Strain Based Analysis for Dented Pipelines

by
Mahyar Mehranfar

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Structural Engineering

Department of Civil and Environmental Engineering

University of Alberta

Abstract

It is easier and cheaper to transmit oil and gas by pipeline, but their failure can cause considerable environmental and societal consequences. The denting of pipelines is one of the significant challenges faced by those in the oil and gas industry. The formation of dents in the wall of the pipeline can cause lower pressure capacity. Analytical and numerical models, such as the finite element method (FEA), can predict this issue.

The traditional way for recognizing the seriousness of the dent is to test the dent depth. But unfortunately, this method cannot predict the probability of failure accurately. Based on previous research, there are two ways to assess the seriousness of the dents. The first method is to model the pipe by finite element method. While very accurate, the finite element method is very computationally demanding and time consuming. The second method utilizes the dent profile to perform strain-based analysis. While very fast, the method suffers from lack of accuracy particularly in predicting the strains in the longitudinal direction.

The first objective of this research was to develop a technique that takes into consideration the membrane strains in the longitudinal direction. The second objective was to test the performance of the new technique on a variety of pipeline dents. The developed method is based on the three-dimensional mathematical model proposed by Okoloekwe et al.

In the original model proposed by Okoloekwe et al, it was assumed that the displacement in the mid-line of the pipeline is zero, but we found this displacement and added it to the displacement in the horizontal direction. Our study found that the modification yielded significantly better longitudinal strain distribution than the conventional procedure. The newly developed methods provide an increase in accuracy and speed of the analytical process without sacrificing accuracy.

A number of two-dimensional and three-dimensional models were examined to verify the method. Contrary to the longitudinal results, these results were very accurate in the circumferential direction. With respect to the FEA results, our proposed technique is much faster, more accurate, and more reliable than previously developed analytical methods.

Preface

This thesis is an original work by Mahyar Mehranfar.

My responsibilities included collecting data, analyzing it, and developing the manuscript. Both S. Adeeb and Nader Yoosef-Ghodsi contributed to concept creation, provided technical support, and reviewed the manuscript, while S. Adeeb also worked as the primary supervisor. During and outside of class as well as our academic meetings, I have learned quite a bit from his guidance.

Acknowledgements

Throughout my research, I have received the continuous support and guidance of my supervisor, Samer Adeeb. During and outside of class as well as our academic meetings, I have learned quite a bit from his guidance.

I would like to thank the Faculty of Graduate Studies at the University of Alberta, for providing me the right environment for my research.

For providing me with the ideal research environment, I would like to thank the Faculty of Graduate Studies at the University of Alberta.

Enbridge Inc. also deserves appreciation for the financial support, onsite training, and valuable data access that enabled the study to be completed more rapidly and with more rigor.

In closing, I would like to thank my wife Yasaman, my family, and friends for all their love and support during the years I spent studying.

TABLE OF CONTENTS

List of Tables

List of Figures

CHAPTER 1: INTRODUCTION

1.1 Background and Problem Statement

In the third quarter of 2019, compared to a day in the previous quarter, worldwide demand increased from 435,000 barrels a day to 1.1 million barrels a day (i.e., more than double) (www.cnbc.com, 2019). Although pipelines are the easiest and cheapest way of transporting oil and gas, their failure might have dire environmental and societal consequences. One of the significant challenges faced by those in the Oil and Gas industry is denting of pipelines. Dents are defined as a severe disturbance of the circular cross-section of the pipes which are formed by contact with an external body (Cosham & Hopkins, 2004, Makhlouf, A. S. H., & Aliofkhazraei, M. (Eds.), 2015). A dent is a permanent deformation of the pipe's circular cross section caused by the plastic deformation. Dents distort the cross section of the pipe significantly (Cosham and Hopkins, 2003). The following terminology is used for classifying dents.

1. Smooth dents are caused by a smooth change in the shape of the pipe wall.

2. Kinked dents as a result of an abrupt change in the curvature of the pipe wall.

3. Plain dents lead to a smooth dent without any thickness reductions.

4. Constrained dents are defined as dents are prevented from rebounding and re-rounding as a result of the persistent surface to surface contact with indenter.

5. Unconstrained dents result from a situation in which a dent is permitted to rebound and re-round elastically when the indenting body is removed.

There are many leading causes for mechanical damages that can cause dents in pipelines; such as human activities in transportation or installation, mechanical third-party line strikes, or natural disasters like earthquakes and landslides.

The traditional way for recognizing the seriousness of the dent is to test the depth. Dents acceptance criteria in various codes and standards are based on the dent depth. Table 1-1 provides acceptance criteria for dents according to their dent depth. For example, for pipes with an outside diameter (OD) more than 101.6, plain dents deeper than 6% of the (OD) according to CSA Z662-19 (CSA,2026) are not acceptable. For other pipelines, CSA Z662-19 states that the dent should not be deeper than 6mm. On the other hand, when using depth-based judgments, this sometimes leads to unnecessary excavation or mischaracterization for dents less than 6% OD. For example, the National Energy Board reported a leak at a crack within a dent with a maximum depth of 0.51% of the pipe OD, demonstrating the unreliability of this approach (National Energy Board Safety Advisory 2010). A new approach for solving the problems of depth-based criteria, is strain-based assessment (Noronha et al., 2005, Okoloekwe, C., 2017).

Table 1-1. Acceptability limits for plain dents (Race, et al., 2010).

	PLAIN DENTS	
	Constrained	**Unconstrained**
ASME B31.8	**Up to 6% OD or strain level up to 6%**	
ASME B31.4	**Up to 6% OD in pipe diameters > NPS4”**	

	Up to 6 mm in pipe diameters < NPS 4”	
API 1156	**Up to 6% OD but > 2% OD requires a fatigue assessment**	
EPRG	**≤ 7% OD at a hoop stress of 72% SMYS**	
PDAM	**Up to 10% OD**	**Up to 7% OD**
CSA Z662	**Up to 6 mm for ≤ 101.61 mm OD, or < 6% OD for > 101.6 mm OD**	

Table 1 Acceptability limits for plain dents

There is a good connection between the shape of a dent and the associated mechanical strain; strain can be predicted by the analytical and numerical models with regard to size and location. Finite Element Analysis (FEA) is an acceptable tool for analyzing the mechanical behavior of pipelines subjected to various mechanical disturbances such as dents and gouges. On the other hand, FEA is costly and time-consuming, so it is ineffectual for analyzing numerous dents (Okoloekwe, C., Kainat, M., Langer, D., Hassanien, S., Roger Cheng, J., and Adeeb, S., 2018). A quick and accurate evaluation of dents requires the employment of an analytical approach. The non-mandatory equations presented in Appendix R of the ASME B31.8 2018 (ASME, 2016) codes evaluate strains based on the minimum radius of curvature of each dent in the axial and circumferential direction. It is specified that the limitation for strain value for both the inner and outer sides of plain dents in the pipelines is 6%.

Furthermore, in the presence of stress concentration like cold worked areas and seam welds, this limitation will be diminished (ASME, 2016). On the other hand, with ASME equations, the length of a dent, which is needed for strain analysis, is not well defined. The length corresponding to half depth of the dent is suggested as a familiar way of settling the length of the dent. Although the previous method is straightforward, the research by Noronha et al. (Noronha et al., Martins, R.R., Jacob, B.P. and Souza, E., 2005) shows that compared to the FEA method, this solution would underestimate the values of the longitudinal strain. Plus, there are no agreed upon accurate methods for approximating the size of the radius of curvature of the dented pipe, which is the paramount parameters used by the ASME equations to estimate the maximum strain within the dented region. Also, the ASME equations assume that the maximum strain will occur at the peak of the dent which is not always the case. Okoloekwe et al. (Okoloekwe, C et al., 2018) proposed to solve these problems by applying piecewise spline functions to interpolate the dent topology accompanied by equations to calculate the radius of curvature of a dented pipe.

The ASME equations yield a sensible prediction of the bending component of the strain, which is reported by applying this technique. An analytical method based on the interpolation of the dented geometry of pipelines to gauge the strains in dented pipes is the latest evaluation by Okoloekwe et al. (Okoloekwe, C et al., 2018). By comparing the results of their method with the strains predicted by FEA, Okoloekwe et al. showed that there is a reasonable agreement between both. Whereas the FEA model for dent with 12% OD depth and 35 mm indenter diameter, in contrast, the model magnifies the value of the maximum equivalent plastic strain (PEEQ) by about 14% (Li, Y., Hassanien, S., Okoloekwe, C., & Adeeb, S., 2019).

The first objective for this thesis is to enhance the strain computational technique developed by Okoloekwe et al., for application by pipeline operators in order to increase the accuracy of the model. My preliminary analysis has shown that membrane strains in the longitudinal direction need to be accounted for in any model. Furthermore, by including the horizontal movement of the neutral axis in the Okoloekwe et al.'s model, accurate results that are comparable to the finite element analysis simulation results were obtained for all indentation depths. Continually in the second objective, we will suggest a new method on how to increase the accuracy of the original method for predicting the stain in the dented pipeline. For calibrating the upgraded models, we will use the profiles that we obtained from ILI tools.

Canada is a world leader with respect to dent evaluation techniques in pipelines integrity programs. Consequently, the result of this project will set Canada apart as a leader in design advancement and fundamental knowledge for training highly qualified engineers and researchers. What is planned in the first objective in regard to upgrading the accuracy of Okoloekwe et al. is an available mathematical method for strain analysis of the dented pipelines. In our introductory results, what is mentioned in the last part reveals that for upgrading the correctness of Okoloekwe et al.'s method, we should add the longitudinal displacement of the neutral axis. That will be found by the first objective, an analytical prediction for relative horizontal movement associated with membrane stain in the longitudinal direction of the dented pipeline.

To achieve the thesis objective, we will create a series of verified 3D and 2D nonlinear FE models in different situations, such as various indenter shapes and different dent depths. By using the developed method, for predicting the horizontal membrane displacement of the neutral axis of the dented pipelines, a general estimation equation will be developed and added to the strain computational method developed by Okoloekwe et al.

For the second objective, we will create a series of nonlinear FE models with numerous dent depths and shapes. We will try to predict the dent's length and determine the longitudinal strain according to the ASME equations based on the generated form. Subsequently, we will try to gain and compare results that we obtain from FE models with ASME equations for increasing the accuracy of analytical models.

Chapter 2 - The improvement of A strain-based modeling approach for analyzing dented pipeline severity in the longitudinal direction.

2.1 Abstract

Canada is ranked as the 3rd highest country of oil reserve with 9.8% of the world share of oil, and the 4th oil producer in 2019 with a significant increase in oil production in last few years (British Petroleum. https://www.bp.com/. 2019). Although pipelines are the easiest and cheapest method of transmitting oil and gas, their failure will cause a disaster. The denting of pipelines is one of the significant challenges faced by those in the oil and gas industry. Dents can cause a lower pressure capacity in the pipeline because of forming in the pipe wall. Analytical and numerical models can predict this issue, such as finite element analysis (FEA), which can analyze the pipelines based on the strain. The time problem with an FEA model can be solved with a quick and accurate evaluation of dents that urges the employment of an analytical approach. The non-mandatory equations presented in Appendix R of the ASME B31.8 2007 codes evaluate strains based on the minimum radius of the curvature of each dent in longitudinal and circumferential directions. The three-dimensional mathematical model which is presented by Okoloekwe et al. allows operators of pipelines to rapidly determine the severity of a dent by selecting the strain measurement that allows the model to remain consistent with its governing assumptions or use an assumed free formulation to account for the nonlinearity associated with the deformation. As it is known, in comparison to FEA, the model developed by Okoloekwe et al. predicts strain much more quickly than the FEA method. Consequently, this chapter seeks to improve the process by increasing the accuracy of the current model in the longitudinal direction.

Keywords:

pipe, FEA, dent, pressure, and strain

2.2 Introduction

More than 840,000 km of pipelines are in operation in Canada. Essentially, energy products from natural sources are transported across vast distances via these pipelines, which are by far the cheapest and most convenient mode of transport for those products, but also the most complicated. Additionally, there are over 100,000 workers employed by the oil and gas sector in Canada (https://thecanadianencyclopedia.ca/en/article/pipeline). Pipelines can be harmed by denting, cracking, and loss of metal or their combination as a result of these problems (Kiefner & Leewis, 2011). Dents with no corrosion, gouges, cracks, welds or other areas of increased stress are known as plain dents. According to the National Standard of Canada for oil and gas systems (CSA Z662:19), plain dents are deeper than 6 mm for pipe 101.6 mm outer diameter (OD) or smaller or deeper than 6% of OD. Plain dents are created by mechanical damage without changing the thickness of the wall. It is also possible that these dents can threaten or lead to the growth of the Potential corrosion or corrosion that already exists. As a result, these pipes will most likely crack in the areas where deformation has taken place. According to the CSA Z662:19 standard, the traditional way of judging the seriousness of the dents is based on their depth. Focusing on determining the severity of the dent based on the depth method can cause problems for both harmful and unnecessary dents, leading to unnecessary excavation for deep dents regarding repairing or neglecting moderate dents which can be dangerous for pipe safety because of their overall size and sharpness (Gao & McNealy, 2008). Strain-based models for dent assessment have shown that using the depth of the dent can be dangerous because of not caring about the sharpness of the dent profile which can lead to high local plastic strains. ASME B31.8-2018 estimates local strains as a function of the dent depth and length measured along the axis of the pipe. Based on ASME methods, recent research at the University of Alberta evaluated an alternative method for estimating the stress component of a pipe without the need to use finite elements (Okoloekwe, 2018). Using spline functions, Okoloekwe (2018) demonstrated how one can estimate the radius of curvature of a dented surface from which the localized strain can be calculated anywhere in a dented section of a pipeline. Okoloekwe (2018) demonstrated that their analysis approach was both accurate and conservative when compared to FEA (Woo, 2019). According to the last version of the American pipeline standard ASMEB31.8-2018 (Fig. 1), strains comprise two main components in longitudinal and circumferential directions for the pipe wall. Two separate bending and membrane strains exist for each direction.

Pipe wall strain has two main components: longitudinal and circumferential components. Each of them can be further divided into membrane strains and bending strains. From bottom to top, the membrane strain is the constant calculated by averaging the strain over the cross section, while the bending component is the linear fit of the strain after subtracting the membrane strain. A major challenge is determining membrane strains, as calculations must be made. In terms of bending components, the process is straightforward. As a result of the wall thickness of the pipe and the curvature of the dent, the maximum bending strain will be available on the pipe wall surface. From the measured dent shape, both axial and circumferential curvatures can be calculated (Lukasiewicz, Czyz, Sun, & Adeeb, 2006; Noronha et al., 2010).

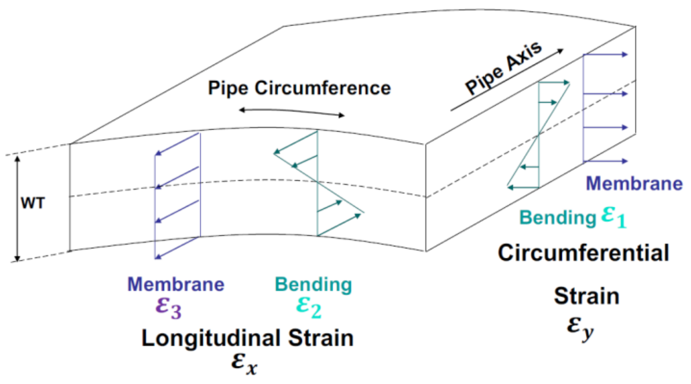


Fig. 1 Strain components in the pipe wall

In ASME B 31.8-2018, the following equations are assumed for circumferential bending (ε_1), longitudinal bending (ε_2), longitudinal membrane (ε_3) strains and the strain for inside and outside of the pipe surface. The ASME B 31.8-2018 ignores the membrane strain for the circumferential direction.

$$\varepsilon_1 = \left(\frac{t}{2}\right)\left(\frac{1}{R_0} - \frac{1}{R_1}\right) \qquad (1)$$

$$\varepsilon_2 = \frac{t}{(2R_2)} \qquad (2)$$

$$\varepsilon_3 = \left(\frac{t}{2}\right)\left(\frac{d}{L}\right)^2 \qquad (3)$$

R_0 is the initial pipe surface radius.

L indicates the length of the dent.

t represents the thickness of the pipe.

d denotes the dent depth.=

As displayed in Fig. 2, there are non-reentrant and reentrant positions for dents. For the non-reentrant dent, the surface of the dent is in the same direction as the surface of the pipe. Otherwise, the dent is assumed reentrant. R_1 is positive and negative for non-reentrant and reentrant dents, respectively (Fig. 2).

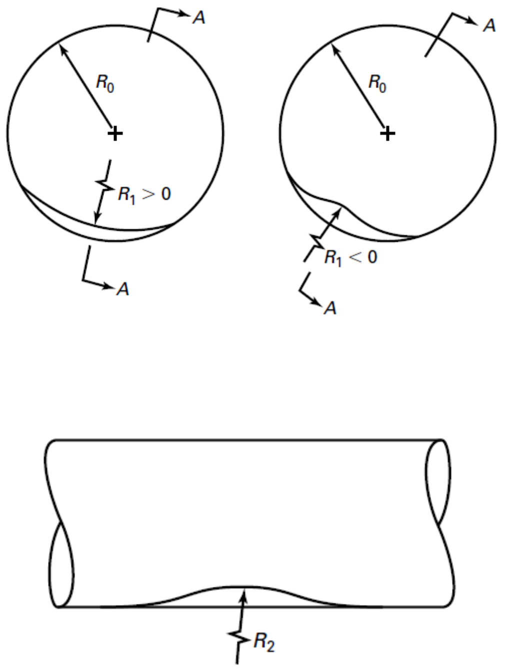


Fig. 2 Dent geometry

To calculate a dented section's equivalent total strain based on these strain components, combine them accordingly

$$\varepsilon_i = \sqrt{{\varepsilon_1}^2 - \varepsilon_1(\varepsilon_3 + \varepsilon_2) + (\varepsilon_3 + \varepsilon_2)^2}$$

$$\varepsilon_o = \sqrt{{\varepsilon_1}^2 + \varepsilon_1(\varepsilon_3 - \varepsilon_2) + (\varepsilon_3 - \varepsilon_2)^2}$$

The strain around the inner and outer surfaces of the pipe wall is equal to ε_i and ε_o. Positive and negative values should be considered for ε_1 and ε_2 regarding calculating the combined strain on the inside and outside pipe surface.

Noronha et al. (2005) estimated strain levels by using fourth-order B-spline curves for interpolating the dent contour and concluded that the results have various differences with those of the finite element method (FEM) for a small number of sensors. The obtained results using a high-resolution tool presented a high match with the FEM. Using a low-resolution caliper tool might result in large strain mispredictions if bending strains are predicted from the equations in the region closest to the dent apex. Moreover, the B31.8 standard does not provide a definition of length, which complicates the estimation of global longitudinal membrane strain, since longitudinal strain depends significantly on length.

In addition, Lukasiewicz et al. (2006) asserted that the B31.8 code equation for the longitudinal membrane strain is simply inaccurate, which calculates the component of the membrane strain in the longitudinal direction. The calculation of this method relies on neglecting the circumferential strain. It was also presented that the estimation of longitudinal strains using their method is highly simple and comes up with an alternative method for evaluating the strain by calculations based on radial displacement. In this method, 2 degrees of freedom results per node are compared with the large elastoplastic method in the FEM by the three-dimensional (3D) shell pipe model for 5 degrees of freedom for the node. The bending strain is calculated in both directions by the following equations.

$$\varepsilon_x^{-b} = \frac{t}{2}\frac{\partial^2 w}{\partial x^2} \tag{5}$$

$$\varepsilon_y^{-b} = \frac{t}{2}\frac{\partial^2 w}{\partial y^2} \tag{6}$$

where t denotes the pipe wall thickness.

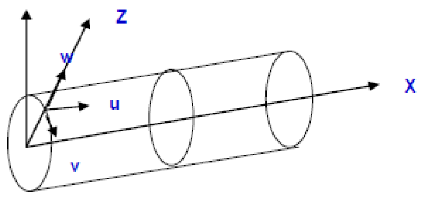


Fig. 3 Coordinate system and displacement

The equations for membrane strains are as follows:

$$\varepsilon_x^m = \frac{\partial u}{\partial x} + \frac{1}{2}\left(\frac{\partial w}{\partial x}\right)^2 + \varepsilon_x^0 \tag{7}$$

$$\varepsilon_y^m = \frac{\partial v}{\partial y} - \frac{w}{R} + \frac{1}{2}\left(\frac{\partial w}{\partial y}\right)^2 + \varepsilon_y^0 \tag{8}$$

where ε_x^m and ε_y^m are membrane strains in longitudinal and circumferential directions, respectively.

The shear strain is:

$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} + \left(\frac{\partial w}{\partial x}\right)\left(\frac{\partial w}{\partial y}\right) \tag{9}$$

The maximum values for strains in longitudinal and circumferential directions are:

$$\varepsilon_x = \varepsilon_x^m \pm \varepsilon_x^{-b} \tag{10}$$

$$\varepsilon_y = \varepsilon_y^m \pm \varepsilon_y^{-b} \tag{11}$$

where positive and negative signs are used for the outer and inner side of the pipe. The equivalent strain is a function of longitudinal and circumferential strains.

$$\varepsilon_{\mathrm{eq}} = \frac{2}{\sqrt{3}}\sqrt{(\varepsilon_x{}^2 + \varepsilon_{\mathrm{x}}\varepsilon_y + \varepsilon_y^2)} \qquad (12)$$

Noronha et al. (2005) Assert that by assuming circumferential and longitudinal directions as main directions, radial and circumferential strains are negligible compared to other strains. The equivalent strain is presented by:

$$\varepsilon_{eqv} = \frac{1}{1+\nu}\sqrt{\frac{1}{2}[(\varepsilon_I - \varepsilon_{II})^2 + (\varepsilon_{II} - \varepsilon_{III})^2 + (\varepsilon_{III} - \varepsilon_I)^2} \qquad (13)$$

By using $\nu = 0.5$, and ignoring the elastic strain as it is negligible compared to the plastic strain, the last equation can be rewritten as:

$$\varepsilon_{eqv} = \frac{\sqrt{2}}{3}\sqrt{[(\varepsilon_I - \varepsilon_{II})^2 + (\varepsilon_{II} - \varepsilon_{III})^2 + (\varepsilon_{III} - \varepsilon_I)^2} \qquad (14)$$

where ε_I represents the principal strain in the longitudinal direction.

ε_{II} is the principal strain in the circumferential direction.

ε_{III} indicates the principal strain in the radial direction.

By assuming the radial strain as a combination of longitudinal and circumferential strains the following equation is obtained:

$$\varepsilon_I + \varepsilon_{II} + \varepsilon_{III} = 0 \qquad (15)$$

By inserting that in Eq. (14), it will precisely resemble Eq. (15).

Woo et al. (2017) assured that a connection exists between the obtained results from ASME B31.8 and finite element analysis (FEA) models. In this method, FEA results are based on run inline inspection (ILI) tools. 22 models were created in their research. Using the ASME B31.8 equations is computationally robust. On the other hand, FEA results provided lower strains compared to ASME B31.8, especially for sharper dents. Likewise, Okoloekwe et al. (2018) evaluated a comprehensive strain-based model for determining the severity of dents in the pipe while ignoring the initial imperfection of the pipe, the stress concentrator, discontinuities, and internal pressure cycles. The present study aims at improving the non-destructive model presented by Okoloekwe et al. In order to overcome this problem, a simple method was developed to enhance the previous method in the longitudinal direction and obtain results that are very close to those obtained through using finite element software (ABAQUS).

2.3 Method

2.3.1 Modeling of Dents

By finite element analysis, we can solve complicated problems by fragmenting them into smaller units. The literature has published several studies using FEA to validate full-scale denting tests, compare results with analytical models, and develop new methods of assessing dent severity (Woo, 2019).

The intended pipeline model in this study was created using the commercial finite element software ABAQUS (version 2019). Instead of analyzing the entire pipe, the lagrangian strain distribution along the thickness of the wall will be studied numerically and analytically.

Fig. 4 and 5 illustrates a 2D and 3D model of the wall thickness generated along the longitudinal plane of the symmetry of a pipe.

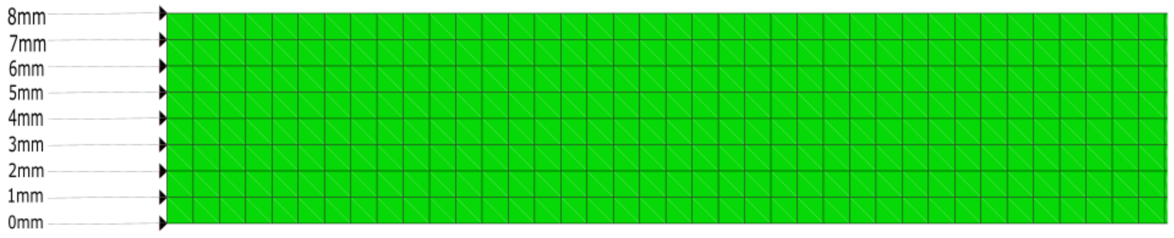


Fig. 4 2D model of wall thickness

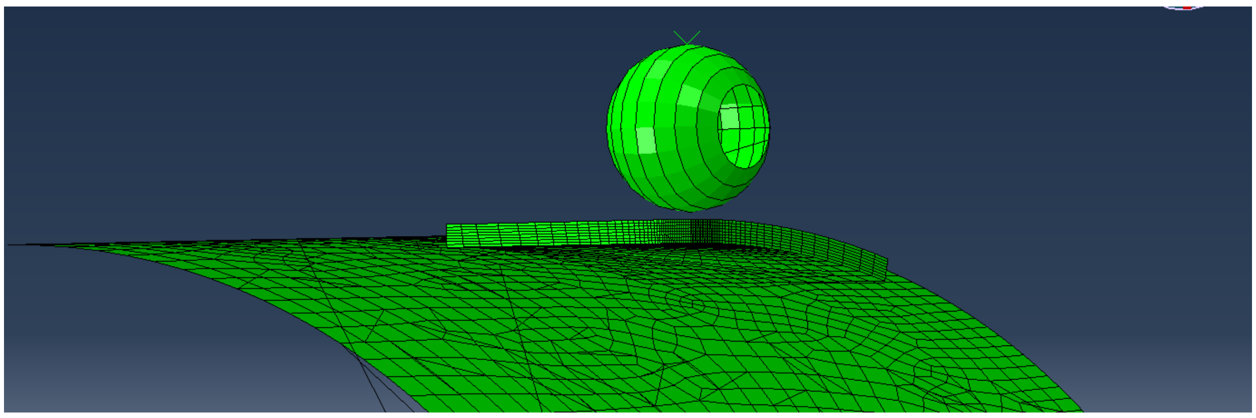


Fig. 5-a 3D model of wall thicknes

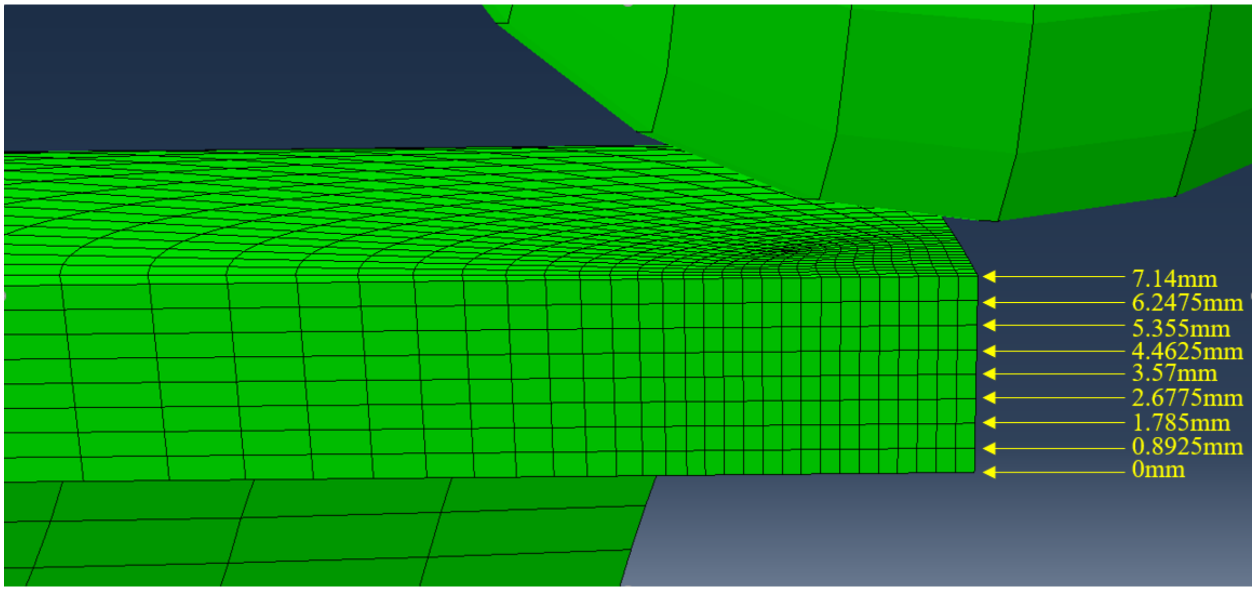


Fig. 5-b 3D model of wall thickness

Fig. 5 illustrates a 2D model of the wall thickness generated along the longitudinal plane of the symmetry of a pipe. The pipe had a length of 300 mm. The 300 mm length was utilized to ensure that the formed dent would be a localized deformity and no interaction would happen between the end boundary condition and the dent.

Fig. 6 illustrates a 3D model of the wall thickness generated along the longitudinal plane of the symmetry of a pipe. We examined 159 different models (the work took more that 600 hours to complete) in 7 categories, in the first category the pipe had a length of 600, 900, 1200, 1500, 1800, 2100, 2400, 2700, 3000, 3300, 3600 and 4000 mm with the OD of 762 mm. In addition, the wall thickness is 7.14 mm. The second category is the same as the first one. For the third category, the length of the pipe is 1250 mm with an OD of 762 mm and for each indentation depth, we have used these three-wall thicknesses for the pipeline, 6.8044, 7.14, and 7.4756 mm. For the fourth category, the length of the pipe is 2500 mm with an OD of 762 mm and for each indentation depth, we have used these three-wall thicknesses for the pipeline, 6.8044, 7.14, and 7.4756 mm. For the fifth category, the length of the pipe is 2500 mm with an OD of 762 mm and for each indentation depth we have used these three-wall thicknesses for the pipeline, 6.8044, 7.14, and 7.4756 mm with the different indentation depth compared to the previous category. For the sixths category, the length of the pipe is 1100 mm with an OD of 323.8 mm and for each indentation depth, we have used these three-wall thicknesses for the pipeline, 6.052, 6.35, and 6.649 mm. For the sixths category, the length of the pipe is 1100 mm with an OD of 323.8 mm and for each indentation depth we have used these three-wall thicknesses for the pipeline, 6.052, 6.35, and 6.649 mm with the different indentation depth compared to the previous category.

For the 2D model, a 2D analytical rigid shell was used to model the indenter. The indenter had a spherical shape and was 100 mm in diameter. Four bilinear nodes and plane stress elements with reduced integration and hourglass control were employed to mesh the studied model. The restrained translation was applied for both sides of the pipeline. In addition, a surface-to-surface (standard) interaction was considered for this model between the indenter and the pipe, and the degree of smoothing for the master surface was 0.2. Further, isotropic hardening plasticity was defined for the pipe material, a penalty was used for friction formulation instead of the Lagrange multiplier (standard). The pipe material was modeled as the elastic-plastic material with Young's modulus of 200 GPa and a poisons ratio of 0.3, and the pipe was an X-52 with a yield stress of 345 MPa. For steel materials, Poisson's ratio was approximately assumed to be 0.3.

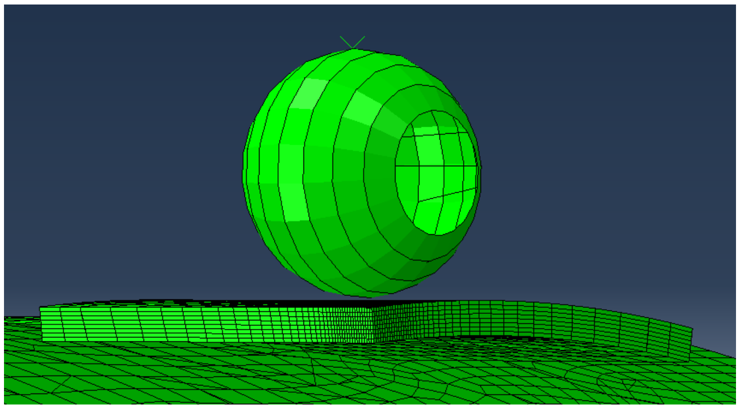
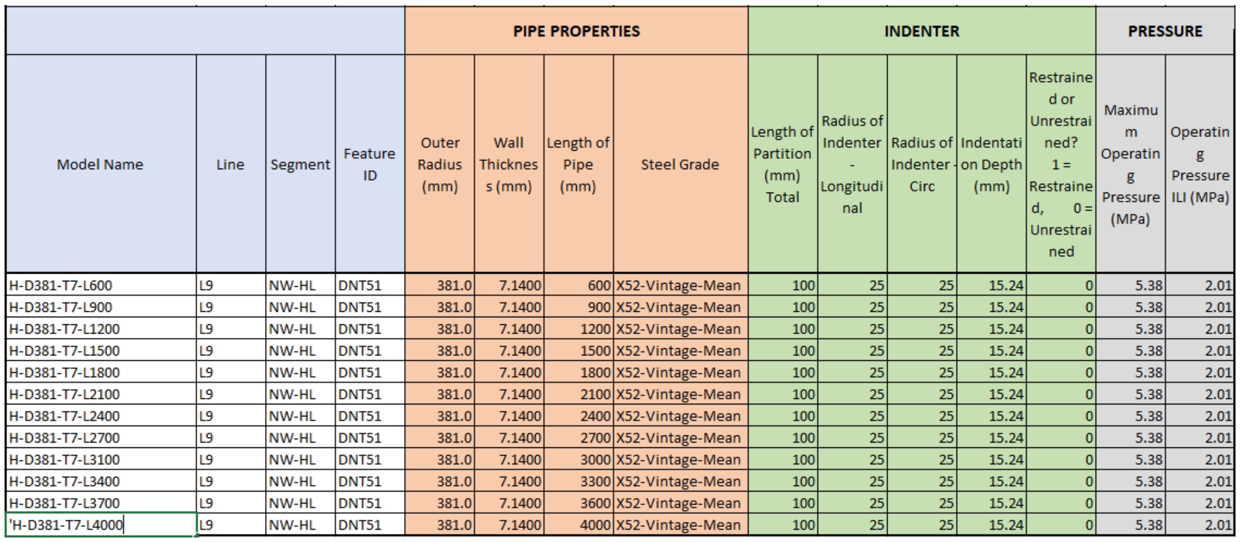


Fig. 6 3D model of indenter

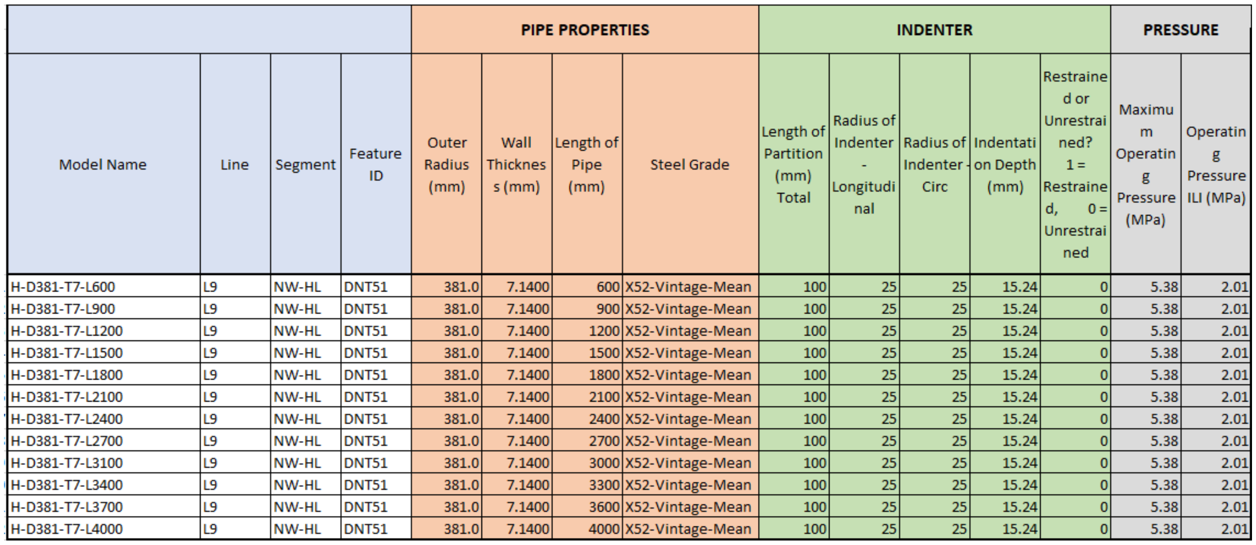
For the 3D model as, it is presented in Fig 6, the indenter shapes investigated were, a spherical indenter, the radius of the indenter in the longitudinal section, and the circumferential section for the first to the seventh category is 25mm and for the sixth and seventh category is 10mm. 15.24 is being conducted as a dent depth for the first and second categories of the 3D model. For the third category, three dent depths have been used which are 12.74, 15.24, and 17.74mm. A total of three dent depths were used for the fourth category, which were 5.12, 7.62, and 10.12mm. Three types of dent depth have been used in the fifth category, which are 27.98, 30.48, and 32.98mm. Dent depths of 2.35, 4.85, and 7.35mm were used for the sixth category. And finally, for the seventh category, three dent depths have been used which are 5.02, 7.52, and 10.02mm.

