MECHANICAL MODELING OF A POMELO PEEL BIOINSPIRED FOAM

A Thesis

by

JONEL A ORTIZ

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Daniel McAdams
Committee Members,	Matt Pharr
	Junuthula Narasimha Reddy
Head of Department,	Andreas Polycarpou

December 2017

Major Subject: Mechanical Engineering

Copyright 2017 Jonel A Ortiz

ABSTRACT

Many naturally occurring structures have been shown to exhibit superior mechanical properties when compared to their constituent materials, such as the pomelo fruit, which, due to its hierarchical arrangement, allows the material to excel in both static and dynamic situations. For the case of the pomelo fruit (Citrus maxima) the peel has been shown to handle sudden shock and/or impact remarkably well, despite its relatively weak constituent materials. It has also been shown to exhibit a rather favorable quasi-static response with its high amount of energy dissipation. There have been many efforts to experimentally quantify aspects of the pomelo fruit along with interesting work taking place that ventures to fabricate biomimetic materials deriving from the pomelo fruit such as the utilization of a modified investment foam casting process to fabricate a foam-like material that resembles the mechanical structure of the pomelo peel. Little effort has been made to build mechanical models or finite element models that would allow users to tailor some aspects of the pomelo fruit peel to fit a certain need. The work presented here endeavors to define a feasible method that can be used to quickly build a finite element model using commonly known methods such as a Voronoi tessellation of space to construct the foam on a micrometer scale, while focusing on capturing the unique physics that are occurring due to the arrangement and structure of the pomelo peel. Using the network of beams, an input file can then be written that can be used with a finite element solver to quickly determine the response of the structure.

Of high importance was validating such a model with published literature, and upon completion a tool was made available that would enable designers to modify parameters such as the porosity of the structure at any given point, the cell count, material selection, etc. in a design scenario. The model was validated not only by having its mechanical response compared to the typical response of cellular materials, but was further validated when compared directly to compaction studies of the pomelo peel.

DEDICATION

To my brother, who showed me to always pursue your deepest passions no matter the cost, and who showed me to live the happiest life.

ACKNOWLEDGEMENTS

A big thank you to my parents, my sister, and the rest of my family that have been the best support system somebody could ask for along with continuing to inspire me on a daily basis.

I would also like to thank my committee Chair, Dr. McAdams, for all the help he has given me along with the interest he has sparked in me throughout my graduate studies at Texas A&M. If it was not for him, the work presented here would not have existed.

Thanks to all the friends, colleagues, and staff I have met over the past 2 years, learning and studying at Texas A&M has truly been an amazing experience.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a thesis committee consisting of Dr. Daniel McAdams and Dr. Matt Pharr of the Department of Mechanical Engineering and Dr. J.N. Reddy of the Department of Civil Engineering.

All work for the thesis was completed by the student, Jonel Ortiz, under the advisement of Dr. Daniel McAdams of the Department of Mechanical Engineering.

Funding Sources

This work was made possible in part by The National Science Foundation under Grant Number 1240483. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of The National Science Foundation.

NOMENCLATURE

B/CS	Bryan/College Station
TAMU	Texas A&M University
HPRC	High Performance Research Computing
OSD	Onset Strain of Densification
GUI	Graphical User Interface
SEM	Scanning Electron Microscope

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

1. INTRODUCTION

The pomelo is a fruit native to South and Southeast Asia, with a peel ranging from 2-3 cm that most likely accounts for most of the energy dissipation (Figure 1). The fruit itself can experience falls from 10m to 15m in its natural environment and must be able to withstand that impact without cracking or at the very least allowing the inner fruit to be damaged. The peel, also known as the pericarp, consists of three layers starting with an exocarp referred to as the flavedo, a mesocarp often called the albedo, and the endocarp which surrounds the seed/center of the fruit. The three layers vary with respect to different parameters commonly used in describing cellular materials, such as the *relative density* (ρ/ρ_s), where ρ is the density of the foam and ρ_s is the density of the solid of which the foam is made [1]. There are many other naturally occurring structures that are composed of different layers, albeit serving a different unique purpose, such as the walls of nuts and the shells of the coconuts [2].

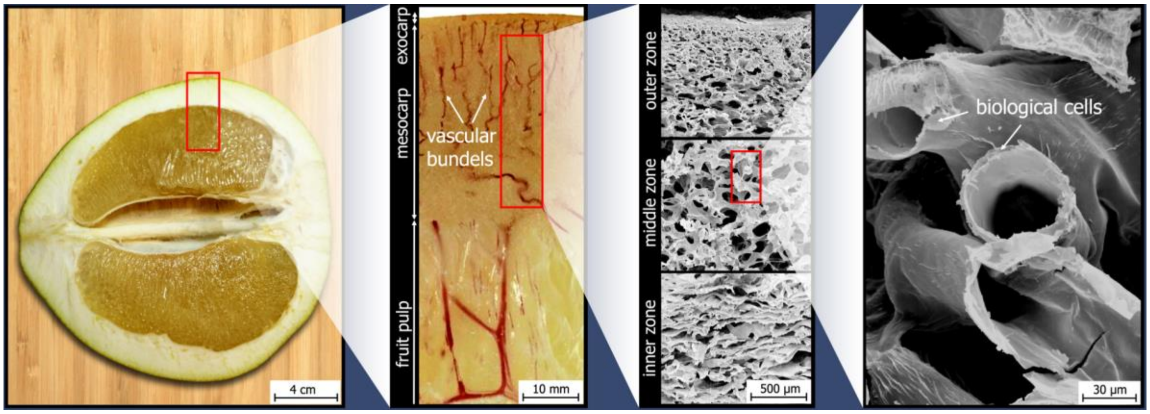


Figure 1. Composition of the pomelo fruit peel, adapted from Thielen et al. [3]

Throughout the course of history, nature has designed and optimized a topology that allows the pomelo to thrive despite it having to withstand such high impact forces. Borrowing from this structure, the work presented here investigates whether the material geometry can be applied to various applications and utilized to possibly outperform other materials that do not capture the unique structure of the pomelo peel. Being able to simulate and replicate the key traits of the pomelo peel would not only prove enlightening but also serve as an additional option to designers looking for materials to suit their needs.

Impact damping and energy dissipating materials can have a host of applications ranging from simple packaging to high speed vehicular impacts along with other safety concerns in a wide range of industries. As a form of beginning work with the pomelo, building a functional mechanical model under quasi-static and dynamic conditions that replicates real-world results would serve as a foothold to expanding the model to include high impact crushing and other loading scenarios. For the pomelo peel, some of the critical characteristics that will be modified and tested in this paper are the gradient to which the foam varies from dense to less dense (when related to foams, this is often called the porosity of the foam), the overall thickness of the foam, and many properties related to the individual struts in the design. It has been seen that the overall structure of the foam can help in dispersing up to 90% of the impact energy [4], and a unique characteristic of the peel known as the vascular bundles (Figure 2) act as stiffening elements within the structure, showing a higher resistance to compressive forces than the surrounding foamy peel [5]. Assessing the full impact of design variables such as these could expand the applicability and property range of a bioinspired pomelo peel foam.

