. Finally, the use of mechanical APDL is being discussed in §5.5.

5.1 Fixed edges, a/t = 500

The first plot is made based upon a shell with membrane thickness t = 0.12 mm and fixed at the bottom and top edges. This means for the bottom edge that the rotations and displacements in all directions are 0. For the top edge holds almost the same, however displacements in the z-direction (axial direction) are allowed. The maximum amplitude of the imperfections e = 0.06 mm, which means that e/t = 0.5. In figure 5.1, the deformed shape (enlarged) is shown.

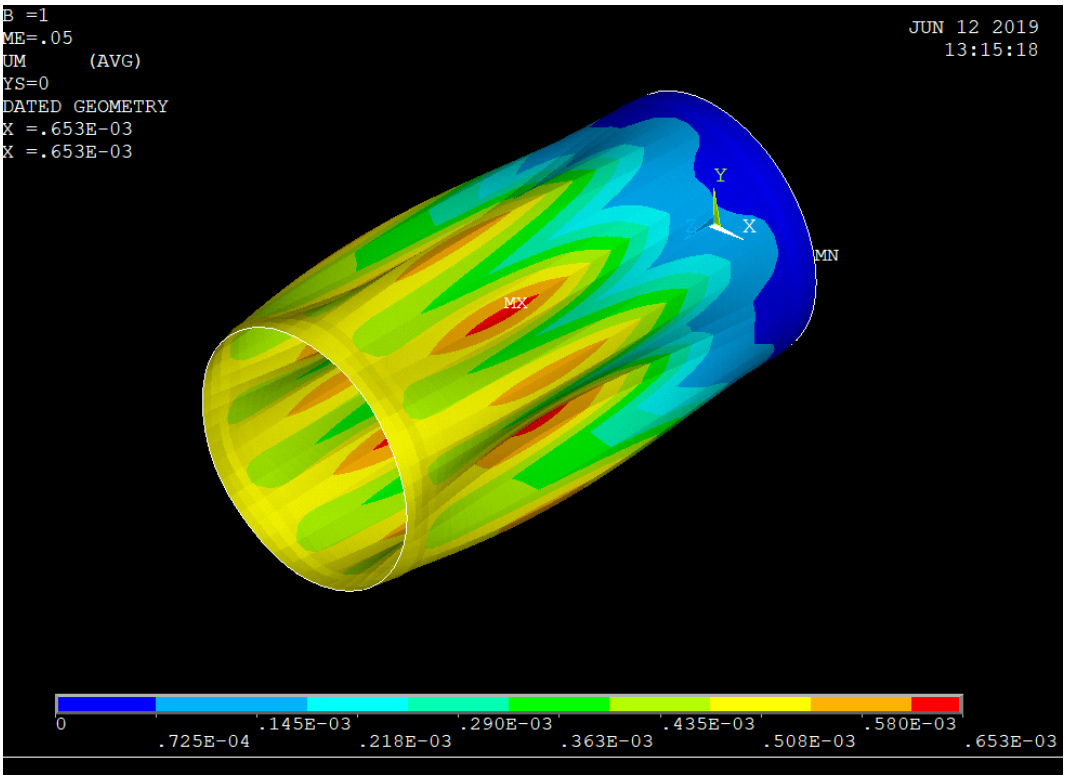


Figure 5.1.a – deformed shape: fixed edges, t = 0.12 mm, a = 60 mm, a/t = 500, e = 0.06 mm, e/t = 0.5

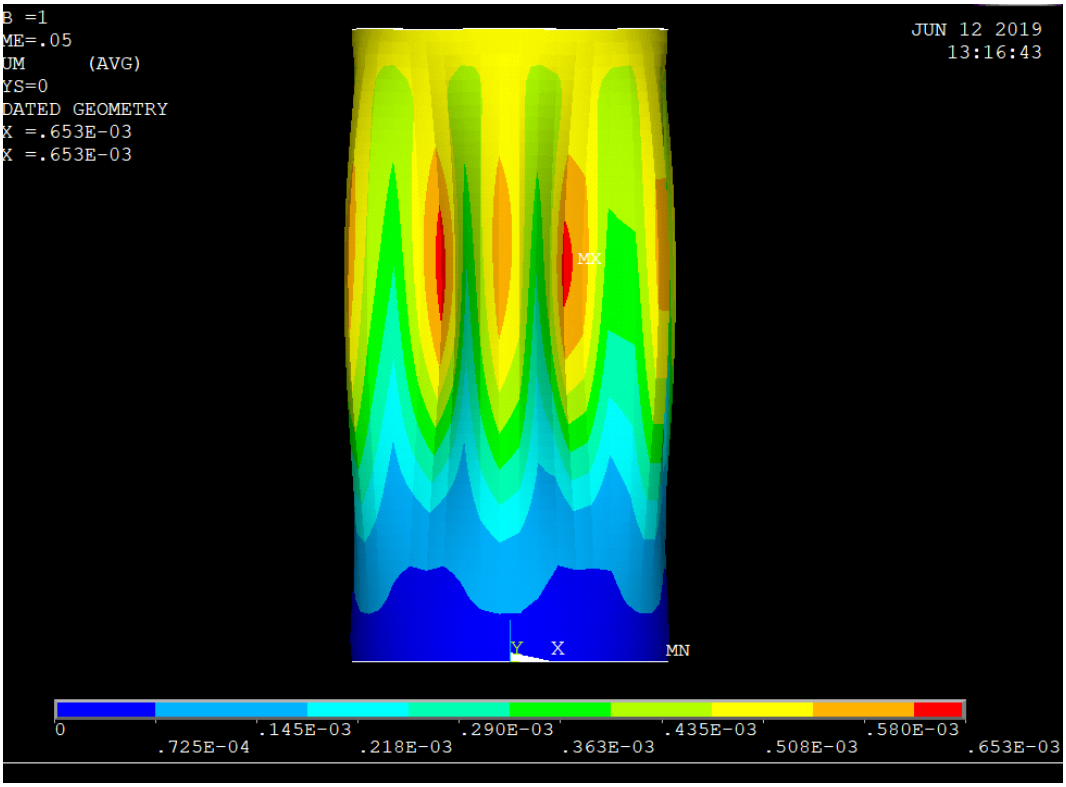


Figure 5.1.b – deformed shape: fixed edges, t = 0.12 mm, a = 60 mm, a/t = 500, e = 0.06 mm, e/t = 0.5

From figure 5.1 a and b, it can be seen that there are 10 waves around the circumference of the cylinder (5 inward and 5 outward). The circumference has a total length of $2\pi \cdot a = 2\pi \cdot$

60 = 377 mm. So, the buckling length in lateral direction becomes approximately 377 / 10 = 38 mm. This is the buckling length in lateral direction. The buckling length in axial direction can be estimated based upon the total length of the cylinder. As the total length is 250 mm, the buckling length in axial direction is assumed to be approximately 200 mm.

Next, the plot of the mean membrane force and the mean curvature were extracted from Ansys. With the used APDL code, it was only possible to show these two plots for the last sub-step. It was not possible to produce an animation in which the plots for all the sub-steps are shown automatically. By adding the APDL-code **`STEP,1,x`** above the calculation of the mean membrane force and mean curvature it is possible to plot the desired contour plots for sub-step **x** of the non-linear calculation.

The next figures, show the mean membrane force and the mean curvature for the last load-step of the non-linear calculation of the same model, as described above. In this case, the maximum number of iterations (which was equal to 100), was met after performing 8 sub-steps. This means that the non-linear calculation was stopped after 7 sub-steps. Figure 5.2 shows the mean membrane force after 7 sub-steps and figure 5.3 shows the mean curvature after 7 sub-steps.

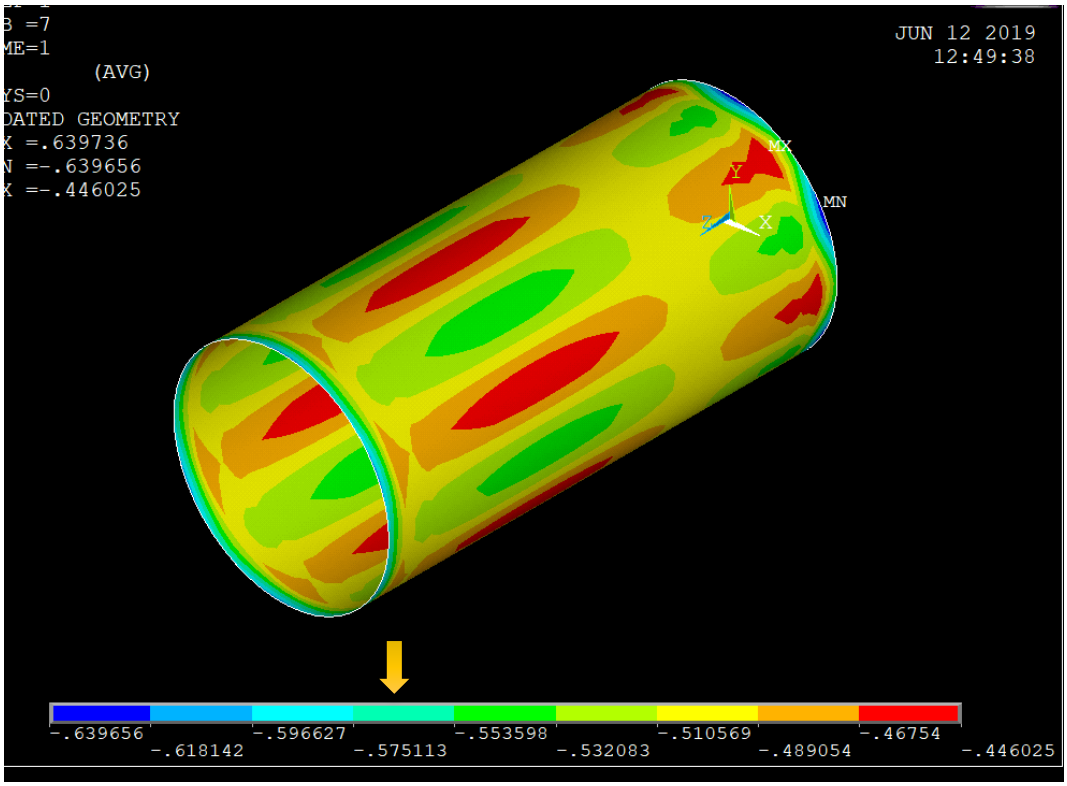


Figure 5.2 – (nxx+nyy)/2: fixed edges, t = 0.12 mm, a/t = 500

The top edge (the edge on the left) shows the same colour around the whole circumference. This colour represents the compressive load applied at the moment of the last sub-step. By assuming that the non-linear calculation started to divergence at the moment of buckling, the shown – last – sub-step of loading is just before buckling. This would mean that the buckling load of the cylinder is equal to the membrane force at the top edge, which is about -0.58 N/mm (see orange arrow above the colour-scale).