In the following tables the specifications of these models are presented.



				PIPE PROPERTIES				INDENTER					PRESSURE	
Model Name	Line	Segment	Feature ID	Outer Radius (mm)	Wall Thickness (mm)	Length of Pipe (mm)	Steel Grade	Length of Partition (mm) Total	Radius of Indenter - Longitudinal	Radius of Indenter - Circ	Indentation Depth (mm)	Restrained or Unrestrained? 1 = Restrained, 0 = Unrestrained	Maximum Operating Pressure (MPa)	Operating Pressure ILI (MPa)
H-D381-T7-L600	L9	NW-HL	DNT51	381.0	7.1400	600	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L900	L9	NW-HL	DNT51	381.0	7.1400	900	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L1200	L9	NW-HL	DNT51	381.0	7.1400	1200	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L1500	L9	NW-HL	DNT51	381.0	7.1400	1500	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L1800	L9	NW-HL	DNT51	381.0	7.1400	1800	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L2100	L9	NW-HL	DNT51	381.0	7.1400	2100	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L2400	L9	NW-HL	DNT51	381.0	7.1400	2400	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L2700	L9	NW-HL	DNT51	381.0	7.1400	2700	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L3100	L9	NW-HL	DNT51	381.0	7.1400	3000	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L3400	L9	NW-HL	DNT51	381.0	7.1400	3300	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L3700	L9	NW-HL	DNT51	381.0	7.1400	3600	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
'H-D381-T7-L4000	L9	NW-HL	DNT51	381.0	7.1400	4000	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01

Table 2. Specification of the models for the first category



				PIPE PROPERTIES				INDENTER					PRESSURE	
Model Name	Line	Segment	Feature ID	Outer Radius (mm)	Wall Thickness (mm)	Length of Pipe (mm)	Steel Grade	Length of Partition (mm) Total	Radius of Indenter - Longitudinal	Radius of Indenter - Circ	Indentation Depth (mm)	Restrained or Unrestrained? 1 = Restrained, 0 = Unrestrained	Maximum Operating Pressure (MPa)	Operating Pressure ILI (MPa)
H-D381-T7-L600	L9	NW-HL	DNT51	381.0	7.1400	600	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L900	L9	NW-HL	DNT51	381.0	7.1400	900	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L1200	L9	NW-HL	DNT51	381.0	7.1400	1200	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L1500	L9	NW-HL	DNT51	381.0	7.1400	1500	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L1800	L9	NW-HL	DNT51	381.0	7.1400	1800	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L2100	L9	NW-HL	DNT51	381.0	7.1400	2100	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L2400	L9	NW-HL	DNT51	381.0	7.1400	2400	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L2700	L9	NW-HL	DNT51	381.0	7.1400	2700	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L3100	L9	NW-HL	DNT51	381.0	7.1400	3000	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L3400	L9	NW-HL	DNT51	381.0	7.1400	3300	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L3700	L9	NW-HL	DNT51	381.0	7.1400	3600	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-D381-T7-L4000	L9	NW-HL	DNT51	381.0	7.1400	4000	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01

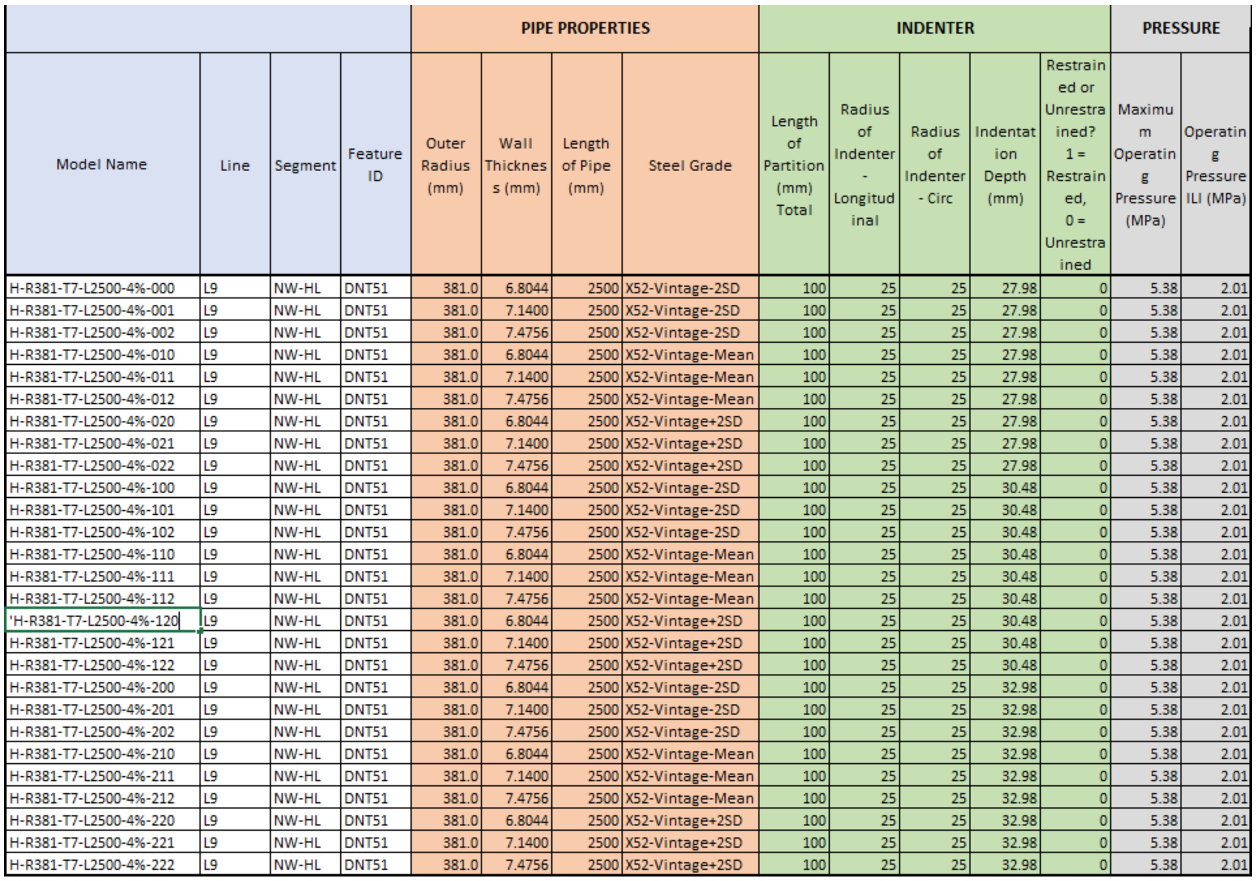
Table 3. Specification of the models for the second category

				PIPE PROPERTIES				INDENTER					PRESSURE	
Model Name	Line	Segment	Feature ID	Outer Radius (mm)	Wall Thickness (mm)	Length of Pipe (mm)	Steel Grade	Length of Partition (mm) Total	Radius of Indenter - Longitudinal	Radius of Indenter - Circ	Indentation Depth (mm)	Restrained or Unrestrained? 1 = Restrained, 0 = Unrestrained	Maximum Operating Pressure (MPa)	Operating Pressure ILI (MPa)
H-R381-T7-L1250-000	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage-2SD	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-001	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage-2SD	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-002	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage-2SD	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-010	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage-Mean	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-011	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage-Mean	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-012	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage-Mean	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-020	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage+2SD	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-021	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage+2SD	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-022	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage+2SD	100	25	25	12.74	0	5.38	2.01
H-R381-T7-L1250-100	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage-2SD	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-101	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage-2SD	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-102	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage-2SD	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-110	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-111	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-112	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage-Mean	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-120	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage+2SD	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-121	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage+2SD	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-122	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage+2SD	100	25	25	15.24	0	5.38	2.01
H-R381-T7-L1250-200	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage-2SD	100	25	25	17.74	0	5.38	2.01
H-R381-T7-L1250-201	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage-2SD	100	25	25	17.74	0	5.38	2.01
H-R381-T7-L1250-202	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage-2SD	100	25	25	17.74	0	5.38	2.01
H-R381-T7-L1250-210	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage-Mean	100	25	25	17.74	0	5.38	2.01
H-R381-T7-L1250-211	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage-Mean	100	25	25	17.74	0	5.38	2.01
H-R381-T7-L1250-212	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage-Mean	100	25	25	17.74	0	5.38	2.01
H-R381-T7-L1250-220	L9	NW-HL	DNT51	381.0	6.8044	1250	X52-Vintage+2SD	100	25	25	17.74	0	5.38	2.01
H-R381-T7-L1250-221	L9	NW-HL	DNT51	381.0	7.1400	1250	X52-Vintage+2SD	100	25	25	17.74	0	5.38	2.01
H-R381-T7-L1250-222	L9	NW-HL	DNT51	381.0	7.4756	1250	X52-Vintage+2SD	100	25	25	17.74	0	5.38	2.01

Table 4. Specification of the models for the third category

				PIPE PROPERTIES				INDENTER					PRESSURE	
Model Name	Line	Segment	Feature ID	Outer Radius (mm)	Wall Thickness (mm)	Length of Pipe (mm)	Steel Grade	Length of Partition (mm) Total	Radius of Indenter - Longitudinal	Radius of Indenter - Circ	Indentation Depth (mm)	Restrained or Unrestrained? 1 = Restrained, 0 = Unrestrained	Maximum Operating Pressure (MPa)	Operating Pressure ILI (MPa)
H-R381-T7-L2500-1%-000	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-001	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-002	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-010	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-011	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-012	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-020	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-021	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-022	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-100	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-101	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-102	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-110	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-111	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-112	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-120	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-121	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-122	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-200	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-201	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-202	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-210	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-211	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-212	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-220	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-221	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-222	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	10.12	0	5.38	2.01

Table 5. Specification of the models for the fourth category



				PIPE PROPERTIES				INDENTER					PRESSURE	
Model Name	Line	Segment	Feature ID	Outer Radius (mm)	Wall Thickness (mm)	Length of Pipe (mm)	Steel Grade	Length of Partition (mm) Total	Radius of Indenter - Longitudinal	Radius of Indenter - Circ	Indentation Depth (mm)	Restrained or Unrestrained? 1 = Restrained, 0 = Unrestrained	Maximum Operating Pressure (MPa)	Operating Pressure ILI (MPa)
H-R381-T7-L2500-4%-000	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-001	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-002	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-010	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-011	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-012	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-020	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-021	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-022	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	27.98	0	5.38	2.01
H-R381-T7-L2500-4%-100	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	30.48	0	5.38	2.01
H-R381-T7-L2500-4%-101	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	30.48	0	5.38	2.01
H-R381-T7-L2500-4%-102	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	30.48	0	5.38	2.01
H-R381-T7-L2500-4%-110	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	30.48	0	5.38	2.01
H-R381-T7-L2500-4%-111	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	30.48	0	5.38	2.01
H-R381-T7-L2500-4%-112	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	30.48	0	5.38	2.01
'H-R381-T7-L2500-4%-120	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	30.48	0	5.38	2.01
H-R381-T7-L2500-4%-121	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	30.48	0	5.38	2.01
H-R381-T7-L2500-4%-122	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	30.48	0	5.38	2.01
H-R381-T7-L2500-4%-200	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	32.98	0	5.38	2.01
H-R381-T7-L2500-4%-201	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	32.98	0	5.38	2.01
H-R381-T7-L2500-4%-202	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	32.98	0	5.38	2.01
H-R381-T7-L2500-4%-210	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	32.98	0	5.38	2.01
H-R381-T7-L2500-4%-211	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	32.98	0	5.38	2.01
H-R381-T7-L2500-4%-212	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	32.98	0	5.38	2.01
H-R381-T7-L2500-4%-220	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	32.98	0	5.38	2.01
H-R381-T7-L2500-4%-221	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	32.98	0	5.38	2.01
H-R381-T7-L2500-4%-222	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	32.98	0	5.38	2.01

Table 6. Specification of the models for the fifth category

				PIPE PROPERTIES				INDENTER					PRESSURE	
Model Name	Line	Segment	Feature ID	Outer Radius (mm)	Wall Thickness (mm)	Length of Pipe (mm)	Steel Grade	Length of Partition (mm) Total	Radius of Indenter - Longitudinal	Radius of Indenter - Circ	Indentation Depth (mm)	Restrained or Unrestrained? 1 = Restrained, 0 = Unrestrained	Maximum Operating Pressure (MPa)	Operating Pressure ILI (MPa)
H-D323-T635-L1100-EPS20p-IND10-DD485-000	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-2SD	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-001	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-2SD	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-002	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-2SD	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-010	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-Mean	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-011	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-Mean	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-012	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-Mean	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-020	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage+2SD	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-021	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage+2SD	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-022	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage+2SD	80	10	10	2.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-100	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-2SD	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-101	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-2SD	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-102	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-2SD	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-110	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-Mean	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-111	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-Mean	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-112	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-Mean	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-120	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage+2SD	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-121	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage+2SD	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-122	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage+2SD	80	10	10	4.85	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-200	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-2SD	80	10	10	7.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-201	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-2SD	80	10	10	7.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-202	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-2SD	80	10	10	7.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-210	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-Mean	80	10	10	7.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-211	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-Mean	80	10	10	7.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-212	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-Mean	80	10	10	7.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-220	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage+2SD	80	10	10	7.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-221	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage+2SD	80	10	10	7.35	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-222	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage+2SD	80	10	10	7.35	0	11.25	4.22

Table 7. Specification of the models for the sixths category

				PIPE PROPERTIES				INDENTER					PRESSURE	
Model Name	Line	Segment	Feature ID	Outer Radius (mm)	Wall Thickness (mm)	Length of Pipe (mm)	Steel Grade	Length of Partition (mm) Total	Radius of Indenter - Longitudinal	Radius of Indenter - Circ	Indentation Depth (mm)	Restrained or Unrestrained? 1 = Restrained, 0 = Unrestrained	Maximum Operating Pressure (MPa)	Operating Pressure ILI (MPa)
H-D323-T635-L1100-EPS20p-IND10-DD485-000	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382-2SD	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-001	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382-2SD	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-002	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382-2SD	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-010	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382-Mean	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-011	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382-Mean	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-012	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382-Mean	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-020	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382+2SD	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-021	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382+2SD	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-022	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382+2SD	80	10	10	5.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-100	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382-2SD	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-101	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382-2SD	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-102	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382-2SD	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-110	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382-Mean	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-111	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382-Mean	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-112	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382-Mean	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-120	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382+2SD	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-121	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382+2SD	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-122	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382+2SD	80	10	10	7.52	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-200	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382-2SD	80	10	10	10.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-201	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382-2SD	80	10	10	10.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-202	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382-2SD	80	10	10	10.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-210	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382-Mean	80	10	10	10.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-211	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382-Mean	80	10	10	10.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-212	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382-Mean	80	10	10	10.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-220	L9	NW-HL	DNT51	161.9	6.0520	1100	X52-Vintage-382+2SD	80	10	10	10.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-221	L9	NW-HL	DNT51	161.9	6.3500	1100	X52-Vintage-382+2SD	80	10	10	10.02	0	11.25	4.22
H-D323-T635-L1100-EPS20p-IND10-DD485-222	L9	NW-HL	DNT51	161.9	6.6490	1100	X52-Vintage-382+2SD	80	10	10	10.02	0	11.25	4.22

Table 8. Specification of the models for the seventh category

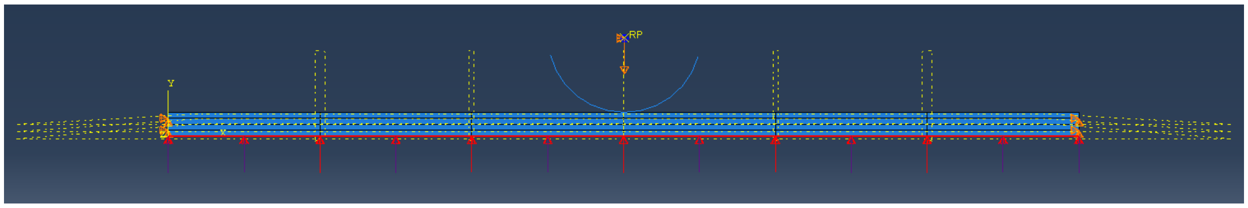


Fig. 7 internal pressure direction

For the 2D model, the study was performed on unpressurized and pressurized pipes with restrained dents. For the pressurized condition, the internal pressure effect was applied as an upward distributed line load with an intensity rate of 8 MPa along the lower edge of the model (Fig. 7). Overall, 10 parameters were evaluated, including 10 different dent depths (i.e., 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, and 5% OD) for 9 various levels along the thickness of the pipe (i.e., 0, 1, 2, 3, 4, 5, 6, 7, and 8 mm).

In the 3D model for the first to the fifth model, the maximum operating pressure is 5.38 MPa, and operating pressure ILI is 2.01MPa and for the sixth and seventh categories are 11.25 and 4.22 MPa.

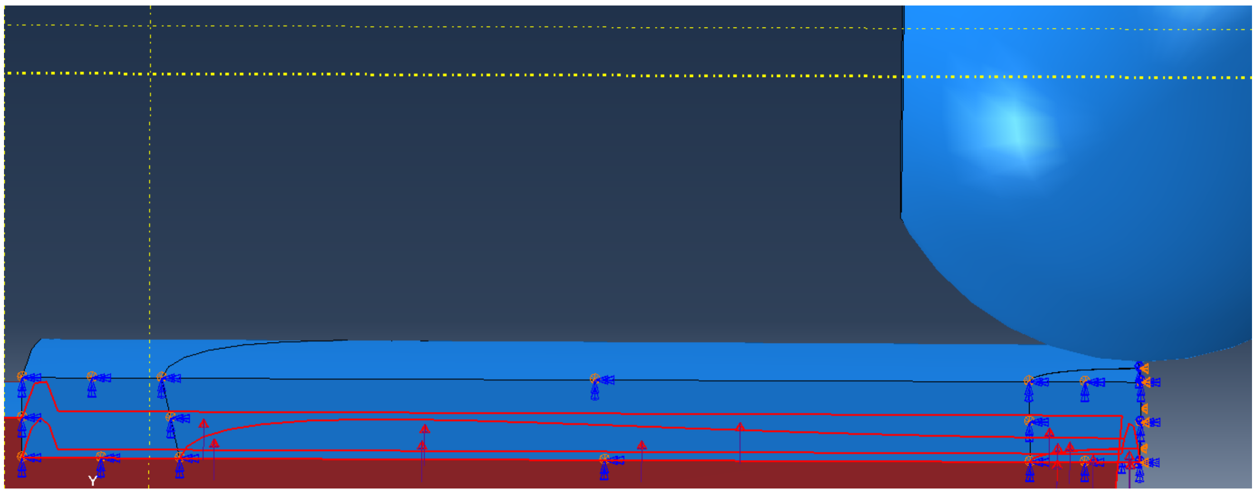
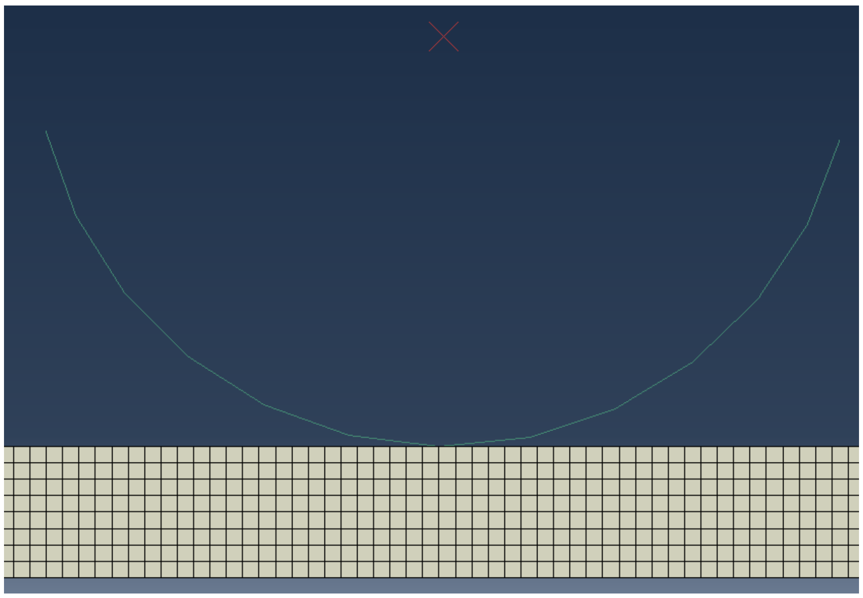


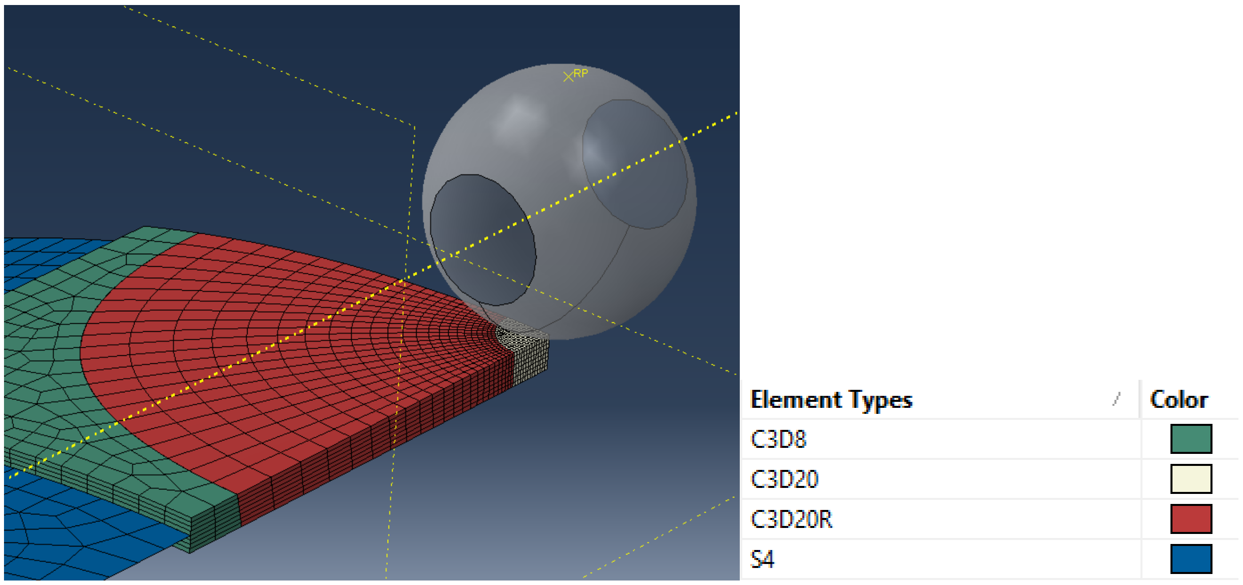
Fig. 8 maximum operating pressure direction

The assumption used in 2D analysis was plane stress and the element types in 2D and 3D analyses are presented in the following figures.



Element Types	Edge	Color
ARSSE	✱	🟩
CPS4R	✱	⬜
RNODE2D	✱	🟥

Fig. 9-a Element types in 2D direction



Element Types	Color
C3D8	🟩
C3D20	⬜
C3D20R	🟥
S4	🟦

Fig. 9-b Element types in 3D direction

Where ARSSE is Analytic rigid surface (extruded), CPS4R is 4-node bilinear, reduced integration with hourglass control, RNODE2D is Reference node (two-dimensional). Also, the elements in the 3D direction are C3D8 which is a general-purpose linear brick element, fully integrated (2x2x2 integration points), C3D20 is a general-purpose quadratic brick element (3x3x3 integration points), C3D20R element is a general-purpose quadratic brick element, with reduced integration (2x2x2 integration points) and S4 is a fully integrated, general-purpose, finite-membrane-strain shell element

2.3.2 Dent Profile Interpolation

The data points representing the dented pipe were extracted from numerical models. The profiles in Fig. 10 depict displacements along the longitudinal direction.

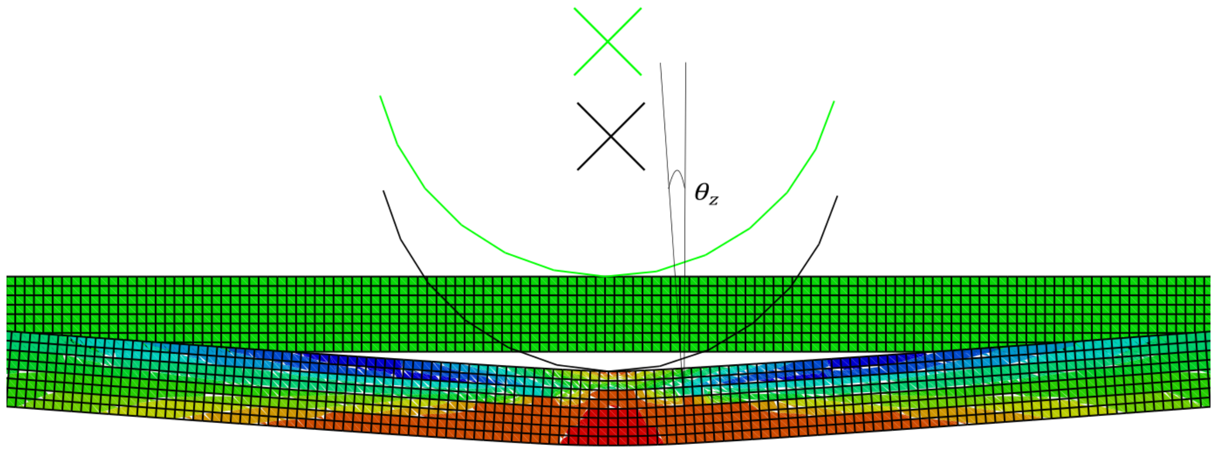


Fig. 10 displacement along the longitudinal direction

2.3.3 Loading Step

Pressurization, denting, and removal of the indenter were applied in a sequence designed to simulate an unconstrained dent that forms during operation. Different pressure values were considered for the experiment's pressurization phase to simulate the pressurized fluid moving in a pipeline at real-life operating pressures.

FEA followed with an indentation step, which involved translating the indenter downwards to create different depths of indentation. As a final step, the indenter was shifted vertically upward so that it was no longer in contact with the pipe. The loading sequence is depicted in Figure 11.

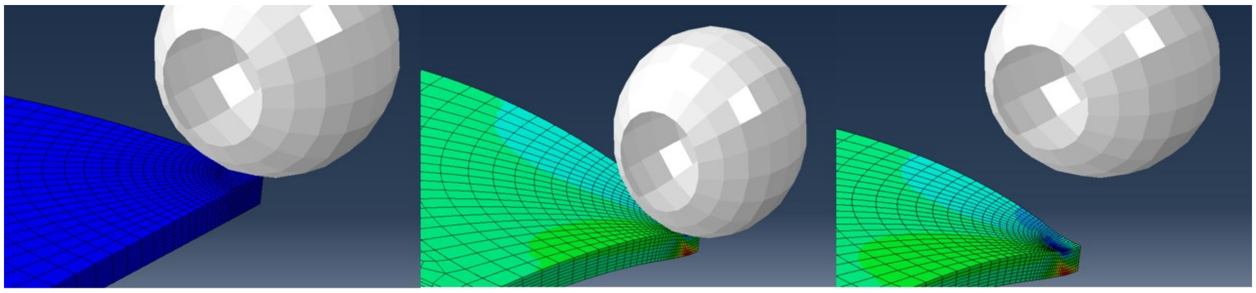


Fig. 11 Loading steps

2.3.4 Displacement Discretization

In order to construct a 3D model of the morphology of the pipeline, we should make use of a cylindrical coordinate system, which allows the determination of the deformed pipeline coordinates, as explained in (Okoloekwe, et al., 2019).

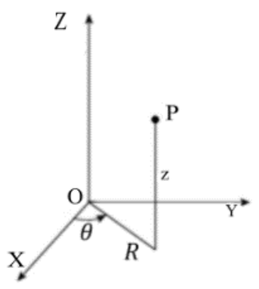


Fig. 12 cylindrical coordinate System

In a cylinder coordinate system, the global displacement field can be expressed by Equation (16):

$$u = u_r e_r + u_\theta e_\theta + u_z e_z \qquad (16)$$

with components in the radial, circumferential, and longitudinal local directions.

The gradient of the displacement vector can be computed taking into consideration that the basis vectors e_r and e_θ are dependent upon the angle and e_z is not dependent upon the angle. Mathematically, the displacement gradient is represented by equation (17).

$$\nabla u = \begin{bmatrix} \frac{\partial u_r}{\partial R} & \frac{\partial u_r}{R\,\partial\theta} - \frac{u_\theta}{R} & \frac{\partial u_r}{\partial z} \\ \frac{\partial u_\theta}{\partial R} & \frac{u_r}{R} + \frac{\partial u_\theta}{R\,\partial\theta} & \frac{\partial u_\theta}{\partial z} \\ \frac{\partial u_z}{\partial R} & \frac{\partial U_2}{R\,\partial\theta} & \frac{\partial u_z}{\partial z} \end{bmatrix} \qquad (17)$$

A deformed pipeline's mid surface is defined by its radius R.

There is a great deal of difficulty in deriving a theoretical solution for local deformations of a pipeline without simplifying assumptions associated with geometrical and material nonlinearities. In the original method, the the pipe wall's mid surface is assumed to be straight and uniform before deformation. To gain better strain results we will add mid-surface displacement to the results. The hypothetical radius of the mid surface of the deformed pipe is evaluated by:

$$R_{hyp}(z) = \int_{-\pi}^{\pi} \frac{R_m(\theta,z)\,\partial\theta}{2\pi} \qquad (18)$$

where R_m is the radius of the mid surface of the deformed pipeline and ϕ is the angular distortion of the deformed pipeline.

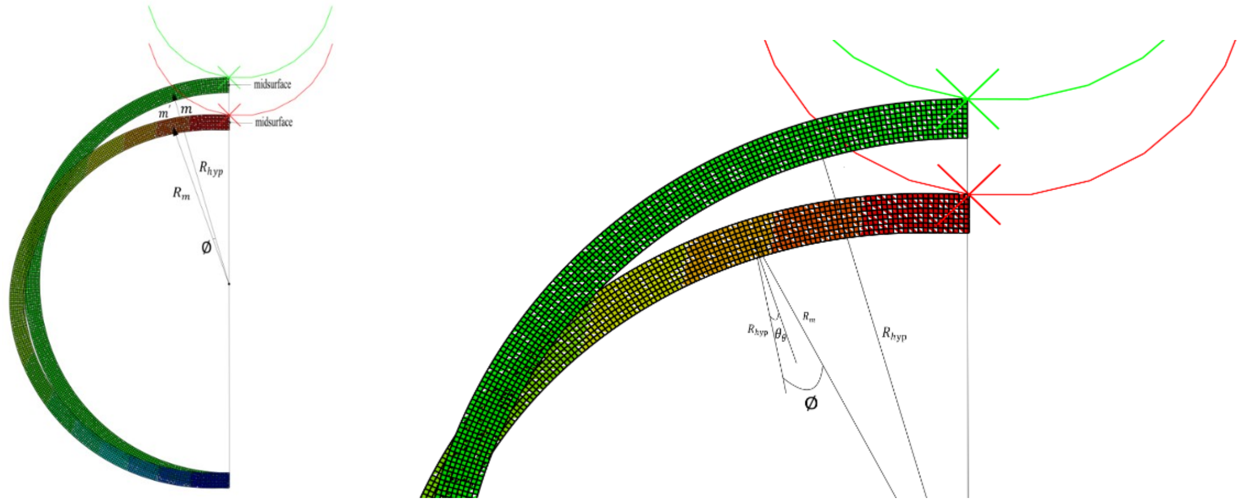


Fig. 13 displacement along the circumferential direction

Based on the assumption that displacements along the thickness of the pipe wall are linearly distributed, the longitudinal deformations associated with the indentation are evaluated. Therefore, longitudinal displacement is simulated with large displacements and rotations as a function of longitudinal slope, θ_z, of the pipe wall as shown in equation (19).

The longitudinal displacement is given by:

$$u_z = t_v \mathrm{Sin}(\theta_z) \qquad (19)$$

Where u_z is the longitudinal displacement and θ_z is the slope of the deformed pipe wall in the longitudinal direction

The circumferential displacement is given by:

$$u_\theta = R_m \mathrm{Sin}(\emptyset) - t_v \mathrm{Sin}(\emptyset - \theta_\theta) \qquad (20)$$

Where u_θ is the circumferential displacement,θ_θ is the slope of the deformed pipe wall in the circumferential direction and $\emptyset$ is the angular distortion of the deformed pipe.

The radial displacement:

$$u_r = R_m(\theta, z)\mathrm{Cos}(\emptyset) - R_{hyp}(z) \qquad (21)$$

Where u_r is the radial displacement.

In Okoloekwe's original method, it was assumed that the horizontal displacement in the middle layer, as shown in the figure below, was zero, and that each of the nodes shifted vertically when the dent was created. To find the horizontal displacement of the middle layer, first we found its primary location and then their secondary location, then we found the displacement of each part and by collecting the displacement of all parts, the displacement of the middle layer was obtained.

As will be shown in the results, adding the displacement of the mid surface of the pipe significantly improves the accuracy of prediction, therefore, Eq. (22) is modified as follows:

$$u_z = t_v \mathrm{Sin}(\theta_z) + um \quad (22)$$

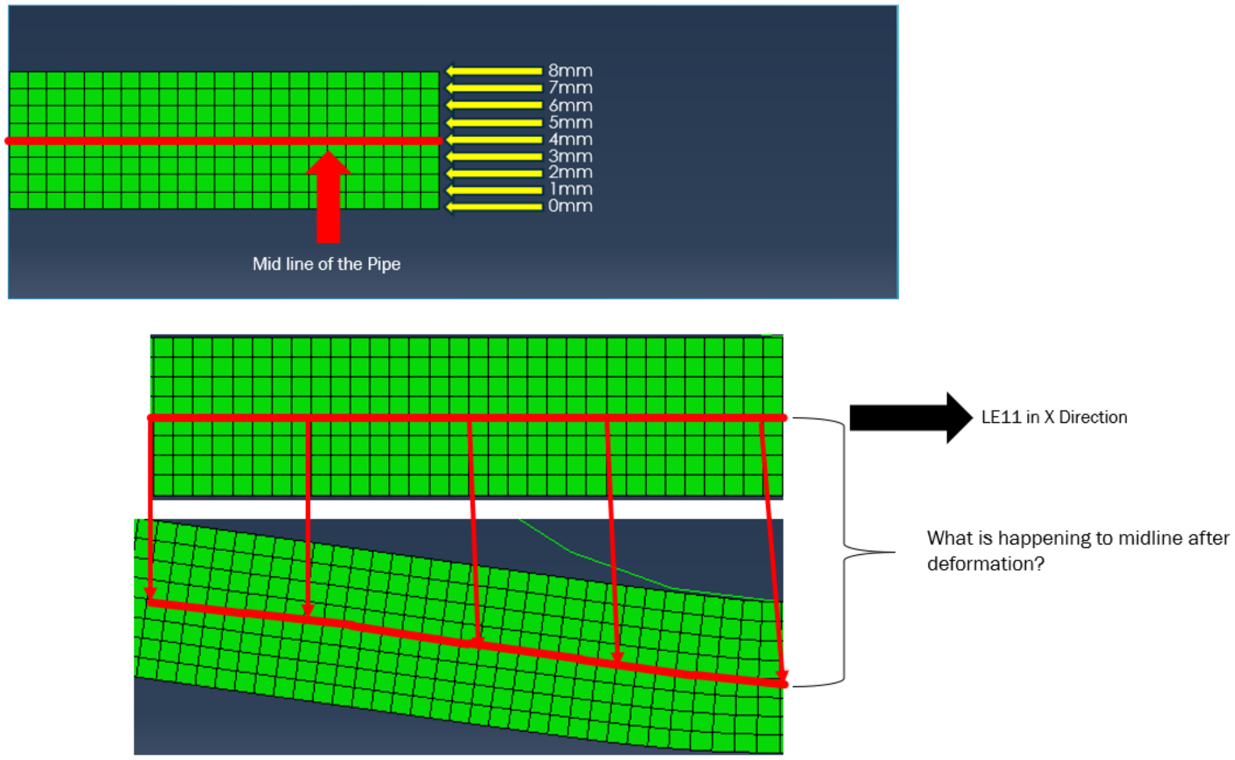


Fig. 14 displacement in the mid-line of the pipe in the longitudinal direction

where u_m and t_v indicate the displacement of the mid surface in the deformed pipe and the coordinate normal to the mid surface of the pipe, respectively. Additionally, t and t_v are the thickness of the pipe and along with that $(\frac{-t}{2} < t\_v < \frac{t}{2})$, respectively.