Recently there has been many projects determining the different material properties of both the pomelo peel itself and open cell foams that are inspired by the pomelo peel but none to generate a model that would accurately reflect the material properties of either one. To be able to tune the design variables mentioned, first a functional mechanical model must be developed that can be used in a finite element software in a way that can predict how the material will behave before fabricating such a foam. Then, there needs to be a way to validate the results derived from the finite element model, which can be done by checking the computed property values and comparing them with experimental results from either the work presented here or other published projects.

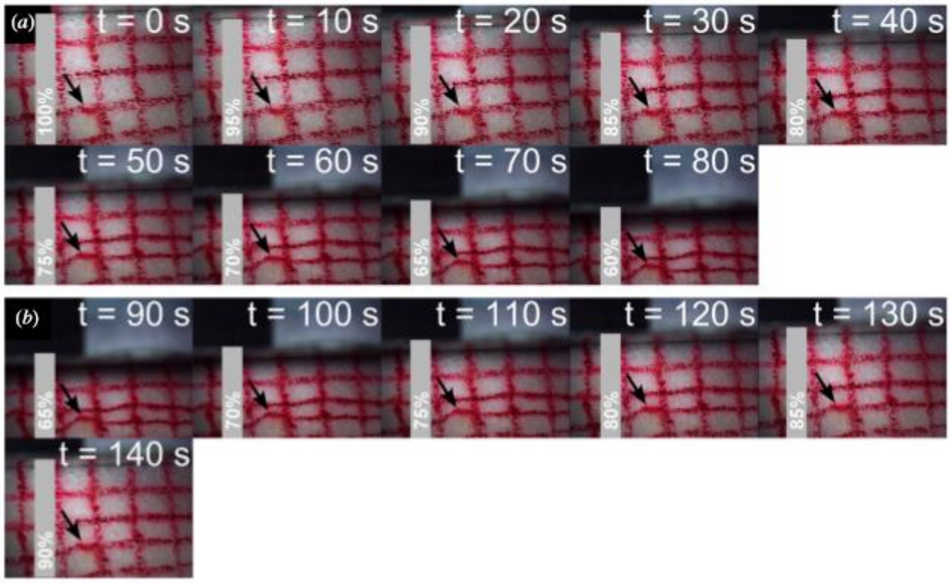


Figure 2. Compaction behavior of a vascular bundle, adapted from Thielen et al. [5]

2. LITERATURE REVIEW

There have been many efforts made to fabricate a bioinspired material with a structure similar to the pomelo fruit peel, such as utilizing a modified investment casting process [6] using $Bi_{57}Sn_{43}$ which performed better than a typically popular aluminum alloy that is usually used in casting processes ($AlSi_7Mg$). Another approach was taken by Sebastian et al. [7] which produced an aluminum/aluminum-silicon-alloy composite tensile specimen to resemble the struts present in the pomelo peel. It can be seen in related literature that varying the overall thickness of the peel and retaining certain characteristics can result in what seem to be different structures but, upon further investigation, are structures that actually share similar principles. Nut shells exhibit something like this by showing a similar hierarchy when compared to the pomelo peel while simultaneously exhibiting toughness comparable to ceramic and glass [8].

As a starting point, a quite famous article written by Ashby and Medalist [1] gives great descriptions of what stress-strain curves should look like for different cellular solids. For the purposes of the work presented here, a focus will be made on the general curve given on pg. 1758 of the same article, shown in Figure 3.

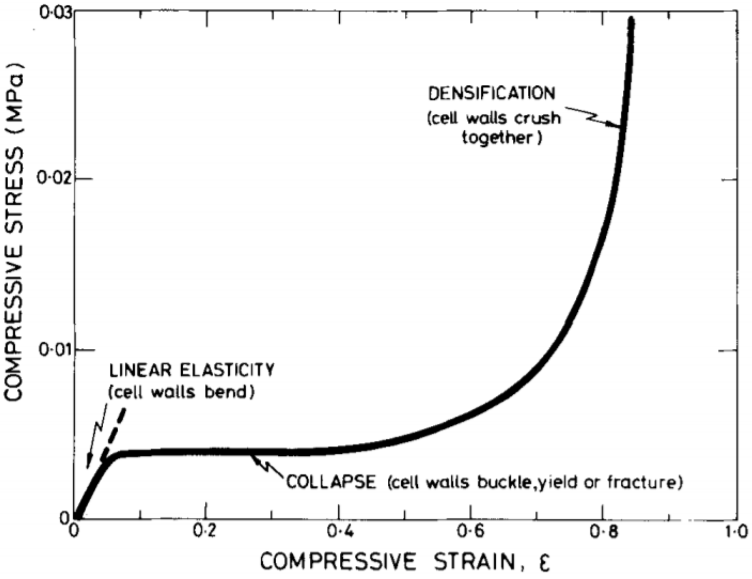


Figure 3. Stress-strain curve for cellular a solid, adapted from Ashby and Medalist [1]

As can be seen, there are three primary regions that can be discerned in Figure 3, the linear-elastic regime, the stress plateau (or collapse) region, and finally the densification region. There are two primary types of foams known as open-cell and closed-cell foams. Mechanically speaking, most foams can be idealized to behave like open-cell foams and, from the microstructure of the pomelo peel given in Figure 4 with the help of a scanning electron microscope (SEM), the model utilized in this study will approximate the peel as such. The main difference between an open-cell and closed-cell foam is that the cells (or pores) of the foam are either closed off from each other or separated by struts (also known as ligaments). This is surprising, since although on a macro scale the peel appears to be a closed-cell foam, studying the peel on the micrometer scale reveals its open-cell like topology.

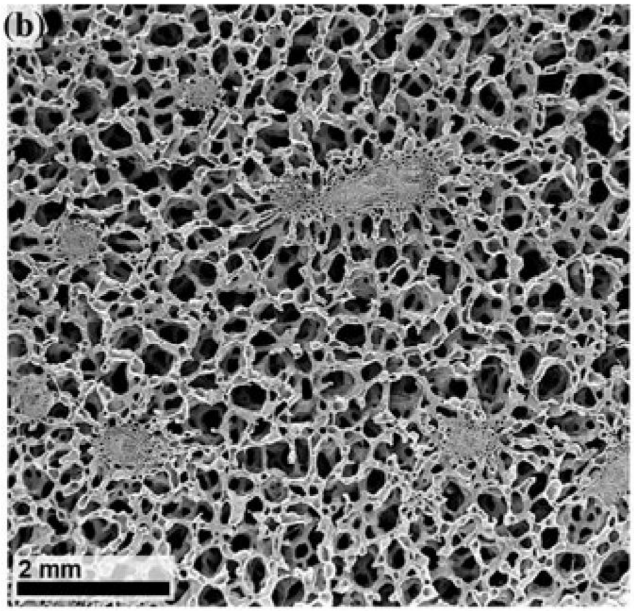


Figure 4. SEM image of the pomelo peel, adapted from Thielen et al. [2]