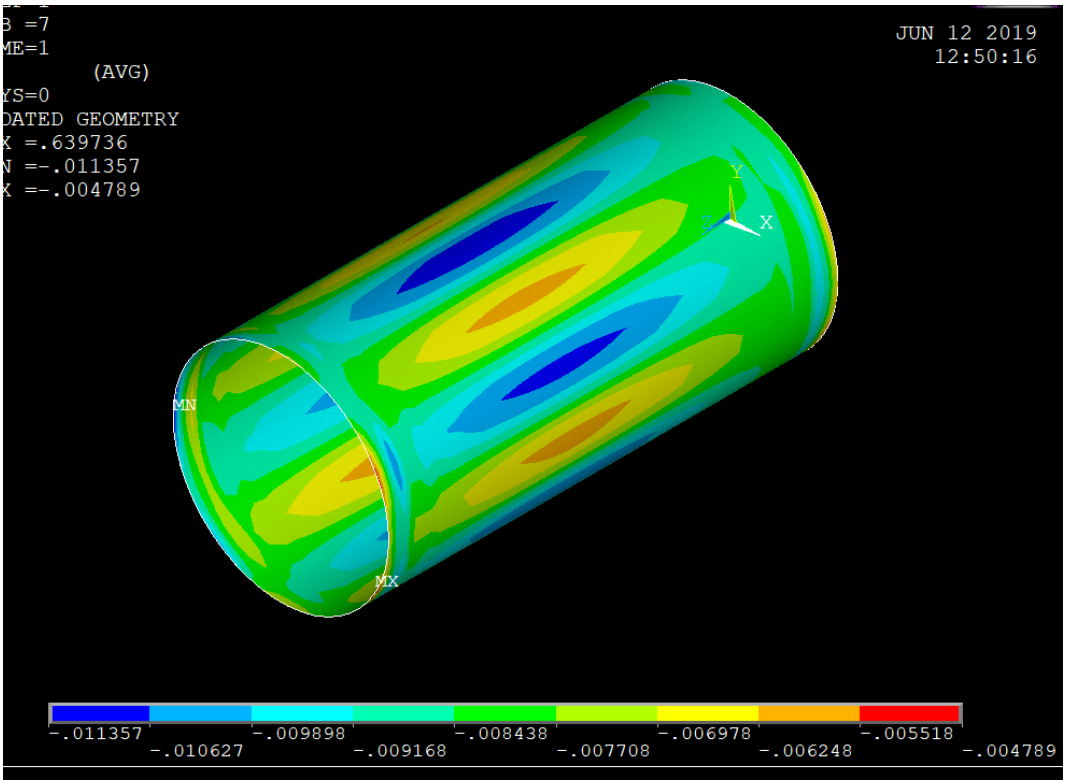


Figure 5.3 – (kxx+kyy)/2: fixed edges, t = 0.12 mm, a/t = 500

In order to get a better understanding of the plot of the mean curvature, the curvature was also plotted in each direction alone. These plots can be seen in figure 5.4 (kxx) en 5.5 (kyy).

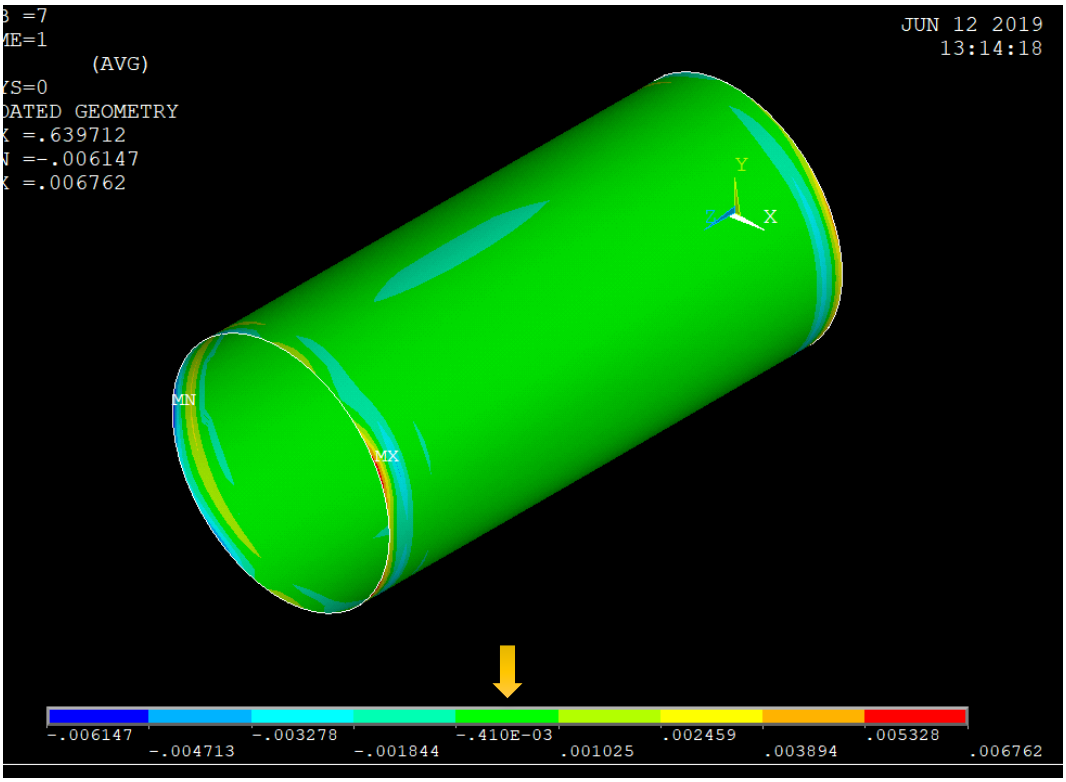


Figure 5.4 – kxx alone: fixed edges, t=0.12 mm, a = 60 mm, a/t = 500

In the plot of figure 5.4, it can be seen that the biggest part (green zone) of the curvature kxx (axial direction) is between -0.0004 /mm and 0.0010 /mm. According to the legend, the green zone runs from a negative curvature to a positive curvature. This means that it cannot be concluded at which spots the curvature is positive or negative from this picture. Somewhere, within the green zone of the colour-scale, the sign of the curvature flips. For instance, this *flipping of sign* could be at the location of the arrow.

By looking at the corresponding deformed shape of the cylinder, the obtained curvature plot kxx seems to be correct. There are no spots in the curvature plot, that cannot be correct. The curvature in axial direction is very small and doesn't change much in axial direction, this complies with the observation in figure 5.1 a and b.

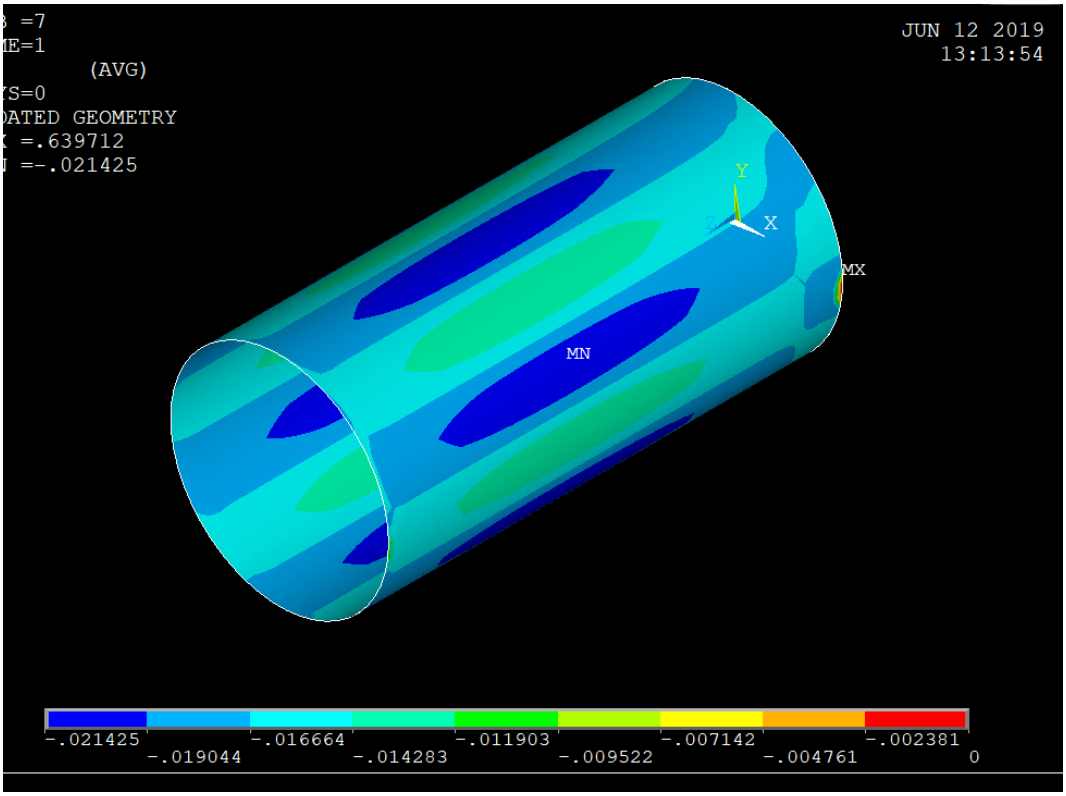


Figure 5.5 – kyy alone: fixed edges, t=0.12 mm, a = 60 mm, a/t = 500

The curvature in lateral direction (kyy, figure 5.5) runs from -0.021 /mm until about -0.012 /mm (the zones up and until red are ignored as these zones are assumed to be edge disturbances). The original curvature of the shell in lateral direction (before adding the imperfections), is equal to one over the radius of the cylinder. So, the initial kyy is equal to -1/a = -1/60 = -0.0167 /mm. This curvature is negative as the shell is convex.

At the moment the non-linear analysis did divergence, the curvature in lateral direction lies exactly around the initial curvature. This can be seen by taking the mean of the observed interval. The mean value of the interval is equal to (-0.021 – 0.012) / 2 = -0.167 /mm. This means that the initial kyy is accurately in the middle of the range of the curvature just before divergence. From this, one can conclude that during the formation of buckles, at some places the curvature becomes bigger and at some places the curvature becomes smaller. Both with the same amount.

The value of the curvature can be checked with a simple calculation using the formula of Sagitta. This formula reads:

$$a = \frac{1}{2}s + \frac{1}{8}\frac{l^2}{s}$$

The parameters of this formula can be found in figure 5.6.

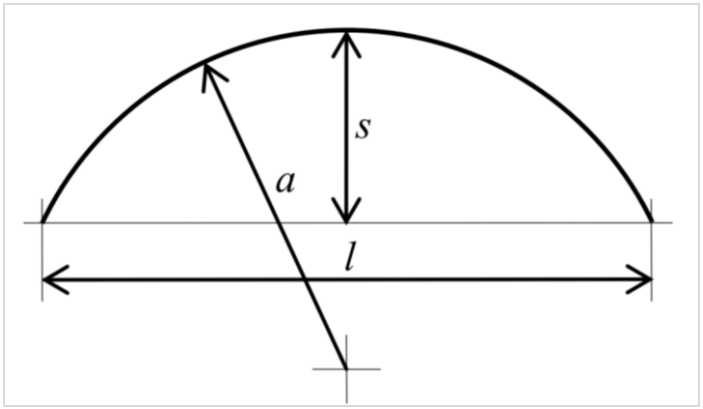


Figure 5.6 – Sagitta (Hoogenboom)

Parameter a is equal to the radius R and $k = \frac{1}{R}$. By using this, the formula can be re-written into (for $s \ll l$):

$$k = \frac{8s}{l^2}$$

For the buckled cylinder, the s can be divided into 3 different contributions. The curvature due to the initial shape (s_1), the added imperfection (s_2) and the deformation due to buckling (s_3). These contributions are shown in figure 5.7.

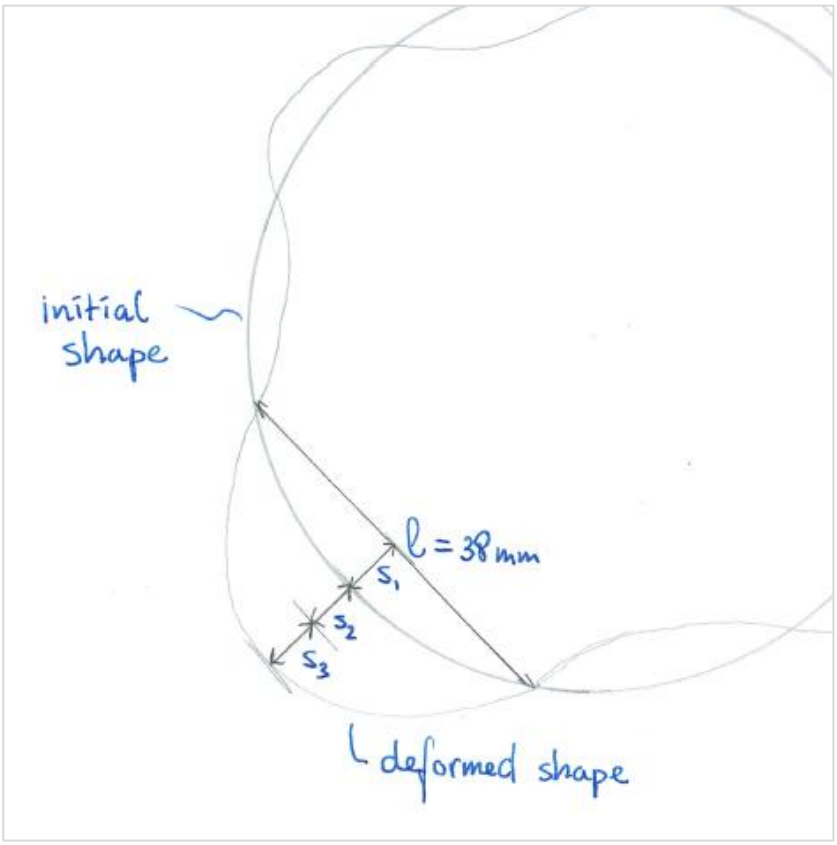


Figure 5.7 – Contributions to the total s.