The slopes along the circumference and the longitudinal axis across the circumference are calculated using Eqs. (23) and (24), respectively

$$\theta_\theta = \mathrm{ArcTan}\left(\frac{\partial R_m}{R_m\,\partial\theta}\right) \quad (23)$$

$$\theta_z = \mathrm{ArcTan}\left(\frac{\partial u_r}{\partial z}\right) \quad (24)$$

For strain measurement in the circumferential direction, it is assumed that the linear strain or the small strain is calculated by using Eq. (25):

$$\varepsilon_L = \frac{1}{2}(\nabla u + \nabla u^T) \tag{25}$$

For large deformations and rotations that result from strain, the Lagrangian strain measure includes nonlinear terms. In Lagrangian strain analysis, the expression represents by

$$\varepsilon_{NL} = \frac{1}{2}(\nabla u + \nabla u^T + \nabla u \nabla u^T) \tag{26}$$

Using the results obtained from the FEA, the equation ArcTan $\left(\frac{\partial u_r}{\partial z}\right)$ in Mathematica was used to calculate the θ_z in the original method and the displacement of the nodes in the horizontal direction was obtained. By obtaining U1, the amount of Lagrangian strain in different layers of pipes was computer. And their diagrams were drawn, an example of which will be displayed in the result section.

Also, Mathematica software was used for interpolating the graphs. Moreover, Gaussian Filter was applied to remove the noise effect (i.e., flatting and decreasing noisy points), especially at 4 and 5 mm height of the pipe.

2.4 Results:

An example of a numerical model for the deformed pipes and the indenter is illustrated in figure 15.

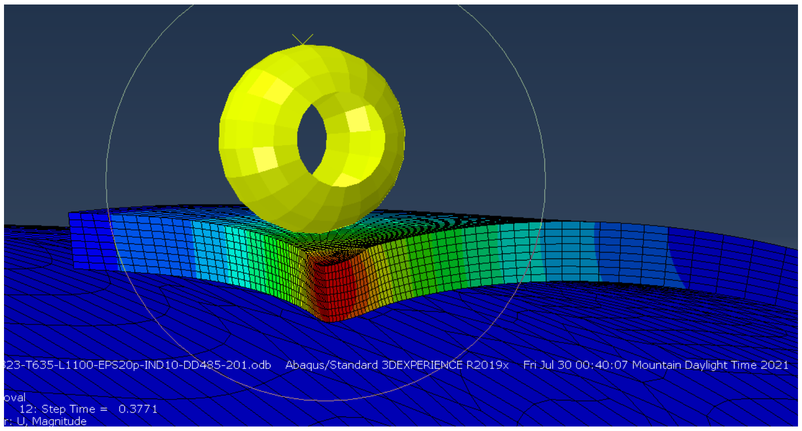

23-T635-L1100-EPS20p-IND10-DD485-201.odb Abaqus/Standard 3DEXPERIENCE R2019x Fri Jul 30 00:40:07 Mountain Daylight Time 2021
oval
12: Step Time = 0.3771
r: U, Magnitude

Fig. 15 Numerical Models

2.4.1 Numerical Models

Deformation Analysis:

Fig. 15 to 23 show the graphs representing the longitudinal displacement (U1) in 9 different Levels (0 to 8 mm) of the pipe from inside to outside of the 2D model subjected to a 10mm dent. It is observed that without adding the mid surface displacement to the original expression there are considerable differences between the “true” displacement, which is calculated by the FEA model, and the original method. The horizontal axis in the graphs represents the longitudinal length of the pipe and the vertical axis represents the U1 in the longitudinal direction for the pipe. The graphs shows that the U1 displacement is concentrated between the apex of the dent and supports. As can be seen in Fig. 16-24 in the different layers of the pipe, after adding U1 displacement in the mid surface which is used in Eq.22, the prediction of the two methods (modified and FEA methods) became very close.

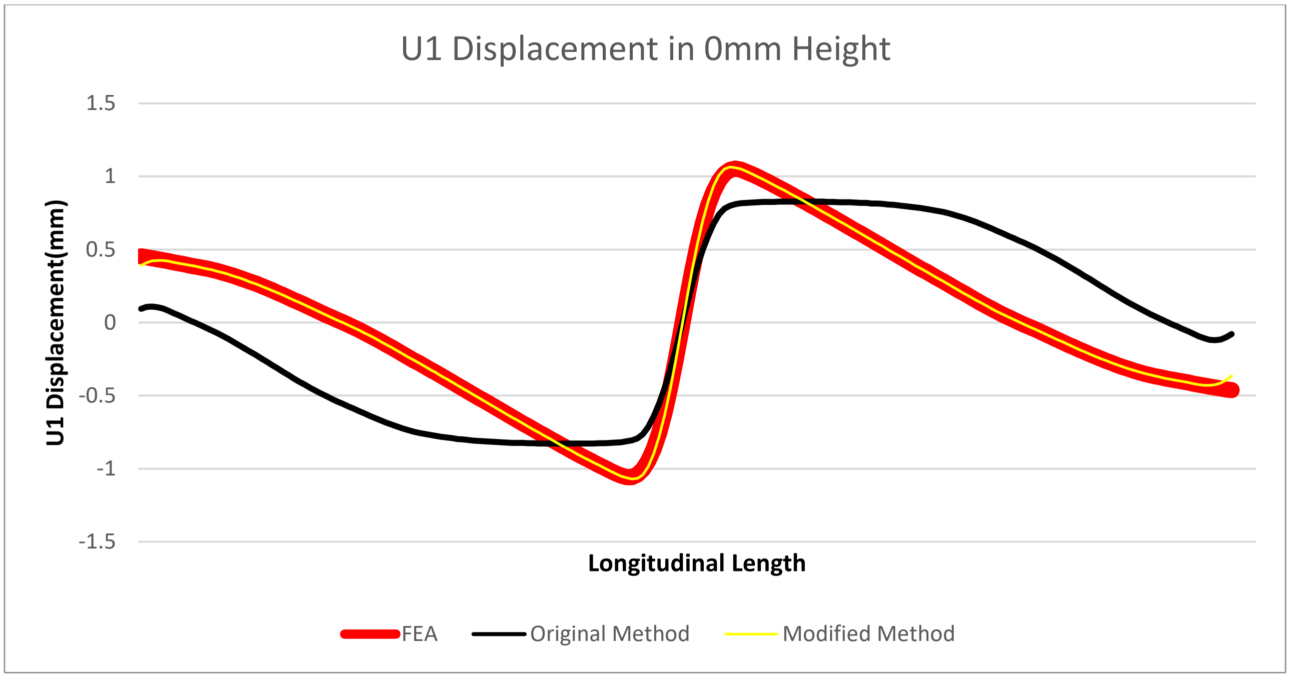


Fig. 16 U1 Displacement in the longitudinal direction in 0mm Height

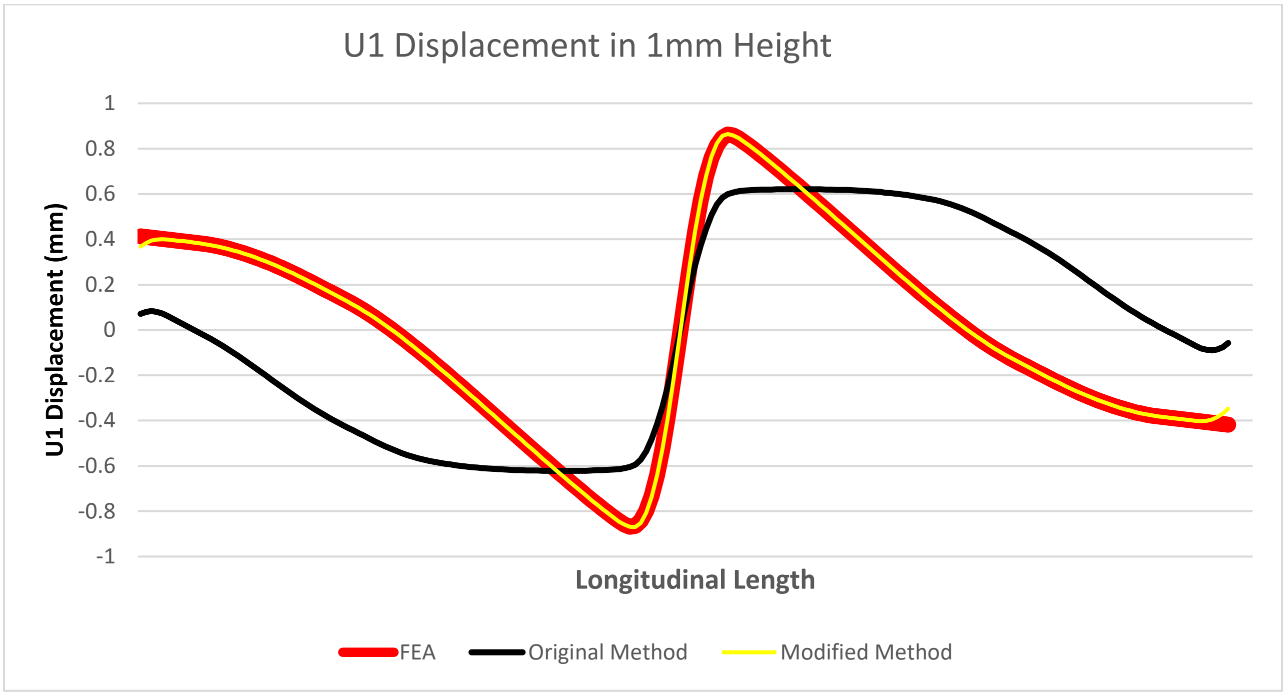


Fig. 17 U1 Displacement in the longitudinal direction in 1mm Height

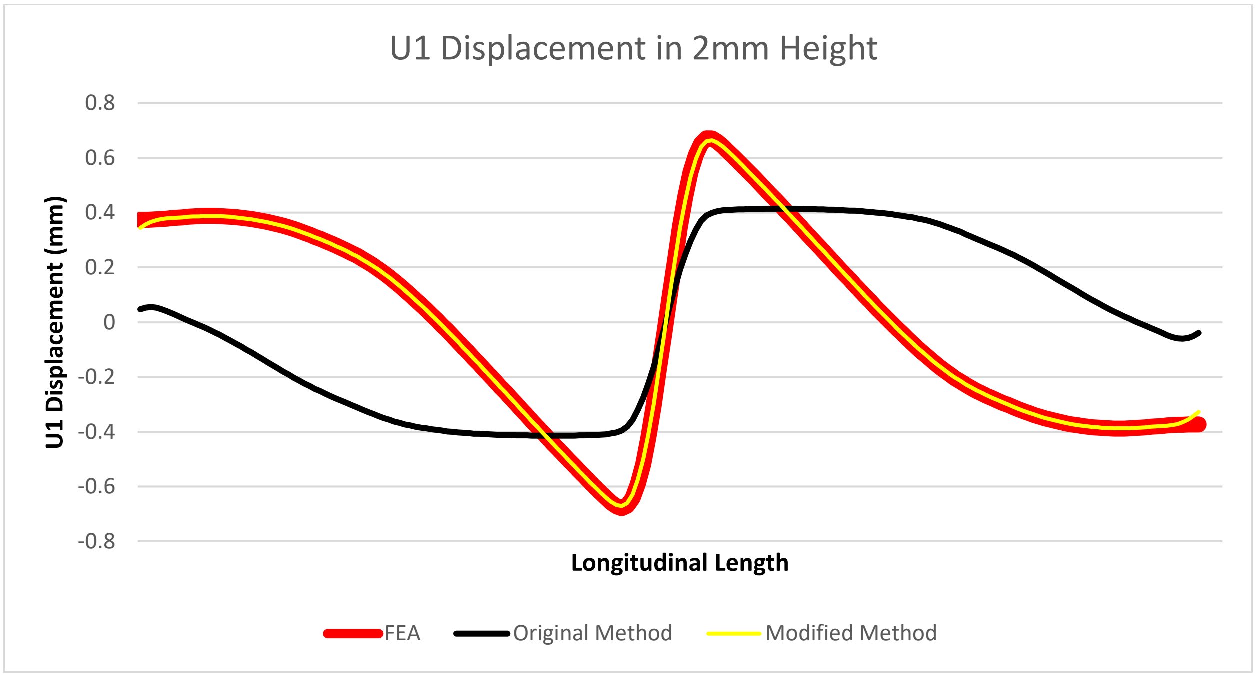


Fig. 18 U1 Displacement in the longitudinal direction in 2mm Height

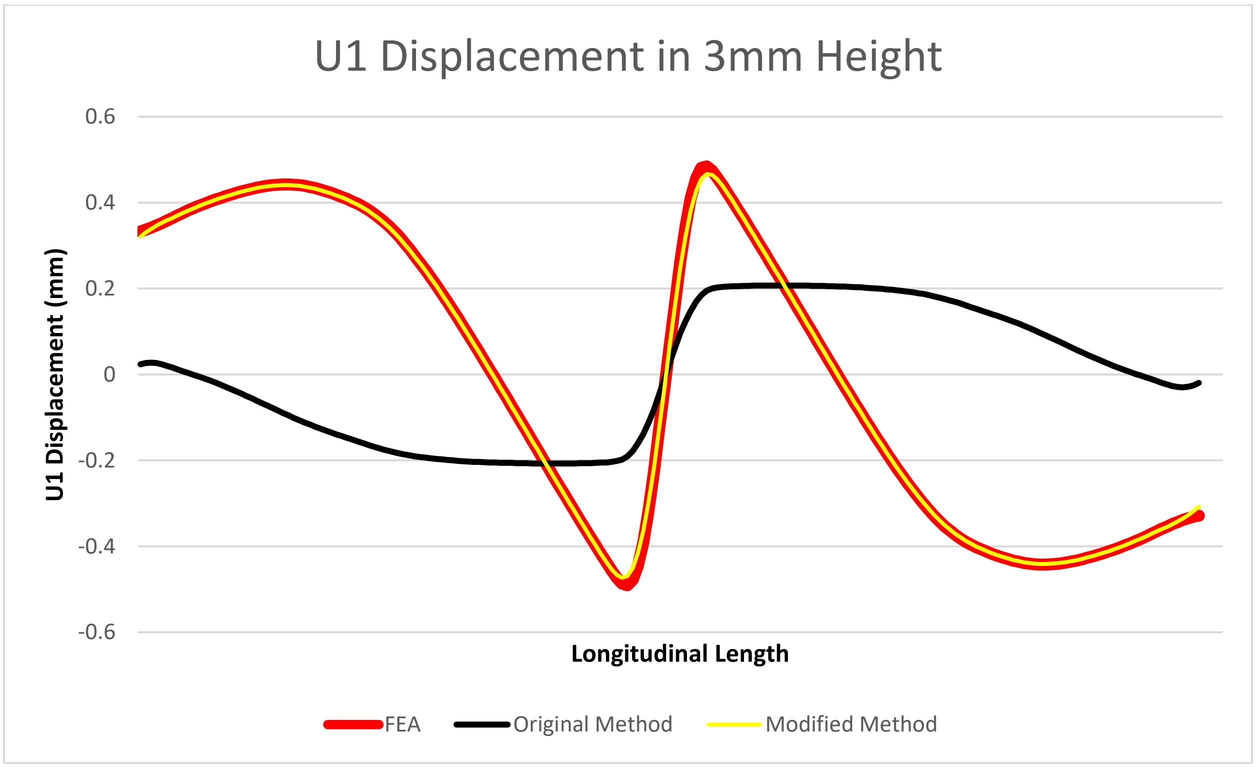


Fig. 19 U1 Displacement in the longitudinal direction in 3mm Height

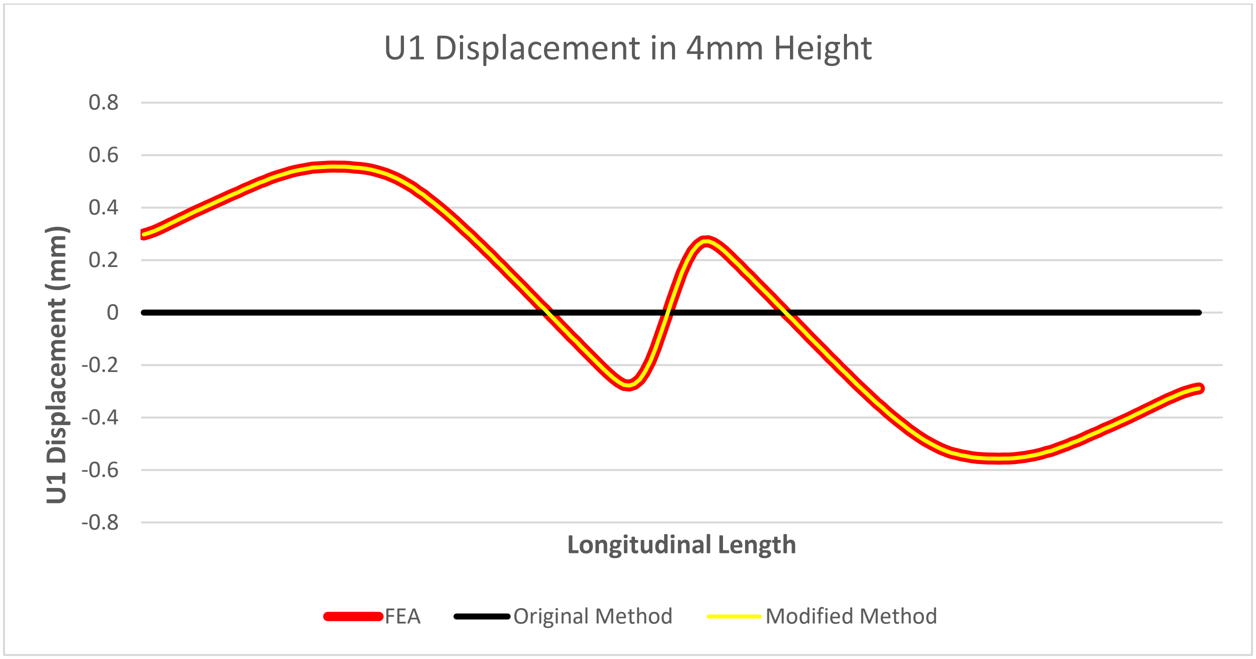


Fig. 20 U1 Displacement in the longitudinal direction in 4mm Height

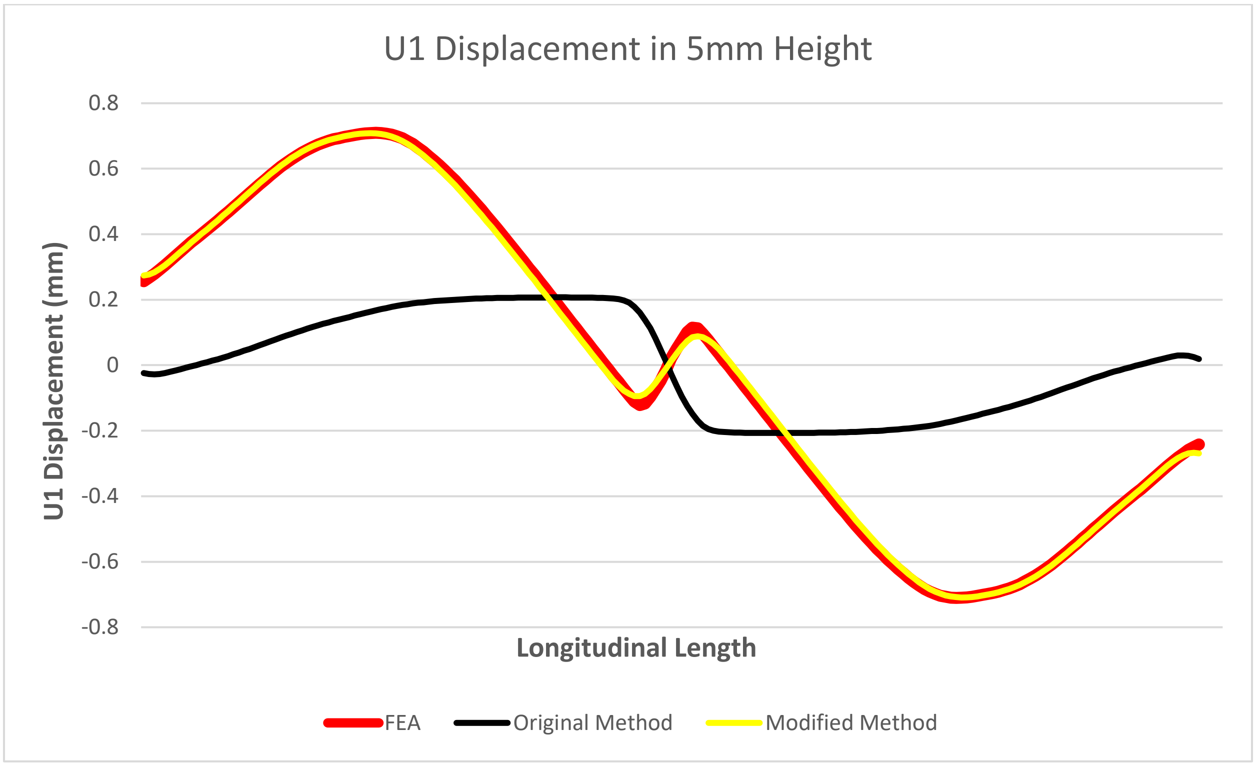


Fig. 21 U1 Displacement in the longitudinal direction in 5mm Height

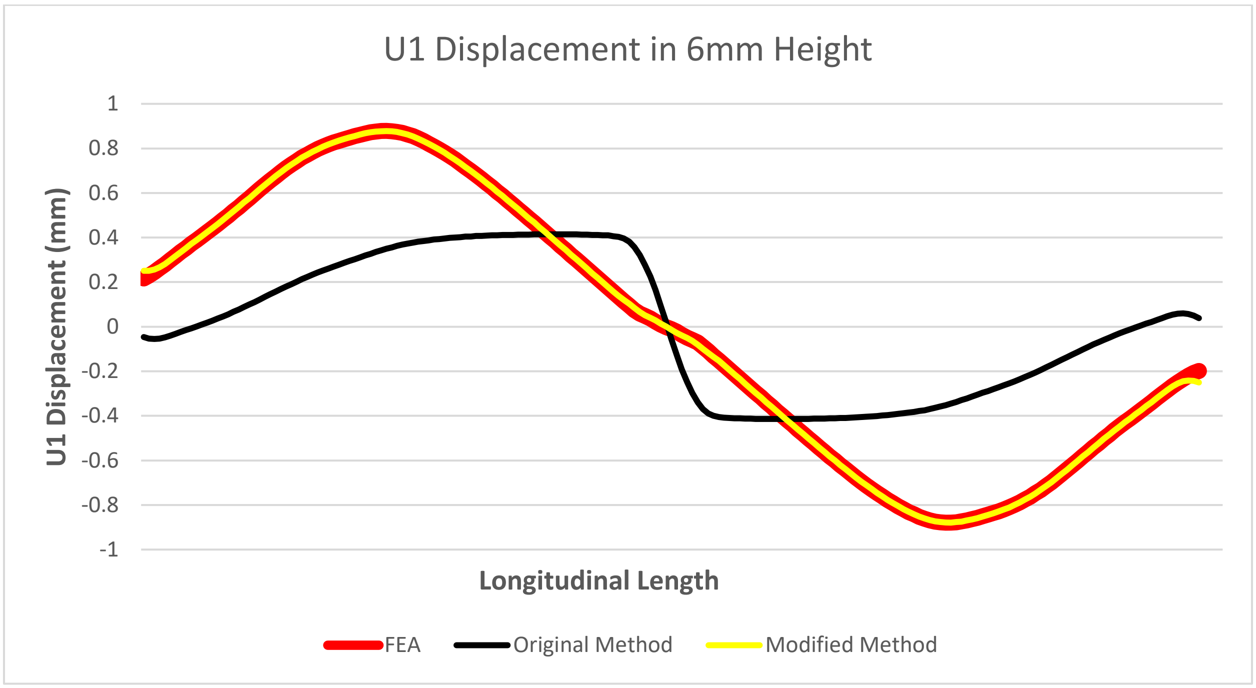


Fig. 22 U1 Displacement in the longitudinal direction in 6mm Height

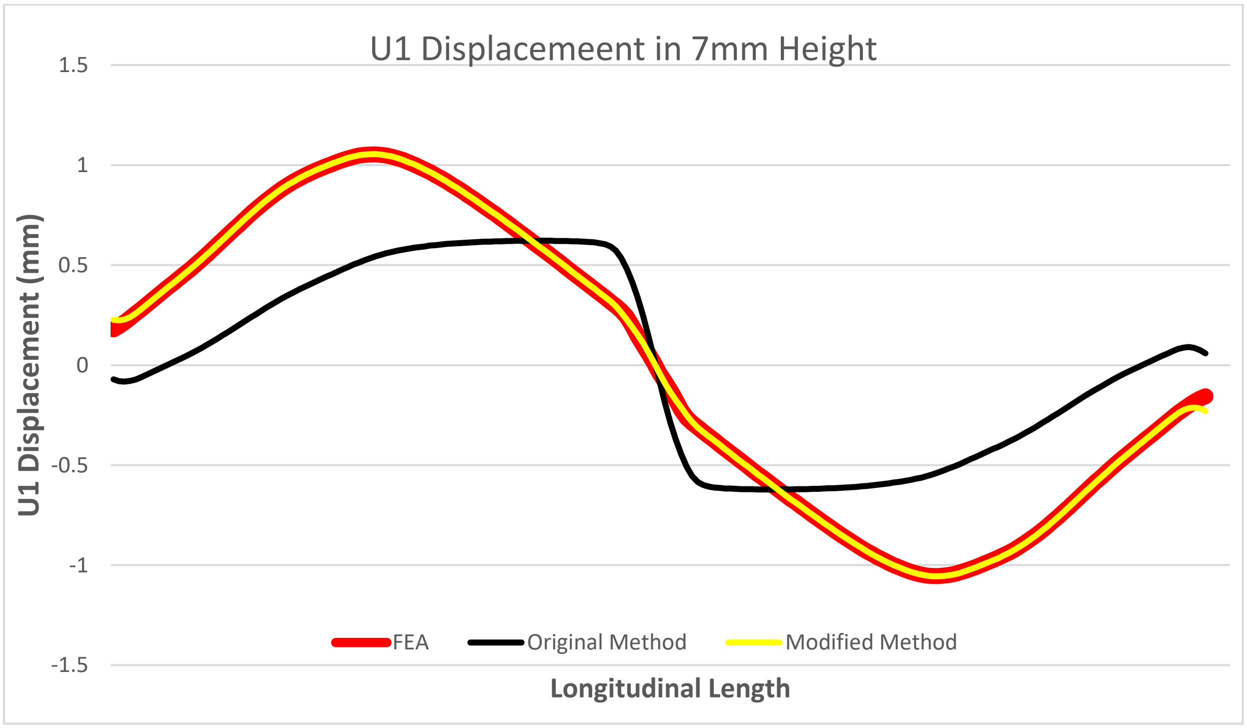


Fig. 23 U1 Displacement in the longitudinal direction in 7mm Height

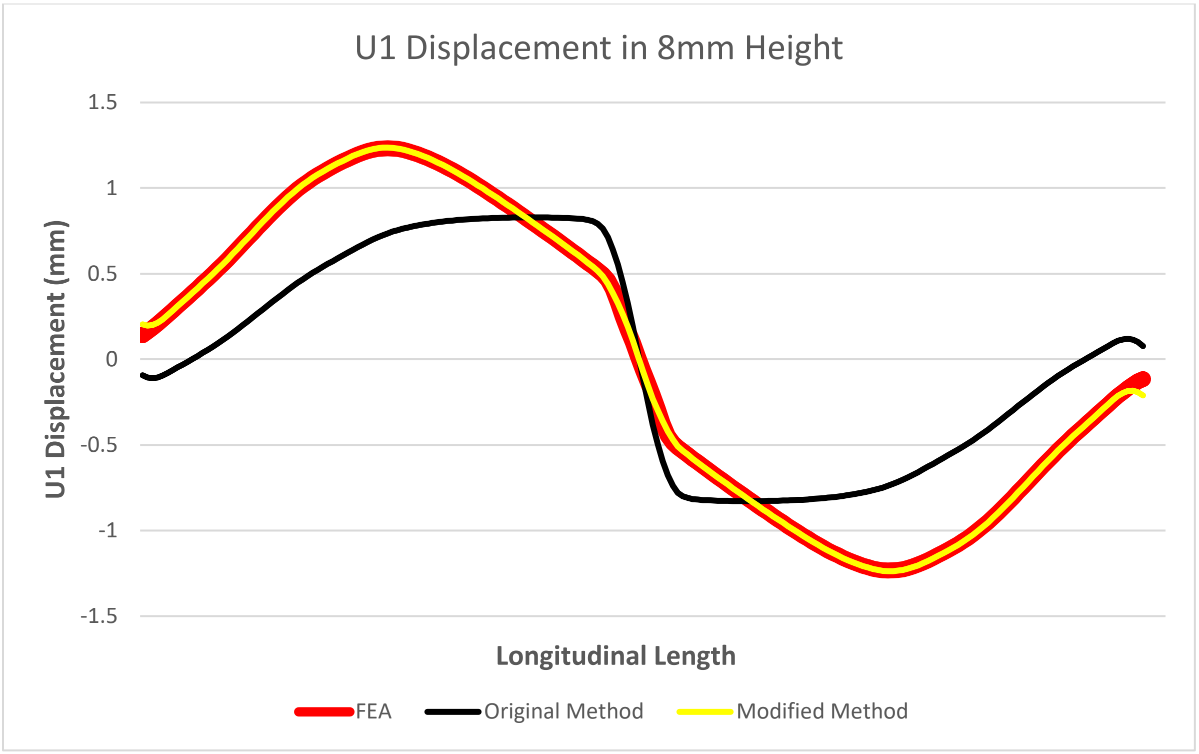


Fig. 24 U1 Displacement in the longitudinal direction in 8mm Height

2.4.2 Strain Analysis

As mentioned in the previous sections, the159 different 3D models in 7 different categories are used to verify a new method to predict the Lagrangian strain in the longitudinal direction. To determine the efficiency of the new method we obtained the ratio of logarithmic strain (LE 33) in the longitudinal direction in the original and modified method to the results that came from FEA and used the average and standard deviation of the data in each category as the criteria for evaluating the new method. By understanding that a closer average to 1 and a smaller standard deviation to 0 represent more accurate results, the greater the level of confidence in the results. The below graphs show an example of the results in nine different layers

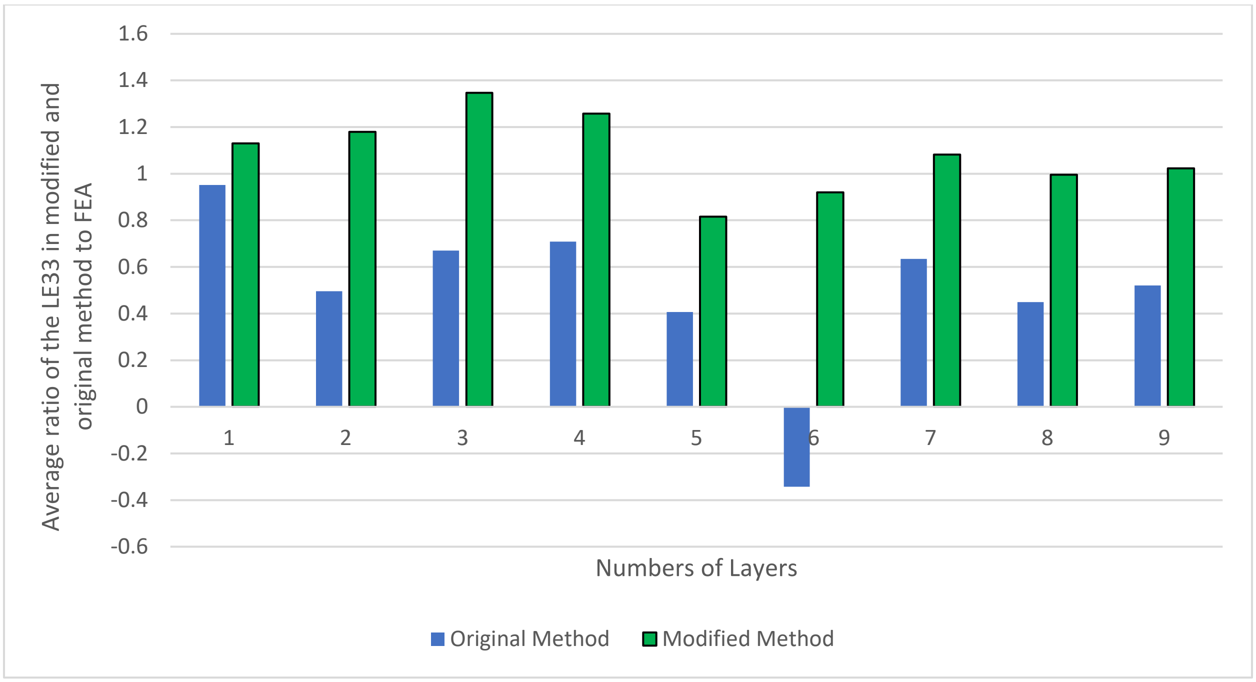


Fig. 25 The average ratio of the LE33 in modified and original method to FEA method

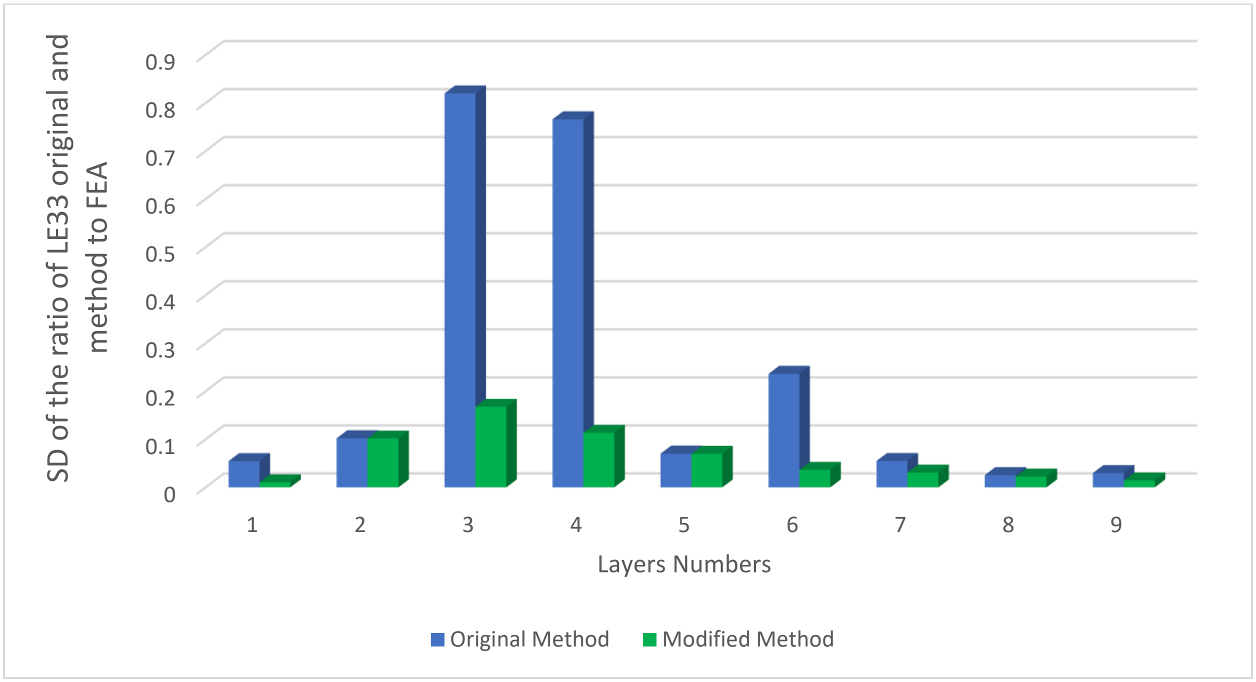


Fig. 26 The standard deviation ratio of the LE33 in modified and original method to FEA method

The logarithmic strain (LE11) in the longitudinal direction distributions for the 2D model are presented in Figs. 27 to 35. The horizontal axis in the graphs represents the longitudinal length of the pipe and the vertical axis represents the strain in the longitudinal direction for the pipe (the first 50 mm and the last 50 mm of the pipe have been removed because the critical points are in the middle part). For the pressurized dented pipes, the maximum strain concentrations are in the center and toward the shoulders of the dent. Unlike the previous situation for the dented pipe without pressure, the maximum strain considerations are just in the center part of the pipe.

