Winter 12-21-2018

Modeling and Simulation of the Thermoforming Process in Thermoplastic-Matrix Composite Materials

Philip M. Bean
University of Maine, philip.m.bean@maine.edu

MODELING AND SIMULATION OF THE THERMOFORMING PROCESS IN THERMOPLASTIC-MATRIX COMPOSITE MATERIALS

By

Philip Michael Bean

B.S. University of Maine, 2016

A THESIS

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

(in Mechanical Engineering)

The Graduate School

The University of Maine

December 2018

Advisory Committee:

Dr. Roberto A. Lopez-Anido, P.E., Professor of Civil Engineering, Co-Advisor

Dr. Senthil S. Vel, Professor of Mechanical Engineering, Co-Advisor

Dr. Vincent Caccese, Professor of Mechanical Engineering

Dr. Zhihe Jin, Professor of Mechanical Engineering

MODELING AND SIMULATION OF THE THERMOFORMING PROCESS IN THERMOPLASTIC-MATRIX COMPOSITE MATERIALS

By Philip Michael Bean

Thesis Co-Advisors: Dr. Roberto Lopez-Anido and Dr. Senthil Vel

An Abstract of the Thesis Presented
in Partial Fulfillment of the Requirements for the
Degree of Master of Science
(in Mechanical Engineering)
December 2018

Thermoplastic-matrix composite materials have unique advantages over traditional thermosets including faster processing, improved fracture toughness, and recyclability. These and other benefits have caused increasing interest in the use of these materials in both aerospace and automotive industries. Due to the differences in behavior, these materials require a different type of manufacturing process to thermoset matrix composites. This manufacturing process generally involves using pre manufactured tape-layers. These layers, containing both thermoplastic-matrix and fiber-reinforcement, are aligned to the desired orientation, and stacked up into a "tailored blank" using an automated tape layup machine. They are then heated to the thermoplastic melting temperature in an oven and stamped to shape using a large press. Due to some complex behaviors in processing, however, it is necessary to simulate the forming process prior to manufacturing. Simulation can help to avoid costly trial-and-error type process tuning in order to avoid manufacturing defects like wrinkles and tears and to optimize the use of material. A research effort has been undertaken in order to streamline the process of material characterization toward simulation, which are accomplished using commercially available software. This includes a variety of material characterization tests, as well as forming tests in order to compare simulated results, such as predicted wrinkles

and fiber reorientations in complex-shaped parts, to real parts manufactured under the same conditions as the simulations. Finally, a tool has been developed which allows the extension of these forming simulations to inform more accurate structural analyses of as-formed parts.

ACKNOWLEDGEMENTS

First, I would like to thank my advisory committee for their guidance throughout this research. Their input has been invaluable through the entire process.

I would also like to thank Nick Fitzpatrick and the rest of the ASCC Structural Thermoplastics team. Without their help, the physical side of this work would have been nearly impossible. In addition, I'd like to thank the entire staff of the ASCC for fostering this research, and to acknowledge the Harold W. Alfond Graduate Research Assistantship for funding my work.

Last, I want to thank my family. Without their support through this entire process, I would undoubtedly have gone insane.

Thank You!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Simulation tools are very important for the purpose of optimizing the stamp-thermoforming process. In order to make full use of these tools, research has been done to demonstrate and verify a method of material characterization toward simulation, and further, to combine the forming models with other solid-modeling techniques for final stress analysis.

1.1 Thermoplastic-Matrix Composites

Continuous fiber reinforced thermoplastic composite materials (CFRTPs) contain fiber reinforcement in a meltable thermoplastic matrix. This is different from the more common thermoset composites in that the matrix is a different type of polymer. In thermoset composites, the reinforcing fibers are embedded in a matrix of a chemically-cured resin, whereas in thermoplastic composites, the fibers are embedded in a thermoplastic resin. This difference gives CFRTPs some unique advantages. For example, thermoplastic composites tend to have higher fracture toughness than their thermoset counterparts. Due to their reprocessability by melting, thermoplastic composites are much more easily recyclable than thermosets. The melt-processing also lends itself nicely to rapid-manufacturing technologies (unlike thermosets which generally require a relatively long resin-cure time). Finally, the melt process also gives the additional benefit of allowing assembly or repair by welding (in addition to the standard gluing and bolting).

1.1.1 Material Systems

Because of the usefulness of thermoplastic composites, there have been a variety of different material systems developed, thus expanding the versatility and usability of this class of materials. The primary differentiations between these material systems (like with all composites) are those of fiber-composition, fiber-architecture, and matrix-composition.

Fiber Composition While there are a variety of reinforcing fiber materials available, ranging from natural fibers like cellulose to aramids like Kevlar® and even some metals, there are in general two standard fiber types that are used in many of the common engineered composite systems(both thermoplastic and thermoset). These are carbon and glass. In general, if the composite is needed to be exceptionally lightweight, stiff, or strong, carbon is the fiber of choice. Carbon-fiber, however, is very expensive, and can therefore only be used in applications where the performance is the driving factor, and cost is secondary. Glass-fiber, while it does have good performance in many applications, is not as lightweight, stiff, or strong as carbon. It is, however, significantly less costly than carbon, making it the fiber of choice when cost is a significant factor in the design. For this research, glass-fiber was used, since it is less costly, and very commonly used. Also, for the purposes of forming models, the fiber composition has very little effect on the behavior, since the primary deformation mode is viscous shear and matrix flow[1, 2, 3].

Fiber Architecture Despite the variety of different fiber architectures, the majority of composites can be classified into one of two categories: continuous, and discontinuous. Discontinuous-fiber systems (also called short-fiber or chopped-fiber) typically produce the lowest-performance composites. They are, however, the easiest to manufacture due to the ability of the fibers to slide relative to one another. In the case of very short fibers, these composites can even be injection molded like neat thermoplastics. Continuous-fiber systems can be further divided into three subcategories: knit, woven, and unidirectional. These tend to form a hierarchy in which increasing performance corresponds to decreasing formability. Thus, knit systems are the most formable[4], but lowest performance. Woven systems have much higher performance than knits, but are much more difficult to form without defects. The highest performance systems are unidirectional ones. These are slightly better performing than wovens, but also slightly more difficult to form[5, 6]. In this research, unidirectional systems are used due to the ability to produce tailored blanks through automated

tape placement. In addition, the high performance of these is a critical factor for engineering uses.

Matrix Composition While there is a lot of variety in the choice of fiber systems, it is dwarfed by the level of variety available in the selection of matrix polymer. This variety stems directly from the variety seen in polymer chemistry and plastics manufacture, since a significant number of the thermoplastics that have been previously developed for other purposes, have also been adapted for use in composite materials. the Periodic Table of Thermoplastics (in Appendix A) shows a comparison of basic thermoplastics and their properties. Polypropylene was chosen for this research primarily because, being a commodity plastic, it is very inexpensive. In addition, the relatively low melt temperature makes it possible to easily press-form using 3D printed molds without thermally damaging the mold.

1.1.2 Manufacturing

As previously mentioned, properties of thermoplastics can lead to some very rapid manufacturing technology which can make composites more competitive in industries where rapid mass-production is imperative (like the automotive industry). One important factor here is that thermoplastics can be pre-manufactured into prepreg "blanks". These blanks, which have an indefinite shelf-life, can then be heated and formed to shape in a variety of ways all classified as "thermoforming" .While there are many variations on each of the listed methods, there are only a few basic types of forming process:

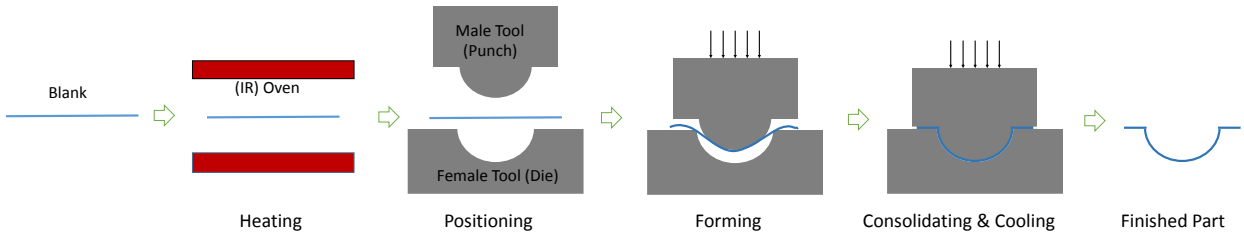


Figure 1.1: Diagram of the stamp-forming process

Press Forming One of the simplest versions of thermoforming (called stamping, press-forming, or stamp-thermoforming) involves using a set of matched dies to press the hot blank to the desired shape. This process, analogous to sheet-metal stamping, can occur very rapidly with cycle times nearing five minutes[7]. In addition, this method has the added benefit of being able to apply significant pressures to the blank in order to form complex geometries.

Roll Forming For forming long members with a constant cross-section, roll-forming is the method of choice. Roll forming is a cross between stamping and extrusion where a continuous sheet of material is heated and run through a series of matched rollers until it reaches the desired shape[8]. The biggest advantage of this method is that it allows for nearly continuous

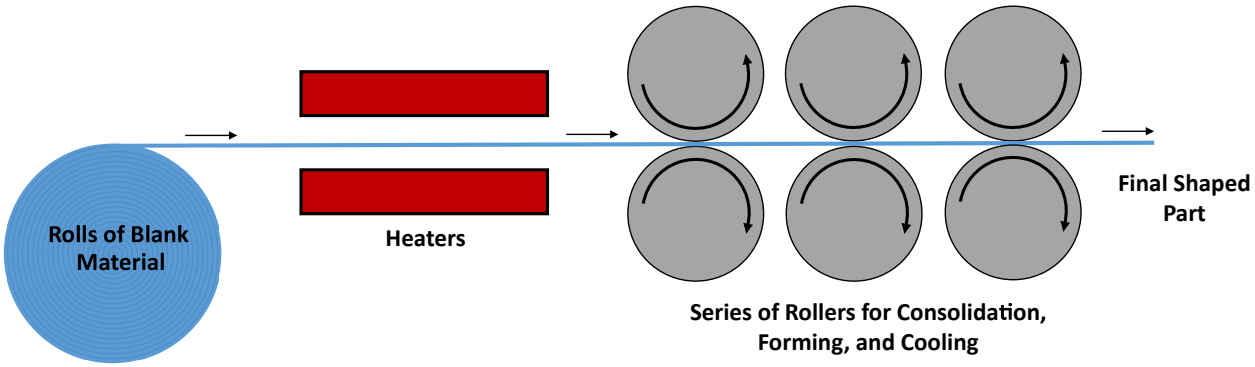


Figure 1.2: Diagram of the roll-forming process

manufacture of any open-contoured prismatic shape. This advantage, however, is also the largest drawback of roll-forming, since it is restricted to members of constant cross-section, and therefore cannot form much in the way of complex shapes.

Vacuum Forming In the same way that press-forming and roll forming are adaptations of previously existing techniques (metal stamping and extrusion respectively), vacuum forming is an adaptation of some previously existing techniques for thermoset composites, as well as some similar techniques for neat plastics (also called vacuum forming). In this process, the blank is placed on a one-sided mold, and a high-temperature vacuum bag (or rubber

membrane) placed over it, and sealed. By heating the system (either using a heated mold[9] or an autoclave/oven [5]) and pumping the air out of the bag, the surrounding atmosphere will force the blank to conform to the mold. The two largest drawbacks of this method are the reliance on manual blank placement which slows the process, and the reliance on vacuum which limits the applied pressure to that of the surrounding atmosphere. This pressure limit, while it can be improved by using a pressurized autoclave, severely limits the complexity of shapes which can be formed, and the quality of final consolidation[5].

While each method has benefits, the inherent simplicity and versatility of the stamp-forming process make it the technique of choice for research purposes. In addition, since another objective of this research group is to demonstrate automation, the ability of this method to be adapted readily to automated processing was also a major consideration in the decision to adopt stamp-forming.

1.2 Simulation

Since stamp thermoforming is such a promising technology, it is valuable to develop methods that allow its use to become more widespread. One of the largest drawbacks to press-forming (as well as other thermoforming techniques) is the difficulty in forming complex shapes without defects. Trouble occurs because these composites are nearly inextensible in the fiber directions (due to the high stiffness of the fiber reinforcements). This forces the material to conform to a shape almost exclusively by shear deformation, which is problematic because at high levels of shear, wrinkles can develop which compromise the structural integrity of the part. These kinds of defects can be avoided in a variety of ways:

- By adding tension to the blank using one or more blank-holders[5].
- By altering process parameters like temperature and press-speed.
- By modifying the forming process to include multiple stages.

- By changing the part shape to one more easily formed.

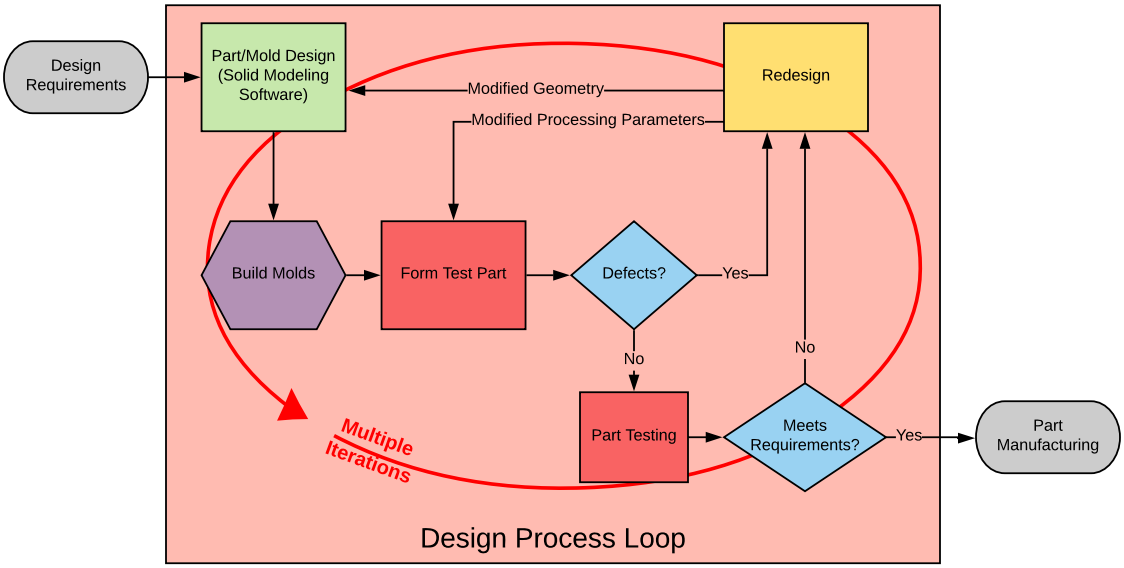


Figure 1.3: Flowchart of the traditional thermoforming design process

Traditionally, these methods of mitigating defects have been implemented in a "trial and error" type manner as shown in Figure 1.3. This can be costly and time consuming due to the presence of physical manufacturing within the design-process loop(shown purple and red). In addition, it requires a high level of intuition developed from significant experience. In order to increase process efficiency, then, the trial and error method must be replaced with something more definite. Since accurate computer models have been used in a variety of other sectors of manufacturing to mitigate similar issues, development of modeling tools has been considered as the most promising method of minimizing these defects. A useful thermoforming simulation must be able to accurately predict the behavior of a preform blank under the forming conditions. This process can be used to predict important parameters like fiber reorientation, fiber-volume-fraction, residual stresses, and even the thermal process history of the polymer (this is particularly valuable when dealing with semi-crystalline polymers whose mechanical properties depend on their thermal cycle). Along with this, simulations can also predict manufacturing defects like wrinkling, bridging, and spring-in. In addition to the immensity of the predictive data, Figure 1.4 shows that including simulations into

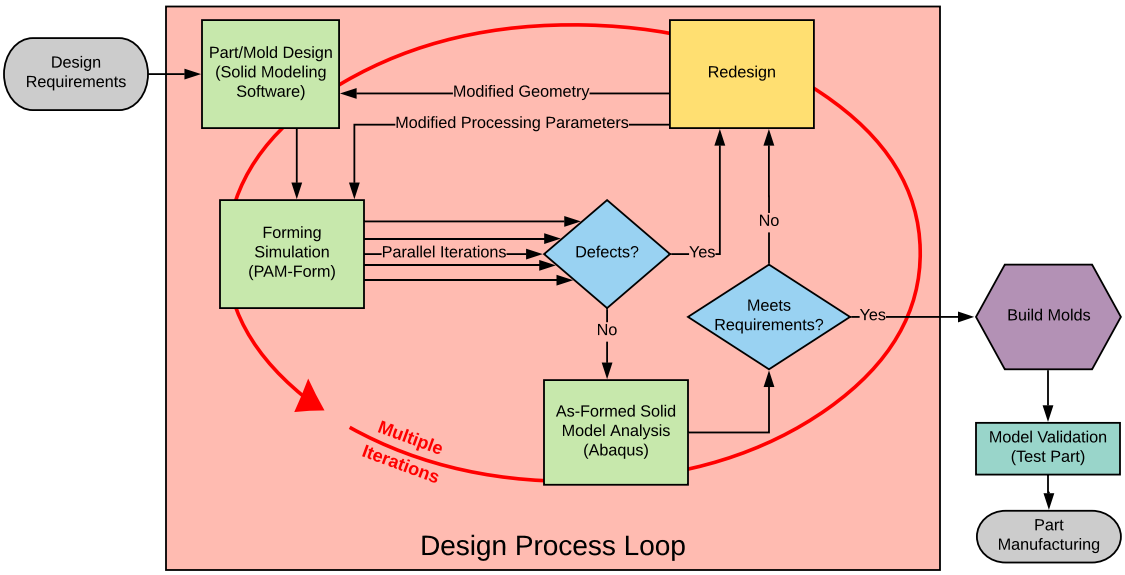


Figure 1.4: Flowchart of the simulation-based design process

the design flow allows the costly manufacturing processes to be removed from the repetition of the loop. Since manufacturing (particularly of molds) is a time consuming and expensive process[10, 11, 12], this has the potential to greatly improve design time and cost. Finally, the ability to run many different simulations together allows for further optimization of processing parameters (including things like fiber architecture/layup, temperature profile, and forming speed) to reach the best outcome in a single step simply by choosing the best result-set.

1.2.1 Mechanical Behavior

Due to the nature of thermoplastic composites, modeling the forming behavior is necessarily a complex process. Some simplified models have been used for dry layups, and while these cannot be readily adapted to thermoplastics, understanding of them has been a stepping stone for more complex models

Kinematic Model

A kinematic model relies solely on the geometry of the forming process for woven fabrics. It assumes that the fibers are inextensible, but can rotate freely about the crossover points. This type of model can be very useful for blank sizing and fiber tracking, but the lack of stress/strain/force calculations limit its usefulness in modeling the actual forming process[13].

Membrane Model

A membrane model assumes the sheet deformations to be restricted to the plane of the fibers. These models demonstrate in-plane-shear as a driving factor in the forming process, and can be well implemented by experimental characterization of the material under tensile and shear loads. This type of model can be used to accurately predict the strain-field, and can generally indicate wrinkling by determining critical locations where the sheet is under either compressive strain, or shear that is above the fabric-locking angle. Wrinkling behavior, though, is tied closely to the out-of-plane bending stiffness of the material, so these in-plane calculations cannot fully predict wrinkling locations, and are completely incapable of predicting wrinkle size & shape[14]

Finite Elements

The next step in analytic detail is a full 3-D finite element analysis. Finite element analysis (FEA) means that the part in question is simulated as a series of interconnected "elements". These have a simple geometry and load-state, allowing their behavior to be more readily calculated. The behaviors of each element can then be put together to determine the state of the entire part. Since FEA elements can be created which account for both in and out-of-plane behaviors, this type of analysis can fully simulate wrinkling, bridging, and internal stress under forming conditions.

1.3 State of the Art

While there have been efforts to create new methods of simulating thermoforming, much of the work in recent years has been focused on characterizing material properties to be used in existing FEA simulation tools designed for thermoforming (such as PAM-Form or AniForm). Despite the large amounts of work on this topic, it is still unclear what method of characterization is best suited for thermoplastic materials to be simulated. Likely, there is no single best method. Each type of test has advantages and disadvantages which makes each better suited for certain situations. Some of the different tests are discussed in Section 2.2.

1.4 Scope of Research

The purpose of this research is to verify the characterization and simulation processes for thermoforming manufacturing. In particular, the focus of this research is on optimization of the automated manufacturing process through the use of simulations. In addition, the further utilization of the forming simulations is investigated by using the simulated "as-formed" part in a solid model in order to more accurately predict the final part behavior. This can be summarized as:

- Develompment of Material Characterization Method
- Validation of Dynamic Properties
- Comparison of Simulations and Forming
- Use of Forming Models to Generate As-Formed Structural Models

CHAPTER 2
MATERIAL CHARACTERIZATION

The accuracy of a simulation is heavily dependent on the accuracy of the material properties that are input, therefore, it is necessary to accurately characterize a material before simulating its forming. Ideally, this would mean compiling data on every possible behavior of the material in question. Practically, though, a model that detailed is an unreachable goal, and the characterization process becomes a balancing act between accuracy and measurability. This chapter is a summary of a set of characterization methods which have been developed in order to gather the necessary properties to model unidirectional fiber-reinforced thermoplastics using the PAM-Form simulation software.

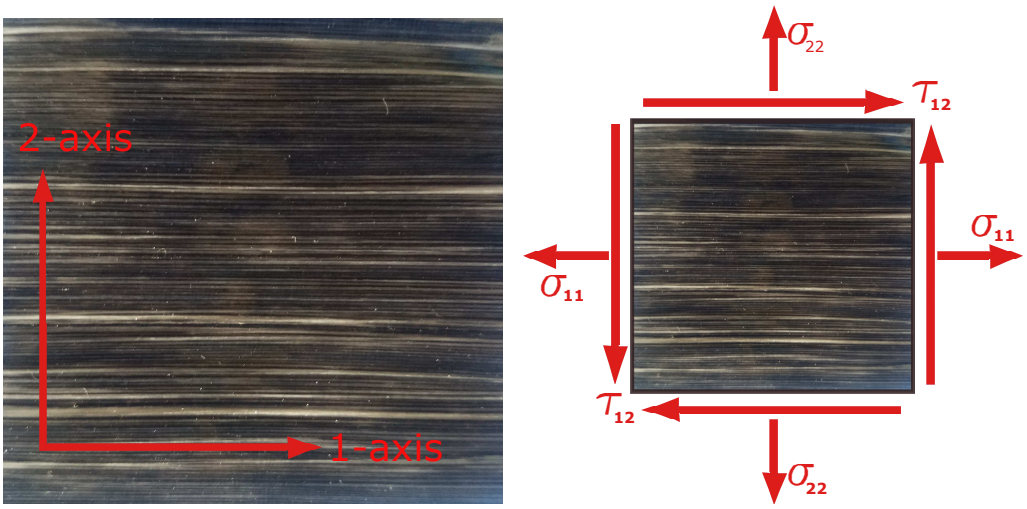
2.1 Required Properties

PAM-Form has a series of necessary and optional properties that can be defined in order to produce an accurate model. The majority of the properties investigated are in-plane properties of the laminae (layers). The directions discussed are shown in Figure 2.1, and are described as follows:

- **Fiber Direction** refers to the stresses and strains parallel to the fibers in the material (σ_1 and ε_1) and are governed by the longitudinal stiffness E_1^t
- **Transverse** refers to the stresses and strains perpendicular to the fibers in the material (σ_2 and ε_2) and are governed by the transverse stiffness E_2^t
- **Shear** refers to the shearing stresses and strains observed in the 1-2 plane (τ_{12} and γ_{12}) and are governed by the shear stiffness G_{12}

In this chapter, we will discuss several critical properties that are necessary for a PAM-Form Model. These are:

- In-Plane shear modulus (G_{12})



(a) Uniderectional lamina (b) Stresses in a unidirectional lamina

Figure 2.1: Unidirectional composite: stresses in material coordinates

- Tensile moduli in the fiber and transverse directions (E_1^t & E_2^t)
- Bending moduli in the fiber and transverse directions (E_1^b & E_2^b)
- Friction coefficient (μ)

2.2 Shear Stiffness

Due to the high stiffness in the fiber direction, continuous-fiber reinforced composites must rely on their ability to deform in an in-plane shear mode (shear strain γ_{12}) in order to conform to complex geometries [3]. This phenomenon can be avoided by using chopped-strand, knitted, or stretch-broken (aligned discontinuous) fibers. These are generally not used in high performance applications due to their decreased mechanical properties. Some stretch-broken-fiber materials, however, are being shown to have strength values comparable to equivalent continuous-fiber systems.[4, 15, 16]

Since shear deformation is such an important factor in thermoforming, it is critical to have proper representation of this behavior. The shear properties of thermoplastics can be determined using a variety of tests whose data can be used in any of the applicable models. Some of the commonly used characterization methods are:

- Picture-Frame Test
- Bias-Extension Test
- Torsional Rheometry

Picture-Frame Test The Picture-Frame test (a trellis-frame test) is commonly used to find the shear deformation behavior of fabrics. The test is done by clamping the fabric to a pin-jointed frame. By pulling on opposite corners of the frame using a standard tensile test machine, the fabric is forced into a pure rhombus-type shear deformation. Figure 2.2 shows how this process works.

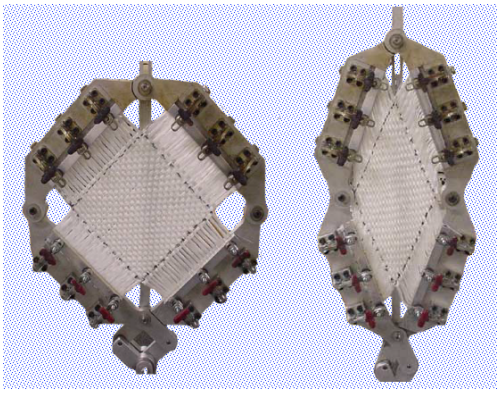


Figure 2.2: Example of a picture-frame test.[1]

One advantage to this type of test is that it induces pure shear deformation equal to the nominal deformation of the frame itself [18] which can simplify data analysis by allowing the measurements to be directly implemented without significant analysis. Another is that the large sample size can help to minimize small-scale effects such as those caused by spatial material variations, or microstructural defects. On the other hand, this test is very sensitive to edge/clamping effects and misalignment [13]. It is also more difficult to implement for unidirectional materials due to tensioning and ply-splitting effects [19, 20]. Also, since the process temperature of thermoplastic composites (at which the models must be run) is well above ambient, a costly environment chamber (fitted to the tensile tester) is required.

[1]Adapted from [17] with permission from Elsevier

Bias-Extension Test Unlike the Picture-Frame, the Bias-Extension test requires no specialized equipment, but rather relies on simple test grips in a standard tensile test frame. By aligning a biaxial sample 45° from the direction of applied force, shear deformation can be induced in the sample [21]. Due to the complex combination of tension and shear, however, data from this type of test requires significant analysis to isolate the shear behavior, and make it usable. Also, this test cannot be directly implemented to characterize unidirectional materials. Finally, like the picture frame, an environment chamber is needed for process-temperature characterization.

Torsional Rheometry A rheometer is an instrument generally used to characterize the viscosity of liquids as they change with temperature and shear rate. This is generally done by placing a small sample of the liquid between two parallel plates, and rotating them relative to one another. This "Torsional Rheometry" can also be used to characterize the shear stiffness of unidirectional materials. In these tests, a rheometer is used to dynamically induce shear deformation in a sample through torsion. This dynamic test utilizes an oscillating torsional deformation to gather a continuous stream of data on both elastic and viscous responses of the material. Since torsion is primarily a shear mode of deformation, the shear properties can be determined from the response. Also, most torsional rheometers include temperature, amplitude, and frequency controls, which means that the time and temperature dependence of the material can be easily tested with no additional equipment. Difficulties arise in that, like in bias extension tests, significant data processing is required to isolate the desired properties [19]. This was the test chosen, as it best fit our current capabilities, and the required equipment was already in-house. In addition, since it doesn't require costly environmental chambers, but can instead rely on the rheometer's own environment control, this option is highly cost effective.

Torsion Tests Torsion tests were performed on a Bohlin Gemini II rheometer using the extended-temperature cell (ETC) in order to achieve the desired testing conditions. This

machine can be used to run an oscillatory strain-controlled deformation while sweeping through different temperatures or frequencies. Tests were first performed using the temperature sweep, and then later using frequency sweeps at the various temperatures in order to fully capture the behavior and inform a data lookup table for the simulation.

(a) Bohlin rheometer with ETC

(b) Rheometer solids fixture

Figure 2.3: Photos of the torsion test setup

The samples used were unidirectional polypropylene/E-glass composite manufactured by consolidating 18 layers of PolyOne prepreg unidirectional tape (material code IE 6034) hereafter referred to as PP/Glass. Consolidation was achieved using an infrared oven and press. In order to achieve good consolidation, an aluminum frame was used to hold the edges in place and prevent fiber movement. In addition aluminum caul sheets (covered with non-stick material) were used to hold everything together during transport. The resulting panels ere then cut into samples using a CNC waterjet cutter. After cutting, the samples were then

conditioned in a standard laboratory environment ($23°C$ and 50% humidity) for a minimum of 40hrs in accordance with ASTM D618-13 Procedure A [22], and then dimensioned. The width and thickness were measured using a micrometer (precision ± 0.001mm), and the length with a caliper. This sample length is less critical, as the gage-length of each sample is determined by the rheometer settings (to micron precision).

Sample size varied depending on the test, but generally was in one of two categories with a 35mm gage section:

- 10mm by 4mm rectangular prismatic section.
- Dogbone sample with a 4mm by 4mm square prismatic gage section.

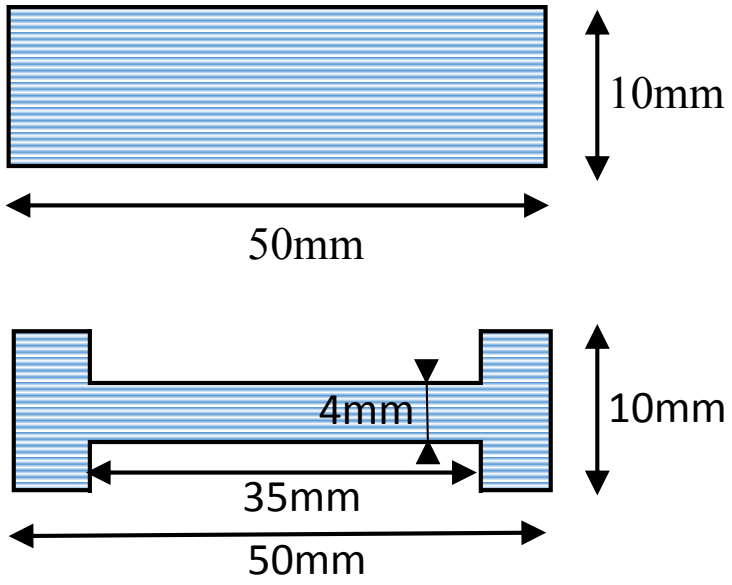


Figure 2.4: Nominal dimensions of the two test types

Torsion Results From these tests, there are several pieces of information that can be found by inspection of the data. The data of interest here is the storage modulus G', which, while not exactly the same, is analog to the material's modulus of rigidity (the shear modulus G). The fist thing to observe in the plot of the storage modulus versus temperature in Figure 2.5 for a heating-cooling cycle, is that the general shape of the plot is similar

to those for many neat thermoplastic polymers. In addition, the difference between the upper (heating) portion of the curve and the lower (cooling) portion appear to be due to the crystallization kinetics within the composite. This may be due to the fact that crystallization (on cooling) occurs at a lower temperature than melting (on heating) as is sometimes seen in semi-crystalline polymers. Finally, the spread between datasets in the cooling curve in comparison with the heating curve appears to indicate that initially all of the samples have the same crystallinity (as would be expected, since these are pieces from the same panel), but upon cooling, the slight changes in cooling rate, and internal architecture cause the samples to diverge as they form different levels of crystallinity. When analyzing the heating data

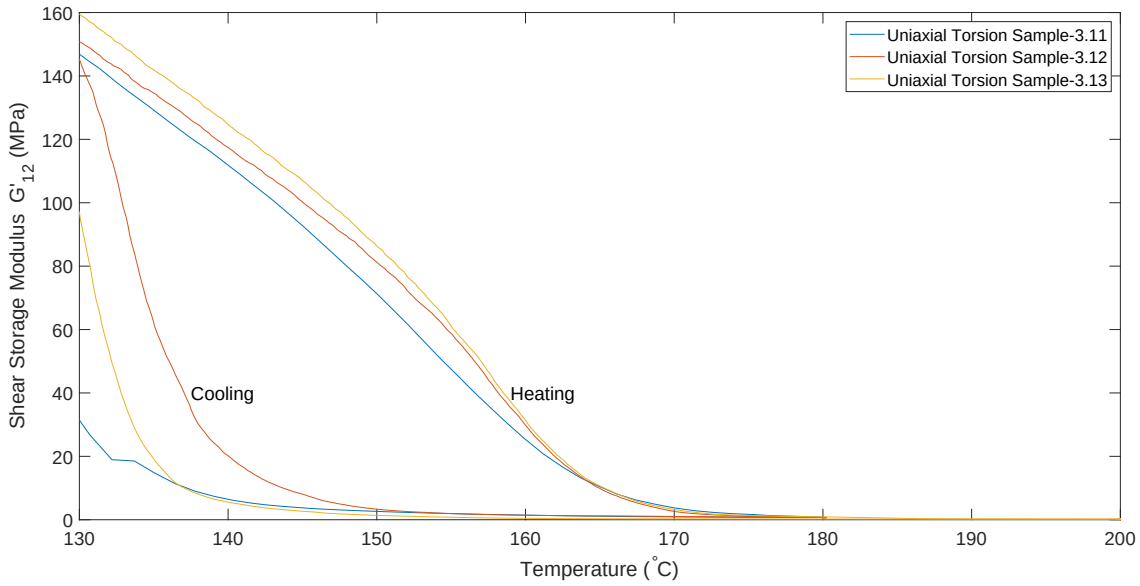


Figure 2.5: Heating and cooling response of PP/Glass composite

gathered using samples from different panels (the cooling portion of these curves has been removed for clarity), it is clear that there are significant differences between each panel. In Figure 2.6 , we can see that each dataset from any given panel varies only slightly from the others, whereas the differences between panels is quite large. While it is possible that the difference seen in panel 4 is geometry driven (since these were dogbone specimens), the fact that panels 3 and 5 are so different indicates that there is something deeper happening. It is speculated that the variation between panels may be driven by one of two things:

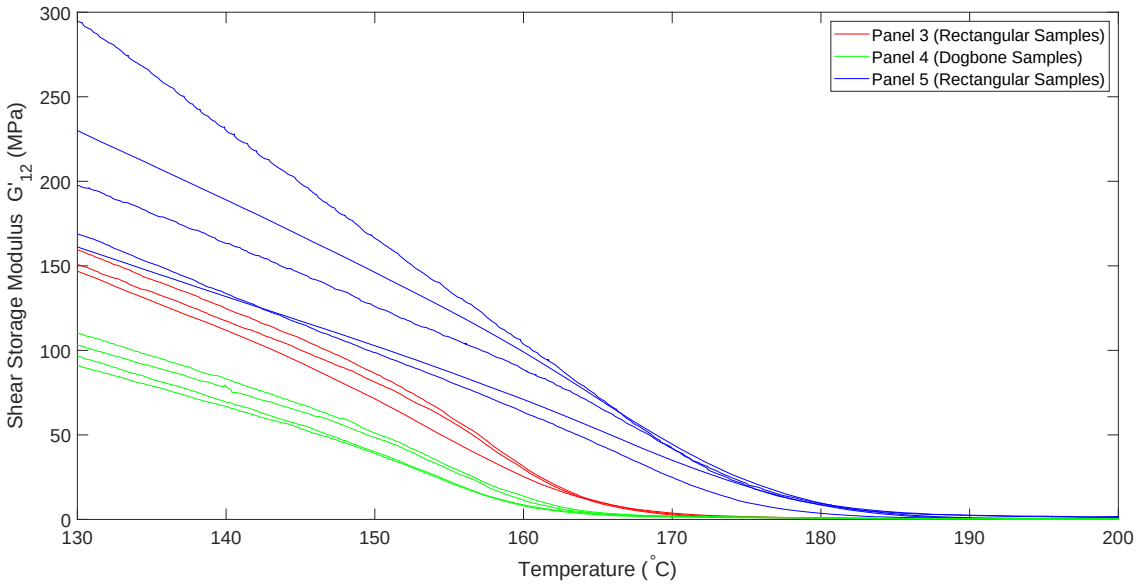


Figure 2.6: Shear stiffness for samples from different panels

1. A change in crystallinity due to minute changes in cooling rate at the time of initial panel consolidation. This could only be mitigated by an overhaul on the press equipment in order to fully control the temperature profile for the consolidation.

2. A difference in moisture content despite adhering to ASTM D618-13 Procedure A [22] before testing. This would be mitigated by strict adherence to a testing schedule, limiting the time that samples remain untested after the mandatory 48 hours acclimatization.

For the purposes of this research, It has been decided to simply take the data from the individual panels as separate datasets. In order to simplify, all data from each panel has been averaged to present a single curve for use in the modeling efforts in Figure 2.7.

Finally, Figures 2.8 and 2.9 show that while there is some variation in the shear modulus with respect to strain amplitude and oscillation frequency (strain rate), this variation is comparable in magnitude to the spread between different panels. Since this interaction is relatively weak, It has been decided to ignore them for the purpose of this research.

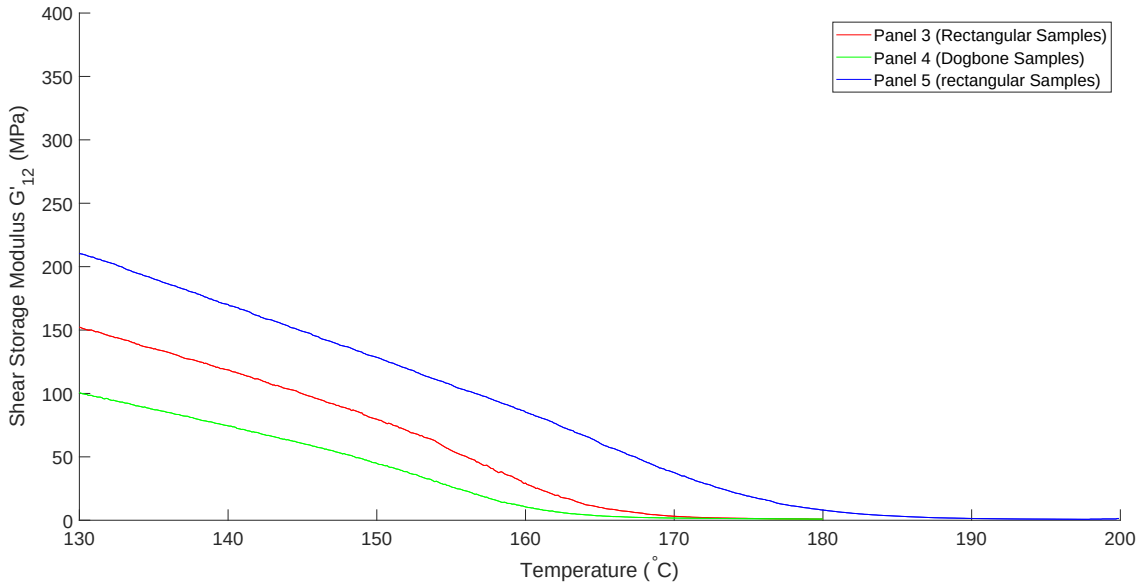


Figure 2.7: Average shear stiffnesses for various sample panels

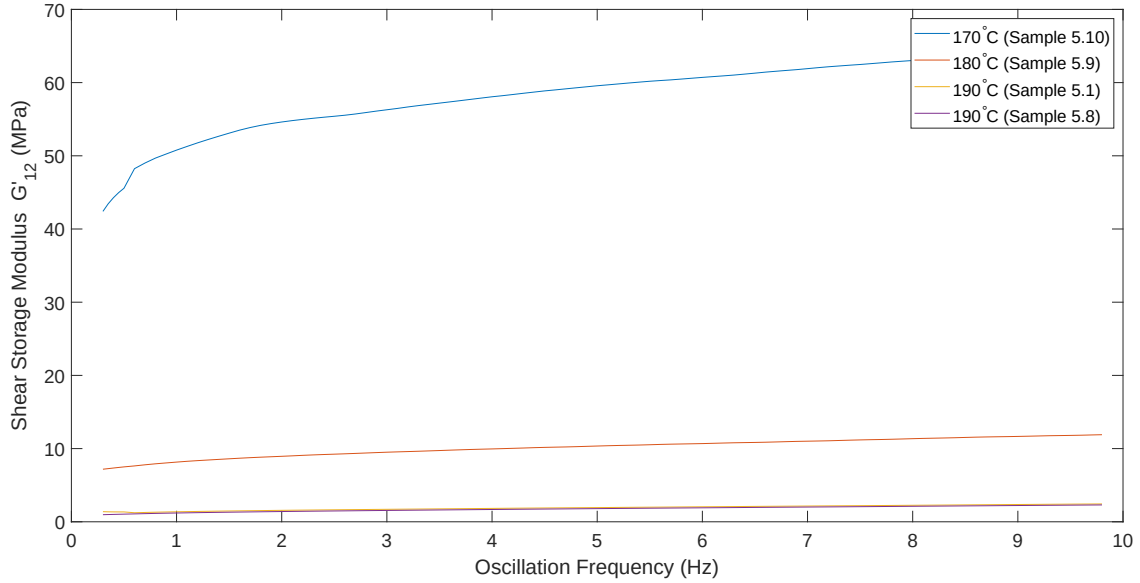


Figure 2.8: Shear stiffness variation with frequency

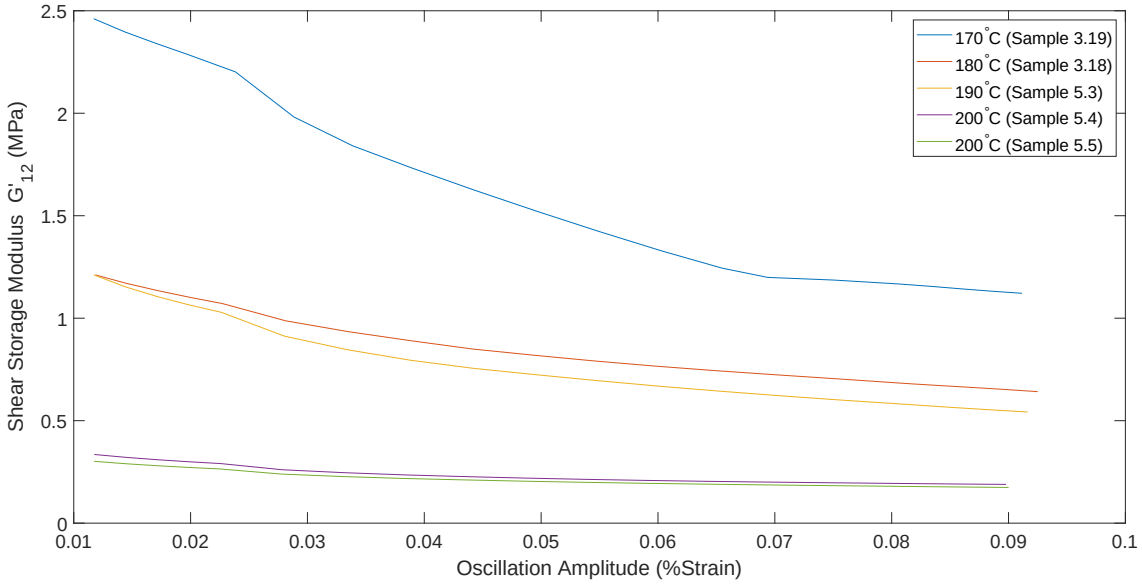


Figure 2.9: Shear stiffness variation with strain amplitude

2.3 Tensile Stiffness

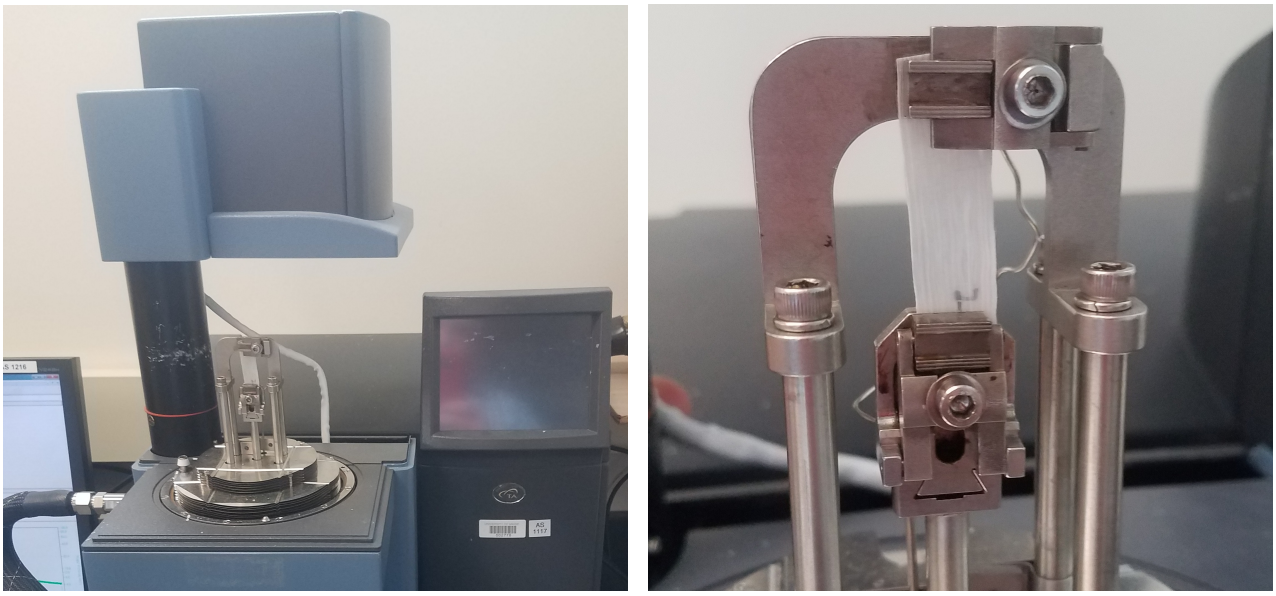
The fiber direction tensile stiffness in these composites tends to be orders of magnitude higher than the shear stiffness at processing temperatures, thus mandating that the material will deform primarily in shear. This means that the accuracy of that stiffness is much less critical to the accuracy of the model. It can be assumed that in unidirectionl materials the transverse stiffness will be important to the model, as the transverse direction will have a relatively low stiffness at process temperature. For this reason, measurements must still be made in order to get reasonable values of the tensile modulus in both the fiber and transverse directions.

2.3.1 Dynamic Mechanical Analysis (DMA)

In order to measure these properties, and for similar reasons to those discussed in Section 2.2 regarding the rheometer, A dynamic mechanical analysis (Q-800 DMA by TA Instruments) machine has been used. This machine, like the rheometer, utilizes an oscillating deformation, and gathers a continuous stream of data on the material response as several parameters -most importantly temperature- are varied. The only difference being that the DMA machine applies a tensile load rather than a torsional one.

Material and Test Setup For the tension DMA, thin film samples were manufactured by hot-pressing single tapes of the same PP/Glass. This pressing was necessary in order to remove the spool-curl in the material, as well as to smooth the surface for more reliable thickness measurements. Several samples were made to the largest allowable size (8mm x 30mm) in both fiber and transverse directions. The samples' thicknesses were measured using a micrometer, and the width/length measured optically using a photo microscope.

Fiber Tension Results Figure 2.11 shows the tensile stiffness results from the fiber-direction DMA testing. The first thing to note is these curves have a similar shape to those in the shear data(Figure 2.5). This is due to the fact that matrix softening is the driving



(a) Q-800 machine with therml chamber open (b) DMA film tension fixture

Figure 2.10: Photos of the tensile test setup

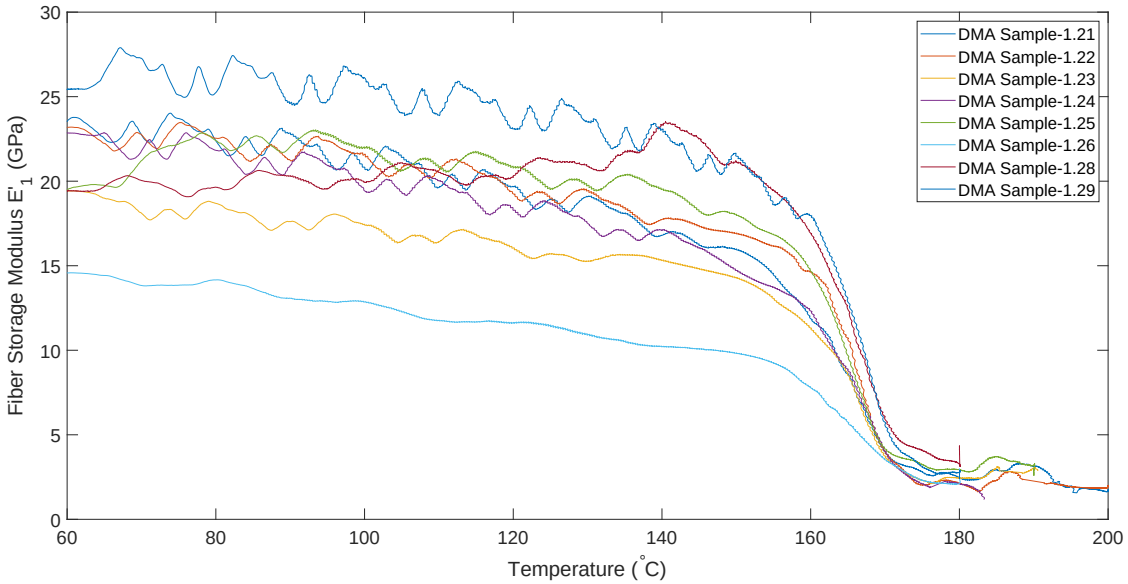


Figure 2.11: Heating response of fiber stiffness

factor for both datasets (also for the other deformation modes, which will be discussed later). Another interesting point is that these curves are about two orders of magnitude higher than those found from shear. This disparity is why some forming models simply treat the fibers as inextensible, since these materials are so much stiffer in tension than shear, any tensile deformations will tend to be trivial in comparison to those seen in shear modes. Like with the rheometry data, an average has been taken over the several samples, and a curve fit to smooth the data, as seen in Figure 2.12.

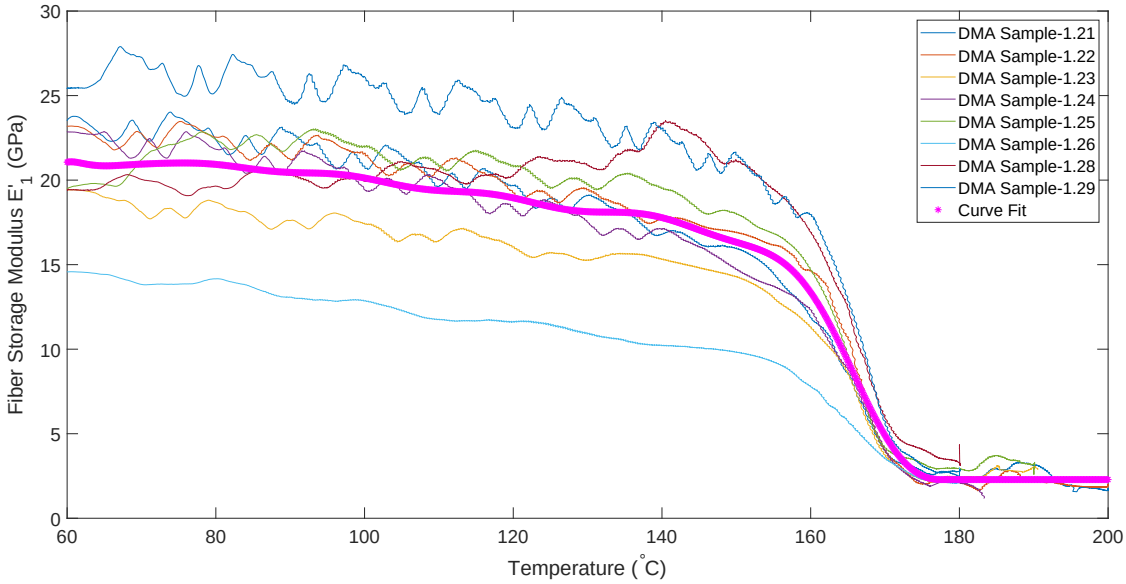


Figure 2.12: Average heating response of fiber stiffness

Transverse Tension Results In the same way, the DMA was utilized for the transverse tensile modulus, the results of which are given in Figure 2.13. It is important to note that these samples, being only a single unidirectional layer, have no reinforcement in the transverse direction. This makes them quite fragile, which is likely the source of the variance in this data. In addition, note that unlike the fiber tension, the transverse tension actually goes to zero at approximately $170°C$. This is due to the fact that transverse tensile behavior is driven by the matrix, so when the polypropylene begins to melt, there is no longer any resistance. For the fiber tension tests, even after the matrix is melted, the fibers still provide some stiffness.

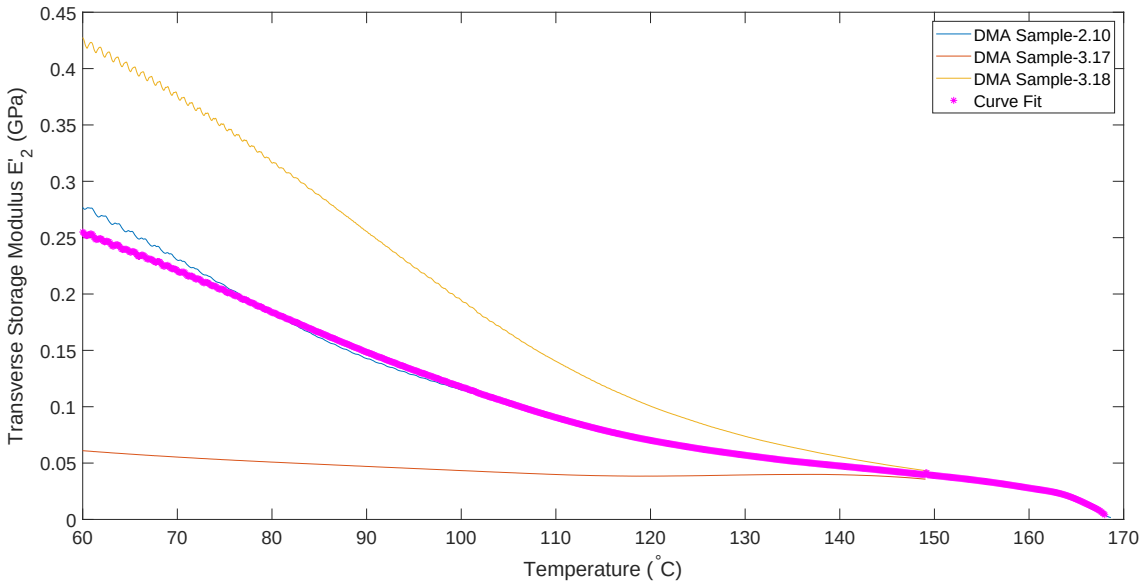


Figure 2.13: Heating response of transverse stiffness

2.4 Bending Stiffness

The stiffness of the individual layers in bending is also an important property, since, while the shear deformation determines the critical areas for wrinkling, the bending is what drives the size and distribution of wrinkles in the high shear areas. Unlike homogeneous materials, however, the bending rigidity of molten composite laminae cannot be directly inferred from the tensile stiffness. Due to interactions between the fibers and matrix as well as fiber migration, the stiffness in bending can often be significantly smaller than the corresponding tensile stiffnesses. This necessitates a separate battery of tests for the bending stiffness of the tapes.

Material and Test Setup Like for the tensile tests, the Q-800 DMA machine was utilized for bending measurements. The fixture used was a 3-point bending fixture (see Figure 2.14). This allows the oscillating force of the DMA to be applied in a bending type deformation rather than directly in tension. This was done for several samples of the largest allowable size for the bending fixture (15mm x 50mm) in both fiber and transverse directions. Like the tension samples, these were manufactured by press-smoothing individual tapes of the

PP/Glass material. Dimensions were also measured as previously, using micrometer and microscope.

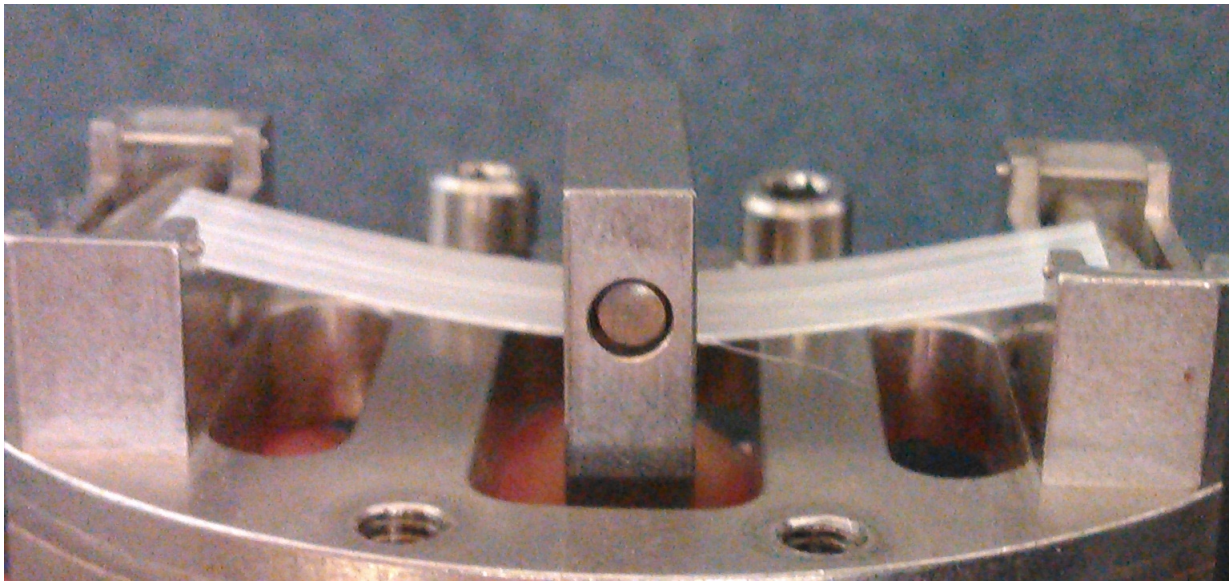


Figure 2.14: 3-point bending DMA fixture

One additional note is that the weakness of this material in the transverse direction precluded any bending tests. It has been reasoned that, since transverse-tensile stiffness and transverse-bending stiffness are both matrix driven, the tensile stiffness can be applied in place of the bending values here.