In the way of computational modeling, there have been numerous studies completed in order to predict the mechanical behavior of a variety of foams, including monodisperse and polydisperse foams. Until recently, it was difficult to model a cellular structure up through densification because such a model would have to include contact forces between ligaments and that would prove to be computationally very expensive. Gaitanaros et al. [9] has been able to do so with impressive agreement when compared to experimental values. Figure 5 shows a computationally generated stress-strain graph for an open cell material undergoing compression. The tractability of modeling an open cell structure with contact is increased when a beam to beam contact algorithm is used. When using a beam to beam contact algorithm there is generally a loss in resolution of the stress occurring at the contact patch between beams, but this is traded off for with a reduction in computational time and effort. This method of analysis is especially suited for simulation of cellular solids since it is not primarily concerned with high resolution of the stresses at the contact patches been each of the beams, rather, the primary concern of such a model is to extract the overall compaction behavior of the structure that arises from these large number of contact interactions. Such a contact algorithm becomes especially becomes useful when introducing many beam elements in the model such that, at any given time, there can be hundreds if not thousands of beams exhibiting the nonlinearities associated with contact stresses.

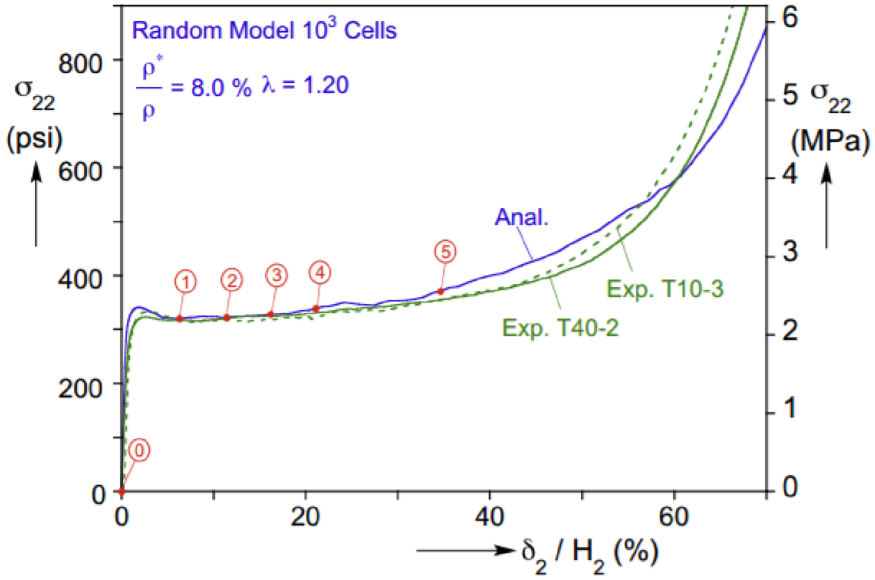


Figure 5. Computational compressive stress-strain curve for an open cell foam, adapted from Gaitanaros et al. [9]

When compared to their own experimental values and the general cellular compression graph depicted in Figure 3 there are some immediate similarities that can be drawn. Primarily the three regions (elastic, plateau, and densification) are captured, as well as the slight variation it has with experimental values that is primarily due to the way each model is produced which introduces some level of randomness into the structure and is a trait that is also introduced for the work presented here.

The pomelo peel itself, as stated earlier, can be modeled as an open-cell structure due to its topology on a micro scale. However, there are still some unique and possibly beneficial characteristics of the peel in that it has a gradient porosity across the peel, which has various impacts during compression or cyclic loading. A typical stress strain curve in which a pomelo peel sample is cyclically loaded through quasi-static compression is in Figure 6. By inspection of the figure, one can note the energy dissipation properties the peel seems to have through its hysteretic response, and is one of the responses that is investigated further in this work.

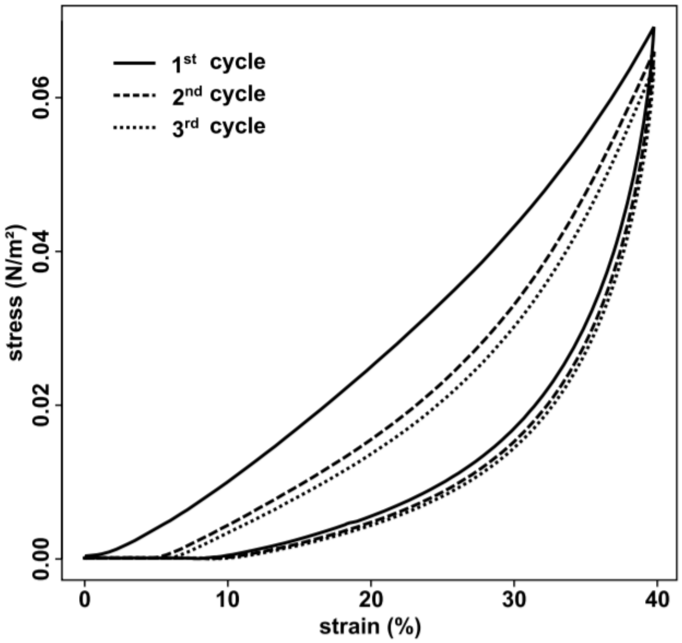


Figure 6. Stress-strain response for a pomelo peel, adapted from Thielen et al. [5]

It is hypothesized in this paper that the stress-strain curve depicted for the pomelo peel can be related to the stress strain curve for that of a typical open cell foam by comparing their nonlinear regions. Due to the change in relative density throughout different segments of the pomelo peel, there are portions of the peel, such as the middle, that have very little material available to resist compressive/tensile loads on the top or bottom of the peel. Due to this, there is no linear elastic regime that can be seen in the curve for the pomelo peel. Rather, it seems the mechanical behavior of the peel skips this regime entirely and picks up possibly where the plateau region is and exhibits a behavior known as the onset strain of densification (OSD). OSD is defined as when the cell walls begin interacting and contacting each other, essentially the ligaments and the foam in general get so compressed that all the empty space in the cell is removed and the ligaments begin compressing each other. Thielen et al. [5] showed that conventional methods could not be used in determining the beginning of the OSD and instead used a variation of the OSD defined as the maximum of the energy absorption efficiency. However, for the model presented in this paper, because there are regions in which there is a very small amount of material present to elastically absorb compression, it seems the material rather is immediately experiencing buckling and is forced to recruit larger and larger parts of the peel to deform until the entire peel is compressed. Another factor that may be influencing the behavior of the foam would be the cell lumen contained within the ligaments themselves. As compression occurs, these cells burst and lose some of their mechanical strength. For the purposes of this paper, the main aspects studied will be the gradient porosity along with the effect it has on computational models, and if it can be used to accurately assess the capabilities of a similarly manufactured foam.

On a macroscopic level, in compression testing the pomelo peel acts as a viscoelastic material, showing reactions to different percentage strains that are time dependent. As can be seen from Figure 7, a typical loading of a peel sample shows force relaxation on various levels of strain. For quasi-static loading, the modulus was found to vary linearly with respect to strain, and can be used to justify some results from a mathematical derivation, which is one of the results that this paper will be using.