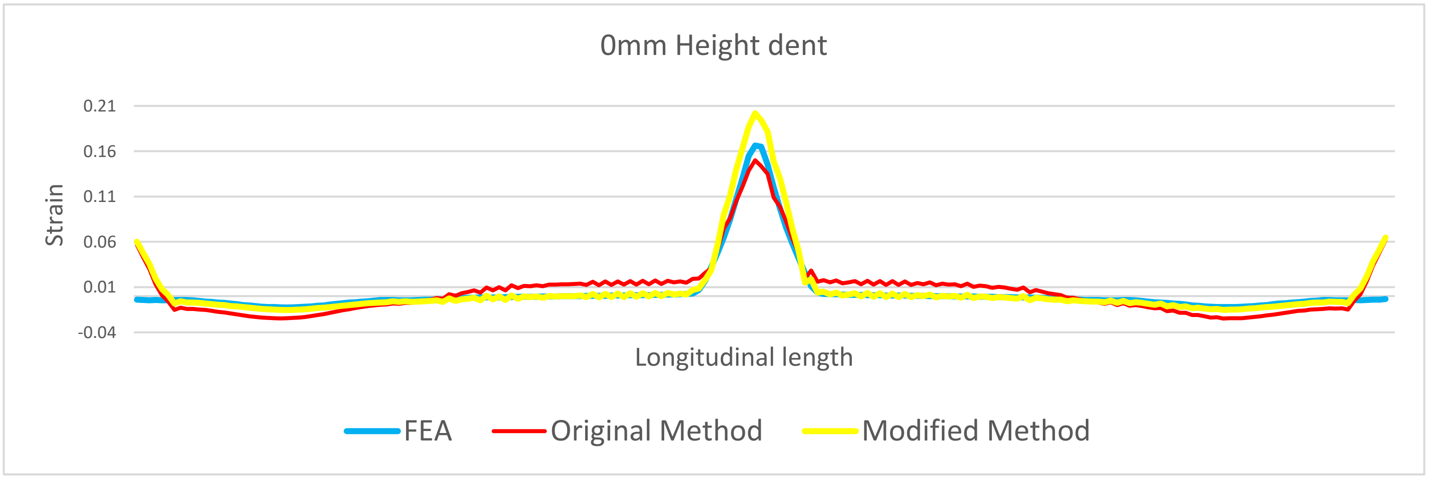


Fig. 27 Plots of the LE11 in the longitudinal direction in the 0 mm height of the pipe

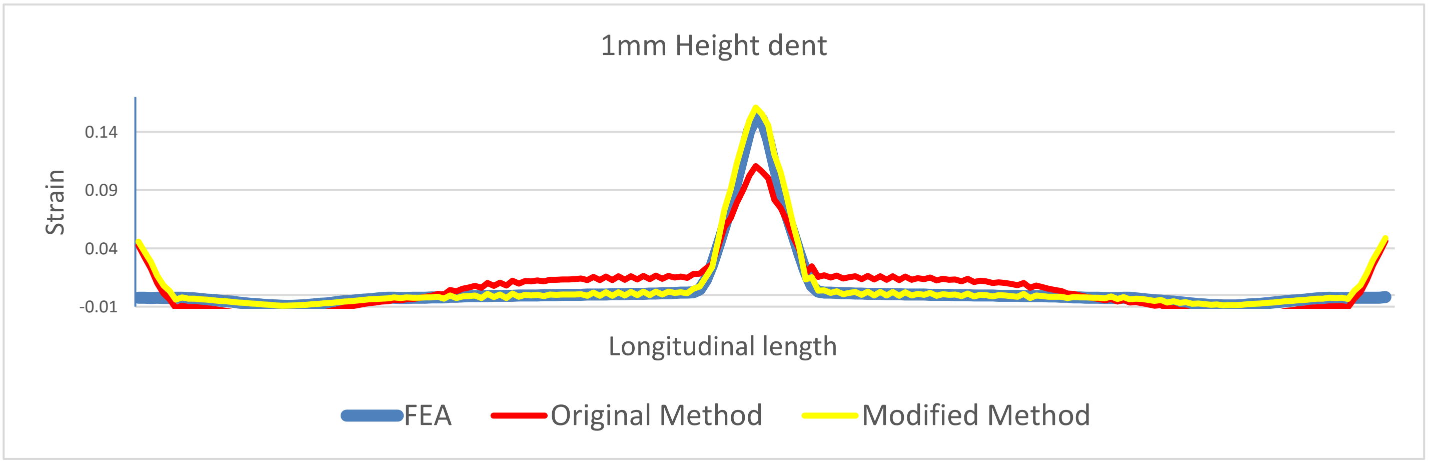


Fig. 28 Plots of the LE11 in the longitudinal direction in the 1 mm height of the pipe

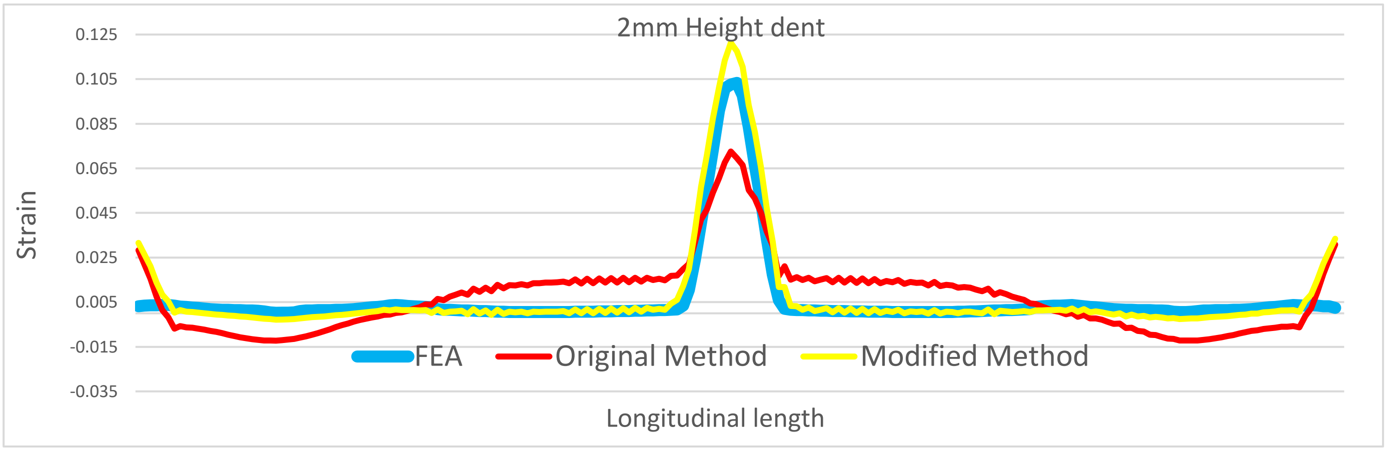


Fig. 29 Plots of the LE11 in the longitudinal direction in the 2 mm height of the pipe

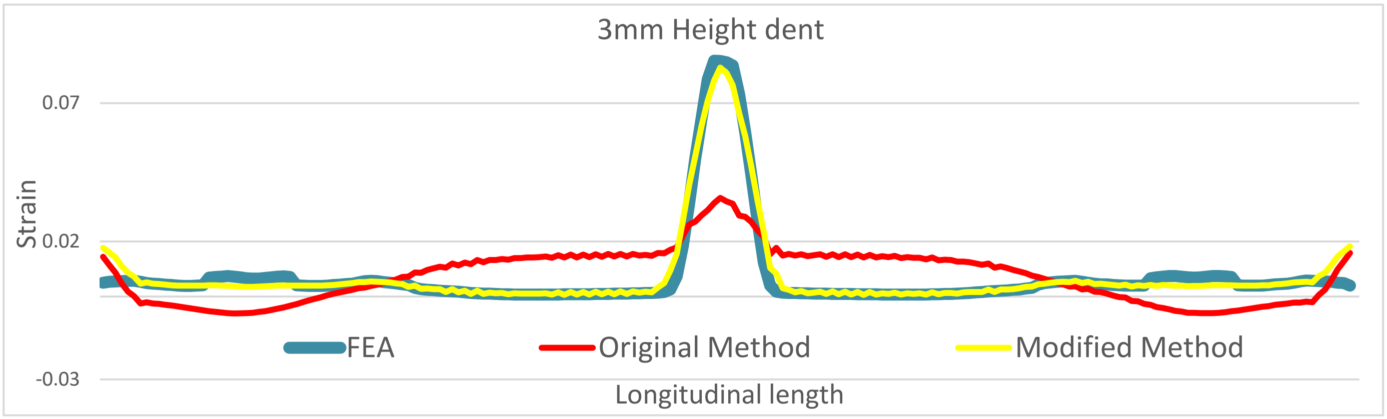


Fig. 30 Plots of the LE11 in the longitudinal direction in the 3 mm height of the pipe

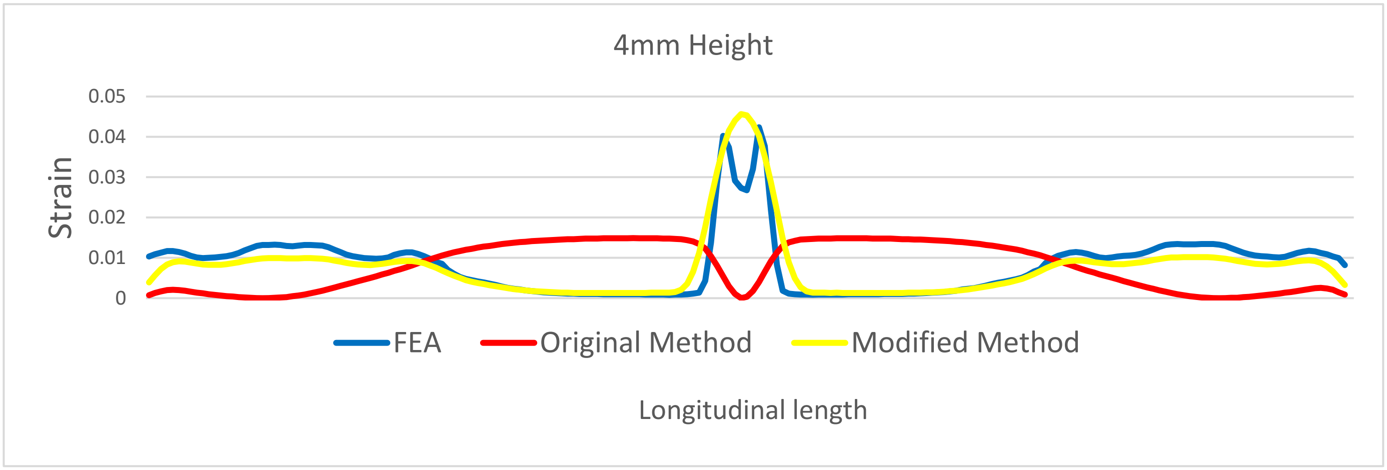


Fig. 31 Plots of the LE11 in the longitudinal direction in the 4 mm height of the pipe

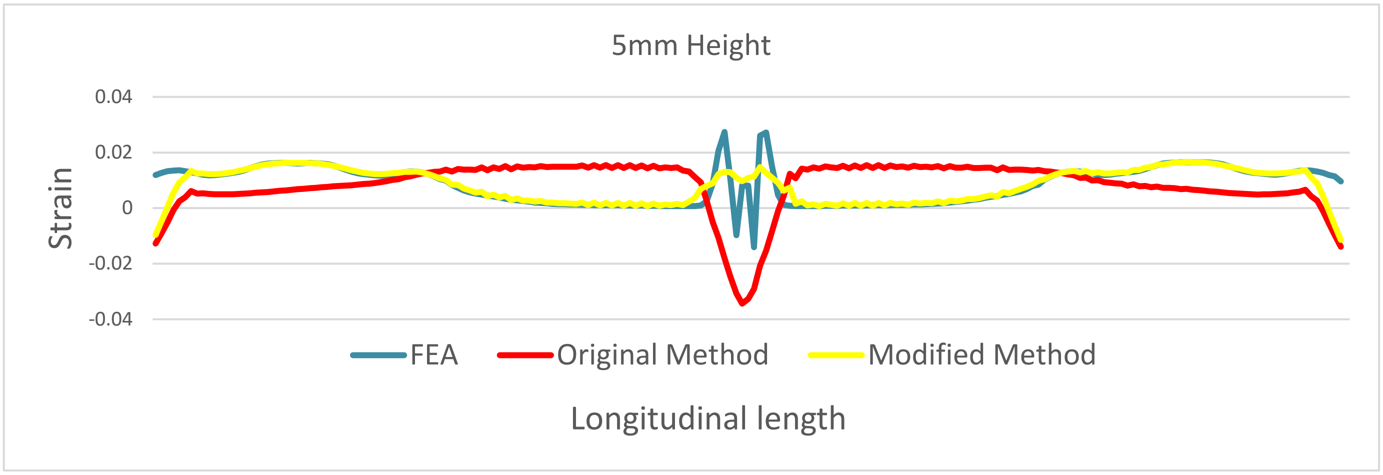


Fig. 32 Plots of the LE11 in the longitudinal direction in the 5 mm height of the pipe

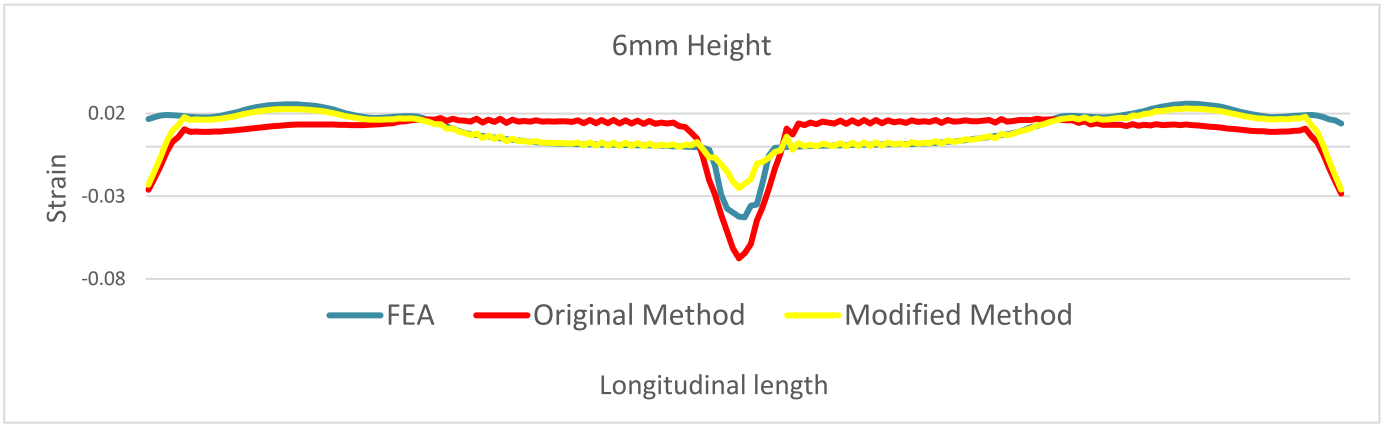


Fig. 33 Plots of the LE11 in the longitudinal direction in the 6 mm height of the pipe

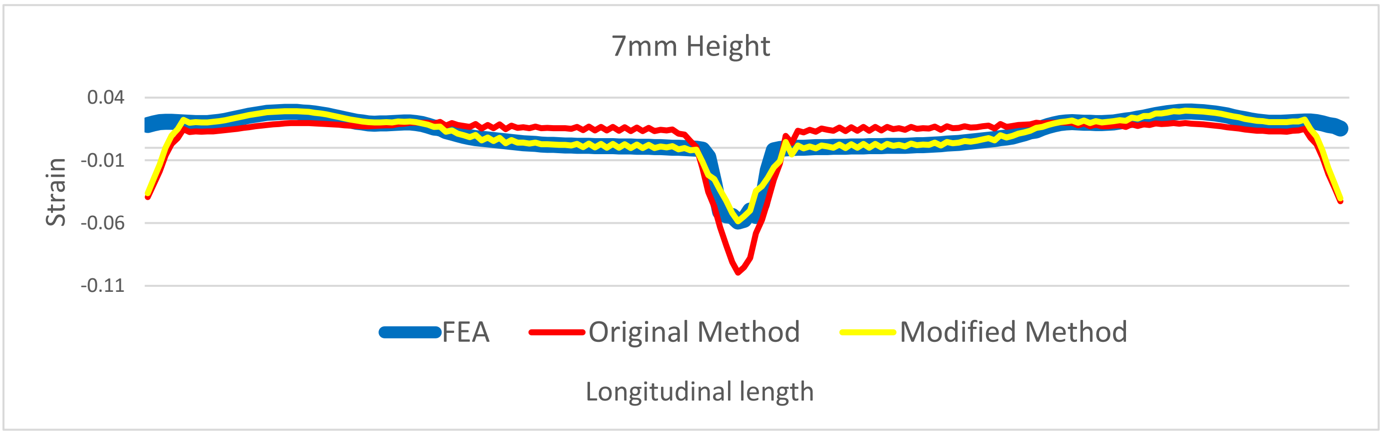


Fig. 34 Plots of the LE11 in the longitudinal direction in the 7 mm height of the pipe

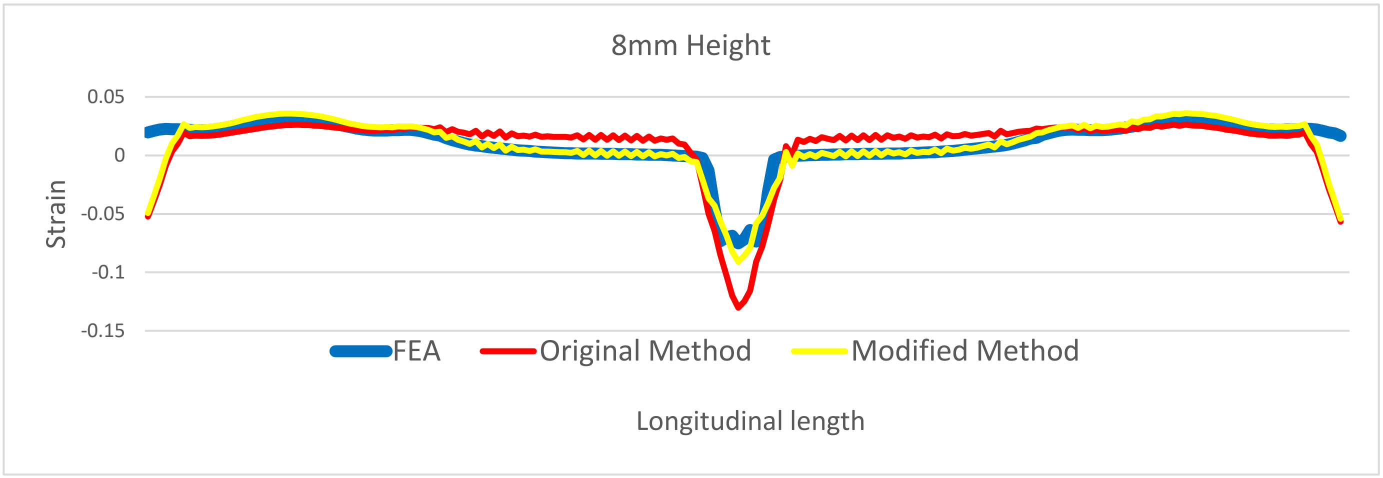


Fig. 35 Plots of the LE11 in the longitudinal direction in the 7 mm height of the pipe

2.5 Discussion

According to the results from around 180 different two-dimensional and three-dimensional models the use of the modified method can produce comparably more accurate strains. According to Figures 27 to 35, the results obtained for strain in all layers appear to be much more accurate and closer to the FEA results than in previous studies. Moreover, it is worth noting that the prediction of the strain is a little bit more accurate than the previous results, which will improve reliability. According to Woo et al's discussions about this issue in their research, they confirm that although Okoloekwe results are more accurate than ASME B31.8, they do not have enough correlation with the results obtained from FEA. There is a notable difference between layers, especially in the middle and upper ones. As a result of making changes to this method, we achieved significant results. In other words, the results obtained by this method are more consistent with those obtained by the FEA method. Moreover, the reliability of these results exceeds one, which infers that they are reliable.

The new developed method for measuring strain is much faster than FEA. which takes a long time to run. For example, to run a 3D model requires at least two and a half hours. Depending on the circumstances, this time may increase. It should be noted that this time is unacceptable in industrial activities where time is a very important element. However, this method has succeeded in calculating the strain results from the displacement of the nodes in a very short time, and as mentioned in the first part, the results are close to reality and have significant reliability. As Okoloekwe et al. emphasized if the dent profile is not aligned with the most severe peak (off axis peaks), a three-dimensional analysis of dented pipelines will be necessary. It would be feasible to have these analyses carried out by smart inline inspection devices by using algorithms which are readily programmable, thus reporting a strain estimate instantly. This information can be used by operators to determine where they should focus their resources for dent management.

2.6 Conclusion

Through an analytical approach across all aspects of pipeline strain analysis, we are developing an effective strategy for allocating pipe dent repair resources. According to the present study, the modified method generated significantly better longitudinal strain distribution in the longitudinal direction than the conventional procedure. The new developed method provides a substantial improvement in in terms of accuracy without compromising the speed of the analytical method. According to these findings, a similar level of accuracy can be achieved by using this method to predict the maximum strains in dented regions as that achieved with FEA whose accuracy has been shown by other researchers. This convenient new method is feasible for system-wide implementation from both a time and resource standpoint which means operators would be able to assess a large number of dents with high reliability in a short period of time.

2.7 Acknowledge:

The authors acknowledge the financial support from Enbridge Pipelines Inc.

Chapter 3 - The improvement of A strain-based modeling approach for analyzing dented pipeline severity in the circumferential direction.

3.1 Introduction

Pipelines are the primary means of transporting many petroleum products and natural gas in North America and throughout the world. Approximately 700,000 km of energy pipelines are in service in Canada alone (Yukon Government, 2011). This is why maintaining and repairing these pipelines is becoming more and more important every day. As revealed in the previous chapter of possible pipeline damage, the dent is one of the biggest dangers facing pipelines. As is mentioned, FEA is a reliable method for analyzing stresses and strains within a dented region, though it is inefficient for analyzing many dent models as each is computationally costly (Woo et al., 2019).

In the previous chapter, after obtaining strain and displacement results from the FEA method, as well as Okoloekwe et al.'s strain base method (which will be called as original method), we concluded that changes should be made in this method in the longitudinal direction to obtain acceptable results. In this chapter, the main objective is to first examine whether or not the original method in the circumferential direction is acceptable based on criteria and standards, by using different 2D and 3D models in different conditions. In this way, it would be unnecessary to model each dent with FEA, though it would benefit from obtaining an accurate result from FEA. Comparing the results based on the original method with the results from the FEA method will then be done.

3.2 Method

3.2.1 Modeling of Dents in the circumferential direction

In this study, ABAQUS (version 2019) subroutine was used to create the 3D pipeline models. In Appendix A, you will find scripts for generating models and also in Appendix B you will find Mathematica code which is used for extracting results. Instead of analyzing the entire pipe, numerically and analytically we will investigate the lagrangian strain distribution along with its thickness. Fig. 36 and 37 show how the thickness of a pipe's wall is represented in a 2D and 3D model on the circumferential plane of the symmetry.

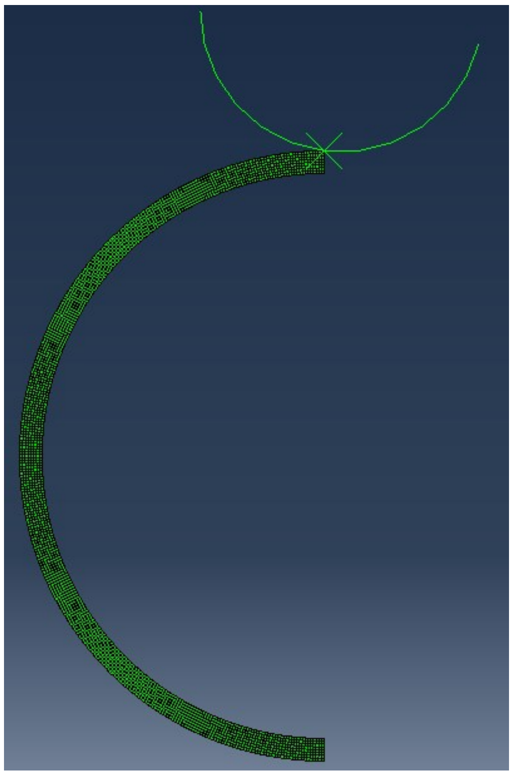


Fig. 36-a 2D model of wall thickness in the circumferential direction

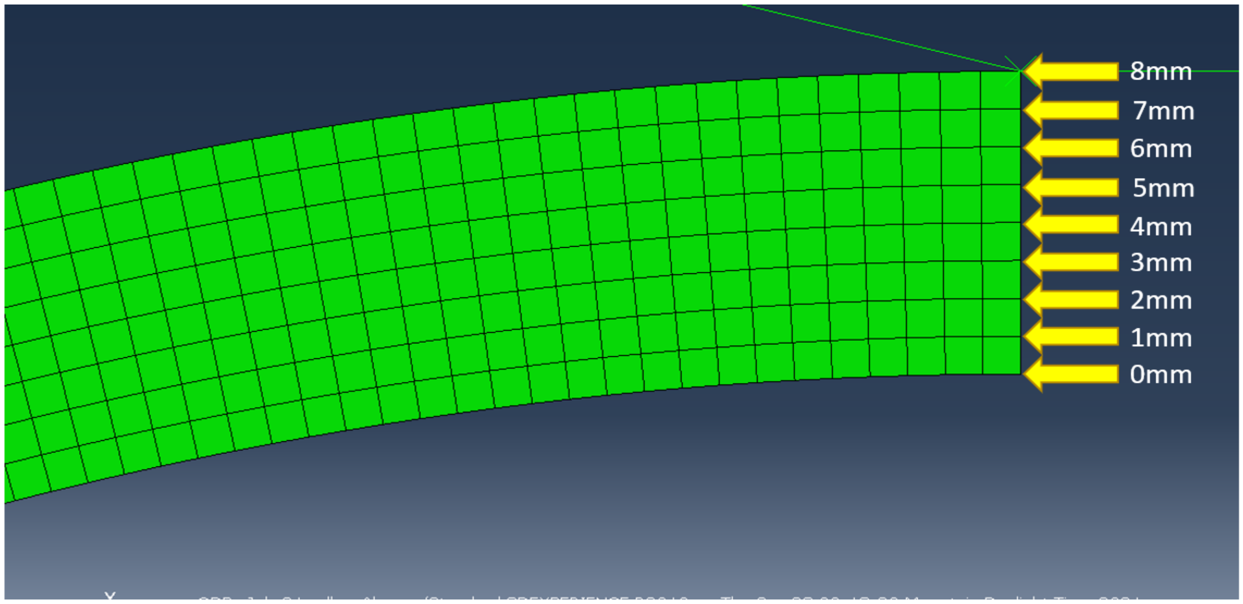


Fig. 36-b 2D model of wall thickness in the circumferential direction

An illustration of the wall thickness of a pipe, generated along the circumferential plane by a two-dimensional modeling technique, is shown in Fig. 33. A 300-mm-long pipe had a 216-mm outer diameter (OD) and the ratio of the outer diameter to the thickness is 27. To ensure that the formed dent would be a localized deformity, the 300 mm length was chosen to ensure that there would be no interaction between the dent and the end boundary condition.

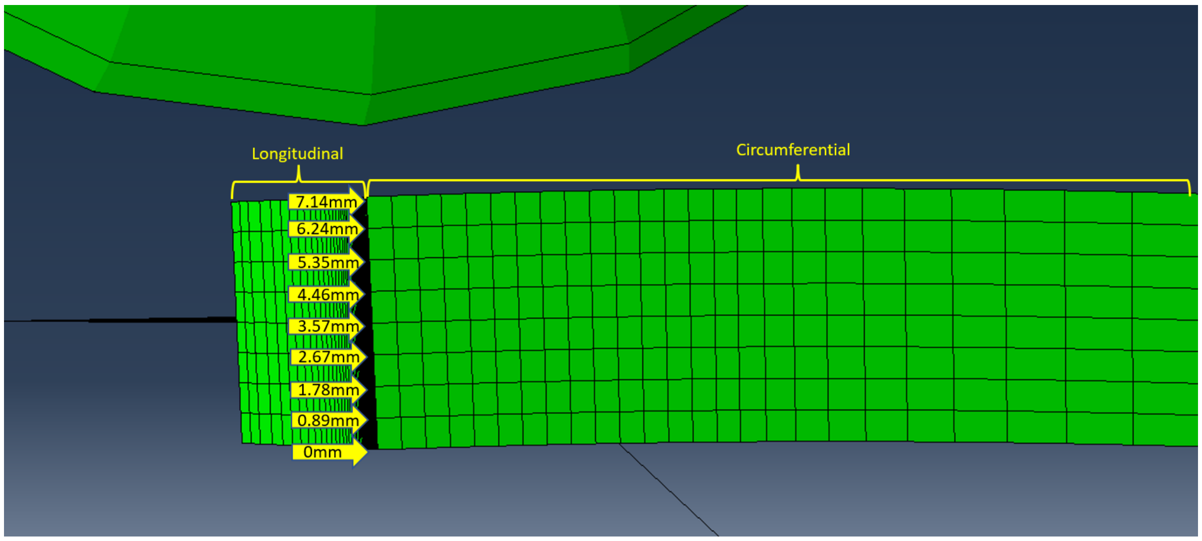
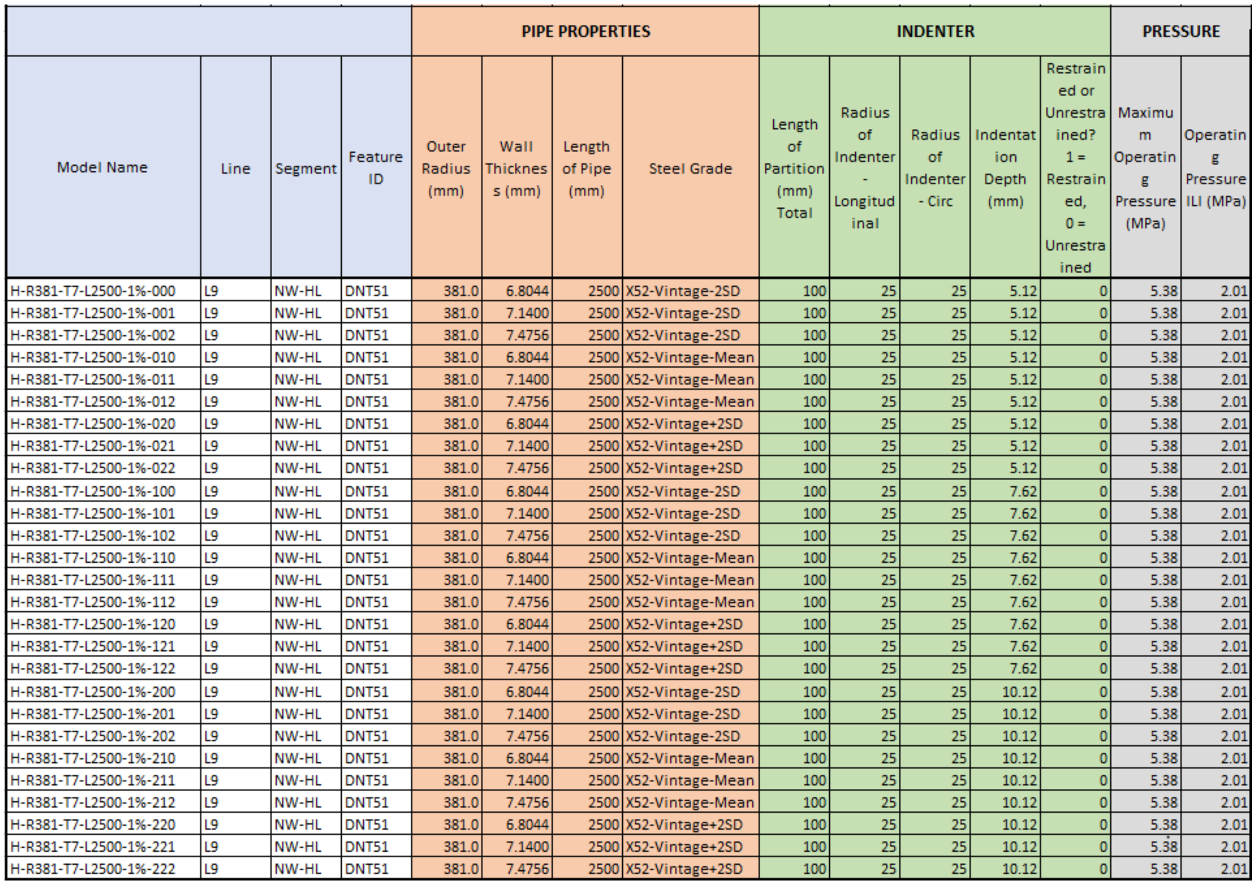


Fig. 37 3D model of wall thickness in the circumferential direction

In Fig. 37, the wall thickness generated along a circumferential plane is visualized in a 3D model.

In order to verify the main method, more than 40 two-dimensional models in different conditions and 159 three-dimensional models have been performed to evaluate the reliability of this method.

Among the models presented in the following table, for example in the fourth category, these models have been studied in certain types along a certain length of pipe with different thicknesses and different types of steels at different indentation depths and at certain pressures to obtain very comprehensive and reliable results. Running the models for each of which took about three hours and a total of about 600 hours.



				PIPE PROPERTIES				INDENTER					PRESSURE	
Model Name	Line	Segment	Feature ID	Outer Radius (mm)	Wall Thickness (mm)	Length of Pipe (mm)	Steel Grade	Length of Partition (mm) Total	Radius of Indenter - Longitudinal	Radius of Indenter - Circ	Indentation Depth (mm)	Restrained or Unrestrained? 1 = Restrained, 0 = Unrestrained	Maximum Operating Pressure (MPa)	Operating Pressure ILI (MPa)
H-R381-T7-L2500-1%-000	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-001	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-002	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-010	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-011	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-012	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-020	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-021	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-022	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	5.12	0	5.38	2.01
H-R381-T7-L2500-1%-100	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-101	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-102	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-110	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-111	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-112	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-120	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-121	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-122	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	7.62	0	5.38	2.01
H-R381-T7-L2500-1%-200	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-201	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-202	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-210	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage-Mean	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-211	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage-Mean	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-212	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage-Mean	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-220	L9	NW-HL	DNT51	381.0	6.8044	2500	X52-Vintage+2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-221	L9	NW-HL	DNT51	381.0	7.1400	2500	X52-Vintage+2SD	100	25	25	10.12	0	5.38	2.01
H-R381-T7-L2500-1%-222	L9	NW-HL	DNT51	381.0	7.4756	2500	X52-Vintage+2SD	100	25	25	10.12	0	5.38	2.01

3.2.2 Displacement Discretization

As outlined in the previous chapter, a cylindrical coordinate system was used for the analytical modelling of the dented pipe, which allows the determination of the deformed pipeline coordinates, as explained in (Okoloekwe, et al., 2019).