Bending Results The bending data in Figure 2.15 clearly follows a similar trend to all previous data. In addition, there are a few interesting features. First, notice that most of the curves either drop off at approximately $170°C$, or behave erratically thereafter. This is due to the matrix being molten, and leaving the sample incredibly flexible; too flexible to be stable in the bending fixture. One problem with this is that the typical processing temperature is in the $180°C$ - $190°C$ range, which is clearly higher than this failure temp. By comparing this data with that from the tensile tests, a solution presents itself. Figure 2.16 shows that for much of the heating curve, the bending stiffness is lower than the tensile stiffness, but near the melt temperature the two curves converge. This behavior was utilized in order to predict the bending stiffness at higher temperatures. For this purpose, a curve

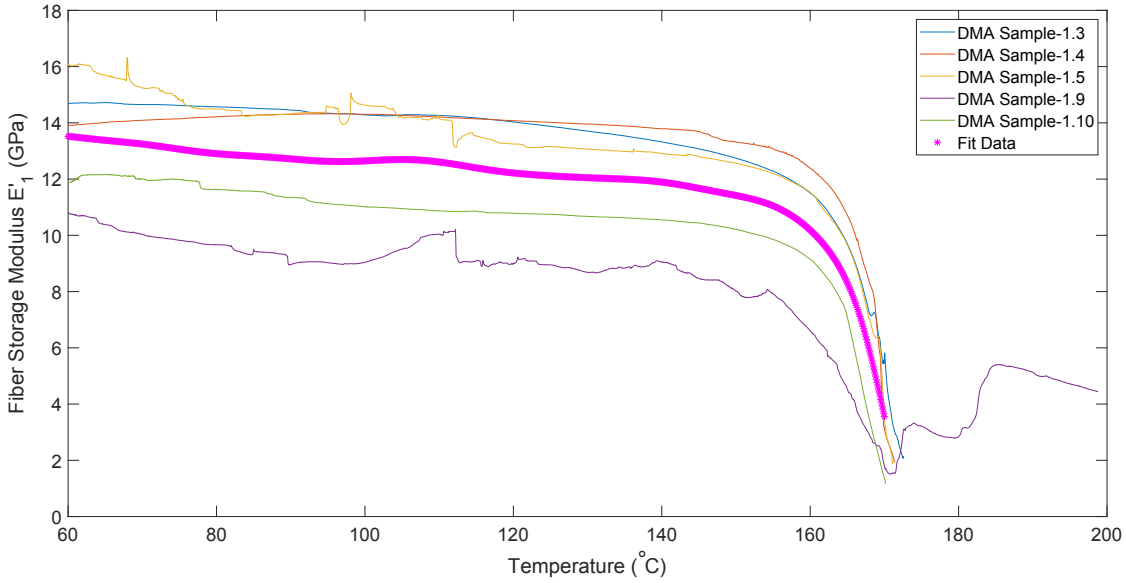


Figure 2.15: Fiber bending stiffness with fit curve

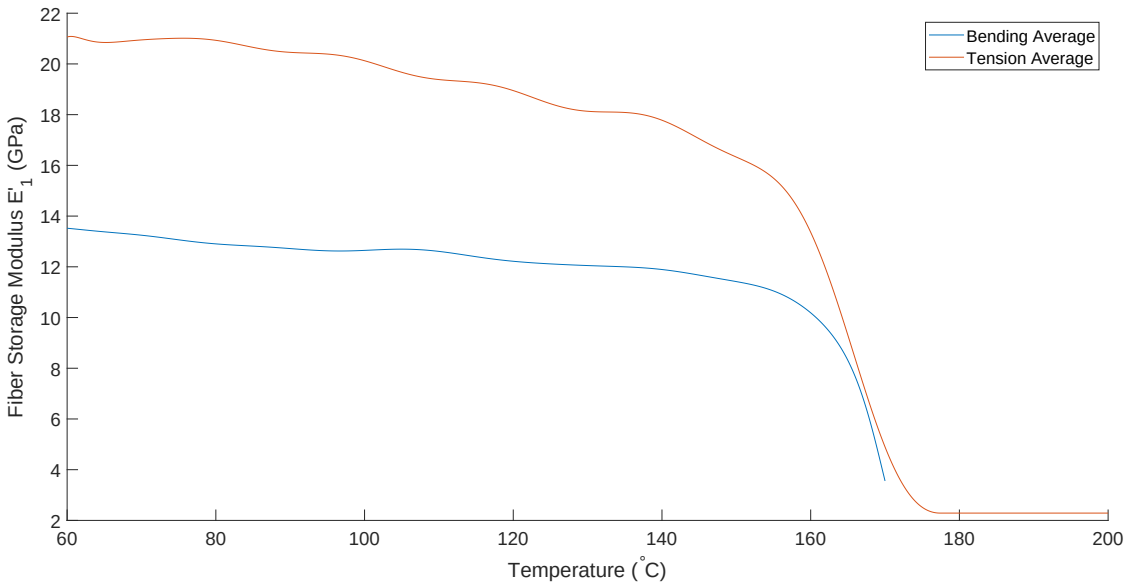


Figure 2.16: Comparison of bending and tensile stiffnesses

was fit which follows the bending stiffness for the lower temperatures, and then smoothly transitions to following the tensile stiffness. This will be the curve used in later forming models (Figure 2.17).

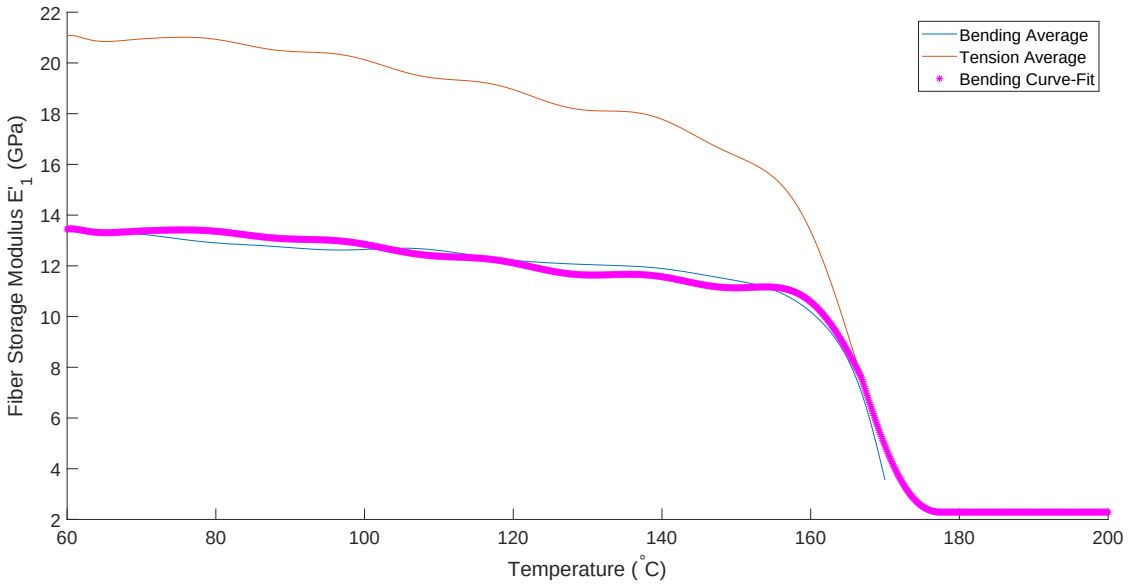


Figure 2.17: Curve fit for bending stiffness

2.5 Contact Interactions

The next important set of properties are those which govern the contact interactions between the layers, as well as between blank and mold. These are very important to the forming simulation, as all of the force transfer between layers occurs by sliding contact. This means that these parameters are a driving factor in most wrinkle formations during processing. In PAM-Form, these interactions are characterized using a coefficient of friction (which can vary with temperature, rate, and pressure), and a separation stress which simulates the gluing (or stickiness) between the molten laminae. Several tests have been developed in order to directly measure these properties including the ply-pullout test for friction[23], and a ply-peeling test for gluing (As described in[24] Adapted from [25]). Due to a variety of constraints, it was decided to bypass direct measurement, and approximate these interaction parameters by utilizing the viscous properties of the neat polymer.

Contact Approximation Contact interactions between fiber reinforced thermoplastics at forming temperatures can become a very complex topic, and has been studied directly by several researchers[26, 27, 28]. For this study, however, a simple hydrodynamic model has

been chosen. By assuming that there is always a layer of matrix between the fiber layers, the traction can be estimated by:

$$\tau = \frac{\eta \cdot v}{d} \tag{2.1}$$

Where:

- τ is the shear traction at the interface
- η is the matrix viscosity
- v is the sliding velocity
- d is the matrix layer thickness

This equation (from [28]) is actually derived directly from the fluid dynamics definition of viscosity, where $\tau = \eta \cdot \dot{\gamma}$. for a parallel plate setup, $\dot{\gamma}$ can be approximated as $\frac{v}{d}$. If we estimate d as 1/2 the nominal lamina thickness, the only unknown here is η, which can be measured using a rheometer.

Rheometry The matrix viscosity was measured by doing an oscillatory torsion test on rectangular neat-polypropylene samples (approximately 50 x 12.5 x 3.4mm) in the same manner as the ply shear tests. The only differences were that here, the samples contained no fiber-reinforcement, and that the parameter of interest was viscosity rather than storage modulus. Figure 2.18 shows that the viscosity was divided into three bands. By coloring the data taken at different oscillation frequencies (Figure 2.19), it became clear that this division is due to frequency changes. This means that a closer look must be taken to the frequency dependence of the viscosities.

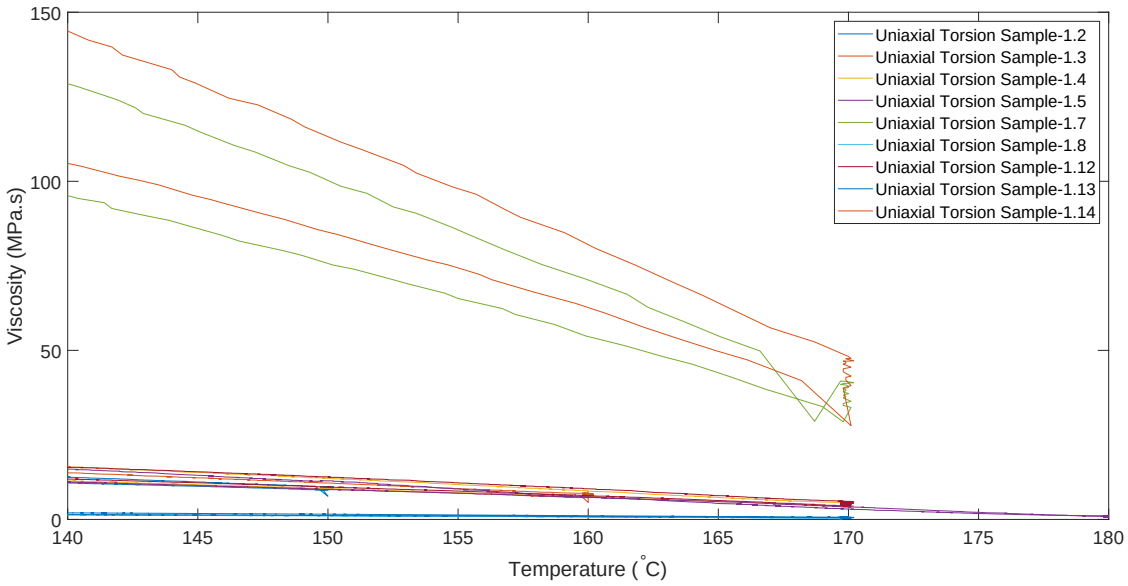


Figure 2.18: Neat polypropylene viscosity vs temperature

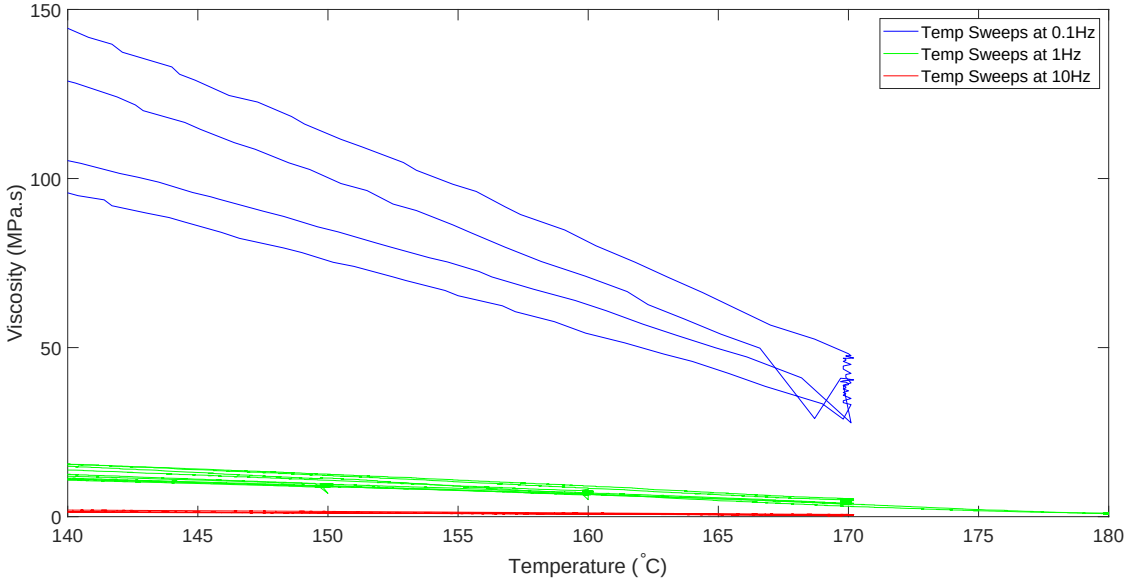


Figure 2.19: Neat polypropylene viscosity colored by frequency

Plotting some frequency sweep data at various temperatures(Figure 2.20) confirms a very heavy dependence on frequency, however, the details of this relation were unclear until the plots were switched to log-log scaling in Figure 2.21. The data then formed straight lines on the plot. Straight lines on a log-log plot mean that there is a linear relation between

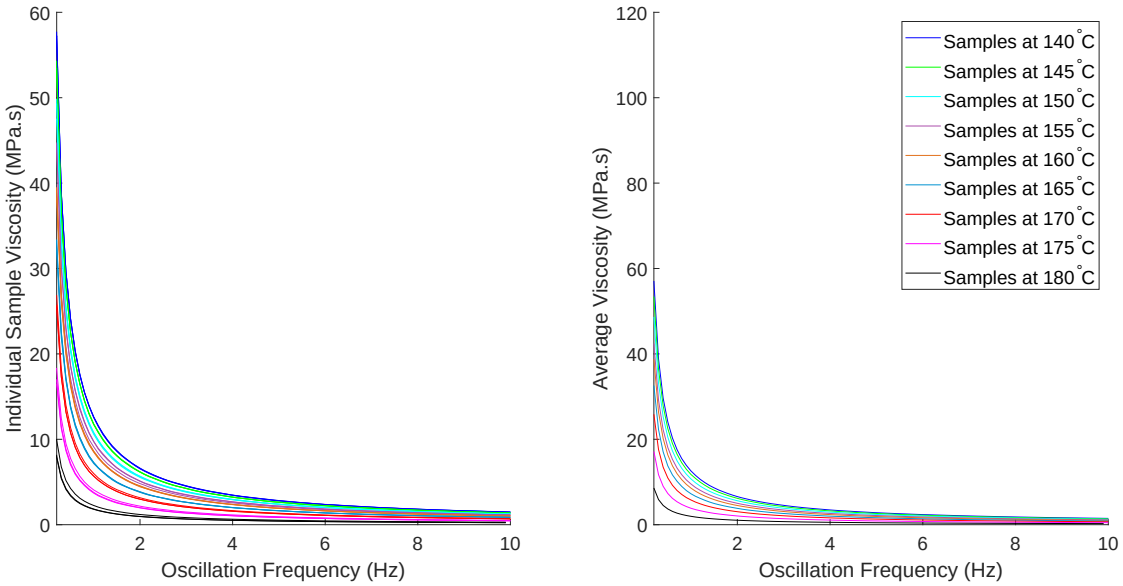


Figure 2.20: Various frequency sweep data

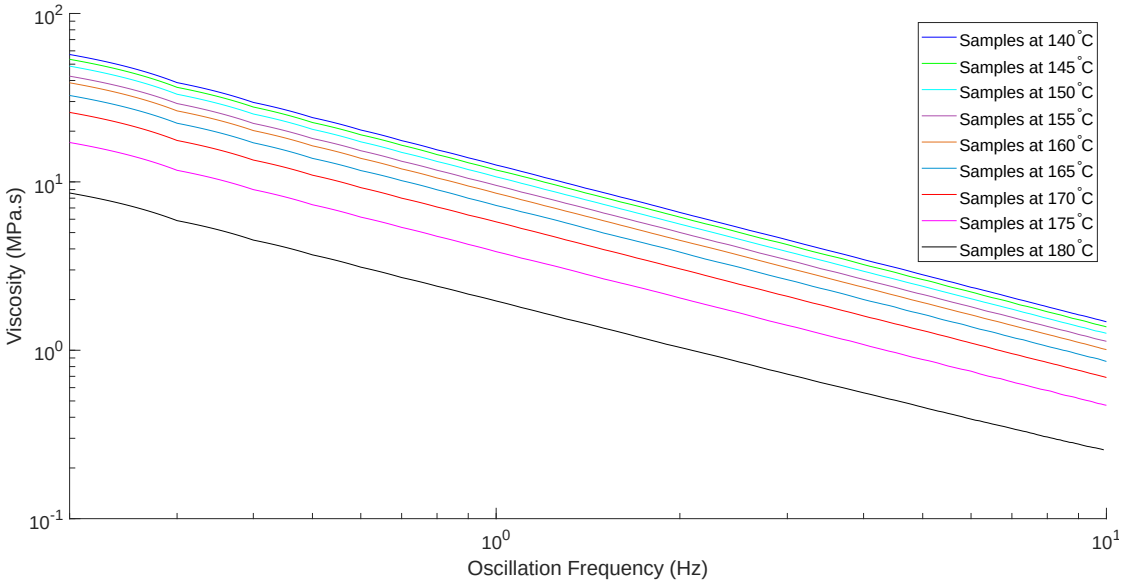


Figure 2.21: Various frequency sweep data on log-log axes

logarithms of the data. This will take the form:

$$ln(\eta) = M \cdot ln(f) + b \tag{2.2}$$

Rearranging using the properties of the natural log yields a power law relation between viscosity and frequency of the form:

$$\eta = \bar{C} \cdot f^{M} \tag{2.3}$$

Where $\bar{C} = e^b$ and M can be determined from the slope and intercept of the log-log lines. Using a linear regression (MatLab function "polyfit"), the values of C, and M were found for the various temperatures studied. Looking a the various values, the exponent M is nearly constant (at -0.926). This matches the data, since M also represents the slope of the log-log plots, and looking at the graph, we can see that the various temperature lines are effectively parallel. The C value, on the other hand, follows a decreasing curve which, when plotted against temperature, can be well approximated by a quadratic function (see Figure 2.22) This

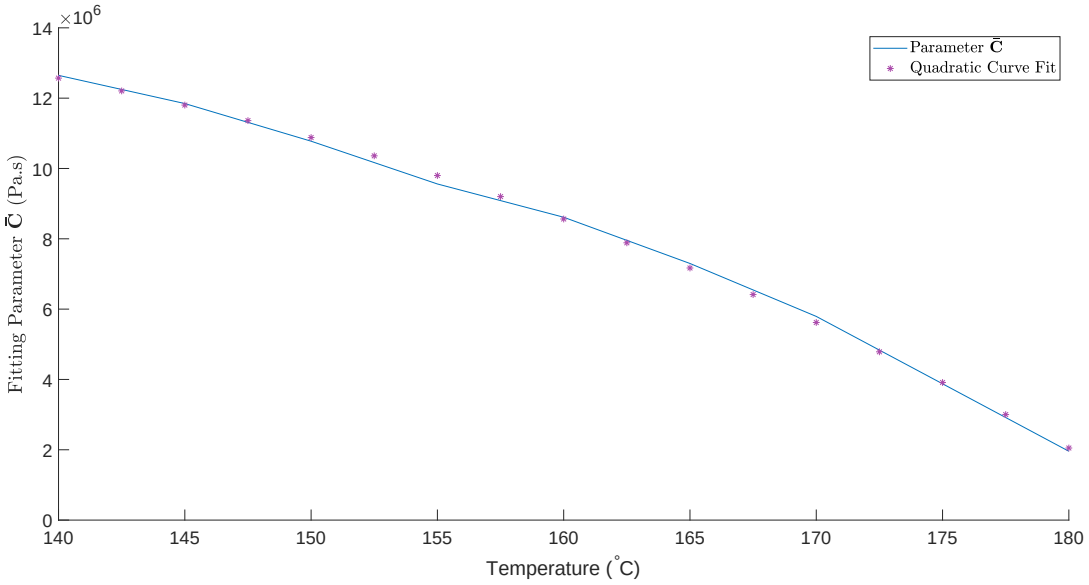


Figure 2.22: Plot of the function parameter $\bar{C}$

relation, then, can be used to fully represent the dependence of viscosity on both frequency and temperature as the equation:

$$\eta(f, T) = (A \cdot T^2 + B \cdot T + C) \cdot f^M \tag{2.4}$$

Where A,B,T,and M are fitting parameters shown in Table 2.1 Using this fit curve, comparisons made between the fit, and original data in Figure 2.23 show good correlation.

Application Using the measured data, a relation has been created where viscosity can be obtained from temperature and oscillation frequency. In the forming model, however, there is no oscillation, but rather steady state sliding motion, so how do these correlate? The

A	$-3133\left(\frac{Pa.s^2}{^\circ C^2}\right)$
B	$7.395 \times 10^5\left(\frac{Pa.s^2}{^\circ C}\right)$
C	$-2.956 \times 10^7\,(Pa.s^2)$
M	-0.9260

Table 2.1: Curve fitting parameters

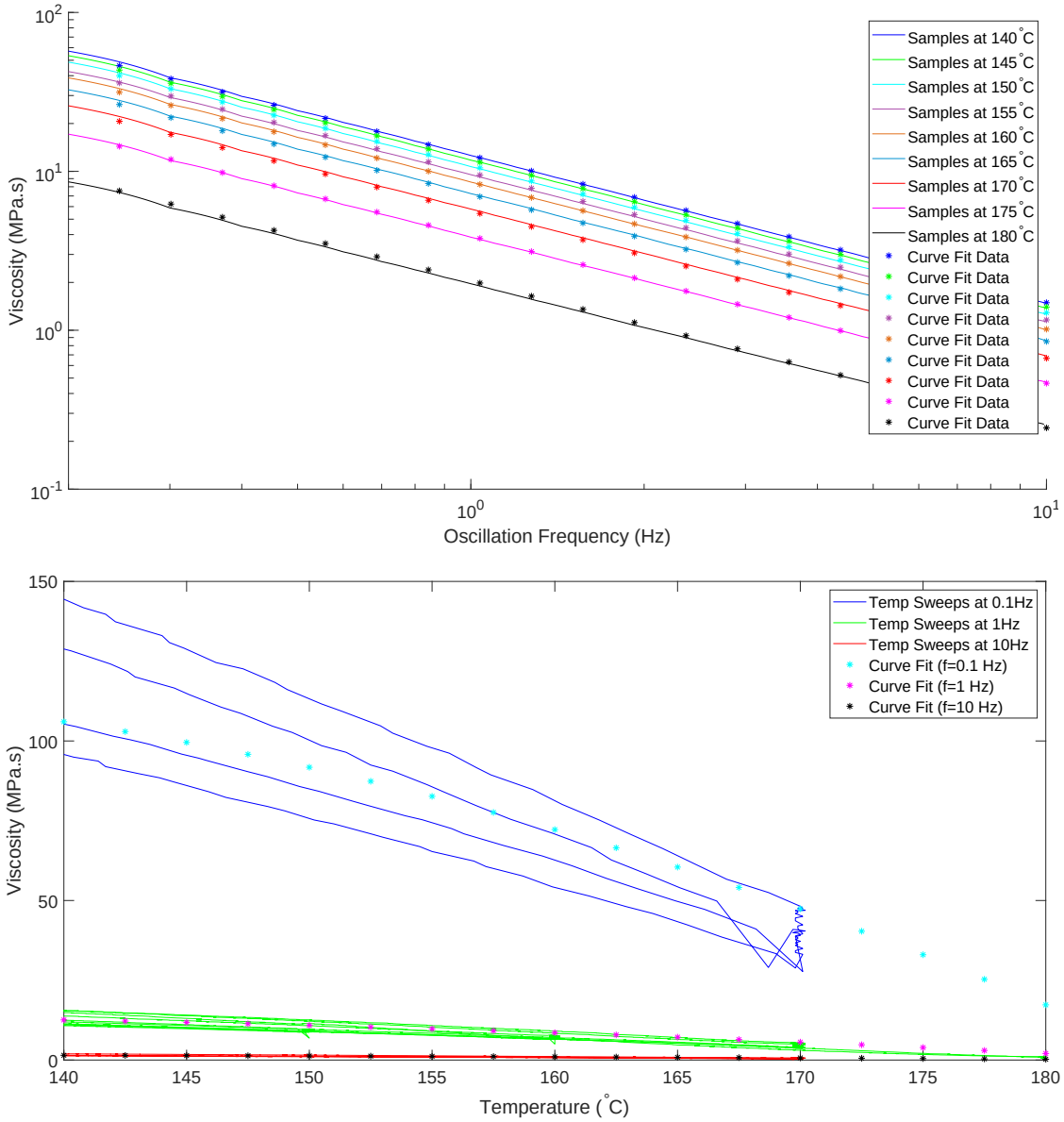


Figure 2.23: Curve fits of viscosity data

Cox-Merz Rule [29] states that viscosity as a function of shear rate ($\dot{\gamma}$) corresponds directly to viscosity as a function of angular frequency (ω) This relation (Equation (2.5)) is then combined with the fit curve in Equation (2.4) to yield Equation (2.6).

$$\eta(\omega) \approx \eta(\dot{\gamma}) \tag{2.5}$$

$$\eta = C(T) \cdot (f)^M = C(T) \cdot \left(\frac{\omega}{2\pi}\right)^M \approx C(T) \cdot \left(\frac{\dot{\gamma}}{2\pi}\right)^M \tag{2.6}$$

The relation in Equation (2.6) can then be applied to the friction relationship from Equation (2.1) to yield Equation (2.7), which simplifies to Equation (2.8).

$$\tau = \frac{\eta \cdot v}{d} = C(T) \cdot \left(\frac{\dot{\gamma}}{2\pi}\right)^M \cdot \frac{v}{d} \tag{2.7}$$

$$\tau = \frac{C(T)}{(2\pi)^M} \cdot \left(\frac{v}{d}\right)^{M+1} \tag{2.8}$$

2.6 Thermal Properties

In many forming applications,the temperature changes quite gradually, and the thermal gradients (and therefore property variations) are quite small; so small, in fact, that these cycles can be accurately simulated isothermally (that is: with constant temperature). For some forming cycles, however, the variation of temperature (and subsequently properties) within the blank becomes an important factor. This is most critical in high-speed applications where the heating-cooling cycle is very fast, generating high temperature gradients. Simulation tools can account for these thermal variations, and even calculate the temperature distribution throughout the process, thus increasing accuracy for these cycles. In order to utilize the thermal functionality in PAM-Form, there are several additional material properties needed (for both the blank and mold). Here is a list of the minimum required material parameters:

- Thermal Conductivity of the Blank

- Thermal Conductivity of the Mold
- Thermal Conductivity of the Blank-Mold Interface
- Specific Heat of the Blank
- Specific Heat of the Mold

The tests required, particularly for the interface conductivity, beyond the scope of this research. For this reason, it was decided to keep the simulations simple by treating them as isothermal.

2.7 Conclusion

In this chapter, the characterization of material using dynamic methods were discussed. The experimental methods presented provided material properties for modeling:

- Intra-Ply Shear Stiffness
- Fiber Tensile Stiffness
- Transverse Tensile Stiffness
- Fiber Bending Stiffness
- Transverse Bending Stiffness
- Inter-Ply Sliding Resistance

Using this data, a material model will be implemented for PAM-Form simulations in Chapter 3, in order to verify both the modeling methods, and the material data.

CHAPTER 3
FORMING SIMULATIONS

3.1 Introduction

The forming behavior of thermoplastic composites is governed by the complex interaction of the fiber-structure and the matrix viscosity. In order to model this fully, using finite element analysis, one must model everything at the meso-scale (the scale of the fiber structure). This level of detail would make such a simulation too slow for practical uses, especially for simulating the forming of large parts. Commercial forming simulation software bypasses this issue by using the effective properties of the overall composite. By using these "smeared" or "average" properties, the behavior of a composite can be modeled using much larger elements, and therefore be sufficiently fast to be practical in industry. Some alternative methods for bypassing the level of detail required include the kinematic models discussed in subsection 1.2.1.

3.1.1 PAM-Form

PAM-Form by ESI-Group, is an explicit finite element tool derived from ESI's dynamic crash simulation tool (PAM-Crash). This software contains a number of additional tools that make it quite valuable for forming simulations, including some advanced contact modeling and a designated composites material model (MAT140). This model simplifies the simulation by neglecting interactions between the different deformation modes. This means that shear doesn't affect tension or bending, and tension or bending in one direction don't affect those in other directions. It behaves exactly like an analytical model of an anisotropic material with poisson's ratio of zero ($\nu = 0$). By treating these deformation modes as separate, it means that each has its own modulus (stiffness), and is governed entirely by that modulus. Were there interactions between the deformation modes, additional factors would be needed to quantify this relationship. While modeling in this way is not technically correct, it has

been shown to produce accurate results, and has the advantage of further simplifying the process. This improves both setup and computational efficiency, making simulation more practical in industry.

This decoupled modeling works well for elastic materials, but gets a bit more complicated for viscoelastic behavior. In the case of a material with relevant viscoelasticity, there are two options that can be used for modeling. The first makes use of the theoretical detail of the behavior. In this setup, the moduli are defined as constants, but a (possibly temperature dependent) viscosity is also defined. This allows the use of a Maxwell material model in order to simulate the viscoelastic interactions of the material. The viscosity must be determined using some sort of rheological testing (relaxation tests can be used as well), and then the other moduli must somehow be determined in absence of the viscosity, or inferred by extensive data-processing.

The second setup defines the moduli as functions of strain rate and temperature directly, without using a viscosity term. This allows the viscous nature of the material to be characterized without having to do any explicit characterization of the viscosity. Instead, simple elastic tests are performed in tension and shear at varied temperatures and strain-rates in order to create a "lookup table" of the stress-strain curves at each of those states. An interpolation scheme is then utilized in order to fully define the material, allowing the instantaneous stiffness values to be determined at each model time-step. Thus, a material element is modeled simply using only the moduli, but at each time-step the moduli are updated (from the lookup table) in order to reflect the current temperature and strain rate.

$$\varepsilon_{11} = \frac{\sigma_{11}}{E_1(\varepsilon_{11}, \dot{\varepsilon}_{11}, T)} \tag{3.1}$$

$$\varepsilon_{22} = \frac{\sigma_{22}}{E_2(\varepsilon_{22}, \dot{\varepsilon}_{22}, T)} \tag{3.2}$$

$$\gamma_{12} = \frac{\tau_{12}}{G_{12}(\gamma_{12}, \dot{\gamma}_{12}, T)} \tag{3.3}$$

Where:

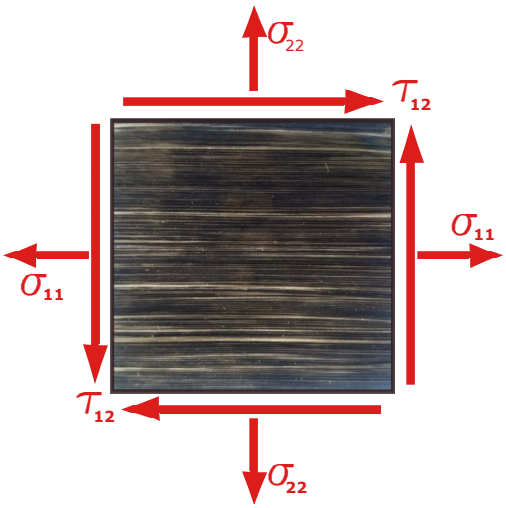


Figure 3.1: Stresses in a unidirectional lamina

ε_{11} & σ_{11} are the strain and stress in the fiber direction

ε_{22} & σ_{22} are the strain and stress in the transverse direction

γ_{12} & τ_{12} are the shear strain and shear stress

E_1 & E_2 are the moduli in the fiber and transverse directions respectively. These may be functions of strain (ε), strain rate($\dot{\varepsilon}$), and temperature (T).

G_{12} is the shear modulus, which may be a function of shear strain (γ), shear rate ($\dot{\gamma}$), and temperature (T).

From Figure 3.1 and equations. (3.1 – 3.3), we can see the decoupled model in action. The strain in each direction is governed by only the modulus in that direction, and are completely independent of the other direction's stress or strain. The moduli may also be functions of other variables like strain, rate, and temperature. These variations are implemented in a lookup table, wherein the values of the moduli for each element are chosen based on its current state.

The second characterization method, while it is more empirical than theoretical, is much more straightforward to implement in the PAM-Form software (partially due to intrinsic tools for this sort of modeling). There are fewer types of tests for characterization, and there is less data-processing required in order to achieve a working model[30]. This straightforward implementation made the empirical model the best for the purposes of this research.

Regardless of the choice of modeling method, this tool can allow a full simulation of the material behavior under a variety of forming situations. In addition, the advanced contact modeling tools makes it possible to simulate forming of multi-layered laminates with inter-ply slip. This advanced contact modeling capability is one reason that explicit solvers are sometimes preferred over the (technically more accurate) implicit solvers.[31]

One difficulty in using a software like PAM-Form is that the models can occasionally become unstable. In some cases, the material properties interact with the geometry such that the model deformation goes to infinity unpredictably. Since these instabilities appear to exist within the solver itself, it is difficult to predict when they'll occur, and equally difficult to resolve them. In addition to the unpredictable instabilities, another issue that occasionally arises is that where significant wrinkling or excessive stress occurs, the model will crash. It cannot account for the extremely large stresses and rotations involved, and so stops calculation. Unlike the other type of model failure, these are still quite informative, since wrinkles large enough to crash the solver are significant defects in a real part.

3.2 Case Studies

A variety of forming simulations have been done in this thesis, both to demonstrate the capabilities of the PAM-Form software, and as validation tools to check the correctness of the material characterizations. These parts cover a wide range of different classes of thermoforming, from the relatively simple curvature of the sinusoidal corrugation to the more complicated curvature of a hemisphere and even a real-world automotive part: the differential-case cover.

3.2.1 Sinusoidal Corrugation

This part was designed by Sunil Bhandari[10, 11, 12] as a tool to demonstrate feasibility of using a 3D-printed mold for stamp-thermoforming (see Figure 3.2). This part was designed to represent the web section of a corrugated-web beam. Using a corrugated part like this in

the web of an I-beam (Figure 3.3) increases the stability in the flanges, by forcing them to remain parallel. In addition to showing of feasibility, this part was also used in the process of commissioning thermoforming equipment. Due to the design shape, the curvatures

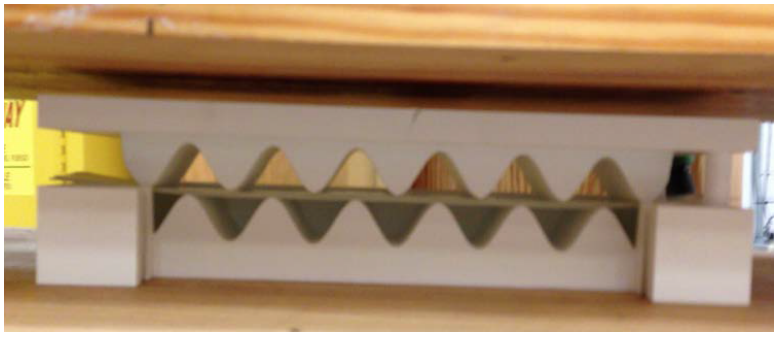


Figure 3.2: corrugation being thermoformed [10]

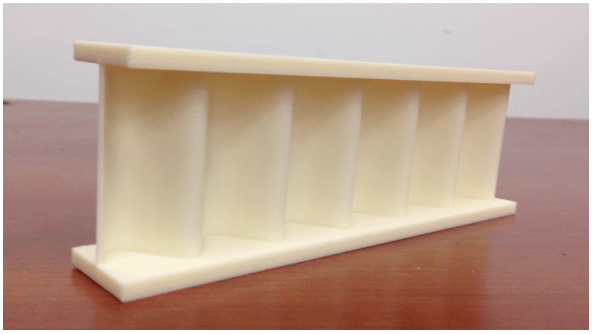
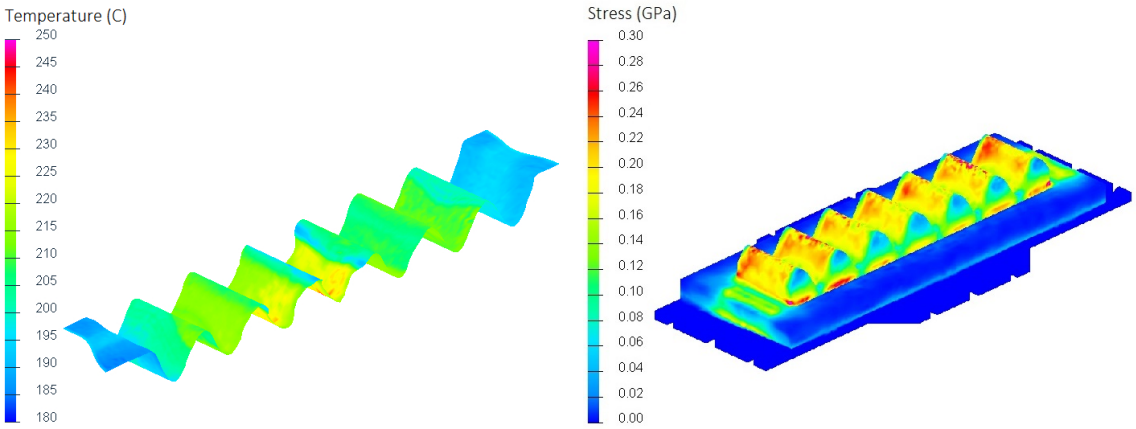


Figure 3.3: Model of a corrugated-web beam [10]

in this part are entirely single, thus requiring no shearing to form. The part, then, was simply utilized as a known geometry with which to learn and demonstrate the capabilities of PAM-Form.

Since this geometry was utilized primarily as a demonstration and learning tool, the material properties used were those given in an ESI provided example material (a woven fabric detailed in Appendix B) with a layup of $[0_4]$ which was 4mm thick. Forming of the part was then done under a variety of conditions, and modeling types ranging from isothermal to temperature variable to use of a non-rigid mold (like a 3D printed one). The models used a 679mm (26.75") long corrugation with amplitude of 25.4mm (1") and wavelength of 101.6mm (4"). For temperature varying models, the blank had an initial temperature of

215°C and the mold temperature was held constant at 30°C. Some of the resulting contours are shown in Figure 3.4



(a) Temperature contour (b) Mold stress contour

Figure 3.4: Results from various corrugation simulations

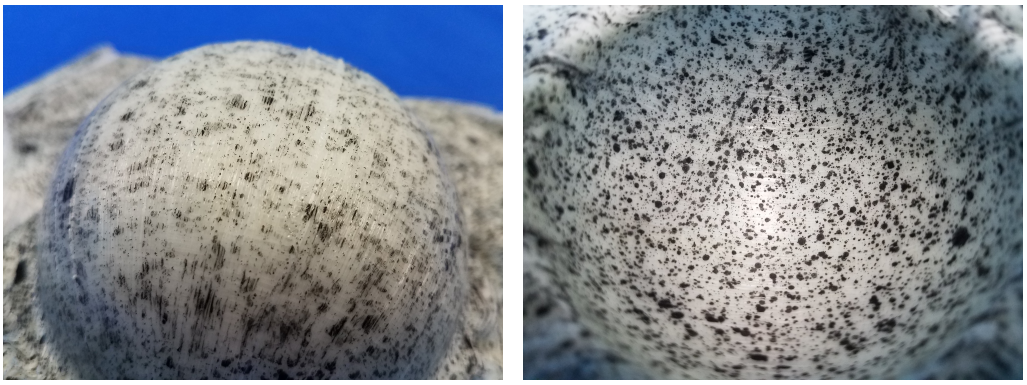
3.2.2 Hemisphere

For the purpose of forming, single-curvature parts like the sinusoidal corrugation are rare, whereas the majority of real parts contain areas of double curvature (that is locations where curvature occurs in multiple directions). For this reason, it is important to simulate the forming of doubly curved parts. A hemisphere is a good case study in that it contains double curvature, while retaining a simple shape. The hemisphere to be formed was 76.2mm diameter (3”). This size achieved a good balance between material usage, and deformation visibility, in order to optimize the process. By trial and error, the layup and processing parameters were optimized to achieve good consolidation and shape. For this study a simplified manual process was utilized whereby a preconsolidated blank with a layup of $[(\pm 45°)_5]$ was placed in a convection oven set at 200°C for 5 minutes. The softened blank was then carefully transferred to a 10-ton pneumatic press, and stamped between a set of hemispherical aluminum molds shown in Figure 3.5. Because of the setup of these molds, there is nothing to restrain the shape at the intersection between the hemisphere and the flat region, so that the material in this region is quite wrinkled. This issue could be somewhat

Figure 3.5: Hemispherical aluminum molds

mitigated by providing the mold with a "shoulder" to constrict this region once the mold closes. Due to this issue, the comparisons from model to forming will be restricted to the hemisphere itself, and the adjacent regions will be ignored.

In order to compare the forming to the simulation, it was necessary to accurately measure the deformation of the blank during forming. This was done using the Aramis digital image correlation (DIC) system. This system relies on a stochastic dot pattern on the surface of the object. Aramis tracks the positions of each dot (done by comparing photographs of the object), and uses this information to generate displacement and strain fields. Since the dot pattern is generally applied as paint on the object surface, there is the potential for degradation when a large deformation occurs (like in the case of thermoforming). Such pattern degradation can make Aramis ineffective. Fortunately, the pattern generally stays intact for some portions of the object, thus allowing us to measure at least some data from most samples. One example of the degradation is shown in Figure 3.6(a). As the picture shows, the entire crown of the part has been stretched such that the Aramis pattern is unreadable. Fortunately, the concave side of the part does not see as much stretching, and can therefore be used for measurements.



(a) Degraded Aramis pattern (b) Useable Aramis pattern

Figure 3.6: Good and bad Aramis patterns

After analysis using the Aramis software, a nearly complete strain field has been measured. The shear angle (shown in Figure 3.7) is another way of expressing shear strain. It is simply the shear strain, which is typically described in radians, converted to degrees to make it more intuitive. This data will be discussed more fully in Section 3.2.4.

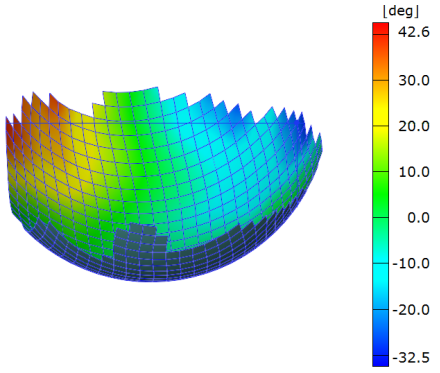


Figure 3.7: Shear angle field in a hemispherical part

3.2.3 Differential-Case Cover

In the process of commissioning the ASCC's thermoforming equipment, a test part was needed in order to begin forming parts. The cover for an automotive differential was chosen as a good demonstration of equipment capabilities, while remaining simple enough for an initial commissioning test part. This part was designed to replace the original metal cover on a Chrysler 9.25" differential model (taken from a 1998 Dodge Dakota). The cover went through

several phases of design and testing including initial modeling, scale mold forming, and fully automated full-scale forming. Thermoforming was done using three different materials including PP/Glass, recycled-chopped-glass-fiber reinforced, and recycled-chopped-carbon-fiber reinforced polypropylene. The original steel part and thermoformed composite part are seen in Figure 3.8

(a) Original steel part (b) Thermoformed composite part

Figure 3.8: Comparison of differential covers

3.2.3.1 Geometry

The geometry chosen was derived directly from the existing metal part. The original part's dimensions were digitized using a FARO Edge ScanArm® ES (3D measurement device). From this initial measured data, a 3D model was developed, which was then used in creating a mold designed to form the part. The mold was then scaled down to 50% and 3D printed using a Fortus 900mc FDM 3D printer (from Stratasys®) out of Ultem9085® thermoplastic material. Printing at 50% scale allowed us to verify that it was a correct geometry without wasting the large amounts of machine time and material required to print the full mold. In addition, the small mold allowed for an additional amount of forming experience with smaller quantities of composite material(using a $[\pm 45^{\circ}]_{10}$ layup). After the geometry was

verified, the full-scale mold was printed, and a $[\pm 45^\circ]_{13}$ layup was developed (which also included additional patches on critical areas at the crest of the cover). The forming process has been tested extensively for both the full and half-scale parts, and simulations run using PAM-Form for both as well.

Qualitative Results on Half-Scale Part Initial simulations were run using a set of generic material properties modified from an example file from ESI®. Despite these properties being technically incorrect for the PP/Glass material, several of the defects seen in the real part were qualitatively predicted by this model. For example, the separation between tapes within a layer which was visible in early iterations of the process was also visible in the model (see Figure 3.9). The figure shows a PAM-Form model result in which the different plies are rendered in different colors. The top color (blue) is a partial ply that we labeled the "patch" or "bandaid" which was added to help mitigate the ply separation. Next we see the first full layer (pink) which has developed gaps that reveal the underlying layers (ligh blue and another pink).

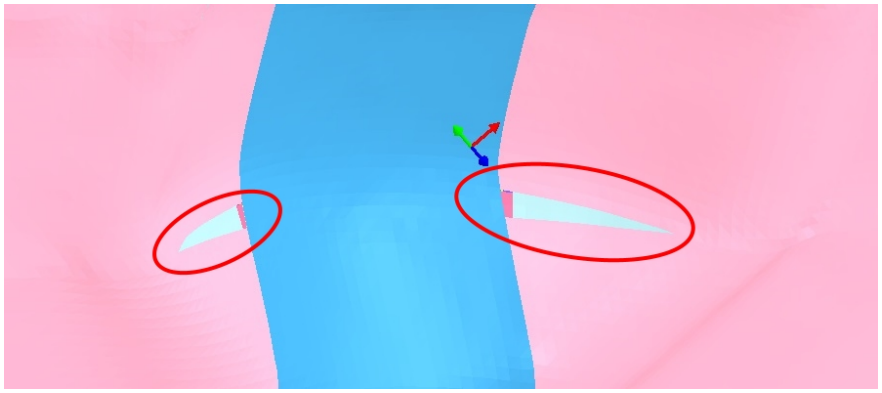
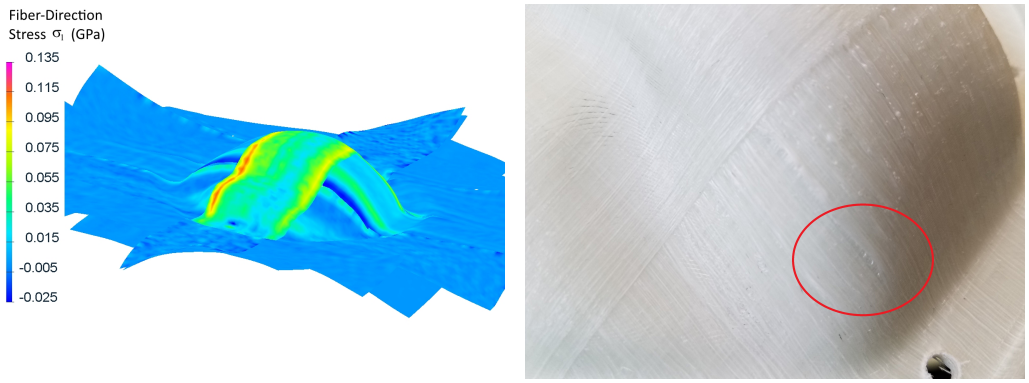


Figure 3.9: Model of tape separation

Another surprisingly accurate prediction was that of stress concentrations in an area found to be one critical for ply splitting/tearing (Figure 3.10). While the actual values of the stresses may not reflect those seen in the real process (remember this is a generic material),

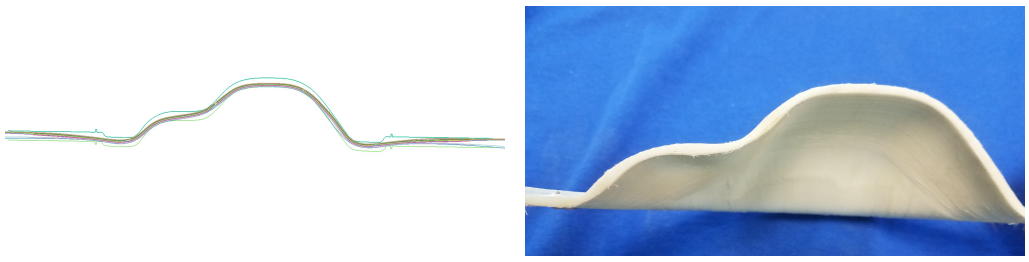
it is helpful to note that patterns, such as concentrations in critical regions, seem to be practically independent of the material properties.



(a) Contour of stress concentrations (b) Photo of layer tearing

Figure 3.10: Comparison between simulated stress and forming defect

The final preliminary qualitative prediction was in the general trend of thickness variation within the part. Upon cutting an example part along its symmetry axis, the variations in thickness are clearly visible. Taking a similar cut in the simulated part shows critical pinching locations in the same vicinity as the thinner portions of the formed part(see Figure 3.11). This again points to the insensitivity of trends to changes in material property. It is assumed that smaller defects such as wrinkles may require an experimentally generated material model to accurately predict.



(a) Model of thickness variation (b) Photo of thickness variation

Figure 3.11: Comparison of simulated and as-formed thickness variation

Qualitative Results of Full-Scale Part The material properties measured (see Chapter 2) were applied to the PAM-Form models of the full-scale differential cover. As with the half-scale covers and the example materials, there is much to be learned from simple inspection of the model results. The first thing that was seen was an unreasonably large amount of tape-separation. This separation, as shown in Figure 3.12 was much more than is ever seen in the actual part, thus indicating that the friction model was not correct. This is not

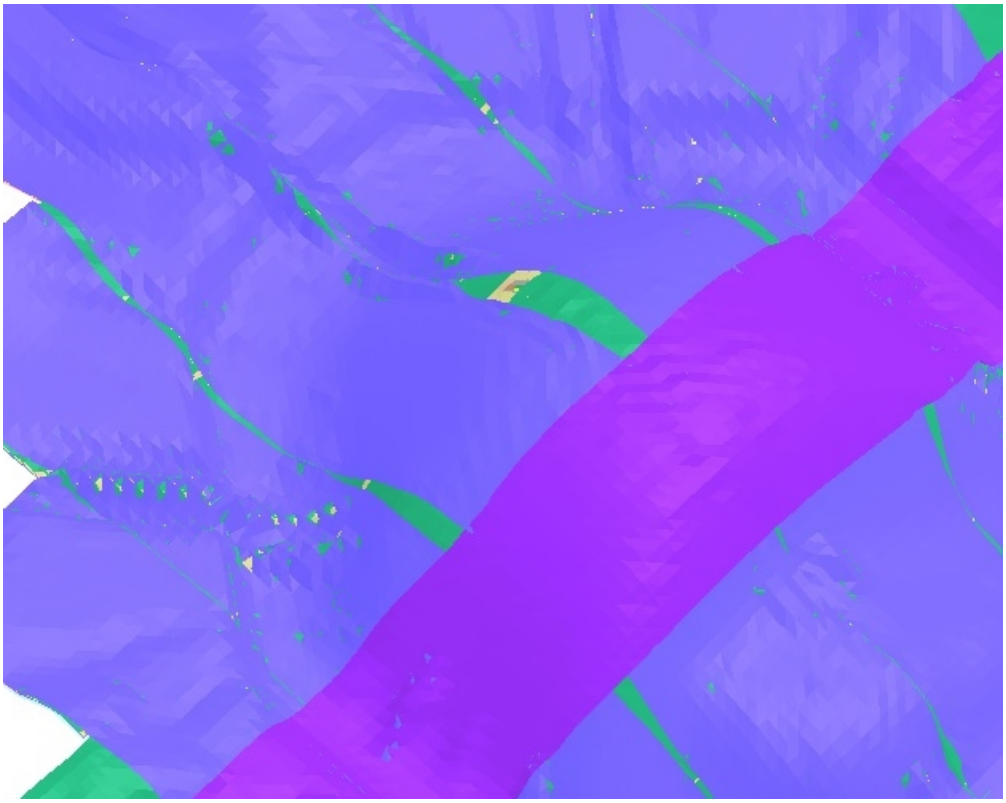


Figure 3.12: Overpredicted tape separation

particularly surprising since the model used was the very simplest form that such a model could take. Detailed friction modeling, however, was beyond the scope of this research, and so the models were reverted to the more accurate (but less scientific) constant-friction model used in preliminary tests. This model treats the contact interaction as having a constant coefficient of friction $\mu = 0.8$, and a gluing-stress of 1MPa. This resolved the separation issue, but should be addressed in more detail in any future investigation.

Another useful bit of information to be gleaned from qualitative inspection was that the models predicted wrinkles in several locations where the actual part exhibits some partial wrinkle formation. This information was used in order to inform the "critical locations" on which more detailed analysis was performed in Section 3.2.4.

3.2.4 Numerical Comparisons

Differential Cover In order to validate the models, a numerical comparison was needed for relating the simulated results with experimental ones. The comparison chosen was to use digital image correlation (the D.I.C. system Aramis) to measure the shear angle in an experimental part (full-size). Taking the extrema at critical locations, and comparing these to the predicted values can give a reasonable quantitative comparison.

The locations chosen for inspection were known critical locations for wrinkling, also being the regions in which the highest shear-angles are observed. In addition to this, the shear angle in the patch (along the crest) was also inspected. See Figure 3.13 for the locations of these inspection-points.

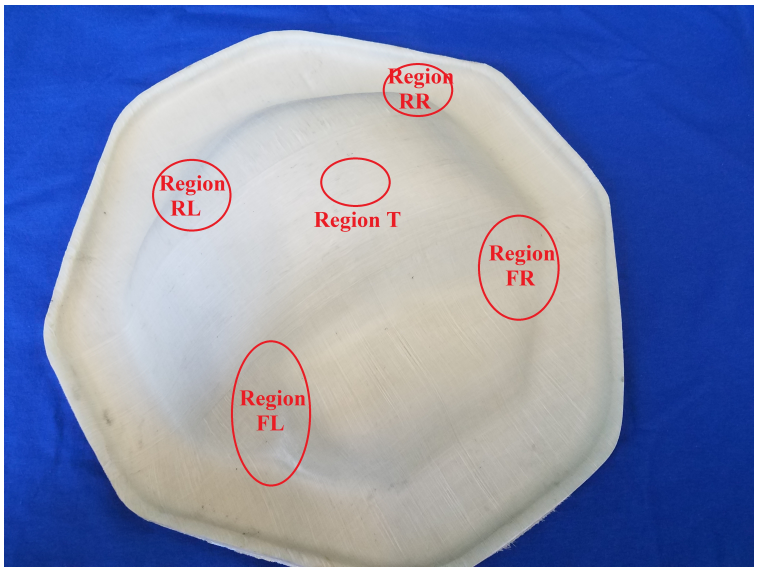


Figure 3.13: Critical regions in differential cover forming

Using the Aramis system, a strain field was measured on both top and bottom surface of the formed part. Then the peak shear-angle magnitude from each of the critical regions

Region	Measured Angle	Generic Material Properties		Measured Material Properties (Panel 3)		Measured Material Properties (Panel 5)	
		Angle	%Err	Angle	%Err	Angle	%Err
T	7.64°	12.0°	-57%	15.6°	-104%	8.83°	-16%
FR	21.2°	29.0°	-37%	13.2°	38%	8.02°	62%
FL	21.5°	25.9°	-20%	10.9°	49%	8.42°	61%
RR	18.0°	36.1°	-100%	12.2°	32%	10.5°	42%
RL	10.8°	32.9°	-159%	17.3°	-37%	11.0°	13%

(a) Shear Angles on the Top Surface of the Differential cover

Region	Measured Angle	Generic Material Properties		Measured Material Properties (Panel 3)		Measured Material Properties (Panel 5)	
		Angle	%Err	Angle	%Err	Angle	%Err
T	6.89°	13.5°	-96%	9.72°	-41%	7.77°	-13%
FR	24.0°	34.1°	-42%	11.6°	52%	8.72°	64%
FL	26.0°	23.2°	11%	13.6°	48%	9.00°	65%
RR	19.7°	34.0°	-73%	13.5°	32%	11.7°	40%
RL	28.0°	37.8°	-35%	11.8°	58%	11.8°	58%

(b) Shear Angles on the Bottom Surface of the Differential Cover

Table 3.1: Shear angle at critical locations in a differential cover

was taken, and averaged with those from other datasets. Measurements from two formed parts were used, which number was limited by the amount of available material for forming. These measured values were then compared with those taken from the surface layers of PAM-Form models, thus determining which material model best simulates the actual material. As discussed in Chapter 2, regarding Figure 2.6 the data necessitated there be 3 different material models in which the shear modulus (G_{12}) has been changed to reflect the differences seen in samples form the three different panels. Table 3.1 shows the results of this comparison. Note that panel 4 material (M4) is omitted due to inexplicable numerical instability in the corresponding model, thus preventing its completion. From the table, the differences between the different measured properties' model and the generic properties' model can be seen.

The first thing to note is that all the models overestimate the shear in the top point of the part. This can be explained in that the peak of the part contacts the mold first, thus facilitating a more rapid cooldown and limiting the shear deformation at this point. Since the simulations treated the process as isothermal, this phenomenon was not captured. For this reason, this point will not be included in the average error measure.

Additionally, with the exception of the top point, the measured material models tend to underestimate the shear, where the generic material overestimates it. This would suggest that the actual shear stiffness in the part is below that found in the measured models, but above that in the generic model.

Finally, an average was taken over the magnitudes of the errors from each material model (referred to hereafter as GM, M3, & M5 respectively), in order to have a single quantity on which to judge each model's merit. The results are as follows:

- GM had an average error of 60%
- M5 had an average error of 51%
- M3 had an average error of 43%

From this, it is clear that M3 (panel 3 material) is the best of the models here tested. In addition, since not all errors are overestimates (nor underestimates), taking error's sign into account paints a better picture, as the above numbers become 58%, 48%, and 30% respectively.