The total s can be calculated using: $s = s_1 + s_2 + s_3 = \frac{kl^2}{8} + \frac{1}{2}t + u$.

Where $\frac{kl^2}{8}$ can be derived from the Sagitta formula, $\frac{1}{2}t$ is the added shape imperfection and u is the displacement due to buckling (see figure 5.1 a and b).

Filling in the formula for s: $\frac{1}{60}\frac{38^2}{8} + 0.06 + 0.000653 = 3.0690$ mm. Based upon this value for s, the curvature in lateral direction becomes: $k = \frac{8s}{l^2} = \frac{8 \cdot 3.0690}{38^2} = 0.0170$ /mm (which is actually -0.0170 /mm due to the sign convention). Compared to the original curvature (-0.0167 /mm), the curvature just before divergence has only become 2% larger, based upon Sagitta.

The plot of kyy does comply with the deformed shape of figure 5.1 a and b. The interval of the obtained curvature is exactly around the initial curvature. However, the curvature based upon the deformation using Sagitta, is much smaller than the curvature of figure 5.5. The curvature shown in figure 5.5 is about 25% bigger than the curvature expected from the deformations. So, maybe the curvature plot isn't correct.

Besides, it can be concluded that by comparing figure 5.4 and 5.5, the curvature in y-direction (lateral direction) is much bigger than the curvature in x-direction (axial direction). The absolute value of the curvature kxx is < 5% of the absolute value of the curvature kyy. This corresponds to the deformed shape of figure 5.1 a and b. In these figures it can be seen that curvature in axial direction (kxx) is much smaller compared to the curvature in lateral direction (kyy).

5.2 Comparison between different values for a/t

After the analysis was done for one value of the slenderness (a/t = 500). The analysis was also performed for other values of a/t. This was done for a/t = 250 and a/t = 100. In order to get these values for a/t, the value for the membrane thickness t was adjusted to 0.24 mm respectively 0.60 mm. All the other parameters and boundary conditions were left the same as for the analysis discussed in §5.1. This means that the comparison also can be made with a thickness of t = 0.12 mm involved (these results are shown in §5.1).

In table A2.1 of appendix 2, the results for different values of the membrane thickness t just before divergence of the non-linear analysis are displayed next to each other. The displayed plots are the ones for the mean membrane force and the mean curvature.

By comparing the results of the three different membrane thicknesses, it can be concluded that the distribution of the mean membrane force and the mean curvature have both the same shape for all three thicknesses. In axial direction the distribution is almost the same for all the different thicknesses. However, in lateral direction both the mean membrane force and the mean curvature have a wider distribution when the thickness becomes bigger.

It also can be seen that a red zone for the mean membrane force, always corresponds with a (dark) blue zone in the mean curvature plot, even when this pattern is a little bit rotated around the circumference (like for t = 0.60 mm).

The fact that the pattern becomes wider in the lateral direction, means that the shape of the buckled shell changes with different shell thicknesses t.

5.3 Comparison between different boundary conditions

Besides comparing the slenderness of the cylinder, also the effect of the chosen boundary conditions has been investigated. In this paragraph, the analysis was done by allowing the top and bottom edge to rotate. Like in §5.2, the analysis in which rotations at the top and bottom edge are allowed was done for three different thicknesses.

The results of this analysis can be seen in table A2.2 of appendix 2. The results in table A2.2 doesn't differ a lot from the results of table A2.1 (also in appendix 2). Also the differences between the different membrane thicknesses t are the same as for the fixed rotations.

As the colour-scale differs for every plot, the values for the different boundary conditions might differ. By comparing the plots for one thickness (t = 0.12 mm), it can be seen that also the numerical values for the mean membrane force and the mean curvature doesn't differ a much. In table A2.3 and A2.4 of appendix 2, it can be seen that the absolute values in both plots are slightly lower for fixed rotation boundary conditions.

Another interesting boundary condition is when the top edge is not only allowed to rotate, but also is allowed to displace in all directions. Displacements in z-direction were already allowed. However, by allowing displacements in the x,y-plane, the top edge of the cylinder isn't forced to remain a circle. So, it can be assumed that the buckled shape is much different for these boundary conditions.

In figure 5.8, the deformed shape of a cylinder for which displacements in all directions are allowed at the top edge can be seen. The shape of the cylinder just before meeting the divergence indeed differs a lot from the ones obtained earlier.

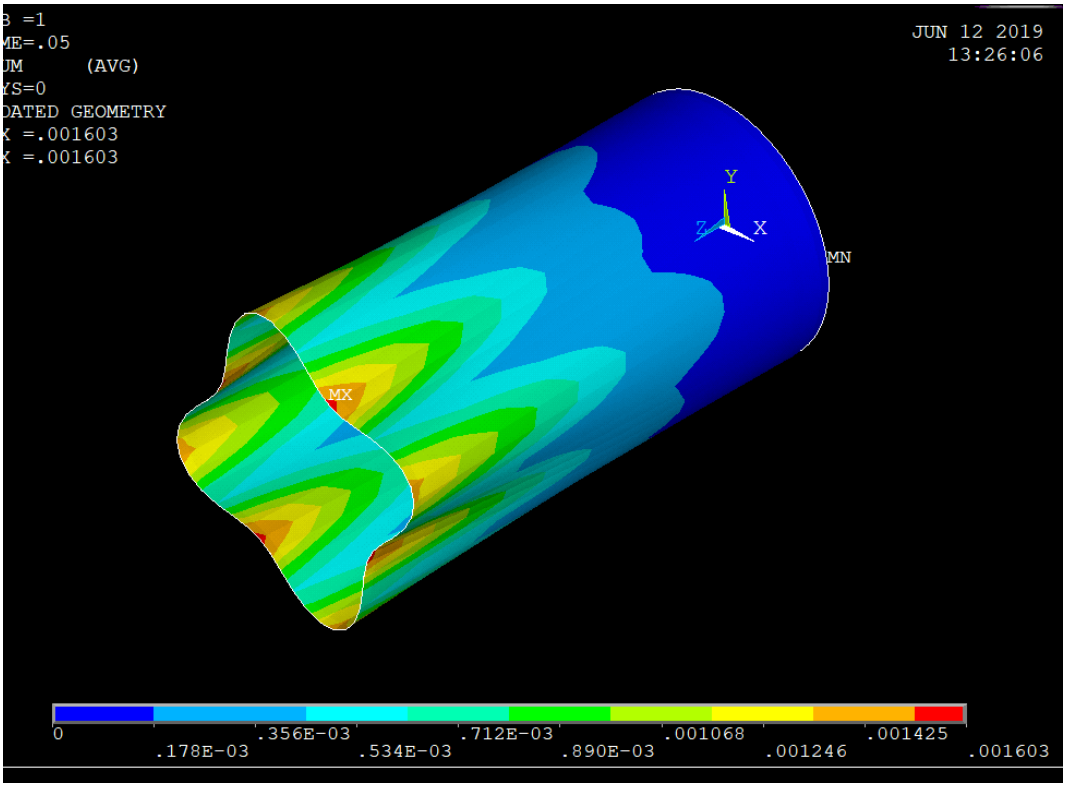


Figure 5.8 – deformed shape (exaggerated): t = 0.12 mm, a/t = 500

It can be observed that the buckles has been shifted towards the top edge of the cylinder. Also the number of waves in the circumference (lateral direction) changed from 10 (see §5.1) to 8 (4 inward and 4 outward).

In figure 5.9 the mean membrane force of the cylinder for which the top edge may displace in all directions is plotted.

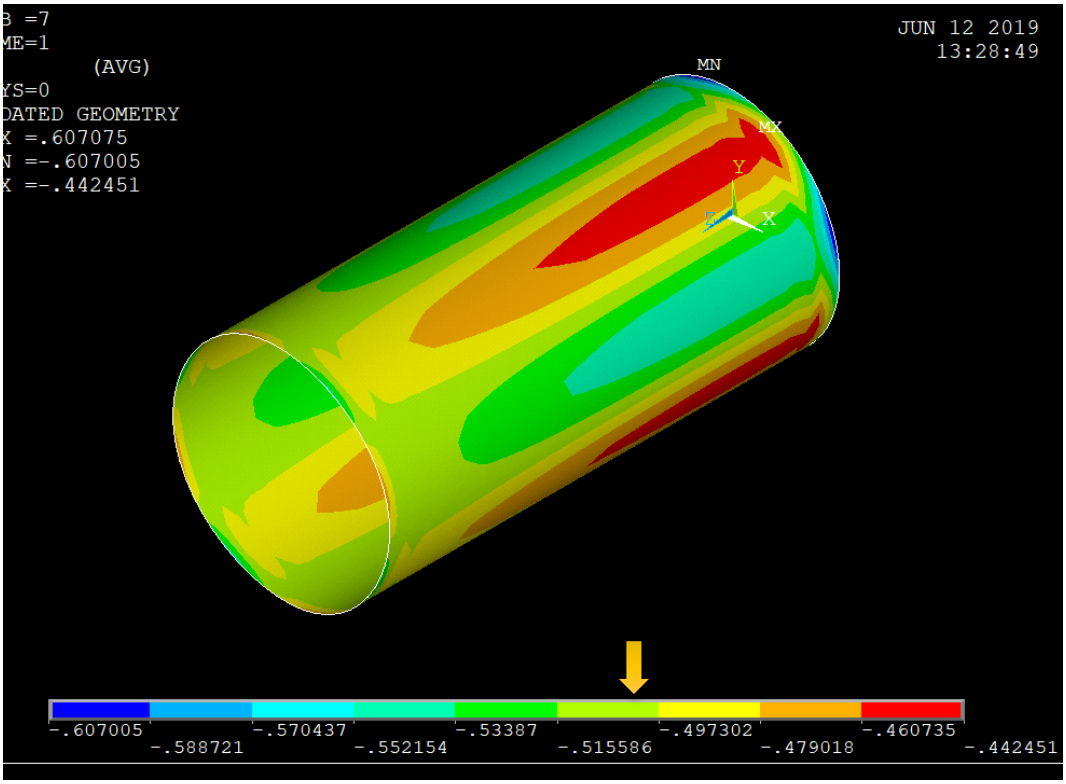


Figure 5.9 – (nxx + nyy)/2: t = 0.12 mm, a/t = 500

The plot above shows that the loading at the top edge is equal to approximately -0.5 N/mm, just before divergence (see the orange arrow along the colour-scale). At the bottom edge, the mean membrane force is approximately between -0.55 and -0.44 N/mm. This is up to 10% smaller and larger than the average. The average is more or less equal to the load along the top edge. As it is assumed that the divergence of the non-linear analysis starts at the moment of buckling, it can also be assumed that the loading at the top edge is the critical buckling load.

The mean curvature for this boundary conditions is also plotted. This plot can be seen in figure 5.10.