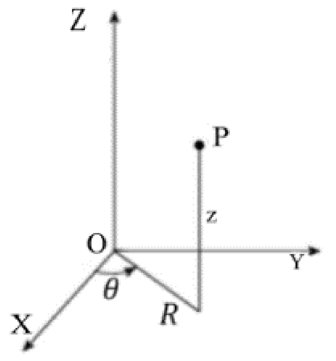


Fig. 38 cylindrical coordinate System

In a cylinder coordinate system, the global displacement field can be expressed by Equation (27):

$$\mathrm{u} = \mathrm{u_r e_r} + \mathrm{u_\theta e_\theta} + \mathrm{u_z e_z} \qquad (27)$$

The displacement gradient is represented by equation (28).

$$\nabla u = \begin{bmatrix} \frac{\partial \mathrm{u_r}}{\partial \mathrm{R}} & \frac{\partial \mathrm{u_r}}{\mathrm{R}\,\partial\theta} - \frac{\mathrm{u_\theta}}{\mathrm{R}} & \frac{\partial \mathrm{u_r}}{\partial \mathrm{z}} \\ \frac{\partial \mathrm{u_\theta}}{\partial \mathrm{R}} & \frac{\mathrm{u_r}}{\mathrm{R}} + \frac{\partial \mathrm{u_\theta}}{\mathrm{R}\,\partial\theta} & \frac{\partial \mathrm{u_\theta}}{\partial \mathrm{z}} \\ \frac{\partial \mathrm{u_z}}{\partial \mathrm{R}} & \frac{\partial \mathrm{U_2}}{\mathrm{R}\,\partial\theta} & \frac{\partial \mathrm{u_z}}{\partial \mathrm{z}} \end{bmatrix} \qquad (28)$$

A deformed pipeline's mid surface is defined by its radius R.

There is a great deal of difficulty in deriving a theoretical solution for local deformations of a pipeline without the simplifying assumptions associated with geometrical and material nonlinearities.

In the original method, the pipe walls' mid surface is assumed to be straight and uniform prior to deformation. To gain better strain results we will add mid-surface displacement to the results. The hypothetical radius of the mid surface of the deformed pipe is evaluated by:

$$R_{hyp}(z) = \int_{-\pi}^{\pi} \frac{R_m(\theta,z)\,\partial\theta}{2\pi} \tag{29}$$

where *Rm* is the radius of the mid surface of the deformed pipeline and ϕ is the angular distortion of the deformed pipeline.

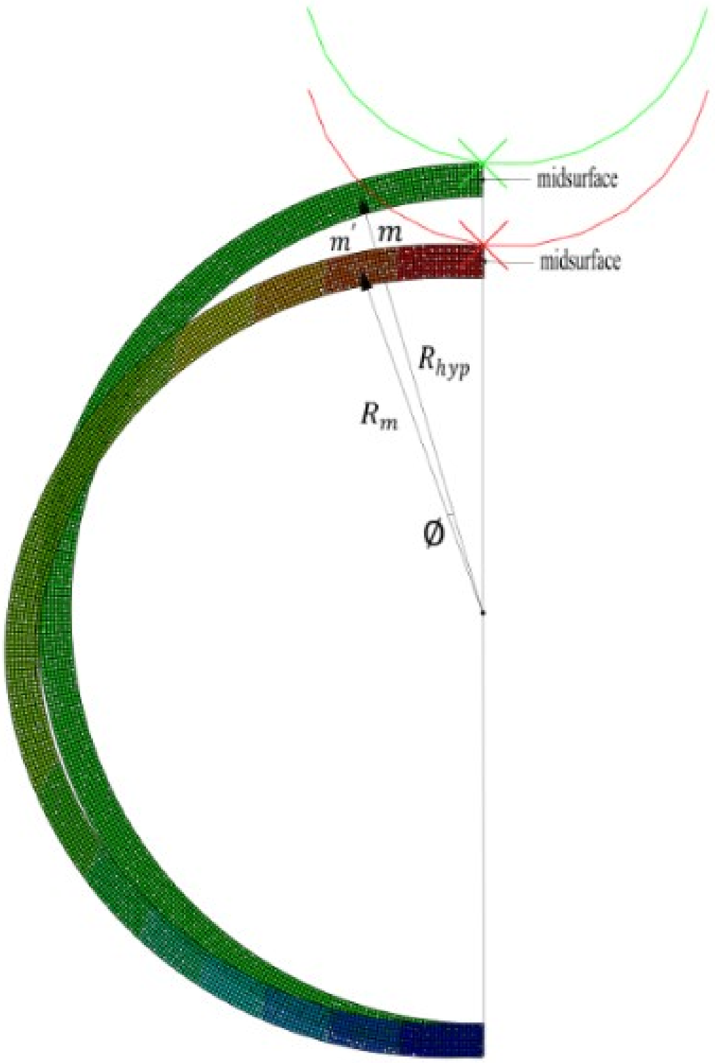


Fig. 39 displacement along the circumferential direction

The circumferential displacement:

$$u_\theta = R_m \mathrm{Sin}(\emptyset) - t_v \mathrm{Sin}(\emptyset - \theta_\theta) \qquad (30)$$

For strain measurement in the circumferential direction, it is assumed that the linear strain or the small strain is calculated by using Eq. (31):

$$\varepsilon_L = \frac{1}{2}(\nabla u + \nabla u^T) \qquad (31)$$

3.3 Results:

The analysis for this direction can be produced by solving the expression 30 along its circumferential axis. In order to calculate R_m (the radius after deformation), we obtained the coordinates of each node and then calculated R_m by using the $\sqrt{x^2 + y^2}$.

3.3.1 Numerical Models

An example of a numerical model for the deformed pipes and the indenter is illustrated in figure 40.

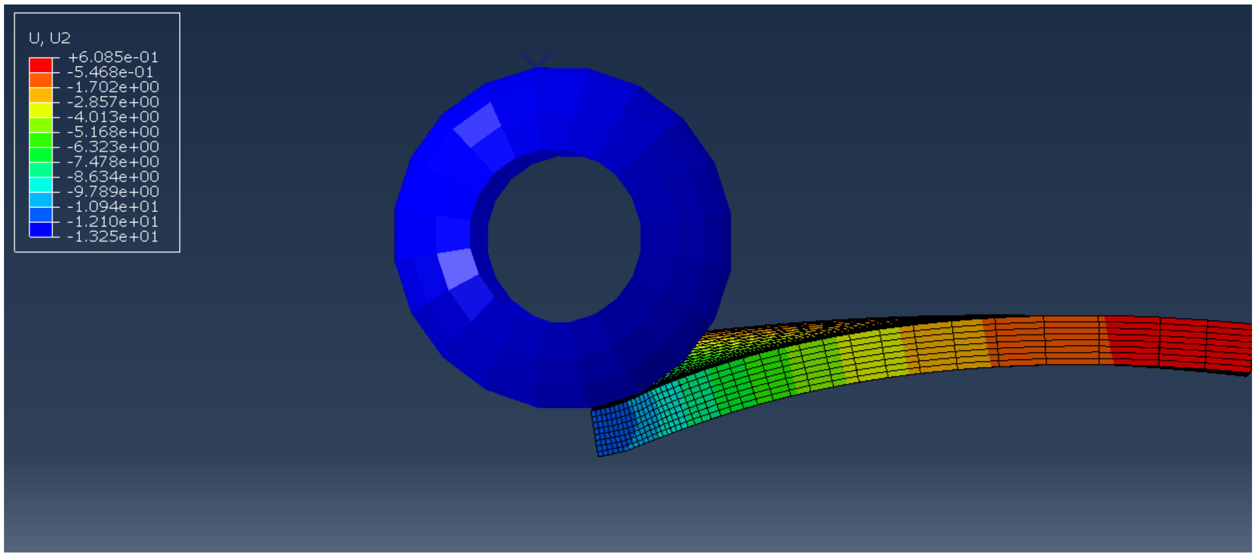


Fig. 40 Numerical Models

3.3.2 Strain Analysis

It was already mentioned that 159 unique 3D models are used to test an original method for determining the logarithmic strain (LE11) in the circumferential direction and 40 different 2D models are also examined. We used the average and standard deviation of the data in each category as the criteria to correctly diagnose the efficiency of the original method and we calculated the ratio of LE11 in the original method to the FEA results. It is more reasonable to have higher levels of confidence in the results of the analysis if the average is closer to 1 and the standard deviation is close to 0. The graphs below show an example of these results.

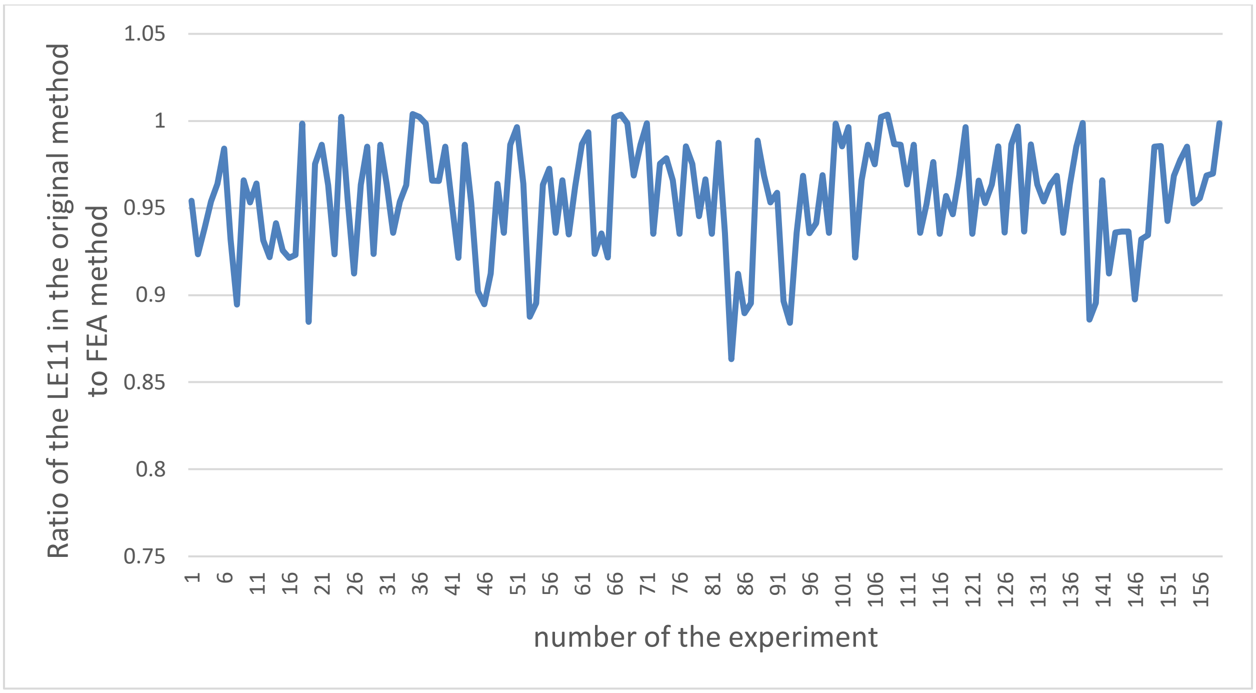


Fig. 41 Ratio of the LE11 in the top of the pipe in the circumferential direction in 3D model

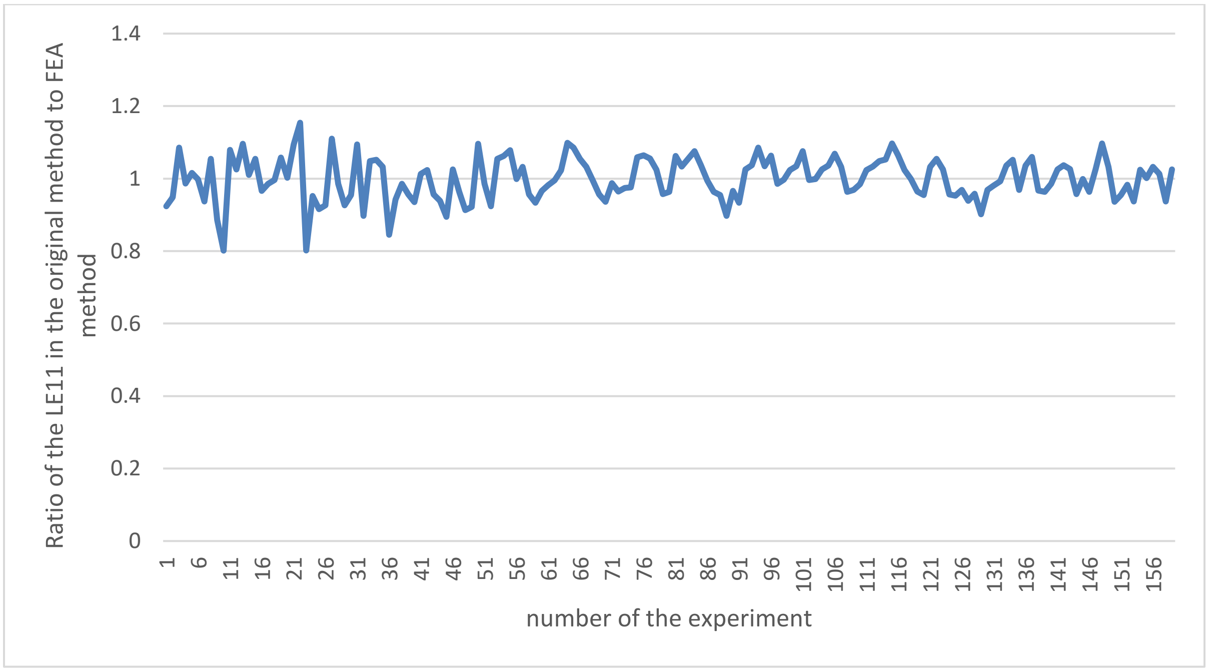


Fig. 42 Ratio of the LE11 in the bottom of the pipe in the circumferential direction in 3D model

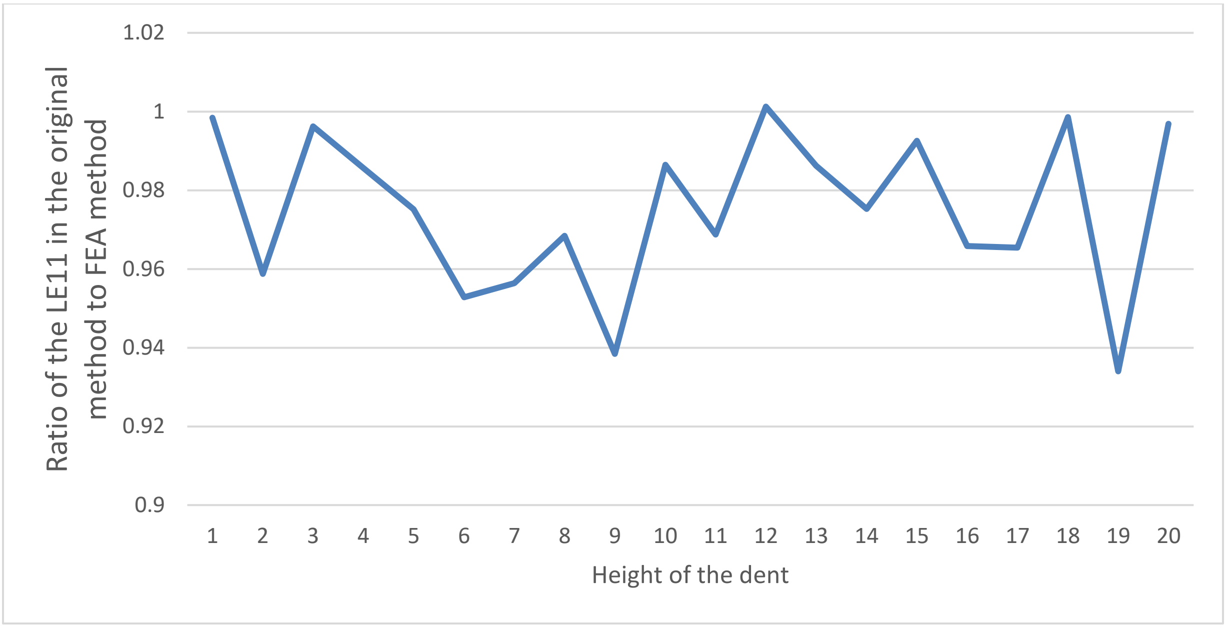


Fig. 43 Ratio of the LE11 in the bottom of the pipe in the e circumferential direction in 2D model with pressure

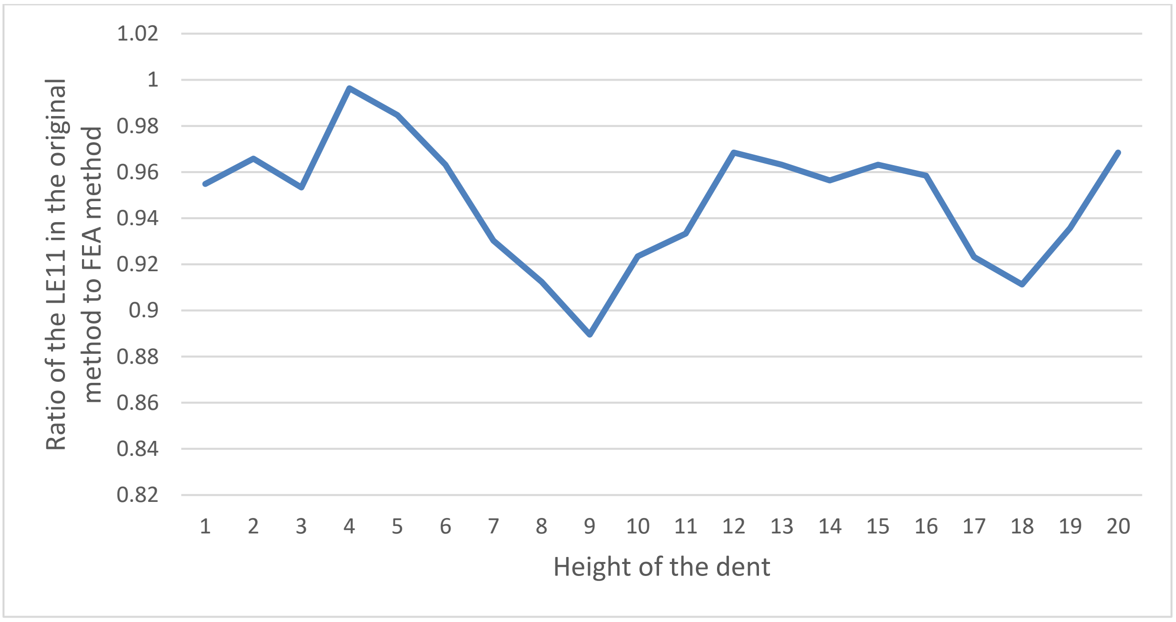


Fig. 44 Ratio of the LE11 in the top of the pipe in the circumferential direction in 2D model with pressure

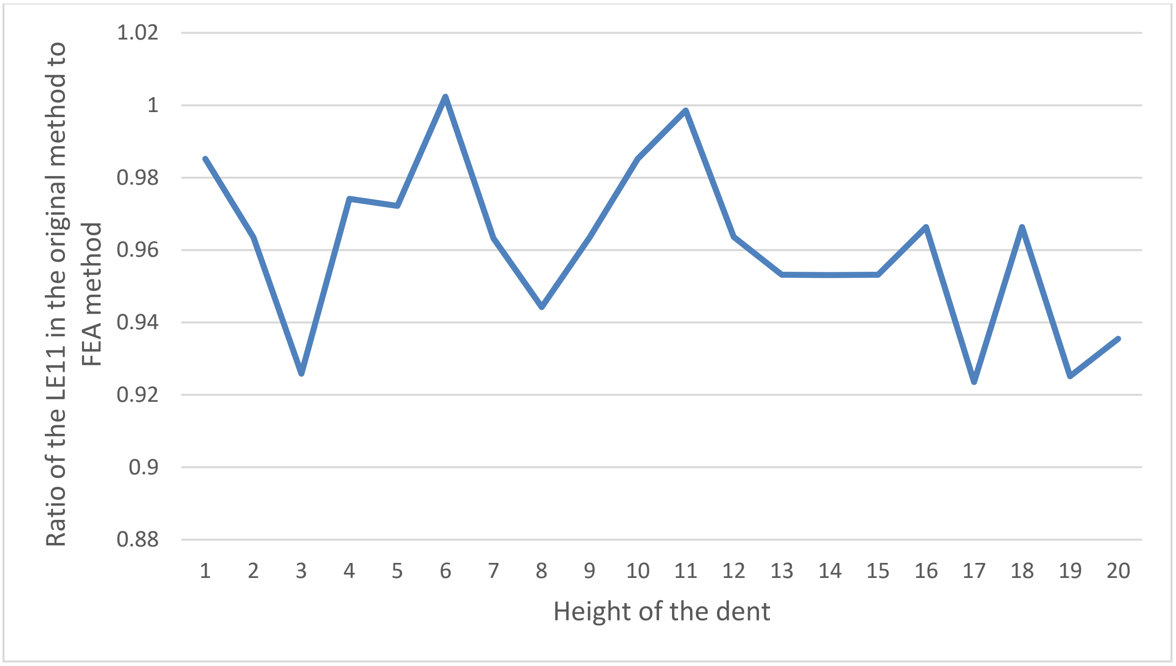


Fig. 45 Ratio of the LE11 in the bottom of the pipe in the circumferential direction in 2D model without pressure

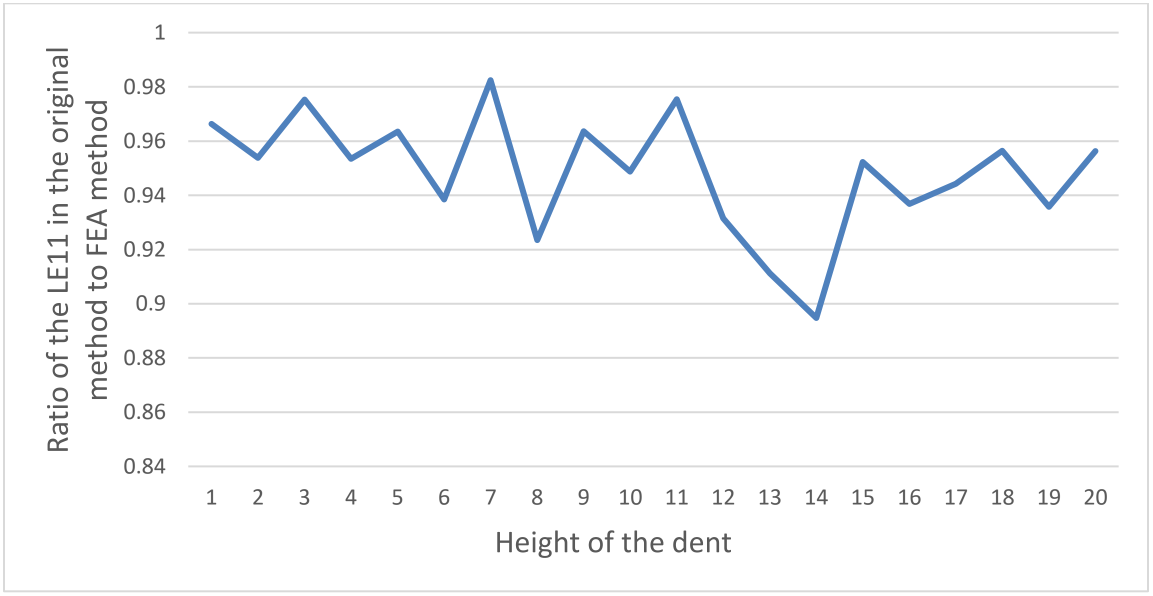


Fig. 46 Ratio of the LE11 in the top of the pipe in the in the circumferential direction in 2D model with pressure

In the following graphs the average ratio and standard deviation ratio for different categories of the pipe is presented.

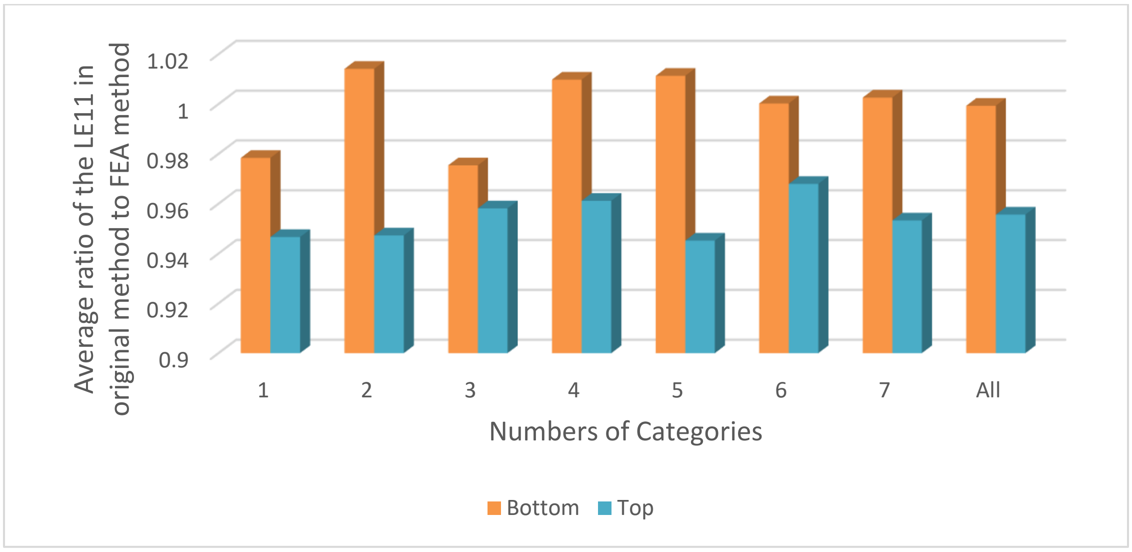


Fig. 47 Average ratio of the LE11 in the bottom of the pipe in the circumferential direction in 3D model

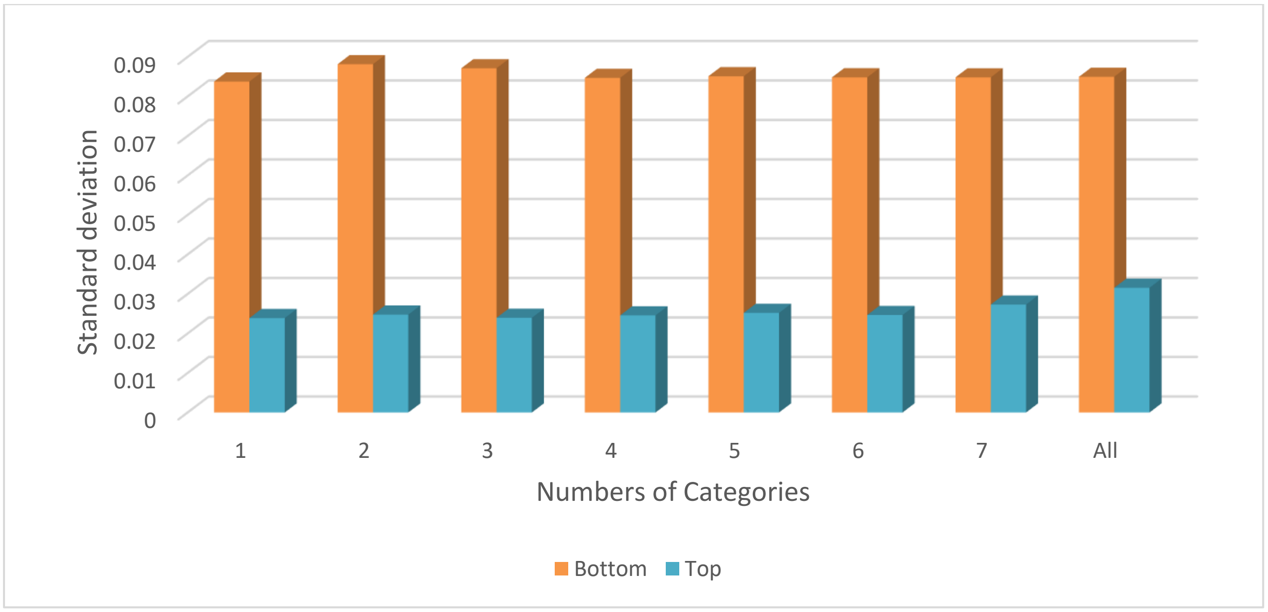


Fig. 48 Standard deviation of the LE11 in the bottom of the pipe in the circumferential direction in the3D model

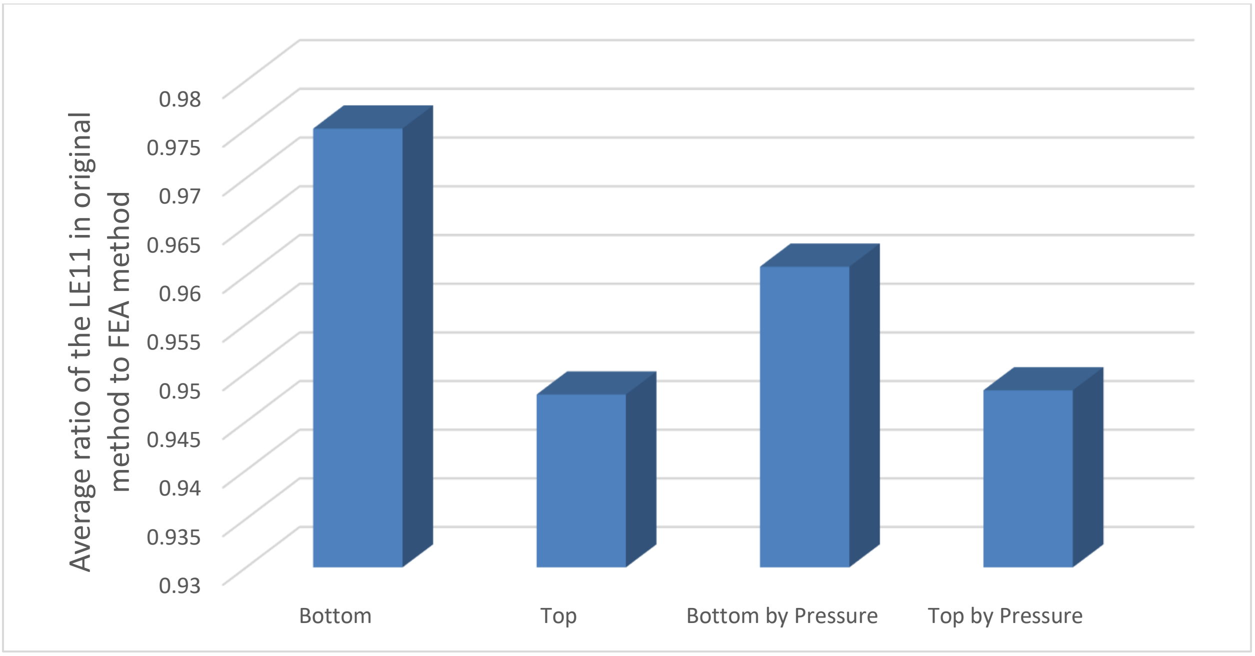


Fig. 49 Average ratio of the LE11 in the circumferential direction in 2D model

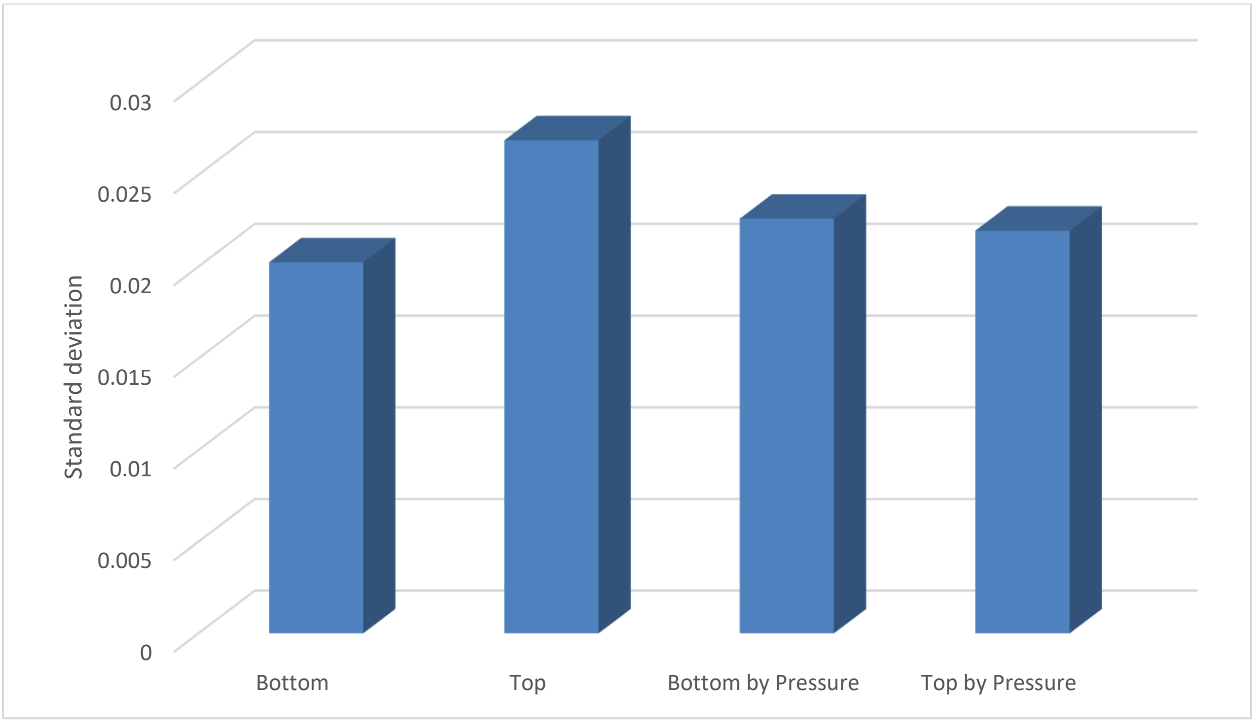


Fig. 50 Standard deviation of the LE11 in the circumferential direction in 2D model

3.4 Conclusion

A brief summary of the conclusion of this chapter follows the detailed discussion in previous sections. As part of the process of verifying the method, two-dimensional and three-dimensional models were examined. It was found that the results in this section were very accurate in contrast with the longitudinal results discussed in Chapter 2. In other words, the current method is much faster than the FEA method, and the accuracy and reliability of the results are acceptable based on the FEA results. Graphs 40 through 43 show that in the original method and the FEA method, the average ratio is very close to one, which indicates the method's high accuracy. Where for the 3D models the ratio quantity was reported as 0.978 +/- 0.0837, 1.014 +/- 0.088, 0.975 +/- 0.0871, 1.009 +/- 0.0847, 1.001 +/- 0.0851, 1.000 +/- 0.0848, 1.0026 +/- 0.0848 for the first to seventh categories in the bottom side of the pipeline and for all of them is 0.999 +/- 0.0849. And for the top side of the pipe in the 3D models are 0.946 +/- 0.0239, 0.947 +/- 0.0247, 0.958 +/- 0.0239, 0.961 +/- 0.0245, 0.945 +/- 0.0252, 0.967 +/- 0.0246, 0.953 +/- 0.0273 and for all of them is 0.955 +/- 0.0315. And also, for 2D models the ratio quantities are 0.975 +/- 0.020, 0.947 +/- 0.0268 for the bottom and top sides of the pipeline without pressure. As well as 0.960 +/- 0.022, 0.948 +/- 0.0219 for the bottom and top side of the pipeline under pressure.

CHAPTER 4: CONCLUSIONS AND FUTURE RESEARCH

4.1 Summary and Conclusions

Our main goal is to further develop a strain-based modeling approach for assessing the severity of dented pipelines. Many factors determine how vulnerable pipes are to dent-related risks. As has been stated many times since the first introduction, material and the length of the pipe are involved in this case. In addition, shape, size location, interaction with pipe features, and operation properties are the important factors for determining the severity of the dent. Having an accurate method to perform dents integrity assessments will increase the reliability of pipelines while ensuring that resources are utilized efficiently. Finite element analysis has been proposed for assessing dents, but this method requires a significant amount of computing time and can only be used in very limited contexts. Therefore, a useful method is that solves the time problem (with which the FEA has a problem) and can avoid spending too much time analyzing models while being accurate enough to be believable in the industry. Developing such a method was the purpose of this thesis.

In the second chapter of this thesis, we focused on improving the results of a method that had previously been developed. First, the results were examined in the longitudinal direction, for which different models in different conditions were considered. By adding the displacement of the middle layer to the displacement of the other layers in the horizontal direction, the logarithmic strains reached considerable reliability because, in the previous method, the amount of these strains was less than the amount estimated by the FEA method. While in the developed method, all the maximum strains obtained are more than the results obtained in the FEA method. All of the results pass these conditions since the difference between them is up to 6%.