Hemisphere As with the differential cover, hemispherical parts were formed from PP/Glass, and shear measurements taken using Aramis DIC. These measurements were then compared to simulated results using the several material models. Severe wrinkling prevented those models using M5 from completing, so they are omitted here. Unlike the differential cover, however, the symmetric nature of this part means that there are no distinct locations to compare. Instead, the peak magnitude of shears in the bias directions (the $\pm^\circ$ directions as shown in Figure 3.14) were averaged to generate a comparison criterion. The results of

the comparison can be found in Table 3.2. Here, notice that M3 is no longer the best,

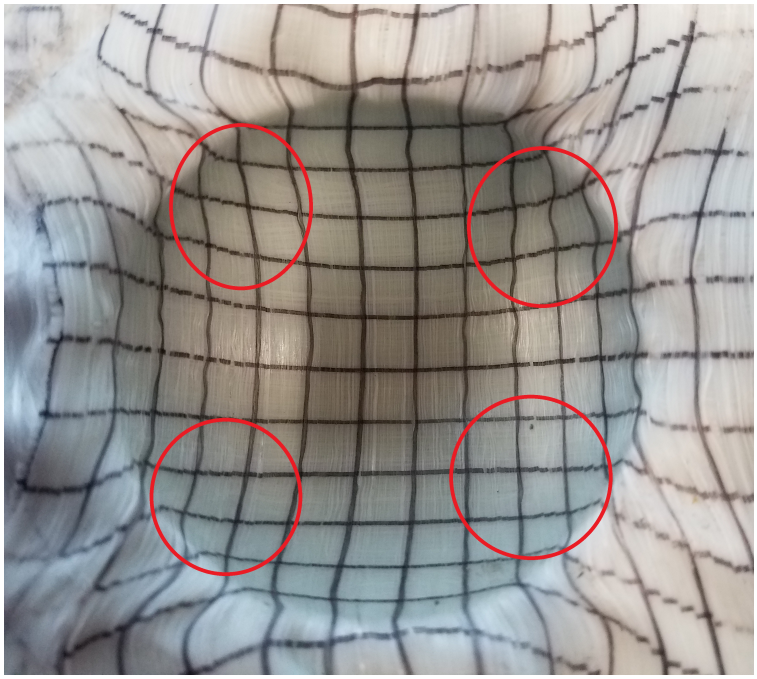


Figure 3.14: Critical regions in hemisphere forming

	Measured Angle	Generic Material Properties	Measured Material Properties (Panel 3)	Measured Material Properties (Panel 4)
Angle	32.8°	44.6°	19.0°	44.7°
% Error	-	-36%	42%	-36%

Table 3.2: Shear angle at critical locations in a hemisphere

but performs slightly worse than both other materials. Additionally, the relative predictions of the different models can give a bit more insight into the other models. Since M3 again underpredicts the shear, it can be inferred that M5 would greatly underpredict due to its higher shear stiffness. This over-stiffness in shear may be a contributor into the wrinkling failure of that model. Likewise, since M4 overpredicts shear here by slightly more than does GM, it can be inferred that M4 would overpredict the shear in the differential cover (barring numerical instability). Finally, from the combination of these models, it can be inferred that the true shear-stiffness (G_{12}) of the parts should fall somewhere between that measured from panels 3 and 4. This last point is proven in that a material model has been tested which

uses a shear stiffness calculated from a weighted average of that in M4 and M3. This model produces an average error of 29% (9% including ± error) in the differential cover and 12% in the hemisphere, thus proving that the true shear stiffness does, in fact, lie between those measured.

It can be noted here that the errors measured here are larger than those expected for a predictive tool. There are a number of possible causes for this discrepancy. The first possible source of error is the Aramis experiments. Since only a small number of tests were performed (2 each for hemisphere and differential cover), an inaccurate value may have been measured. Next, the model itself has a number of possible errors, starting with the sliding friction between individual layers, as well as between the blank and tool. Since this was beyond the scope of this research, a simple model was attempted, but was shown to be inaccurate. For the models in question, an approximate friction value was used. It may be that a better friction model could produce a more accurate simulation. Another possible source of the error is in the isothermal nature of the simulations. It is possible, as discussed regarding the differential-cover's measurements on 'point-T', that the cooling as the part interacts with the mold has a significant effect on the shearing behavior. Lastly, the discrepancies in the shear stiffness between different sample panels is likely another source of error. Each of these sources should be investigated in any future research, as well as possible simple methods of characterization, so these interactions can be reliably modeled.

CHAPTER 4
STRUCTURAL ANALYSIS

In order to conduct structural simulations of the "as-formed" part, we integrated the results of the forming simulation software(PAM-Form) with a structurel finite element analysis software (Abaqus). This is important, since variations in fiber-orientation, fiber-density, and part-thickness are all drivers in determining the strength of any structural member. Properly including this information, which is readily available from forming simulation results, in the structural analysis is expected to make the model more accurate in comparison to the "idealized" part. In order to convert between the forming and structural simulation softwares' formats, a MatLab script has been developed.

4.1 Mesh Conversion Method

The first step in transferring results of the forming simulation into a structural analysis model involves translating the mesh information from the format used by PAM-Form to that used by the Abaqus software. This begins by gathering the data from the PAM-Form database. PAM-Form stores the data in an hdf5 formatted file, which can be likened to a self-contained file structure of folders and subfolders. Once the paths to the desired data are known, it is a simple task to read all of the data using the MatLab command h5read. The problem that next arises is that PAM-Form simulates a multi-layered blank as many shell-meshes (one for each layer). While it is possible to model a laminate this way in Abaqus, it requires tie constraints to establish perfect bonding between the layers. This process is made overly complex when the layer meshes aren't properly aligned, and misalignment is inevitable when the meshes must first undergo a forming simulation. This means that the layered mesh must first be consolidated into a single shell mesh to represent the entire laminate, including fiber orientation and layer-thickness information. This mesh can then be used to model the part in Abaqus.

In order to capture the shape of the part in the single-layer mesh, it must represent the midsurface of all of the meshes in the multi-layer model. A series of nodes and elements (S4R 4-noded quadrilaterals for Abaqus analysis; hereafter referred to as "A-nodes" and "A-elements") are assigned planar positions in a rectangular region defined by the extreme x and y positions of the existing PAM-Form mesh (the "P-elements"). The A-elements are defined slightly smaller than the P-elements in order to better capture detail. The A-elements will later be given multilayered properties based on the arrangement of the nearby single-layered P-Elements. For each A-node, the z-position is found where the node is projected onto corresponding P-elements. This is accomplished by first eliminating all P-elements where the A-node is outside their rectangular bounding box (whose limits fully contain the P-element). The isoparametric shape-functions are then solved for the position of the A-node in the P-element's internal coordinates. Initially, the local coordinates were found using Matlab's intrinsic nonlinear solver fsolve, however, this proved too inefficient, and slow for reasonable use with large models. In order to improve the speed, the isoparametric relations were analytically reduced to a quadratic equation, which was then relatively straightforward to solve. Finding the local coordinates allows for a refined determination of whether a A-node is "inside" an P-element, as well as determining the interpolated z-position of the A-node within the P-element. By determining the z-position of each A-node with respect to each of the corresponding P-elements, and then averaging the extreme values, the A-mesh can correctly take the shape of the as-formed part.

For the purpose of generating a "clean" mesh that fully represents the part, however, those A-nodes which don't correspond to any P-elements must be removed from the mesh. At the edges of the part, however, this can cause issues in which a A-element has one corner that is outside the part, and therefore has no z-position, and must be removed. Since this would create an incorrect, and unreasonable edge geometry, it must be fixed. This is done by moving each of these overhanging corners to the edge of the nearest P-elements, thus defining their z-position. After this entire process, the resultant mesh follows the contours

of the PAM-Form model. The mesh is composed of elements which are rectangular with respect to the x-y plane except at the edges, where the mesh is distorted in order to reflect the correct edge shape.

4.1.1 Laminate Data

Once the mesh geometry has been defined, the next bit of information needed is the characteristics of the layup for each element. This includes fiber direction, layer thickness, and fiber volume fraction. Initially, this seems simple, since PAM-Form already reports the vector fiber direction, however, the P- and A- meshes generally will not align perfectly, so an approximation must be made. The chosen approximation uses a weighted average of the directions of the nearby elements, based on the area of overlap between P- and A- elements. This weighted average is given by:

$$\mathbf{D}_e = \sum_{i=1}^{n} \frac{A_i}{A_e} \cdot \mathbf{D}_i \tag{4.1}$$

Where:

A_e is the area of the A-element in question
A_i is the area of overlap between i^{th} P-element and the A-element
$\mathbf{D}_e$ is the fiber direction vector of the A-element
$\mathbf{D}_i$ is the fiber direction vector of the i^{th} P-element

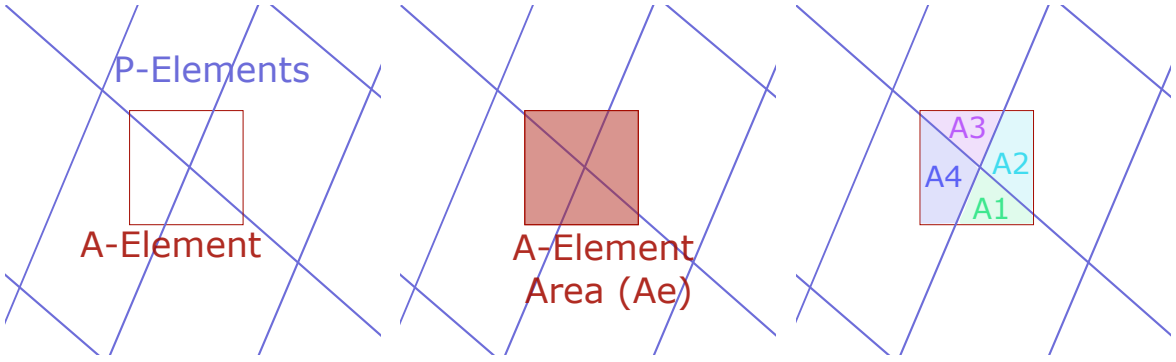


Figure 4.1: Overlapping areas in mesh conversion

There are several things to note here. First, the areas 1-4 are the areas of overlap between the P-elements and A-element (see Figure 4.1). Second, this expression can be expanded to include any number of P-elements. Finally, in some cases, there will be gaps or overlaps in the P-mesh. in these locations, the sum of overlapping areas will not equal the area of the A-element (the sum will be larger for overlaps, and smaller for gaps). Since this relationship will alter the local fiber volume fraction, it is important to capture this effect. Thus we define the fiber ratio:

$$R_{fib} = \frac{\sum_{i=1}^{n}(A_i)}{A_e} \tag{4.2}$$

This ratio represents the local areal packing density of the lamina within the given element. This ratio will be exactly one when the P-mesh is clean, but greater than one if there are overlaps(more fiber), and less than one if there are gaps(less fiber).

The next piece of data that must be gathered is the thickness of each layer. Since PAM-Form treats every layer as an individual mesh, the z-positions for each node that were gathered in the geometry section can be used to infer the thickness of the layers. The PAM-mesh for each layer represents the midsurface of that layer. By treating the gaps between midsurfaces as equally shared between each of the bounding layers, the approximate thickness can be calculated(see Figure 4.2). Note that in locations that might be otherwise characterized as voids, this will give the neighboring layers more thickness. Since all but the largest voids in forming will fill with neat resin, and since this thickness will also be used in decreasing the fiber volume fraction, the voids will simply be represented by its bounding layers being thicker, but with lower volume fraction. The measured thickness is then transformed from a z-direction measurement into a true thickness measurement by transforming it into the direction parallel to the A-element's normal vector.

The final piece of laminate data that must be determined is the local fiber volume fraction. Since volume fraction is not directly calculated in the current version of forming models, we need to make an approximation. We know that fiber volume fraction is the ratio of fiber

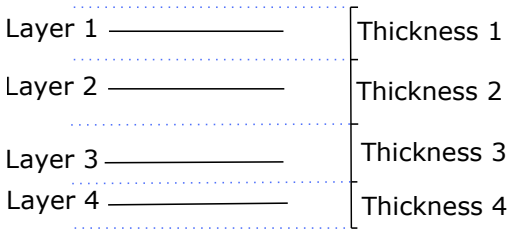


Figure 4.2: Layer thickness from gaps between midsurfaces

volume to total volume

$$V_f = \frac{V_{fiber}}{V_{total}} \tag{4.3}$$

If we assume that the volume of fiber remains constant except in instances of gaps or overlaps (which are captured by the fiber ratio in Equation (4.2)), then:

$$V_{fiber} = V_{fiber_0} \cdot R_{fib} \tag{4.4}$$

We can also assume that the total volume is given by the initial volume multiplied by the stretch ratios (λ) in each of the coordinate directions.

$$V_{total} = V_{total_0} \cdot \lambda_1 \cdot \lambda_2 \cdot \lambda_3 \tag{4.5}$$

The stretch ratios are given by:

$$\lambda_1 = [\varepsilon_1 + 1] \qquad \lambda_2 = [\varepsilon_2 + 1] \qquad \lambda_3 = \left[\frac{t}{t_0}\right]$$

Where ε_1 & ε_2 are the strains in the fiber & transverse directions respectively, and t & t_0 are the current & initial thicknesses respectively. Combining equations. (4.3 – 4.5), we get:

$$V_f = \frac{V_{fiber_0} \cdot R_{fib}}{V_{total_0} \cdot \lambda_1 \cdot \lambda_2 \cdot \lambda_3} \tag{4.6}$$

Since $\frac{V_{fiber_0}}{V_{total_0}}$ is simply the initial fiber volume fraction (V_{f_0}), this reduces to:

$$V_f = V_{f_0} \cdot \left(\frac{R_{fib}}{\lambda_1 \cdot \lambda_2 \cdot \lambda_3}\right) \tag{4.7}$$

The next step is to write the data gathered into a file readable by Abaqus. The one difficulty here is that Abaqus does not calculate effective lamina properties from the volume

fraction, but rather only accepts existing properties. In order to estimate the effective lamina elastic properties before inputting them into Abaqus, the Mori-Tanaka Method (as described in Appendix C.1) [32] has been used. Likewise, a method adapted in [33] from [34] and [35] has been used to estimate the strength properties (see Appendix C.2). Also, since the thermoplastic-matrix composite material is procured as a preprocessed tape, the constituent properties are not given. For the purpose of these models, typical properties for polypropylene [36, 37] and E-glass [34] have been used. These typical properties are outlined in Table 4.1. Note that both fiber and matrix are isotropic, resulting in fewer material parameters than the equivalent transversely isotropic material.

	E-Glass Properties	Polypropylene Properties
E	73 GPa	1.1 GPa
G	30 GPa	0.387 GPa
ν_{12}	0.23	0.42
ρ	2.54 $\frac{g}{cm^3}$	0.91 $\frac{g}{cm^3}$
F_t	3450 MPa	33.1 MPa
F_c	-	48.3 MPa

Table 4.1: Typical properties of E-glass and polypropylene

One final note on this conversion process is to highlight its reliance on 2D overlapping by ignoring the z-axis for much of the process. While this works fine for many shapes, it may cause complications for parts with large regions of steep incline or near-vertical faces. Many formable parts, however, do not have significant near-vertical regions, and therefore should not have any issue with this process.

4.1.2 Validation of Conversion Method

The mesh conversion method was applied to the forming of a thermoplastic composite part (automobile differential-case cover) that was prototyped in the Alfond Advanced Manufacturing Laboratory for Structural Thermoplastics. The distribution of thickness and fiber orientation generated by the mesh conversion method were compared to the expected results to very the procedure. First, PAM-Form results for the differential cover, can be seen to

visibly maintain the correct shape (Figure 4.3), thus demonstrating that the shape is rea-

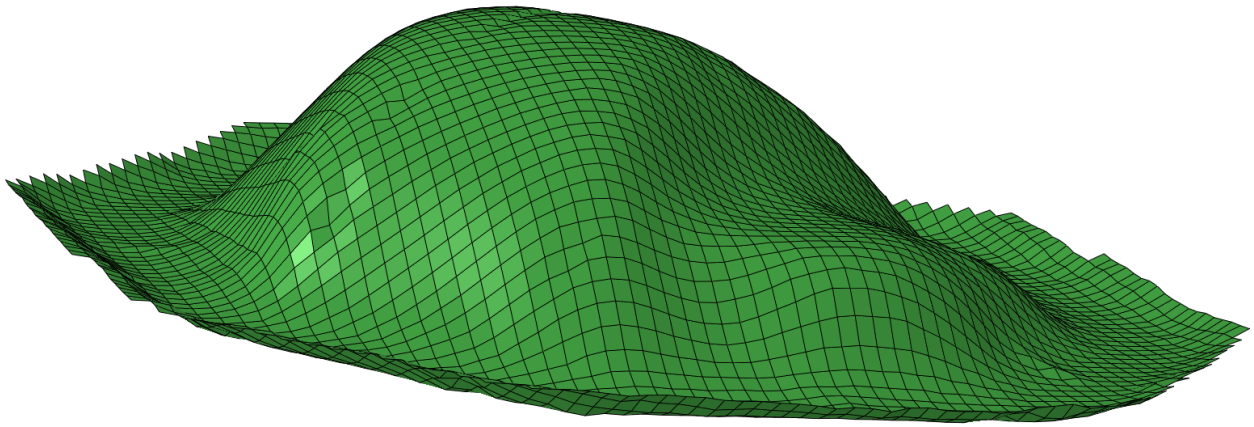
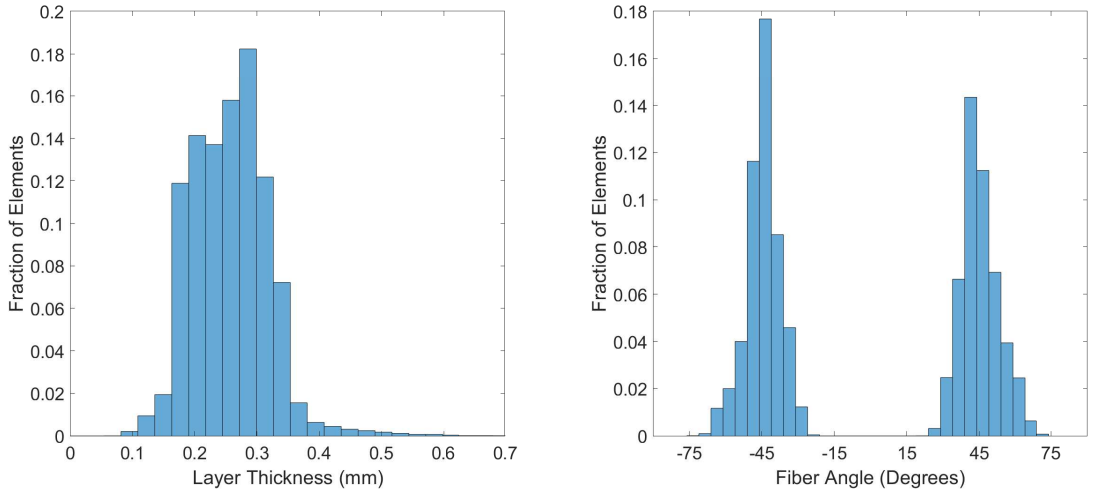


Figure 4.3: Differential cover as the forming simulation predicts

sonably accurate. Second, the local thicknesses (Figure 4.4a) can be seen to vary somewhat, but centered near the initial value of 0.35mm (although it tends toward lower, indicating significant compression). Likewise, the fiber angle (the fiber direction vector mapped to a layup angle on the local coordinate axes) can be seen to vary bimodally in Figure 4.4b. This distribution, however, has peaks at 45° and −45°, which are the nominal fiber directions. This verification example provides evidence that the mesh conversion method produces a reasonable model of the as-formed part for finite element structural analysis.



(a) Thickness distribution

(b) Fiber-angle distribution

Figure 4.4: Distributions of thickness and angle in the transformed model

4.2 Case Study

Since the purpose of this study is to demonstrate the value in using as-formed models versus the idealized model, a case study was required. To this end, a simple specimen was developed which can be relatively easily formed and tested. This specimen, seen in Figure 4.5, consists of a flat plate $254mm \times 76.2mm$ ($10'' \times 3''$) with an incomplete hemisphere (spherical cap) pressed in the center. This specimen simulates a variety of real-world parts wherein bumps or ridges are added to plates for stiffening. The purpose for this particular form-factor, though, was to produce a part which is not only formable, but also testable. Hence, the plate size is within the limits of a standard tension test frame.

The hemisphere, which will hereafter be referred to as the "bump" intersects the plate with a radius of 25.4mm (1"). The height of the bump was varied as fraction of this intersection radius between different models, and has been tested for values of $h/r = 0.25$, 0.5, and 0.75. A model with a full hemispherical bump (h/r=1) was also attempted, but wrinkled excessively, so as to be unusable. A $[\pm 45^\circ_2]$ layup was used for all models.

Figure 4.5: A CAD rendering of the model geometry

4.2.1 Model

In order to demonstrate the differences, two methods needed to be utilized. First, PAM-Form was used to simulate stamp-forming of the part, the forming results were translated into an Abaqus model, and a structural test was performed. The boundary conditions used were chosen to represent gripping in a tension test machine:

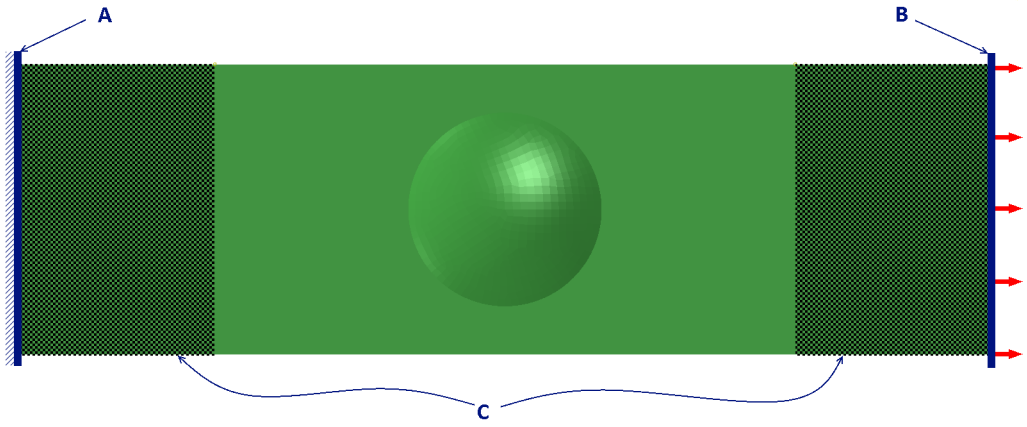


Figure 4.6: Boundary conditions on the model

A The nodes along one end of the model were fixed to all motion

B The nodes along the opposite end, were fixed to all but motion along the plate's long axis, and were translated 0.5 mm in the tension direction. This corresponds with a tensile strain of 0.2% in the equivalent flat plate.

C A region 50.8 mm (2”) from either end was fixed to all out-of-plane motion to approximate the restriction caused by a tension tester’s grips.

Next, an equivalent model was generated using the intrinsic tools in Abaqus. The fiber directions were assigned according to the global coordinate axes, thus presenting an “idealized” part. The part was meshed using S4R quadrilateral elements (since these are the type generated from the PAM-Form model), and the same boundary conditions were used.

In order to verify the mesh size, a convergence study has been performed. By running the model using different numbers of elements, and comparing a critical result between them, we can determine whether the element size has a significant effect on the results. In this case, the result compared is the maximum failure index (discussed below). The results can be seen in Figure 4.7. The curve shows that the result is changing rapidly with mesh-size

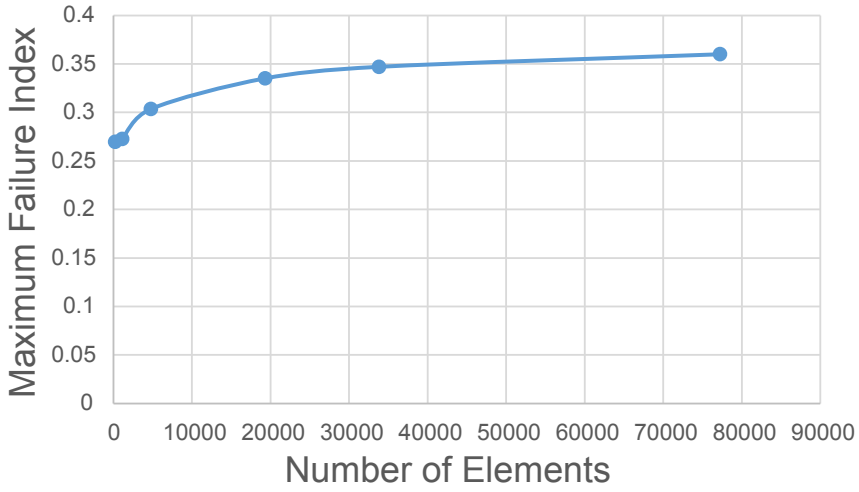


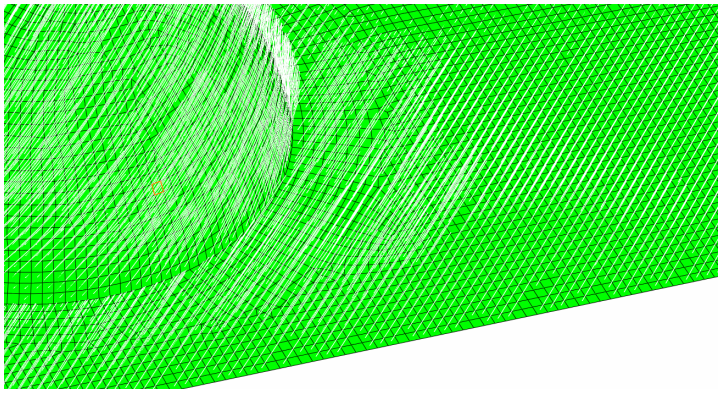
Figure 4.7: Mesh convergence plot for the model (h/r=0.5)

when using fewer elements, but levels off significantly when using more elements. The model having 19304 elements was chosen, since it corresponds with an element size which is slightly smaller than the PAM-Form mesh, which allows the translation scripts to function well. For finer meshes, while the results still change a bit, the translation process takes far too much time to be used extensively. In addition, the changes exhibited by these models are within the realm of experimental errors (7.4%). For these reasons, all models discussed have a

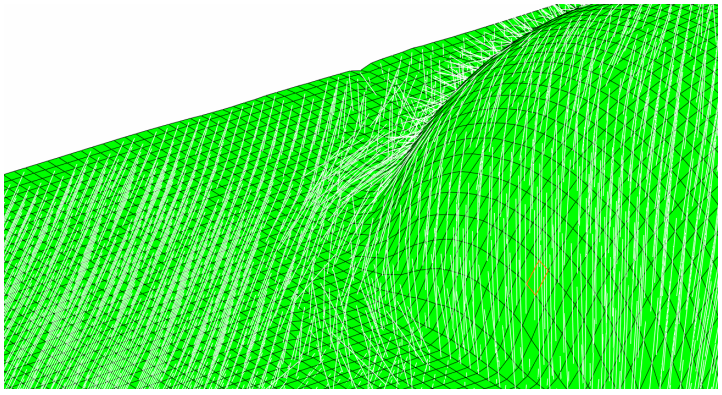
nominal element size of 1 mm. Due to the differences in size of the bump, the number of elements in each model varied from 19,304 to 22,858.

4.2.2 Results

The first thing to look at when inspecting these models is the fiber directions. In the idealized model, the fiber direction is established at the given angle, and is largely unaffected by the bump, whereas for the formed model, the fiber directions near the bump are much different from those further away. This effect becomes more pronounced as the bump height increases, and is particularly vivid for the bump with h/r=0.75 as seen in Figure 4.8, where the fiber directions are rendered as white lines. As can be expected, those models containing



(a) Ideal fiber directions



(b) As-formed fiber directions

Figure 4.8: Comparisons of fiber orientation between model types ($h/r = 0.75$)

a lesser bump demonstrate this effect to a smaller degree, since they require less fiber deformation to form.

4.2.2.1 Numerical Comparison

In order to compare the stresses within each of the parts, another model was first run. A flat plate of the same layup and under the same loads was modeled, and the average stress in the middle region was taken as the baseline for our comparison. Due to the central bump, each of the models exhibits stress concentrations, and these were utilized as a direct representation of the models' differences. By taking the flat-plate stress as the far field stress σ_0, the concentration factor can be calculated as $K = \sigma/\sigma_0$. It can be noted that at the discontinuity between plate and bump, the stress field and fiber architecture are more complicated than can be fully represented by shell elements, since shells assume negligible out-of-plane stresses. For this reason, the part may require a complete 3D analysis for truly accurate representations. For the purpose of this research, however, it is assumed that the shell assumptions provide a reasonable approximation. Figure 4.9 shows a representative

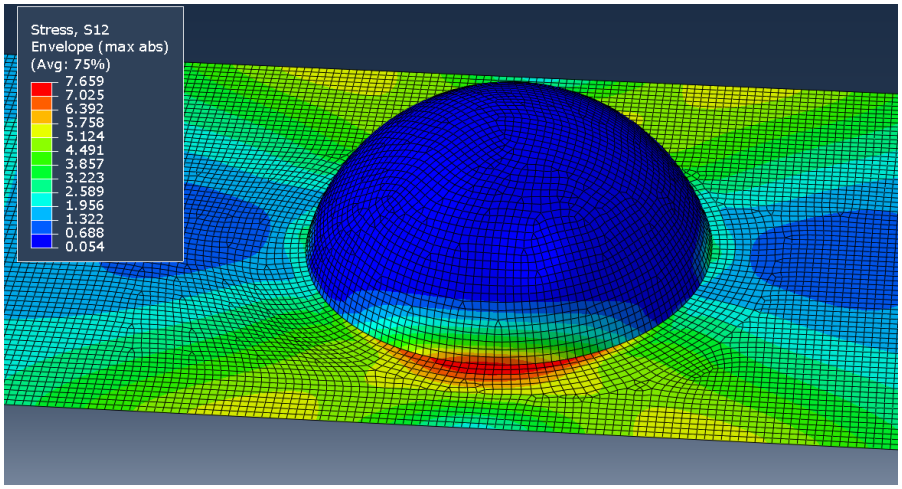


Figure 4.9: Shear stress envelope showing concentration region in an idealized model where h/r=0.75 (MPa)

plot of the in-plane shear stress within the model. While the stress patterns do vary slightly between the layers, and the other stress components produce different patterns, they all share the concentration region near the edge of the bump. A further series of contours demonstrating this are presented in Appendix D.

The results of each of the models was analyzed, and a stress concentration factor generated for each of the in-plane stresses (Fiber Stress σ_1, Transverse Stress σ_2, and Shear Stress τ_{12}). This factor takes the average of the several highest stress elements from the model, and compares them against the far field stress. Note that the stresses taken were the largest magnitudes seen through the thickness, rather than from an individual layer. The end results are shown in Figure 4.10

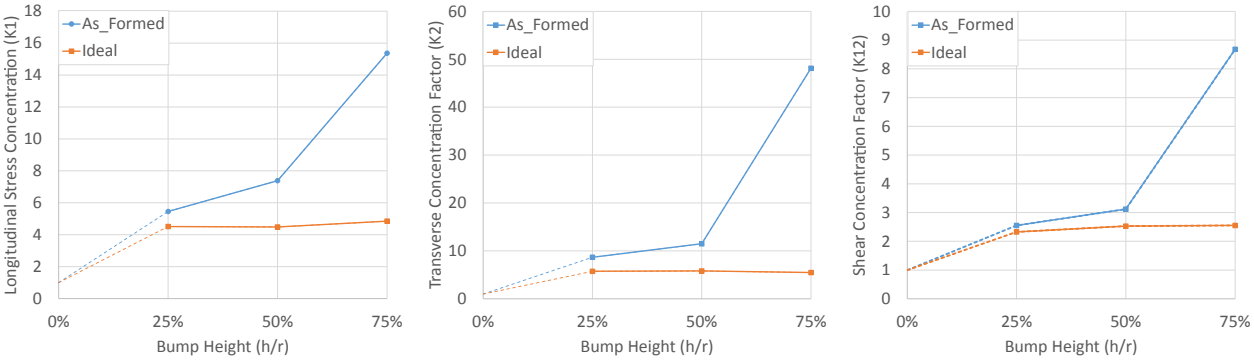


Figure 4.10: Stress concentration comparisons

Looking at these plots, a clear pattern emerges. For the bump with h/r=0.25, the ideal and formed part share a similar concentration factor, but as the bump-size increases, the factor rises for the formed part at a much larger rate than does the ideal part. This seems to align with intuition, since the smaller bump requires less fiber migration or deformation, and therefore the formed and ideal parts are quite similar. As the bump gets larger, more deformation occurs, making the two more different. The difference as a percentage of the formed-part stress is characterized in Figure 4.11.

Another thing to note is the magnitudes of these factors. In order to put these concentrations in perspective, the equivalent model with a hole in place of the bump was also simulated in the same manner, and results in concentration values of $K_1^h = 5.7$, $K_2^h = 5.2$, and $K_{12}^h = 3.2$. The ideal part generally hovers near or below this value, while the formed-part starts nearby, but generally becomes much higher for the larger bumps. This can be explained as being caused by the warping in the part which the bump induces. Since the part

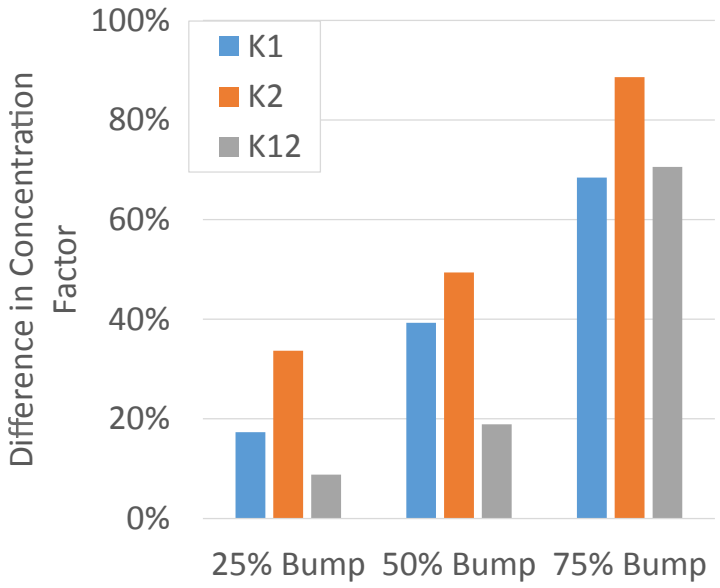


Figure 4.11: Differences in stress concentration between model types

is experiencing significant warping, the spurious stresses may aggravate the concentration factor.

Next, the safety factor in the part was assessed. To do this, the Abaqus intrinsic Tsai-Wu failure model was implemented. Tsai-Wu uses a quadratic formulation to calculate a failure index within the part. This is determined by solving Equation (4.8) for the failure index (I_f). Here $f_{i,j}$ are strength factors calculated from the engineering-strengths of the lamina, σ_i and τ_{12} are applied stresses. Based on this model, the safety factor for proportional increase of stresses (S_f) is given as the reciprocal of the failure index.

$$f_1\frac{\sigma_1}{I_f} + f_2\frac{\sigma_2}{I_f} + f_{11}\left(\frac{\sigma_1}{I_f}\right)^2 + f_{22}\left(\frac{\sigma_1}{I_f}\right)^2 + f_{66}\left(\frac{\tau_{12}}{I_f}\right)^2 + 2f_{12}\sigma_1\sigma_2 = 1 \tag{4.8}$$

Since the failure index (and thus safety factor) is determined by the relationship between all of the stresses within any given element, it is not surprising that these metrics follow similar patterns to the underlying stresses. This pattern can be seen in Figure 4.12.

Based on these analyses, the minimum safety factor has been taken for each model in order to compare the strength to first failure. The data is seen in Table 4.2.

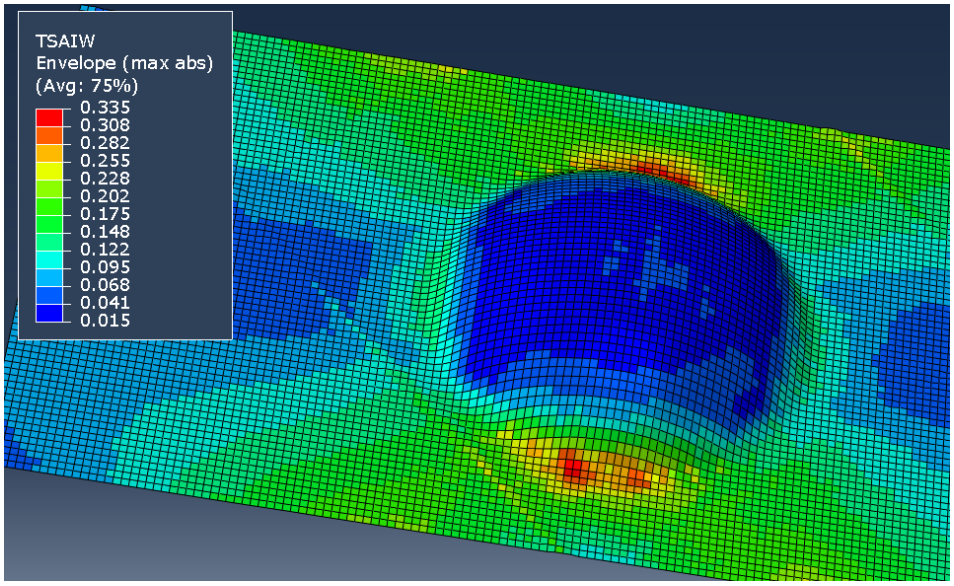


Figure 4.12: Failure index envelope for the as-formed model (h/r=0.5)

	Idealized Model		As-Formed Model	
h/r	S_f^{min}	Ply	S_f^{min}	Ply
0.25	4.01	1	3.61	1
0.50	3.62	1	2.98	1
0.75	3.51	1	1.02	2

Table 4.2: Factors of safety for various models with a $[\pm 45_2^\circ]$ layup

This data makes it clear that the as-formed part is weakened significantly as the bump height increases, whereas the idealized part retains significant strength in all configurations. It is also worth noting that ply-1 fails first in all but one model. Since the atypical model is the as-formed one with largest deformation, it can be assumed that this is an artifact of the spatially varying layup properties (fiber content and thickness) or of the small wrinkle that is visible in the forming simulations (Figure 4.13).

The final step in this analysis was to change the layup, so as to determine what effect this would have on both concentrations and safety factor. for the comparison, a $[0/90]_2$ layup was tested. For the purpose of comparison, the models were generated identically to those discussed above, with the exception of the layup. The first difference to notice is that this layup appears to be less formable than the previous one, as evidenced by the region of larger wrinkles (Figure 4.14), where the previous model contained only a single small

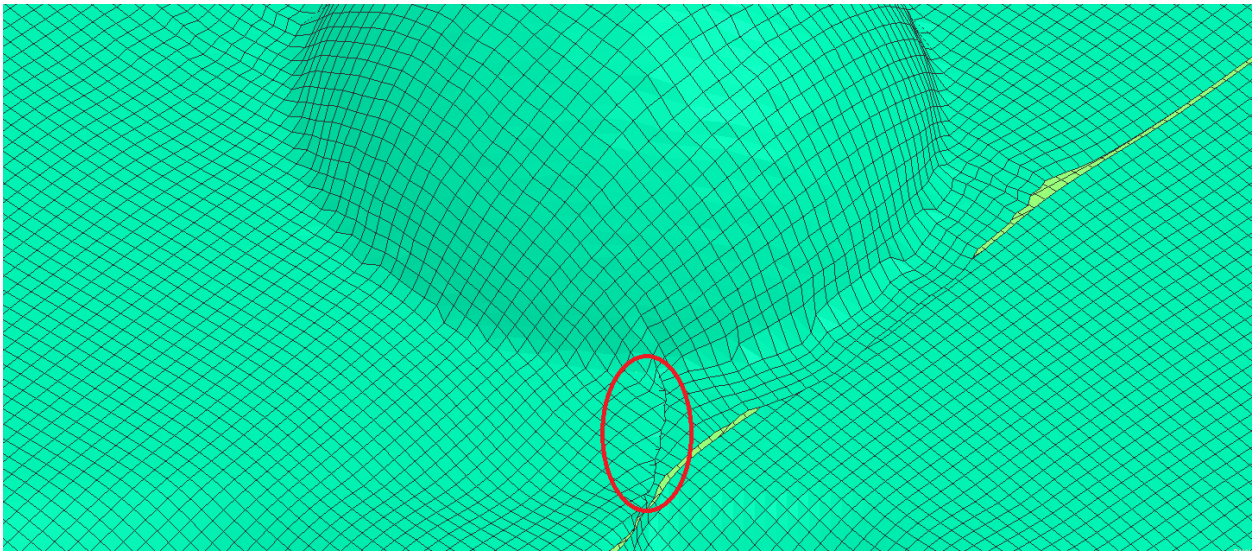


Figure 4.13: Wrinkling in the plate with bump (h/r=0.75)

wrinkle. Another thing to note is that Figure 4.15 shows much less fiber deviation than the earlier models, thus indicating that the idealized and as-formed model results may be closer together here.

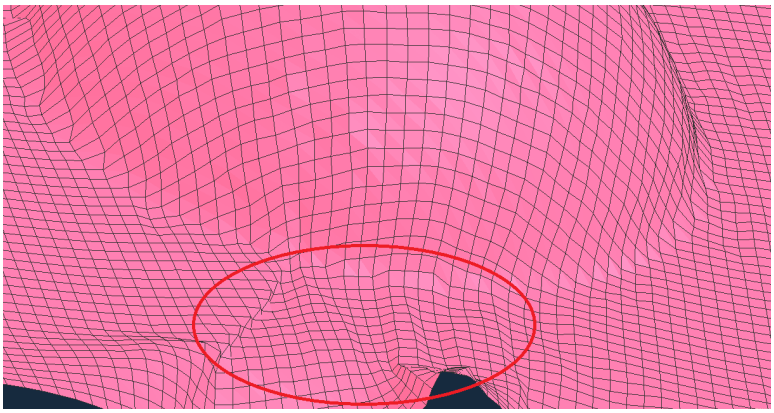
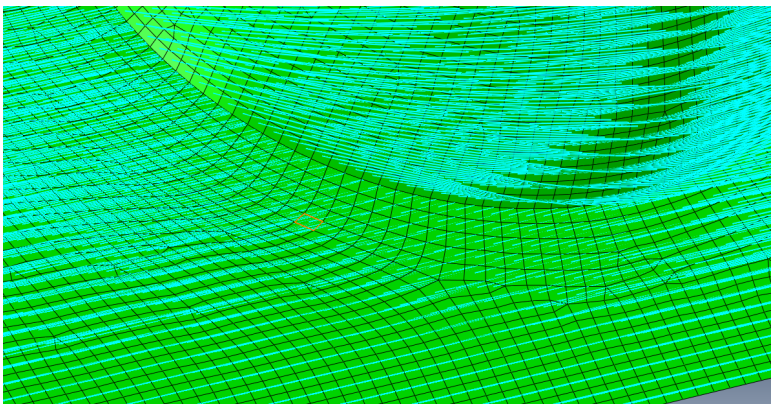
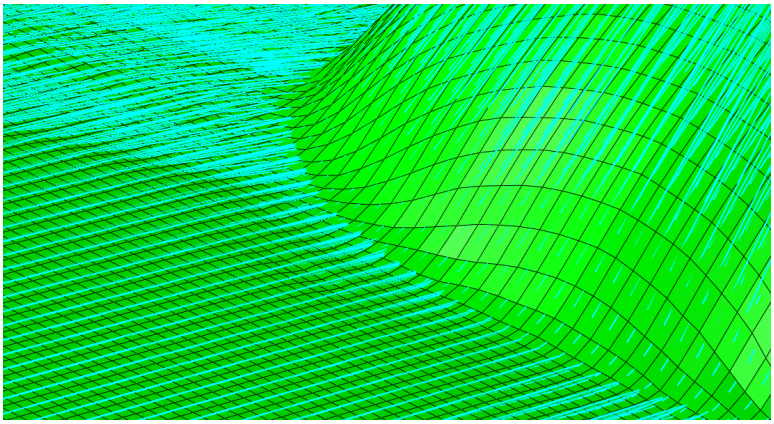


Figure 4.14: Wrinkling in the plate with bump (h/r=0.75)



(a) Ideal fiber directions



(b) As-formed fiber directions

Figure 4.15: Comparisons of fiber orientation between two model types ($h/r = 0.75$)

As seen in Figures 4.16 and 4.17, the stress concentration factors remain similar for most cases[1]. This indicates that the fiber deviation is a significant factor in determining these values, since the values diverge significantly in those models having significant fiber migration, whereas they remain similar for those models having small fiber migration. The only exception to the similarity of concentrations is in the transverse stress in the model with the largest bump. This can be seen as a reflection of the wrinkles mentioned above.

Lastly, Table 4.3 shows the safety factors for this set of models. It is important to note that, since the layups are different, the displacement-controlled loading applies different

[1]A flat plate with a $[0/90]_2$ will have negligible shear stress. Calculating the shear stress concentration, then, produces meaningless large values. These have been omitted.

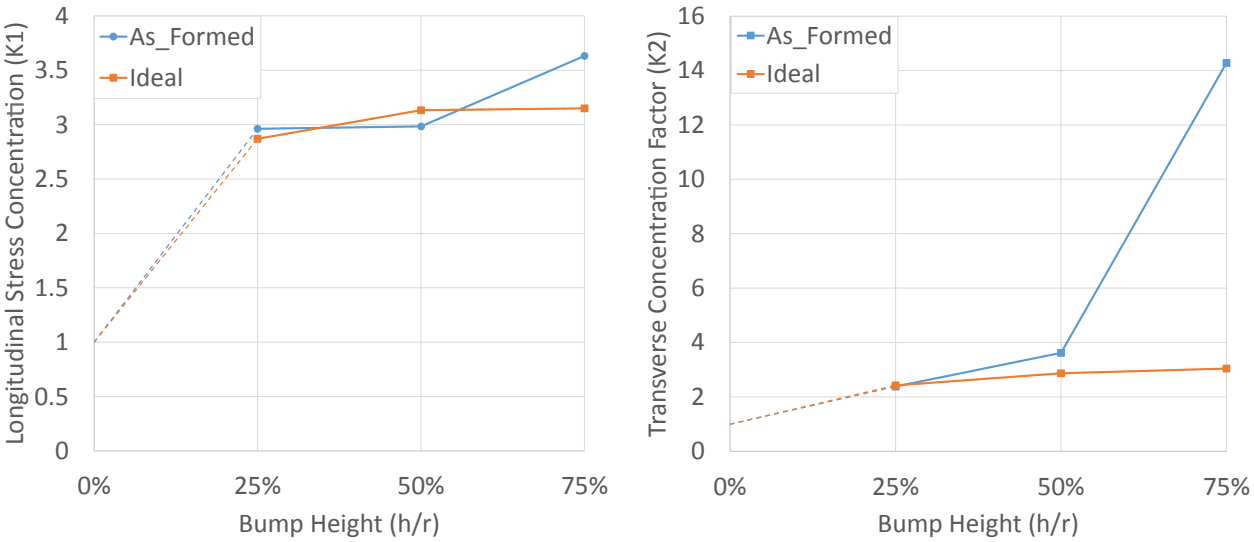


Figure 4.16: Stress concentration comparisons

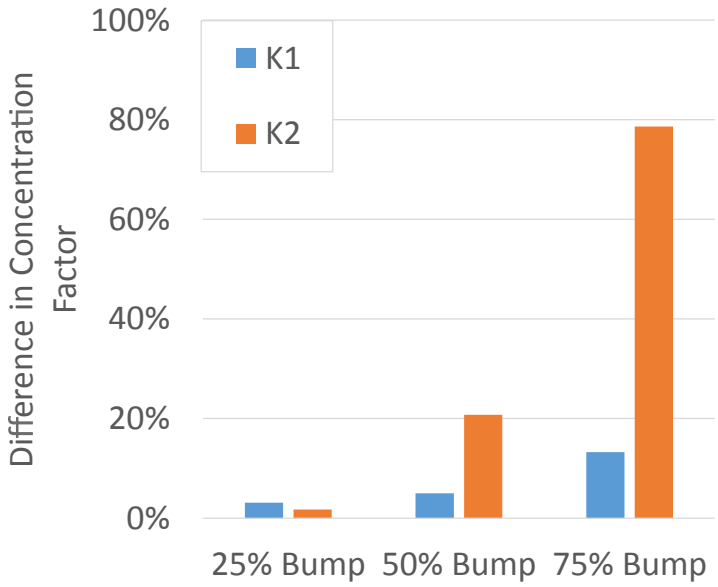


Figure 4.17: Differences in stress concentration between model types

forces to the different models. This means that comparing the safety factors between these two layups does not correlate to comparing strengths. It does, however, illustrate the power of this method in allowing the strength of a given part to be predicted.

It is worth noting that the wrinkled model is the weakest by far, having already failed at the applied displacement. This failure is seen to occur within the wrinkled region, as well as

happening within a 0° layer (L3) unlike all of the other models which fail in a 90° layer (L2 or L4). Again, this highlights the weakening effect of the wrinkles within the part.

	Idealized Model		As-Formed Model	
h/r	S_f^{min}	Ply	S_f^{min}	Ply
0.25	2.55	2	2.62	2
0.50	2.12	2	1.82	4
0.75	1.96	2	0.65	3

Table 4.3: Factors of safety for various models with a $[0/90]_2$ layup

While the above results are interesting, they do not yet prove conclusively that as-formed simulations are superior to idealized ones. That must be tested by comparing the simulations to real-world tests. That said, these models do provide compelling evidence that there is, in fact, a significant difference between these models, which is worth experimental investigation. It also provides a set of simple test cases on which to begin that investigation.

CHAPTER 5
CONCLUSIONS

Simulation tools are very important for the purpose of optimizing the stamp-thermoforming process. In order to make full use of these tools, research has been done to demonstrate and verify a method of material characterization toward simulation, and further, to combine the forming models with other solid-modeling techniques for final stress analysis.

5.1 Material Characterization

In order to have an accurate simulation, one must first have accurate information on the behavior of the constituent materials. A novel method of material characterization has been developed which relies entirely on DMA-type data. Advantages of this method include:

- Small sample sizes necessitates less material spent on characterization.
- Variable sweeps (Temperature, Amplitude, and Rate) generate large volumes of data from relatively few samples, thus shortening time required.
- Can be done on equipment that is readily available for characterizing viscosity

The material model generated by this method has been tested, and after interpolating the shear data, it performed quite well in comparison to the formed parts.

5.2 Forming Simulations

The simulation of thermoformed parts has been validated for two representative case studies, both of which contain doubly curved surfaces that require shearing to form. Resulting simulations not only predict critical regions for wrinkling, tearing, and ply separation, they also predict the shear deformation with reasonable accuracy. This is encouraging in terms of the usability of these methods for industrial optimization.

5.3 As-Formed Structural Analysis

A tool has been developed which allows for the use of thermoforming simulation results to inform more accurate structural models. This tool has been tested on a simple case study, and produces results that clearly differ from those of ideal models. While not proving that the new models are better, it certainly provides ample evidence that this should be further investigated.

5.4 Future Work

As mentioned several times within the body of this text, this field has much remaining to be investigated. Some avenues of future research which are relevant here include:

- Investigation of the source of discrepancy in shear stiffness between different panels of material. This would include characterization of the effects of processing temperature history and moisture content on the stiffness. Clarifying this would also allow for a more detailed look into the effects of rate and amplitude.

- Implementation of methods better relating the rheometer-measured viscosity to interply friction. As discussed in the text, the simplest flow model failed to accurately predict the behavior of the material in forming. A method which allows accurate friction relations to be derived from simple tests would be invaluable.

- Research into simple methods to characterize the thermal interface behavior between composite and mold, thus allowing for thermal analyses to be included in the forming simulation. This would improve the accuracy of the models by mitigating issues such as those which arose in the differential cover comparisons, where the shear angle was limited by localized cooling. Thermal analysis could also allow for crystallinity predictions, which could better inform later structural analyses.

- Continuation of the study relating as-formed models to structural simulation. This would include experimental validation, which could be done on a standard tensile tester using the model herein developed.

5.5 Final Thoughts

The tendency of thermoplastic composites to wrinkle has made simulation a must for optimized manufacturing, but the material models required are complex, and time consuming to create. The methods described here have the potential to enhance thermoplastic composites manufacturing by simplifying and speeding the process. In addition, the usefulness of these models is being extended into the realm of bettering predictions of the part's actual performance, thus improving the design process as well. The future of thermoplastics is bright!

REFERENCES

[1] W. F. Stanley and P. J. Mallon, "Intraply shear characterisation of a fibre reinforced thermoplastic composite," *Composites Part A: Applied Science and Manufacturing*, vol. 37, no. 6 SPEC. ISS., pp. 939–948, 2006.

[2] G. Lebrun, M. N. Bureau, and J. Denault, "Evaluation of bias-extension and picture-frame test methods for the measurement of intraply shear properties of PP/glass commingled fabrics," *Composite Structures*, vol. 61, no. 4, pp. 341–352, 2003.

[3] G. McGuinness and C. ÓBrádaigh, "Characterisation of thermoplastic composite melts in rhombus-shear: the picture-frame experiment," *Composites Part A: Applied Science and Manufacturing*, vol. 29, no. 1-2, pp. 115–132, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1359835X97000614

[4] M. Duhovic, "Deformation characteristics of knitted fabric composites," Ph.D. dissertation, University of Auckland New Zealand, 2004.

[5] K. Vanclooster, "Forming of Mulitlayered Fabric Reinforced thermoplastic Composites," Ph.D. dissertation, University of Leuven, 2009.

[6] S. P. Haanappel, R. Ten Thije, U. Sachs, B. Rietman, and R. Akkerman, "Formability analyses of uni-directional and textile reinforced thermoplastics," *Composites Part A: Applied Science and Manufacturing*, vol. 56, pp. 80–92, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.compositesa.2013.09.009

[7] J. Diaz and L. Rubio, "Developments to manufacture structural aeronautical parts in carbon fibre reinforced thermoplastic materials," *Journal of Materials Processing Technology*, vol. 143-144, no. 1, pp. 342–346, 2003.

[8] C. Lynam, A. S. Milani, D. Trudel-Boucher, and H. Borazghi, "Predicting dimensional distortions in roll forming of comingled polypropylene/glass fiber thermoplastic composites: On the effect of matrix viscoelasticity," *Journal of Composite Materials*, vol. 48, no. 28, pp. 3539–3552, 2014. [Online]. Available: http://jcm.sagepub.com/cgi/doi/10.1177/0021998313511650

[9] K. C. Warren, "Resistance Welding of Thermoplastic Composites for Industrial Scale Wind Turbine Blades," Master's thesis, University of Maine, 2012.

[10] S. Bhandari, "Feasibility of Using 3D Printed Molds for Thermoforming Thermoplastic Composites," Master's thesis, University of Maine, 2016.

[11] S. Bhandari and R. Lopez-Anido, "Finite element modeling of 3d-printed part with cellular internal structure using homogenized properties," *Progress in Additive Manufacturing*, pp. 1–12, 2018.

[12] S. Bhandari and R. Lopez-Anido, "Finite element analysis of thermoplastic polymer extrusion 3d printed material for mechanical property prediction," *Additive Manufacturing*, vol. 22, pp. 187–196, 2018.

[13] A. Willems, "Forming Simulation of Textile Reinforced Composite Shell Structures (PhD Thesis)," Ph.D. dissertation, University of Leuven, 2008.

[14] P. Boisse, N. Hamila, E. Vidal-Salle, and F. Dumont, "Simulation of wrinkling during textile composite reinforcement forming. Influence of tensile, in-plane shear and bending stiffnesses," *Composites Science and Technology*, vol. 71, no. 5, pp. 683–692, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.compscitech.2011.01.011

[15] P. Wang, N. Hamila, P. Boisse, P. Chaudet, and D. Lesueur, "Thermo-mechanical behavior of stretch-broken carbon fiber and thermoplastic resin composites during manufacturing," *Polymer Composites*, vol. 36, no. 4, pp. 694–703, 2015.

[16] S. Black, "Aligned discontinuous fibers come of age," *High-Performance Composites*, vol. 16, no. 2, pp. 42–47, 2008.

[17] X. Q. Peng and J. Cao, "A continuum mechanics-based non-orthogonal constitutive model for woven composite fabrics," *Composites Part A: Applied Science and Manufacturing*, vol. 36, no. 6, pp. 859–874, 2005.

[18] J. Cao, R. Akkerman, P. Boisse, J. Chen, H. S. Cheng, E. F. de Graaf, J. L. Gorczyca, P. Harrison, G. Hivet, J. Launay, W. Lee, L. Liu, S. V. Lomov, A. Long, E. de Luycker, F. Morestin, J. Padvoiskis, X. Q. Peng, J. Sherwood, T. Stoilova, X. M. Tao, I. Verpoest, A. Willems, J. Wiggers, T. X. Yu, and B. Zhu, "Characterization of mechanical behavior of woven fabrics: Experimental methods and benchmark results," *Composites Part A: Applied Science and Manufacturing*, vol. 39, no. 6, pp. 1037–1053, 2008.

[19] S. P. Haanappel, "Forming of UD fibre reinforced thermoplastics," Ph.D. dissertation, University of Twente, 2013.

[20] S. P. Haanappel and R. Akkerman, "Shear characterisation of uni-directional fibre reinforced thermoplastic melts by means of torsion," *Composites Part A: Applied Science and Manufacturing*, vol. 56, pp. 8–26, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.compositesa.2013.09.007

[21] P. Harrison, M. J. Clifford, and A. C. Long, "Shear characterisation of viscous woven textile composites: A comparison between picture frame and bias extension experiments," *Composites Science and Technology*, vol. 64, no. 10-11, pp. 1453–1465, 2004.

[22] ASTM, *ASTM D618-13: Standard Practice for Conditioning Plastics for Testing*, American Society for Testing and Materials, 2013.

[23] G. Lebrun, M. N. Bureau, and J. Denault, "Thermoforming-Stamping of Continuous Glass Fiber/Polypropylene Composites: Interlaminar and ToolâĂŞLaminate Shear Properties," *Journal of Thermoplastic Composite Materials*, vol. 17, no. 2, pp. 137–165, 2004.

[24] R. Banks, A. P. Mouritz, S. John, F. Coman, and R. Paton, "Development of a new structural prepreg: Characterisation of handling, drape and tack properties," *Composite Structures*, vol. 66, no. 1-4, pp. 169–174, 2004.

[25] ASTM, *ASTM D3167-10: Standard Test Method for Floating Roller Peel Resistance of Adhesives*, American Society for Testing and Materials, 2012.

[26] Q. Chen, P. Boisse, C. H. Park, A. Saouab, and J. Breard, "Intra/inter-ply shear behaviors of continuous fiber reinforced thermoplastic composites in thermoforming processes," *Composite Structures*, vol. 93, no. 7, pp. 1692–1703, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.compstruct.2011.01.002

[27] J. S. Lightfoot, M. R. Wisnom, and K. Potter, "A new mechanism for the formation of ply wrinkles due to shear between plies," *Composites Part A: Applied Science and Manufacturing*, vol. 49, pp. 139–147, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.compositesa.2013.03.002

[28] Y. R. Larberg and M. Åkermo, "On the interply friction of different generations of carbon / epoxy prepreg systems," *Composites Part A: Applied Science and Manufacturing*, vol. 42, no. 9, pp. 1067–1074, 2011.

[29] TA Instruments, "Rheology Applications Note the Principles and Applications of the Cox-Merz Rule," pp. 1–4. [Online]. Available: http://www.tainstruments.com/pdf/literature/RN14.pdf

[30] *PAM-Form User Guide*, ESI-Group, 2015.

[31] A. Margossian, S. Bel, and R. Hinterhoelzl, "On the characterisation of transverse tensile properties of molten unidirectional thermoplastic composite tapes for thermoforming simulations," *Composites Part A: Applied Science and Manufacturing*, vol. 88, pp. 48–58, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.compositesa.2016.05.019

[32] T. Chen, G. J. Dvorak, and Y. Benveniste, "Mori-Tanaka Estimates of the Overall Elastic Moduli of Certain Composite Materials," *Journal of Applied Mechanics*, vol. 59, no. September 1992, p. 539, 1992.