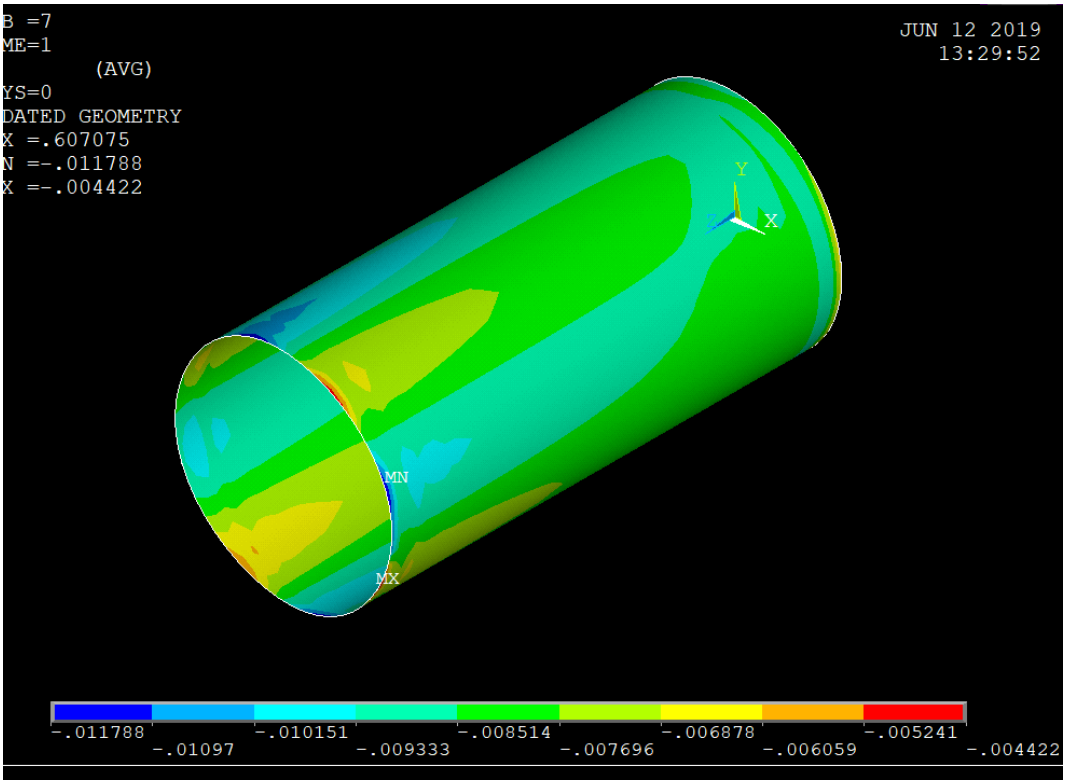


Figure 5.10 – (kxx + kyy)/2: t = 0.12 mm, a/t = 500

For most of the surface, the curvature as shown in figure 5.x runs from about -0.010 /mm until -0.007 /mm. Again, the mean curvature is divided into kxx (figure 5.11) and kyy (figure 5.12).

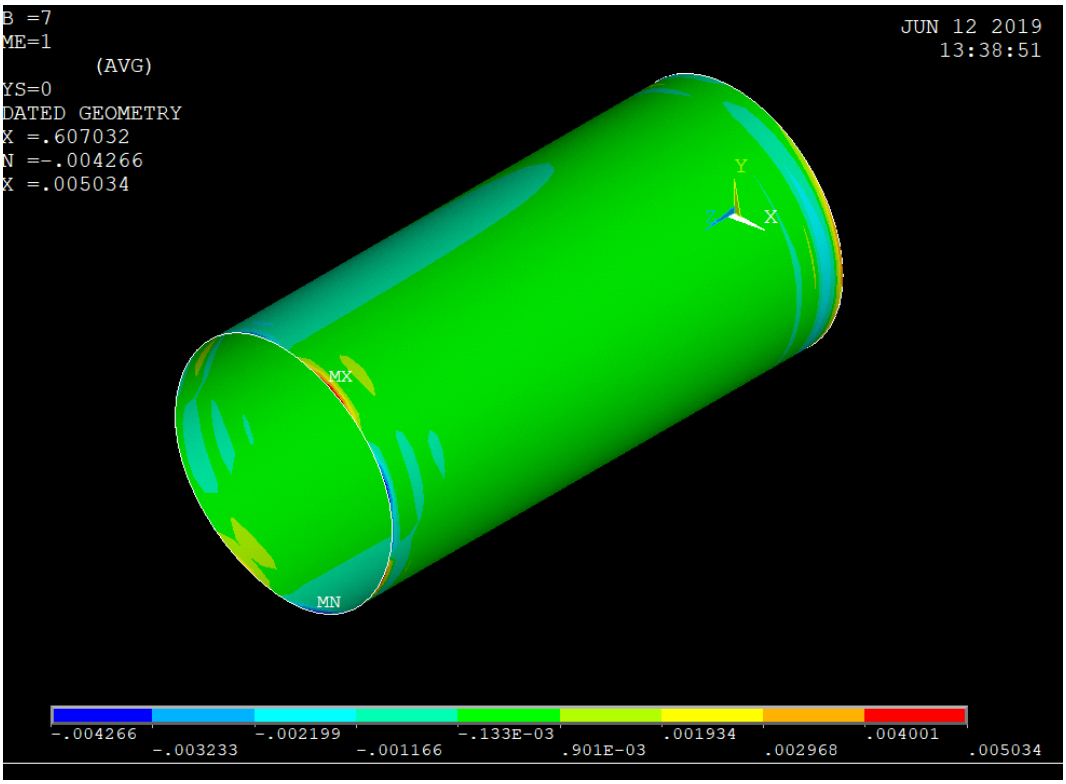


Figure 5.11 – kxx alone: t = 0.12 mm, a/t = 500

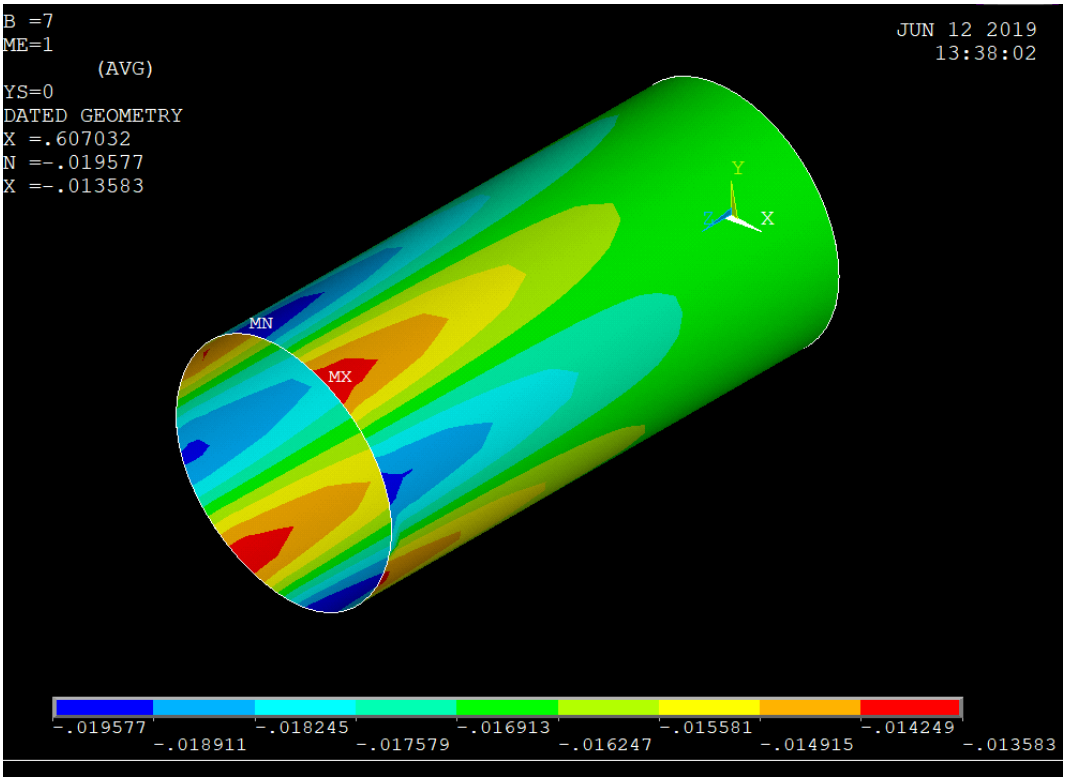


Figure 5.12 – kyy alone: t = 0.12 mm, a/t = 500

It can be seen that kxx is both positive and negative, again the “flip of sign” lies somewhere in the green zone. Also for this boundary conditions, the contribution of kxx to the average

curvature is very low compared to kyy. The kxx runs from about -0.0001 /mm until 0.0009 /mm, where for kyy holds: -0.0196 /mm < kyy < -0.0135 /mm.

The curvature kyy of the perfect shell is equal to -0.0167 /mm. Again, this is accurately in the middle of the kyy interval of figure 5.12 (kyy just before divergence). The kyy just before divergence is up to 20% larger or smaller than the kyy of the perfect, undeformed shell.

5.4 Conclusions from the contour plots

The mean membrane force plots of figure 5.2 and figure 5.9 show that the applied load just before divergence is around -0.58 N/mm and -0.5 N/mm. As it is assumed that the divergence is caused by buckling, these two values are assumed to be the critical buckling loads. However, by comparing these values to the theory and experimental observations about imperfections, one can determine whether it is possible that the divergence is caused by buckling.

The theoretical critical buckling load is determined by the formula:

$n_{cr} = -0.6 \cdot \frac{E \cdot t^2}{a}$ (Hoogenboom)

Filling in the values for the analysed cylinder, the critical buckling value becomes:

$n_{cr} = -0.6 \cdot \frac{2.1 \cdot 10^5 \cdot 0.12^2}{60} = 30$ N/mm

This value is much larger than the observed values from the analysis in Ansys. However, from experiments it has become clear that the shape imperfections dramatically lower the critical load. The factor between n_{ult} (ultimate load, obtained in experiments) and the n_{cr} (theoretical critical load), is around 0.25 for a/t < 500 (Hoogenboom). This means that the practical buckling load is approximately 4 times lower than the theoretical critical buckling load due to the shape imperfections.

Taking into account the shape imperfections, results into a critical load of -30 / 4 = -7.5 N/mm. This is still much larger than the values obtained with the analysis in Ansys. Based upon this observation, one can conclude that the divergence of the non-linear calculation is caused by another phenomena than buckling. However, it is not known which phenomena this divergence has caused. This means that the obtained contour plots are not *just before buckling*, as the non-linear calculation started to divergence too early.

Comparing the buckled shape from figure 5.1 and figure 5.8 to the theoretical expectations of the buckled shape, like in figure 2.2, it can be seen that the shape obtained with Ansys doesn't comply with the theory. So, also the shape of the buckles show no agreement between theory and observations. The buckling shape is almost independent of the a/t ratio and independent of boundary conditions with the displacements in x- and y-direction fixed for the top edge. When displacements in x- and y-direction of the top edge are allowed, the buckled shape changes.

Also the theoretical buckling length and the obtained buckling length can be compared. The theoretical buckling length can be obtained with the formula:

$l_{buc} = 1.7 \cdot \sqrt{at}$ (Hoogenboom)

For the analysed cylinder, the value for the buckling length becomes $l_{buc} = 1.7 \cdot \sqrt{60 \cdot 0.12} = 4.6$ mm in the axial direction. The buckling length observed in figure 5.1 is approximately 200 mm in the axial direction and 38 mm in the lateral direction.

The buckled shape and buckling length could differ due to the way of loading. In the performed analyses, the shell has been loaded with a force along the top circumference. Applying the load in this way, the circumference of the top edge starts to deform in such a way that it doesn't lie in one plane. This means that the deformation in axial direction is not the same for the same z-value. This can be concluded from the blue waved-pattern at the bottom edge of figure 5.1 and 5.8. Maybe the results of the analyses will comply with the theory when the loading is applied with the help of a pre-described deformation.

According to figure 5.5 and 5.12, the curvature changes significantly (20%) during loading just before divergence, but still being away from buckling. However, the curvature that is expected from the observed deformations (figure 5.1) is much larger than obtained from the analysis (5.5). Using the deformation of figure 5.1 to calculate the new Sagitta and the new curvature, the curvature is expected to change only 2%. This observation raises the question whether the curvature plot is correct.