Using code in Mathematica, as outlined in the Appendix section, the displacements in the horizontal direction were first determined by using the vertical displacements and then compared graphically with the FEA results. Based on displacement, the logarithmic strain results were calculated in the longitudinal direction for both the original and modified method and compared to FEA results.

Finally, in the third chapter, the original method was first verified by 159 different 3D models in 7 categories and 40 2D models. The results obtained and their comparison with the FEA results revealed that there was no need to modify the original method as opposed to the longitudinal direction results which required changes to the method.

According to these results, both the modified method in the longitudinal direction and the original method in the circumferential direction can provide similar results to what is already known about FEA, which is already proven to be an accurate representation of reality by other researchers. These methods are useful because they allow users to achieve results in a shorter amount of time than using FEA alone and might be feasible for system-wide use from both a time and resources perspective.

4.1.1 Accomplishments

The following are the most important achievements of this research.

1- Propose and accurate and realistic method by adding the horizontal displacement in the middle layer of the pipe to the horizontal displacement in other levels.

2- Illustrating the function of and accuracy of the modified method and the original method in the longitudinal and circumferential direction by using multiple models in different conditions.

3- Achieving results with acceptable speed compared to FEA method.

4.2 Future Research

This study has proven that it is possible to assess dents in the longitudinal direction by using the modified method and the circumferential direction by using the original method. There are many different types of dent types and pipe properties found on pipeline systems, but the research did not examine all of them.

Efficiencies and accuracy of the proposed methods may be further improved through future research.

In the future, it would be better to focus on improving the accuracy of horizontal displacement predictions than to bring results closer to reality. The results of the FEA method were also used as a way to contribute to the horizontal displacement of other layers in the modified method for horizontal displacement in the middle layer, which had been viewed as zero in the original method. It is recommended that future research on horizontal displacement prediction in the middle layer be based on ILI data. This makes the modified method of predicting horizontal displacement in the middle layer complete.

Bibliography

Kiefner J, Leewis K. Pipeline Defect Assessment – A Review &amp; Comparison of Commonly Used Methods. Worthington, Ohio, USA: Kiefner and Associates, Inc.; 2011. 238 p. Report No.: PR-218-05405.

Gao, M., McNealy, R., Krishnamurthy, R., and Colquhoun, I., 2008, "Strain-Based Models for Dent Assessment—A Review," ASME Paper No. IPC2008-64565.

ASME, 2016, "Gas Transmission and Distribution Piping Systems," American Society of Mechanical Engineers, New York, Standard No. ASME B31.8-2016.

Lukasiewicz, S. A., Czyz, J. A., Sun, C., and Adeeb, S., 2006, "Calculation of Strains in Dents Based on High Resolution In-Line Caliper Survey," ASME Paper No. IPC2006-10101.

Noronha, D. B. Jr, R. Martins, B.P. Jacob, E. Souza, "The use of B- Splines in the Assessment of Strain Levels with Plain Dents". Proceedings of the Rio Pipeline Conference and Exposition 2005, Rio de Janeiro.

Rosenfeld, M. J., Porter, P. C., and Cox, J. A., 1998, "Strain Estimation Using Vetco Deformation Tool Data," ASME Paper No. IPC1998-2047.

Noronha, D. B., Martins, R. R., Jacob, B. P., and de Souza, E., 2010, "Procedures for the Strain Based Assessment of Pipeline Dents," Int. J. Pressure Vessels Piping, 87(5), pp. 254–265.

Woo, J., Muntaseer, K., and Adeeb, S., 2017, "Development of a Profile

Matching Criteria to Model Dents in Pipelines Using Finite Element Analysis," ASME Paper No. PVP2017-65278.

Okoloekwe, C., Kainat, M., Langer, D., Hassanien, S., Roger Cheng, J., and Adeeb, S. (May 14, 2018). "Three-Dimensional Strain-Based Model for the Severity Characterization of Dented

Pipelines." *ASME.* ASME J Nondestructive Evaluation. *August 2018; 1(3):* 031006. https://doi.org/10.1115/1.4040039

J. Woo, M. Kainat, C. Okoloekwe, S. Hassanien and S. Adeeb. Integrity Analysis of Dented Pipelines using Artificial Neural Networks. Pipeline Science and Technology. 2019; 3(2): 92–104.

APPENDIX A: Python Scripts for Model Generation and Results Extraction

There are Python code examples in the appendix that can be run in Abaqus to automatically generate FEA model, extract profiles from FEA output files, and extract the maximum, LE11, and LE33 from FEA output files.

The code for the first category:

```
from abaqus import *
from abaqusConstants import *
import __main__


import regionToolset
import section
import regionToolset
import displayGroupMdbToolset as dgm
import part
import material
import assembly
import step
import interaction
import load
import mesh
import optimization
import job
import sketch
import visualization
import xyPlot

import displayGroupOdbToolset as dgo

import connectorBehavior

import sketch

import sys, math

from numpy import *


# File location of Input Variables text file

file_path = 'D:/To Mahyar5/'

input_file = open(file_path + 'Input_Variables.txt')


for line in input_file:


    # Read input variables from file - each row of the input file
is a new model

    extracted_line = line

    extracted_list = extracted_line.split()

    modelname = '' + str(extracted_list[0])

    line_number = str(extracted_list[1])

    segment = str(extracted_list[2])

    feature_ID = str(extracted_list[3])

    outer_radius = float(extracted_list[4])

    wall_thickness = float(extracted_list[5])

    length_of_pipe = float(extracted_list[6])/2

    steel_grade = '' + str(extracted_list[7])

    length_of_partition = float(extracted_list[8])
    radius_of_indenter_l = float(extracted_list[9])

    radius_of_indenter_c = float(extracted_list[10])

    indentation_depth = float(extracted_list[11])

    restrained_or_unrestrained = float(extracted_list[12])

    max_op_pressure = float(extracted_list[13])

    op_pressure_ILI = float(extracted_list[14])

    indentation_dist_1 = float(extracted_list[15])

    indentation_dist_2 = float(extracted_list[16])

    shellMesh_1 = float(extracted_list[17])

    shellMesh_2 = float(extracted_list[18])

    solid1Mesh_thick = int(extracted_list[19])

    solid1Mesh_esz = float(extracted_list[20])

    solid2Mesh_thick = int(extracted_list[21])

    solid2Mesh_esz1 = float(extracted_list[22])

    solid2Mesh_esz2 = float(extracted_list[23])

    solid3Mesh_thick = int(extracted_list[24])

    solid3Mesh_esz1 = float(extracted_list[25])

    solid3Mesh_esz2 = float(extracted_list[26])


#
****************************************************************
****************************************************************
****************************************************************
***********


    # MODEL
    # Create the model

    Model = mdb.Model(name=modelname)


#
****************************************************************
****************************************************************
****************************************************************
***********


    # MATERIALS


    # Create 1: X48 Mean material

    X48Mean = Model.Material(name='X48-Mean')

    X48Mean.Elastic(table=((210000,0.3), ))

    X48Mean.Plastic(table=((250, 0), (300, 0.0002849), (325,
0.0008025), (350, 0.0020044), (364, 0.0032307), (380, 0.0054378),
(400, 0.010094), (420, 0.018156), (440, 0.0317562), (460,
0.0541619), (480, 0.0902838), (500, 0.1473745), (520, 0.2359729),
(540, 0.3711686), (560, 0.5742716)))


    # Create 2: X48-2SD material

    X48Minus2SD = Model.Material(name='X48-2SD')

    X48Minus2SD.Elastic(table=((210000,0.3), ))

    X48Minus2SD.Plastic(table=((250, 0), (300,
0.000705970961777676), (325, 0.00198858019837109), (338.516,
0.00329883738975704), (350, 0.00496687543845044), (360,
0.00700047897053932), (370, 0.00976036237883449), (380,
0.0134751561461588), (400, 0.0250132234118651), (420,
0.0449911312849774), (440, 0.0786929187369147), (460,
0.134214963714089), (480, 0.223726383436636), (500,
0.365198888944934),     (520,     0.58474878843564),     (540,
0.919768321764559), (560, 1.42306437177443)))


    # Create 3: X48+2SD material

    X48Plus2SD = Model.Material(name='X48+2SD')

    X48Plus2SD.Elastic(table=((210000,0.3), ))

    X48Plus2SD.Plastic(table=((250,          0),          (300,
0.000121781127099019),   (325,   0.000343033284647601),   (350,
0.000856794006841056),  (370,  0.00168367821871355),  (389.484,
0.00312993030620379),    (400,    0.00431482129492429),    (420,
0.00776104255555761),    (440,    0.0135746551308026),    (460,
0.0231522972467571),     (480,     0.0385931612089181),     (500,
0.0629973960954917),     (520,     0.100870107102088),     (540,
0.158661515783046),      (560,      0.245480894416341),      (580,
0.374030182526673)))


    # Create 4: X52 (Modern) Mean material

    X52ModernMean = Model.Material(name='X52-Modern-Mean')

    X52ModernMean.Elastic(table=((210000,0.3), ))

    X52ModernMean.Plastic(table=((250,         0),          (300,
0.000108749400313421),   (350,   0.000765108984090265),   (386,
0.00250790249168916),    (400,    0.00385309480590589),    (425,
0.00799018025619187),    (450,    0.0158785213016368),    (475,
0.0303922268704767),     (500,     0.0562560817910754),     (525,
0.101038773636097),      (550,      0.176585060648352),      (575,
0.301043870179888), (600, 0.501693625474847)))


    # Create 5: X52 (Modern)-2SD material

    X52ModernMinus2SD = Model.Material(name='X52-Modern-2SD')

    X52ModernMinus2SD.Elastic(table=((210000,0.3), ))
    X52ModernMinus2SD.Plastic(table=((250, 0), (300,
0.000379379749836455), (350, 0.00266913522415045), (355.768,
0.00325794158634255), (375, 0.00617017858484425), (400,
0.0134417857877633), (425, 0.0278742924375295), (450,
0.0553933117959858), (475, 0.10602537019849), (500,
0.196253203927263), (525, 0.352480699253238), (550,
0.616029108579458), (575, 1.05021221109713), (600,
1.75019265925738)))


    # Create 6: X52 (Modern)+2SD material

    X52ModernPlus2SDMaterial = Model.Material(name='X52-
Modern+2SD')

    X52ModernPlus2SDMaterial.Elastic(table=((210000,0.3), ))

    X52ModernPlus2SDMaterial.Plastic(table=((300, 0), (350,
0.000208969884082084), (375, 0.000528485497630174), (400,
0.00119211396421991), (425, 0.00250926737883032), (430.232,
0.00291227908278148), (450, 0.00502073480401995), (475,
0.00964156680721831), (500, 0.0178760268475106), (525,
0.0321338119930988), (550, 0.0561860224429583), (575,
0.0958108616982069), (600, 0.159693156009829), (625,
0.260658912802615), (650, 0.417346755971966)))


    # Create 7: X52 (Vintage) Mean material

    X52VintageMean = Model.Material(name='X52-Vintage-Mean')

    X52VintageMean.Elastic(table=((210000,0.3), ))

    X52VintageMean.Plastic(table=((300, 0), (325,
0.000243343470806226), (350, 0.000808401541595963), (375,
0.00204445005495397), (386.55, 0.00300842511484258), (400,
0.00461170168451115), (425, 0.00970711940734048), (450,
0.0194227497102881), (475, 0.0372984724789057), (500,
0.069153542026477), (525, 0.1243098892775), (550,
0.217356043233442), (575, 0.370645026859678), (600,
0.617774154719637)))
```

```
    # Create 8: X52 (Vintage)-2SD material

    X52VintageMinus2SD = Model.Material(name='X52-Vintage-2SD')

    X52VintageMinus2SD.Elastic(table=((210000,0.3), ))

    X52VintageMinus2SD.Plastic(table=((250, 0), (300,
0.000592760847717716), (325, 0.00166968975773868), (343.29,
0.0032904053042160 9), (350, 0.00417038299704368), (375,
0.00964057857621192), (400, 0.0210020812703612), (425,
0.0435521116293746), (450, 0.086549127812528), (475,
0.165659048342813), (500, 0.306635279235096), (525,
0.550732500043958)))


    # Create 9: X52 (Vintage)+2SD material

    X52VintagePlus2SD = Model.Material(name='X52-Vintage+2SD')

    X52VintagePlus2SD.Elastic(table=((210000,0.3), ))

    X52VintagePlus2SD.Plastic(table=((300, 0), (350,
0.000211589268884013), (386, 0.000773409973880419), (400,
0.00120705681215113), (425, 0.00254072041267275), (429.81,
0.00291379978767979), (450, 0.00508366844873115), (475,
0.00976242141587416), (500, 0.0181000983363231), (525,
0.0325366012233866), (550, 0.0568903000660922), (575,
0.0970118266893791), (600, 0.161694869451273), (625,
0.263926206545354), (650, 0.422578092317531)))


    # Create 10: X70 Mean material

    X70Mean = Model.Material(name='X70-Mean')

    X70Mean.Elastic(table=((210000,0.3), ))

    X70Mean.Plastic(table=((400, 0), (450, 0.000256613102790624),
(500, 0.00111823835026017), (531, 0.00238891341903746), (550,
0.00368596371377681), (575, 0.00634181640181685), (600,
0.010623523690843), (625, 0.017390748028355), (650,
0.0278927421634957), (675, 0.0439181340326769), (700,
0.0679928987112506), (750, 0.155710854525954), (800,
0.337899549041618), (850, 0.699503176923773)))
```

```
    # Create 11: X70-2SD material

    X70Minus2SD = Model.Material(name='X70-2SD')

    X70Minus2SD.Elastic(table=((210000,0.3), ))

    X70Minus2SD.Plastic(table=((350,          0),          (400,
0.00016864952235037),    (425,    0.000394595341202337),    (450,
0.00082541498423713),   (475,   0.00161807714428539),   (493.84,
0.0026058444457115),    (500,    0.00303062468504242),    (525,
0.00547641946035201),    (550,    0.00960235978363681),    (575,
0.0163996444834247),     (600,    0.0273580773723157),    (625,
0.0446778446382285),     (650,    0.0715562345329387),    (675,
0.112570990905168),      (700,     0.174186994811403),     (750,
0.398688873865535),      (800,     0.864975438358521),     (850,
1.79044951237618)))


    # Create 12: X7+2SD material

    X70Plus2SD = Model.Material(name='X70+2SD')

    X70Plus2SD.Elastic(table=((210000,0.3), ))

    X70Plus2SD.Plastic(table=((450,          0),          (475,
0.000127696364806613),    (500,    0.000355255084223991),    (525,
0.000749267965892302),   (550,   0.00141394909645165),   (568.16,
0.00215465085781429),    (575,    0.00250897871999982),    (600,
0.00427436139425718),    (625,    0.00706454292508614),    (650,
0.011394600362419),     (675,     0.0180019995245585),     (700,
0.027928220440992),     (750,     0.0640950457196469),     (800,
0.139212923480197), (850, 0.28830503476204)))


    # Create 13: X46 Mean material

    X46Mean = Model.Material(name='X46-Mean')

    X46Mean.Elastic(table=((210000,0.3), ))

    X46Mean.Plastic(table=((239,  0.0),  (250,  0.00005),  (278,
0.00018), (300, 0.00047), (325, 0.00126), (350, 0.003095), (364,
0.004967), (380, 0.008337), (400, 0.015446), (420, 0.027755), (440, 0.04852), (460, 0.08273), (480, 0.13788), (500, 0.225044), (520, 0.3603156), (540, 0.566731), (560, .876826),(580, 1.33597)))


    # Create 14: X46-2SD material

    X46Minus2SD = Model.Material(name='X46-2SD')

    X46Minus2SD.Elastic(table=((210000,0.3), ))

    X46Minus2SD.Plastic(table=((230,  0),(250,  0.00012),  (300, 0.00119), (325, 0.00314),(338, 0.00504), (350, 0.00767), (360, 0.01077), (370, 0.01497), (380, 0.02062), (400, 0.03818), (420, 0.06859), (440, 0.11988), (460, 0.20438), (480, 0.34061), (500, 0.55592), (520, 0.89006), (540, 1.4),(550, 1.74475)))


    # Create 15: X46+2SD material

    X46Plus2SD = Model.Material(name='X46+2SD')

    X46Plus2SD.Elastic(table=((210000,0.3), ))

    X46Plus2SD.Plastic(table=((250, 0.0), (300, 0.00019), (325, 0.000529),(340, 0.000923), (364, 0.002119), (380, 0.0035638), (400, 0.006612), (420, 0.01189), (440, 0.020795), (460, 0.035464), (480, 0.0591135), (500, 0.0965), (520, 0.1545), (540, 0.24301), (560, 0.37599), (580, .57287),(600, .8604853),(610, 1.04927),(625, 1.4044)))


#
*****************************************************************
*****************************************************************
*****************************************************************
***********


    # SECTIONS
    Model.HomogeneousSolidSection(name='Solid Section', material=steel_grade, thickness=None)

    Model.HomogeneousShellSection(name='Shell Section', preIntegrate=OFF, material=steel_grade, thicknessType=UNIFORM, thickness=wall_thickness, thicknessField='', nodalThicknessField='', idealization=NO_IDEALIZATION, poissonDefinition=DEFAULT, thicknessModulus=None, temperature=GRADIENT, useDensity=OFF, integrationRule=SIMPSON, numIntPts=9)


#
*******************************************************************************************************************************************************************************************************************


    # PARTS


    # Basic Calculations

    ang_rad = length_of_partition/outer_radius

    ang_deg = ang_rad*(math.pi/180)

    r1 = outer_radius

    r2 = outer_radius-wall_thickness


    # Create Pipe Shell Part

    pipeShellSketch = Model.ConstrainedSketch(name='Pipe Shell Sketch',sheetSize=outer_radius*2)

    pipeShellSketchg = pipeShellSketch.geometry

    midRadius = outer_radius-(0.5*wall_thickness)
    createSShellCircle                                        =
pipeShellSketch.CircleByCenterPerimeter(center=(0.0,0.0),
point1=(0.0,midRadius))

    pipeShellSketch.ConstructionLine(point1=(0.0,0.0),
angle=90.0)

    trimShellCurve1 = pipeShellSketchg.findAt((midRadius,0),)

    pipeShellSketch.autoTrimCurve(curve1=trimShellCurve1,
point1=(outer_radius,0))

    pipeShellPart    =    Model.Part(name='Pipe    Shell    Part',
dimensionality=THREE_D,type=DEFORMABLE_BODY)

    pipeShellPart.BaseShellExtrude(sketch=pipeShellSketch,
depth=length_of_pipe)



pipeShellPart.DatumPlaneByPrincipalPlane(principalPlane=XYPLANE,
offset=length_of_partition)


pipeShellPart.DatumPlaneByPrincipalPlane(principalPlane=XZPLANE,
offset=midRadius*math.cos(ang_rad))


    f = pipeShellPart.faces

    pickedFaces = f.getSequenceFromMask(mask=('[#1 ]', ), )

    d1 = pipeShellPart.datums

    pipeShellPart.PartitionFaceByDatumPlane(datumPlane=d1[2],
faces=pickedFaces)

    f = pipeShellPart.faces

    pickedFaces = f.getSequenceFromMask(mask=('[#1 ]', ), )

    d2 = pipeShellPart.datums

    pipeShellPart.PartitionFaceByDatumPlane(datumPlane=d2[3],
faces=pickedFaces)
```

```
    f = pipeShellPart.faces

    pipeShellPart.RemoveFaces(faceList          =          f[1:2],
deleteCells=False)

    e1 = pipeShellPart.edges

    pipeShellPart.RemoveRedundantEntities(edgeList = e1[3:4])

    v1 = pipeShellPart.vertices

    pipeShellPart.RemoveRedundantEntities(vertexList = v1[4:5])

    # pipeShellPart.checkGeometry()

    session.viewports['Viewport:
1'].partDisplay.geometryOptions.setValues(datumPlanes=OFF)


    # Set-up pipe sketch

    pipeSolidSketch  =  Model.ConstrainedSketch(name='Pipe  Solid
Sketch',sheetSize=outer_radius*2)

    pipeSolidSketchg = pipeSolidSketch.geometry


    # Sketch the pipe section using circle tool

    createSolidCircle1                                              =
pipeSolidSketch.CircleByCenterPerimeter(center=(0.0,0.0),
point1=(0.0,outer_radius))

    createSolidCircle2                                              =
pipeSolidSketch.CircleByCenterPerimeter(center=(0.0,0.0),
point1=(0.0,outer_radius-wall_thickness))



    # Createconstruction lines

    pipeSolidSketch.ConstructionLine(point1=(0.0,0.0),
angle=90.0)
```

```
    pipeSolidSketch.ConstructionLine(point1=(0.0,0.0),  point2=(-
r1*math.sin(ang_rad), r1*math.cos(ang_rad)))


    # Auto-trim the circle so semi-circle (left-hand side) remains

    trimSolidCurve1 = pipeSolidSketchg.findAt((r1, 0.0),)

    trimSolidCurve2 = pipeSolidSketchg.findAt((r2, 0.0),)

    pipeSolidSketch.autoTrimCurve(curve1=trimSolidCurve1,
point1=(r1, 0.0))

    pipeSolidSketch.autoTrimCurve(curve1=trimSolidCurve2,
point1=(r2, 0.0))


    trimSolidCurve3                                          =
pipeSolidSketchg.findAt((r1*math.sin(0.018),                 -
r1*math.cos(0.018)),)

    trimSolidCurve4                                          =
pipeSolidSketchg.findAt((r2*math.sin(0.018),                 -
r2*math.cos(0.018)),)

    pipeSolidSketch.autoTrimCurve(curve1=trimSolidCurve3,
point1=(r1*math.sin(0.018), -r1*math.cos(0.018)))

    pipeSolidSketch.autoTrimCurve(curve1=trimSolidCurve4,
point1=(r2*math.sin(0.018), -r2*math.cos(0.018)))


    trimSolidCurve5         =         pipeSolidSketchg.findAt((-
r1*math.sin(ang_rad+0.15), r1*math.cos(ang_rad+0.15)),)

    trimSolidCurve6         =         pipeSolidSketchg.findAt((-
r2*math.sin(ang_rad+0.15), r2*math.cos(ang_rad+0.15)),)

    pipeSolidSketch.autoTrimCurve(curve1=trimSolidCurve5,
point1=(-r1*math.sin(ang_rad+0.15), r1*math.cos(ang_rad+0.15)))

    pipeSolidSketch.autoTrimCurve(curve1=trimSolidCurve6,
point1=(-r2*math.sin(ang_rad+0.15), r2*math.cos(ang_rad+0.15)))
```

```
    # Add connecting lines to close the sketch

    pipeSolidSketch.Line(point1 =(0.0, r1), point2=(0.0, r2))

    pipeSolidSketch.Line(point1         =(-r1*math.sin(ang_rad),
r1*math.cos(ang_rad)),           point2=(-r2*math.sin(ang_rad),
r2*math.cos(ang_rad)))

    #              pipeSolidSketch.Line(point1                =(-
(outer_radius*math.sin(ang_rad)),
outer_radius*math.cos(ang_rad)),        point2=(-((outer_radius-
wall_thickness)*math.sin(ang_rad)),               (outer_radius-
wall_thickness)*math.cos(ang_rad)))


    # Create a 3D deformable part named "Pipe" by extruding the
sketch

    SolidPart        =        Model.Part(name='Solid        Part',
dimensionality=THREE_D,type=DEFORMABLE_BODY)

    SolidPart.BaseSolidExtrude(sketch=pipeSolidSketch,
depth=length_of_partition)


    # Cone 1 Generation

    cone_ang_1 = math.atan(indentation_dist_1/outer_radius)

    firstConeSketch  =  Model.ConstrainedSketch(name='First  Cone
Sketch',sheetSize=(outer_radius+20)*2)

    firstConeSketchg = firstConeSketch.geometry


    firstConeSketch.setPrimaryObject(option=STANDALONE)

    firstConeSketch.ConstructionLine(point1=(0.0,               -
(outer_radius+20)), point2=(0.0, (outer_radius+20)))

    firstConeSketch.FixedConstraint(entity=firstConeSketchg[2])
    firstConeSketch.Line(point1=(0.0,      0.0),      point2=(0.0,
(outer_radius+20)))

    firstConeSketch.Line(point1=(0.0,         (outer_radius+20)),
point2=((outer_radius+20)*math.tan(cone_ang_1),
(outer_radius+20)))

firstConeSketch.Line(point1=((outer_radius+20)*math.tan(cone_ang
_1), (outer_radius+20)), point2=(0.0, 0.0))


    FirstConePart      =      Model.Part(name='First      Cone',
dimensionality=THREE_D, type=DEFORMABLE_BODY)

    FirstConePart.BaseSolidRevolve(sketch=firstConeSketch,
angle=360.0, flipRevolveDirection=OFF)

    f1 = FirstConePart.faces

    FirstConePart.RemoveFaces(faceList         =         f1[0:1],
deleteCells=False)


    # Cone 2 Generation

    cone_ang_2 = math.atan(indentation_dist_2/outer_radius)

    secondConeSketch = Model.ConstrainedSketch(name='Second Cone
Sketch',sheetSize=(outer_radius+20)*2)

    secondConeSketchg = secondConeSketch.geometry


    secondConeSketch.setPrimaryObject(option=STANDALONE)

    secondConeSketch.ConstructionLine(point1=(0.0,              -
(outer_radius+20)), point2=(0.0, (outer_radius+20)))

secondConeSketch.FixedConstraint(entity=secondConeSketchg[2])
```

```
    secondConeSketch.Line(point1=(0.0,     0.0),     point2=(0.0,
(outer_radius+20)))

    secondConeSketch.Line(point1=(0.0,        (outer_radius+20)),
point2=((outer_radius+20)*math.tan(cone_ang_2),
(outer_radius+20)))


secondConeSketch.Line(point1=((outer_radius+20)*math.tan(cone_an
g_2), (outer_radius+20)), point2=(0.0, 0.0))


    secondConePart      =      Model.Part(name='Second      Cone',
dimensionality=THREE_D, type=DEFORMABLE_BODY)

    secondConePart.BaseSolidRevolve(sketch=secondConeSketch,
angle=360.0, flipRevolveDirection=OFF)

    f2 = secondConePart.faces

    secondConePart.RemoveFaces(faceList         =         f2[0:1],
deleteCells=False)




    rootAssembly1 = Model.rootAssembly

    rootAssembly1.Instance(name='First                   Cone-1',
part=FirstConePart, dependent=ON)

    rootAssembly1.Instance(name='Second                  Cone-1',
part=secondConePart, dependent=ON)

    rootAssembly1.Instance(name='Solid  Part-1',  part=SolidPart,
dependent=ON)

    rootAssembly1.InstanceFromBooleanMerge(name='Pipe       Solid
Part',    instances=(rootAssembly1.instances['Solid    Part-1'],
rootAssembly1.instances['First                         Cone-1'],
rootAssembly1.instances['Second           Cone-1'],           ),
keepIntersections=ON, originalInstances=DELETE, domain=GEOMETRY)
```

```
    pipeSolidPart = Model.parts['Pipe Solid Part']
    f3 = pipeSolidPart.faces
    pipeSolidPart.RemoveFaces(faceList     =     f3[0:1]+f3[6:7],
deleteCells=False)
    del rootAssembly1.features['Pipe Solid Part-1']


    pipeSolidPart_1   =   Model.Part(name='Pipe   Solid   Part-1',
objectToCopy=Model.parts['Pipe Solid Part'])
    pipeSolidPart_2   =   Model.Part(name='Pipe   Solid   Part-2',
objectToCopy=Model.parts['Pipe Solid Part'])
    pipeSolidPart_3   =   Model.Part(name='Pipe   Solid   Part-3',
objectToCopy=Model.parts['Pipe Solid Part'])


    del Model.parts['Solid Part']
    del Model.parts['Pipe Solid Part']
    del Model.parts['First Cone']
    del Model.parts['Second Cone']


    f1 = pipeSolidPart_1.faces
    pipeSolidPart_1.RemoveFaces(faceList                         =
f1[0:1]+f1[1:5]+f1[7:9]+f1[11:13]+f1[14:16], deleteCells=False)
    f1 = pipeSolidPart_2.faces
    pipeSolidPart_2.RemoveFaces(faceList                         =
f1[1:2]+f1[3:4]+f1[6:12]+f1[13:15], deleteCells=False)
    f1 = pipeSolidPart_3.faces
    pipeSolidPart_3.RemoveFaces(faceList                         =
f1[2:3]+f1[4:7]+f1[9:11]+f1[12:14]+f1[15:16], deleteCells=False)
```

```
    # Assign sections to shell part

    f = pipeShellPart.faces

    faces = f.getSequenceFromMask(mask=('[#1 ]', ), )

    region = pipeShellPart.Set(faces=faces, name='Set-1')

    pipeShellPart.SectionAssignment(region=region,
sectionName='Shell              Section',              offset=0.0,
offsetType=MIDDLE_SURFACE,                        offsetField='',
thicknessAssignment=FROM_SECTION)


    # Assign sections to solid parts

    c = pipeSolidPart_1.cells

    cells = c.getSequenceFromMask(mask=('[#1 ]', ), )

    region = pipeSolidPart_1.Set(cells=cells, name='Set-1')

    pipeSolidPart_1.SectionAssignment(region=region,
sectionName='Solid              Section',              offset=0.0,
offsetType=MIDDLE_SURFACE,                        offsetField='',
thicknessAssignment=FROM_SECTION)


    c = pipeSolidPart_2.cells

    cells = c.getSequenceFromMask(mask=('[#1 ]', ), )

    region = pipeSolidPart_2.Set(cells=cells, name='Set-1')

    pipeSolidPart_2.SectionAssignment(region=region,
sectionName='Solid              Section',              offset=0.0,
offsetType=MIDDLE_SURFACE,                        offsetField='',
thicknessAssignment=FROM_SECTION)


    c = pipeSolidPart_3.cells

    cells = c.getSequenceFromMask(mask=('[#1 ]', ), )

    region = pipeSolidPart_3.Set(cells=cells, name='Set-1')
    pipeSolidPart_3.SectionAssignment(region=region,
sectionName='Solid Section', offset=0.0,
offsetType=MIDDLE_SURFACE, offsetField='',
thicknessAssignment=FROM_SECTION)



    # Ring Indenter

    # Re-define input variables ("longitudinal" radius variable
must always be larger than "circumferential" radius variable for
geometry to work)


    if radius_of_indenter_l < radius_of_indenter_c:

        radius_of_indenter_l = float(extracted_list[10])

        radius_of_indenter_c = float(extracted_list[9])


    else:

        radius_of_indenter_l = float(extracted_list[9])

        radius_of_indenter_c = float(extracted_list[10])


    # Set-up the indenter sketch

    indenterSketch = Model.ConstrainedSketch(name='Indenter
Sketch-1', sheetSize = radius_of_indenter_l*2)

    indenterSketchg = indenterSketch.geometry


    # Define geometry of indenter

    indenterSketch.ConstructionLine(point1=(0.0, -
radius_of_indenter_l), point2=(0.0, radius_of_indenter_l))
```

```
indenterSketch.FixedConstraint(entity=indenterSketchg.findAt((0,
radius_of_indenter_l),))


    # Sketch circle based on radius of indenter in longitudinal
and circumferential direction

indenterSketch.CircleByCenterPerimeter(center=(radius_of_indente
r_l - radius_of_indenter_c,0.0), point1=(radius_of_indenter_l -
radius_of_indenter_c,radius_of_indenter_c))


    # Set up center point of circle

    centerX = radius_of_indenter_l - radius_of_indenter_c


    # Use if statement because different curves will have to be
trimmed based on size of indenter


    if centerX >= radius_of_indenter_c:


        # If the difference between the two radii is greater than
the radius of the shorter side

        # Use construction line and auto-trim so only the right
side of the circle and less than half remains

        indenterSketch.ConstructionLine(point1=(centerX + 5,0.0),
angle=90.0)


        # Trim excess parts of circle

        trimLeft        =        indenterSketchg.findAt((centerX-
radius_of_indenter_c,0),)
```

```
        indenterSketch.autoTrimCurve(curve1=trimLeft, 
point1=(centerX-radius_of_indenter_c, 0))


        trimTop                                                 = 
indenterSketchg.findAt((centerX+radius_of_indenter_c*math.sin(0.
00001),radius_of_indenter_c*math.cos(0.00001)),)

        indenterSketch.autoTrimCurve(curve1=trimTop, 
point1=(centerX+radius_of_indenter_c*math.sin(0.00001),radius_of
_indenter_c*math.cos(0.00001)))


    elif centerX == 0:


        # If longitudinal radius = circumferential radius:


indenterSketch.ConstructionLine(point1=(radius_of_indenter_l/2,0
.0), angle=90.0)


        # Trim excess parts of circle

        trimLeft            =            indenterSketchg.findAt((-
radius_of_indenter_l,0),)

        indenterSketch.autoTrimCurve(curve1=trimLeft,   point1=(-
radius_of_indenter_l, 0))


        trimTop                                                 = 
indenterSketchg.findAt((centerX+radius_of_indenter_c*math.sin(0.
00001),radius_of_indenter_c*math.cos(0.00001)),)

        indenterSketch.autoTrimCurve(curve1=trimTop, 
point1=(centerX+radius_of_indenter_c*math.sin(0.00001),radius_of
_indenter_c*math.cos(0.00001)))
```

```
        trimBottom                                                  =
indenterSketchg.findAt((centerX+radius_of_indenter_c*math.sin(0.
00001),-radius_of_indenter_c*math.cos(0.00001)),)

        indenterSketch.autoTrimCurve(curve1=trimBottom,
point1=(centerX+radius_of_indenter_c*math.sin(0.00001),-
radius_of_indenter_c*math.cos(0.00001)))


    else:


        # If the difference between the two radii is less than the
radius of the shorter side

        # Create construction line 5 mm to the right of centerX

        indenterSketch.ConstructionLine(point1=(centerX + 5,0.0),
angle=90.0)


        trimLeft        =        indenterSketchg.findAt((centerX-
radius_of_indenter_c,0),)

        indenterSketch.autoTrimCurve(curve1=trimLeft,
point1=(centerX-radius_of_indenter_c, 0))


        trimTop                                                     =
indenterSketchg.findAt((centerX+radius_of_indenter_c*math.sin(-
0.00001),radius_of_indenter_c*math.cos(-0.00001)),)

        indenterSketch.autoTrimCurve(curve1=trimTop,
point1=(centerX+radius_of_indenter_c*math.sin(-
0.00001),radius_of_indenter_c*math.cos(-0.00001)))


        trimTop2                                                    =
indenterSketchg.findAt((centerX+radius_of_indenter_c*math.sin(0.
00001),radius_of_indenter_c*math.cos(0.00001)),)
```

```
        indenterSketch.autoTrimCurve(curve1=trimTop2,
point1=(centerX+radius_of_indenter_c*math.sin(0.00001),radius_of
_indenter_c*math.cos(0.00001)))


        trimBottom      =      indenterSketchg.findAt((centerX,-
radius_of_indenter_c),)

        indenterSketch.autoTrimCurve(curve1=trimBottom,
point1=(centerX,-radius_of_indenter_c))


    # Create a 3D analytical rigid part named "Indenter" by
revolving the sketch


    indenterPart=Model.Part(name='Indenter',
dimensionality=THREE_D, type=ANALYTIC_RIGID_SURFACE)

    indenterPart.AnalyticRigidSurfRevolve(sketch=indenterSketch)


    # Insert reference point on ring indenter

    RP1          =          indenterPart.ReferencePoint(point=(-
radius_of_indenter_l,0,0))

    RP1ID = RP1.id


#
****************************************************************
****************************************************************
****************************************************************
***********


    # MESHING
    meshDim_1                                                        =
int(round(0.5*math.pi*indentation_dist_1/solid1Mesh_esz))

    # Meshing Pipe Solid Part 1

    e = pipeSolidPart_1.edges

    pickedEdges = e.getSequenceFromMask(mask=('[#5 ]', ), )

    pipeSolidPart_1.seedEdgeByNumber(edges=pickedEdges,
number=meshDim_1, constraint=FIXED)

    pickedEdges = e.getSequenceFromMask(mask=('[#1d0 ]', ), )

    pipeSolidPart_1.seedEdgeBySize(edges=pickedEdges,
size=solid1Mesh_esz, deviationFactor=0.05, constraint=FINER)

    pickedEdges = e.getSequenceFromMask(mask=('[#2a ]', ), )

    pipeSolidPart_1.seedEdgeByNumber(edges=pickedEdges,
number=solid1Mesh_thick, constraint=FINER)

    c = pipeSolidPart_1.cells

    pickedRegions = c.getSequenceFromMask(mask=('[#1 ]', ), )

    elemType1           =           mesh.ElemType(elemCode=C3D20,
elemLibrary=STANDARD)

    pipeSolidPart_1.setMeshControls(regions=pickedRegions,
technique=STRUCTURED)

    pickedRegions1 =(pickedRegions, )

    pipeSolidPart_1.setElementType(regions=pickedRegions1,
elemTypes=(elemType1, ))

    pipeSolidPart_1.generateMesh()



    # Pipe Solid Part 2

    e = pipeSolidPart_2.edges

    pickedEdges = e.getSequenceFromMask(mask=('[#300 ]', ), )
    pipeSolidPart_2.seedEdgeByNumber(edges=pickedEdges,
number=meshDim_1, constraint=FIXED)

    pickedEdges = e.getSequenceFromMask(mask=('[#5 ]', ), )

    pipeSolidPart_2.seedEdgeByNumber(edges=pickedEdges,
number=meshDim_1, constraint=FIXED)

    pickedEdges = e.getSequenceFromMask(mask=('[#8d0 ]', ), )

    pipeSolidPart_2.seedEdgeByBias(biasMethod=SINGLE,
end1Edges=pickedEdges,                  minSize=solid2Mesh_esz1,
maxSize=solid2Mesh_esz2, constraint=FINER)

    pickedEdges = e.getSequenceFromMask(mask=('[#42a ]', ), )

    pipeSolidPart_2.seedEdgeByNumber(edges=pickedEdges,
number=solid2Mesh_thick, constraint=FINER)

    c = pipeSolidPart_2.cells

    pickedRegions = c.getSequenceFromMask(mask=('[#1 ]', ), )

    elemType1          =          mesh.ElemType(elemCode=C3D20R,
elemLibrary=STANDARD)

    pipeSolidPart_2.setMeshControls(regions=pickedRegions,
technique=STRUCTURED)

    pickedRegions1 =(pickedRegions, )

    pipeSolidPart_2.setElementType(regions=pickedRegions1,
elemTypes=(elemType1, ))

    pipeSolidPart_2.generateMesh()


    # Pipe Solid Part 3

    eleDim_1 = (0.5*math.pi*indentation_dist_2)/meshDim_1

    e = pipeSolidPart_3.edges

    pickedEdges = e.getSequenceFromMask(mask=('[#5 ]', ), )

    pipeSolidPart_3.seedEdgeByNumber(edges=pickedEdges,
number=meshDim_1, constraint=FIXED)
```

```
    #           pipeSolidPart_3.seedEdgeBySize(edges=pickedEdges,
size=solid3Mesh_esz1, deviationFactor=0.02, constraint=FINER)

    pickedEdges = e.getSequenceFromMask(mask=('[#1860 ]', ), )

    pipeSolidPart_3.seedEdgeBySize(edges=pickedEdges,
size=eleDim_1, deviationFactor=0.02, constraint=FINER)

    pickedEdges = e.getSequenceFromMask(mask=('[#590 ]', ), )

    pipeSolidPart_3.seedEdgeByBias(biasMethod=SINGLE,
end1Edges=pickedEdges,    minSize=eleDim_1,    maxSize=eleDim_1,
constraint=FINER)

    pickedEdges = e.getSequenceFromMask(mask=('[#620a ]', ), )

    pipeSolidPart_3.seedEdgeByNumber(edges=pickedEdges,
number=solid3Mesh_thick, constraint=FINER)

    c = pipeSolidPart_3.cells

    pickedRegions = c.getSequenceFromMask(mask=('[#1 ]', ), )

    elemType1           =           mesh.ElemType(elemCode=C3D8,
elemLibrary=STANDARD)

    pipeSolidPart_3.setMeshControls(regions=pickedRegions,
technique=SWEEP)

    pickedRegions1 =(pickedRegions, )

    pipeSolidPart_3.setElementType(regions=pickedRegions1,
elemTypes=(elemType1, ))

    pipeSolidPart_3.generateMesh()


    # Pipe Shell Part

    e = pipeShellPart.edges

    pickedEdges = e.getSequenceFromMask(mask=('[#3 ]', ), )

    pipeShellPart.seedEdgeBySize(edges=pickedEdges,
size=shellMesh_1, deviationFactor=0.1, constraint=FINER)

    pickedEdges = e.getSequenceFromMask(mask=('[#24 ]', ), )
    pipeShellPart.seedEdgeByBias(biasMethod=SINGLE,
end1Edges=pickedEdges, minSize=shellMesh_1, maxSize=shellMesh_2,
constraint=FINER)

    pickedEdges = e.getSequenceFromMask(mask=('[#18 ]', ), )

    pipeShellPart.seedEdgeBySize(edges=pickedEdges,
size=shellMesh_2, deviationFactor=0.1, constraint=FINER)

    elemType1 = mesh.ElemType(elemCode=S4, elemLibrary=STANDARD,
secondOrderAccuracy=OFF)

    f = pipeShellPart.faces

    faces = f.getSequenceFromMask(mask=('[#1 ]', ), )

    pickedRegions =(faces, )

    pipeShellPart.setElementType(regions=pickedRegions,
elemTypes=(elemType1, ))

    pipeShellPart.setMeshControls(regions=faces, elemShape=QUAD)

    pipeShellPart.generateMesh()


#
*****************************************************************
*****************************************************************
*****************************************************************
***********


    # ASSEMBLY


    pipeAssembly = Model.rootAssembly

    pipeAssembly.Instance(name='Pipe         Shell         Part',
part=pipeShellPart, dependent=ON)

    pipeAssembly.Instance(name='Pipe     Solid     Part     1',
part=pipeSolidPart_1, dependent=ON)
```

```
    pipeAssembly.Instance(name='Pipe      Solid      Part      2',
part=pipeSolidPart_2, dependent=ON)

    pipeAssembly.Instance(name='Pipe      Solid      Part      3',
part=pipeSolidPart_3, dependent=ON)


    # Set up assembly - position indenter in relation to pipe

    pipeAssembly.Instance(name='Pipe                   Indenter',
part=indenterPart, dependent=ON)

    pipeAssembly.rotate(instanceList=('Pipe     Indenter',     ),
axisPoint=(0.0, 0.0, 0.0), axisDirection=(0.0, 0.0, 1.0), angle=-
90.0)


    #  If  circumferential  radius  is  larger  than  longituindal
radius, the indenter will have to rotated 90 degrees around the y-
axis

    radius_of_indenter_l = float(extracted_list[9])

    radius_of_indenter_c = float(extracted_list[10])

    if radius_of_indenter_c >= radius_of_indenter_l:

        pipeAssembly.rotate(instanceList=('Pipe   Indenter',   ),
axisPoint=(0.0,   0.0,   0.0),   axisDirection=(0.0,   1.0,   0.0),
angle=90.0)

        pipeAssembly.translate(instanceList=('Pipe  Indenter', ),
vector=(0.0, radius_of_indenter_c + outer_radius+2, 0))


    else:

        # Translate the first indenter to its correct relative
position along the pipe segment

        pipeAssembly.translate(instanceList=('Pipe  Indenter', ),
vector=(0.0, radius_of_indenter_l + outer_radius+2,0))
```

```
    #                                session.viewports['Viewport:
1'].assemblyDisplay.geometryOptions.setValues(datumAxes=OFF,
datumPlanes=OFF)

    #                                session.viewports['Viewport:
1'].view.setValues(nearPlane=1546.33,          farPlane=2960.54,
width=1853.66,         height=884.252,         viewOffsetX=154.7,
viewOffsetY=14.2564)


#
****************************************************************
****************************************************************
****************************************************************
***********


    # COUPLING


    # Connecting the Shell and the Solid parts

    s1 = pipeAssembly.instances['Pipe Solid Part 3'].faces

    side1Faces1 = s1.getSequenceFromMask(mask=('[#40 ]', ), )

    region1=pipeAssembly.Surface(side1Faces=side1Faces1,
name='s_Surf-1')

    s1 = pipeAssembly.instances['Pipe Shell Part'].edges

    side1Edges1 = s1.getSequenceFromMask(mask=('[#1 ]', ), )

    region2    =    pipeAssembly.Surface(side1Edges=side1Edges1,
name='m_Surf-1')

    Model.ShellSolidCoupling(name='Shell-Solid     Constraint-1',
shellEdge=region2,                           solidFace=region1,
positionToleranceMethod=COMPUTED)


    s1 = pipeAssembly.instances['Pipe Solid Part 3'].faces
    side1Faces1 = s1.getSequenceFromMask(mask=('[#20 ]', ), )

    region1=pipeAssembly.Surface(side1Faces=side1Faces1,
name='s_Surf-2')

    s1 = pipeAssembly.instances['Pipe Shell Part'].edges

    side1Edges1 = s1.getSequenceFromMask(mask=('[#2 ]', ), )

    region2    =    pipeAssembly.Surface(side1Edges=side1Edges1,
name='m_Surf-2')

    Model.ShellSolidCoupling(name='Shell-Solid     Constraint-2',
shellEdge=region2,                           solidFace=region1,
positionToleranceMethod=COMPUTED)


    # Connecting Solid parts together

    s1 = pipeAssembly.instances['Pipe Solid Part 1'].faces

    side1Faces1 = s1.getSequenceFromMask(mask=('[#1 ]', ), )

    region1=pipeAssembly.Surface(side1Faces=side1Faces1,
name='m_Surf-3')

    s1 = pipeAssembly.instances['Pipe Solid Part 2'].faces

    side1Faces1 = s1.getSequenceFromMask(mask=('[#8 ]', ), )

    region2=pipeAssembly.Surface(side1Faces=side1Faces1,
name='s_Surf-3')

    Model.Tie(name='Solid-Solid  Constraint-1',  master=region1,
slave=region2,   positionToleranceMethod=COMPUTED,   adjust=ON,
tieRotations=ON, thickness=ON)


    s1 = pipeAssembly.instances['Pipe Solid Part 2'].faces

    side1Faces1 = s1.getSequenceFromMask(mask=('[#1 ]', ), )

    region1=pipeAssembly.Surface(side1Faces=side1Faces1,
name='m_Surf-4')

    s1 = pipeAssembly.instances['Pipe Solid Part 3'].faces
    side1Faces1 = s1.getSequenceFromMask(mask=('[#1 ]', ), )

    region2=pipeAssembly.Surface(side1Faces=side1Faces1,
name='s_Surf-4')

    Model.Tie(name='Solid-Solid  Constraint-2',  master=region1,
slave=region2,   positionToleranceMethod=COMPUTED,   adjust=ON,
tieRotations=ON, thickness=ON)


#
****************************************************************
****************************************************************
****************************************************************
***********


    # BOUNDARY CONDITIONS


    # Set up X-Symmetry boundary condition


    e1 = pipeAssembly.instances['Pipe Shell Part'].edges

    edges1 = e1.getSequenceFromMask(mask=('[#28 ]', ), )

    f2 = pipeAssembly.instances['Pipe Solid Part 3'].faces

    faces2 = f2.getSequenceFromMask(mask=('[#10 ]', ), )

    f3 = pipeAssembly.instances['Pipe Solid Part 2'].faces

    faces3 = f3.getSequenceFromMask(mask=('[#2 ]', ), )

    f4 = pipeAssembly.instances['Pipe Solid Part 1'].faces

    faces4 = f4.getSequenceFromMask(mask=('[#10 ]', ), )

    region            =            pipeAssembly.Set(edges=edges1,
faces=faces2+faces3+faces4, name='BC-XSym-Set')

    Model.XsymmBC(name='X-Symmetry',   createStepName='Initial',
region=region, localCsys=None)
```

```
    # Set up Z-Symmetry boundary condition

    f1 = pipeAssembly.instances['Pipe Solid Part 1'].faces

    faces1 = f1.getSequenceFromMask(mask=('[#2 ]', ), )

    f2 = pipeAssembly.instances['Pipe Solid Part 2'].faces

    faces2 = f2.getSequenceFromMask(mask=('[#20 ]', ), )

    f3 = pipeAssembly.instances['Pipe Solid Part 3'].faces

    faces3 = f3.getSequenceFromMask(mask=('[#4 ]', ), )

    e4 = pipeAssembly.instances['Pipe Shell Part'].edges

    edges4 = e4.getSequenceFromMask(mask=('[#4 ]', ), )

    region            =            pipeAssembly.Set(edges=edges4,
faces=faces1+faces2+faces3, name='BC-ZSym-Set')

    Model.ZsymmBC(name='Z-Symmetry',    createStepName='Initial',
region=region, localCsys=None)


    # Set up Bottom boundary condition

    e1 = pipeAssembly.instances['Pipe Shell Part'].edges

    edges1 = e1.getSequenceFromMask(mask=('[#8 ]', ), )

    region  =  pipeAssembly.Set(edges=edges1,  name='BC-FixedBase-
Set')

    Model.DisplacementBC(name='Bottom-Vrtcl-Fix',
createStepName='Initial',   region=region,   u1=UNSET,   u2=SET,
u3=UNSET,   ur1=UNSET,   ur2=UNSET,   ur3=UNSET,   amplitude=UNSET,
distributionType=UNIFORM, fieldName='', localCsys=None)


    # Set up Indenter-Translation boundary condition

    indenterRP           =           pipeAssembly.instances['Pipe
Indenter'].referencePoints
```

```
    indenterRefRegion                                              =
regionToolset.Region(referencePoints=(indenterRP[RP1ID], ))

    Model.DisplacementBC(name='Indenter-Translation',
createStepName='Initial',    region=indenterRefRegion,    u1=SET,
u2=SET,  u3=SET,  ur1=SET,  ur2=SET,  ur3=SET,  amplitude=UNSET,
distributionType=UNIFORM, fieldName='', localCsys=None)


    # Create reference point in center of pipe (on ends away from
indenter)

    RPC2                                                           =
pipeAssembly.ReferencePoint(point=(0,0,length_of_pipe))

    RPCid_2 = RPC2.id


    # Create cylindrical coordinate system for boundary condition

    datumCreate = pipeAssembly.DatumCsysByThreePoints(name='Datum
csys-2',   coordSysType=CYLINDRICAL,   origin=(0.0,   0.0,   0.0),
line1=(1.0, 0.0, 0.0), line2=(0.0, 1.0, 0.0))

    datumID = datumCreate.id

    datumCylind = pipeAssembly.datums[datumID]


    # Select reference point for coupling condition

    refPoints = pipeAssembly.referencePoints

    refPoints2=(refPoints[RPCid_2], )

controlPoint=regionToolset.Region(referencePoints=refPoints2)


    # Select surface for coupling condition

    s1 = pipeAssembly.instances['Pipe Shell Part'].edges

    side1Edges1 = s1.getSequenceFromMask(mask=('[#10 ]', ), )
    couplingFacesRegion                                        =
pipeAssembly.Surface(side1Edges=side1Edges1, name='s_Surf-5')

    datumCoupling = pipeAssembly.datums[datumID]

    # Create coupling boundary condition and allow freedom in the
U1 (radial direction)

    Model.Coupling(name='End-Condition-Coupling-Constraint',
controlPoint=controlPoint,           surface=couplingFacesRegion,
influenceRadius=WHOLE_SURFACE,             couplingType=KINEMATIC,
localCsys=datumCoupling,  u1=OFF,  u2=ON,  u3=ON,  ur1=ON,  ur2=ON,
ur3=ON)


    # Set up fixed ends boundary condition (encastre)

    encastreRP = ( refPoints[RPCid_2], )

    encastreRegion                                             =
regionToolset.Region(referencePoints=encastreRP)

    Model.EncastreBC(name='Fixed-Ends', createStepName='Initial',
region=encastreRegion, localCsys=None)


#
****************************************************************
****************************************************************
****************************************************************
***********


    # INTERACTIONS


    # Set up interaction properties

    Model.ContactProperty('IntProp-1')

    Model.interactionProperties['IntProp-
1'].TangentialBehavior(formulation=PENALTY,
directionality=ISOTROPIC,                slipRateDependency=OFF,
pressureDependency=OFF,                temperatureDependency=OFF,
dependencies=0,   table=((0.5,   ),   ),   shearStressLimit=None,
maximumElasticSlip=FRACTION,                       fraction=0.005,
elasticSlipStiffness=None)

    Model.interactionProperties['IntProp-
1'].NormalBehavior(pressureOverclosure=HARD,  allowSeparation=ON,
constraintEnforcementMethod=DEFAULT)


    # Define master surface - Indenter-1

    s1 = pipeAssembly.instances['Pipe Indenter'].faces

    side1Faces1 = s1.getSequenceFromMask(mask=('[#1 ]', ), )

    masterSurface  =  pipeAssembly.Surface(side1Faces=side1Faces1,
name='m_Surf-7')


    # Define slave surface

    s1 = pipeAssembly.instances['Pipe Solid Part 1'].faces

    side1Faces1 = s1.getSequenceFromMask(mask=('[#4 ]', ), )

    s2 = pipeAssembly.instances['Pipe Solid Part 2'].faces

    side1Faces2 = s2.getSequenceFromMask(mask=('[#10 ]', ), )

    slaveSurface                                                 =
pipeAssembly.Surface(side1Faces=side1Faces1+side1Faces2,
name='s_Surf-7')


    # Update interaction properties

    Model.SurfaceToSurfaceContactStd(name='Int-1',
createStepName='Initial',                   master=masterSurface,
slave=slaveSurface,         sliding=FINITE,         thickness=ON,
interactionProperty='IntProp-1',              adjustMethod=NONE,
initialClearance=OMIT, datumAxis=None, clearanceRegion=None)
```

```
    # Introduce contact controls, initializations and stabilizations

    Model.StdContactControl(name='ContCtrl-1', stabilizeChoice=AUTOMATIC)

    Model.StdInitialization(name='CInit-1')

    Model.StdStabilization(name='CStab-1')


#
**************************************************************************************************************************************************************************************************

    # STEPS & LOADS & OUTPUT SETTINGS


    Model.StaticStep(name='Initial-Pressure', previous='Initial', timePeriod=1, initialInc=1, minInc=1.5e-05, maxInc=1, nlgeom=ON)


    s1 = pipeAssembly.instances['Pipe Solid Part 3'].faces

    side1Faces1 = s1.getSequenceFromMask(mask=('[#8 ]', ), )

    s2 = pipeAssembly.instances['Pipe Solid Part 2'].faces

    side1Faces2 = s2.getSequenceFromMask(mask=('[#4 ]', ), )

    s3 = pipeAssembly.instances['Pipe Solid Part 1'].faces

    side1Faces3 = s3.getSequenceFromMask(mask=('[#8 ]', ), )

    s4 = pipeAssembly.instances['Pipe Shell Part'].faces

    side2Faces4 = s4.getSequenceFromMask(mask=('[#1 ]', ), )

    pressureRegion = pipeAssembly.Surface(side1Faces=side1Faces1+side1Faces2+side1Faces3, side2Faces=side2Faces4, name='Surf-1')
    Model.Pressure(name='Pressure',      createStepName='Initial-
Pressure',    region=pressureRegion,    distributionType=UNIFORM,
field='', magnitude=0.1, amplitude=UNSET)


    # Set up field output and history output requests

    Model.fieldOutputRequests['F-Output-
1'].setValues(variables=('S', 'PEEQ', 'LE', 'EE', 'IE', 'U', 'P',
'CNAREA', 'CSTATUS'), frequency=1)

    Model.historyOutputRequests['H-Output-
1'].setValues(variables=('MASS', ), frequency=LAST_INCREMENT)


    # Set up Indentation-Mean step

    Model.StaticStep(name='Indentation-Mean',  previous='Initial-
Pressure',    maxNumInc=500,    initialInc=0.01,    minInc=1e-05,
maxInc=0.1, nlgeom=ON)

    Model.boundaryConditions['Indenter-
Translation'].setValuesInStep(stepName='Indentation-Mean',   u2=-
(2+indentation_depth))


    # Set up Removal step

    Model.StaticStep(name='Removal', previous='Indentation-Mean',
maxNumInc=500,    initialInc=0.1,    minInc=1e-05,    maxInc=0.1,
nlgeom=ON)

    Model.boundaryConditions['Indenter-
Translation'].setValuesInStep(stepName='Removal', u2=5.0)


    # Set up Pressure-MOP step

    Model.StaticStep(name='Pressure-MOP',     previous='Removal',
timePeriod=1,   initialInc=0.05,   minInc=1.5e-05,   maxInc=0.2,
nlgeom=ON)
```

```
    Model.loads['Pressure'].setValuesInStep(stepName='Pressure-
MOP', magnitude=max_op_pressure)


    # Set up Pressure-Zero

    Model.StaticStep(name='Pressure-Zero',    previous='Pressure-
MOP', initialInc=0.05, minInc=1.5e-05, maxInc=0.2, nlgeom=ON)

    Model.loads['Pressure'].setValuesInStep(stepName='Pressure-
Zero', magnitude=0)


    # Set up P-ILI step

    Model.StaticStep(name='Pressure-ILI',     previous='Pressure-
Zero', timePeriod=1, initialInc=0.05, minInc=1.5e-05, maxInc=0.2,
nlgeom=ON)

    Model.loads['Pressure'].setValuesInStep(stepName='Pressure-
ILI', magnitude=op_pressure_ILI)


    # Suppress removal step if the dent is specified to be
restrained

    if restrained_or_unrestrained == 1:

        Model.steps['Removal'].suppress()


#
****************************************************************
****************************************************************
****************************************************************
***********


    # JOB


    job_ID = modelname
    mdb.Job(name=job_ID,     model=modelname,     description='',
type=ANALYSIS,    atTime=None,    waitMinutes=0,    waitHours=0,
queue=None, memory=90, memoryUnits=PERCENTAGE,

            getMemoryFromAnalysis=True, explicitPrecision=SINGLE,
nodalOutputPrecision=SINGLE,    echoPrint=OFF,    modelPrint=OFF,
contactPrint=OFF,      historyPrint=OFF,      userSubroutine='',
scratch='',

            resultsFormat=ODB,       multiprocessingMode=DEFAULT,
numCpus=6, numDomains=6, numGPUs=1)


input_file.close()
```