[33] C. Seigars, "Feasibility of Hybrid Thermoplastic Composite-Concrete Load Bearing System," Master's thesis, University of Maine, 2018.

[34] I. M. Daniel and O. Ishai, *Engineering mechanics of composite materials*, 2nd ed. New York, NY: Oxford University Press inc., 2006.

[35] E. J. Barbero, *Introduction to Composite Materials Design*, 2nd ed. Boca Raton, Fl: CRC Press, 2011.

[36] Ineos Olefins & Polymers USA, "Typcal Engineering Properties of Polypropylene," 2014. [Online]. Available: https://www.ineos.com/globalassets/ineos-group/businesses/ineos-olefins-and-polymers-usa/products/technical-information--patents/ineos-engineering-properties-of-pp.pdf

[37] Boedecker Plastics Inc., "Specifications for Polypropylene-Natural." [Online]. Available: https://boedeker.com/Product/Polypropylene-Natural

APPENDIX A

TABLE OF THERMOPLASTICS

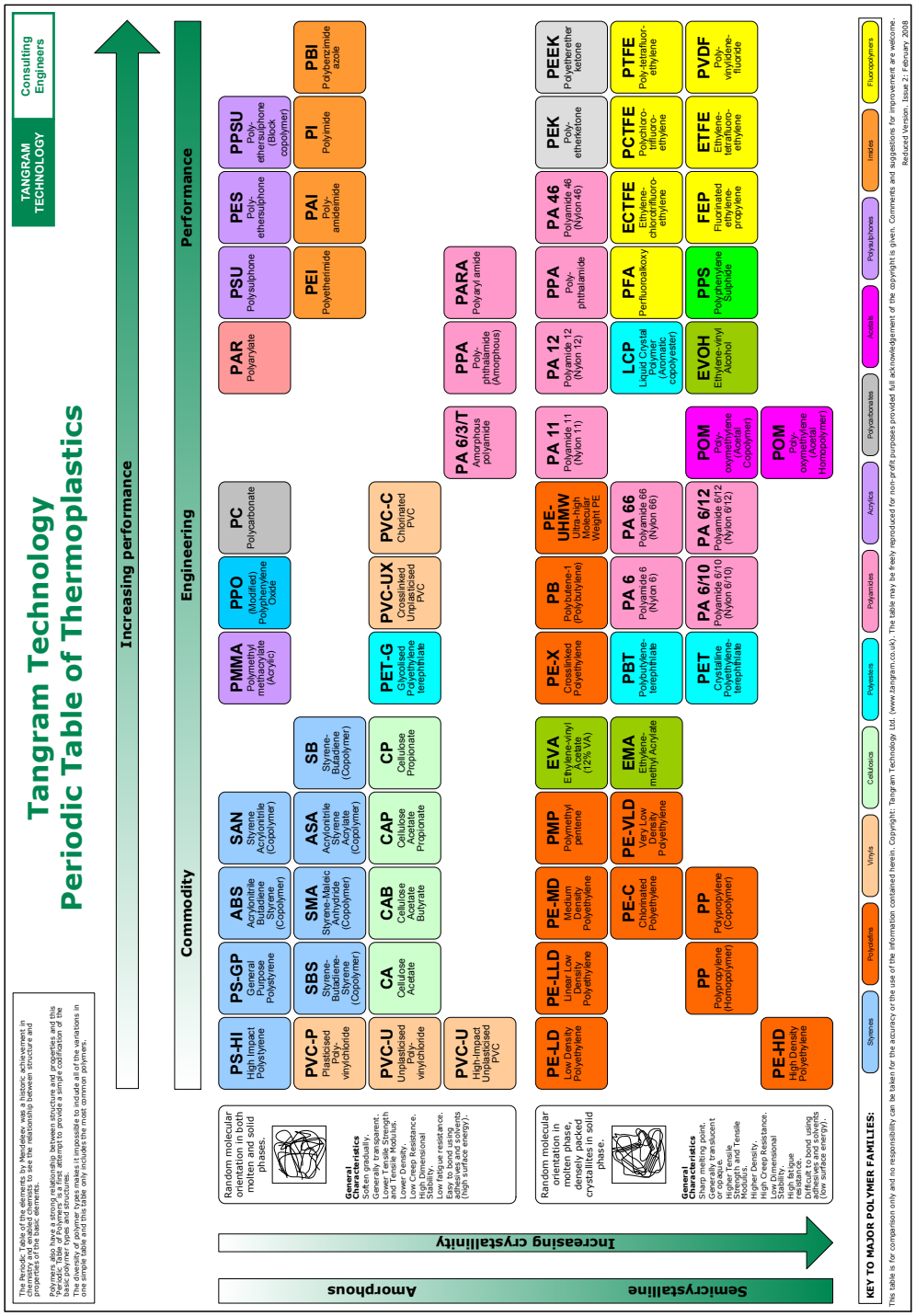


Figure A.1: Periodic Table of Thermoplastics(©Tangram Technologies Ltd. 2006)

APPENDIX B

MATERIAL PROPERTY TABLES

For each of the material models used for simulation, there are a variety of properties required. These are presented in this section.

		Generic Material	Measured Material (Panel-3)	Measured Material (Panel-4)	Measured Material (Panel-5)	Average Material
G_{12}	In-plane shear modulus (MPa)	Table B.2	21.14	0.922	45.19	10.4
E_1	Fiber-direction tensile modulus (GPa)	20	2.288			
B_1	Fiber-bending modulus (GPa)	0.03	2.288			
E_2	Transverse tensile modulus (GPa)	20	0.00413			
B_2	Transverse-bending modulus (GPa)	0.03	Same as transverse tension			
ρ	Density ($^g/_{cm^3}$)	1	1.41			
V_{fo}	Initial Fiber Volume Fraction	0.5	0.36			

Table B.1: Isothermal properties for various material models at $180°C$

Since the measured properties are known to vary with temperature, they are also defined in temperature lookup tables. These are shown consolidated in Table B.3. Note that not all properties are known to the limit of shown temperature due to measurement limitations as

γ_{12}	G_{12}
-1.5	$-2.5 \cdot 10^{-4}$
-0.5	$-1.25 \cdot 10^{-5}$
0	0
0.5	$1.25 \cdot 10^{-5}$
1.5	$2.5 \cdot 10^{-4}$

Table B.2: Lookup table for generic-material shear modulus

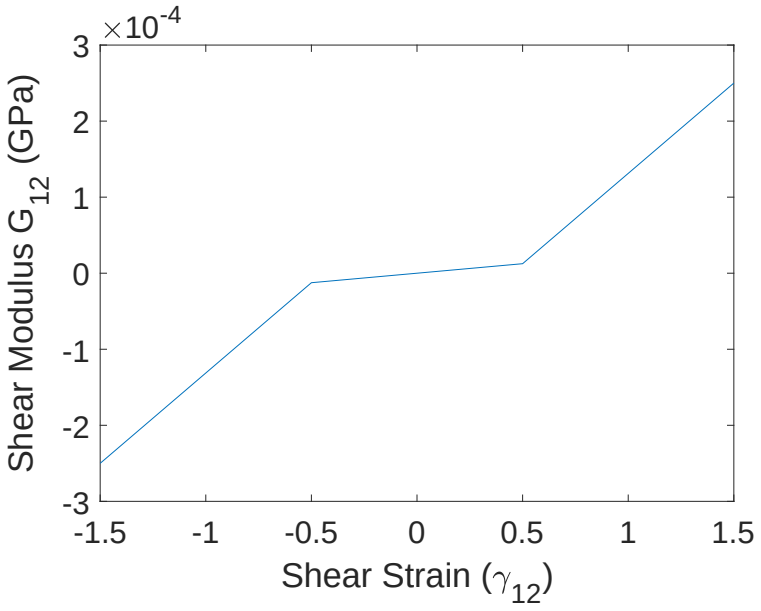


Figure B.1: Plot of lookup data for generic-material shear modulus

the samples lose integrity. For the purpose of models, these values would be extrapolated. Also note that G_{12}^3, G_{12}^4, and G_{12}^5 are the shear moduli from panels 3, 4, and 5 respectively.

T ($°C$)	E_1 (GPa)	B_1 (GPa)	E_2 (MPa)	G_{12}^3 (MPa)	G_{12}^4 (MPa)	G_{12}^5 (MPa)
160	13.36	10.58	27.85	50.02	12.21	100.8
161	12.69	10.31	26.54	46.48	10.08	97.63
162	11.93	9.974	25.22	42.94	8.174	93.93
163	11.11	9.578	23.54	40.95	6.702	90.75
164	10.22	9.125	21.18	37.51	5.691	86.97
165	9.303	8.62	17.93	35.36	4.662	83.07
166	8.365	8.075	14.08	33.9	3.741	80.42
167	7.432	7.432	9.943	32.15	3.12	76.46
168	6.528	6.528	4.13	30.24	2.556	72.84
169	5.676	5.676	-	28.73	2.16	69.31
170	4.896	4.896	-	27.41	1.92	66.89
171	4.206	4.206	-	26.47	1.727	63.63
172	3.618	3.618	-	25.51	1.56	60.62
173	3.14	3.14	-	24.71	1.457	57.88
174	2.775	2.775	-	24.03	1.358	55.64
175	2.518	2.518	-	23.37	1.272	53.46
176	2.362	2.362	-	22.82	1.198	51.61
177	2.294	2.294	-	22.22	1.121	49.51
178	-	-	-	21.72	1.045	47.81
179	-	-	-	21.14	0.9701	46.56
180	-	-	-	-	0.9222	45.19
181	-	-	-	-	-	43.98
182	-	-	-	-	-	42.92
183	-	-	-	-	-	42.06
184	-	-	-	-	-	41.18
185	-	-	-	-	-	40.49
186	-	-	-	-	-	39.74
187	-	-	-	-	-	38.98
188	-	-	-	-	-	38.19
189	-	-	-	-	-	37.63
190	-	-	-	-	-	36.72
191	-	-	-	-	-	36.24
192	-	-	-	-	-	35.76
193	-	-	-	-	-	34.94
194	-	-	-	-	-	34.53
195	-	-	-	-	-	34.09
196	-	-	-	-	-	33.62
197	-	-	-	-	-	33.15

Table B.3: Lookup table of thermally varying properties

APPENDIX C

MATERIAL PROPERTY ESTIMATES

C.1 Mori-Tanaka Method

The Mori-Tanaka method [32] approximates the elastic properties of a fiber reinforced composite using an analytical representation of a simplified unidirectional composite, assuming a transversely isotropic (in the 2-3 plane) fiber, and fully isotropic matrix. The resulting composite property estimates have been shown to match experimental values quite well.

To start, the constituent materials' elastic properties must be converted into the equivalent Hill's elastic moduli:

$$k_f = \left(\frac{4}{E_{2f}} - \frac{1}{G_{23f}} - \frac{4\nu_{12f}^2}{E_{1f}} \right)^{-1}$$

$$l_f = 2\nu_{12f} k_f \qquad m_f = G_{23f}$$

$$n_f = E_{1f} + \frac{l_f^2}{k_f} \qquad p_f = G_{12f}$$

$$k_m = \left(\frac{E_m}{2 - 2\nu_m - 4\nu_m^2} \right)$$

$$l_m = 2\nu_m k_m \qquad m_m = \frac{E_m}{2(1+\nu_m)}$$

$$n_m = E_m + \frac{l_m^2}{k_m} \qquad p_m = \frac{E_m}{2(1+\nu_m)}$$

Where E_{1f}, E_{2f}, and E_m are the fiber-longitudinal-, fiber-transverse-, and matrix- moduli respectively. G_{12f}, and G_{23f} are the fiber's in-plane and intralaminar shear moduli. ν_{12f}, and ν_m are poisson's ratios for fiber (in-plane) and matrix. Finally, $(k_f, l_f, m_f, n_f, p_f)$ and $(k_m, l_m, m_m, n_m, p_m)$ are the Hill's elastic moduli for fibers and matrix respectively. The next step is to use the Mori-Tanaka equations, along with fiber and matrix volume fractions (V_f, and V_m), to determine the effective Hill's elastic moduli for the composite (k, l, m, n, p).

$$k = \frac{V_f k_f (k_m + m_m) + V_m k_m (k_f + m_m)}{V_f (k_m + m_m) + V_m (k_f + m_m)}$$

$$l = \frac{V_f l_f (k_m + m_m) + V_m l_m (k_f + m_m)}{V_f (k_m + m_m) + V_m (k_f + m_m)}$$

$$m = \frac{m_m m_f (k_m + 2m_m) + k_m m_m (V_f m_f + V_m m_m)}{k_m m_m + (k_m + 2m_m)(V_f m_m + V_m m_f)}$$

$$n = V_m n_m + V_f n_f + (1 - V_f l_f - V_m l_m) \left(\frac{l_f - L_m}{k_f - k_m} \right)$$

$$p = \frac{2V_f p_m p_f + V_m (p_m p_f + p_m^2)}{2V_f p_m + V_m (p_f + p_m)}$$

These effective Hill's moduli can then be converted back into the effective engineering properties of the composite $(E_1, E_2, \nu_{12}, G_{12}, G_{23})$.

$$E_1 = n - \frac{l^2}{k} \qquad E_2 = \frac{4m(kn - l^2)}{(k + m)n - l^2}$$

$$\nu_{12} = \frac{l}{2k} \qquad G_{12} = p \qquad G_{23} = m$$

Table C.1 contains the composite properties calculated using this method for the PP/Glass tape with a fiber volume fraction of 0.35 (0.6 by weight), and Figure C.1 demonstrates how these properties vary with changing fiber volume fraction.

	Effective Properties
E_1	26.17 GPa
E_2	2.45 GPa
ν_{12}	0.348
G_{12}	0.786 GPa
G_{23}	0.739 GPa

Table C.1: Effective elastic properties of PP/Glass with 35% fiber

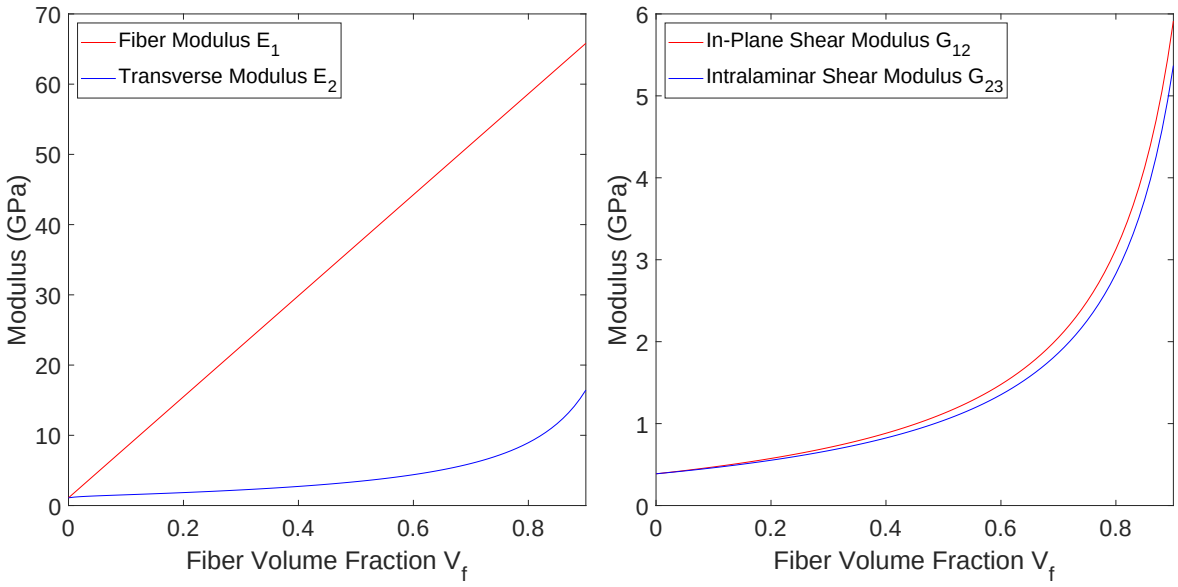


Figure C.1: Composite elastic properties as the fiber volume fraction varies

C.2 Strength Estimate

To calculate the strength properties of a lamina, a method was developed in [33] from equations found in [34] and [35]. This method uses a combination of max-strain, buckling, and fracture mechanics techniques.

The longitudinal tensile strength relies on a max-strain method, wherein the value is determined as the minimum of the strengths assuming either fiber or matrix dominated failure:

$$F_{1tf} = F_{ft}\left[V_f + V_m\frac{E_m}{E_f}\right] \qquad F_{1tm} = F_{mt}\left[V_f\frac{E_m}{E_f} + V_m\right]$$

$$F_{1t} = min\left(F_{1tf}, F_{1tm}\right)$$

Where F_{ft} and F_{mt} are the fiber and matrix tensile strengths respectively (note that this method assumes isotropic constituents, and would require modification in order to be used for transversely isotropic materials). For the remaining parameters, fracture mechanics and buckling analysis are used, and so additional parameters must be introduced in addition to the constiturent strengths and stiffnesses. Since these parameters are not known for PP/glass, the values for E-Glass/Epoxy and E-Glass/Polyester (from table 1.3 in [35]) have

been used. This assumption has proved to be reasonably accurate for a similar material system (PETg/Glass) in [33]. The parameters in question are shown in Table C.2.

G_{Ic}	Mode I fracture toughness	$334^{J}/_{m^2}$	E-Glass/Polyester
G_{IIc}	Mode II fracture toughness	$456^{J}/_{m^2}$	E-Glass/Polyester
t_t	Fracture transition thickness	0.6mm	E-Glass/Epoxy
α_σ	Fiber misalignment	2.97°	E-Glass/Epoxy
C_v	Void adjustment factor	1 (no voids)	-

Table C.2: Fracture and buckling parameters used in analysis

The longitudinal compression strength requires we first calculate the shear strength (F_6) and the compressive strength factor (χ).

$$F_6 = \sqrt{\frac{G_{IIc}G_{12}}{\pi\left(\frac{t_t}{4}\right)}}$$

$$\chi = \frac{G_{12}\alpha_\sigma}{F_6}$$

$$F_{1c} = G_{12}(1 + 4.76\chi)^{-0.69}$$

Transverse tensile strength requires we first determine the tensile strength factor (Λ^0_{22}), and finally, the transverse compressive strength is calculated.

$$\Lambda^0_{22} = 2\left(\frac{1}{E_2} - \frac{\nu^2_{12}E^2_2}{E^3_1}\right)$$

$$F_{2t} = \sqrt{\frac{G_{Ic}}{1.12^2\pi\left(\frac{t_t}{4}\right)\Lambda^0_{22}}}$$

$$F_{2c} = F_{mc}C_v\left[1 + \left(V_f - \sqrt{V_f}\right)\left(1 - \frac{E_m}{E_f}\right)\right]$$

Based on this method, strengths can be calculated for the material at any fiber volume fraction, which is imperative for the PAM-Form conversions. Table C.3 contains the values determined for a 0.35 fiber volume fraction PP/glass.

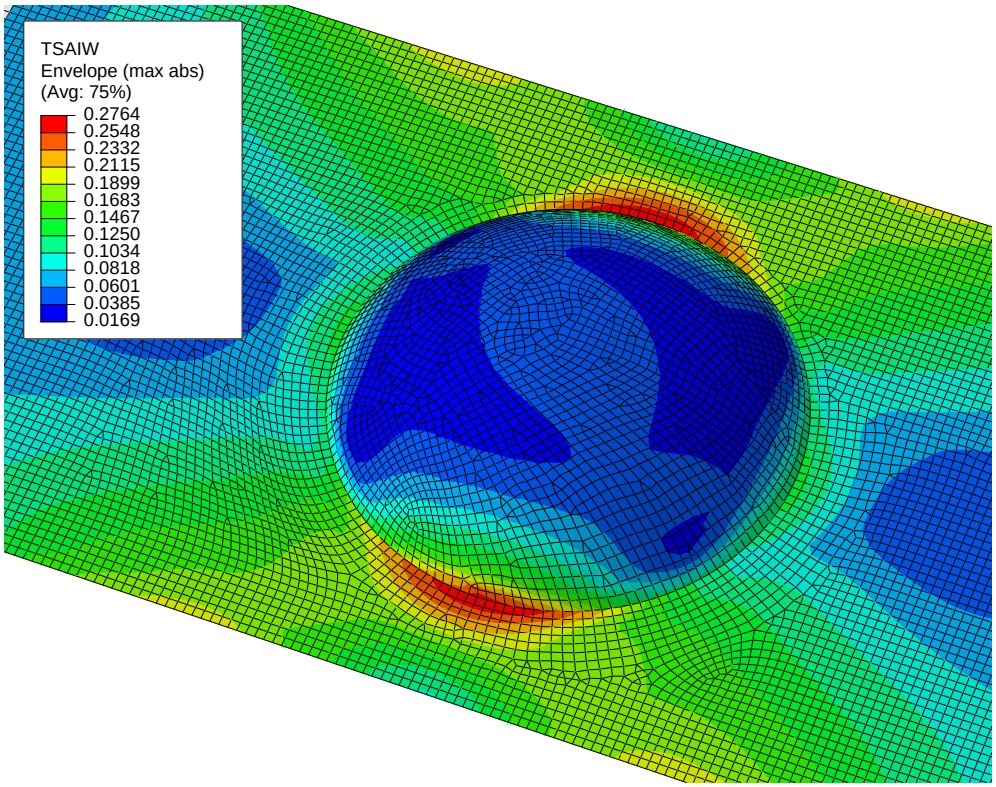
F_{1t}	787.2 MPa
F_{1c}	186.6 MPa
F_{2t}	26.3 MPa
F_{2c}	36.8 MPa
F_6	27.6 MPa

Table C.3: Calculated strengths for PP/glass

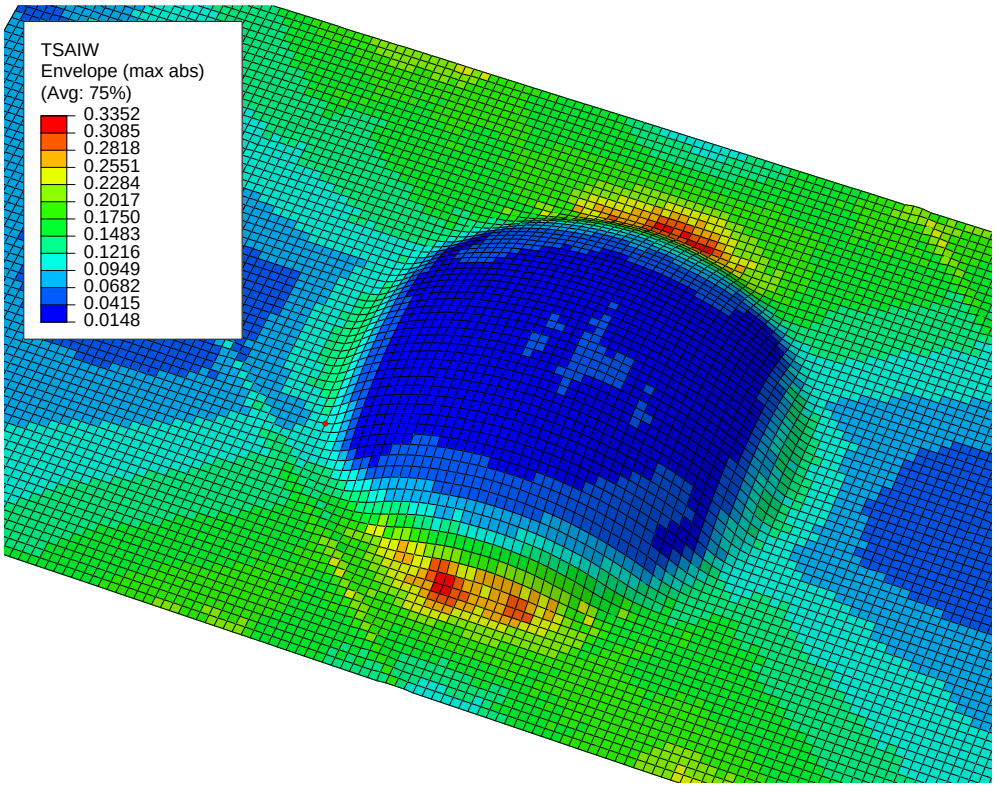
APPENDIX D

STRESS CONTOUR PLOTS

In this appendix, the results contours are presented for the plate with bump (h/r=0.5). In all cases, the stresses shown are in GPa.

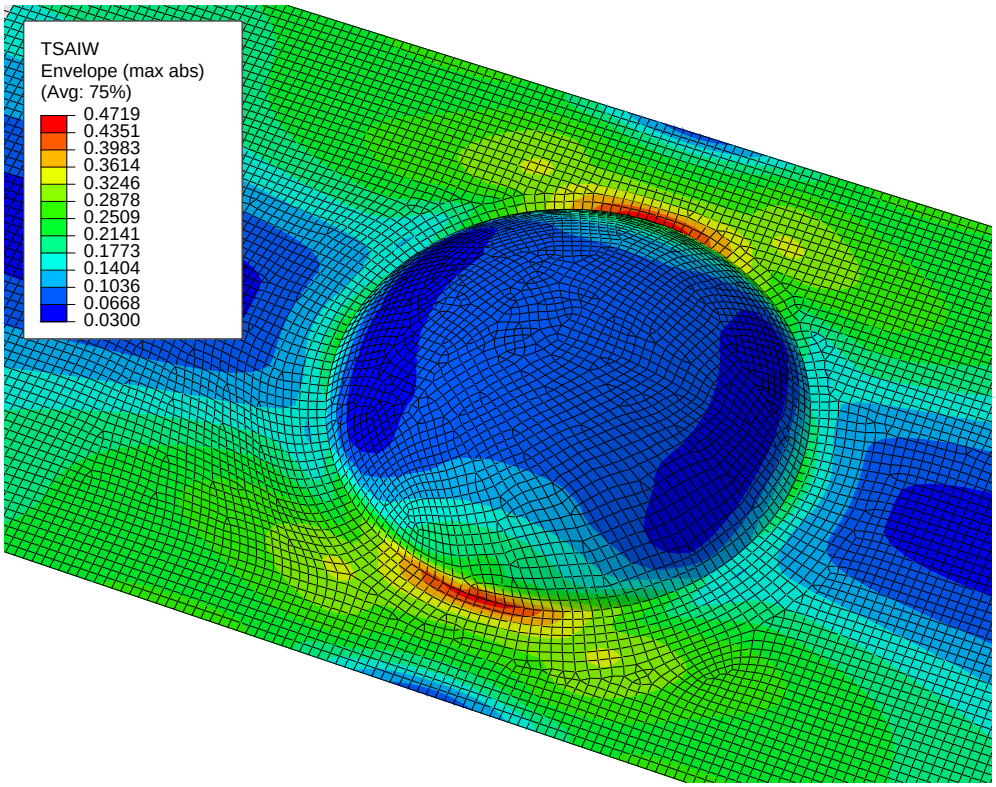


(a) Idealized model

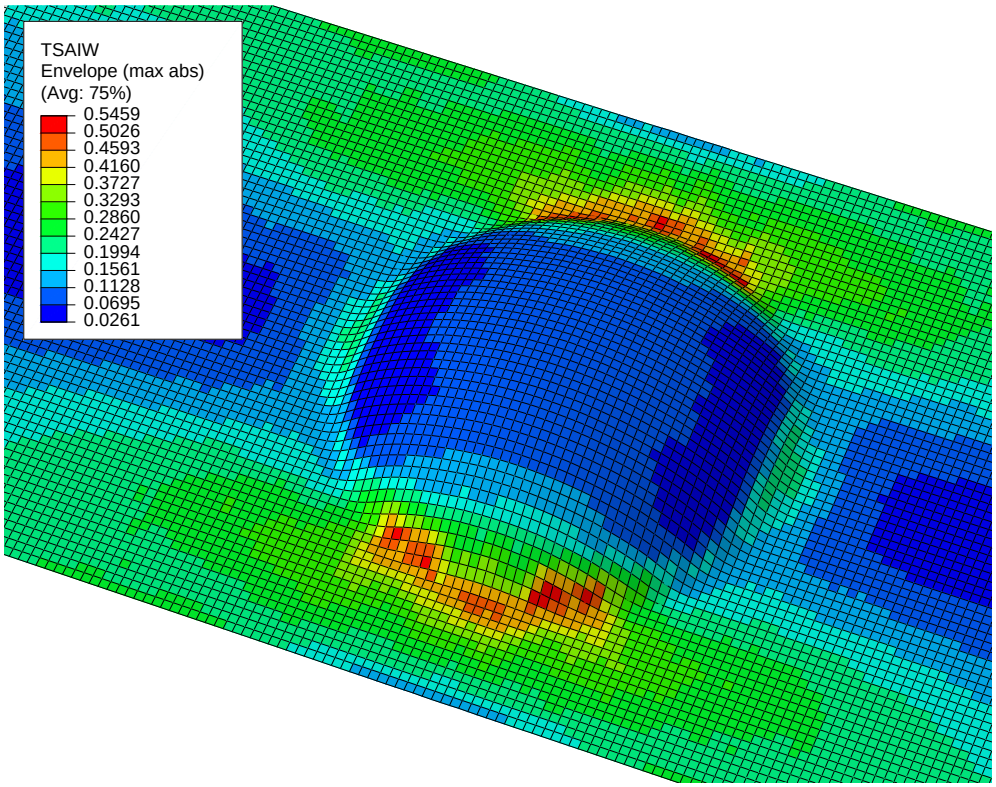


(b) As-formed model

Figure D.1: Failure-index envelope ($[\pm 45^\circ_2]$ layup)

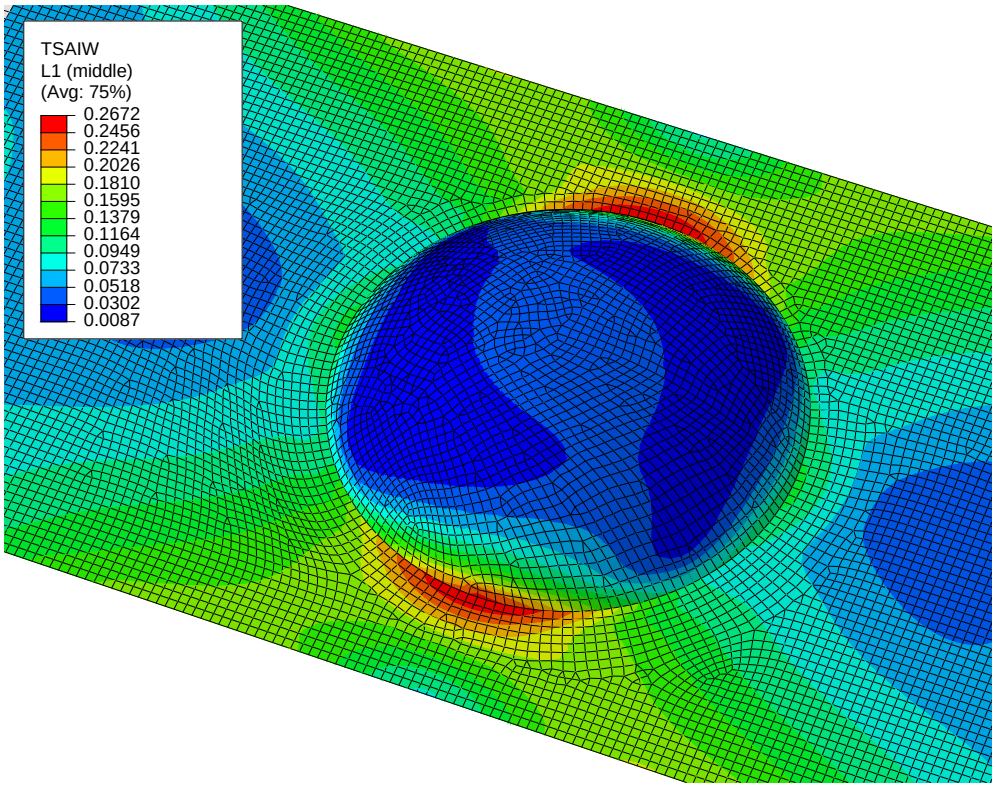


(a) Idealized model

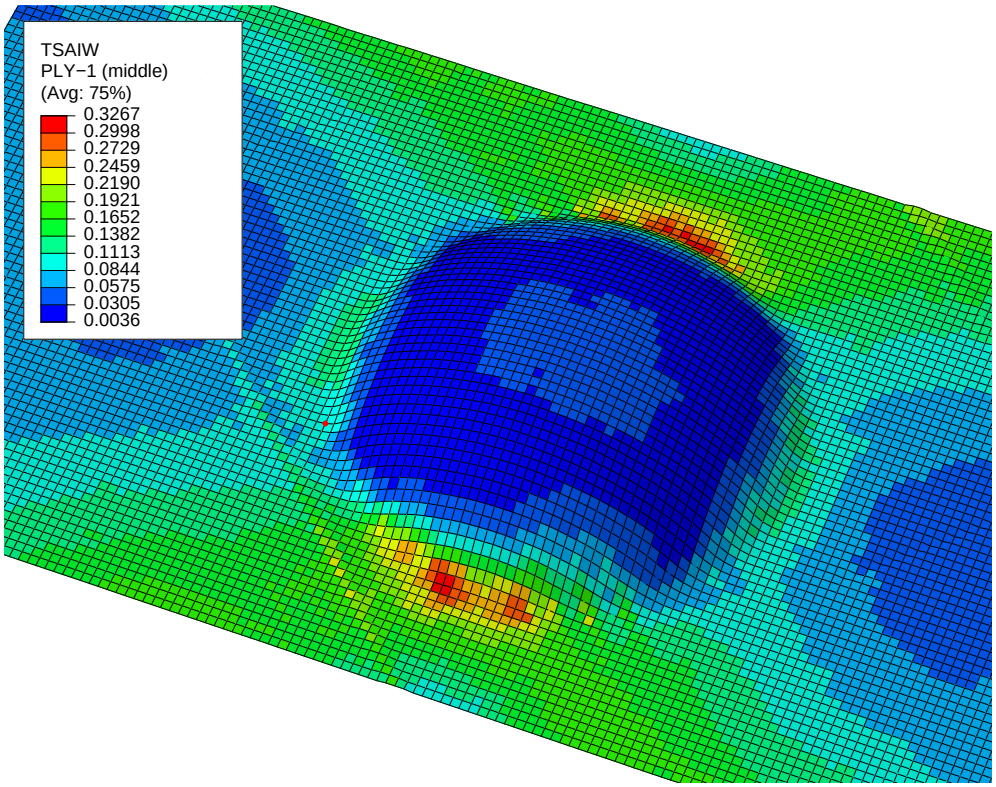


(b) As-formed model

Figure D.2: Failure-index envelope ($[0/90]_2$ layup)

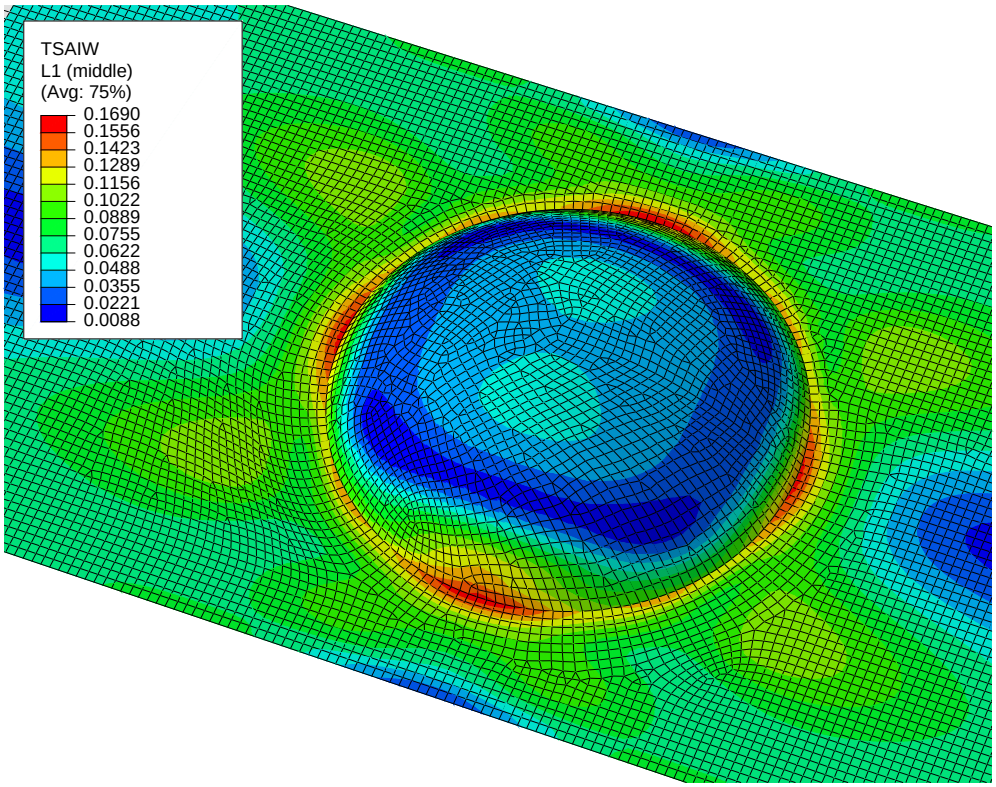


(a) Idealized model

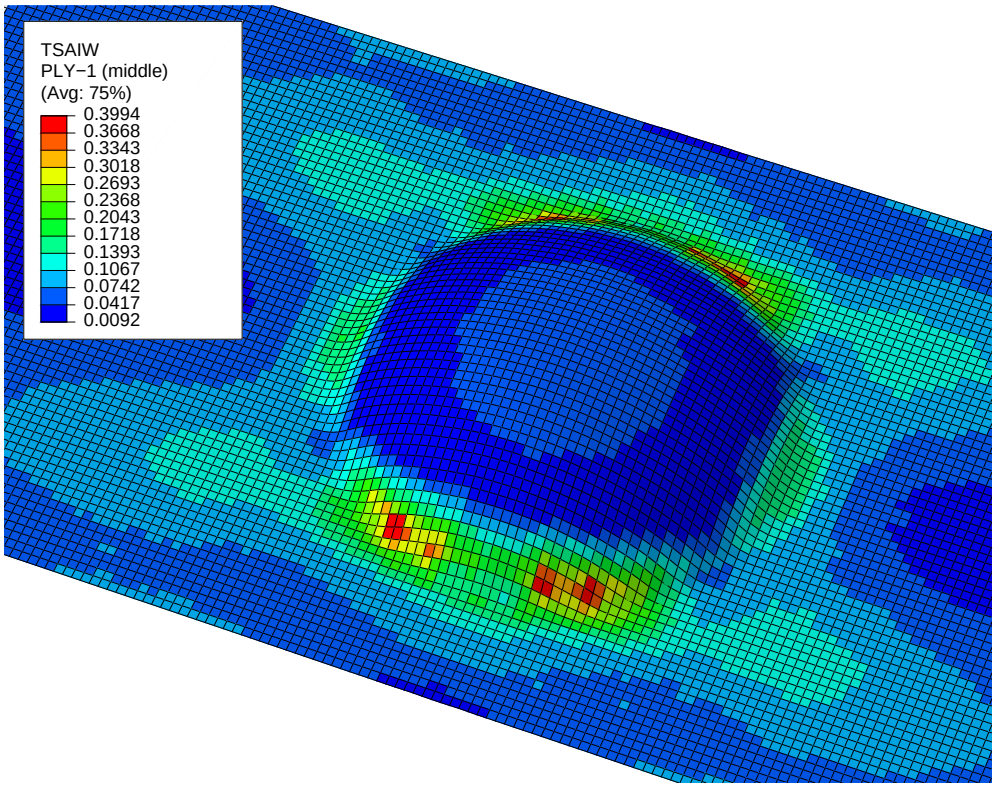


(b) As-formed model

Figure D.3: Failure-index in layer-1 ($[\pm 45^{\circ}_2]$ layup)

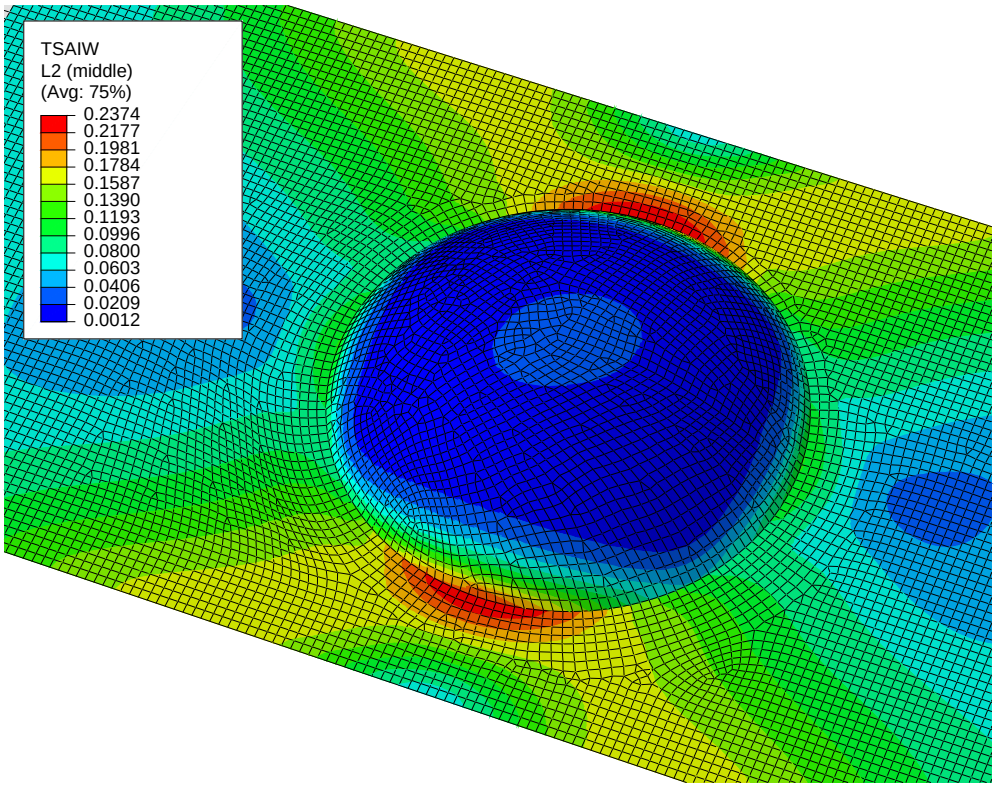


(a) Idealized model

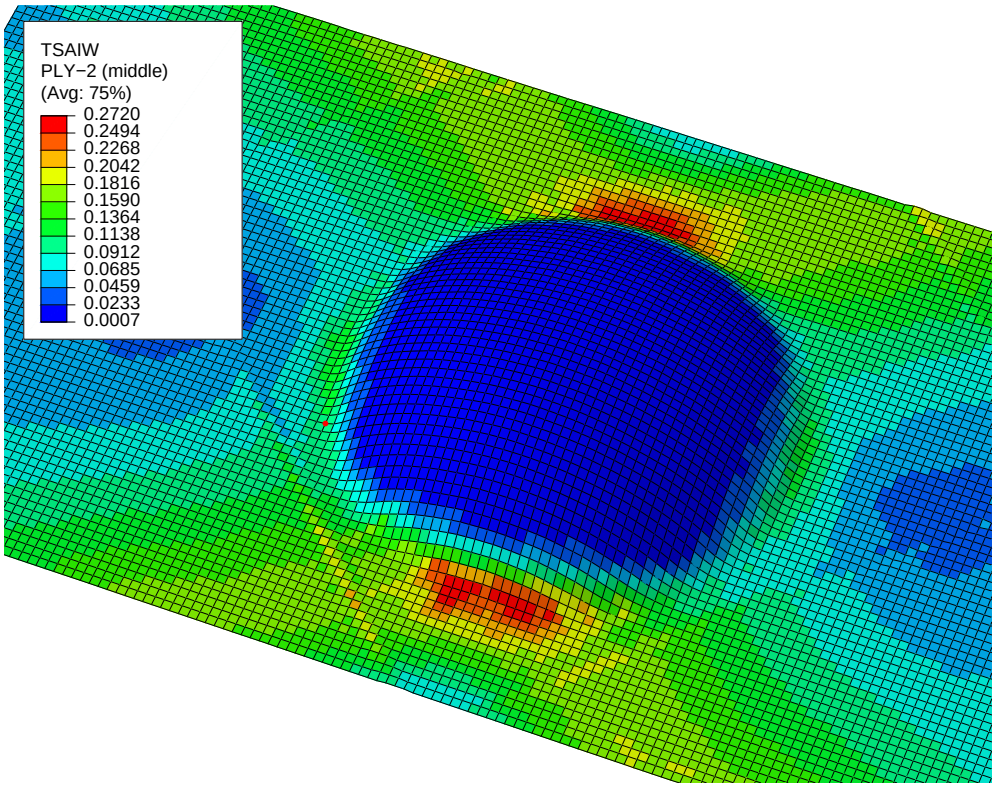


(b) As-formed model

Figure D.4: Failure-index in layer-1 ($[0/90]_2$ layup)

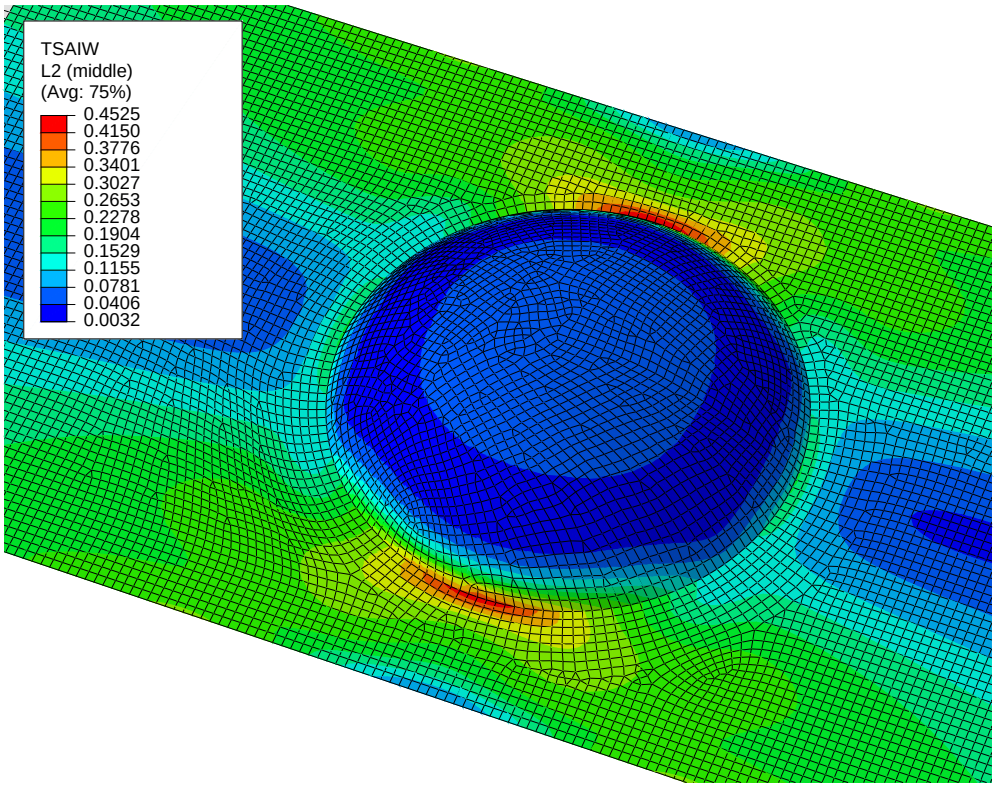


(a) Idealized model

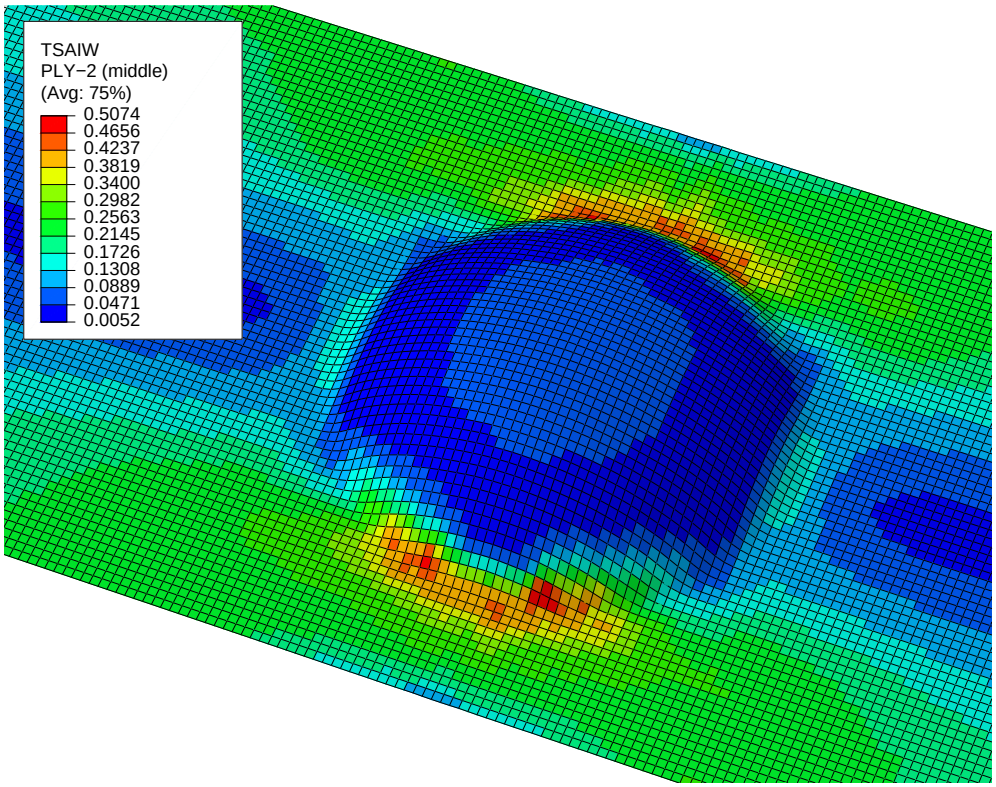


(b) As-formed model

Figure D.5: Failure-index in layer-2 ($[\pm 45^{\circ}_2]$ layup)

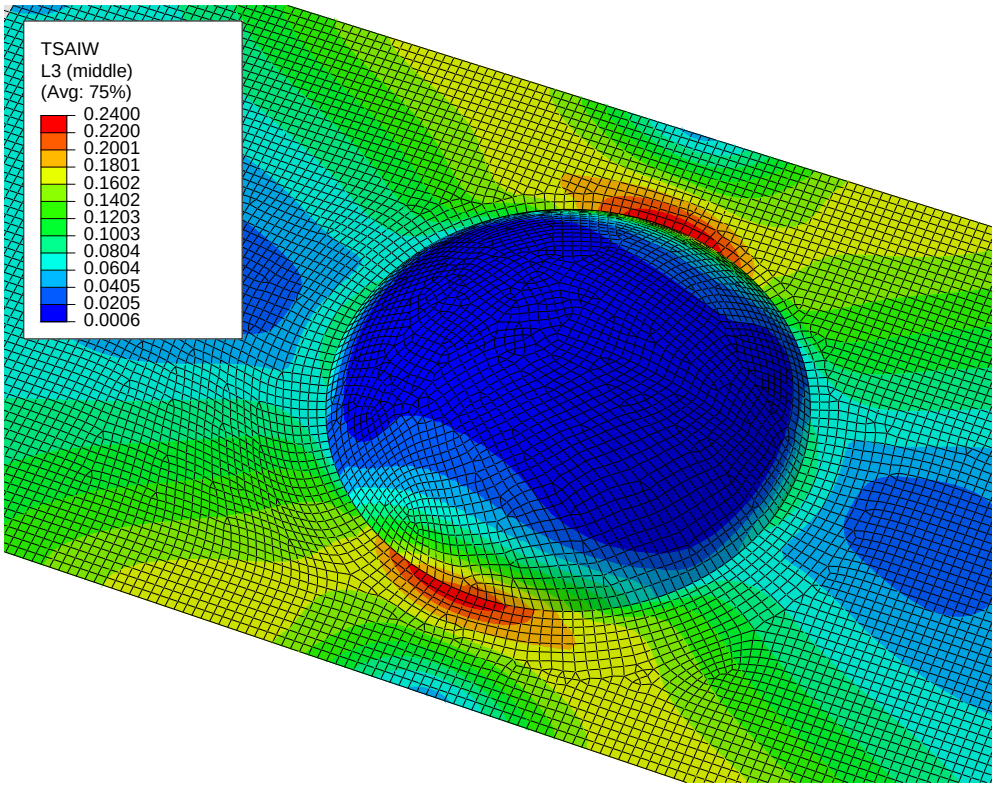


(a) Idealized model

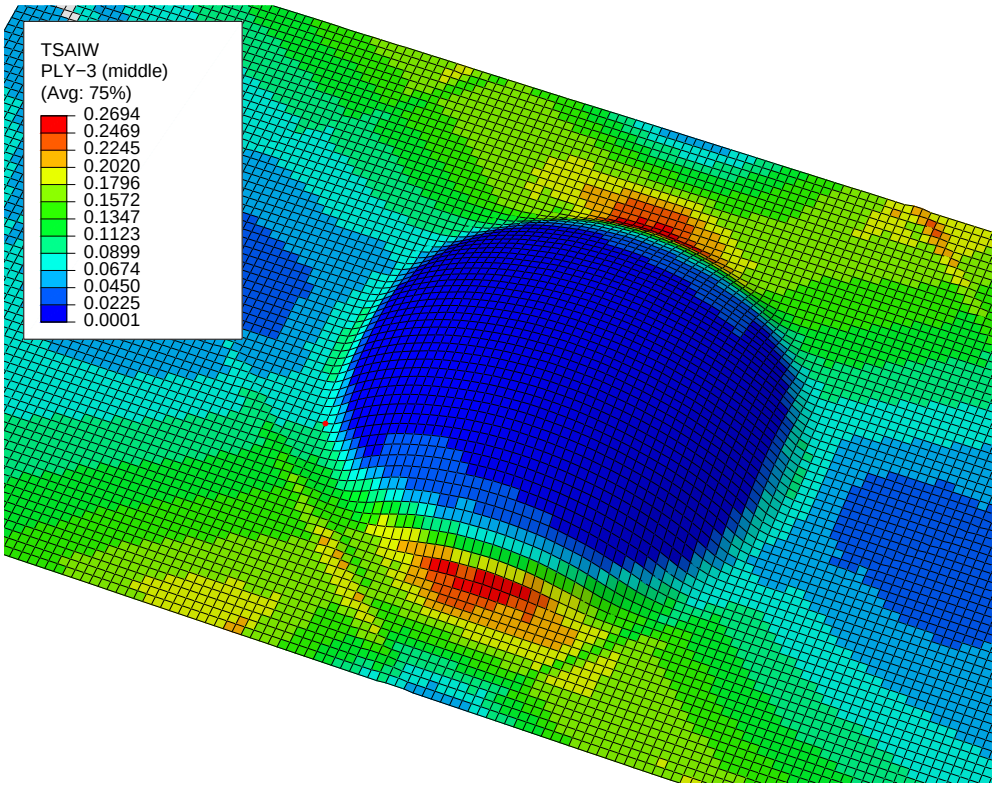


(b) As-formed model

Figure D.6: Failure-index in layer-2 ($[0/90]_2$ layup)

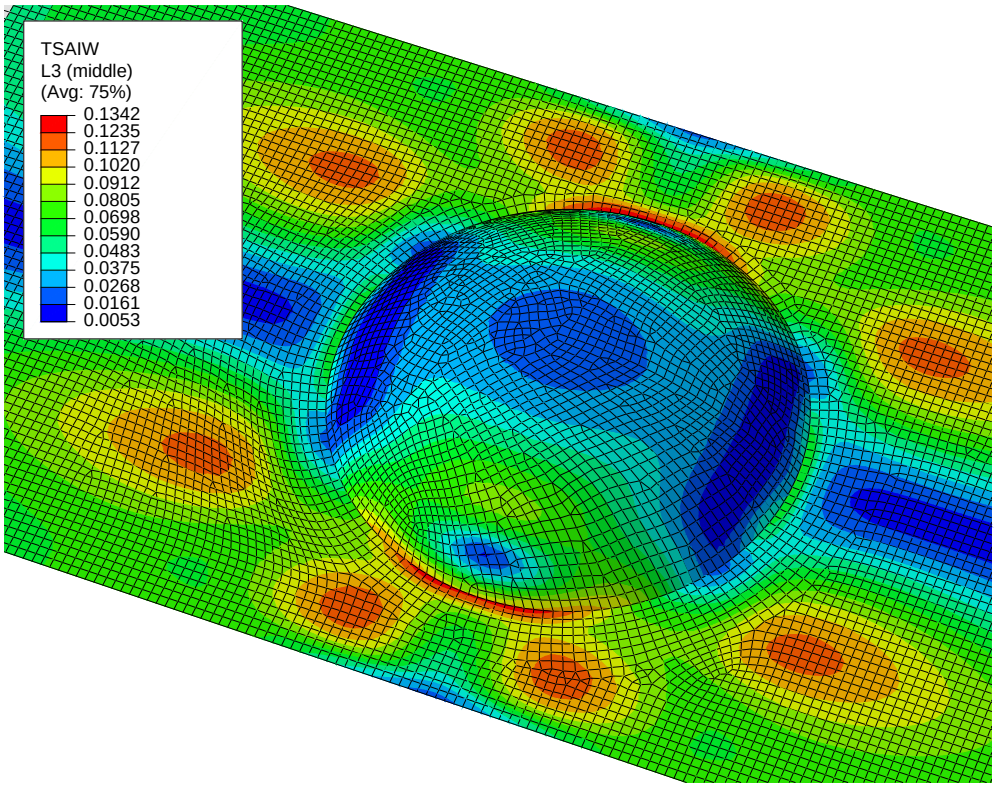


(a) Idealized model

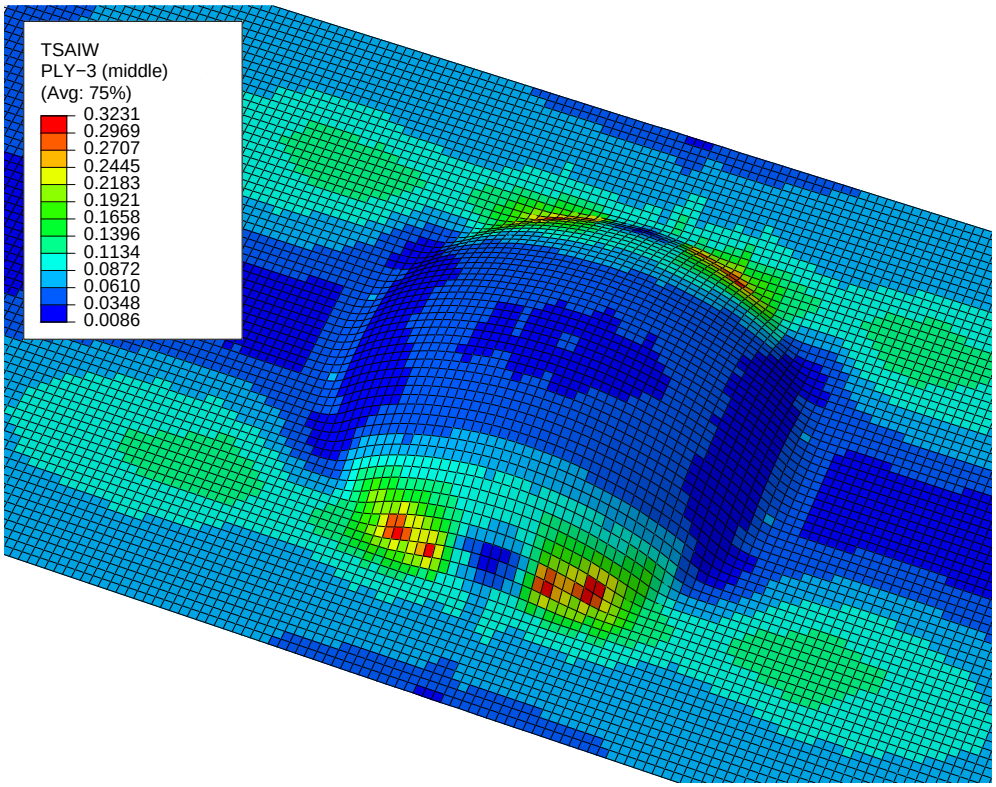


(b) As-formed model

Figure D.7: Failure-index in layer-3 ($[\pm 45^{\circ}_2]$ layup)