It can be seen that in all cases, the curvature in lateral direction dominates the curvature in axial direction as the absolute value of the curvature in axial direction is only 5% of the absolute value of the curvature in lateral direction. Besides, both the magnitude of the imperfections and the mean membrane force has become (10%) larger, just before divergence.

By assuming that all the changes in curvature took place in the lateral direction (which is almost true), the formula for n_{cr} can be combined with $k_{yy} = 1/a$ and written into $n_{cr} = -0.6 \cdot E \cdot t^2 \cdot k$, where k is the mean curvature. As the curvature of the shell with imperfections becomes up to 1.20 times smaller and the normal force becomes up to 1.10 times bigger, the critical load can become 1.20 * 1.10 = 1.32 times smaller. This doesn't explain the shells sensitivity for imperfections, as experiments show critical loads of up to 4.00 times smaller due to the shape imperfections. However, if the problem with the too early divergence (too far from buckling) is solved, the changes in curvature and membrane force may became that large that it does explain experimental results.

6 Conclusions and recommendations

In this last chapter, the main conclusion of this research is drawn. Besides, also some additional conclusions about the contour plots, as well as the used methods and programs are presented. To conclude the research, several recommendations for further investigation are suggested.

6.1 Main conclusions

The main question of this research was:

Is it possible to demonstrate the formation of buckles in a short movie, with the help of contour plots of the mean membrane force and the mean curvature? And can the hypothesis of dr. ir. P. C. J. Hoogenboom be approved?

Based upon this research, it can be concluded that it is possible to make a movie in which the formation of buckles can be demonstrated. Despite this movie is not made yet, the steps that needs to be taken to make the movie are not very complex. During this research, the contour plots of the last sub-step were plotted. This last sub-step is just before divergence of the non-linear analysis. From some simple calculations, it became clear that the applied load just before divergence cannot be the critical buckling load. This means that divergence of the non-linear analysis is reached too early. In order to get contour-plots up to 'just before buckling', it is needed to solve the problem of the too early divergence.

When this problem is solved, the contour plots of every sub-step can be shown using the APDL code **`STEP,1,x`**, where **`x`** is the number of the sub-step. For each sub-step, the plot can be edited into a movie maker software and displayed one after another. In this way a kind of "stop-motion movie" of the changing contour plots is obtained.

Based upon the obtained results, the hypothesis stated in the introduction (page 11) cannot be rejected. This is due to the fact that the non-linear analysis stopped too early. However, it can be seen that during the compressive loading of a shell with imperfections, the imperfections become bigger. The membrane force becomes up to 10% lower and higher and the curvature becomes up to 20% lower as well as higher. As the non-linear analysis stopped too early, the changes in membrane force and curvature are expected to become bigger, as also the deformations will become bigger just before buckling.

Considering the change in both quantities, the critical buckling load of a shell with imperfections can be up to 1.32 times smaller than expected from theory. However, this doesn't explain the experimental results. Experiments show a critical buckling load of about 4.00 times smaller for $a/t < 500$ (Hoogenboom). Solving the problem with the divergence may result in a bigger change of the membrane force and the curvature, who finally may explain the experimental results.

By comparing the applied load at the top edge to the theoretical critical buckling load and taking into account experimental outcomes, one can conclude that the resulting plots are not just before buckling. This follows from the fact that the applied load at the top edge is about 10 times smaller in the results than what can be expected from theory. From this, one can

conclude that the divergence of the non-linear calculation is caused by something other than buckling.

The obtained deformed shape and the buckling length doesn't comply with the theory. The deformed shape differs a lot from expectations, independent of the boundary conditions and the a/t ratio. According to the analysis, the buckling lengths are much bigger than theoretical values. Also this observation is almost independent of the boundary conditions and the a/t ratio. However, during the analysis, buckling isn't reached yet. So, it is uncertain whether the deformed shape and the buckling length will change into results that meets the theory more, when buckling is almost reached. The deformed shape plot shows a blue wave pattern at the bottom edges (figure 5.1 and 5.8). This pattern is probably caused by the way the shell has been loaded. The deformed shape and buckling length may comply more with the theory when the load is applied due to a pre-described deformation, rather than applying a force.

6.2 Conclusions about the used methods and programs

The problem was first tried to solve with the use of Ansys workbench. With this program, the analysis up and until adding the imperfections was done. However, while trying to write a code to plot "user defined" contour plots, it became clear that this was very difficult to do. There were no clear instructions found about how the implement APDL code into Ansys workbench. Ansys workbench is very simple to handle for relatively simple problems. However, when a lot of APDL scripting is involved, it is preferred to not work with Ansys workbench.

As the supervisor was more experienced and was more aware of the possibilities in Ansys mechanical APDL, it was decided to start working with mechanical APDL. The benefit of mechanical APDL is the possibility to run a file with APDL code in one go. In this way the needed analysis and intermediate steps are done in one sequence. Also the code to create the contour plots can be implemented in the same file. Furthermore, one has more influence on how the mesh is created. Drawbacks of using mechanical APDL is the fact that it is needed to search a lot of script on internet, when one only wants to use APDL script. Furthermore, sometimes it can be very difficult to see what the program exactly has done after running the code, especially for unexperienced users.

6.3 Recommendations

In order to make a movie that demonstrates the formation of buckles with contour plots, several problems need to be solved. Also, some remarkable observations need to be investigated in more detail. Based upon the observations and problems during this research, the following recommendations for further investigation are suggested:

- Determine the cause of the divergence of the non-linear analysis before buckling is reached and how the settings of the non-linear buckling analysis can be changed in such a way that the non-linear analysis stops just before buckling.
- Loading the shell with a pre-described deformation rather than a force, in order to have the same axial deformation along each cross-section parallel to the top and bottom edge.
- Determine whether the obtained contour plots are correct or find an explanation why they doesn't comply with the Sagitta-calculation.

- Determine whether it is possible to store the values for the contour plots in a different way. Now they are stored into other quantities (displacement in x- and y direction), storing it in another way could be more neat.
- Determine whether it is possible to make an automatically generated animation of the contour-plots over all the different sub-steps in Ansys, rather than only display the contour plot of the last sub-step or making print-screens of every single sub-step.
- Changing the colour-scale of the kxx plot in such a way that positive and negative curvature can be distinguished in order to make a better judgement about the correctness of this contour-plot.
- Determine whether the APDL-code used in this research (see appendix 1) can be simplified or made more efficient.
- Determine why the buckling shape doesn't comply with the theory and whether the change of some parameters (i.e.: dimensions, magnitude of imperfections or material parameters) will result in a deformed shape which complies with the theory.
- Making the desired movie in which the formation of buckles in shells is demonstrated.

7 Sources

Distillers, O. (n.d.). *55 Gallon Stainless Steel Drum.* Retrieved from Olympic Distillers: https://www.olympicdistillers.com/stainless-steel-barrels/55-gallon-stainless-steel-drum-open-top

Hoogenboom, P. C. (n.d.). *handout 8.* Retrieved from CIE4143 Shell Analysis, Theory and Application: http://homepage.tudelft.nl/p3r3s/b17_schedule.html

Physics. (n.d.). *Synopsis: Crumpling Coke Cans*. Retrieved from Physics: https://physics.aps.org/synopsis-for/10.1103/PhysRevLett.119.224101

Wright, D. (2005, May). *Buckling.* Retrieved from MDP: http://www-mdp.eng.cam.ac.uk/web/library/enginfo/textbooks_dvd_only/DAN/buckling/intro/intro.html

Illustration at the cover pages (colour edited): (Physics, n.d.)