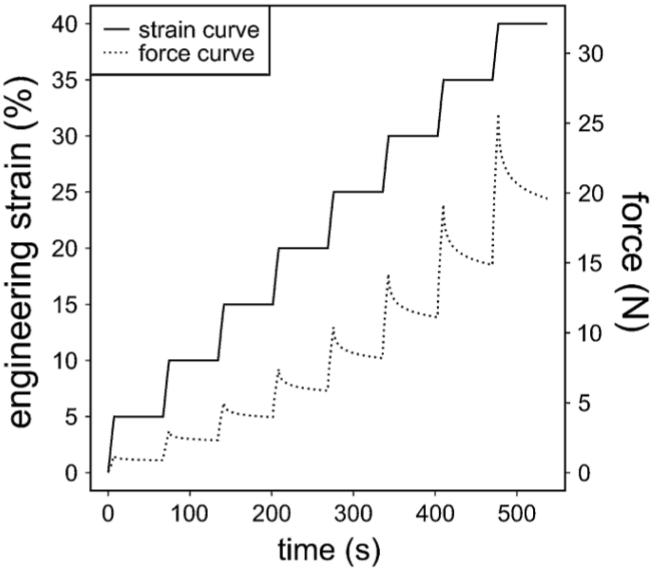


Figure 7. Force relaxation curve for pomelo peel, adapted from Thielen et al. [2]

A close look at the structure of the pomelo peel reveals some interesting aspects about the structure and how they interact with one another. As can be seen from Figure 4, the foam is compiled of a network of struts that vary in number according to the location within the peel. Randomly dispersed within this network, the vascular bundles are the localized areas that are denser then the surrounding region. This combination of struts and vascular bundles seems to have evolved out of a necessity of impact absorption, and has provided the basis of the analysis.

Seidel et al. [8] mapped the distribution of intercellular material, cell lumen, and cell wall material as a function of radial distance, which is shown in Figure 8, where the graph's origin (left end) starts at the exterior of the pulp and moves radially inward towards the pulp.

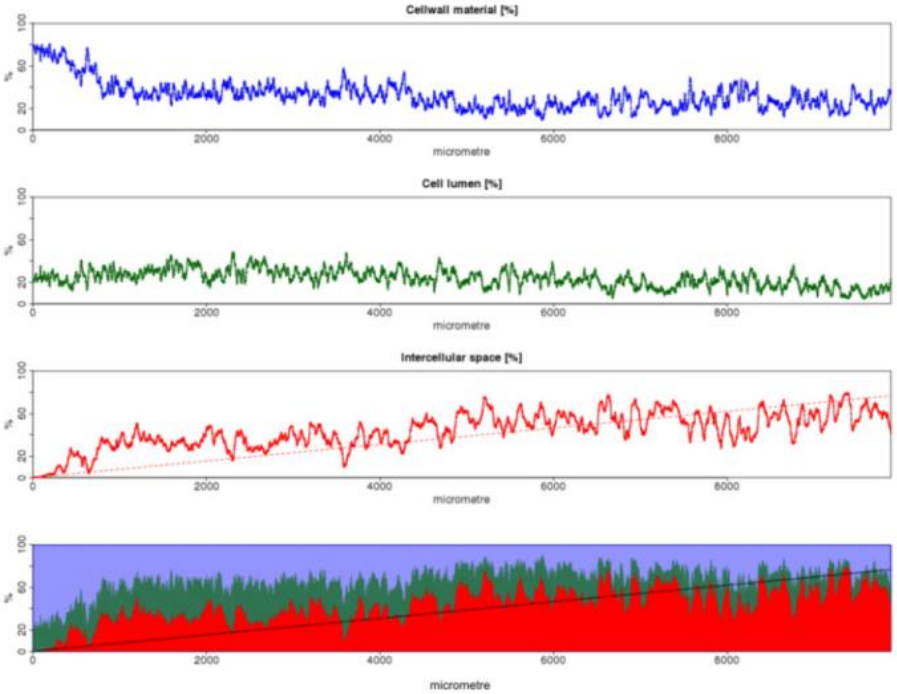


Figure 8. Distribution of cellular, lumen, and empty space within the pomelo peel, adapted from Seidel et al. [8]

Thielen et al. [5] also provided information on the dispersion of the vascular bundles as a function of the radial distance as well, shown in Figure 9. These pieces of information can combine to determine at any cross section within the peel: the amount of struts, the intercellular area, the lumen area, the cell wall area, and the area occupied by any randomly dispersed vascular bundles.

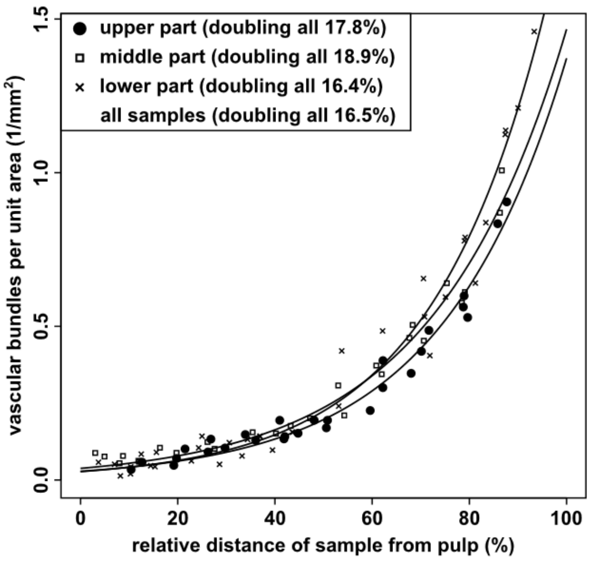


Figure 9. Vascular bundles vs radial distance, adapted from Thielen et al. [5]

When looking at the entire system in compression, it can be seen when the denser end is loaded with a uniform stress, the middle section is the first part of the system that is susceptible to plastic deformation, because it has a smaller effective area to absorb the stress due to the increase in intercellular space. The entire peel then, begins to fail beginning on the smallest area all the way up to the maximum area, where there is no intercellular space. Furthermore, on the cross sections that do not have much cross-sectional area to absorb the loading, the stresses will be even higher assuming that they remain planar. It should be mentioned, however, that that flavedo of the pomelo does have a slight curvature associated with it which would disrupt a uniform distribution of stress but assuming the flavedo to be planar is still a good approximation.

To understand at what points the foam will fail then, a cross sectional area can be represented as cutting through a serious of cylindrical vessels undergoing an axial compressive force. The struts have some various dimensions given by Fischer et al.[7], such as an average outer diameter of 38 micrometers, and an average inner diameter of 31 micrometers. By knowing the amount of intercellular space that should be present at any given cross-sectional area, a specified number of cylindrical vessels with average strut dimensions can be arranged at each interface to fit these physical constraints.