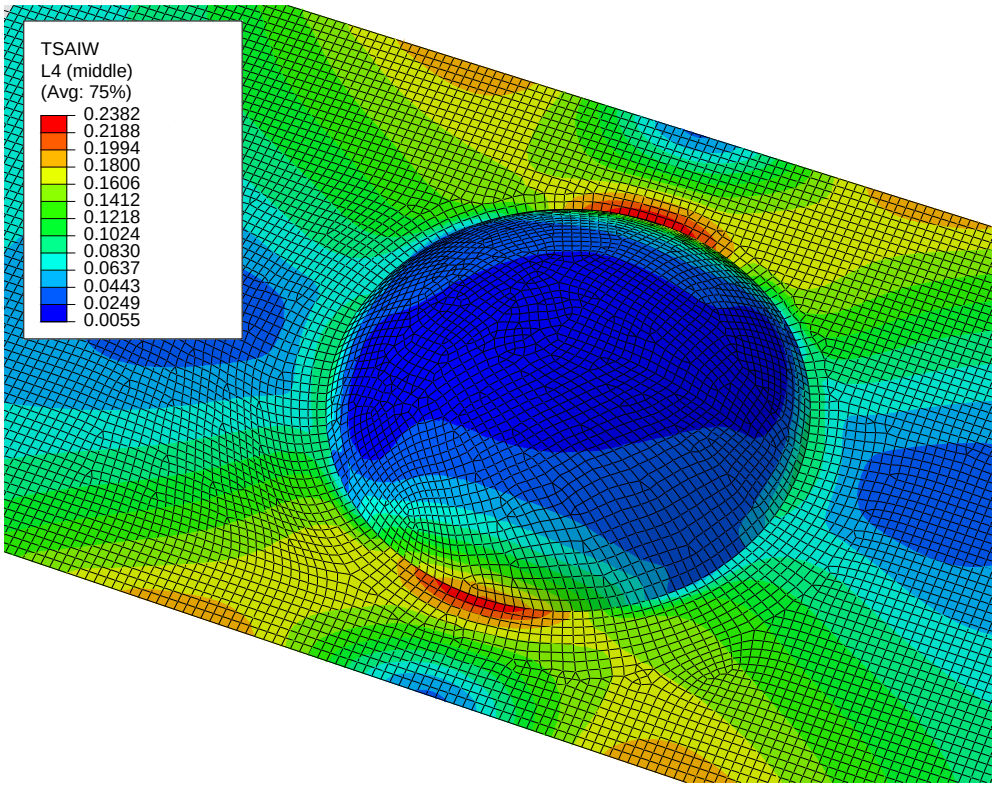


(a) Idealized model

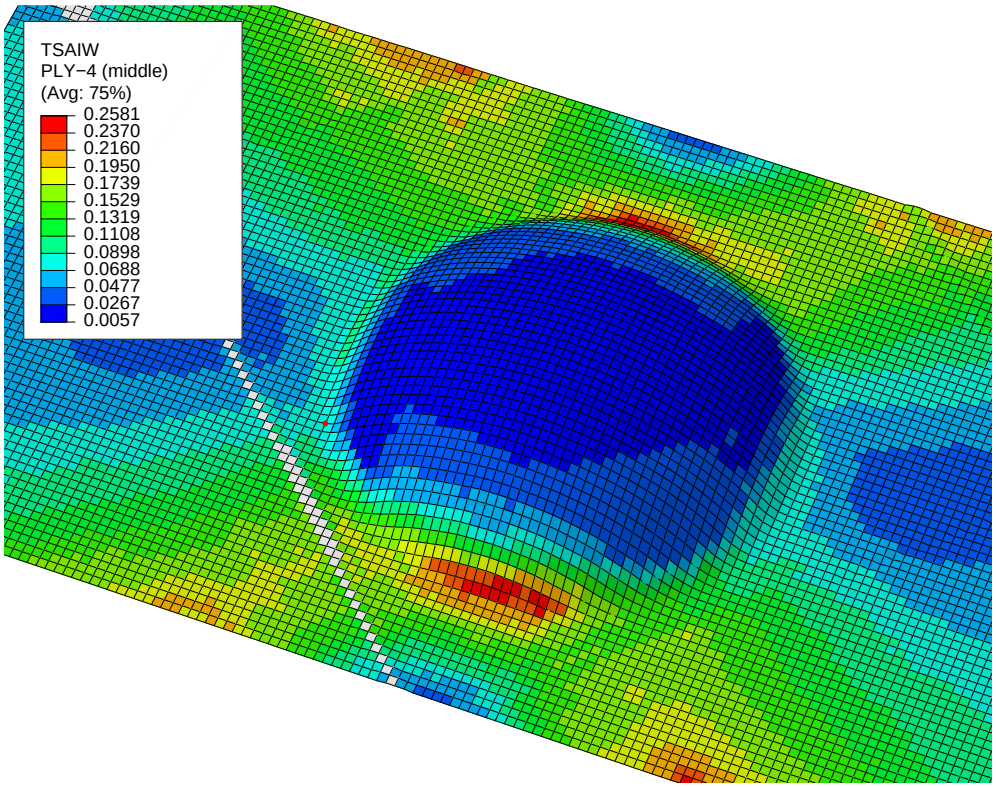


(b) As-formed model

Figure D.8: Failure-index in layer-3 ($[0/90]_2$ layup)

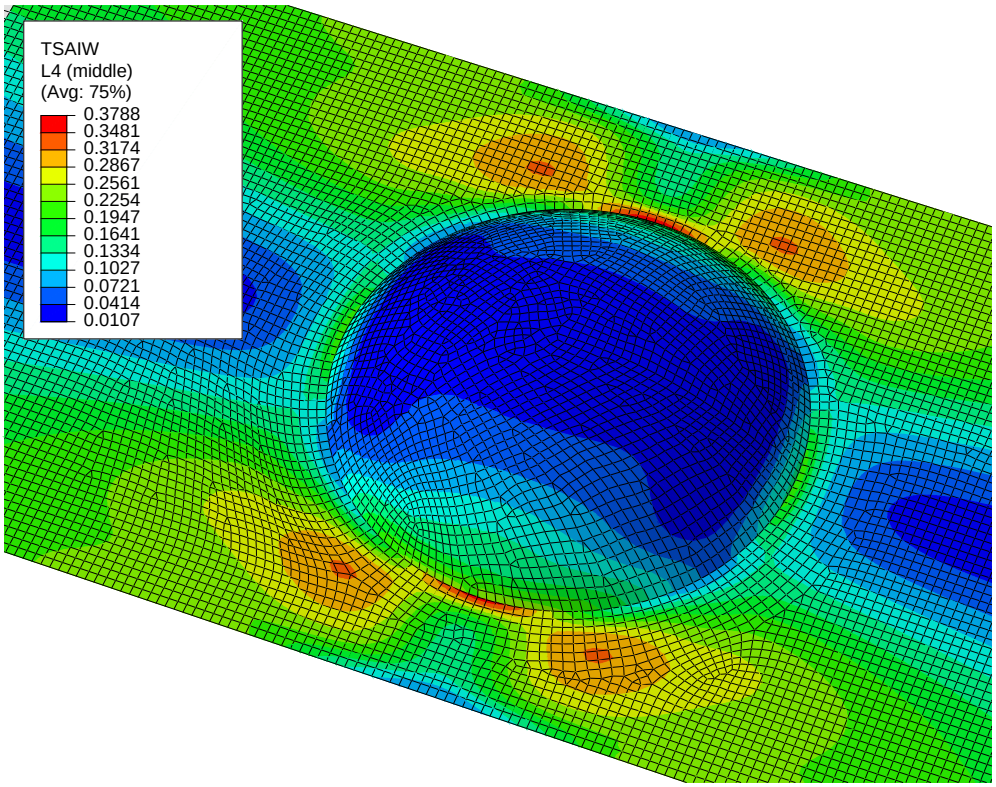


(a) Idealized model

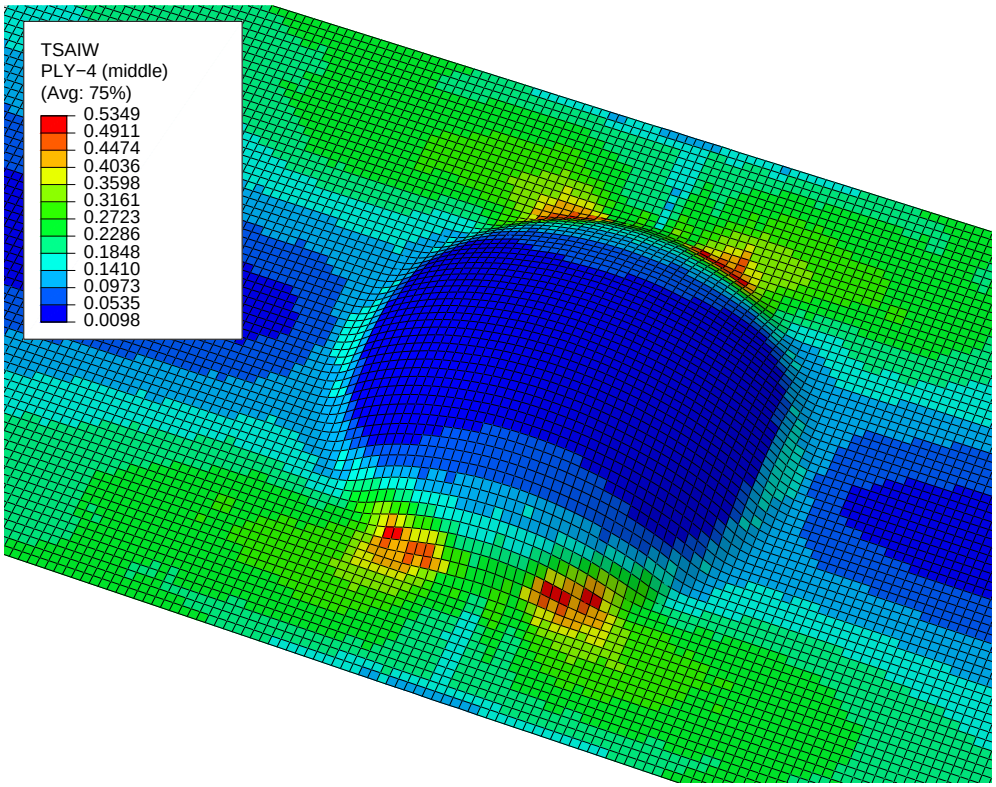


(b) As-formed model

Figure D.9: Failure-index in layer-4 ($[\pm 45^{\circ}_2]$ layup)

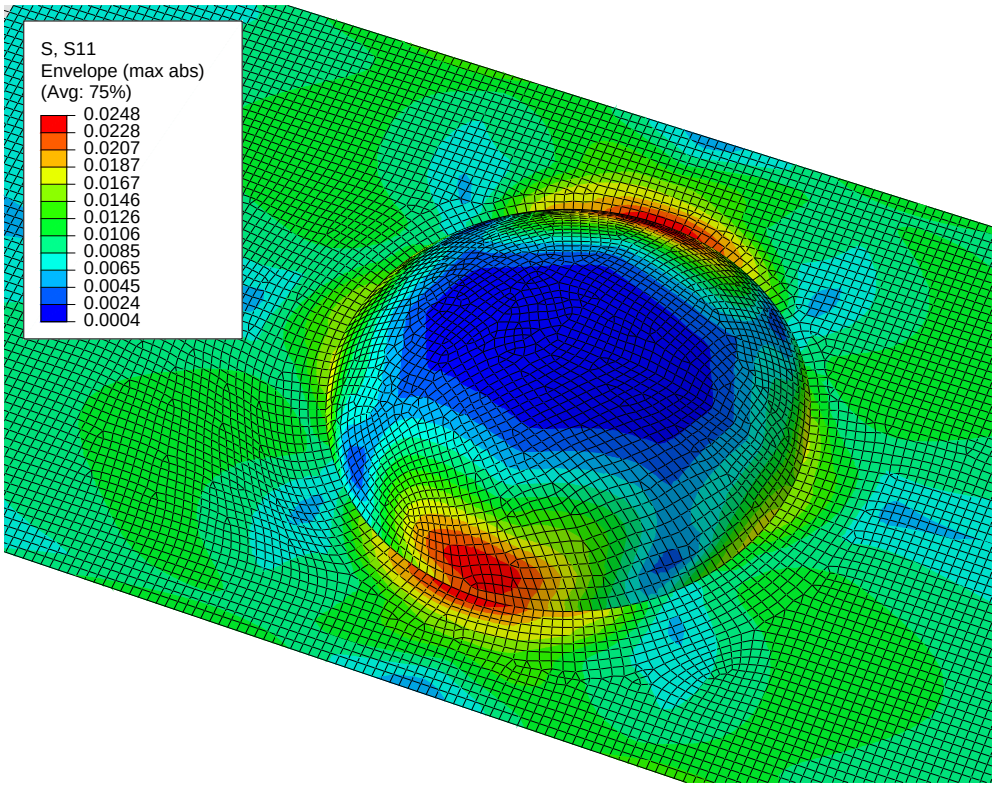


(a) Idealized model

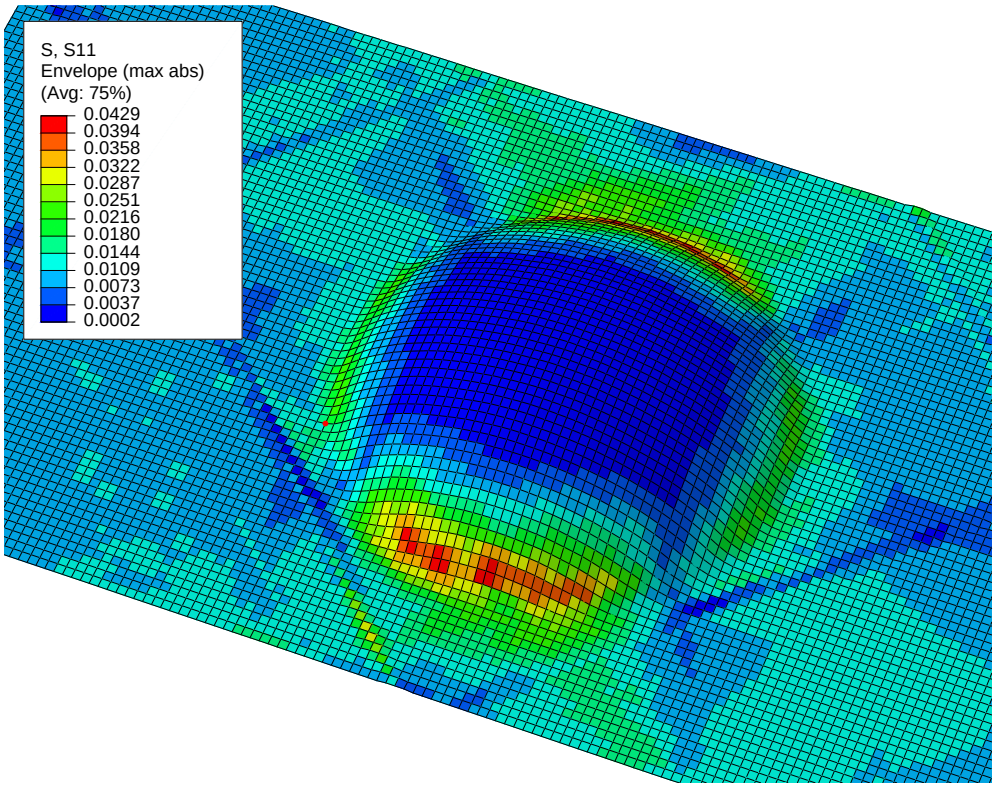


(b) As-formed model

Figure D.10: Failure-index in layer-4 ($[0/90]_2$ layup)

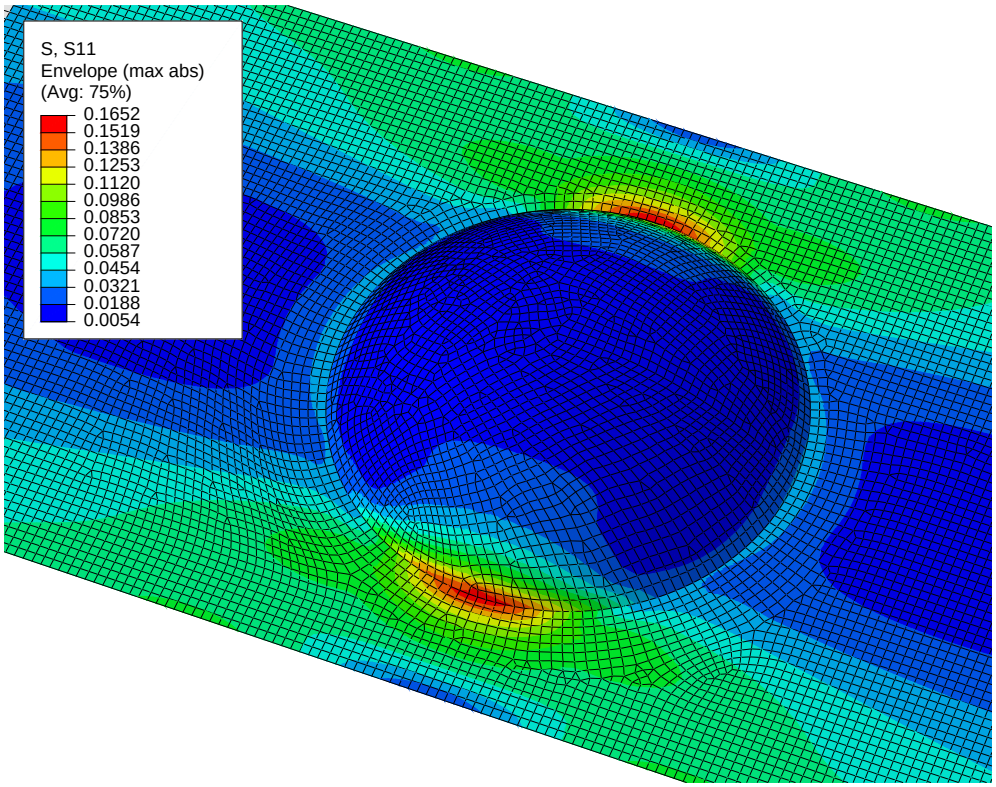


(a) Idealized model

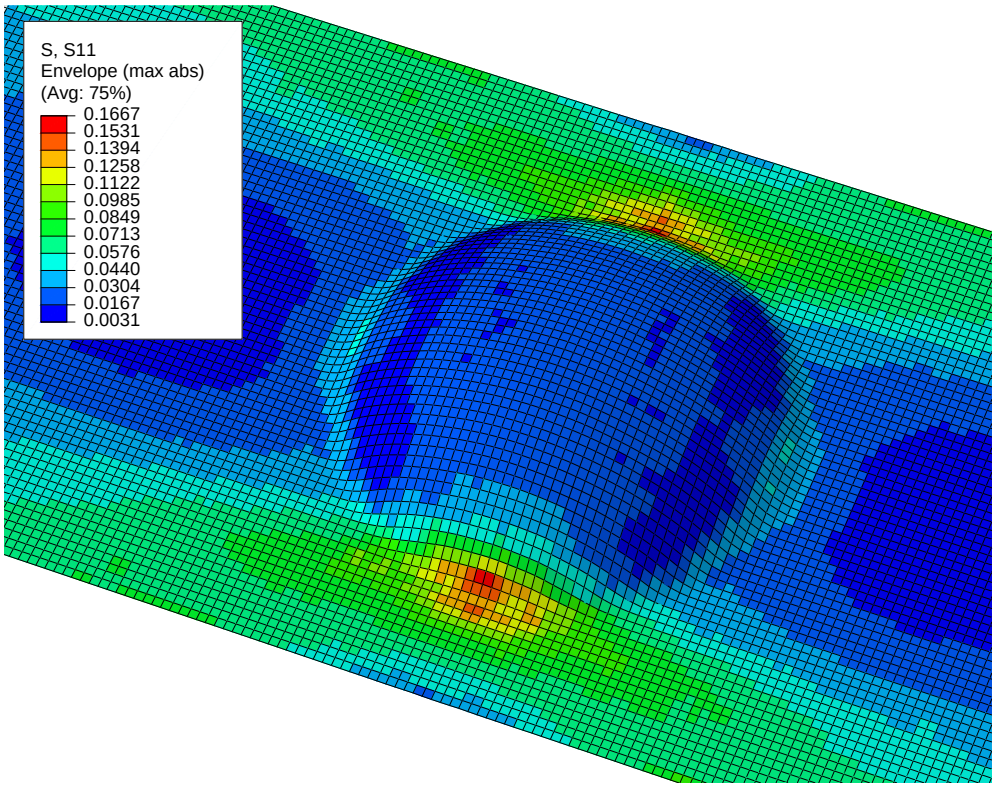


(b) As-formed model

Figure D.11: Fiber-stress envelope ($[\pm 45^\circ_2]$ layup)

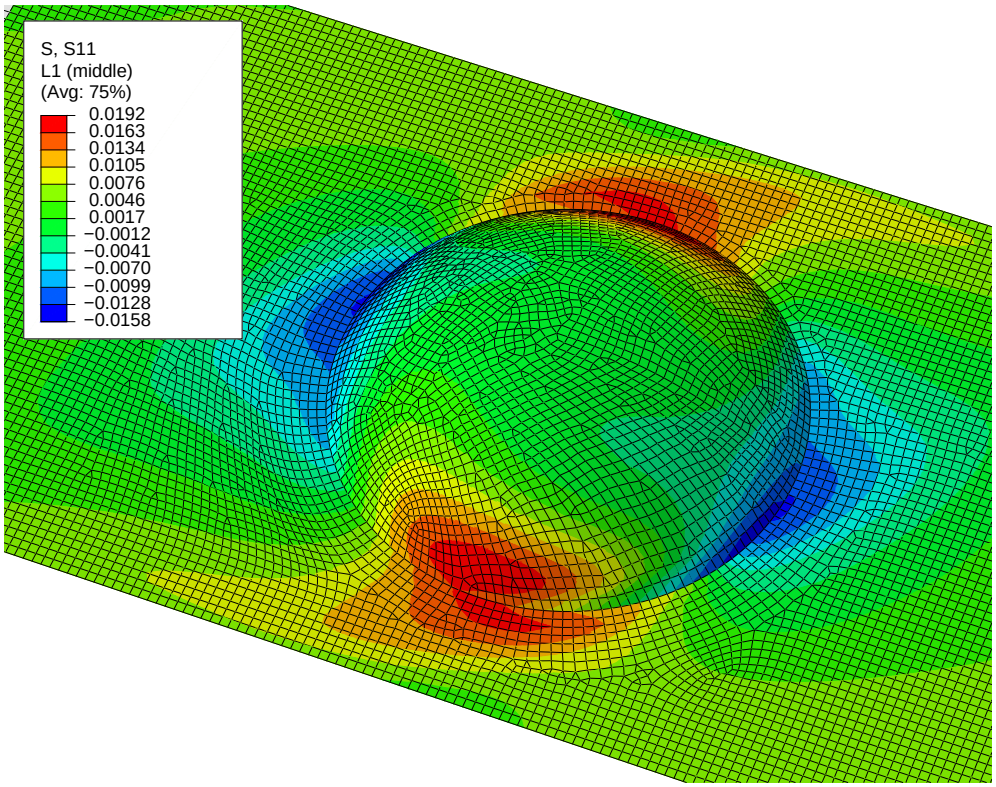


(a) Idealized model

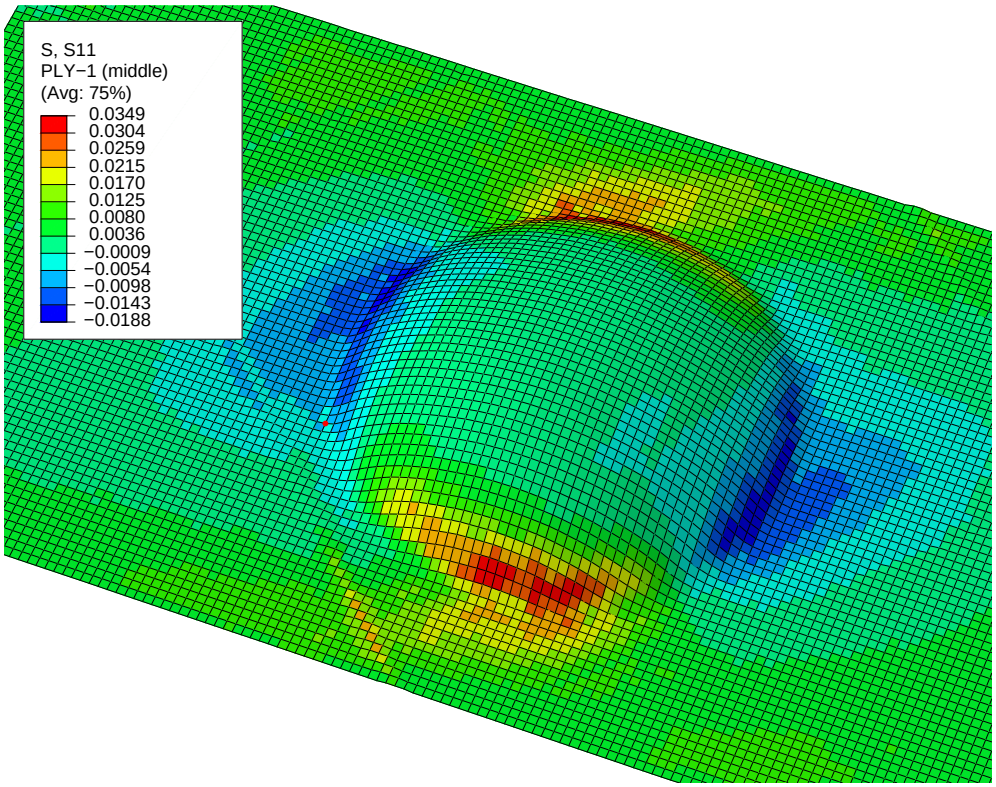


(b) As-formed model

Figure D.12: Fiber-stress envelope ($[0/90]_2$ layup)

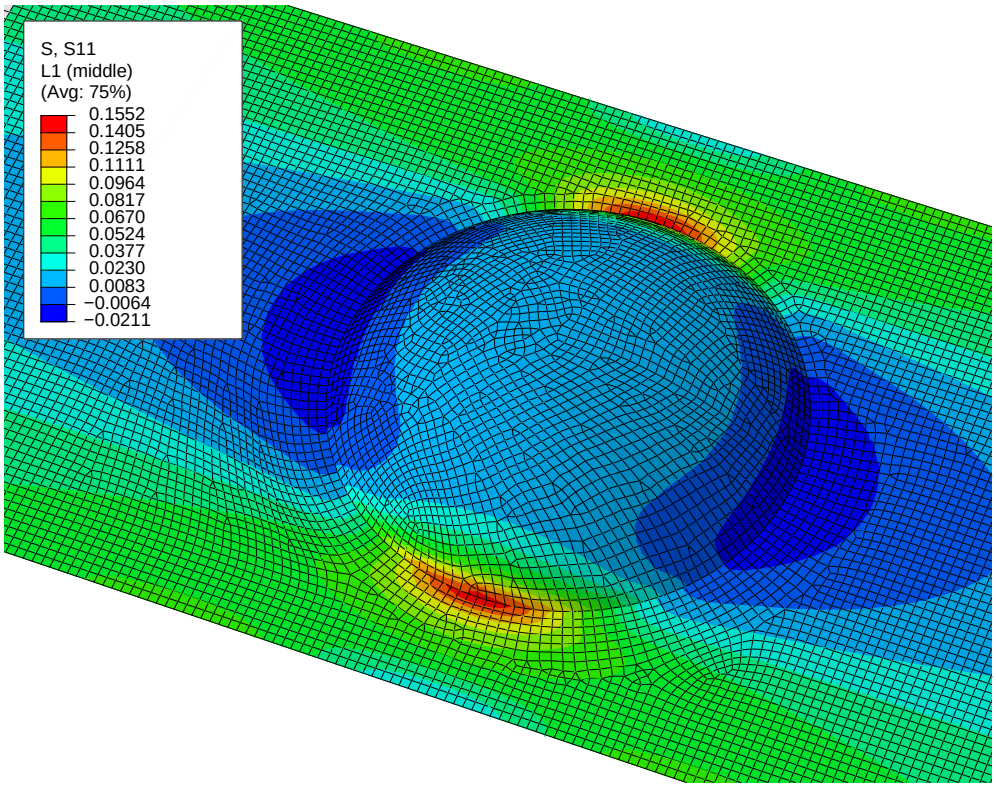


(a) Idealized model

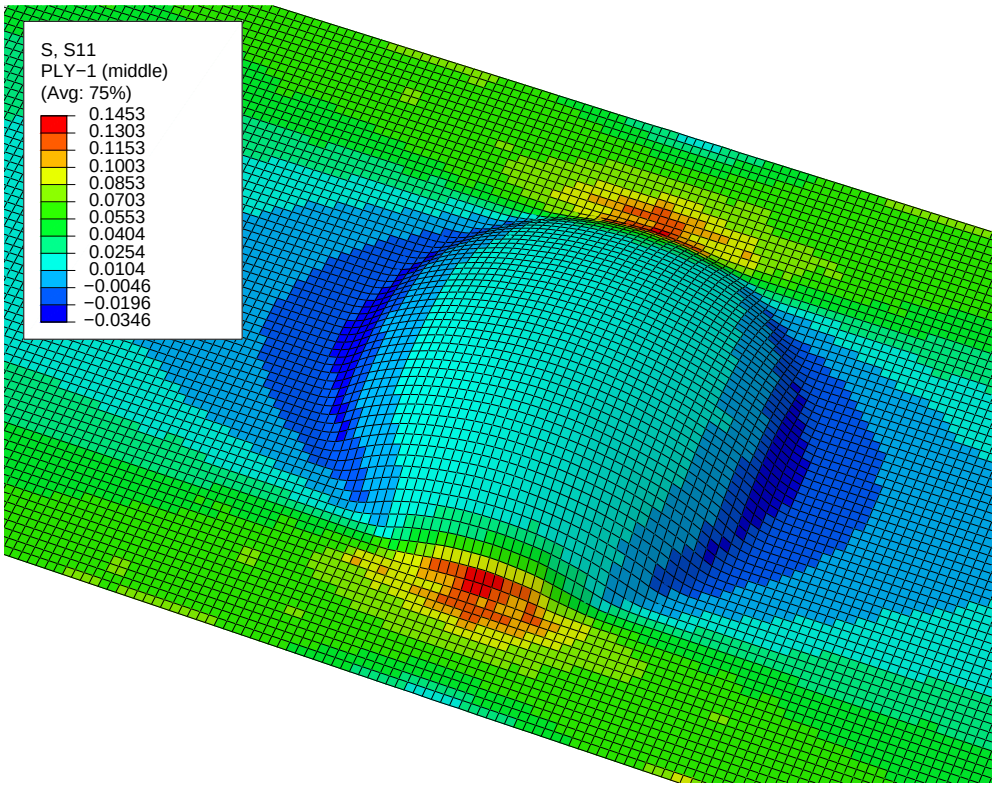


(b) As-formed model

Figure D.13: Fiber-stress in layer-1 ($[\pm 45^{\circ}_2]$ layup)

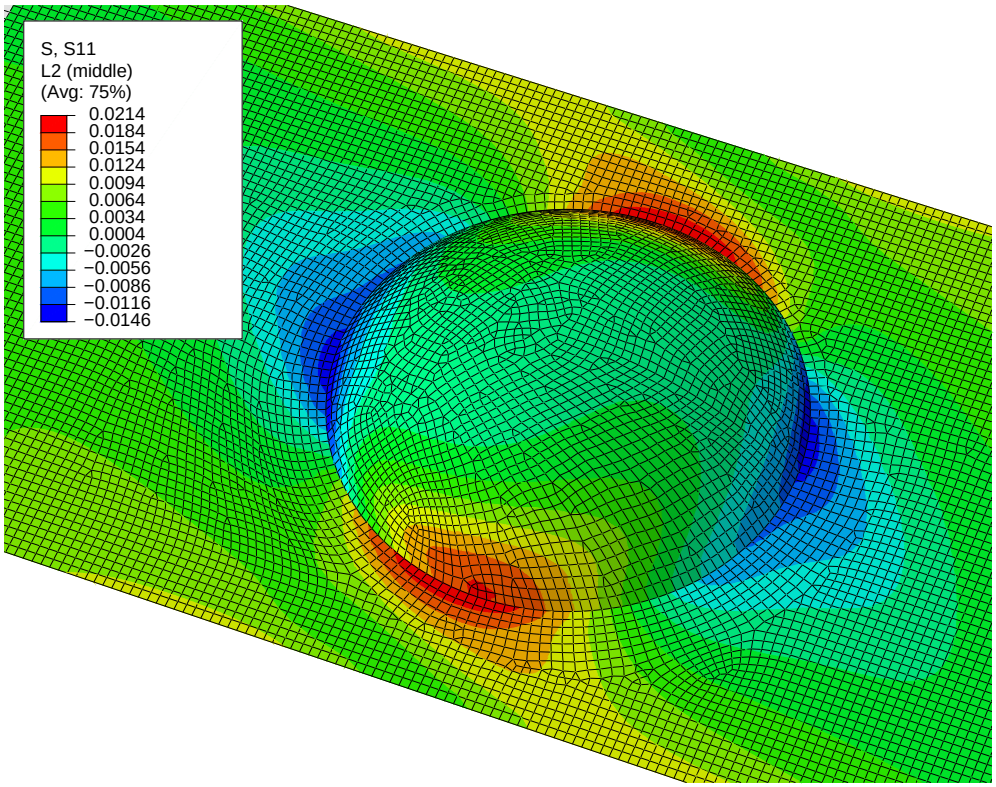


(a) Idealized model

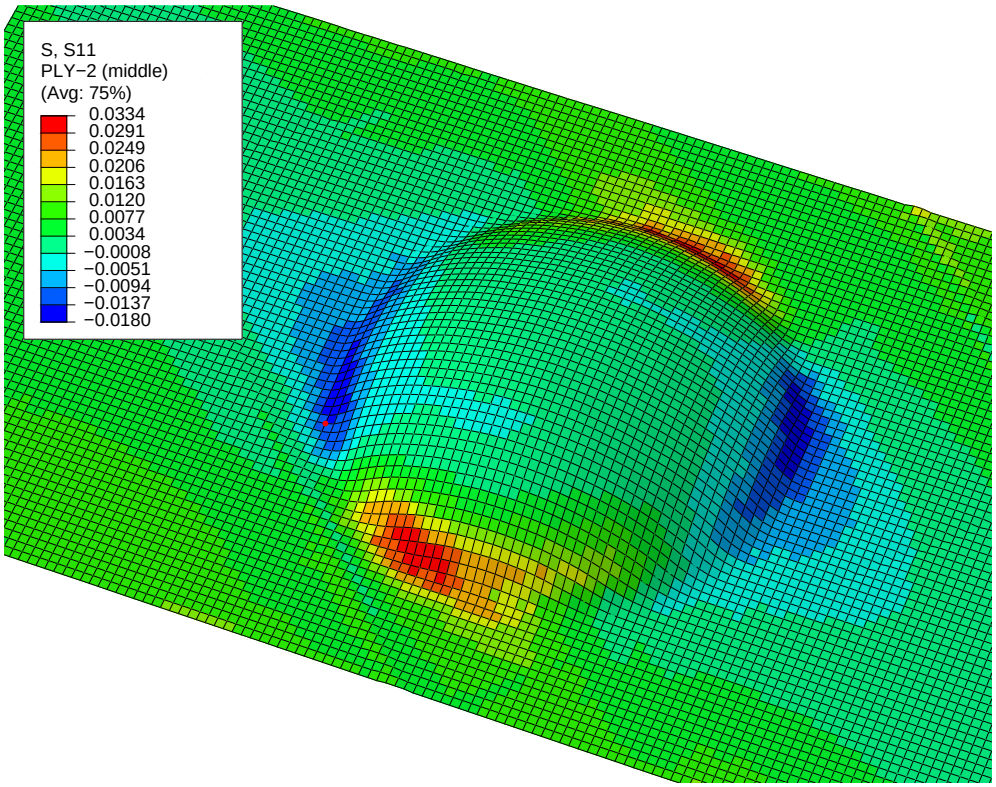


(b) As-formed model

Figure D.14: Fiber-stress in layer-1 ($[0/90]_2$ layup)

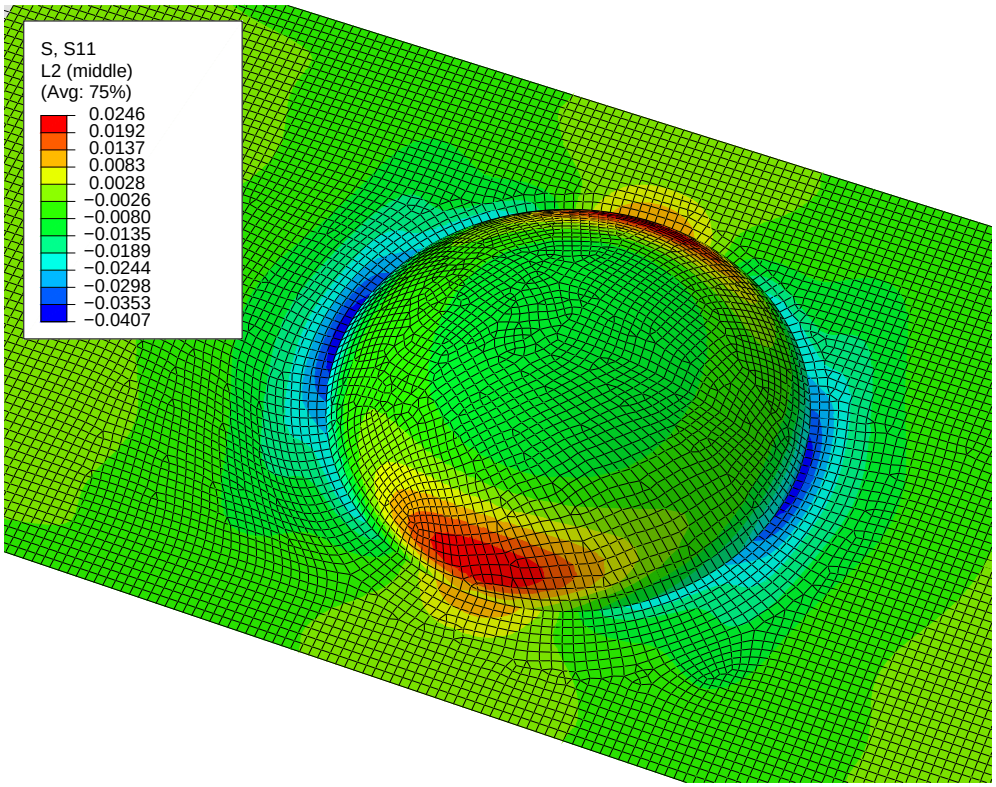


(a) Idealized model

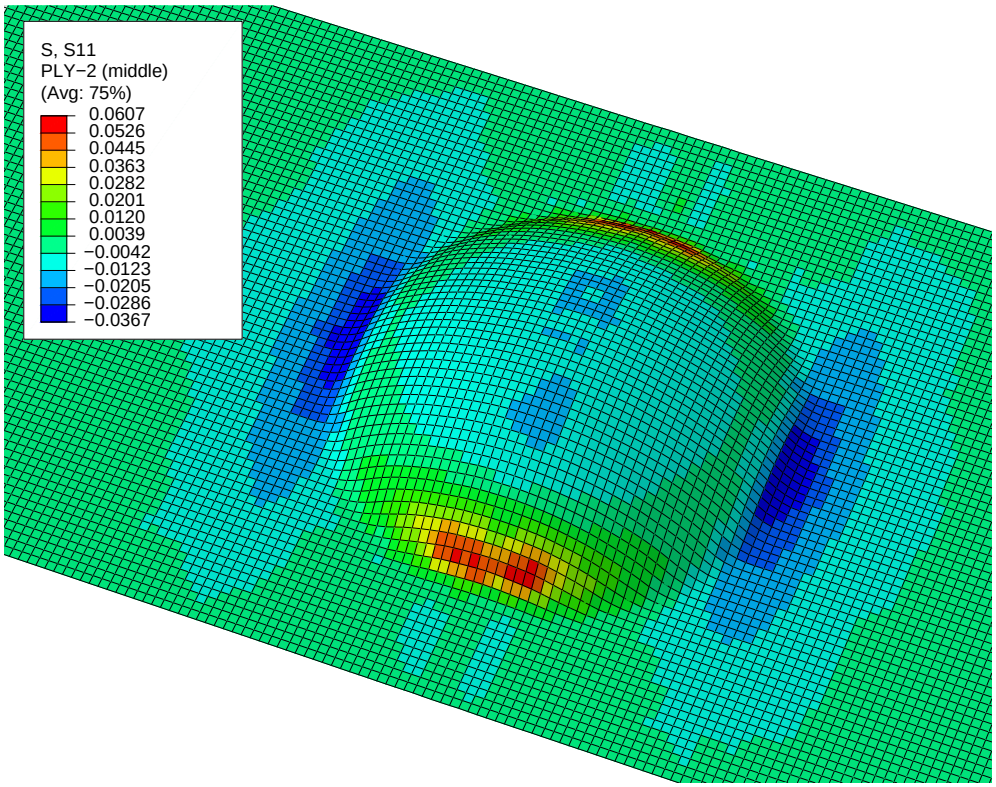


(b) As-formed model

Figure D.15: Fiber-stress in layer-2 ($[\pm 45^\circ_2]$ layup)

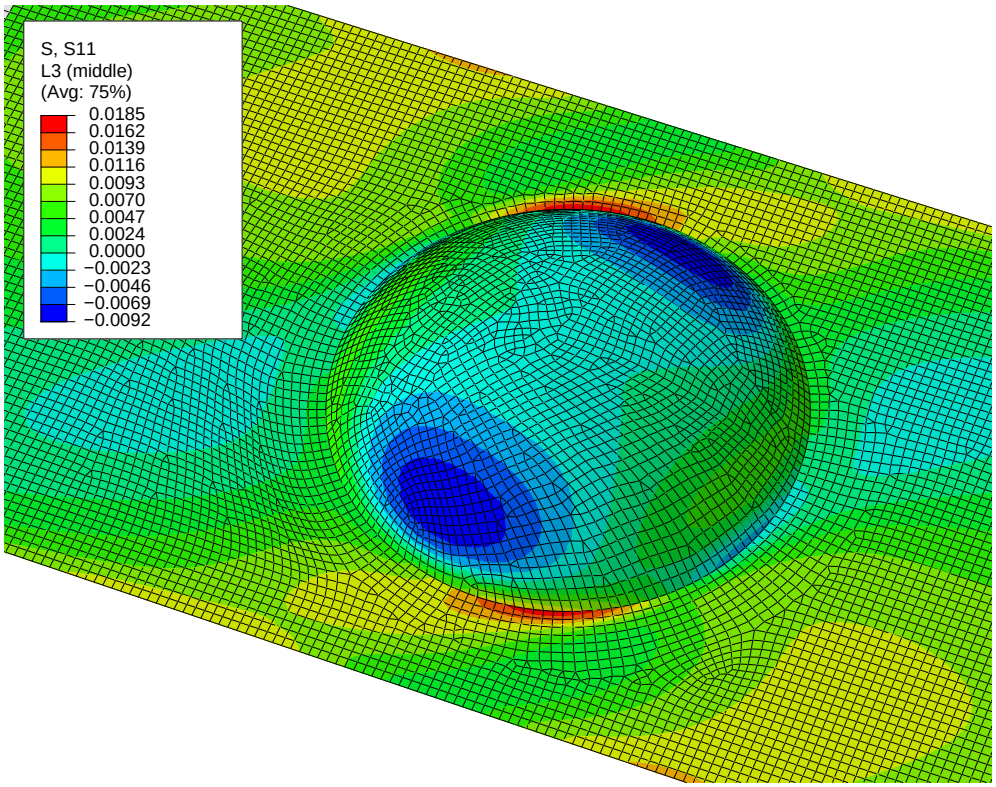


(a) Idealized model

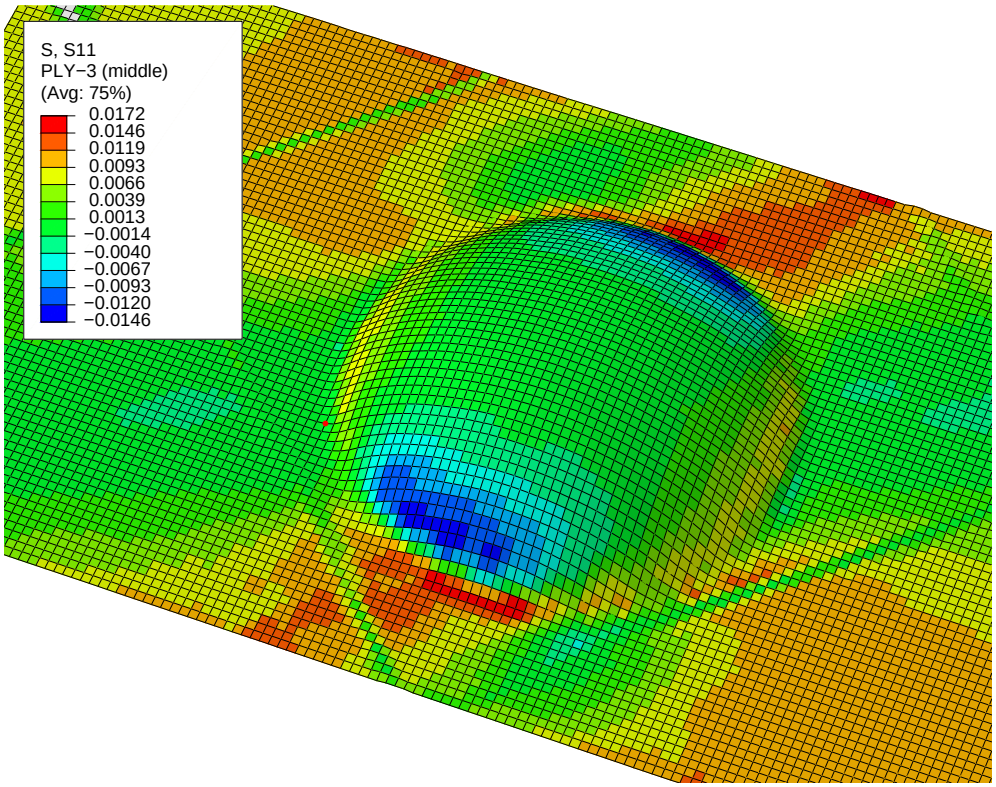


(b) As-formed model

Figure D.16: Fiber-stress in layer-2 ($[0/90]_2$ layup)

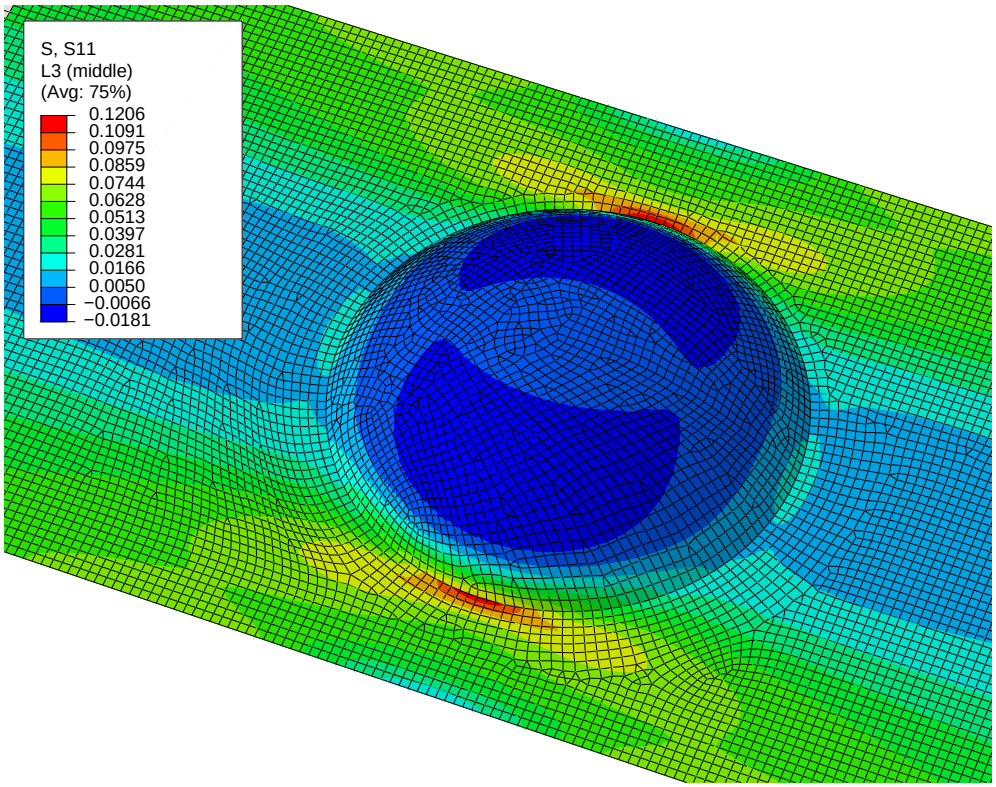


(a) Idealized model

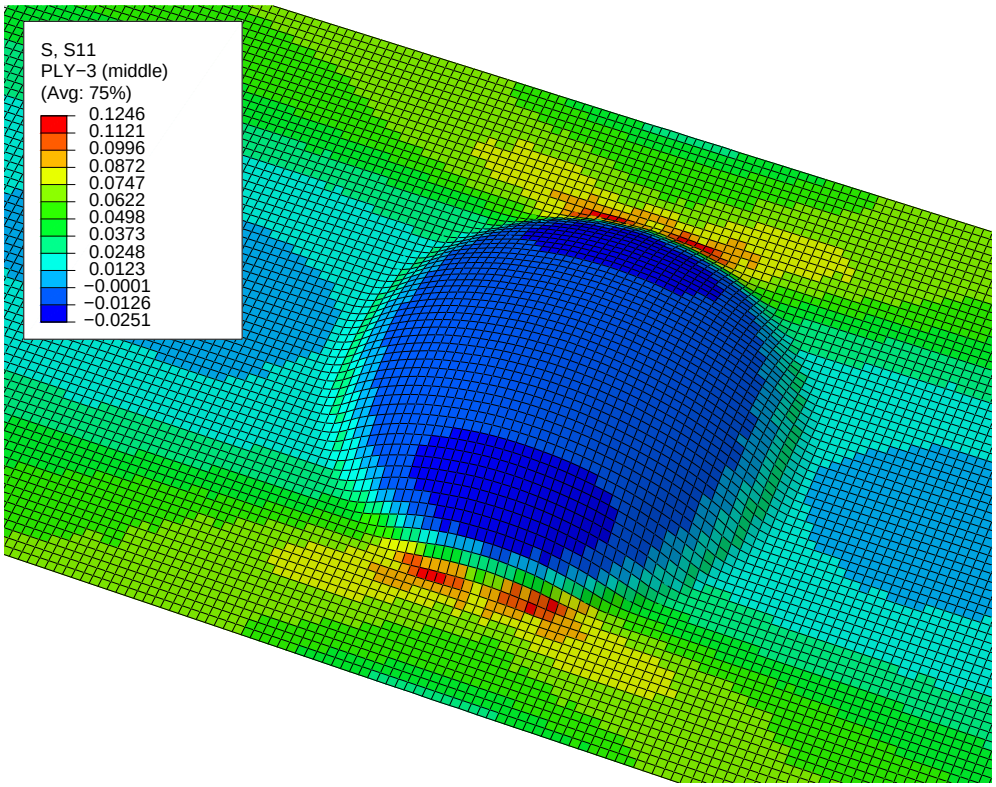


(b) As-formed model

Figure D.17: Fiber-stress in layer-3 ($[\pm 45^\circ_2]$ layup)

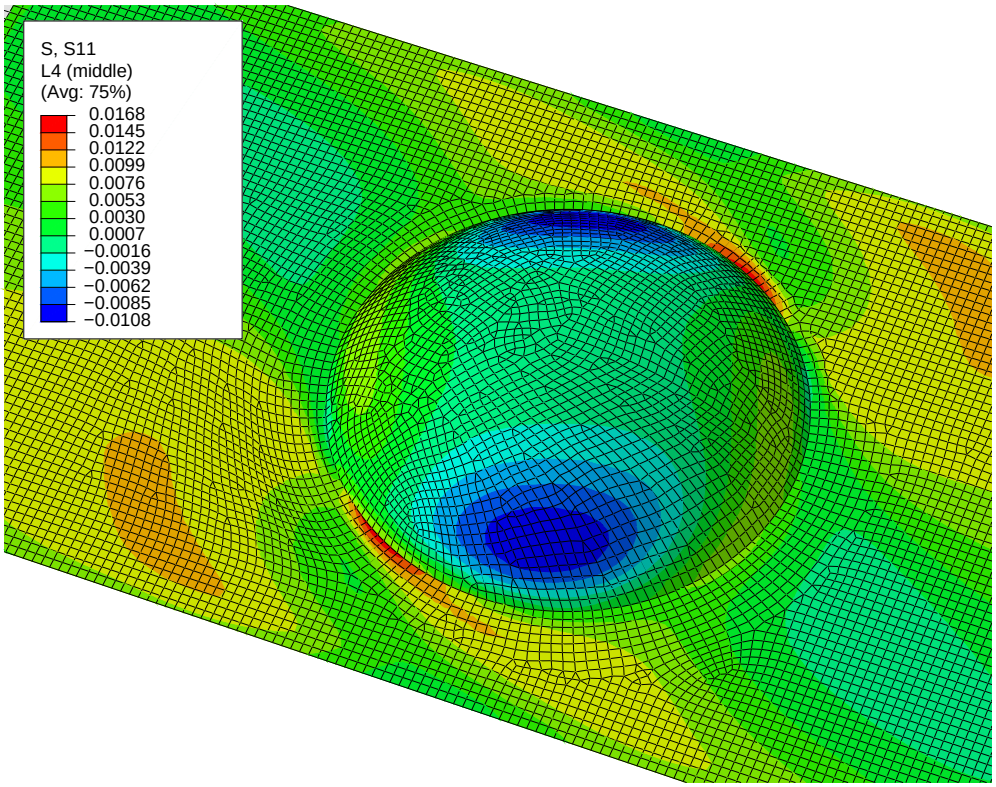


(a) Idealized model

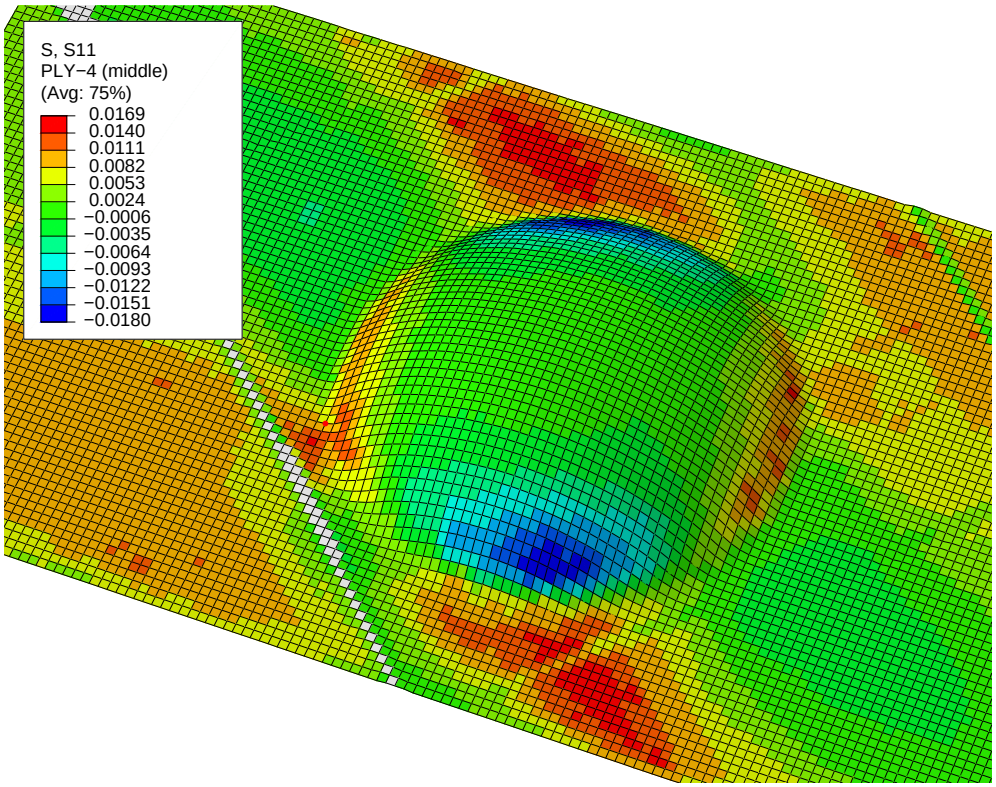


(b) As-formed model

Figure D.18: Fiber-stress in layer-3 ($[0/90]_2$ layup)