APPENDIX B: Mathematica code in the Longitudinal Direction and Result Extraction.

```
Clear[AbaqusDisplacementData,LP0mmX,LP0mmU1,LP0mmU2,LP1mmX,LP1mm
U1,LP1mmU2,LP2mmX,LP2mmU1,LP2mmU2,LP3mmX,LP3mmU1,LP3mmU2,LP4mmX,
LP4mmU1,LP4mmU2,LP5mmX,LP5mmU1,LP5mmU2,LP6mmX,LP6mmU1,LP6mmU2,LP
7mmX,LP7mmU1,LP7mmU2,LP8mmX,LP8mmU1,LP8mmU2,     LE110mm,LE111mm,
LE112mm,LE113mm,     LE114mm,LE115mm,     LE116mm,LE117mm,LE118mm,
PE110mm,PE111mm,         PE112mm,PE113mm,         PE114mm,PE115mm,
PE116mm,PE117mm,PE118mm,LP0mmU1co,LP0mmU2co,LP1mmU1co,LP1mmU2co,
LP2mmU1co,LP2mmU2co,LP3mmU1co,LP3mmU2co,LP4mmU1co,LP4mmU2co,LP5m
mU1co,LP5mmU2co,LP6mmU1co,LP6mmU2co,LP7mmU1co,LP7mmU2co,LP8mmU1c
o,LP8mmU2co];
AbaqusDisplacementData=Import["C:\\Users\\mahya\\Desktop\\Mahyar
Project\\For 2 mm dent depth\\FinalResults2.xlsx"];
Clear[RowIndex];
RowIndex=3;
(*Path at the 0mm depth*)
LP0mmX={};
LP0mmU1={};
LP0mmU2={};
LE110mm={};
PE110mm={};
  (*Path at the 1mm depth*)
LP1mmX={};
LP1mmU1={};
LP1mmU2={};
LE111mm={};
PE111mm={};
(*Path at the 2mm depth*)
LP2mmX={};
LP2mmU1={};
LP2mmU2={};
LE112mm={};
PE112mm={};
(*Path at the 3mm depth*)
LP3mmX={};
LP3mmU1={};
LP3mmU2={};
LE113mm={};
PE113mm={};
(*Path at the 4mm depth*)
LP4mmX={};
LP4mmU1={};
LP4mmU2={};
LE114mm={};
PE114mm={};
(*Path at the 5mm depth*)
LP5mmX={};
LP5mmU1={};
LP5mmU2={};
LE115mm={};
PE115mm={};
```

```
(*Path at the 6mm depth*)
LP6mmX={};
LP6mmU1={};
LP6mmU2={};
LE116mm={};
PE116mm={};
(*Path at the 7mm depth*)
LP7mmX={};
LP7mmU1={};
LP7mmU2={};
LE117mm={};
PE117mm={};
(*Path at the 8mm depth*)
LP8mmX={};
LP8mmU1={};
LP8mmU2={};
LE118mm={};
PE118mm={};

(*Putting the data for Longitudinal Path at different heights in
respective paths*)
While[
  True,
  If[AbaqusDisplacementData[[1]][[RowIndex]][[1]]!="",
    (*Reading all the 0mm depth data*)
    LP0mmX=
AppendTo[LP0mmX,AbaqusDisplacementData[[1]][[RowIndex]][[1]]];
    LP0mmU1                                                    =
AppendTo[LP0mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[2]]];
    LP0mmU2=
AppendTo[LP0mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[3]]];

LE110mm=AppendTo[LE110mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[4]]];

PE110mm=AppendTo[PE110mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[5]]];
    (*Reading all the 1mm depth data*)
    LP1mmX=
AppendTo[LP1mmX,AbaqusDisplacementData[[1]][[RowIndex]][[7]]];
    LP1mmU1                                                    =
AppendTo[LP1mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[8]]];
    LP1mmU2=
AppendTo[LP1mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[9]]];

LE111mm=AppendTo[LE111mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[10]]];
```

```
PE111mm=AppendTo[PE111mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[11]]];
    (*Reading all the 2mm depth data*)
    LP2mmX=
AppendTo[LP2mmX,AbaqusDisplacementData[[1]][[RowIndex]][[13]]];
    LP2mmU1                                                      =
AppendTo[LP2mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[14]]];
    LP2mmU2=
AppendTo[LP2mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[15]]];

LE112mm=AppendTo[LE112mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[16]]];

PE112mm=AppendTo[PE112mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[17]]];
    (*Reading all the 3mm depth data*)
    LP3mmX=
AppendTo[LP3mmX,AbaqusDisplacementData[[1]][[RowIndex]][[19]]];
    LP3mmU1                                                      =
AppendTo[LP3mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[20]]];
    LP3mmU2=
AppendTo[LP3mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[21]]];

LE113mm=AppendTo[LE113mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[22]]];

PE113mm=AppendTo[PE113mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[23]]];
    (*Reading all the 4mm depth data*)
    LP4mmX=
AppendTo[LP4mmX,AbaqusDisplacementData[[1]][[RowIndex]][[25]]];
    LP4mmU1                                                      =
AppendTo[LP4mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[26]]];
    LP4mmU2=
AppendTo[LP4mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[27]]];

LE114mm=AppendTo[LE114mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[28]]];

PE114mm=AppendTo[PE114mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[29]]];
    (*Reading all the 5mm depth data*)
    LP5mmX=
AppendTo[LP5mmX,AbaqusDisplacementData[[1]][[RowIndex]][[31]]];
    LP5mmU1                                                      =
AppendTo[LP5mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[32]]];
```

```
    LP5mmU2=
AppendTo[LP5mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[33]]];

LE115mm=AppendTo[LE115mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[34]]];

PE115mm=AppendTo[PE115mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[35]]];
    (*Reading all the 6mm depth data*)
    LP6mmX=
AppendTo[LP6mmX,AbaqusDisplacementData[[1]][[RowIndex]][[37]]];
    LP6mmU1                                                        =
AppendTo[LP6mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[38]]];
    LP6mmU2=
AppendTo[LP6mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[39]]];

LE116mm=AppendTo[LE116mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[40]]];

PE116mm=AppendTo[PE116mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[41]]];
    (*Reading all the 7mm depth data*)
    LP7mmX=
AppendTo[LP7mmX,AbaqusDisplacementData[[1]][[RowIndex]][[43]]];
    LP7mmU1                                                        =
AppendTo[LP7mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[44]]];
    LP7mmU2=
AppendTo[LP7mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[45]]];

LE117mm=AppendTo[LE117mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[46]]];

PE117mm=AppendTo[PE117mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[47]]];
    (*Reading all the 8mm depth data*)
    LP8mmX=
AppendTo[LP8mmX,AbaqusDisplacementData[[1]][[RowIndex]][[49]]];
    LP8mmU1                                                        =
AppendTo[LP8mmU1,AbaqusDisplacementData[[1]][[RowIndex]][[50]]];
    LP8mmU2=
AppendTo[LP8mmU2,AbaqusDisplacementData[[1]][[RowIndex]][[51]]];

LE118mm=AppendTo[LE118mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[52]]];

PE118mm=AppendTo[PE118mm,AbaqusDisplacementData[[1]][[RowIndex]]
[[53]]];
```

```
        RowIndex=RowIndex+1;
        ,
        Break[];
        ];
      ];

LP0mmU1co=GaussianFilter[LP0mmU1,20];
LP0mmU2co=GaussianFilter[LP0mmU2,5];
LP1mmU1co=GaussianFilter[LP1mmU1,3];
LP1mmU2co=GaussianFilter[LP1mmU2,1];
LP2mmU1co=GaussianFilter[LP2mmU1,3];
LP2mmU2co=GaussianFilter[LP2mmU2,1];
LP3mmU1co=GaussianFilter[LP3mmU1,3];
LP3mmU2co=GaussianFilter[LP3mmU2,1];
LP4mmU1co=GaussianFilter[LP4mmU1,5];
LP4mmU2co=GaussianFilter[LP4mmU2,5];
LP5mmU1co=GaussianFilter[LP5mmU1,3];
LP5mmU2co=GaussianFilter[LP5mmU2,1];
LP6mmU1co=GaussianFilter[LP6mmU1,2];
LP6mmU2co=GaussianFilter[LP6mmU2,1];
LP7mmU1co=GaussianFilter[LP7mmU1,2];
LP7mmU2co=GaussianFilter[LP7mmU2,1];
LP8mmU1co=GaussianFilter[LP8mmU1,2];
LP8mmU2co=GaussianFilter[LP8mmU2,1];

(*Create a Table with x and U1 data*)
Clear[LP0mmXU1Data,LP1mmXU1Data,LP2mmXU1Data,LP3mmXU1Data,LP4mmX
U1Data,LP5mmXU1Data,LP6mmXU1Data,LP7mmXU1Data,LP8mmXU1Data];
LP0mmXU1Data=Table[{LP0mmX[[i]],LP0mmU1co[[i]]},{i,1,Length[LP0m
mX]}];
LP1mmXU1Data=Table[{LP1mmX[[i]],LP1mmU1co[[i]]},{i,1,Length[LP1m
mX]}];
LP2mmXU1Data=Table[{LP2mmX[[i]],LP2mmU1co[[i]]},{i,1,Length[LP2m
mX]}];
LP3mmXU1Data=Table[{LP3mmX[[i]],LP3mmU1co[[i]]},{i,1,Length[LP3m
mX]}];
LP4mmXU1Data=Table[{LP4mmX[[i]],LP4mmU1co[[i]]},{i,1,Length[LP4m
mX]}];
LP5mmXU1Data=Table[{LP5mmX[[i]],LP5mmU1co[[i]]},{i,1,Length[LP5m
mX]}];
LP6mmXU1Data=Table[{LP6mmX[[i]],LP6mmU1co[[i]]},{i,1,Length[LP6m
mX]}];
LP7mmXU1Data=Table[{LP7mmX[[i]],LP7mmU1co[[i]]},{i,1,Length[LP7m
mX]}];
LP8mmXU1Data=Table[{LP8mmX[[i]],LP8mmU1co[[i]]},{i,1,Length[LP8m
mX]}];
```

```
(*Create a Table with x and U2 data*)
Clear[LP0mmXU2Data,LP1mmXU2Data,LP2mmXU2Data,LP3mmXU2Data,LP4mmX
U2Data,LP5mmXU2Data,LP6mmXU2Data,LP7mmXU2Data,LP8mmXU2Data];
LP0mmXU2Data=Table[{LP0mmX[[i]],LP0mmU2co[[i]]},{i,1,Length[LP0m
mX]}];
LP1mmXU2Data=Table[{LP1mmX[[i]],LP1mmU2co[[i]]},{i,1,Length[LP1m
mX]}];
LP2mmXU2Data=Table[{LP2mmX[[i]],LP2mmU2co[[i]]},{i,1,Length[LP2m
mX]}];
LP3mmXU2Data=Table[{LP3mmX[[i]],LP3mmU2co[[i]]},{i,1,Length[LP3m
mX]}];
LP4mmXU2Data=Table[{LP4mmX[[i]],LP4mmU2co[[i]]},{i,1,Length[LP4m
mX]}];
LP5mmXU2Data=Table[{LP5mmX[[i]],LP5mmU2co[[i]]},{i,1,Length[LP5m
mX]}];
LP6mmXU2Data=Table[{LP6mmX[[i]],LP6mmU2co[[i]]},{i,1,Length[LP6m
mX]}];
LP7mmXU2Data=Table[{LP7mmX[[i]],LP7mmU2co[[i]]},{i,1,Length[LP7m
mX]}];
LP8mmXU2Data=Table[{LP8mmX[[i]],LP8mmU2co[[i]]},{i,1,Length[LP8m
mX]}];

(*BSpline Functions*)
Clear[Y0mmU1,Y1mmU1,Y2mmU1,Y3mmU1,Y4mmU1,Y5mmU1,Y6mmU1,Y7mmU1,Y8
mmU1];
Clear[Y0mmU2,Y1mmU2,Y2mmU2,Y3mmU2,Y4mmU2,Y5mmU2,Y6mmU2,Y7mmU2,Y8
mmU2];
(*Interpolate to get the functions for U1*)
Y0mmU1=Interpolation[LP0mmXU1Data,Method->
"Spline",InterpolationOrder->3];
Y1mmU1=Interpolation[LP1mmXU1Data,Method->
"Spline",InterpolationOrder->3];
Y2mmU1=Interpolation[LP2mmXU1Data,Method->
"Spline",InterpolationOrder->3];
Y3mmU1=Interpolation[LP3mmXU1Data,Method->
"Spline",InterpolationOrder->3];
Y4mmU1=Interpolation[LP4mmXU1Data,Method->
"Spline",InterpolationOrder->3];
Y5mmU1=Interpolation[LP5mmXU1Data,Method->
"Spline",InterpolationOrder->3];
Y6mmU1=Interpolation[LP6mmXU1Data,Method->
"Spline",InterpolationOrder->3];
Y7mmU1=Interpolation[LP7mmXU1Data,Method->
"Spline",InterpolationOrder->3];
Y8mmU1=Interpolation[LP8mmXU1Data,Method->
"Spline",InterpolationOrder->3];
```

```
(*Interpolate to get the functions for U2*)
Y0mmU2=Interpolation[LP0mmXU2Data,Method->
"Spline",InterpolationOrder->3];
Y1mmU2=Interpolation[LP1mmXU2Data,Method->
"Spline",InterpolationOrder->3];
Y2mmU2=Interpolation[LP2mmXU2Data,Method->
"Spline",InterpolationOrder->3];
Y3mmU2=Interpolation[LP3mmXU2Data,Method->
"Spline",InterpolationOrder->3];
Y4mmU2=Interpolation[LP4mmXU2Data,Method->
"Spline",InterpolationOrder->3];
Y5mmU2=Interpolation[LP5mmXU2Data,Method->
"Spline",InterpolationOrder->3];
Y6mmU2=Interpolation[LP6mmXU2Data,Method->
"Spline",InterpolationOrder->3];
Y7mmU2=Interpolation[LP7mmXU2Data,Method->
"Spline",InterpolationOrder->3];
Y8mmU2=Interpolation[LP8mmXU2Data,Method->
"Spline",InterpolationOrder->3];

(*Discretized U2 Values*)
(*LP0mmU2DataDiscretized=Table[Y0mmU2Raw[LP0mmXDiscretized[[i]]]
,{i,1,Length[LP0mmXDiscretized]}];
LP0mmU2DataDiscretized0=Table[Y0mmU2Raw[LP0mmX[[i]]],{i,1,Length
[LP0mmX]}];
LP0mmU2DataDiscretized
LP0mmU2DataDiscretized0
LP0mmU2
Y0mmU2Raw[LP0mmX[[Length[LP0mmX]-1]]]*)

Clear[X10mm,X11mm,X12mm,X13mm,X14mm,X15mm,X16mm,X17mm,X18mm];
Clear[Theta0mmU2,Theta1mmU2,Theta2mmU2,Theta3mmU2,Theta4mmU2,The
ta5mmU2,Theta6mmU2,Theta7mmU2,Theta8mmU2];
(*Compute Theta Values based on U2 vallues*)
(*Theta0mmU2=ArcTan[D[Y0mmU2[X10mm],X10mm]];
Theta1mmU2=ArcTan[D[Y1mmU2[X11mm],X11mm]];
Theta2mmU2=ArcTan[D[Y2mmU2[X12mm],X12mm]];
Theta3mmU2=ArcTan[D[Y3mmU2[X13mm],X13mm]];*)
Theta4mmU2=ArcTan[D[Y4mmU2[X14mm],X14mm]];
(*Theta5mmU2=ArcTan[D[Y5mmU2[X15mm],X15mm]];
Theta6mmU2=ArcTan[D[Y6mmU2[X16mm],X16mm]];
Theta7mmU2=ArcTan[D[Y7mmU2[X17mm],X17mm]];
Theta8mmU2=ArcTan[D[Y8mmU2[X18mm],X18mm]];*)

(*Original Position Fields*)
Clear[X0mm,X1mm,X2mm,X3mm,X4mm,X5mm,X6mm,X7mm,X8mm];
Clear[Qz0mm,Qz1mm,Qz2mm,Qz3mm,Qz4mm,Qz5mm,Qz6mm,Qz7mm,Qz8mm];
```

```
Clear[x0mm,x1mm,x2mm,x3mm,x4mm,x5mm,x6mm,x7mm,x8mm];
X0mm={X10mm,X20mm};
X1mm={X11mm,X21mm};
X2mm={X12mm,X22mm};
X3mm={X13mm,X23mm};
X4mm={X14mm,X24mm};
X5mm={X15mm,X25mm};
X6mm={X16mm,X26mm};
X7mm={X17mm,X27mm};
X8mm={X18mm,X28mm};
(*
(*Rotation Matrices*)
Qz0mm={{Cos[Theta0mmU2],Sin[Theta0mmU2]},{-
1*Sin[Theta0mmU2],Cos[Theta0mmU2]}};
Qz2mm={{Cos[Theta2mmU2],Sin[Theta2mmU2]},{-
1*Sin[Theta2mmU2],Cos[Theta2mmU2]}};
Qz4mm={{Cos[Theta4mmU2],Sin[Theta4mmU2]},{-
1*Sin[Theta4mmU2],Cos[Theta4mmU2]}};
Qz6mm={{Cos[Theta6mmU2],Sin[Theta6mmU2]},{-
1*Sin[Theta6mmU2],Cos[Theta6mmU2]}};
Qz8mm={{Cos[Theta8mmU2],Sin[Theta8mmU2]},{-
1*Sin[Theta8mmU2],Cos[Theta8mmU2]}};

(*Final Position Fields*)
x0mm=Qz0mm.X0mm;
x2mm=Qz2mm.X2mm;
x4mm=Qz4mm.X4mm;
x6mm=Qz6mm.X6mm;
x8mm=Qz8mm.X8mm;

(*Displacement Values*)
U0mm=x0mm-X0mm;
U2mm=x2mm-X2mm;
U4mm=x4mm-X4mm;
U6mm=x6mm-X6mm;
U8mm=x8mm-X8mm;
*)

(*Modify Uz *)
Clear[U0mm,U1mm,U2mm,U3mm,          U4mm,U5mm,          U6mm,U7mm,
U8mm,U0mmmodify,Newfunc0mm,InterNewfunc0mm,U1mmmodify,Newfunc1mm
,InterNewfunc1mm,
U2mmmodify,Newfunc2mm,InterNewfunc2mm,U3mmmodify,Newfunc3mm,Inte
rNewfunc3mm,                              U4mmmodify,Newfunc4mm,
InterNewfunc4mm,U5mmmodify,Newfunc5mm,InterNewfunc5mm,U6mmmodify
,Newfunc6mm,InterNewfunc6mm,U7mmmodify,Newfunc7mm,InterNewfunc7m
m,
U8mmmodify,Newfunc8mm,InterNewfunc8mm,U1FE0mm,U1FE0,PInterNewfun
c0,InterNewfunc0,U1FE1mm,U1FE1,PInterNewfunc1,InterNewfunc1,U1FE
2mm,U1FE2,InterNewfunc2,PInterNewfunc2,U1FE3mm,U1FE3,PInterNewfu
nc3,InterNewfunc3,U1FE4mm,U1FE4,InterNewfunc4,PInterNewfunc4,U1F
E5mm,U1FE5,PInterNewfunc5,InterNewfunc5,U1FE6mm,U1FE6,InterNewfu
nc6,PInterNewfunc6,U1FE7mm,U1FE7,PInterNewfunc7,InterNewfunc7,U1
FE8mm,U1FE8,InterNewfunc8,PInterNewfunc8];



U0mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->-4.0},{i,1,Length[LP4mmX]}];
Newfunc0mm=Table[{LP4mmX[[i]],U0mmmodify[[i]]},{i,1,Length[LP4mm
X]}];
U1FE0mm=Table[(Y0mmU1[X10mm])/.{X10mm->
LP0mmX[[i]]},{i,1,Length[LP0mmX]}];
U1FE0=ListLinePlot[Table[{LP0mmX
[[i]],U1FE0mm[[i]]},{i,1,Length[LP0mmX]}],          PlotLegends-
>{"U1FE"}];
PInterNewfunc0=ListLinePlot[Table[{LP4mmX
[[i]],U0mmmodify[[i]]},{i,1,Length[LP0mmX]}],   PlotStyle->Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE    AND    FE    -4mm
HEIGHT"];
Show[PInterNewfunc0,U1FE0,PlotRange->All,ImageSize->1100]


U1mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->-3.0},{i,1,Length[LP4mmX]}];
Newfunc1mm=Table[{LP4mmX[[i]],U1mmmodify[[i]]},{i,1,Length[LP4mm
X]}];
U1FE1mm=Table[(Y1mmU1[X11mm])/.{X11mm->
LP1mmX[[i]]},{i,1,Length[LP1mmX]}];
U1FE1=ListLinePlot[Table[{LP1mmX
[[i]],U1FE1mm[[i]]},{i,1,Length[LP1mmX]}],          PlotLegends-
>{"U1FE"}];
PInterNewfunc1=ListLinePlot[Table[{LP4mmX
[[i]],U1mmmodify[[i]]},{i,1,Length[LP1mmX]}], PlotStyle-> Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE    AND    FE    -3mm
HEIGHT"];
Show[PInterNewfunc1,U1FE1,PlotRange->All,ImageSize->1100]
```

```
U2mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->-2.0},{i,1,Length[LP4mmX]}];
Newfunc0mm=Table[{LP4mmX[[i]],U0mmmodify[[i]]},{i,1,Length[LP4mm
X]}];
U1FE2mm=Table[(Y2mmU1[X12mm])/.{X12mm->
LP2mmX[[i]]},{i,1,Length[LP2mmX]}];
U1FE2=ListLinePlot[Table[{LP2mmX
[[i]],U1FE2mm[[i]]},{i,1,Length[LP2mmX]}],           PlotLegends-
>{"U1FE"}];
PInterNewfunc2=ListLinePlot[Table[{LP4mmX
[[i]],U2mmmodify[[i]]},{i,1,Length[LP2mmX]}], PlotStyle-> Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE    AND    FE    -2mm
HEIGHT"];
Show[PInterNewfunc2,U1FE2,PlotRange->All,ImageSize->1100]


U3mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->-1.0},{i,1,Length[LP4mmX]}];
Newfunc2mm=Table[{LP4mmX[[i]],U2mmmodify[[i]]},{i,1,Length[LP4mm
X]}];
U1FE3mm=Table[(Y3mmU1[X13mm])/.{X13mm->
LP3mmX[[i]]},{i,1,Length[LP3mmX]}];
U1FE3=ListLinePlot[Table[{LP3mmX
[[i]],U1FE3mm[[i]]},{i,1,Length[LP3mmX]}],           PlotLegends-
>{"U1FE"}];
PInterNewfunc3=ListLinePlot[Table[{LP4mmX
[[i]],U3mmmodify[[i]]},{i,1,Length[LP3mmX]}], PlotStyle-> Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE    AND    FE    -1mm
HEIGHT"];
Show[PInterNewfunc3,U1FE3,PlotRange->All,ImageSize->1100]



U4mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->0.0},{i,1,Length[LP4mmX]}];
Newfunc4mm=Table[{LP4mmX[[i]],U4mmmodify[[i]]},{i,1,Length[LP4mm
X]}];
U1FE4mm=Table[(Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]]},{i,1,Length[LP4mmX]}];
U1FE4=ListLinePlot[Table[{LP4mmX
[[i]],U1FE4mm[[i]]},{i,1,Length[LP4mmX]}],           PlotLegends-
>{"U1FE"}];
```

```
PInterNewfunc4=ListLinePlot[Table[{LP4mmX
[[i]],U4mmmodify[[i]]},{i,1,Length[LP4mmX]}],   PlotStyle->Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE AND FE 0mm HEIGHT"];
Show[PInterNewfunc4,U1FE4,PlotRange->All,ImageSize->1100]

U5mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->1.0},{i,1,Length[LP4mmX]}];
Newfunc5mm=Table[{LP4mmX[[i]],U4mmmodify[[i]]},{i,1,Length[LP4mm
X]}];
U1FE5mm=Table[(Y5mmU1[X15mm])/.{X15mm->
LP1mmX[[i]]},{i,1,Length[LP5mmX]}];
U1FE5=ListLinePlot[Table[{LP5mmX
[[i]],U1FE5mm[[i]]},{i,1,Length[LP5mmX]}],           PlotLegends-
>{"U1FE"}];
PInterNewfunc5=ListLinePlot[Table[{LP4mmX
[[i]],U5mmmodify[[i]]},{i,1,Length[LP5mmX]}], PlotStyle-> Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE AND FE 1mm HEIGHT"];
Show[PInterNewfunc5,U1FE5,PlotRange->All,ImageSize->1100]


U6mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->2.0},{i,1,Length[LP4mmX]}];
Newfunc6mm=Table[{LP4mmX[[i]],U6mmmodify[[i]]},{i,1,Length[LP4mm
X]}];
U1FE6mm=Table[(Y6mmU1[X16mm])/.{X16mm->
LP6mmX[[i]]},{i,1,Length[LP6mmX]}];
U1FE6=ListLinePlot[Table[{LP6mmX
[[i]],U1FE6mm[[i]]},{i,1,Length[LP6mmX]}],           PlotLegends-
>{"U1FE"}];
PInterNewfunc6=ListLinePlot[Table[{LP4mmX
[[i]],U6mmmodify[[i]]},{i,1,Length[LP6mmX]}], PlotStyle-> Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE AND FE 2mm HEIGHT"];
Show[PInterNewfunc6,U1FE6,PlotRange->All,ImageSize->1100]

U7mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->3.0},{i,1,Length[LP4mmX]}];
Newfunc7mm=Table[{LP4mmX[[i]],U7mmmodify[[i]]},{i,1,Length[LP7mm
X]}];
U1FE7mm=Table[(Y7mmU1[X17mm])/.{X17mm->
LP7mmX[[i]]},{i,1,Length[LP7mmX]}];
U1FE7=ListLinePlot[Table[{LP7mmX
[[i]],U1FE7mm[[i]]},{i,1,Length[LP7mmX]}],           PlotLegends-
>{"U1FE"}];
```

```
PInterNewfunc7=ListLinePlot[Table[{LP4mmX
[[i]],U7mmmodify[[i]]},{i,1,Length[LP6mmX]}], PlotStyle-> Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE AND FE 3mm HEIGHT"];
Show[PInterNewfunc7,U1FE7,PlotRange->All,ImageSize->1100]


U8mmmodify=Table[(-
1.0*X24mm*Sin[Theta4mmU2]+Y4mmU1[X14mm])/.{X14mm->
LP4mmX[[i]],X24mm->4.0},{i,1,Length[LP4mmX]}];
Newfunc8mm=Table[{LP4mmX[[i]],U8mmmodify[[i]]},{i,1,Length[LP8mm
X]}];
U1FE8mm=Table[(Y8mmU1[X18mm])/.{X18mm-
>LP8mmX[[i]]},{i,1,Length[LP8mmX]}];
U1FE8=ListLinePlot[Table[{LP8mmX
[[i]],U1FE8mm[[i]]},{i,1,Length[LP8mmX]}],          PlotLegends-
>{"U1FE"}];
PInterNewfunc8=ListLinePlot[Table[{LP4mmX
[[i]],U8mmmodify[[i]]},{i,1,Length[LP8mmX]}],   PlotStyle->Black,
PlotLegends->{"U1Chike"},PlotLabel->"U1CHIKE   AND   FE   AT   4mm
HEIGHT"];
Show[PInterNewfunc8,U1FE8,PlotRange->All,ImageSize->1100]



(*Displacement Values - Equations based on Chike's code*)
Clear[U0mm,U1mm,U2mm,U3mm,U4mm,U5mm,U6mm,U7mm,U8mm];
(*U0mm={Y0mmU1[X10mm],Y0mmU2[X10mm]+X20mm*(Cos[Theta0mmU2]-
1.0)};
U1mm={Y1mmU1[X11mm],Y1mmU2[X11mm]+X21mm*(Cos[Theta1mmU2]-1.0)};
U2mm={Y2mmU1[X12mm],Y2mmU2[X12mm]+X22mm*(Cos[Theta2mmU2]-1.0)};
U3mm={Y3mmU1[X13mm],Y3mmU2[X13mm]+X23mm*(Cos[Theta3mmU2]-
1.0)};*)
U4mm={(-
1.0*X24mm*Sin[Theta4mmU2])+Y4mmU1[X14mm],Y4mmU2[X14mm]+X24mm*(Co
s[Theta4mmU2]-1.0)};
(*U5mm={Y5mmU1[X15mm],Y5mmU2[X15mm]+X25mm*(Cos[Theta5mmU2]-
1.0)};
U6mm={Y6mmU1[X16mm],Y6mmU2[X16mm]+X26mm*(Cos[Theta6mmU2]-1.0)};
U7mm={Y7mmU1[X17mm],Y7mmU2[X17mm]+X27mm*(Cos[Theta7mmU2]-1.0)};
U8mm={Y8mmU1[X18mm],Y8mmU2[X18mm]+X28mm*(Cos[Theta8mmU2]-
1.0)};*)


(*Displacement Gradient*)
Clear[DelU0mm,DelU1mm,DelU2mm,DelU3mm,DelU4mm,DelU5mm,DelU6mm,De
lU7mm,DelU8mm];
(*DelU0mm=Table[D[U0mm[[i]],X0mm[[j]]],{i,1,2},{j,1,2}];
DelU1mm=Table[D[U1mm[[i]],X1mm[[j]]],{i,1,2},{j,1,2}];
DelU2mm=Table[D[U2mm[[i]],X2mm[[j]]],{i,1,2},{j,1,2}];
DelU3mm=Table[D[U3mm[[i]],X3mm[[j]]],{i,1,2},{j,1,2}];*)
DelU4mm=Table[D[U4mm[[i]],X4mm[[j]]],{i,1,2},{j,1,2}];
(*DelU5mm=Table[D[U5mm[[i]],X5mm[[j]]],{i,1,2},{j,1,2}];
DelU6mm=Table[D[U6mm[[i]],X6mm[[j]]],{i,1,2},{j,1,2}];
DelU7mm=Table[D[U7mm[[i]],X7mm[[j]]],{i,1,2},{j,1,2}];
DelU8mm=Table[D[U8mm[[i]],X8mm[[j]]],{i,1,2},{j,1,2}];*)

Clear[GreenStrain0mmEqn,GreenStrain1mmEqn,GreenStrain2mmEqn,Gree
nStrain3mmEqn,GreenStrain4mmEqn,GreenStrain5mmEqn,GreenStrain6mm
Eqn,GreenStrain7mmEqn,GreenStrain8mmEqn];
(*GreenStrain0mmEqn=0.5*(DelU0mm+Transpose[DelU0mm]+(Transpose[D
elU0mm].DelU0mm));
GreenStrain1mmEqn=0.5*(DelU1mm+Transpose[DelU1mm]+(Transpose[Del
U1mm].DelU1mm));
GreenStrain2mmEqn=0.5*(DelU2mm+Transpose[DelU2mm]+(Transpose[Del
U2mm].DelU2mm));
GreenStrain3mmEqn=0.5*(DelU3mm+Transpose[DelU3mm]+(Transpose[Del
U3mm].DelU3mm));*)
GreenStrain4mmEqn=0.5*(DelU4mm+Transpose[DelU4mm]+(Transpose[Del
U4mm].DelU4mm));
(*GreenStrain5mmEqn=0.5*(DelU5mm+Transpose[DelU5mm]+(Transpose[D
elU5mm].DelU5mm));
GreenStrain6mmEqn=0.5*(DelU6mm+Transpose[DelU6mm]+(Transpose[Del
U6mm].DelU6mm));
GreenStrain7mmEqn=0.5*(DelU7mm+Transpose[DelU7mm]+(Transpose[Del
U7mm].DelU7mm));
GreenStrain8mmEqn=0.5*(DelU8mm+Transpose[DelU8mm]+(Transpose[Del
U8mm].DelU8mm));*)

Clear[GreenStrain0mmMatrixValues,GreenStrain1mmMatrixValues,Gree
nStrain2mmMatrixValues,GreenStrain3mmMatrixValues,GreenStrain4mm
MatrixValues,GreenStrain5mmMatrixValues,GreenStrain6mmMatrixValu
es,GreenStrain7mmMatrixValues,GreenStrain8mmMatrixValues];
GreenStrain0mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> -4.0},{i,1,Length[LP0mmX]}];
GreenStrain1mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> -3.0},{i,1,Length[LP1mmX]}];
GreenStrain2mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> -2.0},{i,1,Length[LP2mmX]}];
GreenStrain3mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> -1.0},{i,1,Length[LP3mmX]}];
GreenStrain4mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> 0.0},{i,1,Length[LP4mmX]}];
GreenStrain5mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> 1.0},{i,1,Length[LP5mmX]}];
```

```
GreenStrain6mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> 2.0},{i,1,Length[LP6mmX]}];
GreenStrain7mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> 3.0},{i,1,Length[LP7mmX]}];
GreenStrain8mmMatrixValues=Table[GreenStrain4mmEqn/.{X14mm->
LP4mmX[[i]],X24mm-> 4.0},{i,1,Length[LP8mmX]}];

Clear[GreenStrain0mm11,GreenStrain1mm11,GreenStrain2mm11,GreenSt
rain3mm11,GreenStrain4mm11,GreenStrain5mm11,GreenStrain6mm11,Gre
enStrain7mm11,GreenStrain8mm11];
GreenStrain0mm11=Table[GreenStrain0mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain0mmMatrixValues]}];
GreenStrain1mm11=Table[GreenStrain1mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain1mmMatrixValues]}];
GreenStrain2mm11=Table[GreenStrain2mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain2mmMatrixValues]}];
GreenStrain3mm11=Table[GreenStrain3mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain3mmMatrixValues]}];
GreenStrain4mm11=Table[GreenStrain4mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain4mmMatrixValues]}];
GreenStrain5mm11=Table[GreenStrain5mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain5mmMatrixValues]}];
GreenStrain6mm11=Table[GreenStrain6mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain6mmMatrixValues]}];
GreenStrain7mm11=Table[GreenStrain7mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain7mmMatrixValues]}];
GreenStrain8mm11=Table[GreenStrain8mmMatrixValues[[i]][[1,1]],{i
,1,Length[GreenStrain8mmMatrixValues]}];
(*
Print["Green Strain 11 at Path 0mm = ",GreenStrain0mm11];
Print["Green Strain 11 at Path 2mm = ",GreenStrain2mm11];
Print["Green Strain 11 at Path 4mm = ",GreenStrain4mm11];
Print["Green Strain 11 at Path 6mm = ",GreenStrain6mm11];
Print["Green Strain 11 at Path 8mm = ",GreenStrain8mm11];
*)

(*Comparative Plots*)
(*At 0mm position - Inner surface of the pipe*)
XLE110mmTable                    =                  Table[{LP0mmX
[[i]],LE110mm[[i]]},{i,1,Length[LP0mmX]}];
XPE110mmTable                    =                  Table[{LP0mmX
[[i]],PE110mm[[i]]},{i,1,Length[LP0mmX]}];
XGreenStrain0mm110mmTable              =            Table[{LP0mmX
[[i]],GreenStrain0mm11[[i]]},{i,1,Length[LP0mmX]}];

P01=ListLinePlot[XLE110mmTable,                        PlotStyle-
>{Black,Thickness[0.01]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE    AT    0mm    HEIGHT",    AxesLabel->{"Longitudinal    Distance
[mm]","Strain"},PlotLegends->{"LE11 (Abaqus)"}];
P02=ListLinePlot[XPE110mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  0mm  HEIGHT",  AxesLabel-
>{"Longitudinal        Distance        [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
P03=ListLinePlot[XGreenStrain0mm110mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE    AT    0mm    HEIGHT",    AxesLabel->{"Longitudinal    Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11"}];
Show[P01,P02, P03, PlotRange-> All,ImageSize->1100]

(*At 1mm position - Inner surface of the pipe*)
XLE111mmTable                  =                   Table[{LP1mmX
[[i]],LE111mm[[i]]},{i,1,Length[LP1mmX]}];
XPE111mmTable                  =                   Table[{LP1mmX
[[i]],PE111mm[[i]]},{i,1,Length[LP1mmX]}];
XGreenStrain1mm111mmTable             =            Table[{LP1mmX
[[i]],GreenStrain1mm11[[i]]},{i,1,Length[LP1mmX]}];

P11=ListLinePlot[XLE111mmTable,                         PlotStyle-
>{Black,Thickness[0.01]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE    AT    1mm    HEIGHT",    AxesLabel->{"Longitudinal    Distance
[mm]","Strain"},PlotLegends->{"LE11 (Abaqus)"}];
P12=ListLinePlot[XPE111mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  1mm  HEIGHT",  AxesLabel-
>{"Longitudinal        Distance        [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
P13=ListLinePlot[XGreenStrain1mm111mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE    AT    1mm    HEIGHT",    AxesLabel->{"Longitudinal    Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11 (Chike)"}];
Show[P11,P12, P13, PlotRange-> All,ImageSize->1100]

(*At 2mm height from inner surface*)
XLE112mmTable                  =                   Table[{LP2mmX
[[i]],LE112mm[[i]]},{i,1,Length[LP2mmX]}];
XPE112mmTable                  =                   Table[{LP2mmX
[[i]],PE112mm[[i]]},{i,1,Length[LP2mmX]}];
XGreenStrain2mm112mmTable             =            Table[{LP2mmX
[[i]],GreenStrain2mm11[[i]]},{i,1,Length[LP2mmX]}];

P21=ListLinePlot[XLE112mmTable,                         PlotStyle-
>{Black,Thickness[0.01]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE    AT    2mm    HEIGHT",    AxesLabel->{"Longitudinal    Distance
[mm]","Strain"},PlotLegends->{"LE11(Abaqus)"}];
```

```
P22=ListLinePlot[XPE112mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  2mm  HEIGHT",  AxesLabel-
>{"Longitudinal      Distance      [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
P23=ListLinePlot[XGreenStrain2mm112mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  2mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11 (Chike)"}];
Show[P21,P22, P23, PlotRange-> All,ImageSize->1100]

(*At 3mm position - Inner surface of the pipe*)
XLE113mmTable                   =                   Table[{LP3mmX
[[i]],LE113mm[[i]]},{i,1,Length[LP3mmX]}];
XPE113mmTable                   =                   Table[{LP3mmX
[[i]],PE113mm[[i]]},{i,1,Length[LP3mmX]}];
XGreenStrain3mm113mmTable             =             Table[{LP3mmX
[[i]],GreenStrain3mm11[[i]]},{i,1,Length[LP3mmX]}];

P31=ListLinePlot[XLE113mmTable,                         PlotStyle-
>{{Black,Thickness[0.01]},Thickness[0.01]},PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  3mm  HEIGHT",  AxesLabel-
>{"Longitudinal   Distance   [mm]","Strain"},PlotLegends->{"LE11
(Abaqus)"}];
P32=ListLinePlot[XPE113mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  3mm  HEIGHT",  AxesLabel-
>{"Longitudinal      Distance      [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
P33=ListLinePlot[XGreenStrain3mm113mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  3mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11 (Chike)"}];
Show[P31,P32, P33, PlotRange->All,ImageSize->1100 ]

(*At 4mm height from inner surface*)
XLE114mmTable                   =                   Table[{LP4mmX
[[i]],LE114mm[[i]]},{i,1,Length[LP4mmX]}];
XPE114mmTable                   =                   Table[{LP4mmX
[[i]],PE114mm[[i]]},{i,1,Length[LP4mmX]}];
XGreenStrain4mm114mmTable             =             Table[{LP4mmX
[[i]],GreenStrain4mm11[[i]]},{i,1,Length[LP4mmX]}];

P41=ListLinePlot[XLE114mmTable,                         PlotStyle-
>{{Black,Thickness[0.01]},Thickness[0.01]},PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  4mm  HEIGHT",  AxesLabel-
>{"Longitudinal      Distance      [mm]","Strain"},PlotLegends-
>{"LE11(Abaqus)"}];
```

```
P42=ListLinePlot[XPE114mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  4mm  HEIGHT",  AxesLabel-
>{"Longitudinal       Distance       [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
P43=ListLinePlot[XGreenStrain4mm114mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  4mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11 (Chike)"}];
Show[P41,P42, P43, PlotRange-> All,ImageSize->1100]

(*At 5mm position - Inner surface of the pipe*)
XLE115mmTable                  =                 Table[{LP5mmX
[[i]],LE115mm[[i]]},{i,1,Length[LP5mmX]}];
XPE115mmTable                  =                 Table[{LP5mmX
[[i]],PE115mm[[i]]},{i,1,Length[LP5mmX]}];
XGreenStrain5mm115mmTable           =            Table[{LP5mmX
[[i]],GreenStrain5mm11[[i]]},{i,1,Length[LP5mmX]}];

P51=ListLinePlot[XLE115mmTable,                     PlotStyle-
>{Black,Thickness[0.01]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  5mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"LE11(Abaqus)"}];
P52=ListLinePlot[XPE115mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  5mm  HEIGHT",  AxesLabel-
>{"Longitudinal       Distance       [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
P53=ListLinePlot[XGreenStrain5mm115mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  5mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11 (Chike)"}];
Show[P51,P52, P53, PlotRange-> All,ImageSize->1100]

(*At 6mm height from inner surface*)
XLE116mmTable                  =                 Table[{LP6mmX
[[i]],LE116mm[[i]]},{i,1,Length[LP6mmX]}];
XPE116mmTable                  =                 Table[{LP6mmX
[[i]],PE116mm[[i]]},{i,1,Length[LP6mmX]}];
XGreenStrain6mm116mmTable           =            Table[{LP6mmX
[[i]],GreenStrain6mm11[[i]]},{i,1,Length[LP6mmX]}];

P61=ListLinePlot[XLE116mmTable,                     PlotStyle-
>{Black,Thickness[0.01]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  6mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"LE11(Abaqus)"}];
P62=ListLinePlot[XPE116mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  6mm  HEIGHT",  AxesLabel-
>{"Longitudinal       Distance       [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
P63=ListLinePlot[XGreenStrain6mm116mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  6mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11 (Chike)"}];
Show[P61,P62, P63, PlotRange-> All,ImageSize->1100]

(*At 7mm position - Inner surface of the pipe*)
XLE117mmTable                  =                    Table[{LP7mmX
[[i]],LE117mm[[i]]},{i,1,Length[LP7mmX]}];
XPE117mmTable                  =                    Table[{LP7mmX
[[i]],PE117mm[[i]]},{i,1,Length[LP7mmX]}];
XGreenStrain7mm117mmTable            =              Table[{LP7mmX
[[i]],GreenStrain7mm11[[i]]},{i,1,Length[LP7mmX]}];

P71=ListLinePlot[XLE117mmTable,                        PlotStyle-
>{Black,Thickness[0.01]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  7mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"LE11 (Abaqus)"}];
P72=ListLinePlot[XPE117mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  7mm  HEIGHT",  AxesLabel-
>{"Longitudinal       Distance       [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
P73=ListLinePlot[XGreenStrain7mm117mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  7mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11 (Chike)"}];
Show[P71,P72, P73, PlotRange-> All,ImageSize->1100]

(*At 8mm height from inner surface*)
XLE118mmTable                  =                    Table[{LP8mmX
[[i]],LE118mm[[i]]},{i,1,Length[LP8mmX]}];
XPE118mmTable                  =                    Table[{LP8mmX
[[i]],PE118mm[[i]]},{i,1,Length[LP8mmX]}];
XGreenStrain8mm118mmTable            =              Table[{LP8mmX
[[i]],GreenStrain8mm11[[i]]},{i,1,Length[LP8mmX]}];

P81=ListLinePlot[XLE118mmTable,                        PlotStyle-
>{Black,Thickness[0.01]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE  AT  8mm  HEIGHT",  AxesLabel->{"Longitudinal  Distance
[mm]","Strain"},PlotLegends->{"LE11(Abaqus)"}];
P82=ListLinePlot[XPE118mmTable,PlotStyle->Blue,PlotRange-
>Full,PlotLabel->"STRAIN  PROFILE  AT  8mm  HEIGHT",  AxesLabel-
>{"Longitudinal       Distance       [mm]","Strain"},PlotLegends-
>{"PE11(Abaqus)"}];
```

```
P83=ListLinePlot[XGreenStrain8mm118mmTable,PlotStyle-
>{Red,Thickness[0.003]},PlotRange->Full,PlotLabel->"STRAIN
PROFILE   AT   8mm   HEIGHT",   AxesLabel->{"Longitudinal   Distance
[mm]","Strain"},PlotLegends->{"GreenStrain11 (Chike)"}];
Show[P81,P82, P83, PlotRange->All,ImageSize->1100]
```