3. METHODOLOGY

3.1 Development of open-cell structure

For the work presented here, the gradient porosity and variation in foam density across the thickness of the peel will be replicated through a pseudo random Voronoi tessellation of space which a higher concentration of seed points at the boundaries of the computer model and a lower concentration towards the center. Since the geometry is being generated with a process that incorporates randomness, it will never produce two models that are exactly the same. Therefore, there will always be some level of variation when comparing two models or a model and some experimental values, as was seen in Figure 5.

To generate the skeletal open-cell structure, open source C++ libraries collectively known as Voro++ were used, with the full code used given in Appendix A. The algorithm utilizes a commonly used method known as a Voronoi three-dimensional tessellation of space to systematically create cells of various sizes within a given volume. The program starts by defining the volume that will be tessellated, or split, into different cells. For the purposes of this paper a cylindrical volume was used to benchmark results with published compaction data of the pomelo peel. A set number of points are then randomly placed into the volume, known as “seeds”, that will go on to form the pores (or cells) of the material. Using Figure 8 as a model, it can be seen that generally speaking there is more cellular material at the ends of the foam and less material in the middle of the foam, which is also an observation commonly made throughout published literature. To replicate the variation in pore size and empty space that is present in the pomelo peel, there are a high number of these random points placed near the top and bottom ends of the volume space followed by a smaller amount that are then dispersed within the center portion. As these seed points are randomly distributed within the material, the perpendicular bisector is calculated between that point and all the other points to ultimately form a cell around each seed point. The decision-making process is rather straight forward and essentially forms a cube around each seed while cutting away portions of it corresponding to the perpendicular bisecting planes calculated with nearby points. Eventually, the bisectors lie outside of the cell containing a seed, and the process moves on to another seed point. A preliminary visualization of the topology is then visualized in gnuplot, as shown in Figure 10.

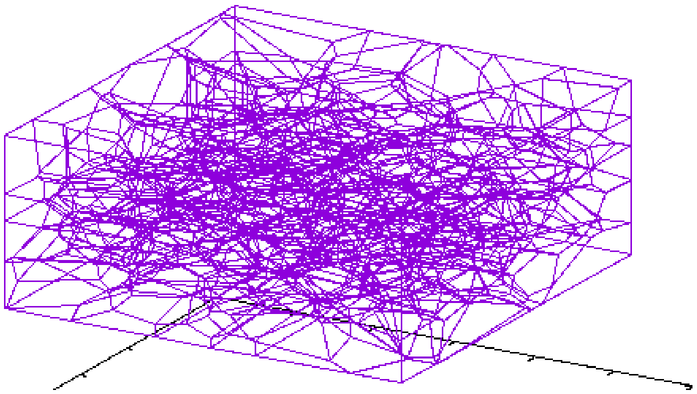
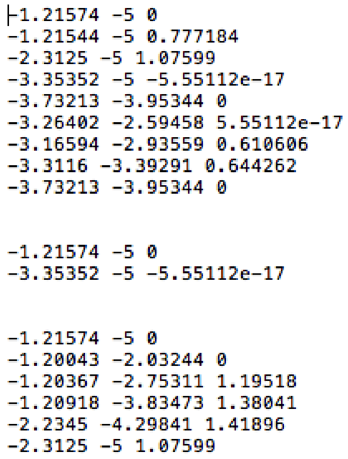


Figure 10. Gnuplot showing a 3D Voronoi tessellation of space using Voro++

3.2 Exporting structure to ABAQUS

Using the file contents of the gnuplot, the points used in the plot are then exported to ABAQUS, a commonly used finite element solver, and connected using 3D beam elements. To complete this, a python script was used that reads points from the gnuplot file, a portion of which is given in Figure 11, and connects them in sequence if there is no empty line separating the consecutive points to be connected.



```
-1.21574 -5 0
-1.21544 -5 0.777184
-2.3125 -5 1.07599
-3.35352 -5 -5.55112e-17
-3.73213 -3.95344 0
-3.26402 -2.59458 5.55112e-17
-3.16594 -2.93559 0.610606
-3.3116 -3.39291 0.644262
-3.73213 -3.95344 0

-1.21574 -5 0
-3.35352 -5 -5.55112e-17

-1.21574 -5 0
-1.20043 -2.03244 0
-1.20367 -2.75311 1.19518
-1.20918 -3.83473 1.38041
-2.2345 -4.29841 1.41896
-2.3125 -5 1.07599
```

Figure 11. Portion of a typical gnuplot's file contents

When the script encounters an empty line, this is gnuplot's way of saying the chain of points is to be broken, and the next point given is not meant to be connected to any previous points, rather only to the points that lie after it (until another empty line is encountered), and the process is repeated until it reaches the end of the text file. However, importing the beams into ABAQUS as geometric features was shown to take a prohibitive amount of time, mainly because each geometric entity must be rendered and added to the model which becomes an arduous task computationally once the model consists of thousands of beams. To remedy this, the input file for ABAQUS was written directly from the gnuplot's file contents, and the input file was imported into ABAQUS which produces something known as an orphan mesh. In short, in orphan mesh is simply a mesh that is not associated with any geometry but instead only has information on the nodal coordinates, element types, and element connectivity's. This has the benefit of greatly reducing the amount of time to import a model into ABAQUS, and once the input file has been written it be submitted and ran without ever opening the graphical user interface (GUI) in ABAQUS. The full python script used is given in Appendix B.

One of the rather subtle features of the python script is it removes duplicate beams that are present in the gnuplot format. Placing multiple wires in the same 3D space would prove to not only be detrimental to a finite element analysis, but also makes the already lengthy process of importing the cellular beam structure unnecessarily longer. The number of duplicate beams in the gnuplot file typically depends on the number of beams present in the model, and can generally vary from around 33% all the way up to around 60% as the number of beams increases.

The second feature the script has the ability of doing is removing wire segments that are below a tolerance value, which can be passed as an input parameter to the python script. As the number of beams in the model is increased, there are multiple beams generated that have either a length of 0, or some value very close to it. From the perspective of a finite element analysis, if beam segments of such short length were permitted to be present in the model it would be prohibitive to the stable time increment that an explicit analysis would be able to have and most certainly cause convergence issues during analysis. To remedy this, if the script encounters a beam segment below the given tolerance value it removes the beam and merges its endpoints, while making the appropriate adjustments to the edges connected to those endpoints. A further step is then taken to remove any more edges that may be present and are below the tolerance once the geometry is in ABAQUS, which is explained in the next section.

As a last step in importing and conditioning the geometry for ABAQUS, the mesh was queried and any beam segments that remained and were below a given tolerance value were extended until they were at or above the tolerance. For the specimens presented here, any beams that were below 200 micrometers were extended until they were at or above that length. This would then conclude the conditioning of the geometry and the mesh to ensure reasonable stable time increments and an overall tractable computational problem.