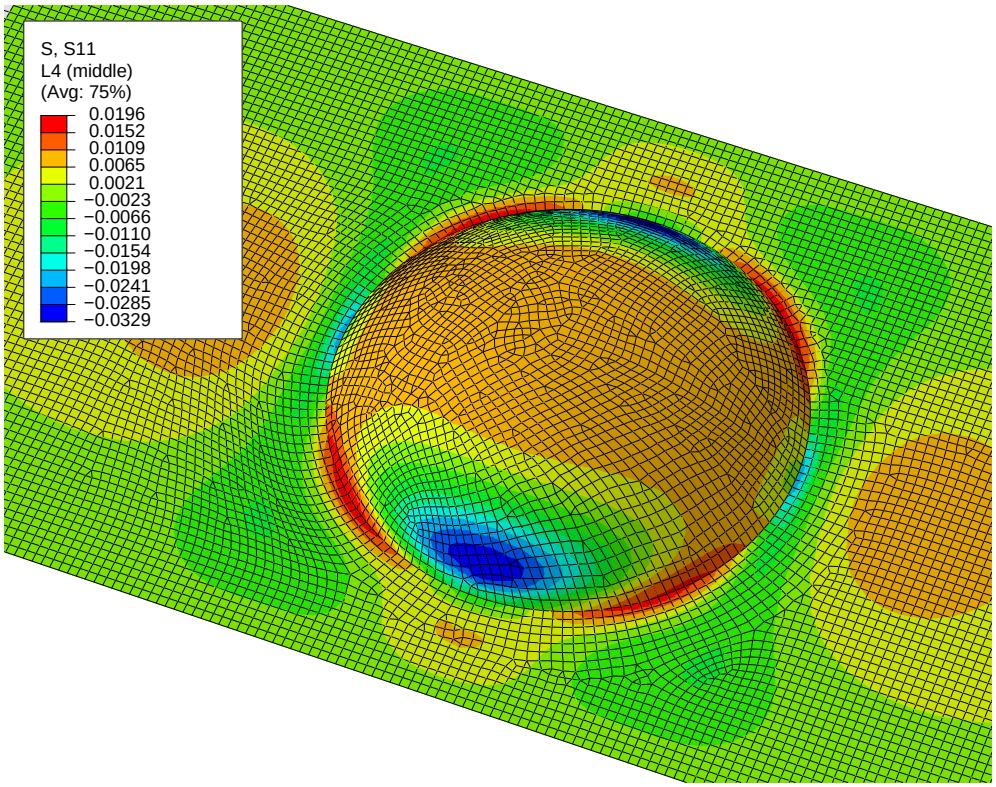


(a) Idealized model

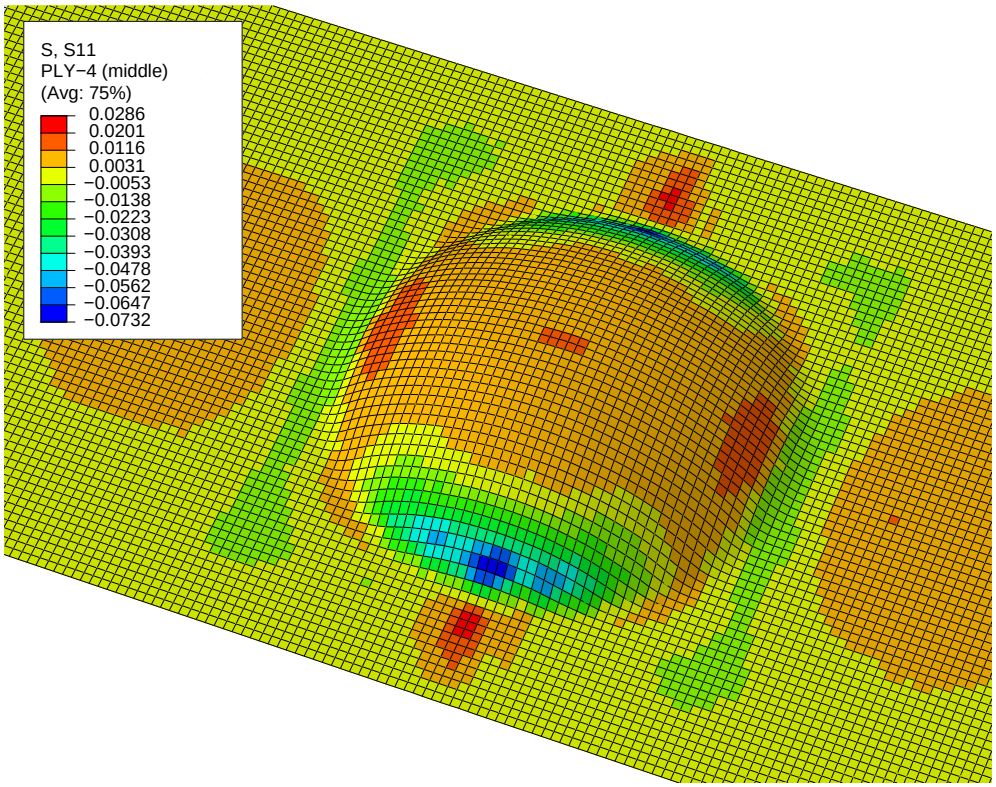


(b) As-formed model

Figure D.19: Fiber-stress in layer-4 ($[\pm 45^\circ_2]$ layup)

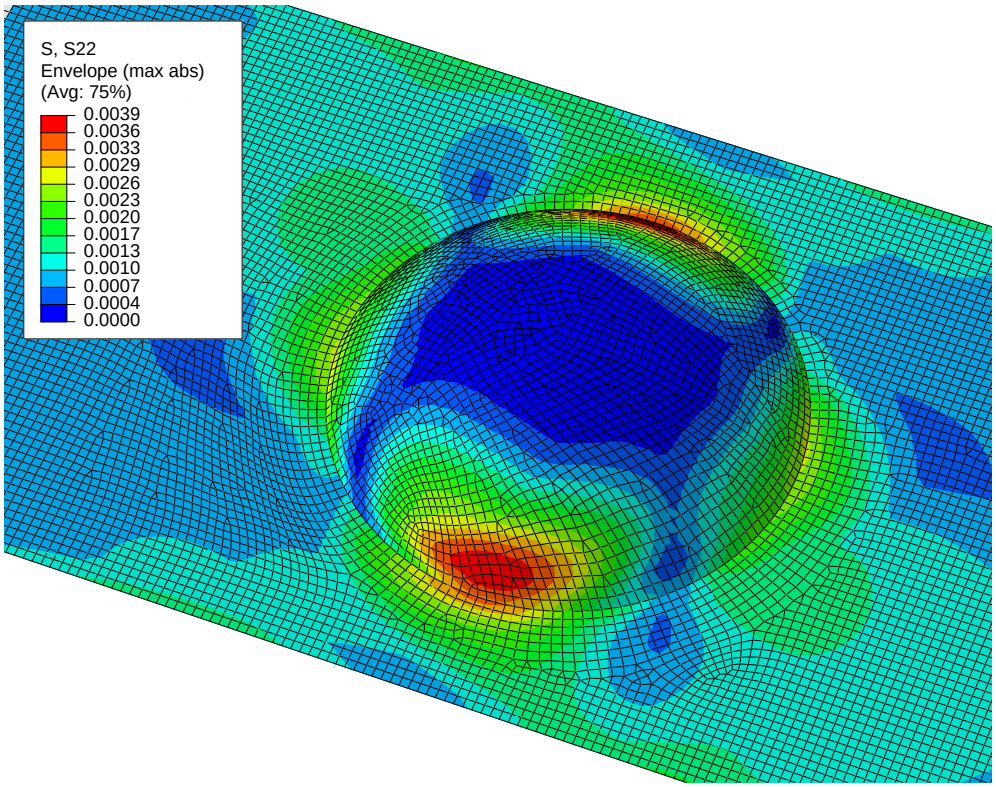


(a) Idealized model

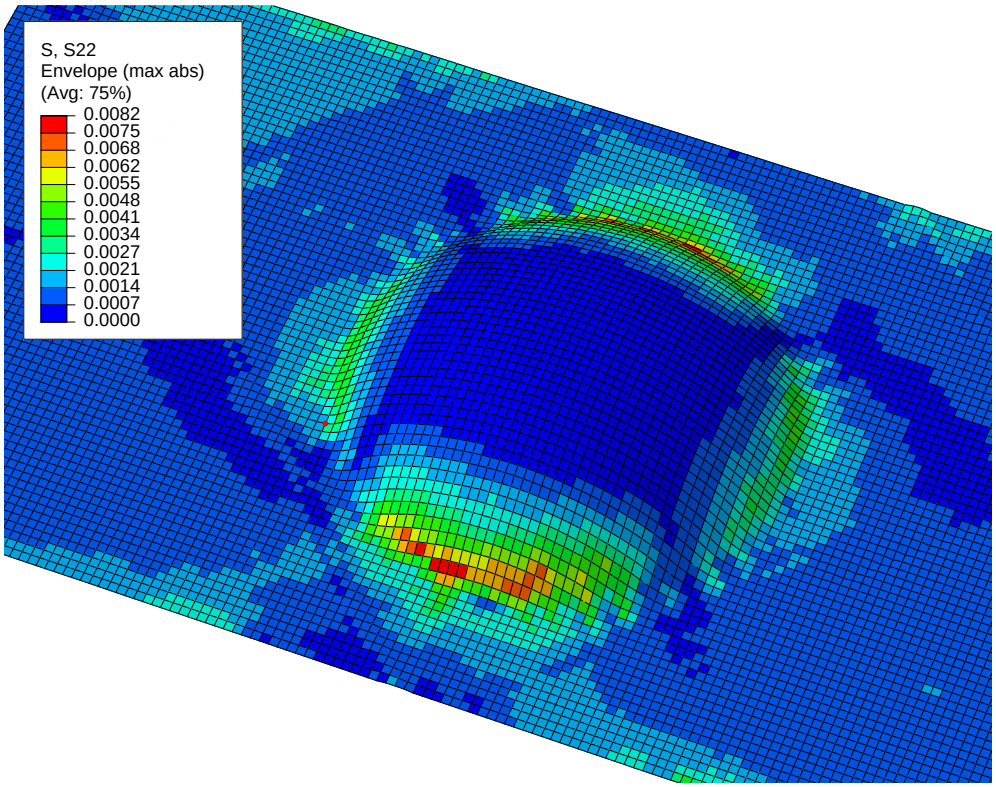


(b) As-formed model

Figure D.20: Fiber-stress in layer-4 ($[0/90]_2$ layup)

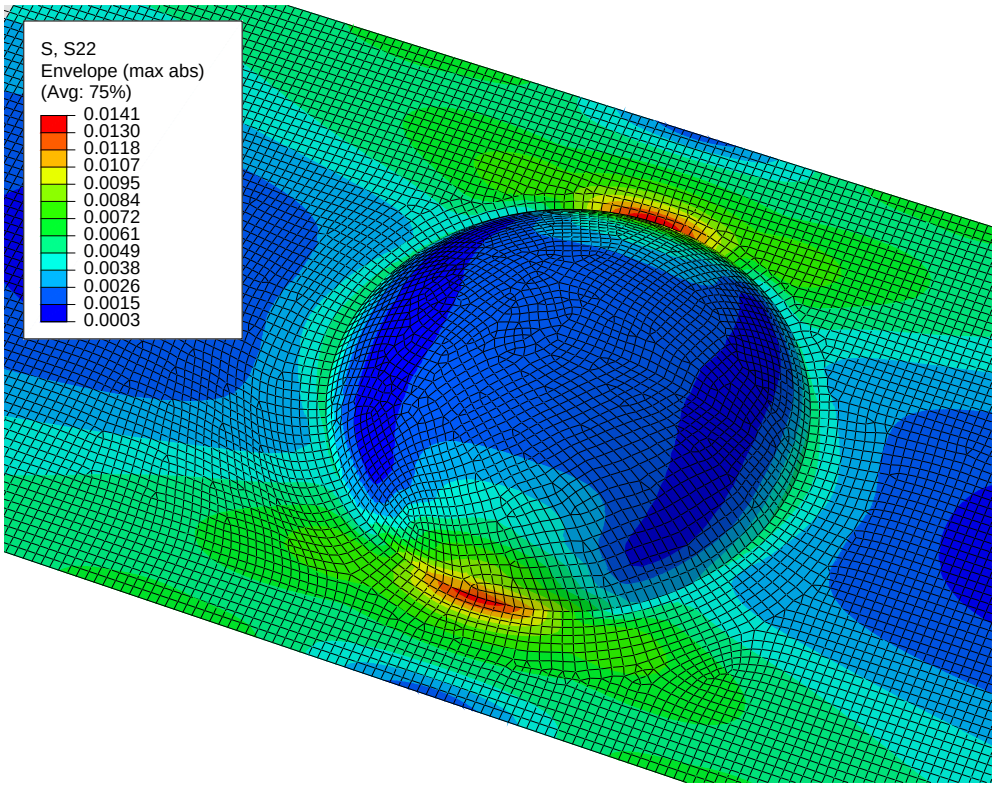


(a) Idealized model

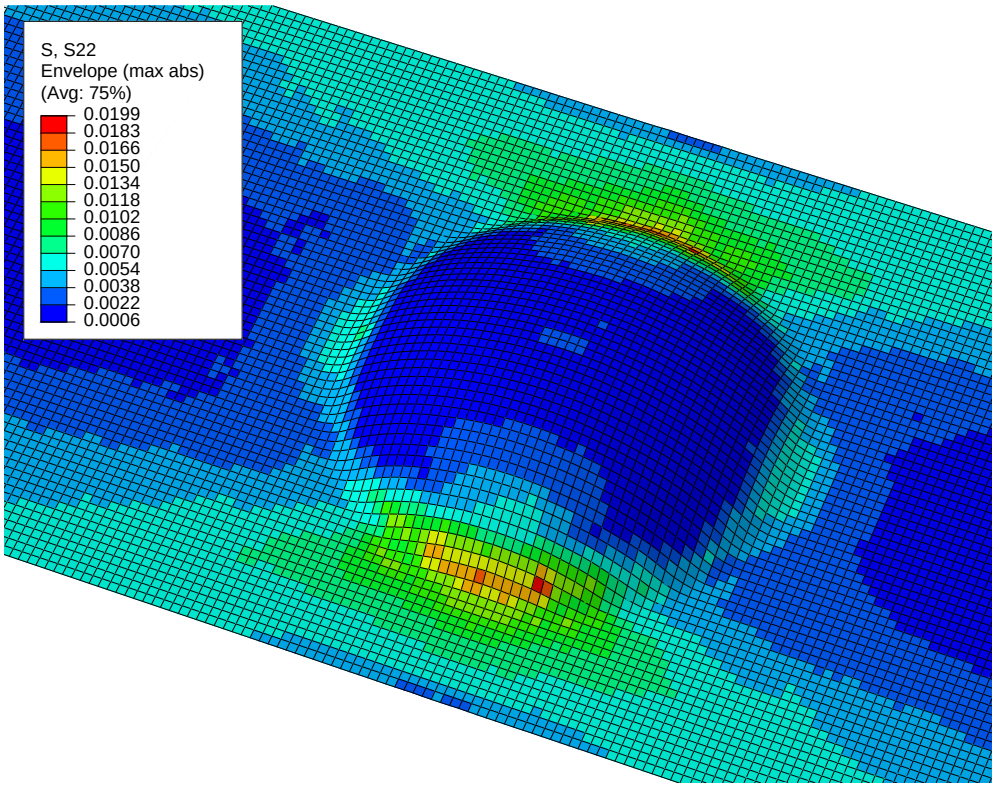


(b) As-formed model

Figure D.21: Transverse-stress envelope ($[\pm 45^\circ_2]$ layup)

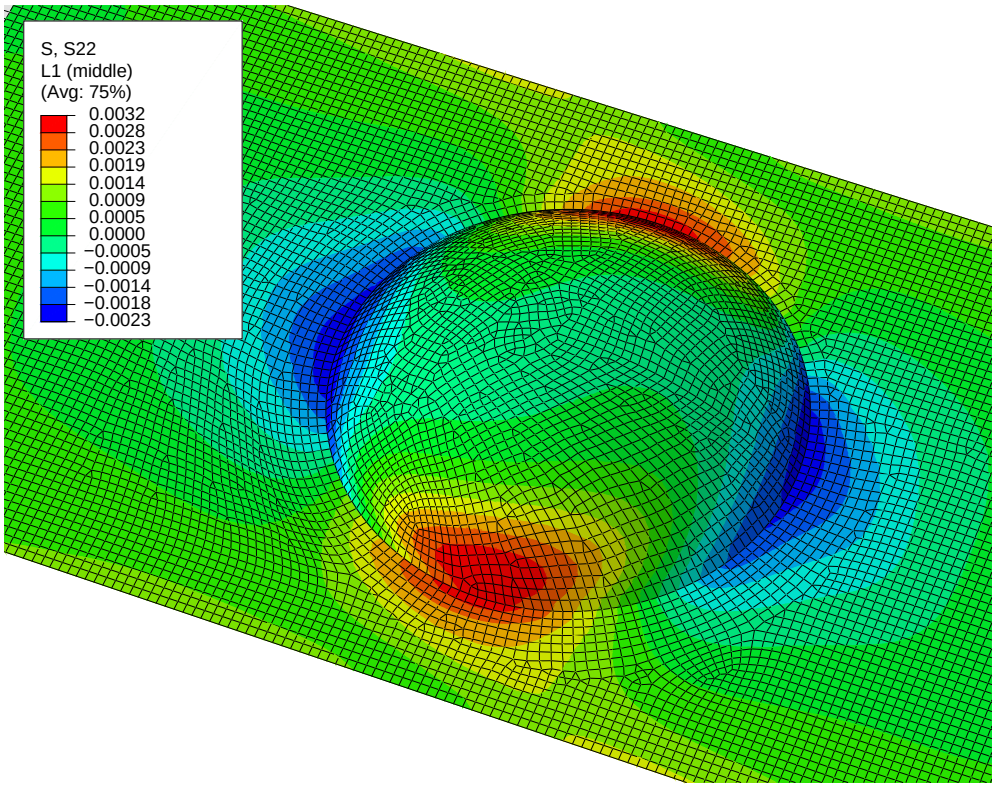


(a) Idealized model

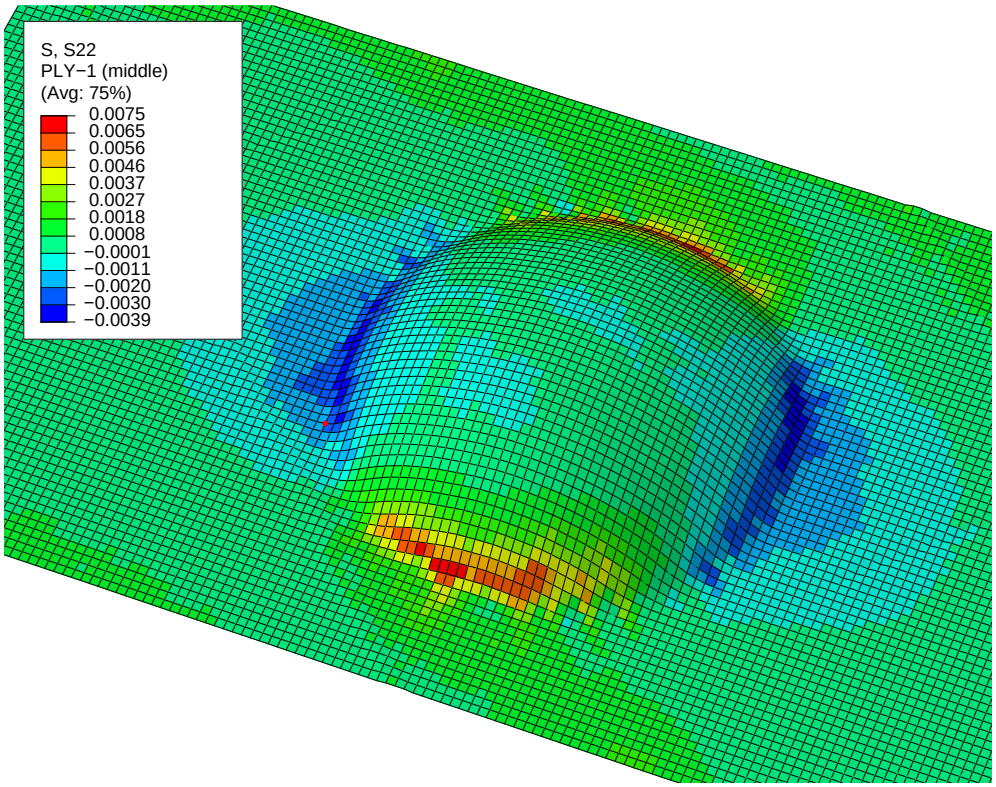


(b) As-formed model

Figure D.22: Transverse-stress envelope ($[0/90]_2$ layup)

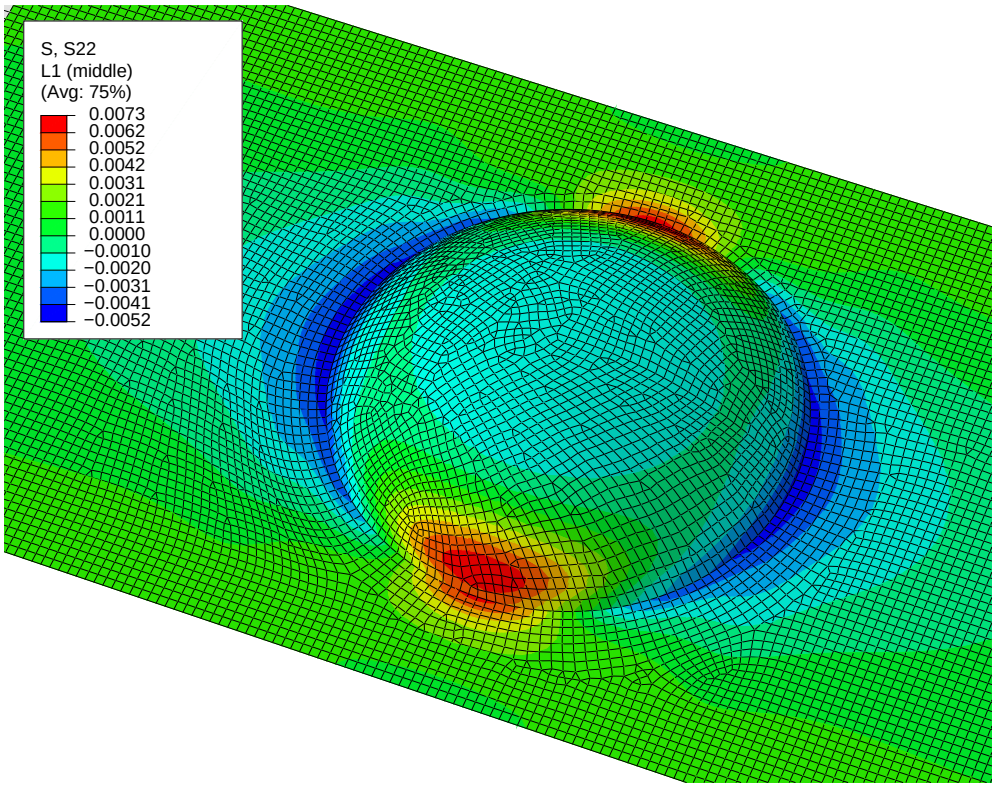


(a) Idealized model

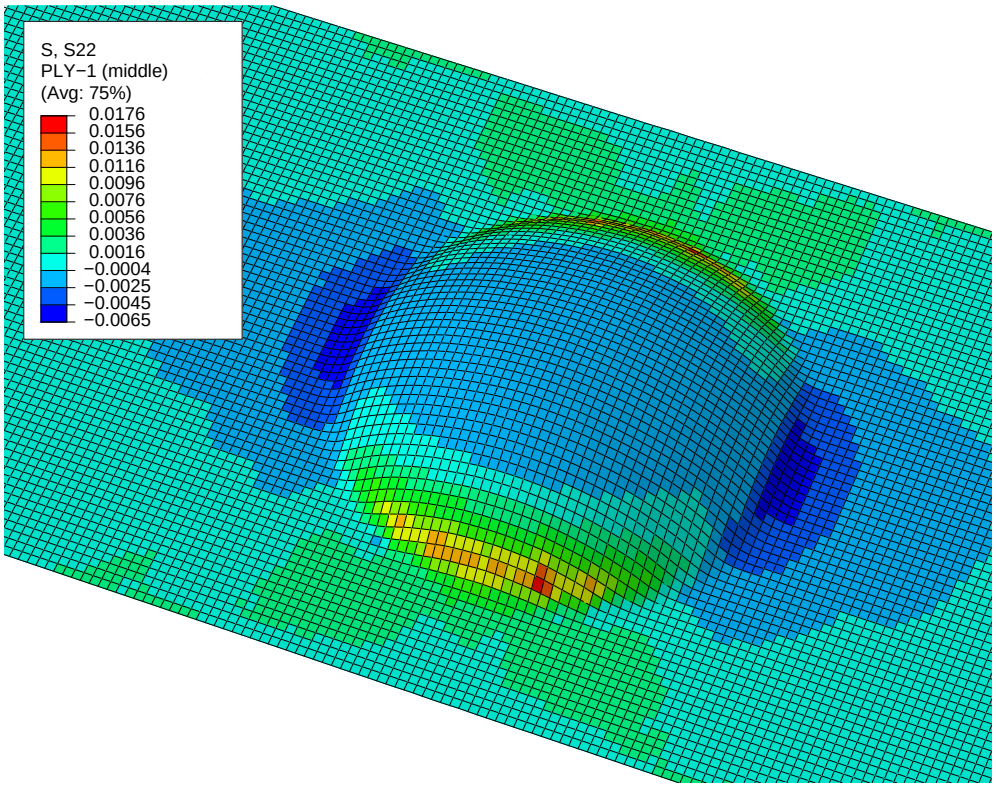


(b) As-formed model

Figure D.23: Transverse-stress in layer-1 ($[\pm 45^{\circ}_{2}]$ layup)

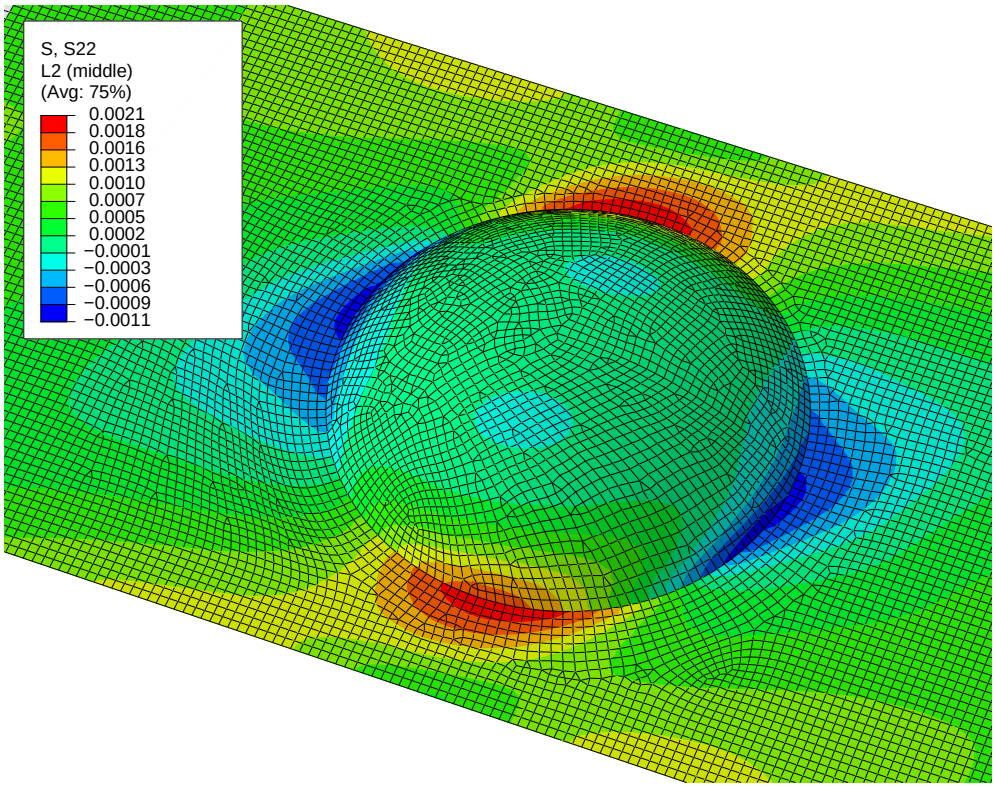


(a) Idealized model

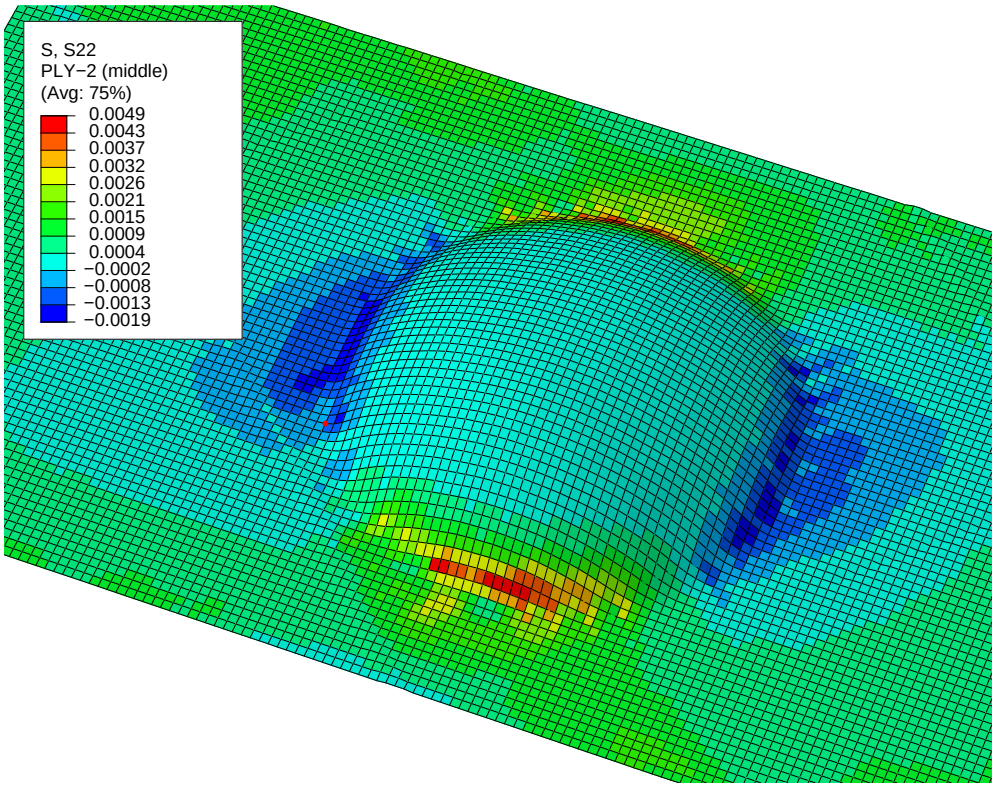


(b) As-formed model

Figure D.24: Transverse-stress in layer-1 ($[0/90]_2$ layup)

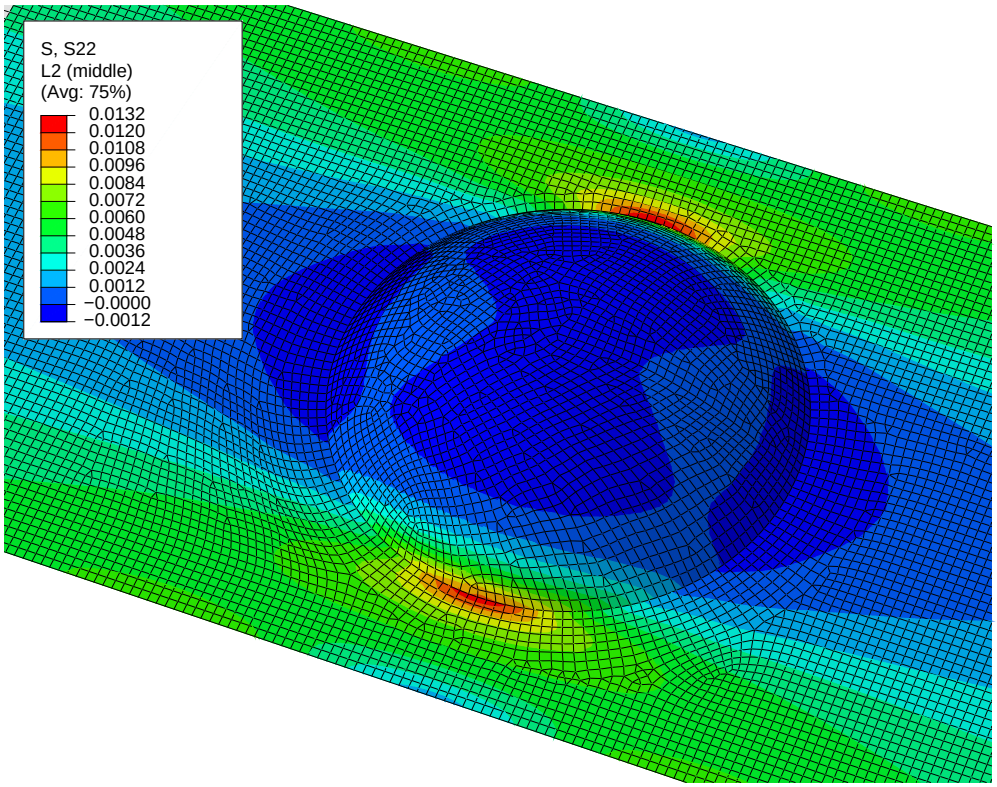


(a) Idealized model

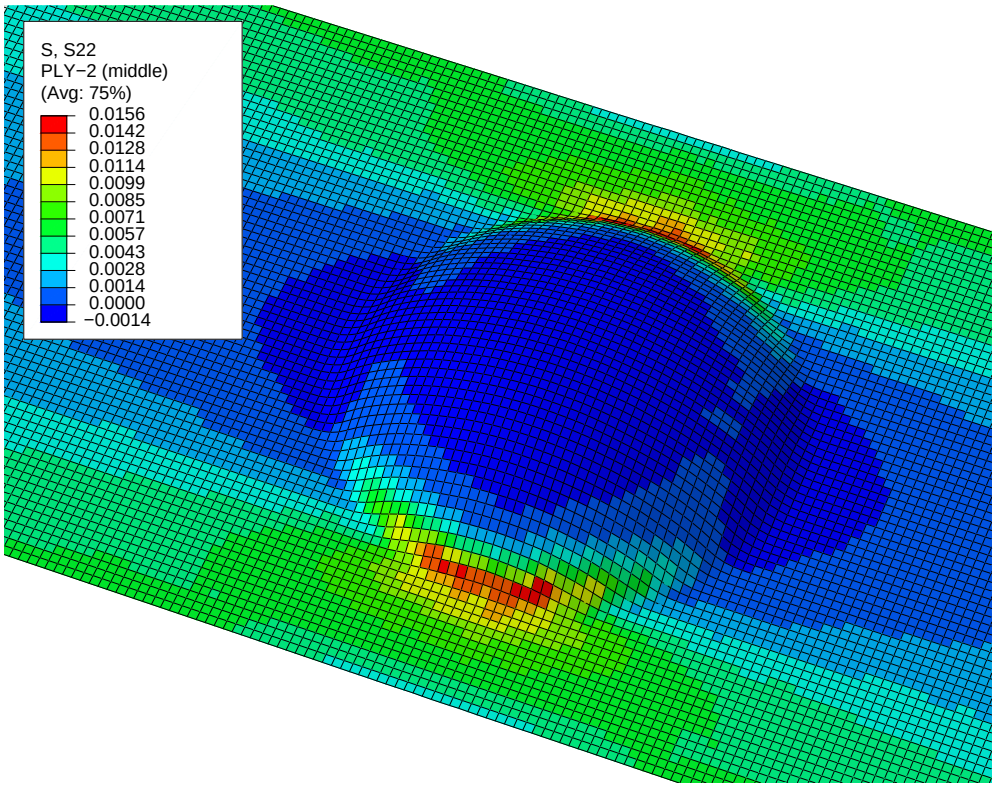


(b) As-formed model

Figure D.25: Transverse-stress in layer-2 ($[\pm 45^{\circ}_{2}]$ layup)

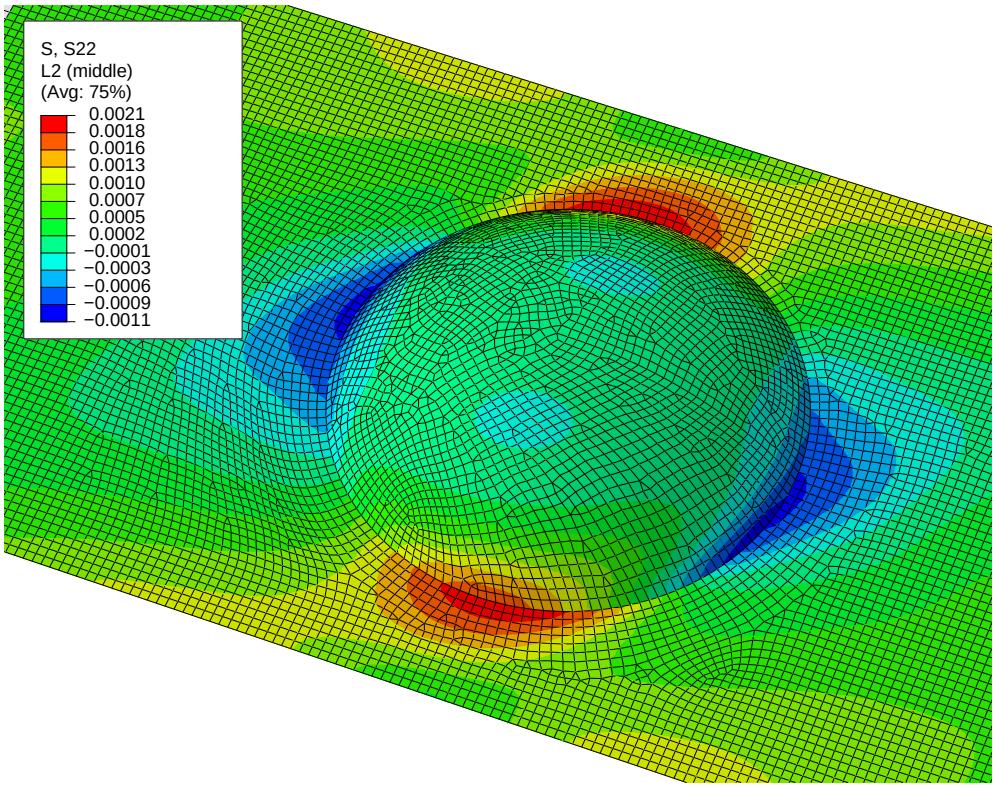


(a) Idealized model

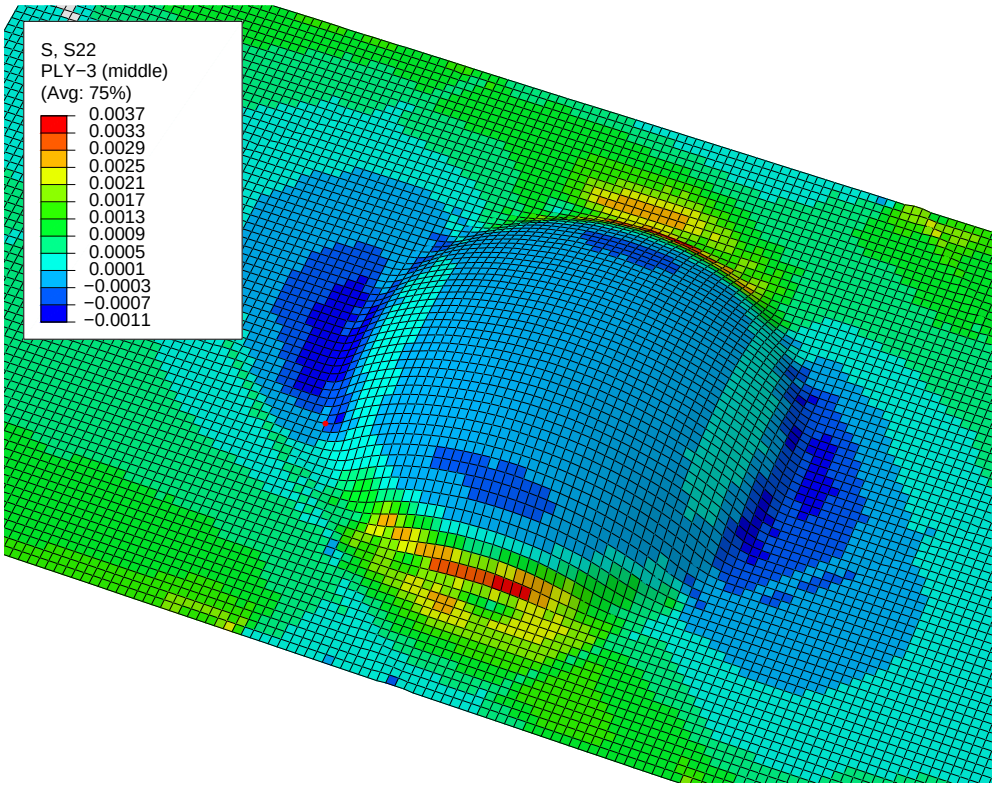


(b) As-formed model

Figure D.26: Transverse-stress in layer-2 ($[0/90]_2$ layup)

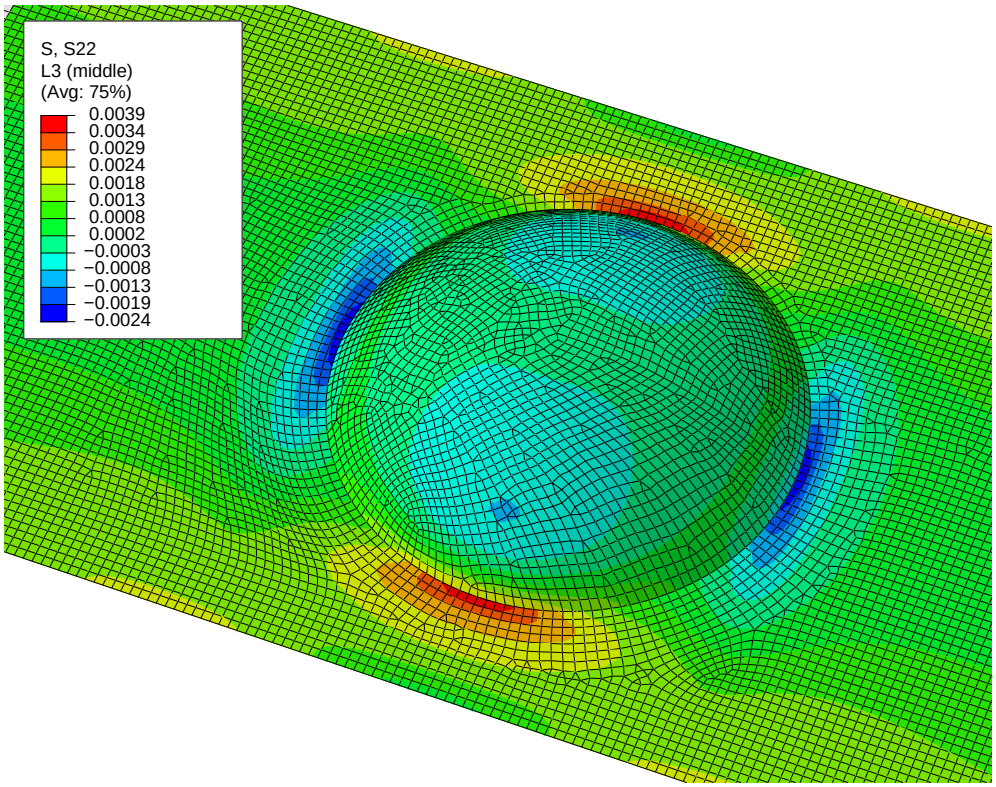


(a) Idealized model

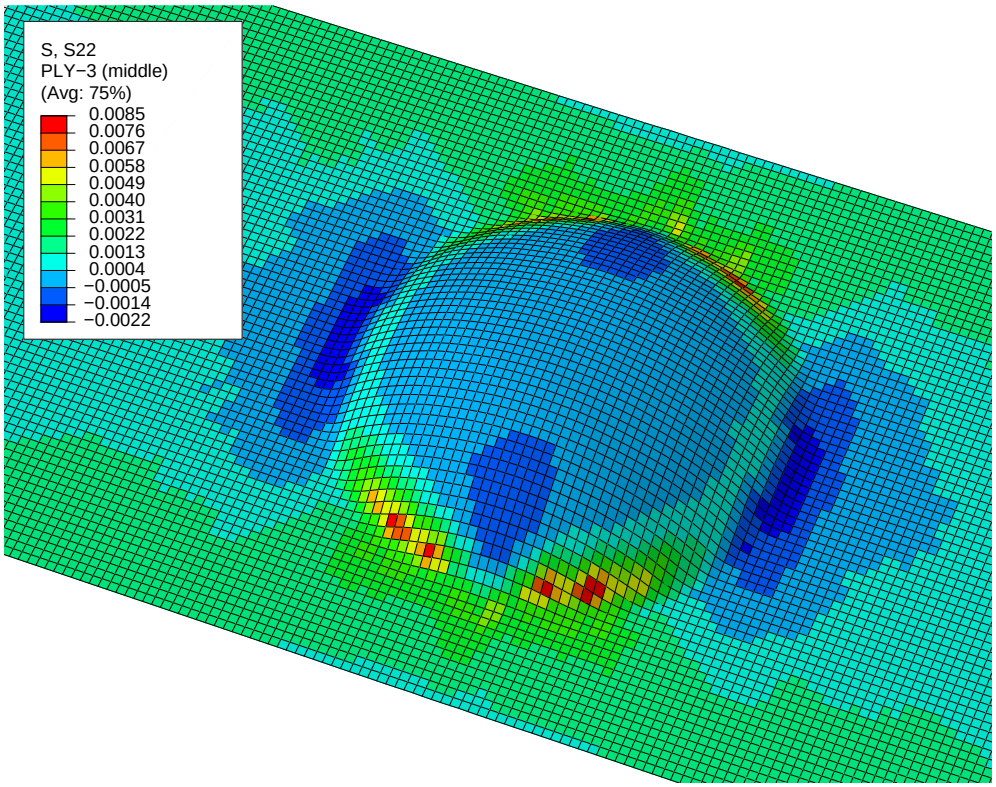


(b) As-formed model

Figure D.27: Transverse-stress in layer-3 ($[\pm 45^{\circ}_{2}]$ layup)

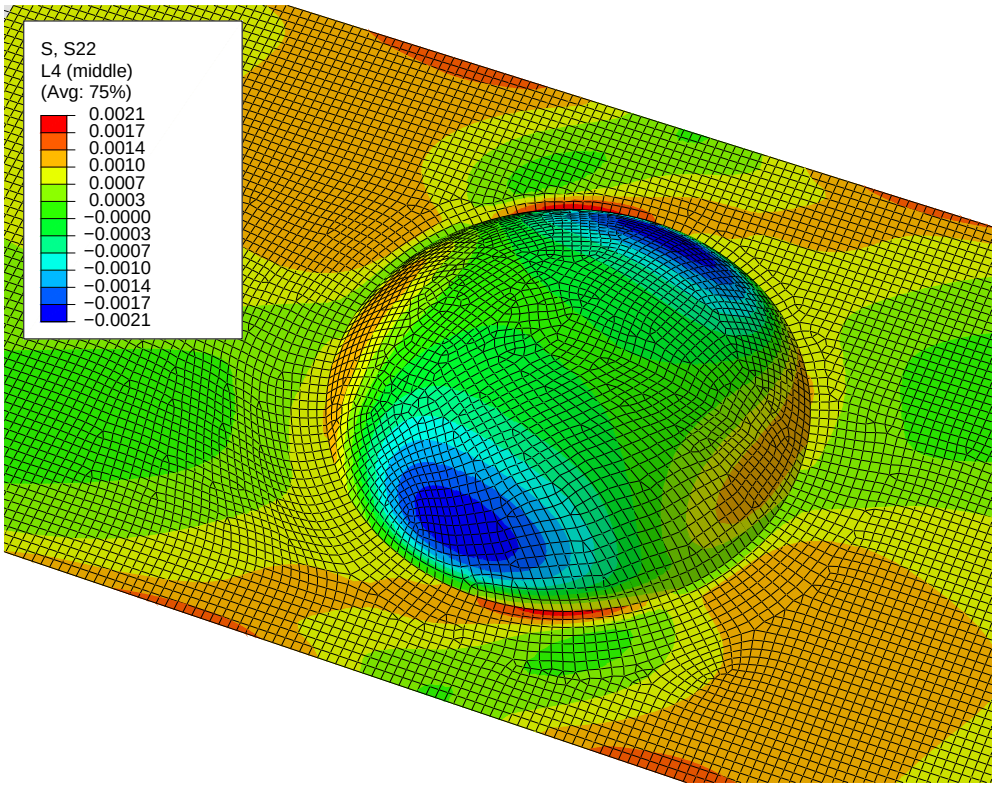


(a) Idealized model

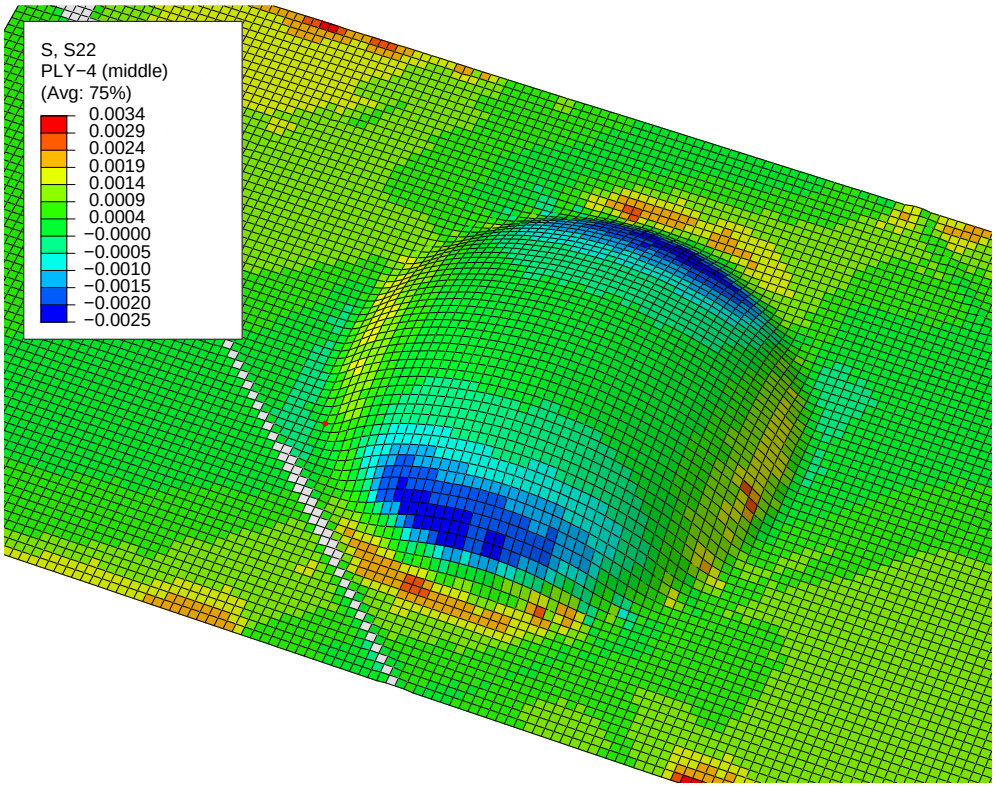


(b) As-formed model

Figure D.28: Transverse-stress in layer-3 ($[0/90]_2$ layup)

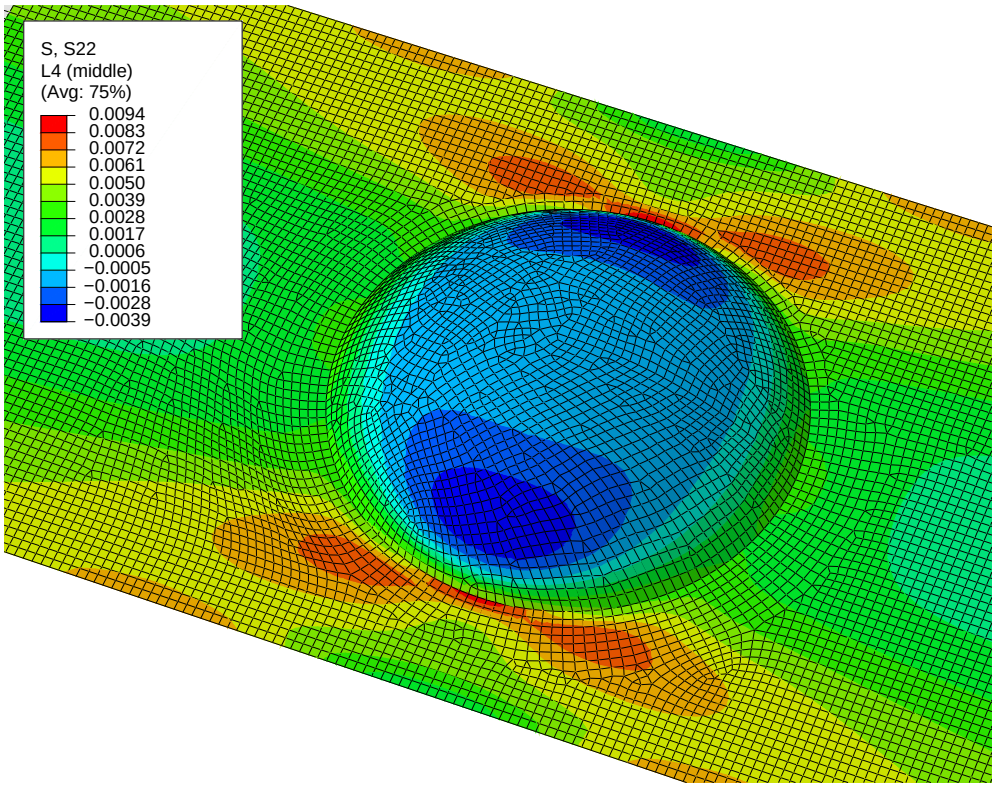


(a) Idealized model

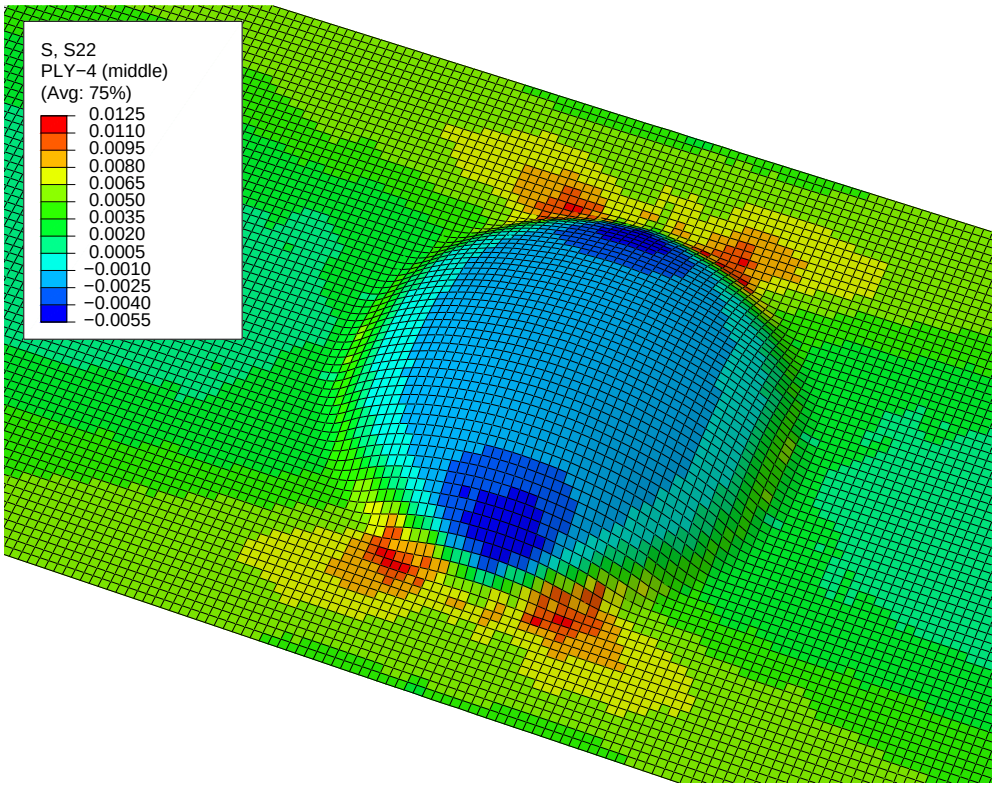


(b) As-formed model

Figure D.29: Transverse-stress in layer-4 ($[\pm 45^{\circ}_{2}]$ layup)

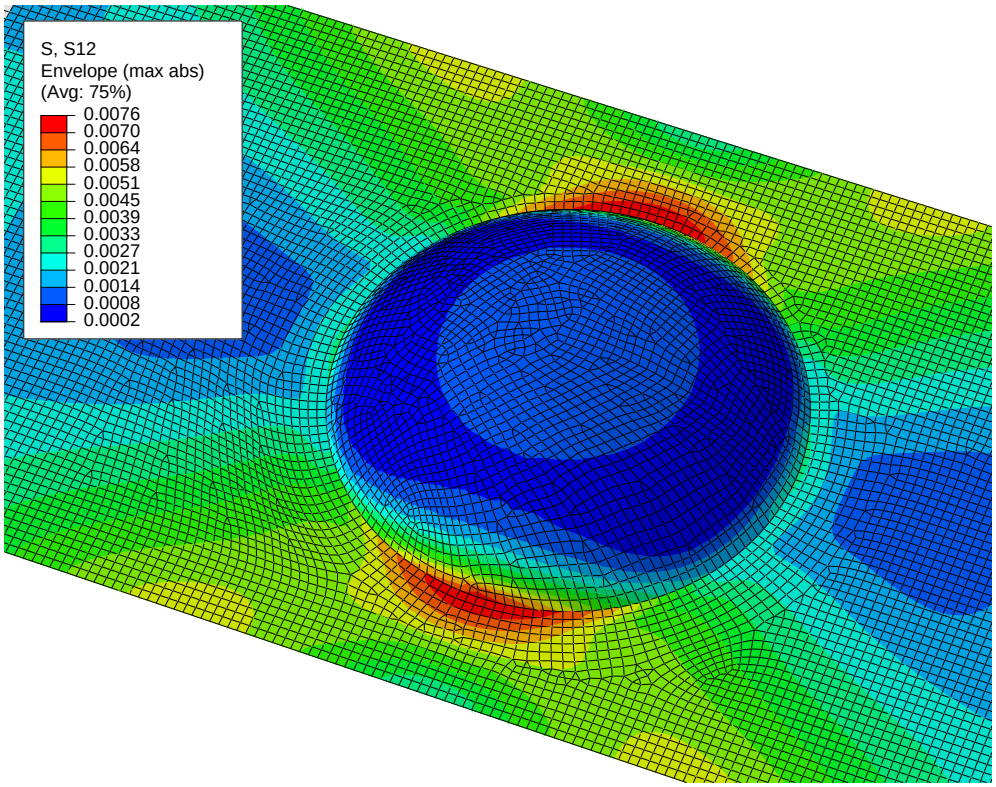


(a) Idealized model

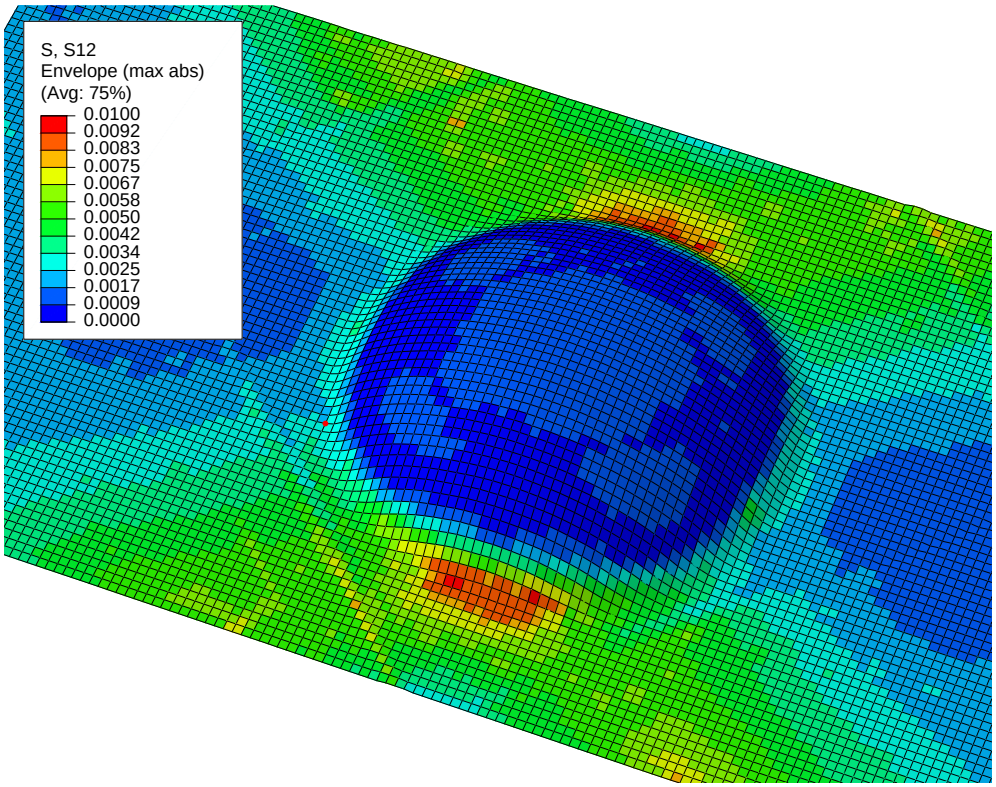


(b) As-formed model

Figure D.30: Transverse-stress in layer-4 ($[0/90]_2$ layup)