Appendix 1: APDL commands (complete code)

```
! shell buckling contour plots, 13 June 2019
! Bachelor Thesis Guido Hogendoorn (4570928)
! Supervisors: dr. ir. P.C.J. Hoogenboom and dr. ir. R. Abspoel

E   = 2.1e5       ! N/mm2     Young's modulus
nu  = 0.35        ! -         Poisson's ratio
h   = 250         ! mm        length
a   = 60          ! mm        radius of curvature
t   = 0.12        ! mm        thickness
q1  = 1           ! N/mm      load along top edge
rho = 7000e-9     ! kg/mm3    mass density
n   = 50          ! -         number of elements in the x or y direction
d   = t/2         ! mm        amplitude of imperfection
pi  = acos(-1)    ! -         pi-value

/PREP7
MPTEMP,,,,,,,,                ! material: isotropic
MPTEMP,1,0
MPDATA,EX,1,,E
MPDATA,PRXY,1,,nu
MPDATA,DENS,1,,rho
ET,1,SHELL181                 ! element type: 4 node quadrilateral
R,1,t,t,t,t, , ,              ! element thickness


*DO,i,0,n                     ! insert nodes
 *DO,j,0,n-1
   x=a*cos(j*2*pi/n)
   y=a*sin(j*2*pi/n)
   z=i*h/n
   N,,x,y,z,,,
 *ENDDO
*ENDDO

*DO,i,1,n                     ! insert elements
 *DO,j,1,n-1
  L=(i-1)*n+j
  E,L+1,L+n+1,L+n,L
 *ENDDO
 L=i*n
 E,L-n+1,L+1,L+n,L
*ENDDO

*DO,i,1,n                     ! boundary conditions bottom edge
```

```
 D,i,UX,0
 D,i,UY,0
 D,i,UZ,0
 D,i,ROTX,0
 D,i,ROTY,0
 D,i,ROTZ,0
*ENDDO

*DO,i,n*n+1,n*(n+1)          ! boundary conditions top edge
 D,i,UX,0
 D,i,UY,0
 D,i,ROTX,0
 D,i,ROTY,0
 D,i,ROTZ,0
*ENDDO

*DO,i,n*n+1,n*(n+1)          ! insert load on top edge
  F,i,FZ,-q1*2*pi*a/n
*ENDDO
FINISH

/SOLU                        ! compute linear elastic
SOLVE
FINISH

/SOLU                        ! compute buckling modes
ANTYPE, MODAL
MODOPT, LANB, 1,,,
MXPAND,20
PSTRES,ON
SOLVE
FINISH

/PREP7                       ! inserting imperfections
UPGEOM,d,1,1,'file',rst
FINISH

/SOLU
NCNV,0                       ! Do not terminate program if not-converged
ANTYPE, STATIC               ! Static analysis
NLGEOM, ON                   ! Nonlinear geometry
TIME, 1                      ! Time at the end of load step
nsubstep = 20                ! # of substeps
NSUBST,nsubstep,,            ! Number of substeps
NEQIT,100,                   ! Max. number of iterations
CNVTOL,STAT                  ! Convergence tolerance (default)
RESCONTROL,DEFINE,ALL,1,     ! Write new files at every substep
OUTRES,NSOL,ALL,,,,          ! Write nodal results at every substep
OUTRES,ESOL,ALL,,,,          ! Write element results at every substep
SOLVE
```

```
FINISH


/POST1
*DO,i,0,n                          ! calculating (nxx+nyy) / 2
 *DO,j,0,n-1
   nnode = j + i*n + 1
   SHELL, MID                      ! Averged value of TOP and BOTTOM
   *GET,s1,NODE,nnode,S,1
   *GET,s2,NODE,nnode,S,2
   *GET,s3,NODE,nnode,S,3
   nxxyy = (s1+s2+s3)*t/2
   DNSOL,nnode,U,X,nxxyy       ! the desired value will be stored into u_x
 *ENDDO
*ENDDO

*DO,i,1,n-1                        ! calculating (kyy+kxx)/2
 *DO,j,1,n-2
   nnodeII = j + i*n + 1
   *GET,x01,NODE,nnodeII-1,LOC,x   ! original x-coordinates
   *GET,x02,NODE,nnodeII,LOC,x
   *GET,x03,NODE,nnodeII+1,LOC,x
   *GET,x04,NODE,nnodeII-n,LOC,x
   *GET,x05,NODE,nnodeII+n,LOC,x

   *GET,y01,NODE,nnodeII-1,LOC,y   ! original y-coordinates
   *GET,y02,NODE,nnodeII,LOC,y
   *GET,y03,NODE,nnodeII+1,LOC,y
   *GET,y04,NODE,nnodeII-n,LOC,y
   *GET,y05,NODE,nnodeII+n,LOC,y

   *GET,z02,NODE,nnodeII,LOC,z     ! original z-coordinates
   *GET,z04,NODE,nnodeII-n,LOC,z
   *GET,z05,NODE,nnodeII+n,LOC,z

   *GET,ux1,NODE,nnodeII-1,U,x     ! displacements in x-direction
   *GET,ux2,NODE,nnodeII,U,x
   *GET,ux3,NODE,nnodeII+1,U,x
   *GET,ux4,NODE,nnodeII-n,U,x
   *GET,ux5,NODE,nnodeII+n,U,x

   *GET,uy1,NODE,nnodeII-1,U,y     ! displacements in y-direction
   *GET,uy2,NODE,nnodeII,U,y
   *GET,uy3,NODE,nnodeII+1,U,y
   *GET,uy4,NODE,nnodeII-n,U,y
   *GET,uy5,NODE,nnodeII+n,U,y

   *GET,uz2,NODE,nnodeII,U,z       ! displacements in z-direction
   *GET,uz4,NODE,nnodeII-n,U,z
   *GET,uz5,NODE,nnodeII+n,U,z
```

```
   x1 = x01 + ux1                    ! new coordinates
   x2 = x02 + ux2
   x3 = x03 + ux3
   y1 = y01 + uy1
   y2 = y02 + uy2
   y3 = y03 + uy3
   z2 = z02 + uz2
   z4 = z04 + uz4
   z5 = z05 + uz5
   r2 = SQRT((x02+ux2)**2+(y02+uy2)**2)
   r4 = SQRT((x04+ux4)**2+(y04+uy4)**2)
   r5 = SQRT((x05+ux5)**2+(y05+uy5)**2)

   ay = SQRT((x2-x1)**2+(y2-y1)**2)      ! calculating kyy
   by = SQRT((x3-x2)**2+(y3-y2)**2)
   cy = SQRT((x3-x1)**2+(y3-y1)**2)
   qy = (ay**2+by**2-cy**2)/(2*ay*by)
   Ryy = cy / (2*SQRT(1-qy**2))
   kyy = -1/Ryy

   ax = SQRT((r2-r4)**2+(z2-z4)**2)      ! calculating kxx
   bx = SQRT((r5-r2)**2+(z5-z2)**2)
   cx = SQRT((r5-r4)**2+(z5-z4)**2)
   qx = (ax**2+bx**2-cx**2)/(2*ax*bx)
   Rxx = cx / (2*SQRT(1-qx**2))

   rcl = ((r4-r5)/(z4-z5))*(z2-z4)+r4    ! sign of kxx
   *IF,r2,LE,rcl,THEN
    kxx = 1/Rxx
   *ELSE
    kxx = -1/Rxx
   *ENDIF

   kxxyy = (kyy + kxx) / 2     ! calculating (kyy + kxx) / 2 and store
   DNSOL,nnodeII,U,Y,kxxyy
 *ENDDO
*ENDDO


dz = 1/1000

                               ! calculating (kyy+kxx)/2 at top edge
*DO,j,1,n-2
 i = n
   nnodeII = j + i*n + 1
   *GET,x01,NODE,nnodeII-1,LOC,x       ! original x-coordinates
   *GET,x02,NODE,nnodeII,LOC,x
   *GET,x03,NODE,nnodeII+1,LOC,x
   *GET,x04,NODE,nnodeII-n,LOC,x
```

```
*GET,y01,NODE,nnodeII-1,LOC,y    ! original y-coordinates
*GET,y02,NODE,nnodeII,LOC,y
*GET,y03,NODE,nnodeII+1,LOC,y
*GET,y04,NODE,nnodeII-n,LOC,y

*GET,z02,NODE,nnodeII,LOC,z      ! original z-coordinates
*GET,z04,NODE,nnodeII-n,LOC,z

*GET,ux1,NODE,nnodeII-1,U,x      ! displacements in x-direction
*GET,ux2,NODE,nnodeII,U,x
*GET,ux3,NODE,nnodeII+1,U,x
*GET,ux4,NODE,nnodeII-n,U,x

*GET,uy1,NODE,nnodeII-1,U,y      ! displacements in y-direction
*GET,uy2,NODE,nnodeII,U,y
*GET,uy3,NODE,nnodeII+1,U,y
*GET,uy4,NODE,nnodeII-n,U,y

*GET,uz2,NODE,nnodeII,U,z        ! displacements in z-direction
*GET,uz4,NODE,nnodeII-n,U,z

x1 = x01 + ux1                   ! new coordinates
x2 = x02 + ux2
x3 = x03 + ux3
y1 = y01 + uy1
y2 = y02 + uy2
y3 = y03 + uy3
z2 = z02 + uz2
z4 = z04 + uz4
z5 = z2 + dz
r2 = SQRT((x02+ux2)**2+(y02+uy2)**2)
r4 = SQRT((x04+ux4)**2+(y04+uy4)**2)
r5 = r2

ay = SQRT((x2-x1)**2+(y2-y1)**2)      ! calculating kyy
by = SQRT((x3-x2)**2+(y3-y2)**2)
cy = SQRT((x3-x1)**2+(y3-y1)**2)
qy = (ay**2+by**2-cy**2)/(2*ay*by)
Ryy = cy / (2*SQRT(1-qy**2))
kyy = -1/Ryy

ax = SQRT((r2-r4)**2+(z2-z4)**2)      ! calculating kxx
bx = SQRT((r5-r2)**2+(z5-z2)**2)
cx = SQRT((r5-r4)**2+(z5-z4)**2)
qx = (ax**2+bx**2-cx**2)/(2*ax*bx)
Rxx = cx / (2*SQRT(1-qx**2))

rcl = ((r4-r5)/(z4-z5))*(z2-z4)+r4    ! sign of kxx
*IF,r2,LE,rcl,THEN
```

```
    kxx = 1/Rxx
   *ELSE
    kxx = -1/Rxx
   *ENDIF

   kxxyy = (kyy + kxx) / 2      ! calculating (kyy + kxx) / 2 and store
   DNSOL,nnodeII,U,Y,kxxyy
*ENDDO

                                ! calculating (kyy+kxx)/2 bottom edge
 *DO,j,1,n-2
 i = 0
   nnodeII = j + i*n + 1
   *GET,x01,NODE,nnodeII-1,LOC,x       ! original x-coordinates
   *GET,x02,NODE,nnodeII,LOC,x
   *GET,x03,NODE,nnodeII+1,LOC,x
   *GET,x05,NODE,nnodeII+n,LOC,x

   *GET,y01,NODE,nnodeII-1,LOC,y       ! original y-coordinates
   *GET,y02,NODE,nnodeII,LOC,y
   *GET,y03,NODE,nnodeII+1,LOC,y
   *GET,y05,NODE,nnodeII+n,LOC,y

   *GET,z02,NODE,nnodeII,LOC,z         ! original z-coordinates
   *GET,z05,NODE,nnodeII+n,LOC,z

   *GET,ux1,NODE,nnodeII-1,U,x         ! displacements in x-direction
   *GET,ux2,NODE,nnodeII,U,x
   *GET,ux3,NODE,nnodeII+1,U,x
   *GET,ux5,NODE,nnodeII+n,U,x

   *GET,uy1,NODE,nnodeII-1,U,y         ! displacements in y-direction
   *GET,uy2,NODE,nnodeII,U,y
   *GET,uy3,NODE,nnodeII+1,U,y
   *GET,uy5,NODE,nnodeII+n,U,y

   *GET,uz2,NODE,nnodeII,U,z           ! displacements in z-direction
   *GET,uz5,NODE,nnodeII+n,U,z

   x1 = x01 + ux1                      ! new coordinates
   x2 = x02 + ux2
   x3 = x03 + ux3
   y1 = y01 + uy1
   y2 = y02 + uy2
   y3 = y03 + uy3
   z2 = z02 + uz2
   z4 = z2 - dz
   z5 = z05 + uz5
   r2 = SQRT((x02+ux2)**2+(y02+uy2)**2)
   r4 = r2
```

```
   r5 = SQRT((x05+ux5)**2+(y05+uy5)**2)

   ay = SQRT((x2-x1)**2+(y2-y1)**2)  ! calculating kyy
   by = SQRT((x3-x2)**2+(y3-y2)**2)
   cy = SQRT((x3-x1)**2+(y3-y1)**2)
   qy = (ay**2+by**2-cy**2)/(2*ay*by)
   Ryy = cy / (2*SQRT(1-qy**2))
   kyy = -1/Ryy

   ax = SQRT((r2-r4)**2+(z2-z4)**2)      ! calculating kxx
   bx = SQRT((r5-r2)**2+(z5-z2)**2)
   cx = SQRT((r5-r4)**2+(z5-z4)**2)
   qx = (ax**2+bx**2-cx**2)/(2*ax*bx)
   Rxx = cx / (2*SQRT(1-qx**2))

   rcl = ((r4-r5)/(z4-z5))*(z2-z4)+r4    ! sign of kxx
   *IF,r2,LE,rcl,THEN
    kxx = 1/Rxx
   *ELSE
    kxx = -1/Rxx
   *ENDIF

   kxxyy = (kyy + kxx) / 2       ! calculating (kyy + kxx) / 2 and store
   DNSOL,nnodeII,U,Y,kxxyy
*ENDDO


*DO,i,1,n-1                      ! calculating (kyy+kxx)/2 begin of
circumference
 j=0
   nnodeII = j + i*n + 1
   *GET,x01,NODE,nnodeII+n-1,LOC,x       ! original x-coordinates
   *GET,x02,NODE,nnodeII,LOC,x
   *GET,x03,NODE,nnodeII+1,LOC,x
   *GET,x04,NODE,nnodeII-n,LOC,x
   *GET,x05,NODE,nnodeII+n,LOC,x

   *GET,y01,NODE,nnodeII+n-1,LOC,y       ! original y-coordinates
   *GET,y02,NODE,nnodeII,LOC,y
   *GET,y03,NODE,nnodeII+1,LOC,y
   *GET,y04,NODE,nnodeII-n,LOC,y
   *GET,y05,NODE,nnodeII+n,LOC,y

   *GET,z02,NODE,nnodeII,LOC,z           ! original z-coordinates
   *GET,z04,NODE,nnodeII-n,LOC,z
   *GET,z05,NODE,nnodeII+n,LOC,z

   *GET,ux1,NODE,nnodeII+n-1,U,x         ! displacements in x-direction
   *GET,ux2,NODE,nnodeII,U,x
   *GET,ux3,NODE,nnodeII+1,U,x
```

```
   *GET,ux4,NODE,nnodeII-n,U,x
   *GET,ux5,NODE,nnodeII+n,U,x

   *GET,uy1,NODE,nnodeII+n-1,U,y          ! displacements in y-direction
   *GET,uy2,NODE,nnodeII,U,y
   *GET,uy3,NODE,nnodeII+1,U,y
   *GET,uy4,NODE,nnodeII-n,U,y
   *GET,uy5,NODE,nnodeII+n,U,y

   *GET,uz2,NODE,nnodeII,U,z              ! displacements in z-direction
   *GET,uz4,NODE,nnodeII-n,U,z
   *GET,uz5,NODE,nnodeII+n,U,z

   x1 = x01 + ux1                         ! new coordinates
   x2 = x02 + ux2
   x3 = x03 + ux3
   y1 = y01 + uy1
   y2 = y02 + uy2
   y3 = y03 + uy3
   z2 = z02 + uz2
   z4 = z04 + uz4
   z5 = z05 + uz5
   r2 = SQRT((x02+ux2)**2+(y02+uy2)**2)
   r4 = SQRT((x04+ux4)**2+(y04+uy4)**2)
   r5 = SQRT((x05+ux5)**2+(y05+uy5)**2)

   ay = SQRT((x2-x1)**2+(y2-y1)**2)       ! calculating kyy
   by = SQRT((x3-x2)**2+(y3-y2)**2)
   cy = SQRT((x3-x1)**2+(y3-y1)**2)
   qy = (ay**2+by**2-cy**2)/(2*ay*by)
   Ryy = cy / (2*SQRT(1-qy**2))
   kyy = -1/Ryy

   ax = SQRT((r2-r4)**2+(z2-z4)**2)       ! calculating kxx
   bx = SQRT((r5-r2)**2+(z5-z2)**2)
   cx = SQRT((r5-r4)**2+(z5-z4)**2)
   qx = (ax**2+bx**2-cx**2)/(2*ax*bx)
   Rxx = cx / (2*SQRT(1-qx**2))

   rcl = ((r4-r5)/(z4-z5))*(z2-z4)+r4     ! sign of kxx
   *IF,r2,LE,rcl,THEN
    kxx = 1/Rxx
   *ELSE
    kxx = -1/Rxx
   *ENDIF

   kxxyy = (kyy + kxx) / 2      ! calculating (kyy + kxx) / 2 and store
   DNSOL,nnodeII,U,Y,kxxyy
*ENDDO
```

```
*DO,i,1,n-1                ! calculating (kyy+kxx)/2 end of circumference
 j=n-1
   nnodeII = j + i*n + 1
   *GET,x01,NODE,nnodeII-1,LOC,x       ! original x-coordinates
   *GET,x02,NODE,nnodeII,LOC,x
   *GET,x03,NODE,nnodeII-n+1,LOC,x
   *GET,x04,NODE,nnodeII-n,LOC,x
   *GET,x05,NODE,nnodeII+n,LOC,x

   *GET,y01,NODE,nnodeII-1,LOC,y       ! original y-coordinates
   *GET,y02,NODE,nnodeII,LOC,y
   *GET,y03,NODE,nnodeII-n+1,LOC,y
   *GET,y04,NODE,nnodeII-n,LOC,y
   *GET,y05,NODE,nnodeII+n,LOC,y

   *GET,z02,NODE,nnodeII,LOC,z         ! original z-coordinates
   *GET,z04,NODE,nnodeII-n,LOC,z
   *GET,z05,NODE,nnodeII+n,LOC,z

   *GET,ux1,NODE,nnodeII-1,U,x         ! displacements in x-direction
   *GET,ux2,NODE,nnodeII,U,x
   *GET,ux3,NODE,nnodeII-n+1,U,x
   *GET,ux4,NODE,nnodeII-n,U,x
   *GET,ux5,NODE,nnodeII+n,U,x

   *GET,uy1,NODE,nnodeII-1,U,y         ! displacements in y-direction
   *GET,uy2,NODE,nnodeII,U,y
   *GET,uy3,NODE,nnodeII-n+1,U,y
   *GET,uy4,NODE,nnodeII-n,U,y
   *GET,uy5,NODE,nnodeII+n,U,y

   *GET,uz2,NODE,nnodeII,U,z           ! displacements in z-direction
   *GET,uz4,NODE,nnodeII-n,U,z
   *GET,uz5,NODE,nnodeII+n,U,z

   x1 = x01 + ux1                      ! new coordinates
   x2 = x02 + ux2
   x3 = x03 + ux3
   y1 = y01 + uy1
   y2 = y02 + uy2
   y3 = y03 + uy3
   z2 = z02 + uz2
   z4 = z04 + uz4
   z5 = z05 + uz5
   r2 = SQRT((x02+ux2)**2+(y02+uy2)**2)
   r4 = SQRT((x04+ux4)**2+(y04+uy4)**2)
   r5 = SQRT((x05+ux5)**2+(y05+uy5)**2)

   ay = SQRT((x2-x1)**2+(y2-y1)**2)      ! calculating kyy
```

```
   by = SQRT((x3-x2)**2+(y3-y2)**2)
   cy = SQRT((x3-x1)**2+(y3-y1)**2)
   qy = (ay**2+by**2-cy**2)/(2*ay*by)
   Ryy = cy / (2*SQRT(1-qy**2))
   kyy = -1/Ryy

   ax = SQRT((r2-r4)**2+(z2-z4)**2)      ! calculating kxx
   bx = SQRT((r5-r2)**2+(z5-z2)**2)
   cx = SQRT((r5-r4)**2+(z5-z4)**2)
   qx = (ax**2+bx**2-cx**2)/(2*ax*bx)
   Rxx = cx / (2*SQRT(1-qx**2))

   rcl = ((r4-r5)/(z4-z5))*(z2-z4)+r4    ! sign of kxx
   *IF,r2,LE,rcl,THEN
    kxx = 1/Rxx
   *ELSE
    kxx = -1/Rxx
   *ENDIF

   kxxyy = (kyy + kxx) / 2      ! calculating (kyy + kxx) / 2 and store
   DNSOL,nnodeII,U,Y,kxxyy
*ENDDO

*GET,kxxyybb,NODE,2,U,Y                ! 4 points at the intersection
DNSOL,1,U,Y,kxxyybb                     of irregularity zones
*GET,kxxyybe,NODE,n-1,U,Y
DNSOL,n,U,Y,kxxyybe
*GET,kxxyytb,NODE,n*n+2,U,Y
DNSOL,n*n+1,U,Y,kxxyytb
*GET,kxxyyte,NODE,n*n+n-1,U,Y
DNSOL,n*n+n,U,Y,kxxyytb

FINISH
```