3.3 Finite element analysis using ABAQUS

Once the skeletal model has been successfully imported into ABAQUS, the compressive testing was completed by constraining the bottom face in all directions and rotations and displacing the top face using a rigid body in the shape of a simply plate to replicate a simple experimental quasi-static compression test. A variety of approaches and boundary conditions were taken throughout the modeling process, and much care must be taken in this step not only to ensure convergence but to be able to accurately reflect the materials real-world response. The units used in the model were cm for length, and other units were adjusted to remain consistent, and the initial dimensions chosen were used to match experimental samples that were prepared by Thielen et al. The beams were given a circular cross section with a radius of .002cm according to experimental studies that have characterized the topology of the pomelo peel. An example of the finished structure is shown in Figure 12.

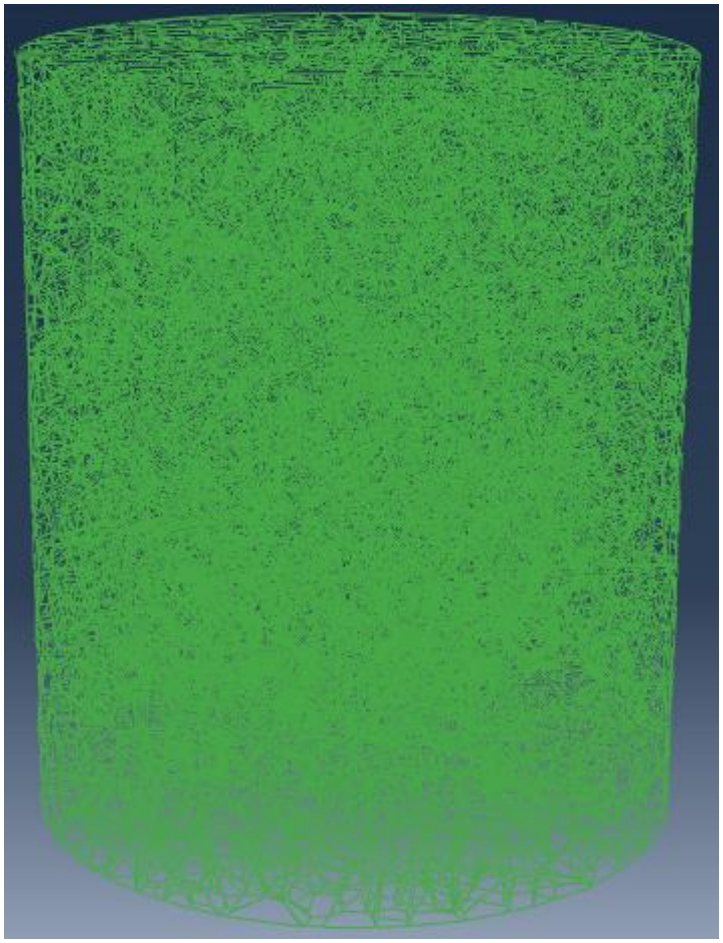


Figure 12. Open cell specimen with 83,771 beam elements with uniform porosity

To begin, the first analysis completed on the specimen was a quasi-static compression. To ensure the test was quasi-static in nature and did not incorporate many dynamic effects the displacement of the top face was defined using a 5th order polynomial that has zero slope at the beginning and the end of the time step to avoid sudden transitions in displacement or velocity, and is shown in Figure 13:

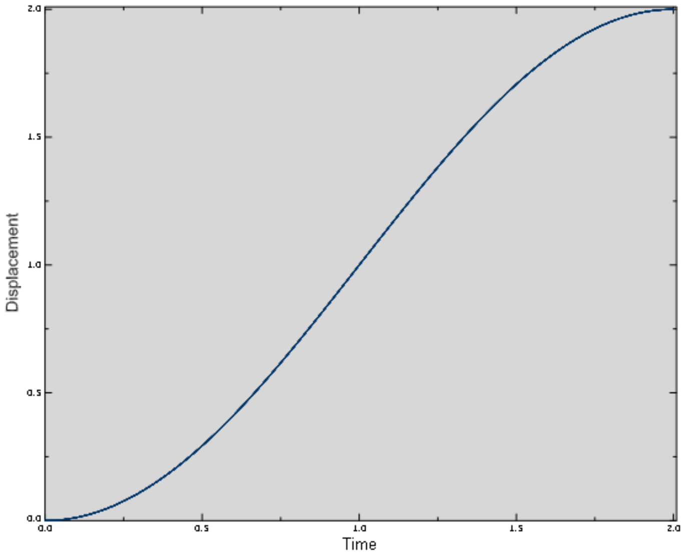


Figure 13. Visualization of displacement of top face vs time

To allow the finite element model to be tractable, there was the utilization of mass scaling on some of the elements that has the effect of increasing the stable time increment for an explicit finite element analysis. In an explicit analysis, the solution is obtained by integrating through time using many small-time increments that are dictated (approximately) by the ratio of the element length divided by the dilatational wave speed. Thus, for elements that have a short length compared to others, they are going to have the smallest transit time of the dilatational wave and therefore constrain the stable time increment for the entire model. To remedy this, the dilatational wave speed is inversely proportional to the density of the material and thus the density can be artificially increased in order to decrease the dilatational wave speed and increase the stable time increment. Explicit dynamic procedures involving quasi-static simulations tend to have very long simulation times compared to that of shorter dynamic responses such as impact or shock. Additionally, mass scaling is the preferred method of reducing the solution time if rate dependencies are included in the problem, which is a possibly for this project. When used correctly, mass scaling can often improve the efficiency of a finite element model by allowing it to run faster while still maintaining a degree of accuracy for the problem at hand. Like the portions of the import process that removed beams that were shorter than a certain length, beams that are above this tolerance but comparatively still very small tend to constrain the stable time increment of the entire model and can cause an analysis to take a prohibitively large amount of time. Mass scaling can be used uniformly over the entire model or only on elements that are below a target stable time increment, the latter being the method that was utilized in this paper. There was a lot of attention taken to ensure that the mass scaling was not significantly altering the solution of our model, with a primary indicator being a comparison of the internal strain energies to the kinetic energy of the model throughout the analysis. For the case of the quasi-static loading, a figure depicting a comparison of the internal strain energy and the kinetic energy throughout the two second compression cycle is provided in Figure 14.