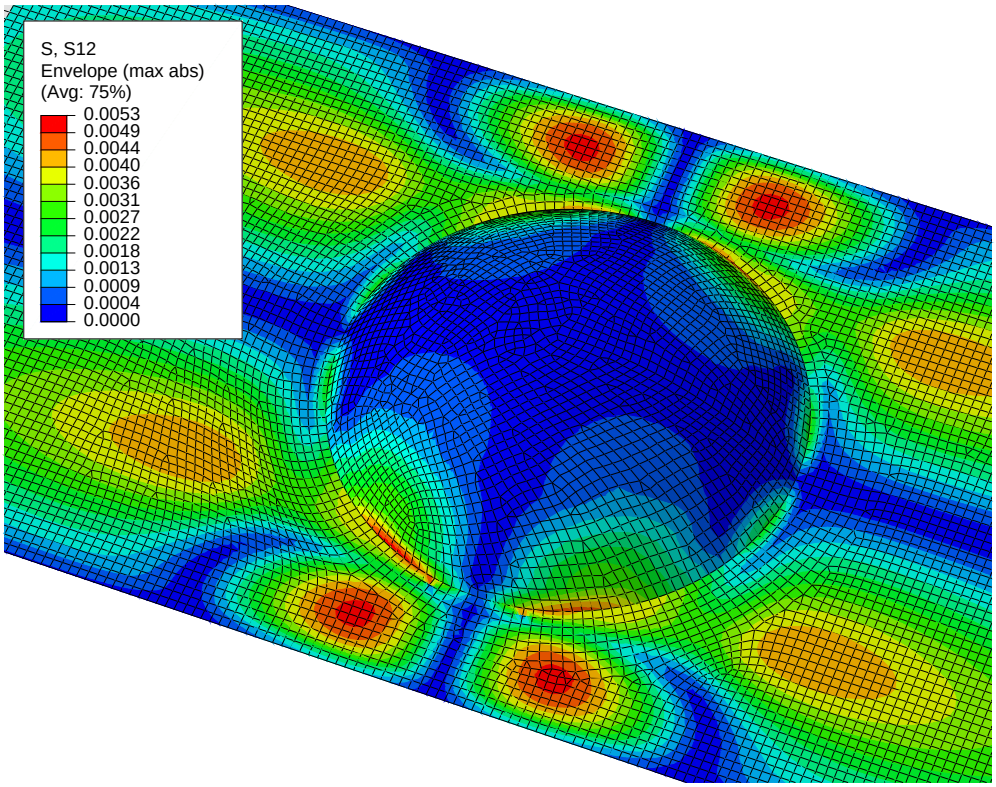


(a) Idealized model

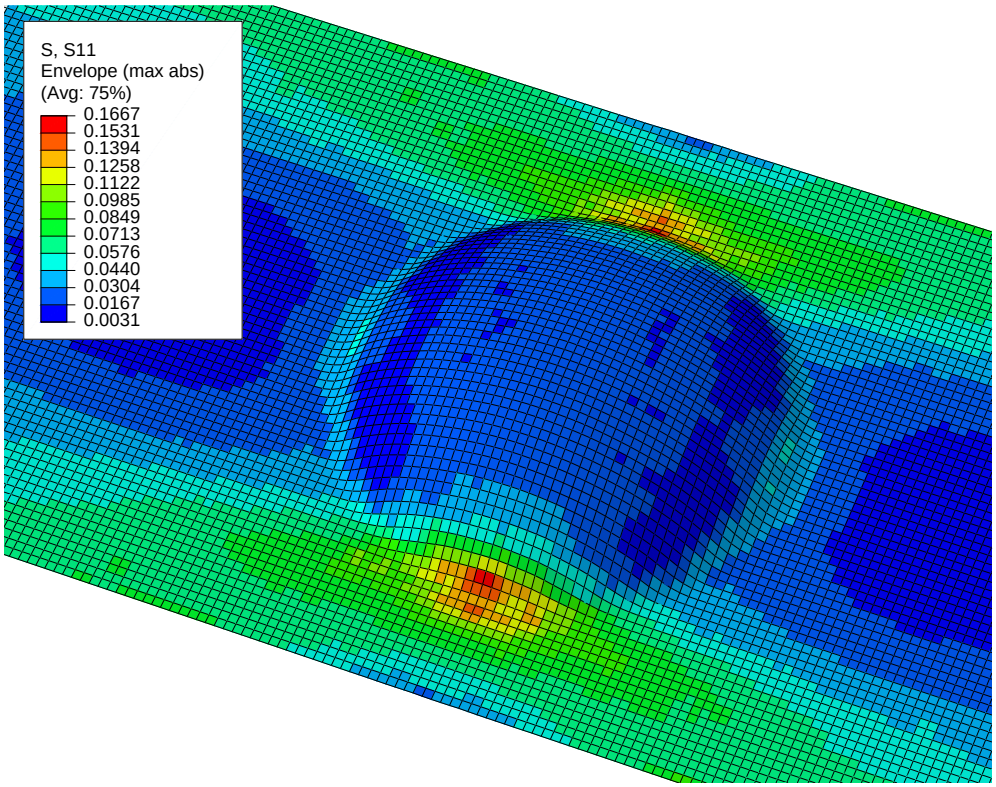


(b) As-formed model

Figure D.31: Shear-stress envelope ($[\pm 45^\circ_2]$ layup)

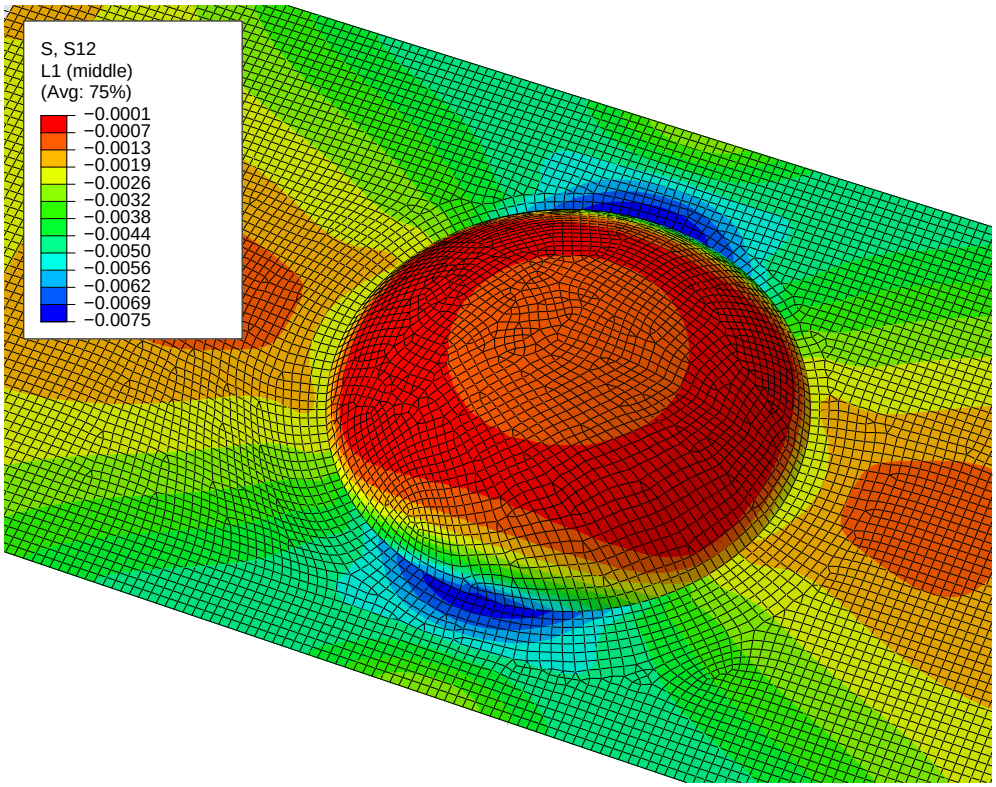


(a) Idealized model

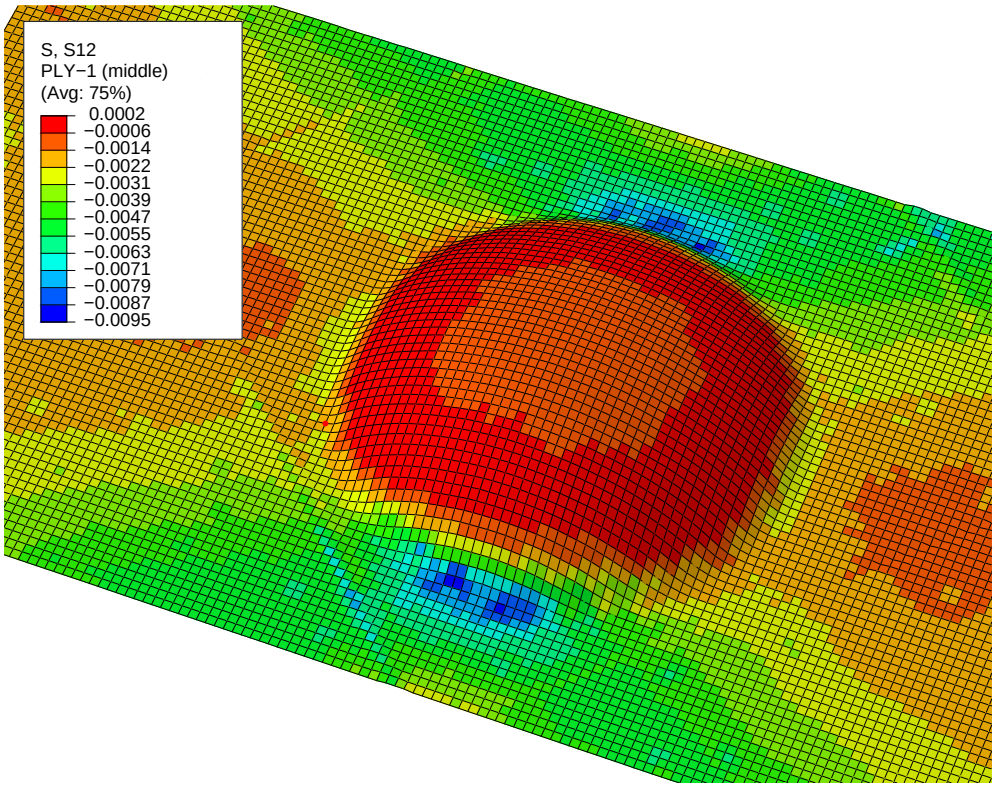


(b) As-formed model

Figure D.32: Shear-stress envelope ($[0/90]_2$ layup)

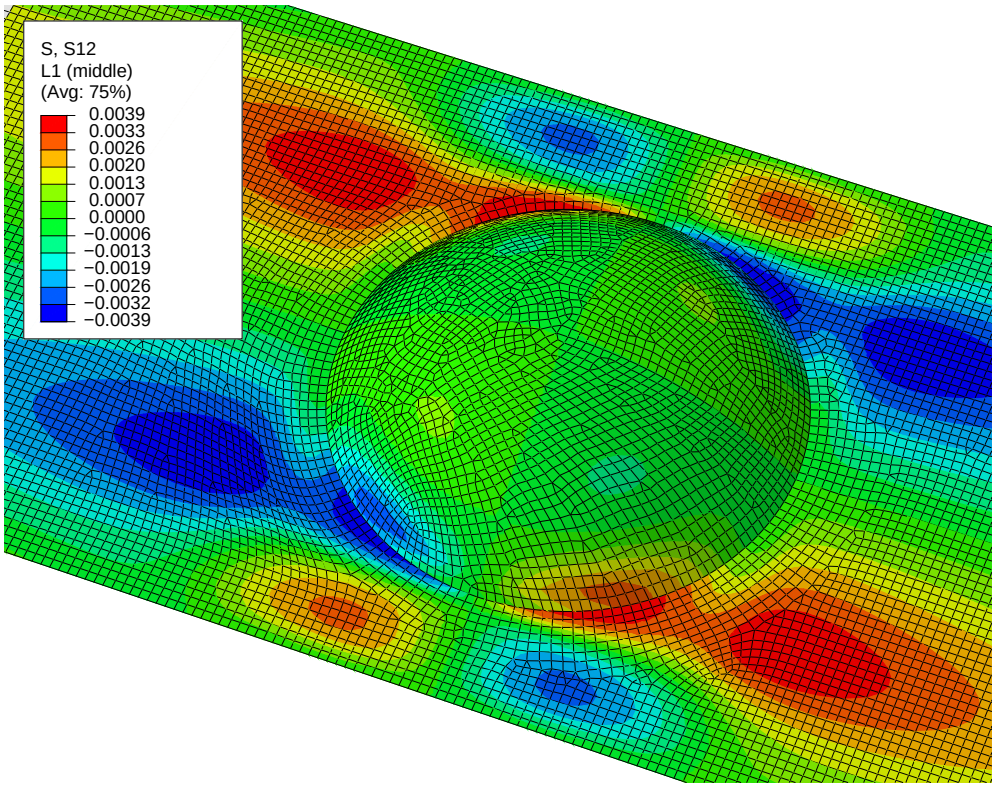


(a) Idealized model

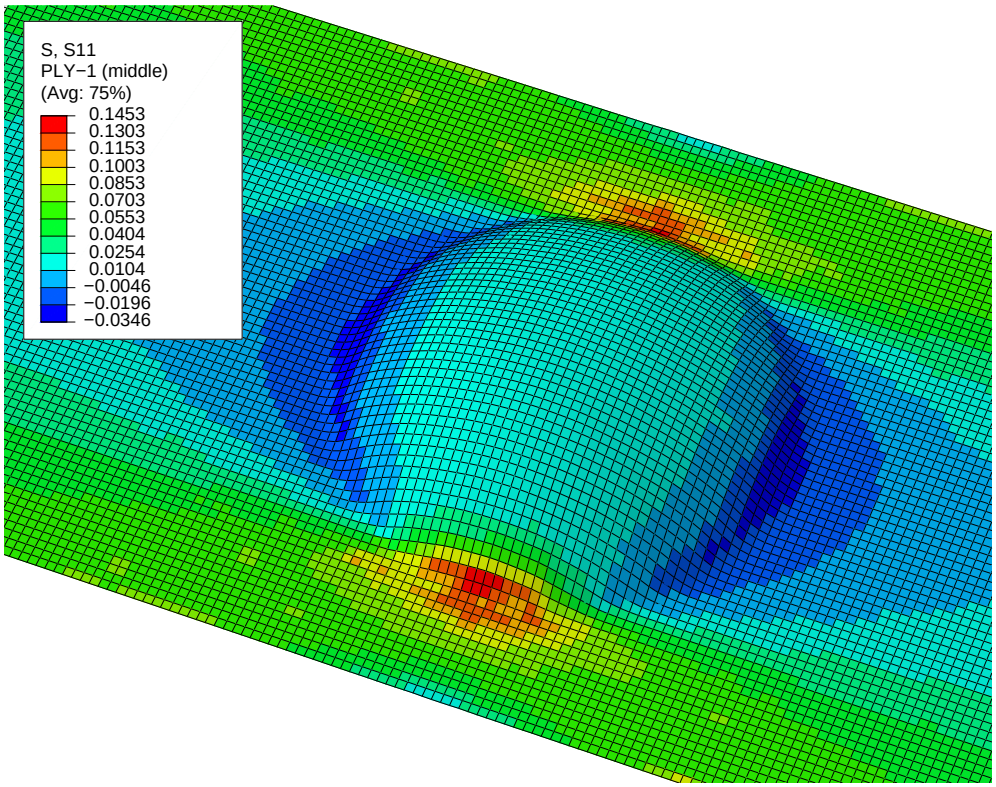


(b) As-formed model

Figure D.33: Shear-stress in layer-1 ($[\pm 45^\circ_2]$ layup)

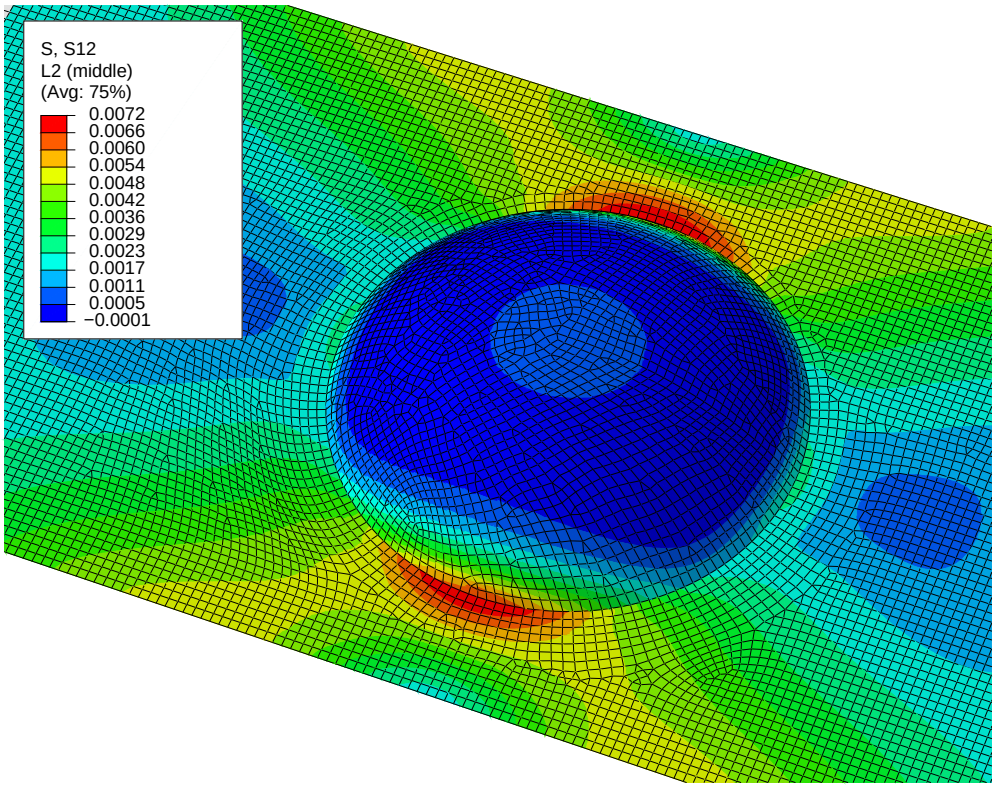


(a) Idealized model

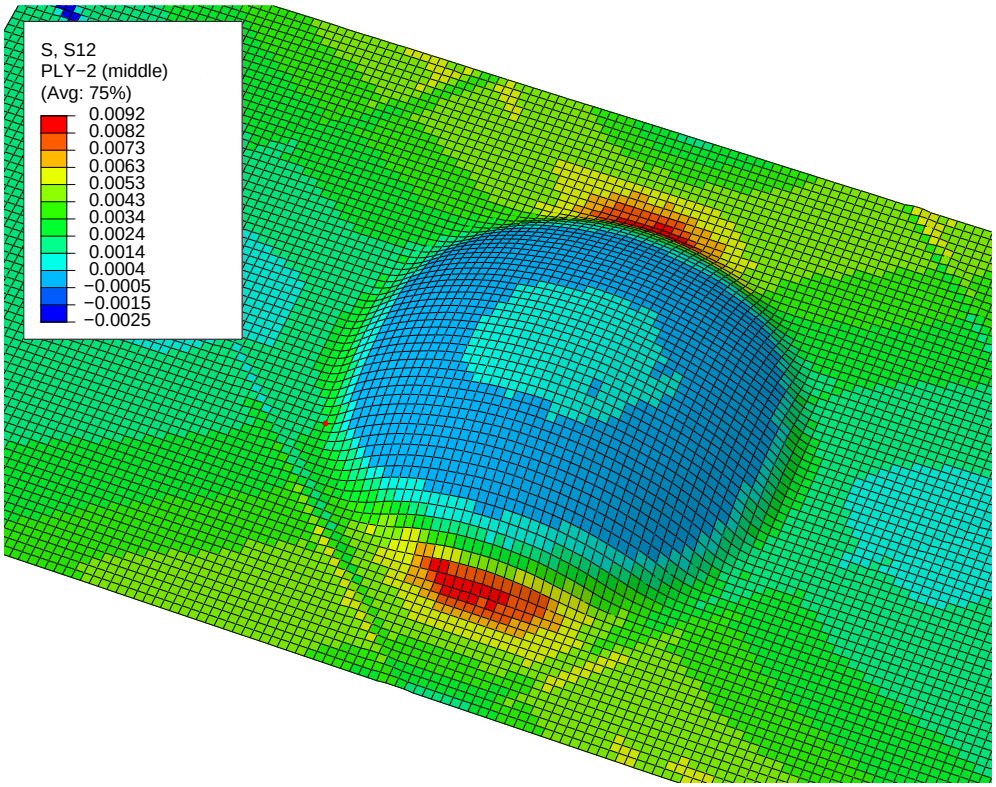


(b) As-formed model

Figure D.34: Shear-stress in layer-1 ($[0/90]_2$ layup)

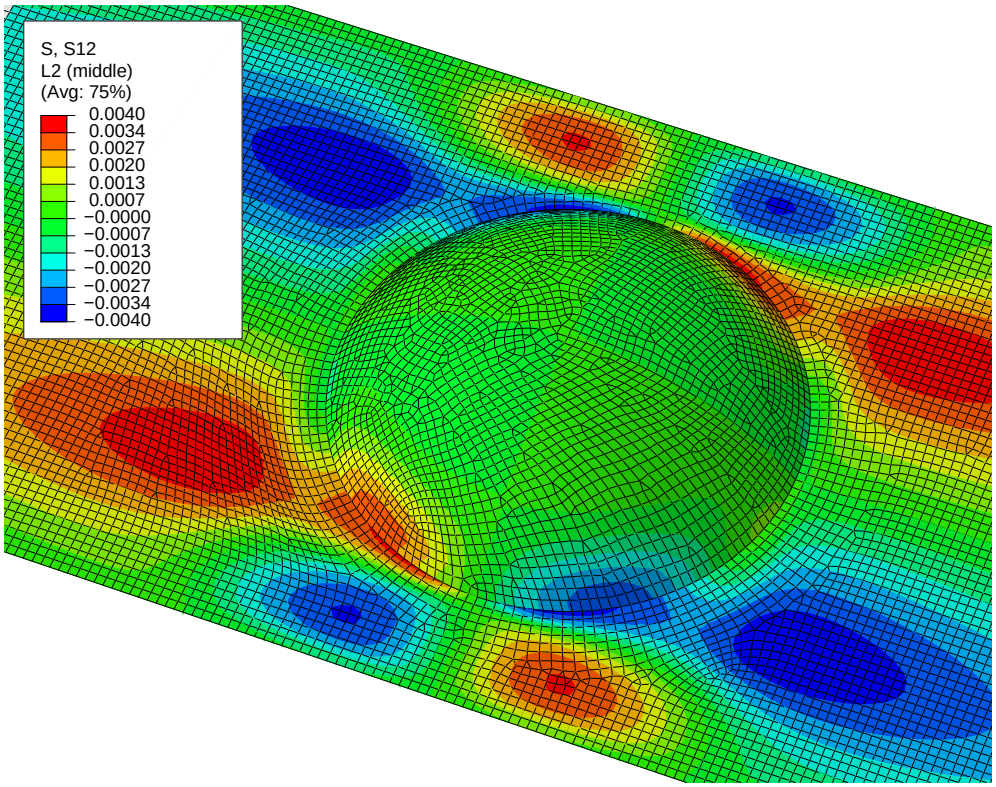


(a) Idealized model

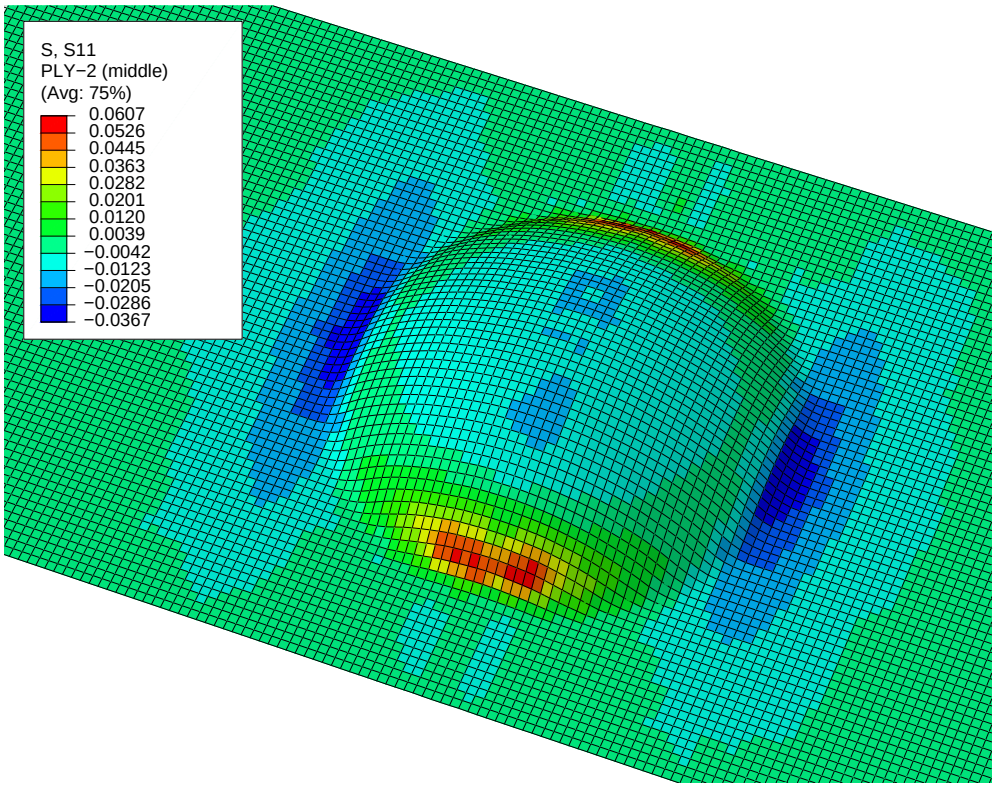


(b) As-formed model

Figure D.35: Shear-stress in layer-2 ($[\pm 45^{\circ}_2]$ layup)

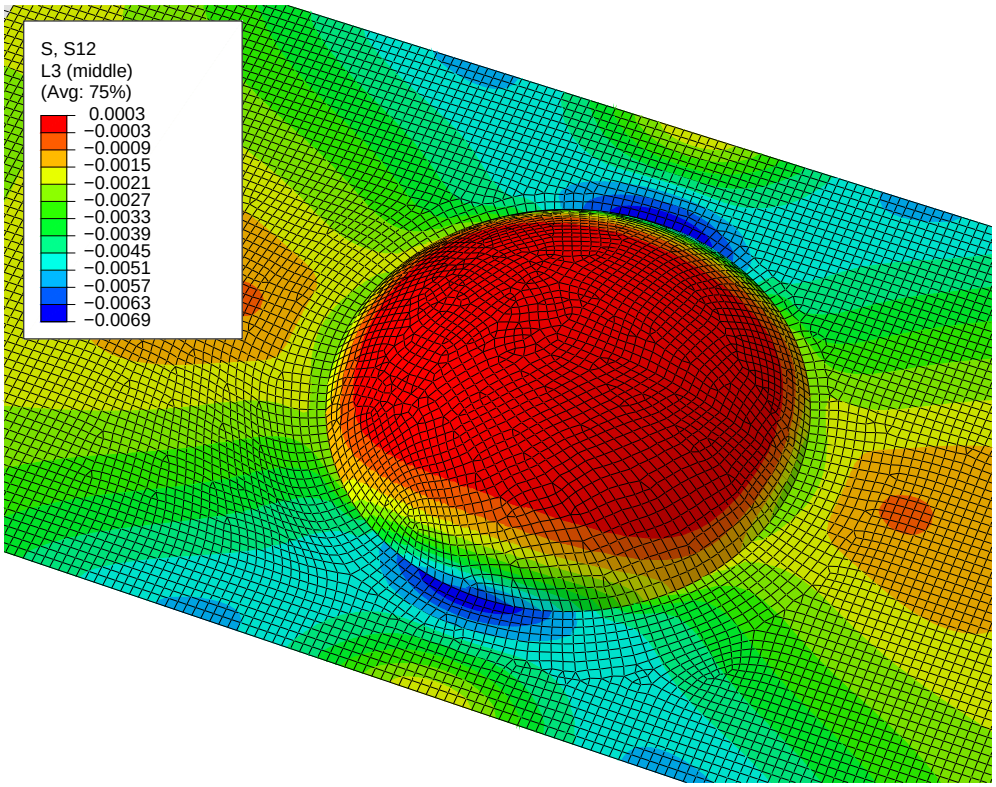


(a) Idealized model

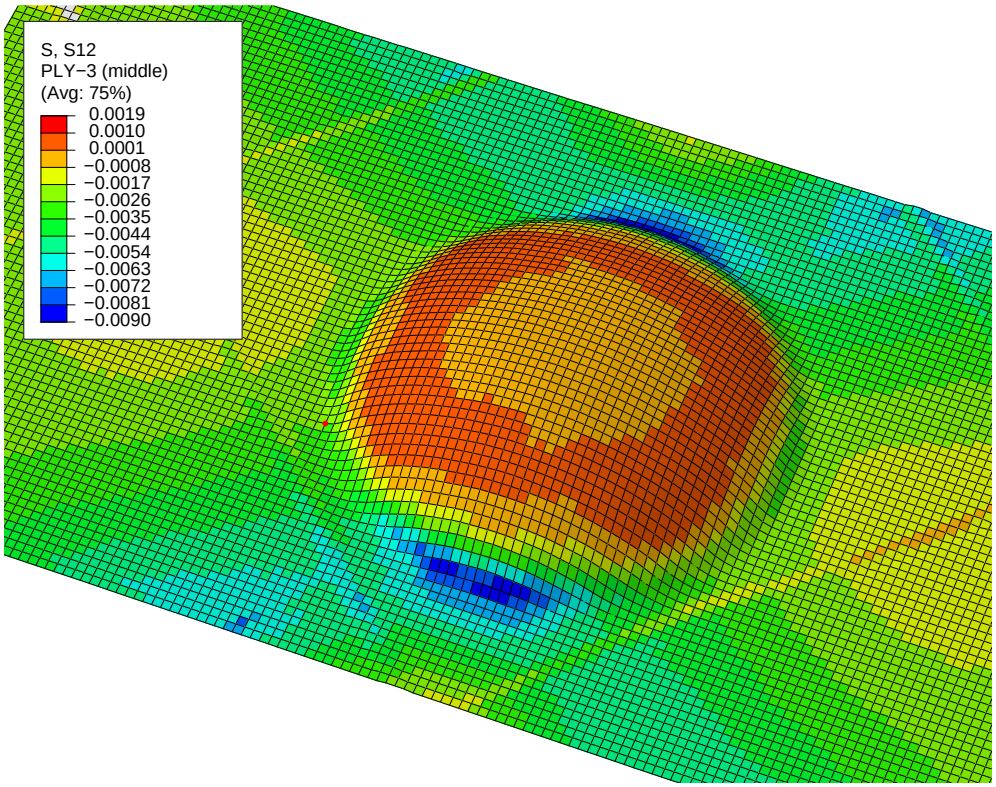


(b) As-formed model

Figure D.36: Shear-stress in layer-2 ($[0/90]_2$ layup)

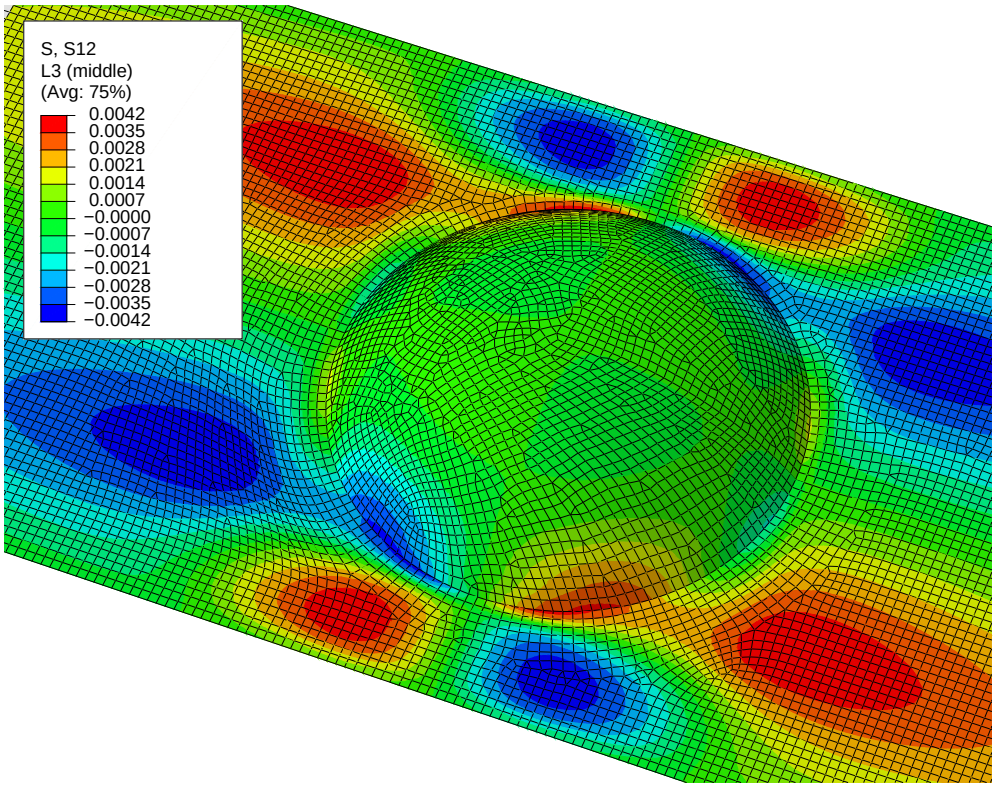


(a) Idealized model

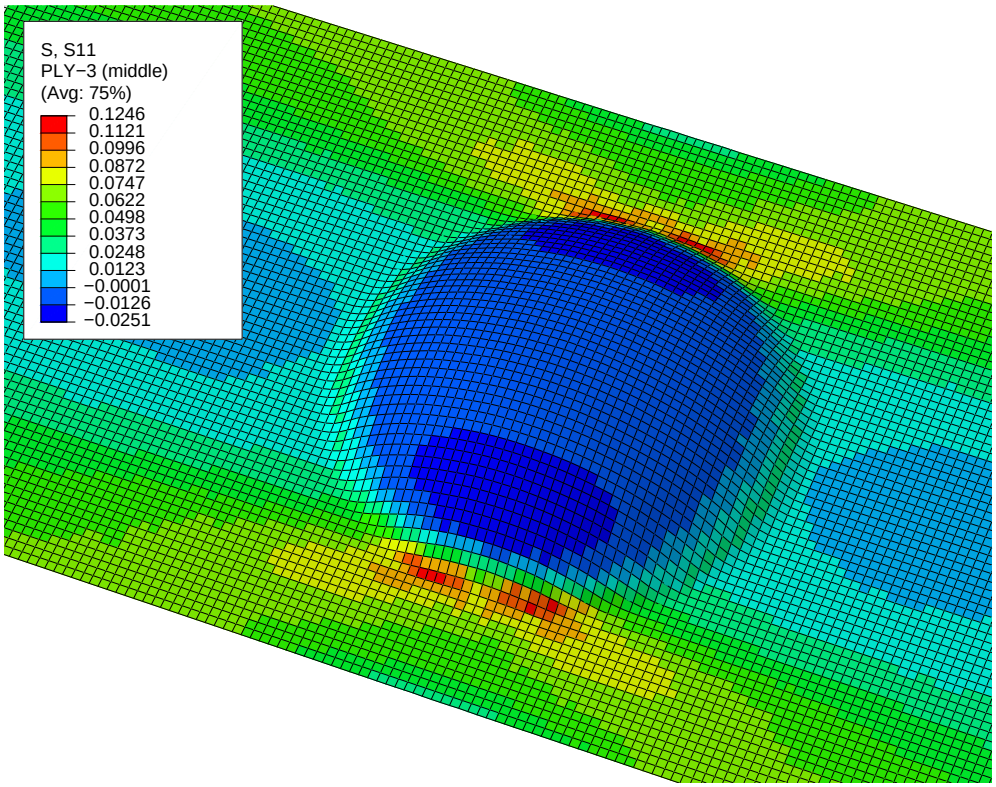


(b) As-formed model

Figure D.37: Shear-stress in layer-3 ($[\pm 45^\circ_2]$ layup)

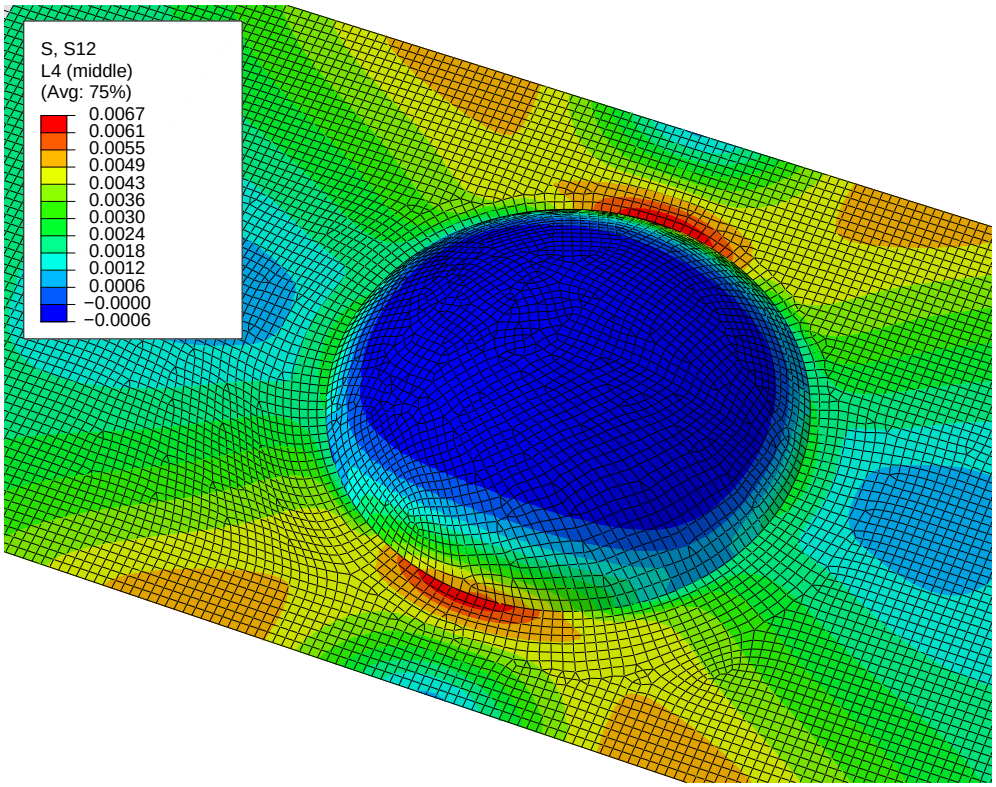


(a) Idealized model

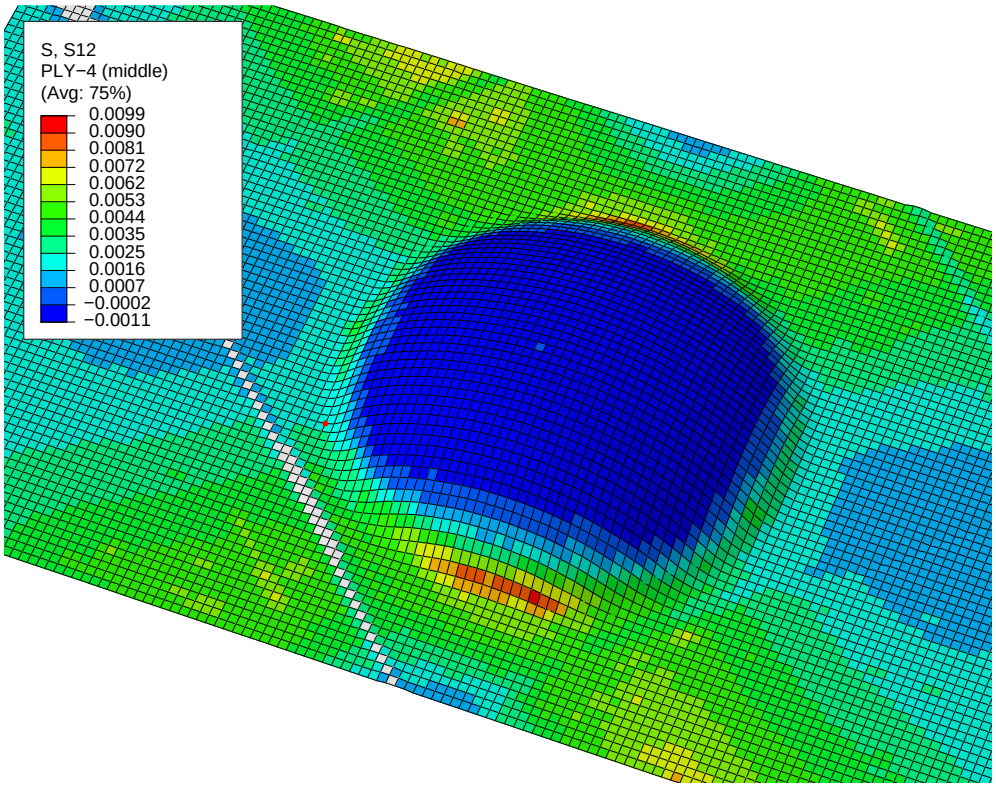


(b) As-formed model

Figure D.38: Shear-stress in layer-3 ($[0/90]_2$ layup)

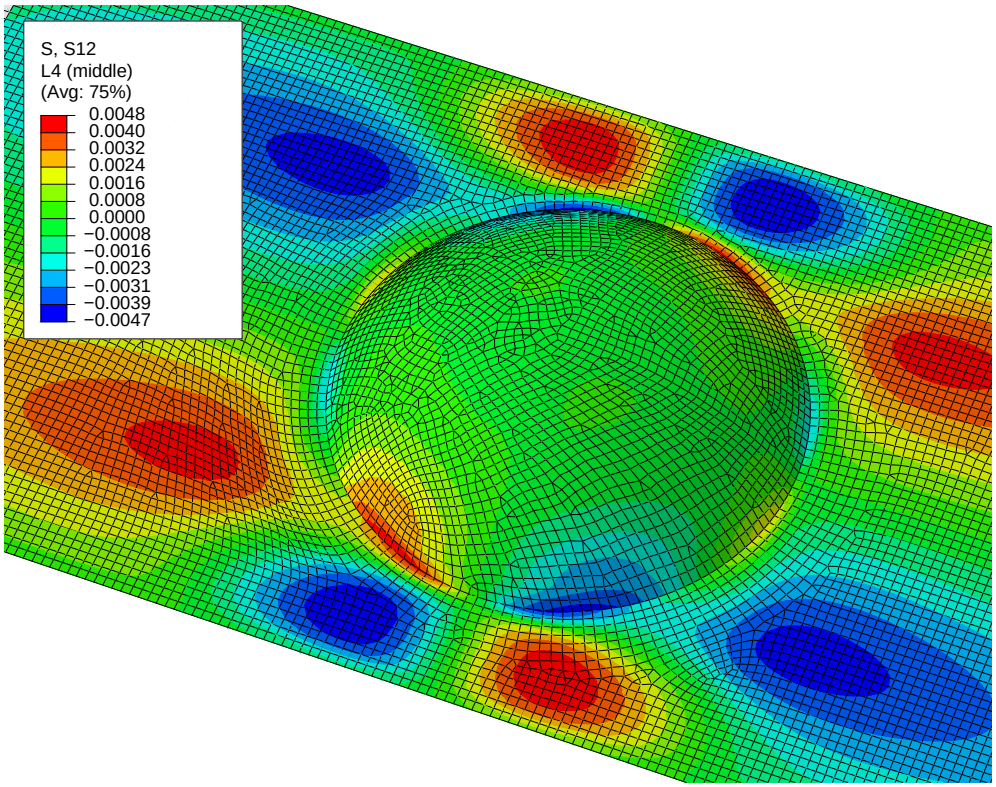


(a) Idealized model

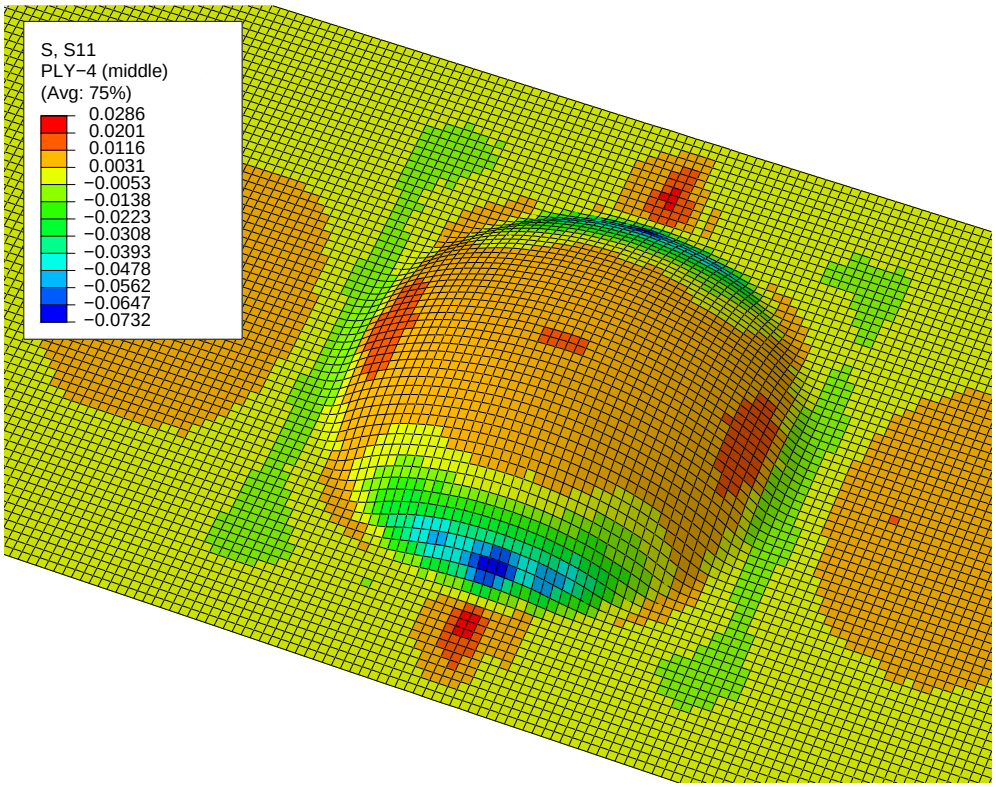


(b) As-formed model

Figure D.39: Shear-stress in layer-4 ($[\pm 45^\circ_2]$ layup)



(a) Idealized model



(b) As-formed model

Figure D.40: Shear-stress in layer-4 ($[0/90]_2$ layup)

APPENDIX E

PAM-FORM TO ABAQUS TRANSLATER

```
function [D,PamData]=PamTranslater(varargin)

% [D,PamData]=PamTranslater(options) gathers data from the output
% of a PAM-Form simulation, andtranslates it into an input file for Abaqus.
% The critical processes are all split out into subfunctions.
%
% D       -- is the structure containing all output and working variables
%
% PamData -- is the structure containing the data read directly from the
%              PAM-Form output file.
%
% Input variables take the form of name,value pairs. Options available are:
% 'TrimDef'  which takes the form of a nx2 array of points which define the
%            default mesh-trimming box. This is useful if you have a large
%            number of similar models, and don't want to enter the points
%            manually for each.
%
% 'Solver'   can be either 'quick' or 'best'. This defines which method is
%            used in plotting abaqus nodes with respect to PAM-Form nodes.
%            'best' uses a full nonlinear solver on the isoparametric
%            shape functions, whereas 'quick' uses a reduced quadratic
%            formulation. 'best' is more stable and accurate, but
%            computationally expensive. Only use when 'quick' provides
%            unsatisfactory results
%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Philip Bean 8/21/2018
```

Initial Preparation

```
if nargin>0
   O=InputParser(varargin);
end


thikdef=0.35;
VFdef=0.36;
% Default fiber volume fraction and thickness



Fiber=struct('E1',73,'E2',73,'G12',30,'G23',30,'nu12',0.23,'ro',2.54,...
    'F1t',3.45);
Matrix=struct('E',1.1,'nu',0.42,'ro',0.906,'Ft',0.0331,'Fc',0.0483);
% Material Component Properties



PamData=PAM_Read;
disp('PAM-Form File Read')
% This gathers the desired nodal and elemental information from the
% PAM-Form ERFH5 database file



[NewEl,NewNodes,Size]=flatmesh(PamData);
% Establishing the new mesh at its basic state containing no z-information
```

```
D=struct('NewEl',NewEl,'NewNod',NewNodes,'PamNod',PamData.AllNodes,...
    'PamEl',PamData.Elements,'Size',Size,'thikdef',thikdef,...
    'VFdef',VFdef,'Fiber',Fiber,'Matrix',Matrix,'TrimDef',O.TrimDef);
% Gathering mesh data into a structure array for easy handling
```

Calculating Translation

```
D=NodeChecker(D,O.Solver);
disp('Nodes Found')
% Finding the new nodes' positions with respect to the old mesh


D=ElmFix(D);
disp('Nodes Reorganized')
% Rearranging Edge-Case Elements such that all nodes fall within the
% old-mesh domain


D=Zpos(D);
disp('Nodes Positioned in Z')
% Using isoparametric shape functions, interpolate the z-position of the
% midsurface

D=ElmRem(D);
% Removing all unused nodes and elements

D=TrimGUI(D);
```

```
% Trimming the mesh



D=Fibdat(D,PamData);
disp('Architecture Data Converted')
% Collecting info on fiber directions, thicknesses,overlaps/gaps,
% and volume fractions



% There is a known error in which many elements reflect too few layers.
% This can be mitigated by running the slower nonlinear-solver based
% translater
if (D.L_percent>10)&&strcmp(O.Solver,'quick')
    Text=[num2str(D.L_percent),'% of elements show fewer than ',...
        num2str(length(PamData.Parts)),' layers. If this is unexpected',...
        ' behavior, the analysis may be redone using a more accurate',...
        ' solver. This solver is, however, much slower'];
    Q=questdlg(Text,'ErrorCheck','continue','redo','redo');
    % If more than 10% of elements show fewer layers than the number of
    % parts in the model, prompt the user to redo using the slower solver,
    % unless it has already been used
    switch Q
        case 'redo'
            D=struct('NewEl',NewEl,'NewNod',NewNodes,'PamNod',...
            PamData.AllNodes,'PamEl',PamData.Elements,'Size',Size,...
            'thikdef',thikdef,'VFdef',VFdef,'Fiber',Fiber,'Matrix',...
```

```
            Matrix,'TrimDef',O.TrimDef);
            D=resolver(D,PamData);

        case 'continue'
            disp('continuing')

        otherwise
            error('a selection must be made')

    end
else
    disp([num2str(round(D.L_percent,2)),'% of elements have fewer than ',...
    num2str(length(PamData.Parts)),' layers'])
    % If the slower solver is already in use, or if fewer than 10% of
    % elements show issue, simply display the percantage.
end



Abaqus_Input2(D)
% Writing the input file

end


function O=InputParser(V)
% This function PArses the input options for the translater

n=length(V);
O=struct('TrimDef',[],'Solver','quick');
```

```
try
    for i=1:2:n
        switch V{i}
            case 'TrimDef'
                O.TrimDef=V{i+1};
            case 'Solver'
                O.Solver=V{i+1};
            otherwise
                error(' ')
        end
    end
catch
    error('invalid input arguments')

end
end

function D=resolver(D,PamData)
% This function recalculates the translation using the slower solver if the
% quicker solver produces errors.
D=NodeChecker(D,'best');
D=ElmFix(D);
D=Zpos(D);
D=ElmRem(D);
D=Trim2(D);
D=Fibdat(D,PamData);
disp([num2str(round(D.L_percent,2)),'% of elements have fewer than ',...
```

num2str(length(PamData.Parts)),' layers'])

% Presenting the final layering comparison

end

Reading PAM-Form Database

function PamDat=PAM_Read

% This function reads the relevant data from the PAM-Form output database.

laststatenumpath=strcat('/post/multistate/TIMESERIES1/',...

'multientityresults/MODEL/TIME/ZONE1_set1/erfblock/indexident');

%datastructure path to the last-state number

partpath='/post/constant/attributes/PART/erfblock/title';

%datastructure path to the parts list

nodepaths={...

'/post/constant/entityresults/NODE/COORDINATE/ZONE1_set0/erfblock';...

...%datastructure path to initial node positions array

...%(must be suffixed with individual dataset)

'/entityresults/NODE/COORDINATE/ZONE1_set1/erfblock';...