Appendix 2: variations in thickness t and boundary conditions

Table A2.1 – comparison between different shell thicknesses t **(Fixed supports)**

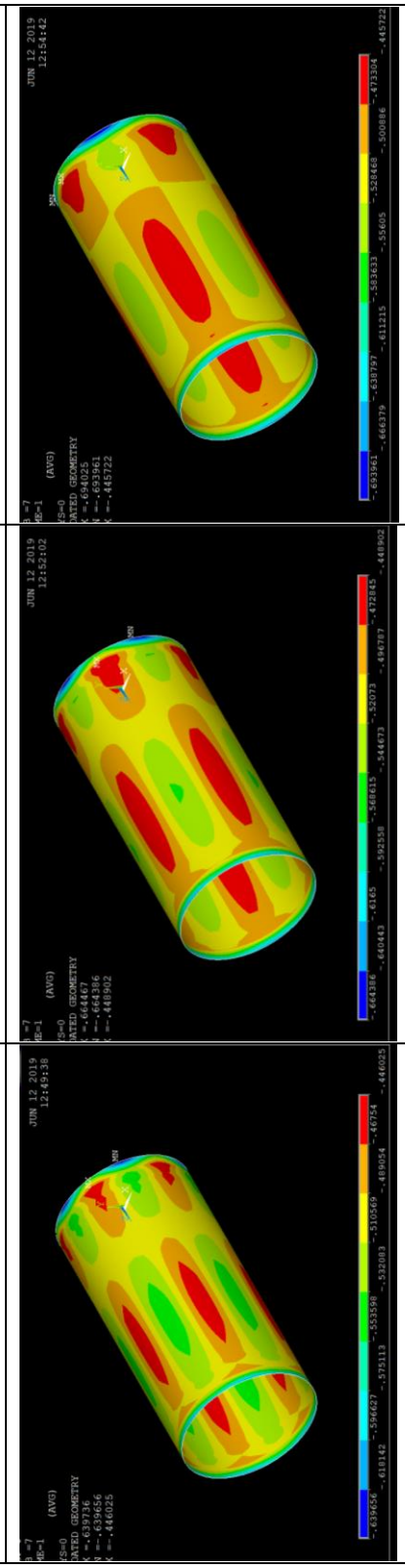
t = 0.12 mm (a/t = 500)	t = 0.24 mm (a/t = 250)	t = 0.60 mm (a/t = 100)
		
(nxx + nyy)/2	(nxx + nyy)/2	(nxx + nyy)/2
(kxx + kyy)/2	(kxx + kyy)/2	(kxx + kyy)/2

Table A2.2 – comparison between different shell thicknesses t **(Edge rotations are free)**