APPENDIX C: Mathematica code in the Circumferential Direction and Result Extraction.

```
DataR    =    Import["C:\\Users\\mahya\\OneDrive\\Desktop\\Excell files\\4mm.xlsx"];

(*Sorts the input data*)

AllData = Table[DataR[[1, i]], {i, 2, 327}];

LongAxis = Table[AllData[[i, 1]], {i, 1, 301}];

LongDisp = Table[AllData[[i, 2]], {i, 1,301}];

CircAxis = Table[AllData[[i, 3]], {i, 1, Length[AllData]}];

CircDisp = Table[AllData[[i, 4]], {i, 1, Length[AllData]}];

DataL     =     Table[{LongAxis[[i]],     LongDisp[[i]]},     {i, Length[LongAxis]}];

DataC     =     Table[{CircAxis[[i]],     CircDisp[[i]]},     {i, Length[CircAxis]}];



(*To further sort Data-Suppresed for this case*)


(*DataL=Table[DataL[[i]],{i,1,Length[DataL],1}];

DataC=Table[DataC[[i]],{i,1,Length[DataC],1}];*)


(*Input Variables*)


t = 8;

ro = (16*25.4) - t;
```

```
rom = (16*25.4) - t/2;


(*Resolution of tool; 64 arms in the circ direction and readings
at 5mm intervals on the longitudinal axis*)

CircRes = 64;

LongRes = 5;


(*Axial Start and End Positions*)

start = 100;

end = 100;


(*Interpolation*)

ri = Interpolation[DataC, Method -> "Spline", InterpolationOrder
-> 3];

yfunc    =    Interpolation[DataL,    Method    ->    "Spline",
InterpolationOrder -> 3];


(*Data Wrangle*)

yi = yfunc[x] /. x -> Table[i, {i, -100, 100, LongRes}];

xi = Table[i, {i, -100, 100, LongRes}];

DataL = Table[{xi[[i]], yi[[i]]}, {i, 1, Length[yi]}];


rt = ri[th] /. th -> Table[i, {i, 0, 3.14, Pi/64}];

the = Table[i, {i, 0, 3.14, Pi/64}];

DataC = Table[{the[[i]], rt[[i]]}, {i, 1, Length[rt]}];
```

```
ri = Interpolation[DataC, Method -> "Spline", InterpolationOrder
-> 3];

yfunc = Interpolation[DataL, Method -> "Spline",
InterpolationOrder -> 3];


PolarPlot[ri[th], {th, 0, 3.14}, PlotRange -> All, AxesOrigin ->
{0, 0}, FrameLabel -> {" X (mm)", "Y (mm) "}, BaseStyle ->
Directive[Bold, 20], Frame -> True, GridLines -> Automatic]


Plot[yfunc[x], {x, -100, 100}, PlotRange -> All, AxesOrigin -> {0,
0}, FrameLabel -> {" Axial Distance (mm)", "Radial Displacement
"}, BaseStyle -> Directive[Bold, 20], Frame -> True, GridLines ->
Automatic]



(*Deformation Gradient in the Longitudinal Axis*)

Theta1 = ArcTan[D[yfunc[x], x]];

u1 = {-x2*Sin[Theta1], yfunc[x] + x2*(Cos[Theta1] - 1), 0};

ua1 = u /. x2 -> t/2;

ua2 = u1 /. x2 -> t/2;

X = {x, x2, x3};

Gradu1 = Table[D[u1[[i]], X[[j]]], {i, 1, 3}, {j, 1, 3}];


(*Green Strain Matrix*)

GreenStrain = 0.5*(Gradu1 + Transpose[Gradu1] +
1*Transpose[Gradu1].Gradu1);


GSB = GreenStrain[[1, 1]] /. x2 -> -t/2;
```

```
GST = GreenStrain[[1, 1]] /. x2 -> t/2.;


(*same as from ABAQUS, with-3.14 correspomding to 180deg, and 3.14
*)


thstart = DataC[[Length[DataC], 1]];

thend = DataC[[1, 1]];


(*The following angles aren't used*)

perpangle = ArcTan[D[ri[th], th]/ri[th]];

perpanglesmall = D[ri[th], th]/ri[th];


(*First approximation of the radius of the midsurface*)

rm = ri[th] + t/2*Cos[perpangle];

rmsmall = ri[th] + t/2*Cos[perpanglesmall];


(*Arc Length as theta varies from start to end in the deformed
configuration*)

lmtablesmall = Table[NIntegrate[rmsmall, {th, thstart, DataC[[i,
1]]}], {i, 1, Length[DataC]}];


romest = NIntegrate[rmsmall, {th, 0, Pi}]/(Pi);


(*Arc Length as th varies from start to end in the undeformed
configuration*)

Oldlmtable = Table[NIntegrate[rom, {th, thstart, DataC[[i, 1]]}],
{i, 1, Length[DataC]}];
```

```
(*Table showing the value of th in the undeformed config and the coresponnding value in the deformed config*)


thtablesmall = Table[{DataC[[i, 1]], lmtablesmall[[i]]/romest + thstart}, {i, 1, Length[Oldlmtable]}];

thnewsmall = Table[{thtablesmall[[i, 2]], thtablesmall[[i, 1]]}, {i, 1, Length[thtablesmall]}];




(*Interpolation Function relating the undeformed and the deformed values of theta*)

rsmall = rmsmall + tv;

thnewintsmall = Interpolation[thnewsmall, Method -> "Spline", InterpolationOrder -> 3];


perpanglesmall = D[rmsmall, th]/rmsmall;


(*midline radial displacement *)


urmsmall = (rmsmall /. th -> thnewintsmall[thold])*Cos[thnewintsmall[thold] - thold] - romest;


uthmsmall = (rmsmall /. th -> thnewintsmall[thold])*Sin[thnewintsmall[thold] - thold];

cc = uthmsmall /. thold -> 0;

uthmsmall = uthmsmall - cc;
```

```
(*radial displacement*)


ursmall = urmsmall - tv + tv*Cos[perpanglesmall /. th -> thnewintsmall[thold]];

(*circumferential displacement*)


uthsmall = uthmsmall - tv*Sin[perpanglesmall+0.00375/. th -> thnewintsmall[thold]];


rsmall = (rmsmall /. th -> thnewintsmall[thold]) + tv;


SU = {{D[ursmall, tv], D[ursmall, thold]/rsmall - uthsmall/rsmall}, {D[uthsmall, tv], 1*ursmall/rsmall + 1/rsmall*D[uthsmall, thold]}};


(*Small Strain Matrix*)


SStrain = 0.5*(SU + Transpose[SU]);


SStrainT = SStrain[[2, 2]] /. tv -> t/2;

SStrainB = SStrain[[2, 2]] /. tv -> -t/2;


epsCB = SStrainB /. thold ->0;

epsCT = SStrainT /. thold -> 0;


epsCB
```