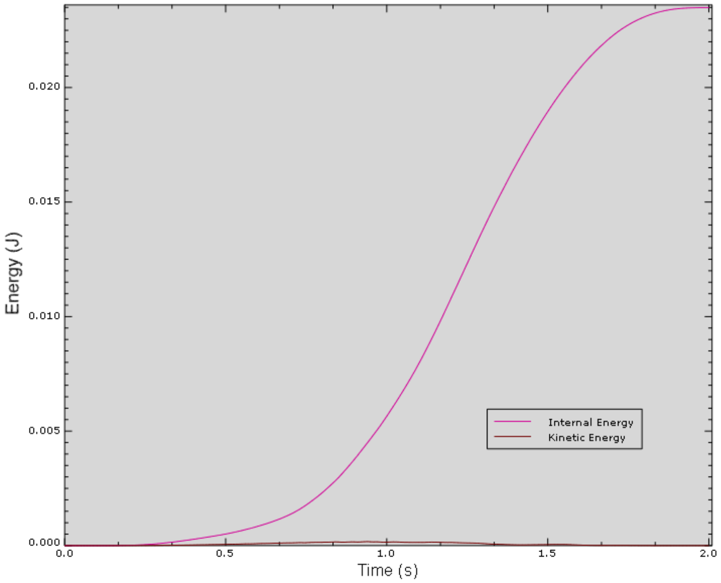


Figure 14. Comparison of internal energy and kinetic energy vs time

As can be seen, throughout the simulation the kinetic energy is a small fraction of the internal strain energies. Since the simulation is attempting to replicate a quasi-static scenario, this condition must be true for every simulation and throughout their entirety. The benefit of using the 5th order polynomial curve can then be seen when looking at this figure, since the kinetic energies are essentially zero at the beginning and end of the time step as result of the zero slope the polynomial curve has at these points in time. The peak kinetic energy is them achieved in the middle of the time step, but remained very low because of the how the loading curve gradually slopes upward as opposed to a ramp or step function that would surely have much more abrupt changes and would elicit a much more dynamic response from the structure.

4. RESULTS

4.1 Quasi-static compressions

Computational specimens were then loaded for a variety of time scales and their mechanical responses were post-processed. The loadings completed were primarily quasi-static in order to benchmark with published literature on the study of pomelo peel samples. Primarily, there were studies such as Thielen et al.[2] that excised a cylindrical specimen from the pomelo peel and tested it under quasi-static conditions. Interestingly, computational models can be seen to exhibit a similar response to experimental studies even without including rate dependent effects. This serves to help validate the hypothesis this paper had in that the compression responses of the peel seemed to simply be hysteretic cycles of a typical foam loaded up to and past densification.

To begin, first a uniform cylindrical specimen was generated and compressed to elicit a typical compressive response that is present in almost all cellular materials with a uniform pore distribution with which will then provide a baseline to compare other models to. The specimen has 8000 pores which resulted in 83,771 beams that were placed in a cylindrical volume with a 1.267 cm radius and 3 cm height and was shown in Figure 12. Eight frames of the compressive cycle are shown in Figure 15 along with the reaction force that was recorded throughout the analysis. Furthermore, a comparison of internal strain energies to kinetic strain energies is shown in Figure 16 and was used to ensure the quasi-static nature of the simulation.