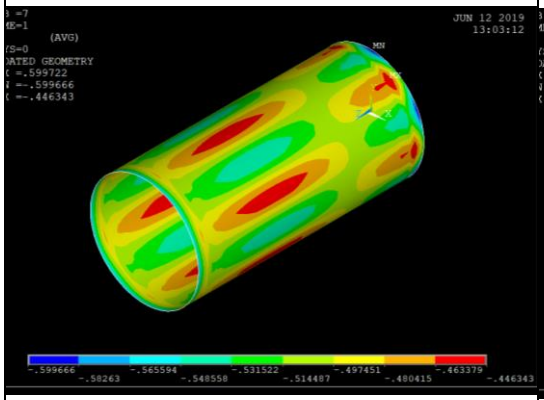
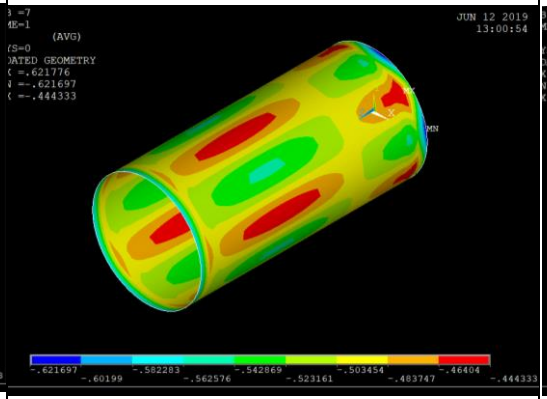
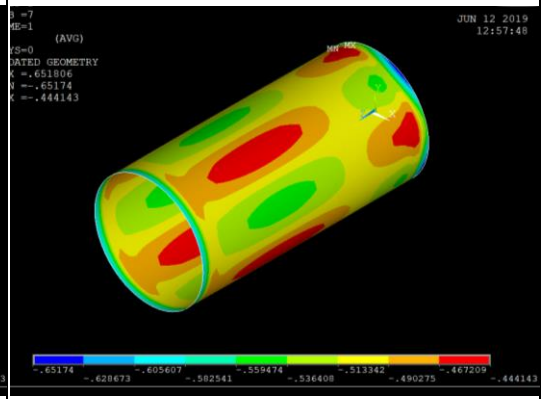
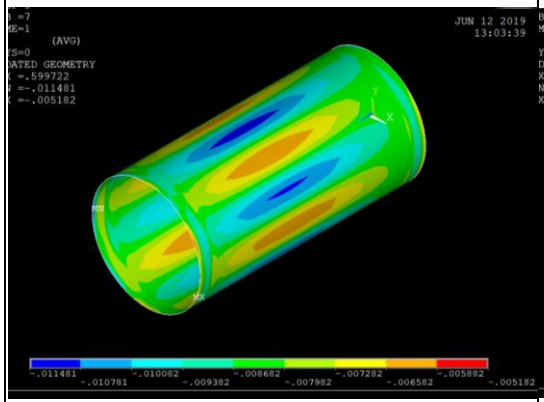
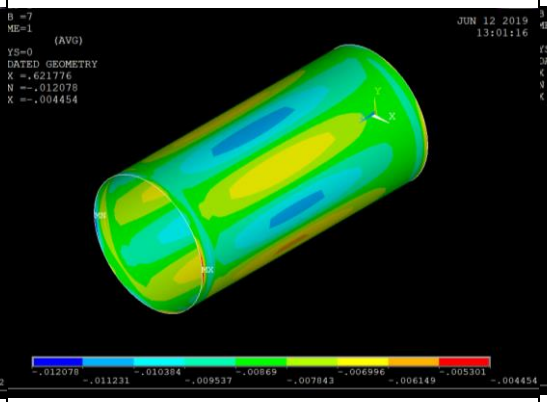
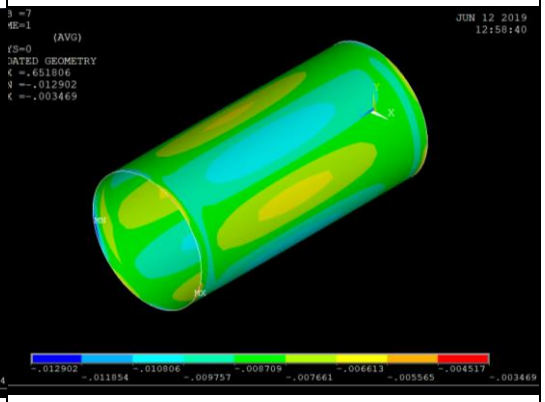
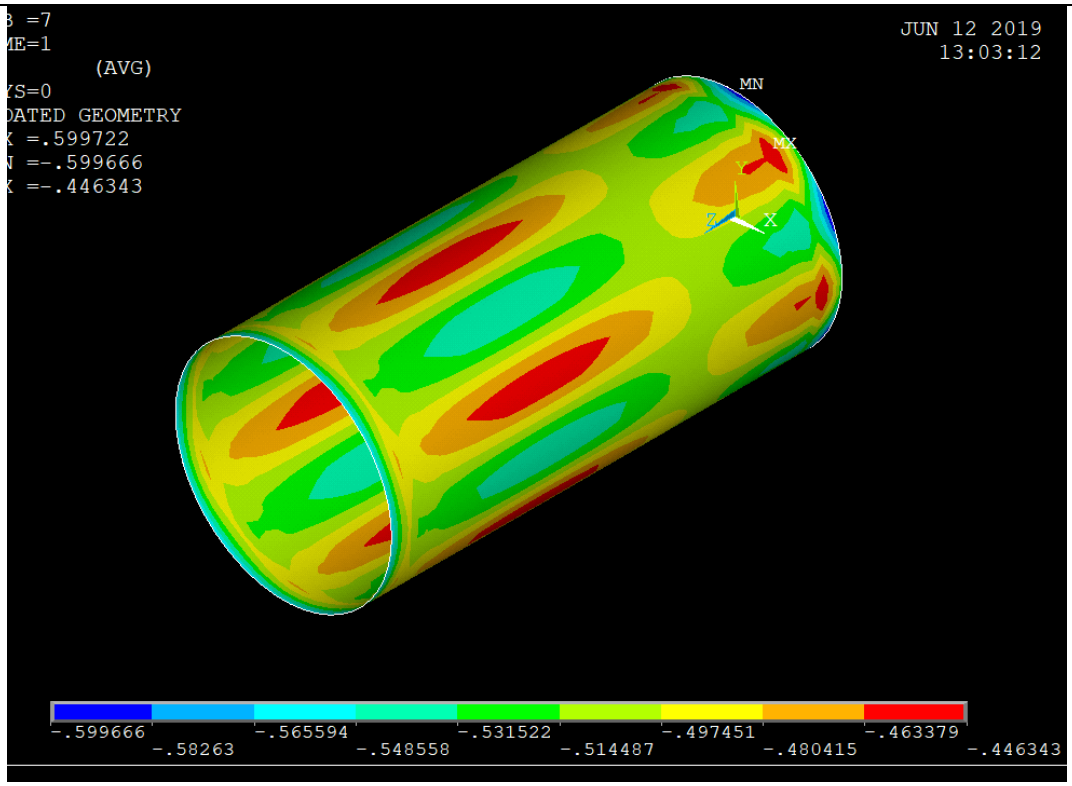
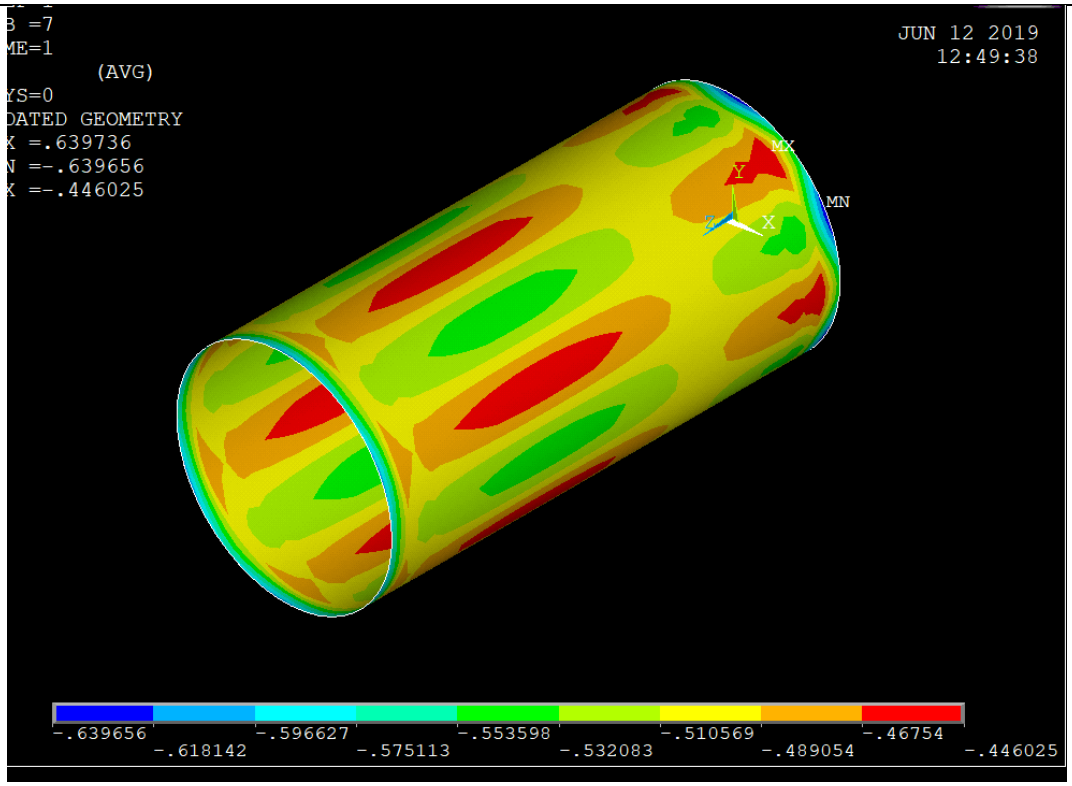
t = 0.12 mm (a/t = 500)	**t = 0.24 mm (a/t = 250)**	**t = 0.60 mm (a/t = 100)**
		
(nxx + nyy)/2	**(nxx + nyy)/2**	**(nxx + nyy)/2**
		
(kxx + kyy)/2	**(kxx + kyy)/2**	**(kxx + kyy)/2**

Table A2.3 – comparison of (nxx + nyy)/2 between different boundary conditions **(for t = 0.12 mm)**.

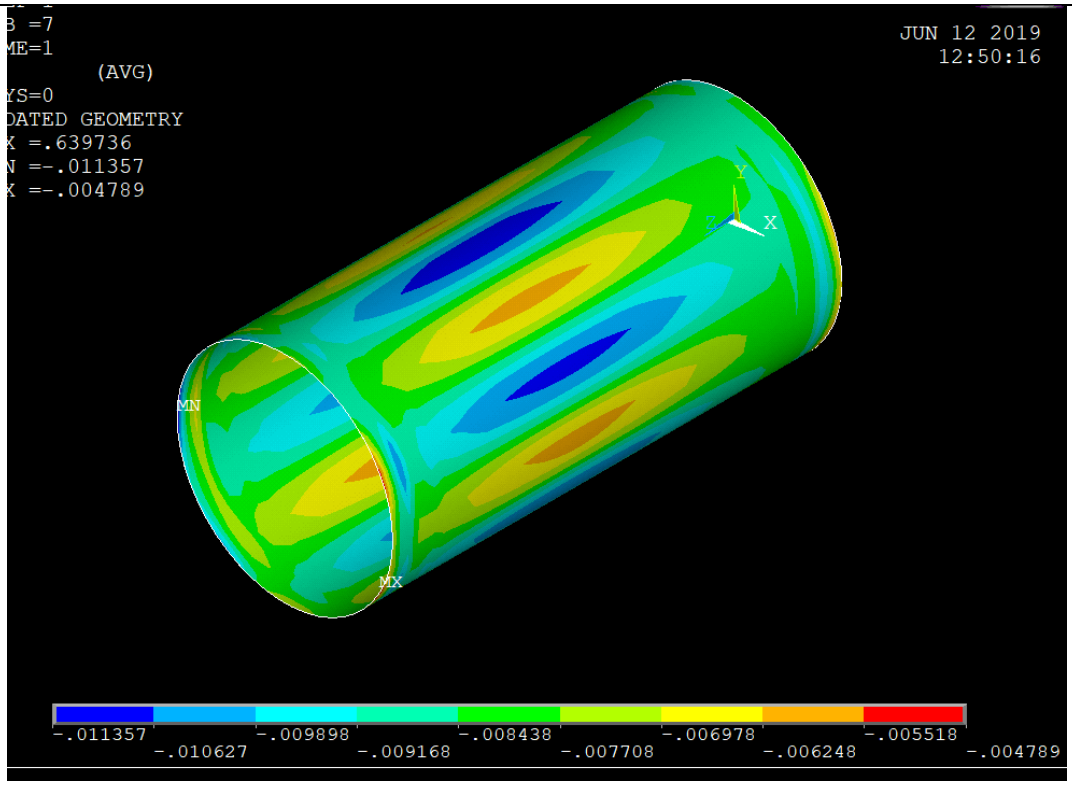


Fixed rotations

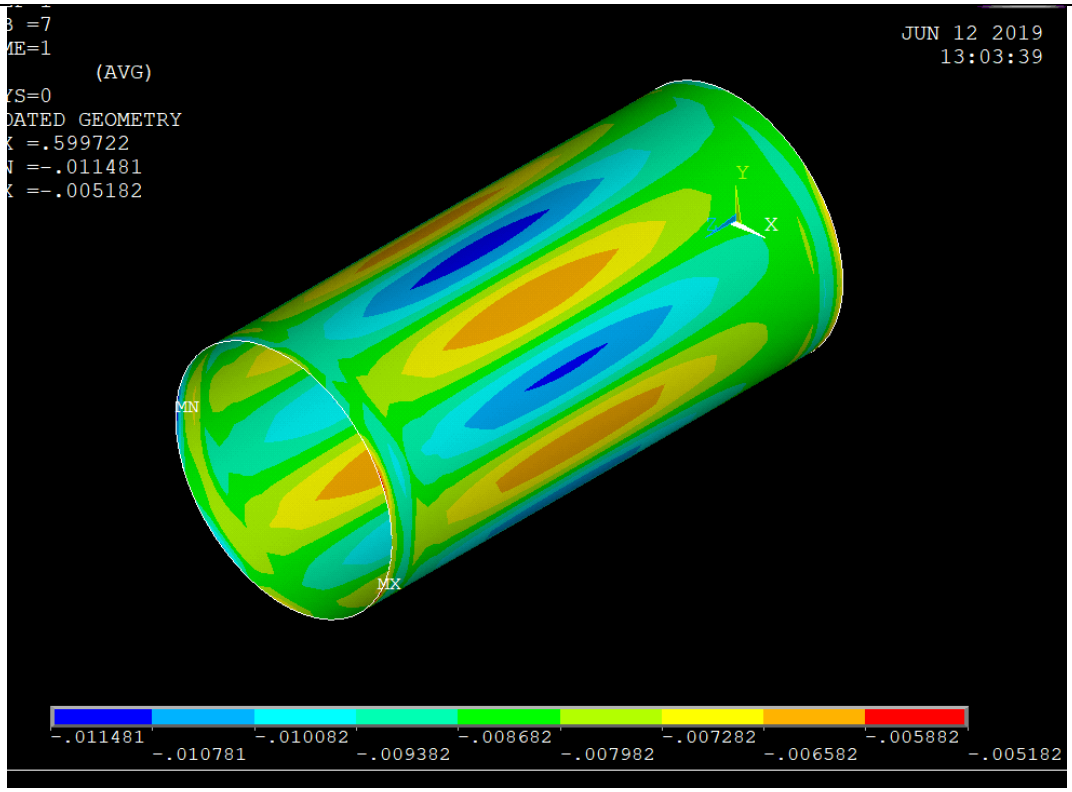


Edge rotations allowed

Table A2.4 – comparison of (kxx + kyy)/2 between different boundary conditions **(for t = 0.12 mm)**.



Fixed rotations



Edge rotations allowed

TU Delft