...%datastructure path to refined node positions array

...%(must be prefixed with path to last state

...% and suffixed with individual dataset)

'/entityresults/NODE/Translational_Displacement/ZONE1_set1/erfblock'};

%datastructure path to all nodes displacements

%(must be prefixed with path to last state

```
nodesets={'/res','/entid'}; %node dataset suffixes

elmpaths={...
    '/post/constant/connectivities/SHELL/erfblock';...1
    ...%datastructure path to initial elements
    ...%(must be suffixed with individual dataset)
    '/connectivities/SHELL/erfblock';...2
    ...%datastructure path to refined elements
    ...%(must be prefixed with path to last state
    ...% and suffixed with individual dataset)
    '/entityresults/SHELL/Fiber 1 Direction/ZONE1_set1/erfblock';...3
    ...%datastructure path to fiber directions
    ...%(must be prefixed with path to last state
    ...% and suffixed with individual dataset
    '/entityresults/SHELL/Fiber 2 Direction/ZONE1_set1/erfblock';...4
    ...%datastructure path to transverse directions
    ...%(must be prefixed with path to last state
    ...% and suffixed with individual dataset)
    '/entityresults/SHELL/Fiber 1 Strain/ZONE1_set1/erfblock';...5
    '/entityresults/SHELL/Fiber 2 Strain/ZONE1_set1/erfblock';...6
    '/entityresults/SHELL/Shear Strain/ZONE1_set1/erfblock'};%7
%datastructure path to in-plane strain components
%(must be prefixed with path to last state
% and suffixed with individual dataset)

elmpaths_New={...
    '/entityresults/SHELL/Fiber_1_Direction/ZONE1_set1/erfblock';...
```

```
    '/entityresults/SHELL/Fiber_2_Direction/ZONE1_set1/erfblock';...
    '/entityresults/SHELL/Fiber_1_Strain/ZONE1_set1/erfblock';...
    '/entityresults/SHELL/Fiber_2_Strain/ZONE1_set1/erfblock';...
    '/entityresults/SHELL/Shear_Strain/ZONE1_set1/erfblock'};
% newer versions of PAM-Form (v2018) replace spaces with underscores in
% paths #3-7


elmsets={'/ic','/idele','/pid','/entid','/res'};%element dataset suffixes




[infile,inpath]=uigetfile('*.erfh5');%Selecting input file
if infile==0
    error('No File Selected')
end
fullfile=strcat(inpath,infile);
F_READ=@(X)h5read(fullfile,strcat(X));

statelist=F_READ(laststatenumpath);
laststatenum=num2str(statelist(end));

LSpath='/post/singlestate/state000000000000';
LSpath(end-length(laststatenum)+1:end)=laststatenum;
%finding final state number, and combining with the rest of the path
```

```
partslist=F_READ(partpath);

I_nodes=F_READ([nodepaths{1},nodesets{1}])';
I_nodnam=F_READ([nodepaths{1},nodesets{2}]);
%gathering initial nodal information

Ielmnam=F_READ([elmpaths{1},elmsets{2}]);
Ielmconn=F_READ([elmpaths{1},elmsets{1}])';
Ielmprt=F_READ([elmpaths{1},elmsets{3}]);
Ielm=[Ielmnam,Ielmconn,Ielmprt];
%Gathering initial Element information
nodes(I_nodnam,:)=I_nodes;
try
    R_nodes=F_READ([LSpath,nodepaths{2},nodesets{1}])';%Refined nodal info
    R_nodnam=F_READ([LSpath,nodepaths{2},nodesets{2}]);
    nodes(R_nodnam,:)=R_nodes;

    Felmnam=F_READ([LSpath,elmpaths{2},elmsets{2}]);
    Felmconn=F_READ([LSpath,elmpaths{2},elmsets{1}])';
    Felmprt=F_READ([LSpath,elmpaths{2},elmsets{3}]);
    Felm=[Felmnam,Felmconn,Felmprt];
    %Gathering refined element info

    elm=zeros(Felmnam(end),6);
    elm(Ielmnam,:)=Ielm;
    elm(Felmnam,:)=Felm;
```

```
    %Combining initial and refined element info
catch
    disp('Note: Mesh Refinement Inactive.')
    disp('If mesh is refined, then the refined-data paths are incorrect.')
    nodnam=I_nodnam;
    nodes(nodnam,:)=I_nodes;
    elm(Ielmnam,:)=Ielm;
    %assuming there is no refined data, the nodal and element data are only
    %that from the initial state.
end
nodenam2=F_READ([LSpath,nodepaths{3},nodesets{2}]);
nodedisp(nodenam2,:)=F_READ([LSpath,nodepaths{3},nodesets{1}])';
%Gathering Nodal Displacements for final positioning
nodespos=nodes+nodedisp;



[chosen,ok]=listdlg('ListString',partslist,'Name','Parts Selection',...
    'PromptString','Select Which Parts To Send');
if ok
    selectelm=elm(find(elm(:,6)==chosen(1)),:);
    Partrange=zeros(length(chosen),1);
    Partrange(1)=length(selectelm(:,1));
    for i=2:length(chosen)
        selectelm=[selectelm;elm(find(elm(:,6)==chosen(i)),:)];
        Partrange(i)=length(selectelm(:,1));
    end
```

```
else
    error('You must select a part')
end

usednodnam=unique([selectelm(:,2);selectelm(:,3);selectelm(:,4);...
    selectelm(:,5)]);
usednod=nodespos(usednodnam,:);
%only choosing nodes in the selected part(s) to return to the main code.
%This allows for the distinction between composite components and tools.
try

fibdirelms=F_READ([LSpath,elmpaths{3},elmsets{4}]);
%list of elements with known fiber directions
fibdirdat=F_READ([LSpath,elmpaths{3},elmsets{5}])';
%rows containing vector direction of the fiber-1 axis
transdirdat=F_READ([LSpath,elmpaths{4},elmsets{5}])';
%rows containing vector direction of the fiber-2 axis

Strainelms=F_READ([LSpath,elmpaths{5},elmsets{4}]);
%list of elements with known strains
S1=F_READ([LSpath,elmpaths{5},elmsets{5}])';%Fiber Strain
S2=F_READ([LSpath,elmpaths{6},elmsets{5}])';%Transverse Strain
S12=F_READ([LSpath,elmpaths{6},elmsets{5}])';%Shear Strain
catch Errors1
    try

elmpaths(3:end)=elmpaths_New;
```

```
fibdirelms=F_READ([LSpath,elmpaths{3},elmsets{4}]);
%list of elements with known fiber directions
fibdirdat=F_READ([LSpath,elmpaths{3},elmsets{5}])';
%rows containing vector direction of the fiber-1 axis
transdirdat=F_READ([LSpath,elmpaths{4},elmsets{5}])';
%rows containing vector direction of the fiber-2 axis

Strainelms=F_READ([LSpath,elmpaths{5},elmsets{4}]);
%list of elements with known strains
S1=F_READ([LSpath,elmpaths{5},elmsets{5}])';%Fiber Strain
S2=F_READ([LSpath,elmpaths{6},elmsets{5}])';%Transverse Strain
S12=F_READ([LSpath,elmpaths{6},elmsets{5}])';%Shear Strain
    catch
        error(Errors1)
    end
end

elorientraw=zeros(max(fibdirelms),6);
elorientraw(fibdirelms,:)=[fibdirdat,transdirdat];
elorient=elorientraw(selectelm(:,1),:);
%generating an array containing material orientation data for each element.
%Each row of this array represents a singl element, and takes the form:
%[fiber1-x,fiber1-y,fiber1-z,fiber2-x,fiber2-y,fiber2-z]


StrainRaw=zeros(max(Strainelms),3);
StrainRaw(Strainelms,:)=[S1,S2,S12];
```

```
Strain=StrainRaw(selectelm(:,1),:);
%generating an array containing the strain information for each element.
%Each row of this array represents a singl element, and takes the form:
%[Fiber 1 Strain, Fiber 2 Strain, Shear Strain]



PamDat=struct('Used_Node',usednod,'Elements',selectelm(:,2:5),...
    'Node_Numbers',usednodnam,'Element_Orientations',elorient,...
    'AllNodes',nodespos,'Parts',Partrange,'Strain',Strain);
% Collecting all the data into a structure array for easy handling
end
```

Generating Basic Mesh

```
function[NewEl,NewNodes,Size]=flatmesh(PamData)
% Thi function establishes a new square mesh over the region bounded by the
% limits of the existing Pam-Form Mesh



A=inputdlg({strcat(['Input Desired Element Size(mm). This should be',...
    'smaller than the PAM-Form Elements'])},'Element Sizer');
Size=str2double(A{1});
% Prompting user for a decision about the new element size in Abaqus mesh
% It is critical for the accuracy of the mesh that this be smaller than the
% typical size of the old PAM-Form elements

Xmin=min(PamData.Used_Node(:,1));
Xmax=max(PamData.Used_Node(:,1));
```

```
Ymin=min(PamData.Used_Node(:,2));
Ymax=max(PamData.Used_Node(:,2));
%These values represent the rectangualr bounding box for the PAM-Form mesh
%in the x-y plane.

% XV=Xmin-Size:Size:Xmax+Size;
% YV=Ymin-Size:Size:Ymax+Size;
% This is the x-y grid for the new nodes.
XL=0:-Size:(Xmin-Size);
XU=Size:Size:(Xmax+Size);
XV=(XL(end):Size:XU(end));

YL=0:-Size:(Ymin-Size);
YU=Size:Size:(Ymax+Size);
YV=(YL(end):Size:YU(end));
% the grid for new nodes centered at (0,0)

W=length(XV);
H=length(YV);
numb=W*H;
NewNodes=zeros(numb,3);
k=0;
for i=1:W
    for j=1:H
        k=k+1;
        NewNodes(k,:)=[XV(i),YV(j),NaN];
    end
```

```
end
% Establishing the matrix of nodal information. Each row in this represents
% the x-y-z position of the node. Note that the z-position is initialized
% as NaN, in order to easily distinguish those nodes whose z-positions are
% unknown.

numb=(H-1)*(W-1);
NewEl=zeros(numb,4);
k=0;
for i=1:W-1
    for j=1:H-1
        N1=(i-1)*H+j;
        N2=i*H+j;
        N3=i*H+j+1;
        N4=(i-1)*H+j+1;
        k=k+1;
        NewEl(k,:)=[N1,N2,N3,N4];
    end
end
% Establishing matrix of nodal connectivies. Each row contains the node
% number for the forur corners of the element
end
```

Locating Nodes Relative to PAM-Form Mesh

```
function Dat=NodeChecker(Dat,mode)
% This function determines which new nodes are within the old mesh. those
% which are have data stored on which elements they lie within, and
```

```
% the local coordinates within that element.
% Dat -- is the data structure of info found preiously
% Dat.NewEl   -- is the matrix of connectivities for the new elements
% Dat.NewNod  -- is the matrix of positions of the new nodes
% Dat.PamNod  -- is the matrix of positions of the old nodes
% Dat.PamEl   -- is the matrix of connectivities for the old elements
% Dat.In_Num  -- is the vector of the number of associated old elements for
%                each new node
% Dat.Interior-- is a logical vector of whether a new node is on the
%                interior (rather than edge or outside) of any old element
% Dat.Val     -- is a cell array of the position info of new nodes with
%                respect to old elements
% mode        -- choses which solver to use. Either 'quick' or 'best'
switch mode
    case 'quick'
        secondcheckfun=@(EX,EY,NX,NY)secondcheck2(EX,EY,NX,NY);
    case 'best'
        secondcheckfun=@(EX,EY,NX,NY)secondcheck(EX,EY,NX,NY);
    otherwise
        error(['Invalid Solver selection &s. solver must be: ',...
            '''quick'' or ''best'''],mode)
end
% this switches between two solver modes. the first is the more efficient
% method, however, there are instances where this method produces erroneous
% results. In those cases, the slower solver is used.

NNod=Dat.NewNod;
```

```
PamNod=Dat.PamNod;
PamEl=Dat.PamEl; %initializing local variables from data structure

tol=0.01; %this tolerance accounts for some numerical error in determining
% the local coordinates. Using the tolerance will ensure that even
% edge-nodes will be noticed
CH=firstcheck (PamNod,PamEl,NNod);
%using firstchck to narrow the nodes inspected closely


NE=length(PamEl(:,1));%Number of old elements
NumNod=length(NNod(:,1));%Number of new nodes
PamX=PamNod(:,1);
PamY=PamNod(:,2);% Vectors of x and y positions of old nodes
XV=PamX(PamEl);
YV=PamY(PamEl);
% Matrices of x and y positions of old nodes (each row is an element)
NNX=NNod(:,1);
NNY=NNod(:,2);% Vectors of x and y positions of new nodes


Dat.In_Num=zeros(NumNod,1);
Dat.Interior=zeros(NumNod,1);
Dat.Val=cell(NumNod,1);%initializing output data structures

for i=1:NE% sorting through the old elements
```

```
    InRange=find(CH(i,:)); %using the result of firstcheck to determine
                           %which new nodes are nearby the ith old element
    NI=length(InRange);%number of nearby nodes
    ElmX=XV(i,:);
    ElmY=YV(i,:);%Vectors of x and y positions of the nodes in the ith elem
    for j=1:NI %looking at each nearby node
        number=InRange(j);

        [z,h]=secondcheckfun(ElmX,ElmY,NNX(number),NNY(number));
        %Determining the positions of the nearby nodes in the ith element's
        %local coordinates
        if(abs(z)<=1+tol)&&(abs(h)<=1+tol)
            Dat.In_Num(number)=Dat.In_Num(number)+1;
            Dat.Val{number}.El{Dat.In_Num(number)}=[i,z,h];
            %if the node is in (or on the edge) the ith element, input the
            %element number and local coordinates in Dat.Val, increment the
            %number in Dat.In representing how many elements associated
            %with the node
            if 1-abs(z)>tol&&1-abs(h)>tol
                Dat.Interior(number)=1;
                %If the node is not sufficiently close to element edge,
                %flag it as interior. This will be useful for refining the
                %new mesh near the edges of the old one.
            end
        end
    end
end
```

```
end

function CH=firstcheck (PamNodes,pamel,Nnodes,varargin)
%This is the first approximation of which New Nodes are near enough to an
%Old element to warrant closer checking

if nargin==4
    tol=varargin{1};
elseif nargin==3
    tol=0;
else
    error('firstcheck cannot be called with more than 4 arguments')
end %The addition of the varible tol allows for the box to be increased by
%[tol] on all sides, thus accounting for element size mismatch, or oddly
%shaped elements

nx=PamNodes(:,1);
ny=PamNodes(:,2); %X and Y positions of  old nodes respectively
L1=min(nx(pamel),[],2)-tol;% vector of the min x-value of each old element
L2=max(nx(pamel),[],2)+tol;% vector of the max x-value of each old element
L3=min(ny(pamel),[],2)-tol;% vector of the min y-value of each old element
L4=max(ny(pamel),[],2)+tol;% vector of the max y-value of each old element
% these L values represent the cartesian bounding box for any given
% element. that is the box defined by the max and min positions in the x
% and y directions. This is an excellent first pass approximation to
% determine which nodes are worth checking in detail. This increases
```

```
% efficiency in the main code by quickly/easily ruling out all those nodes
% which are definitely too far away.

LE=length(pamel(:,1));
LN=length(Nnodes(:,1));
CH=sparse(LE,LN);
x=Nnodes(:,1);
y=Nnodes(:,2);% x and y positions of new nodes
parfor i=1:LN
    CH(:,i)=((L1<=x(i) & L2>=x(i))&...
        (L3<=y(i) & L4>=y(i))); %determining whether a new node is within
    % the cartesian bounding box of each element. CH then is a logical
    % matrix with columns representing new nodes, and rows representing
    % old elements
end
end

function [Z,H]=secondcheck(Nx,Ny,x,y)
%This uses a nonlinear solver along with the isoparametric shape functions
%of a quadrilateral element in order to determine the element local
%coordinates of a given new node with respect to a given old element.
% Nx -- is the vector of x positions of the nodes in the given old element
% Ny -- is the vector of y positions of the nodes in the given old element
% x  -- is the x position of the new node
% y  -- is the y position of the new node
Nx=double(Nx);
Ny=double(Ny);
```

```
    function F=zerofun(z)
        %This function takes input of local coordinates, and uses the shape
        %functions to interpolate for the global coordinates. By
        %subtracting this value from the true global coordinates, a
        %function is created whose zeros are on the correct local
        %coordinates
        SH=QSfun(z(1),z(2));% shape functions

        F=[x-(Nx*SH);...
            y-(Ny*SH)];
        %subtraction of estimated global coordinates from real ones
    end

    X=fsolve(@(Q)zerofun(Q),[0,0],optimoptions('fsolve','Display','none'));
    % calling matlab's nonlinear solver to find the zeros of the function.
    % the initial guess is the local origin.
    Z=X(1);
    H=X(2); %outputting the individual local coordinates
end

function [Z,H]=secondcheck2(Nx,Ny,x,y)
%This uses a reduced form of the isoparametric shape functions in the form
%of a quadratic equation, which can be solved to determine the element
%local coordinates of a given new node with respect to a given old element.
% Nx -- is the vector of x positions of the nodes in the given old element
% Ny -- is the vector of y positions of the nodes in the given old element
% x  -- is the x position of the new node
```

```
% y  -- is the y position of the new node

V=[1,1,1,1;-1,1,1,-1;1,1,-1,-1;-1,1,-1,1];
X=V*transpose(Nx);
Y=V*transpose(Ny);
A=X(2)*Y(4)-X(4)*Y(2);
B=4*(y*X(4)-x*Y(4))+(X(2)*Y(3)-X(3)*Y(2))+(X(1)*Y(4)-X(4)*Y(1));
C=4*(y*X(3)-x*Y(3))+(X(1)*Y(3)-X(3)*Y(1));

z1=(-B+sqrt(B^2-4*A*C))/(2*A);

z2=(-B-sqrt(B^2-4*A*C))/(2*A);

zv=[z1,z2];
r1=isreal(z1);
r2=isreal(z2);

if r1+r2==2
    h1=(Y(1)-4*y+z1*Y(2))/(Y(3)+z1*Y(4));
    h2=(Y(1)-4*y+z2*Y(2))/(Y(3)+z2*Y(4));

    hv=[h1,h2];
    [~,i]=min([norm([z1,h1]),norm([z2,h2])]);
    Z=zv(i);
    H=hv(i);
elseif r1
    h1=(Y(1)-4*y+z1*Y(2))/(Y(3)+z1*Y(4));
```

```
    Z=z1;
    H=h1;
elseif r2
    h2=(Y(1)-4*y+z2*Y(2))/(Y(3)+z2*Y(4));
    Z=z2;
    H=h2;
else
    Z=inf;
    H=inf;
end
end


function Shape=QSfun(z,h)
%Isoparametric Shape functions for quadrilateral elements
% z -- local coordinate z
% h -- local coordinate h
Shape=1/4*[(1-z)*(1-h);...
          (1+z)*(1-h);...
          (1+z)*(1+h);...
          (1-z)*(1+h)];%vector of 4 shape functions
end
```

Finalizing Mesh

```
function Dat=ElmFix(Dat)
% This Function identifies elements containing nodes both inside and
% outside the old mesh (edge cases). These edge cases are then repaired by
% moving the external nodes to the edge of the nearest old element. The
```

```
% movement is defined as being along the vector directly toward the Old
% element's center (in local nonlinear coordinates)
% Dat -- is the data structure of info found preiously
% Dat.NewEl   -- is the matrix of connectivities for the new elements
% Dat.NewNod  -- is the matrix of positions of the new nodes
% Dat.PamNod  -- is the matrix of positions of the old nodes
% Dat.PamEl   -- is the matrix of connectivities for the old elements
% Dat.In_Num  -- is the vector of the number of associate old elements for
%                each new node
% Dat.Interior-- is a logical vector of whether a new node is on the
%                interior (rather than edge or outside) of any old element
% Dat.Val     -- is a cell array of the position info of new nodes with
%                respect to old elements

NEL=Dat.NewEl;
NN=Dat.NewNod;
Pn=Dat.PamNod;
Pe=Dat.PamEl;
InN=Dat.In_Num;
Intr=Dat.Interior; %Intializing local variables from the data structure

Out=(InN(NEL(:,1))<1|InN(NEL(:,2))<1)|(InN(NEL(:,3))<1|InN(NEL(:,4))<1);
%Determining which new elements (NEL) have nodes outside the old mesh
In=(Intr(NEL(:,1))>0|Intr(NEL(:,2))>0)|(Intr(NEL(:,3))>0|Intr(NEL(:,4))>0);
%Determining which new elements have nodes inside the old mesh
Cross=Out&In;%those elements which have nodes both inside and
%              outside the old mesh (edge cases)
```

```
Crossed=find(Cross);%vector of edge element numbers
CRNodes=NEL(Crossed,:);% matrix of nodes in edge elements
%(each element is a row)

CRNums=InN(CRNodes);%matrix of the number of old elements associated
% with each edge element node
CRNodel=cell(1,length(NN(:,1)));% Empty cell array for each node.
% cell will become a matrix of nearby old elements (those associated with
% other nodes in the same new element

for i=1:length(Crossed)
    a=find(CRNums(i,:));
    %vector of nodes inside old elements within the ith edge element
    b=find(~CRNums(i,:));
    %vector of nodes outside old elements within the ith edge element

    List=zeros(1000,1);%preallocating List
    l=0;
    for j=1:(length(a))%cycling through the nodes in the ith edge element
        %which are within old elements
        A=Dat.Val{CRNodes(i,a(j))}.El;
        %nodal data for the jth interior node from the ith edge element
        for k=1:length(A)
            l=l+1;
            List(l)=A{k}(1);%becomes a vector of old elements within which
            %there are nodes from the ith edge element
        end
```

```
    end
    List=List(1:l); %taking only nonzero values of List
    for k=1:length(b)

        CRNodel{CRNodes(i,b(k))}=[CRNodel{CRNodes(i,b(k))};List];
        %filling the cell array for nodes with the vector of associated
        %old elements
    end
end
MovNod=unique(CRNodes(find(~CRNums)));%Vector of nodes to be moved
% These are the nodes outside the old mesh, but a part of edge elements in
% the new mesh

PamX=Pn(:,1);%vector of x-positions of the old nodes
PamY=Pn(:,2);%vector of y-positions of the old nodes
XV=PamX(Pe);%matrix of x-positions (each row represents an old element)
YV=PamY(Pe);%matrix of y-positions (each row represents an old element)
for i=1:length(MovNod)
    ChekEl=unique(CRNodel{MovNod(i)});%the old elements near the ith MovNod
    NumChekEl=length(ChekEl);%number of elements near ith MovNod
    elval=zeros(NumChekEl,4);%preallocated matrix for distances to each
    %nearby old-element
    x=NN(MovNod(i),1);
    y=NN(MovNod(i),2); %position data for the ith MovNod
    for j=1:NumChekEl
        offsets=[XV(ChekEl(j),:)-x;YV(ChekEl(j),:)-y];
```

```
    dist=zeros(1,4);
    for k=1:4
        dist(k)=norm(offsets(:,k));
    end
    dists=sort(dist);
    %finding the magnitude of each offset vector
    elval(j,:)=dists;%finding the the distance to
    %nodes in nearby elements



end
[~,B]=sortrows(elval);%finding which element the node is closest to
b=B(1);
[z,h]=secondcheck2(XV(ChekEl(b),:),YV(ChekEl(b),:),x,y);
%finding the local coordinates associated with the nearest element

Z=1/(max(abs([z,h])))*[z,h];
%shrinking the element so that its largest dimension falls on the
%element edge

OldXVals=XV(ChekEl(b),:);
OldYVals=YV(ChekEl(b),:);%x and y values of the old element in question
SH=QSfun(Z(1),Z(2));
xy=[OldXVals*SH,OldYVals*SH];
%using the shape functions to find the new global coordinate of the node
NN(MovNod(i),1:2)=xy;
Dat.NewNod=NN;
```

```
    Dat.In_Num(MovNod(i))=1;
    Dat.Val{MovNod(i)}.El{1}=[ChekEl(b),Z(1),Z(2)];
    %including new data in the data arrays
end


end

function Dat=Zpos(Dat)
% This Function identifies elements containing nodes both inside and
% outside the old mesh (edge cases). These edge cases are then repaired by
% moving the external nodes to the edge of the nearest old element. The
% movement is defined as being along the vector directly toward the Old
% element's center (in local nonlinear coordinates)
% Dat -- is the data structure of info found preiously
% Dat.NewNod  -- is the matrix of positions of the new nodes
% Dat.PamNod  -- is the matrix of positions of the old nodes
% Dat.PamEl   -- is the matrix of connectivities for the old elements
% Dat.In_Num  -- is the vector of the number of associated old elements
%                for each new element
% Dat.Val     -- is a cell array of the position info of new nodes with
%                respect to old elements

In_Num=Dat.In_Num;
Val=Dat.Val;
NN=Dat.NewNod;
PE=Dat.PamEl;
PN=Dat.PamNod;%Initializing local variables from data structure
```

```
OZV=PN(:,3);%vector of z postions of old nodes
OZelm=OZV(PE);
%matrix of z positions of corners of old elements
%each row represents an element


GoodNodes=find(In_Num);%finding which nodes are assciated with the old mesh
ZDat=cell(length(GoodNodes),1);
for i=1:length(GoodNodes)
    NodeNum=GoodNodes(i);
    ElmNum=In_Num(NodeNum);
    %number of old elements associated with a new node
    Z_and_Num=zeros(ElmNum,2);
    %empty vector of z positions of node w.r.t. each old element
    for j=1:ElmNum
        ElPos=Val{NodeNum}.El{j};% reading element number and local
        %coordinates with respect to old element
        SH=QSfun(ElPos(2),ElPos(3));
        Z_and_Num(j,:)=[OZelm(ElPos(1),:)*SH,ElPos(1)];%using the shape
        %functions to interpolate the z position of this node w.r.t. the
        %j'th associated old element
    end
    zvec=Z_and_Num(:,1);
    NN(NodeNum,3)=mean([max(zvec),min(zvec)]);
    %setting the z position of the node to the midsurface
    %of those calculated from affiliated old elements. defined as the
```

```
    %average of the extrema.
    ZDat{i}=Z_and_Num;
end
Dat.ZDat=ZDat; %Saving the positions w.r.t. each old element in order to
%use later in finding thicknesses
Dat.NewNod=NN;
end

function Dat=ElmRem(Dat)
% This function removes all unused nodes and elements

NNRaw=Dat.NewNod;
NelRaw=Dat.NewEl;
InN=Dat.In_Num;
nodlist=find(InN);%Using the logical vector InN to find the indices of
%those nodes which fall within the bounds of the new mesh

NNClean=NNRaw(nodlist,:);
%Selecting only thosnodes determined to be within the bounds

InEl=(InN(NelRaw(:,1))>0&InN(NelRaw(:,2))>0)&(InN(NelRaw(:,3))>0&...
    InN(NelRaw(:,4))>0);%finding elements within the bounds
NelShrunk=NelRaw(find(InEl),:);%selecting only those elements

NelClean=zeros(size(NelShrunk));
for i=1:length(NelShrunk(:,1))
    for j=1:4
```

```
        NelClean(i,j)=find(nodlist==NelShrunk(i,j));
    end
end
%Translating the node numbers in the selected elements in order to reflect
%the new shorter node matrix.

Dat.NewNod=NNClean;
Dat.NewEl=NelClean;
Dat.Val=Dat.Val(nodlist);%Updating the data structure

Dat=rmfield(Dat,{'In_Num','Interior'});
end
```

Generating Properties

```
function [D,thick]=Fibdat(D,PamDat)
thikdef=D.thikdef;
VFdef=D.VFdef;

XV=D.PamNod(:,1);
YV=D.PamNod(:,2);
PamElX=XV(D.PamEl);
PamElY=YV(D.PamEl);
numelsPAM=length(D.PamEl(:,1));
PamPoly=cell(numelsPAM,1);
parfor i=1:numelsPAM
    PamPoly{i}=polyshape(PamElX(i,:),PamElY(i,:))
end
```

```
%Gathering node and element information in order to generate a polyshape
%object to represent each of the PAM-Form elements


XV=D.NewNod(:,1);
YV=D.NewNod(:,2);
NewElX=XV(D.NewEl);
NewElY=YV(D.NewEl);
numelsNEW=length(D.NewEl(:,1));
NewPoly=cell(numelsNEW,1);
parfor i=1:numelsNEW
    NewPoly{i}=polyshape(NewElX(i,:),NewElY(i,:))
end
%Gathering node and element information in order to generate a polyshape
%object to represent each of the Abaqus elements

Parts=PamDat.Parts;
PartsZ=[0;Parts];
numparts=length(Parts);
A=zeros(numelsNEW,1);
Aover=cell(numelsNEW,1);
CH=firstcheck(D.PamNod,D.PamEl,D.NewNod,D.Size/2);
parfor i=1:numelsNEW
    AoverI=sparse(numelsPAM,numparts);
    A(i)=area(NewPoly{i});%area of i'th new element
    CHI=find(sum(CH(:,D.NewEl(i,:)),2));
    %determining the list of old elements that firstcheck determines to be
```

```
    %near to the i'th new element
    for k=1:length(Parts)
        List=CHI(find((CHI>PartsZ(k)).*(CHI<=PartsZ(k+1))));
        %listing which old elements within the k'th part that firstcheck
        %recommends checking for the i'th new element
        for q=1:length(List)
            j=List(q);
            AoverI(j,k)=area(intersect(NewPoly{i},PamPoly{j}));
            %area of overlap between i'th new element and j'th old element
        end
    end
    Aover{i}=AoverI;
end% These nested loops gather information about the areas occupied by
%each new element, along with information regarding which old elements
%cover portions of their area.

PamDir=PamDat.Element_Orientations;
PamStrain=PamDat.Strain;
Eldat=cell(numelsNEW,numparts);
parfor i=1:numelsNEW
    AoverI=Aover{i};
    for k=1:numparts

        elnums=find(AoverI(:,k));%determines which old elements
        %within the k'th part overlap the i'th new element
        Aval=AoverI(elnums,k);
```

```
if sum(elnums)>0
    Ratio1=Aval/A(i);
    %the area fraction taken by each overlapping element
    FullRatio=sum(Ratio1);
    %the area fraction taken by all the overlapping elements
    % if FullRatio < 1, there are gaps in the layer
    % if FullRatio > 1, there are folds/overlaps in the layer
    % only if FullRatio == 1 is there smooth coverage
    Ratio2=Aval/sum(Aval);
    %The area fraction with respect to total overlapping area. This
    %may vary from Ratio1 in those cases involving gaps or folds.


    dir=sum(Ratio2.*PamDir(elnums,:),1);
    strain=sum(Ratio2.*PamStrain(elnums,:),1);
    %using a weighted average based on the area of overlap of
    %each element in order to estimate the strins and fiber
    %directions within each element


    shift=zeros(length(elnums),1);
    for q=1:length(elnums)
        shift(q)=angler(dir(1:3),PamDir(elnums(q),1:3));
    end
    maxshift=max(abs(shift)); %determining the maximum angle of
    % offset between averaged fiber direction and the direction
```

```
            % from any related element
            if maxshift>30
                warning(strcat(['The fiber orientation algorithm has',...
                    ' detected a discontinuity. There is a change in',...
                    ' fiber angle of magnitude %d degrees']),maxshift)
            end
            Eldat{i,k}=[FullRatio,dir,strain,maxshift];
        end

    end
end
ElmVecs=Elvecs(D.NewEl,D.NewNod);
D.ElmVecs=ElmVecs;
thick=thickness(D.ZDat,D.NewNod,D.NewEl,PamDat.Parts,thikdef,ElmVecs);

%calling the thickness function to determine the thecknesses of each layer
%in each element

AllData=cell(numelsNEW,1);
parfor i=1:numelsNEW
    Dat=zeros(numparts,3);
    N=ElmVecs.N(i,:);
    Z=ElmVecs.Z(i,:);
    H=ElmVecs.H(i,:);
    for j=1:numparts
        if((~isnan(thick(i,j)))&&(~isempty(Eldat{i,j})))
%             if isempty(Eldat{i,j})
```

```
%                 error(strcat(['The array of element thicknesses does',...
%                     ' not correspond to that of fiber data.   %d']),i)
%             end
            ed=Eldat{i,j};
            T=thick(i,j);
            dir1=ed(2:4);
            V=volfrac(ed,T,thikdef,VFdef,i);
            A=fiberangle(N,Z,H,dir1);
            Dat(j,:)=[A,T,V];
        end

    end

    AllData{i}=Dat(find(Dat(:,3)),:);
end

V=[];
numlayers=zeros(length(AllData),1);
for i=1:length(AllData)
  ADD=AllData{i};
  ADDD=ADD(:,3);
  V=[V;ADDD];
  numlayers(i)=length(ADDD);
end
D.Volfrax=unique(V);
% this finds the list of unique fiber volume fractions used in the model.
% Since the volume fraction is rounded to the 3rd decimal place and has
```

```
% bounds at 0 and 0.9, this array is never more than 901 elements long

D.L_percent=length(numlayers(numlayers<numparts))/length(numlayers)*100;
% Checking what percentage of the elements contain fewer layers than the
% PAM-Form model. Those elements in regions where the layers do not overlap
% will be counted here.


D.Layups=AllData;

end

function truthik=thickness(Zdat, Nod, El,Parts,defthik,V)
% This determines the local thicknesses from the vector of Z-positions

N=V.N;

numnodes=length(Zdat); %the number of new nodes
numparts=length(Parts); %the number of PamForm parts
numels=length(El(:,1)); % the number of new elements

thick=NaN(numnodes,numparts);
IZ=thick(1,:);
parfor i=1:numnodes %going through each node
    Zi=Zdat{i};
    Zpart=interp1([0;Parts],0:numparts,Zi(:,2),'next');
    % the part number for each zvalue
```

```
Zpos=Zi(:,1);% the z values alone



if max((Zpart)~=sort(Zpart))>0
    error(strcat(['Z-positions are misordered. This may occur if',...
        'part numbers do not correspond to relative position in',...
        'the layup. Ths can be mitigated by ensuring that the',...
        'PAM-Form parts are ordered as they appear in the layup']))
    % this error will be thrown if the z values are aligned in an order
    % not coincident with the part ordering
end
if length(Zpos)>1
    Gap=Zpos(2:end)-Zpos(1:end-1);
    THK=(1/2)*([Gap(1);Gap]+[Gap;Gap(end)]);
else
    THK=inf;
    % since the above calculation only works if there are multiple
    % parts involved, this if statement will tag the part as having
    % default thickness. Since this value is still in the z-direction,
    % the tag is done using infinity, so that it can be easily found
    % later once the transformation has been done.
end
% This determines the thickness(in the z-direction) of each part in the
% layup assuming that the gap between elements is shared evenly
% between the parts. This also assumes that the top and bottom layers
% each have an element location that directly corresponds to the
% midsurface of that element
```

```
    %i.e. the section of gap that has been
    % shared with the nearest element is exactly half the thickness of the
    % surface element.
    if length(Zpart)~=length(unique(Zpart))
        %This will be tripped iff a there are multiple instances of a
        %particulat part being referenced. This could happen in the case
        %where there are multiple elements from one part overlapping in
        %this location
        [Zpart,THK]=thikchange(Zpart,THK,numparts);
    end
    THKK=IZ;
    THKK(Zpart)=abs(THK');
    thick(i,:)=THKK;
end


truthik=NaN*ones(numels,numparts);
parfor i=1:numels
    THK1=thick(El(i,:),:)*N(i,3);
    % transforming into the thickness direction from the z-direction
    THK1(isinf(THK1))=defthik;
    %replacing those thickness elements (that were tagged as default using
    %infinity) with the default thickness
    truthik(i,:)=mean(THK1,1,'omitnan')*N(i,3);
    % this takes the average of the thicknesses of each part from the four
    % nodes of the i'th element. it then takes the dot product of this
    % thickness with the element normal to convert it into the thickness in
```

```
    % the normal direction rather than  in the z direction.


    % since the previous thickness value for each part is simply the
    % magnitude of a vector in the z-direction, the dot product simplifys
    % to multiplying the thickness value by the z-component of the unit
    % normal vector.
end


end


function [ZPartNew,THKnew]=thikchange(Zpart,THK,numparts)
% this function takes a z-vector in which multiple elements represent a
% single part. it then sums the thicknesses
k=0;
ZZ=zeros(numparts,1);
ZP=ZZ;
for i=1:numparts
    f=find(Zpart==i);
    if ~isempty(f)
        k=k+1;
        ZZ(k)=sum(THK(f));
        ZP(k)=i;


    end
end
ZPartNew=ZP(1:k);
THKnew=ZZ(1:k);
```

```
end

function theta=angler(u,v)
% this function determines the angle between two vectors
    theta=atan2d(norm(cross(u,v)),dot(u,v));
end

function theta=angler2(u,x,y)
% this function determines the angle between the vector u and x, on the x-y
% plane
theta=sign(dot(u,y))*angler(u,x);
% this finds the angle magnitude using angler and assigns a direction based
% on the projection of u onto y.
end

function V=volfrac(ed, thickness,init_thick, init_V,elmnum)
% This function determines the effective fiber volume fraction for the
% element considering the strains, thicknesses, and fiber overlap ratio
strain=ed(8:10);
fibratio=full(ed(1));

L_1=strain(1)+1;
L_2=strain(2)+1;
L_t=thickness/init_thick;
% stretch ratios in the fiber, transverse, and thickness directions
V=round((init_V*fibratio*(1/L_1)*(1/L_2)*(1/L_t)),3);
% volume fraction rounded to thousandth in order to limit the size of the
```

```
% input file
if V<0
    V=0;
    warning(strcat(['The lower bound volume fraction has been reached',...
        'in element #%i.\n  The fiber areal ratio given as: %G\n',...
        'The thickness is given as: %G']),elmnum,fibratio,thickness)
end
if V>0.9
    V=0.9;
    % due to inconsistency in PAM-Form contact behavior, there are
    % occasionlally situations in which this volume analysis yields
    % unreasonable volume-fraction. To solve this, 0.9 has been set as an
    % upper bound due to the theoretical packing maximum (assuming
    % hexagonal close packing) is 0.907.
    warning(strcat(['The upper bound volume fraction has been reached',...
        'in element #%i.\n  The fiber areal ratio given as: %G\n',...
        'The thickness is given as: %G']),elmnum,fibratio,thickness)
end
end

function A=fiberangle(N,Z,H,dir1)
dirflat=dir1-dot(dir1,N)*N;
%projecting the direction onto the element plane by subtracting the portion
%projected onto the normal

A=angler2(dirflat,Z,H);
end
```

```
function V=Elvecs(El,Nod)
% This function calculates the global vectors for the local coordinate
% system of each element. Each row of the N,Z,and H arrays represent the
% normal, and principal planar vectors for the corresponding element

XV=Nod(:,1);
YV=Nod(:,2);
ZV=Nod(:,3); % Vectors of X,Y,and Z values of each node
EX=XV(El);
EY=YV(El);
EZ=ZV(El);% Matrices of X,Y,and Z values of each corner of each element
numels=length(El(:,1));
N=zeros(numels,3);
H=zeros(numels,3);
Z=zeros(numels,3);

zaxis=(QSfun(1,0)-QSfun(-1,0));
haxis=(QSfun(0,1)-QSfun(0,-1));
axes=[zaxis,haxis];
% using isoparametric shape functions to determine the axes of every
% element w.r.t. their corner points
parfor i=1:numels
   AX=[EX(i,:)*axes;EY(i,:)*axes;EZ(i,:)*axes];
   NN=(cross(AX(:,1),AX(:,2)))';
   N(i,:)=NN/norm(NN);
   Z(i,:)=AX(:,1)'/norm(AX(:,1));
```

```
   H(i,:)=AX(:,2)'/norm(AX(:,2));
   %Computing the global vector representation of the i'th element's local
   %axes, then taking the cross product of those vectors to represent the
   %element normal. this is then divided by its magnitude in order to
   %generate a unit normal
end
V=struct('N',N,'Z',Z,'H',H);
end

function M=MaterialEstimate(fiber,matrix,fraction,type)
% Elastic Properties Using Mori-Tanaka Method
M=MoriTanaka(fiber,matrix,fraction,type);

%Strength Properties Using Seigars-Daniel-Barbero Method
M=Strengths(M,fiber,matrix);
end

function composite = MoriTanaka(fiber,matrix,fraction,type)
%%***** Mori-Tanaka Model for unidirectional fiber reinforced composites **

%*** INPUTS ***
%  Fiber properties E1f, E2f, Nu12, G12f, G23f (transversely isotropic)
%  Matrix properties Em, Num (isotropic)
%  Fiber volume fraction Vf

%*** OUTPUTS ***
%  Effective properties of the composite material E1,E2,Nu12,G12,G23
```

```
switch type
    case 'volume'
        Vf=fraction;
    case 'weight'
        WF=fraction;
        rf=fiber.ro;
        rm=matrix.ro;
        Vf=1/(1+(rf/rm)*((1/WF)-1));
    otherwise
        error('the "type" tag must be either "volume" or "weight"')
end
%converting weight fractions to volume fractions

E1f=fiber.E1;
E2f=fiber.E2;
G12f=fiber.G12;
G23f=fiber.G23;
Nu12f=fiber.nu12;

Em=matrix.E;
Num=matrix.nu;
%Gathering variables from material structures

%***  Matrix volume fraction ***
Vm = 1-Vf;

%*** Determine the Hill's elastic moduli for the fiber ***
```

```
kf = 1/(4/E2f-1/G23f-4*Nu12f^2/E1f);
lf = 2*Nu12f*kf;
mf = G23f;
nf = E1f+lf^2/kf;
pf = G12f;

%*** Determine the Hill's elastic moduli for the matrix ***
km =  Em/(2-2*Num - 4*Num^2);
lm = 2*Num*km;
mm = Em/(2*(1+Num));
nm = Em+lm^2/km;
pm = Em/(2*(1+Num));

%*** Calculate the effective Hill's elastic moduli for the composite material ***
k = (Vf*kf*(km + mm) + Vm*km*(kf + mm))/(Vf*(km + mm) + Vm*(kf + mm));
l = (Vf*lf*(km + mm) + Vm*lm*(kf + mm))/(Vf*(km + mm) + Vm*(kf + mm));
m = (mm*mf*(km+2*mm) + km*mm*(Vf*mf+Vm*mm))/(km*mm + (km + 2*mm)*(Vf*mm+Vm*mf));
n =  Vm*nm + Vf*nf +(l - Vf*lf - Vm*lm)*((lf-lm)/(kf-km));
p = (2*Vf*pm*pf+Vm*(pm*pf+pm^2))/(2*Vf*pm + Vm*(pf+pm));

%*** Calculate the effective engineering properties for the composite material ***
E1 = n - l^2/k;
E2 = 4*m*(k*n-l^2)/((k+m)*n-l^2);
Nu12 = l/(2*k);
G12 = p;

G23 = m;
```

```
Nu23 = E2/(2*G23)-1;

composite=struct('E1',E1,'E2',E2,'Nu12',Nu12,'G12',G12,'G23',G23,...
    'Nu23',Nu23,'VolumeFraction',Vf);
end

function M=Strengths(M,Fiber,Matrix)
% Adapted from Seigars: "Feasibility of Hybrid Thermoplastic Composite-Concrete Load
% Bearing System" 2018. Used with Permission
%*********************************%
%* Lamina Strength Calculations *%
%*********************************%
Vf=M.VolumeFraction;
Vm=1-Vf;

Em=Matrix.E*1e9;
E1f=Fiber.E1*1e9;
Fft=Fiber.F1t*1e9;

Fmt=Matrix.Ft*1e9;
Fmc=Matrix.Fc*1e9;

E1=M.E1*1e9;
E2=M.E2*1e9;
nu12=M.Nu12;
G12=M.G12*1e9;
% Gathering necessary properties from respective structure arrays.
```

```
% Note this code assumes units of Pa, while the structures store in units
% of GPa, therefore all strengths and moduli must be multiplied by 1e6.


% Lamina Tensile Strengths %
%fiber dominated failure mode,  F1tf
%matrix dominated failure mode, F1tm
F1tf = Fft*(Vf+Em/E1f*Vm); %(Daniel Equation 5.7)
%Longitudinal tensile strength (fiber dominated)
F1tm = Fmt*(E1f/Em*Vf+Vm); %(Daniel Equation 5.8)
%Longitudinal tensile strength (matrix dominated)
%Check for Fiber or Matric Dominated Failure
%Based on discussion with Ben Smith and Phil Bean, June 2018
F1t = min(F1tf,F1tm);      %Daniel Textbook Section 5.2, pp. 98-100


% Fracture Toughness Mode I %
GIc = 334; % Set to E-glass/Polyester value,(Barbero Table 1.3)
%E-glass/Polyester value of 334 J/m^2 is the default value used from
%Barbero Table 1.3 if the value is not known for the material system
%being analyzed.
lambda022 = 2*(1/E2-nu12^2*E2^2/E1^3); %(Barbero Equation 4.100)

% Transitiion Thickness %
ttran = 0.0006; %Set to E-glass/Epoxy,(Barbero Section 4.4.8)

% Transverse Tensile Strength %
```

```
F2t = sqrt(GIc/1.12^2/pi()/(ttran/4)/lambda022);
%(Barbero Equation 4.99)

% Fracture Toughness Mode II %
GIIc = 456; % Set to E-glass/Polyester,(Barbero Table 1.3)
%E-glass/Polyester value of 456 J/m^2 is the default value used from
%Barbero Table 1.3 if the value is not known for the material system
%being analyzed.
lambda044 = 1/G12; % (Barbero Equation 4.114)

% In-Plane Shear Strength %
F6 = sqrt(GIIc/pi()/(ttran/4)/lambda044); %(Barbero Equation 4.113)

% Fiber Misalignment, see Barbero Table 1.3 %
alphasigma = 2.97*pi()/180; %Set to E-glass/Epoxy (2.97 degrees)
chi = G12*alphasigma/F6; %(Barbero Equation 4.86)

% Longitudinal Compressive Strength %
F1c = G12*(1+4.76*chi)^-0.69; %(Barbero Equation 4.85)

%Adjustment Factor for Voids
Cv = 1;
%Set to 1 if no voids is assumed, else see Barbero Equation 4.102

% Transverse Compressive Strength %
F2c = Fmc*Cv*(1+(Vf-sqrt(Vf))*(1-Em/E1f)); %(Barbero Equation 4.105)
```

```
M.F1t=F1t/1e9;
M.F1c=F1c/1e9;
M.F2t=F2t/1e9;
M.F2c=F2c/1e9;
M.F6=F6/1e9;
%Converting back to GPa, and appending to the material structure
end
```

Generating Input File

```
function Abaqus_Input2(D)
% This function generates an input file for an Abaqus model based on the
% PAM-Form model details stored in the data structure D.

[F,P]=uiputfile({'*.inp','Abaqus Input Files';'*.*','All Files'},...
    'Export As');
F_ID=fopen(strcat(P,F),'w+');
fprintf(F_ID,'%s\n%s\n%s\n%s\n%s\n%s\n%s\n%s\n%s\n%s\n','*Heading',...
    '**Job name: PAM-Form Post','**Generated by: Matlab Script',...
    '*Preprint, echo=NO, model=NO, history=NO, contact=NO',...
    '**','** PARTS','**','*Part, name=MeshPart','*Node, nset=Elms');
for i=1:length(D.NewNod(:,1))
    fprintf(F_ID,'%i , %G , %G , %G\n',i,D.NewNod(i,:));
end
fprintf(F_ID,'\n%s\n','*Element, type=S4R, elset=Elms');

for i=1:length(D.NewEl(:,1))
    fprintf(F_ID,'\n%s\n',strcat(['*Element, type=S4R, elset=El',...
```

```
        num2str(i)]));
    fprintf(F_ID,'%i , %G , %G , %G , %G\n',i,D.NewEl(i,:));
end
fprintf(F_ID,'%s\n%s\n','*Elset,elset=Elms, generate',...
    strcat(['1, ',num2str(length(D.NewEl(:,1))),',  1']));
fprintf(F_ID,'%s\n%s\n%s',...
    '*Orientation, name=Mat_Ori, system=RECTANGULAR',...
    'Mat_Orient','3 , 0.');

fprintf(F_ID,'\n%s\n%','** Section: EntirePart');
for i=1:length(D.NewEl(:,1))
    IS=num2str(i);
    fprintf(F_ID,'%s\n',strcat(['*Shell Section, elset=El',IS,...
        ',  composite, orientation=Mat_Ori']));
    Li=D.Layups{i};
    for j=1:length(Li(:,1))
        fprintf(F_ID,'%-08.4G, 3, M%-3i, %G\n',Li(j,2),1000*Li(j,3),Li(j,1));
    end
end

fprintf(F_ID,'%s\n%s\n',...
    '*Distribution, name=Mat_Orient, location=ELEMENT, Table=ORTAB',...
    ', 1., 0., 0., 0., 1., 0.');
```

```
for i=1:length(D.ElmVecs.Z(:,1))
    fprintf(F_ID,'%i , %i , %i , %i , %i ,%i , %i\n',i,D.ElmVecs.Z(i,:),...
        D.ElmVecs.H(i,:));
end

fprintf(F_ID,'%s\n\n%s\n%s\n','*End Part','** MATERIALS','**');

for i=1:length(D.Volfrax)
fprintf(F_ID,'%s\n%s\n',strcat(['*Material, name=M',...
    num2str(D.Volfrax(i)*1000)]),'*Elastic, type=LAMINA');
C=MaterialEstimate(D.Fiber,D.Matrix,D.Volfrax(i),'volume');
fprintf(F_ID,'%G , %G , %G , %G , %G , %G\n',...
    [C.E1,C.E2,C.Nu12,C.G12,C.G12,C.G23]);
fprintf(F_ID,'%s\n','*Fail Stress');
fprintf(F_ID,'%G , %G , %G , %G , %G , %G , %G\n',...
    [C.F1t,-C.F1c,C.F2t,-C.F2c,C.F6,-0.5,0]);
% Writing material stiffnesses and strengths to the file.
% Note that the abaqus failure analysis requires the compressive strengths
% be negative
end
fprintf(F_ID,'*Distribution Table, name=ORTAB\ncoord3d, coord3d');

fclose(F_ID);
end
```

Mesh Trimming GUI

```
function D3=TrimGUI(D)
```

```
% This presents a gui to trim the new mesh, and remove all unnecessary
% nodes and elements.
close all
defbox=D.TrimDef;
%default box corners for trim box


H.D=D;
H.D2=D;


f=figure('Name','Mesh Trimmer','Position',[100,100,800,500],...
    'Resize','off','MenuBar','none','CloseRequestFcn',@Closer);


H.ax=axes(f,'Units','pixels','Position',[50,50,400,400],...
    'NextPlot','add');




H.Explanations={strcat(['Trim Mode: Define a bounding polygon by',...
    ' adding corners to the list. This can be done by clicking on the',...
    'plot, or by typing into the x ans y boxes']),...
    strcat(['Inspection Mode: Look over the',...
        ' model in order to verify the bounding box'])};
H.Explain=uicontrol('Style','text','Position',[500,400,250,90],...
    'String',H.Explanations{1});
H.TNVal=[];
H.BoxCorners=defbox;
```

```
Modes=uipanel('Parent',f,'Units','pixels',...
    'Position',[100,450,300,50]);
H.TrimButton=uicontrol(Modes,'Style','togglebutton',...
    'Position',[10,10,130,30],'String','Trim','Value',1,...
    'Callback',@Swapper,'FontSize',14);
H.InspectButton=uicontrol(Modes,'Style','togglebutton',...
    'Position',[160,10,130,30],'String','Inspect','Value',0,...
    'Callback',@Swapper,'FontSize',14);
H.DisableList={'XL','XV','YL','YV','AddBtn','CTab','TUndo','TReset',...
    'CutBtn'};


XYPanel=uipanel('Parent',f,'Units','pixels',...
    'Position',[515,360,220,70]);
H.XV=uicontrol(XYPanel,'Style','edit','Position',[10,40,50,20],...
    'HorizontalAlignment','right','Callback',@Val);
H.XL=uicontrol(XYPanel,'Style','text',...
    'Position',[12,41,16,16],...
    'String','X:','BackgroundColor',[1,1,1],...
    'HorizontalAlignment','left');
H.YV=uicontrol(XYPanel,'Style','edit','Position',[10,10,50,20],...
    'HorizontalAlignment','right','Callback',@Val);
H.YL=uicontrol(XYPanel,'Style','text',...
    'Position',[12,11,16,16],...
    'String','Y:','BackgroundColor',[1,1,1],...
    'HorizontalAlignment','left');
H.AddBtn=uicontrol(XYPanel,'Style','pushbutton','String','Add Corner',...
```

```
    'Position',[80,10,130,50],'FontSize',16,'Enable','off',...
    'Callback',@AddFcn);

P=[515,100,220,200];
F=16;
TBox=uipanel('Parent',f,'Units','pixels','Position',P);
H.CTab=uitable(TBox,'Data',H.BoxCorners,'Position',...
    [10,10,100,P(4)-20],'ColumnName',{'X:','Y:'},'RowName','numbered',...
    'ColumnWidth',{38,37.9});
H.TUndo=uicontrol(TBox,'Style','pushbutton','String','Undo',...
    'Enable','off','FontSize',F,'Position',[120,P(4)/2+10,90,30],...
    'Callback',@TEdit);
H.TReset=uicontrol(TBox,'Style','pushbutton','String','Reset',...
    'Enable','off','FontSize',F,'Position',[120,P(4)/2-40,90,30],...
    'Callback',@TEdit);



P=[515,10,220,50];
F=15;
RBox=uipanel('Parent',f,'Units','pixels','Position',P);
H.CutBtn=uicontrol(RBox,'Style','pushbutton','String','Trim',...
    'Enable','off','Position',[P(3)/2-30,10,60,30],'FontSize',F,...
    'Callback',@CutFun);
H.FReset=uicontrol(RBox,'Style','pushbutton','String','Reset',...
    'Enable','off','Position',[P(3)-70,10,60,30],'FontSize',F,...
    'Callback',@Reset);
H.Finish=uicontrol(RBox,'Style','pushbutton','String','Done',...
```

```
    'Position',[10,10,60,30],'FontSize',F,...
    'Callback',@Finish);

H=PlotAll(H);
guidata(f,H);
uiwait
try
    H=guidata(f);
    D3=H.D2;
catch
    error('Window Closed Before Analysis Complete')
end
close all

end

function CutFun(O,~)
H=guidata(O);
X=H.D2.NewNod(:,1);
Y=H.D2.NewNod(:,2);
NodIn=isinterior(H.Poly,X,Y);
%determining which nodes are within the bounding polygon
H.D2.In_Num=NodIn;
H.D2.Interior='hi';
H.D2=ElmRem(H.D2);
H.D2.ZDat=H.D2.ZDat(find(NodIn));
H=CornerClear(H);
```

```
set(H.FReset,'Enable','on')
guidata(O,H)
end

function H=CornerClear(H)
H.BoxCorners=[];
H.TNVal=[];
H.XV.String='';
H.YV.String='';
H=TUpdate(H);
end

function Reset(O,~)
H=guidata(O);
H.D2=H.D;
H=CornerClear(H);
guidata(O,H)
end

function Finish(~,~)

a=questdlg('Done Trimming?','Mesh','yes','no','no');
switch a
    case 'yes'
    uiresume
    set(gcf,'CloseRequestFcn',@(~,~)delete(gcf))
```

```
end


end


function Closer(O,~)
a=questdlg('Mesh Trimming is required. Are you sure you want to quit?',...
    'Exit?','yes','no','no');
switch a
    case 'yes'
        delete(O)
end
end

function AddFcn(O,~)
H=guidata(O);
V=[H.BoxCorners;H.TNVal];
H.BoxCorners=V;
H.TNVal=[];
set(H.AddBtn,'Enable','off')
set(H.XV,'String','')
set(H.YV,'String','')
H=TUpdate(H);
guidata(O,H)


end
```

```
function H=TUpdate(H)
set(H.CTab,'Data',H.BoxCorners)
if size(H.BoxCorners,1)>0
    set(H.TUndo,'Enable','on')
    set(H.TReset,'Enable','on')
else
    set(H.TUndo,'Enable','off')
    set(H.TReset,'Enable','off')
end
H=PlotAll(H);
end

function TEdit(O,~)
H=guidata(O);
switch O.String
    case 'Undo'
        H.BoxCorners=H.BoxCorners(1:end-1,:);
    case 'Reset'
        H.BoxCorners=[];
end
set(H.AddBtn,'Enable','off')
H.TNVal=[];
H=TUpdate(H);
guidata(O,H)
end

function Mouser(O,~)
```

```
C=get(O,'CurrentPoint');
XV=get(O,'XLim');
YV=get(O,'YLim');
ZV=get(O,'ZLim');
CC=round(C(1,:));

H=guidata(O);
set(H.XV,'String',num2str(CC(1)))
set(H.YV,'String',num2str(CC(2)))
H=Change(H);
guidata(O,H)
end

function Val(O,~)
H=guidata(O);
S=O.String;
if isnan(str2double(S))
    set(O,'String','')
end
H=Change(H);
guidata(O,H)
end

function H=Change(H)
X=str2double(H.XV.String);
Y=str2double(H.YV.String);
if (~isnan(X))&&(~isnan(Y))
```

```
    H.TNVal=[X,Y];
    set(H.AddBtn,'Enable','on')
else
    set(H.AddBtn,'Enable','off')
    H.TNVal=[];
end
H=PlotAll(H);

end

function H=PlotAll(H)
plotter(H.D2,H.ax);
set(H.ax,'XLimMode','manual','YLimMode','manual')
H.Overax=copyobj(H.ax,gcf);
H.Overax.ButtonDownFcn=@Mouser;
delete(H.Overax.Children(:))
set(H.Overax,'Color',[1,1,1,0],'XColor',[1,1,1,0],'YColor',[1,1,1,0])
B=H.BoxCorners;
V=H.TNVal;
if ~isempty(B)
    plot(H.ax,B(:,1),B(:,2),'g*')
end
if ~isempty(V)
    plot(H.ax,V(1),V(2),'r*')
end
C=[B;V];
if size(C,1)>2
```

```
    S2=polyshape(C);
    plot(H.ax,S2,'FaceColor','r','FaceAlpha',0)
end
if size(B,1)>2
    S1=polyshape(B);
    H.Poly=S1;
    set(H.CutBtn,'Enable','on')
    plot(H.ax,S1,'FaceColor','g','FaceAlpha',0.3)
else
    H.Poly=[];
    set(H.CutBtn,'Enable','off')
end
end

function Swapper(O,~)
H=guidata(O);
n=O.String;
switch n
    case 'Inspect'
        if H.TrimButton.Value==1
            set(H.TrimButton,'Value',0)
            set(H.InspectButton,'Value',1)
            inspectionmode(1,H)
        else
            set(H.TrimButton,'Value',0)
            set(H.InspectButton,'Value',1)
        end
```

```
    case 'Trim'
        if H.InspectButton.Value==1
            set(H.TrimButton,'Value',1)
            set(H.InspectButton,'Value',0)
            inspectionmode(0,H)
        else
            set(H.TrimButton,'Value',1)
            set(H.InspectButton,'Value',0)
        end
end
guidata(O,H)

end

function inspectionmode(mode,H)
if mode==1
    disp('inspection!!!!')
    set(H.Explain,'String',H.Explanations{2})
    for i=1:length(H.DisableList)
        set(H.(H.DisableList{i}),'Enable','off')
    end
    set(gcf,'ToolBar','Figure','DockControls','off')
    buttonsoff={'Show Plot Tools','Hide Plot Tools','Print Figure',...
        'Save Figure','Open File','New Figure','Insert Legend',...
        'Insert Colorbar','Data Cursor','Edit Plot','Brush/Select Data',...
        'Link Plot','Show Plot Tools and Dock Figure'};
```

```
    for i=1:length(buttonsoff)
        set(findall(gcf,'ToolTipString',buttonsoff{i}),'Visible','off')
    end
    axes(H.ax)
else
    disp('trim!!!')
    set(H.Explain,'String',H.Explanations{1})
    for i=1:length(H.DisableList)
        set(H.(H.DisableList{i}),'Enable','on')
    end

    s=get(findall(gcf,'ToolTipString','Rotate 3D'),'State');
    if strcmp(s,'on')
        rotate3d
    end
    zoom(gcf,'off')
    pan(gcf,'off')



    set(gcf,'ToolBar','none')
    TUpdate(Change(H));

end
end

function plotter(D,ax)
 %gathering x,y,and z values and plotting the mesh
```

```
cla(ax)
X=D.NewNod(:,1);
Y=D.NewNod(:,2);
Z=D.NewNod(:,3);

quadmesh2(D.NewEl,X,Y,Z,ax)
set(ax,'CameraPosition',[0,0,1000],'CameraUpVector',[0,1,0])

end

function hh = quadmesh2(quad,x,y,z,ax,varargin)
%QUADMESH Quadrilateral mesh plot.
%   QUADMESH(QUAD,X,Y,Z,C) displays the quadrilaterals defined in the M-by-4
%   face matrix QUAD as a mesh.  A row of QUAD contains indexes into
%   the X,Y, and Z vertex vectors to define a single quadrilateral face.
%   The edge color is defined by the vector C.
%
%   QUADMESH(QUAD,X,Y,Z) uses C = Z, so color is proportional to surface
%   height.
%
%   QUADMESH(TRI,X,Y) displays the quadrilaterals in a 2-d plot.
%
%   H = QUADMESH(...) returns a handle to the displayed quadrilaterals.
%
%   QUADMESH(...,'param','value','param','value'...) allows additional
%   patch param/value pairs to be used when creating the patch object.
%
```

```
%   See also PATCH.
%
% Script code based on copyrighted code from mathworks for TRIMESH.
% Allan P. Engsig-Karup, apek@mek.dtu.dk.
% Taken from Mathworks File-Exchage
% Modified 7/10/2018 Philip Bean




if nargin == 4 || (nargin > 5 && ischar(z))
  d = tri(:,[1 2 3 4 1])';
  if nargin == 4
    h = plot(ax, x(d), y(d));
  else
    h = plot(ax, x(d), y(d),z,varargin{1},varargin{2:end});
  end
  if nargout == 1, hh = h; end
  return;
end


start = 1;
if nargin>5 && rem(nargin-4,2)==2
  c = varargin{1};
  start = 2;
elseif nargin<4
  error(id('NotEnoughInputs'),'Not enough input arguments');
else
```

```
  c = z;
end

if ischar(get(ax,'color'))
  fc = get(gcf,'Color');
else
  fc = get(ax,'color');
end

h = patch('faces',quad,'vertices',[x(:) y(:) z(:)],'facevertexcdata',c(:),...
  'facecolor',fc,'edgecolor',get(ax,'defaultsurfacefacecolor'),...
  'facelighting', 'none', 'edgelighting', 'flat',...
      'parent',ax,'FaceAlpha',0.5,...
  varargin{start:end});
  axis equal
if ~ishold(ax), view(ax,3), grid(ax,'on'), end
if nargout == 1, hh = h; end
end

function str = id(str)
str = ['MATLAB:quadmesh:' str];

end
```

APPENDIX F

LIPSUM

The following is a selection of a typesetting-placeholder text known as “Lorem Ipsum”

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna.

Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

BIOGRAPHY OF THE AUTHOR

Philip Michael Bean was born in Sidney, Maine. He grew up on a small farm, and graduated from Emmanuel Christian Homeschooling Association in Oakland, Maine. Philip then went on to recieve a Bachelor of Science in Mechanical Engineering degree (with a minor in Mathematics) from the Univeristy of Maine in May 2016. He is a member of the ΠΤΣ(pi tau sigma) mechanical engineering honor society, the ΠΜΕ(pi mu epsilon)mathematical honor society, and the Order of the Engineer. Philip Michael Bean is a candidate for the Master of Science degree in Mechanical Engineering from the University of Maine in December 2018.