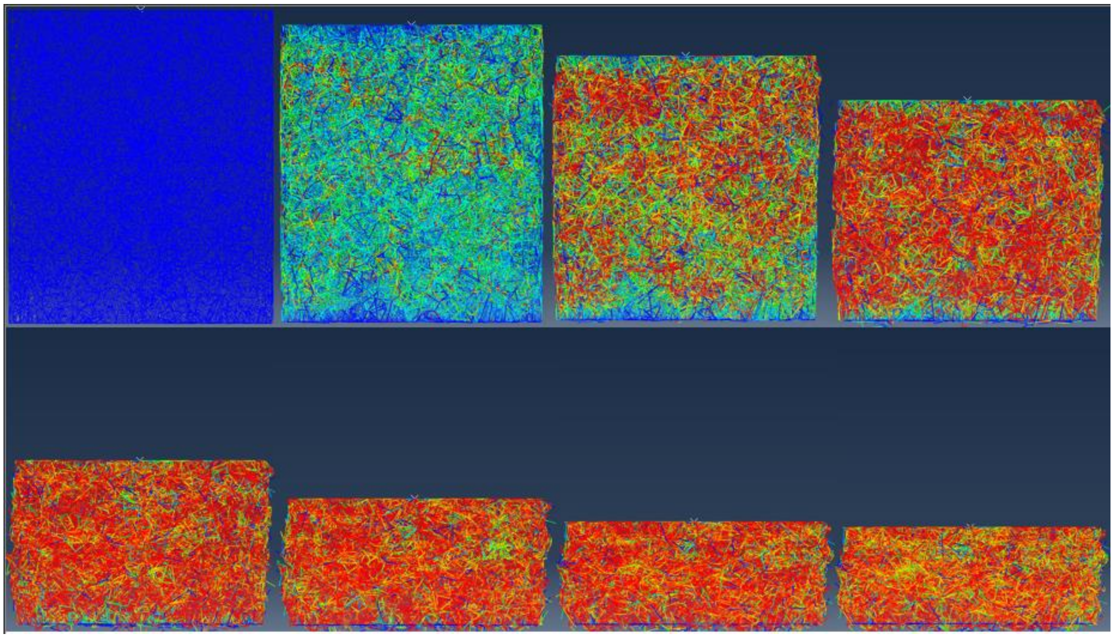


Figure 15. Compressive cycle for a uniform specimen with 83,771 beam elements

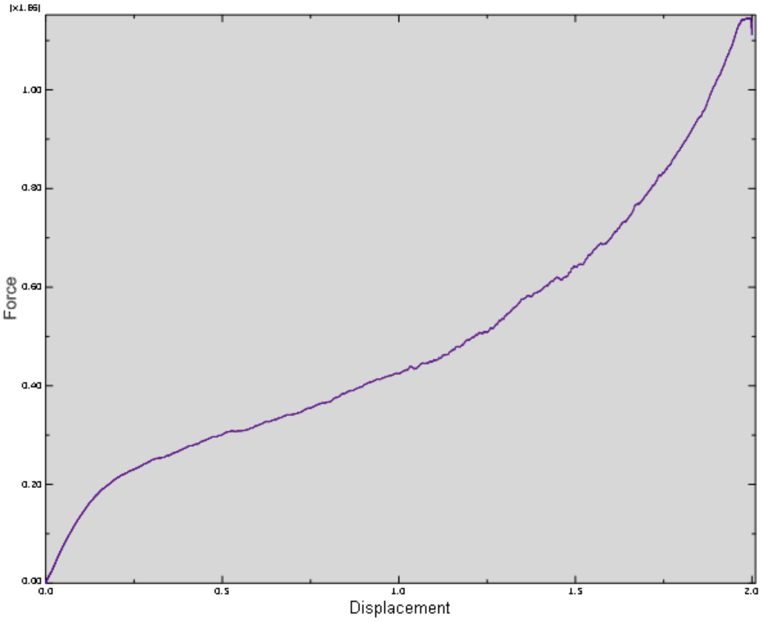


Figure 16. Reaction force for uniform specimen

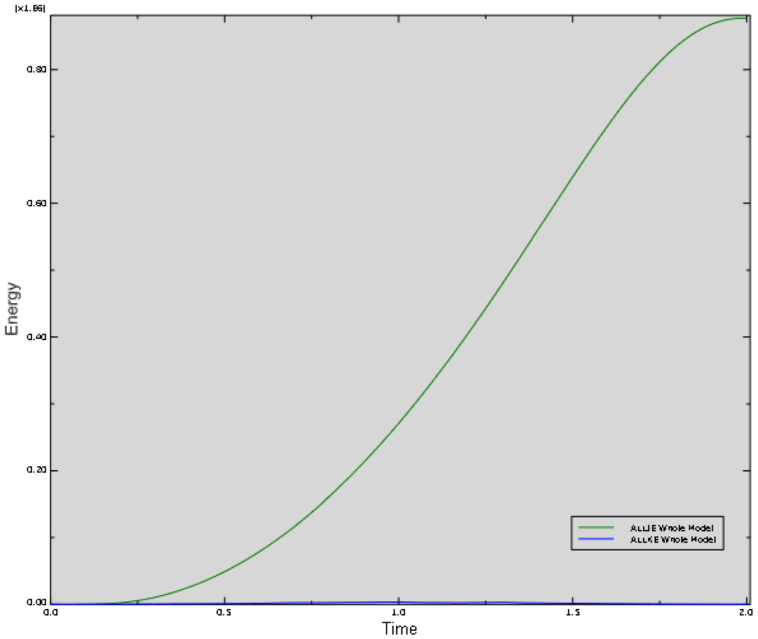


Figure 17. Internal strain energy and kinetic energy vs time for uniform specimen

As can be seen from the figures, the methodology does a good job of accurately modeling the response of typical cellular materials, while finding the optimal mass scaling point that allows the models to be tractable without deviating from a true quasi-static analysis, shown in Figure 17. The three typical regimes present in the compaction of cellular materials (elastic, plateau, and densification) can also be seen when studying the stress contours during the compaction cycle as well. Initially there are many beams that are incorporated in resisting the deformation, and then gradually more beams begin to plastically deform and fail while the beams begin to compact each other. The plateau regime may not be as noticeable in the samples presented here simply because of the number of beam present in the model.

The next sample, provided in Figure 18, is one that has the same number of pores as the uniform sample along with 81,438 beams but now 66% percent of the pores are dispersed within .6cm of the top and bottom faces of the sample to begin replicating the changing porosity present in the pomelo peel.

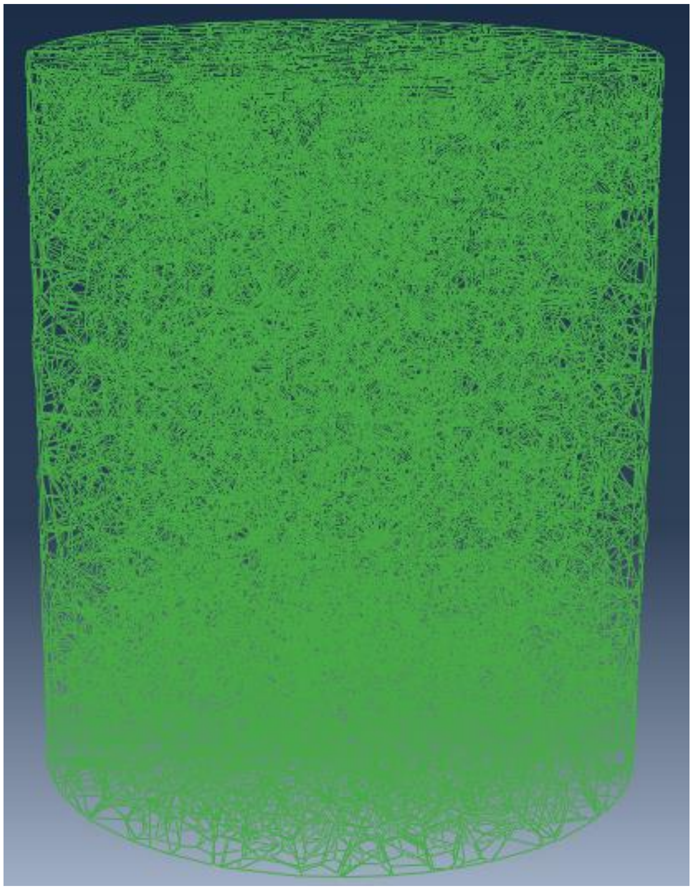


Figure 18. Open cell specimen with 81,438 beams and changing porosity

The analysis conducted on the specimen with the changing porosity mimics that of the uniform specimen, in which the top was displaced by 2 cm with a 5th other polynomial amplitude curve, mass scaling was used on prohibitive elements, and the results were post processed. Eight frames of the compression cycle are depicted in Figure 19 along with a comparison of the reaction forces for the two specimens generated thus far being depicted in Figure 20 (specimen with uniform porosity and specimen with changing porosity), with the uniform porosity specimen being depicted with the red line.

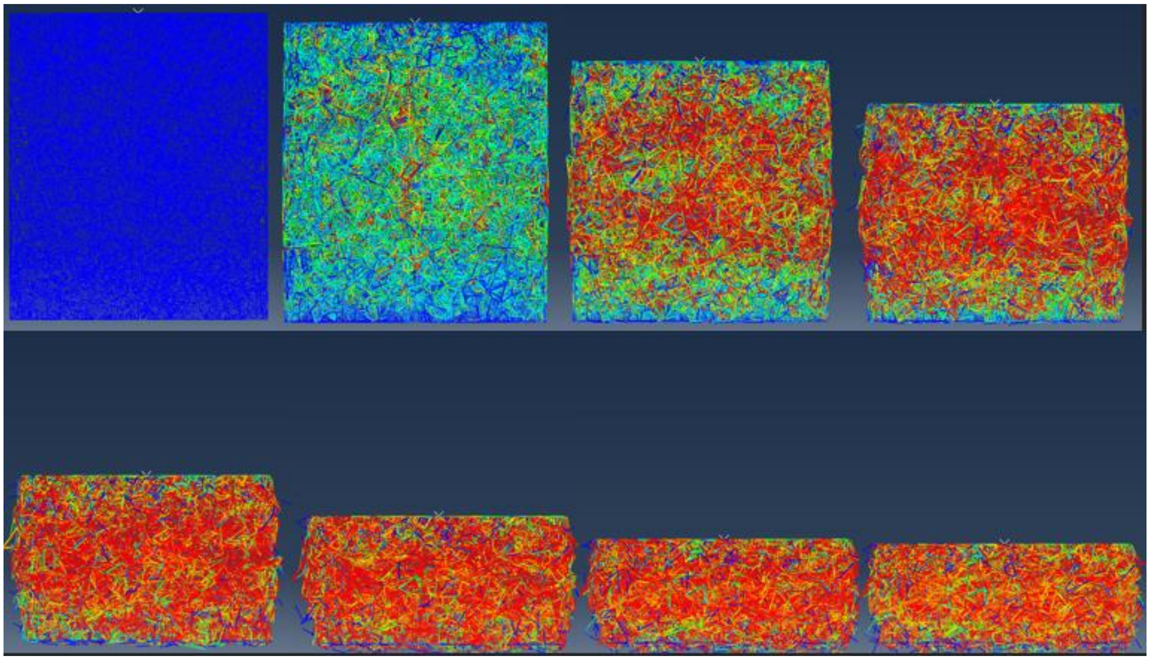


Figure 19. Compressive cycle for specimen with changing porosity

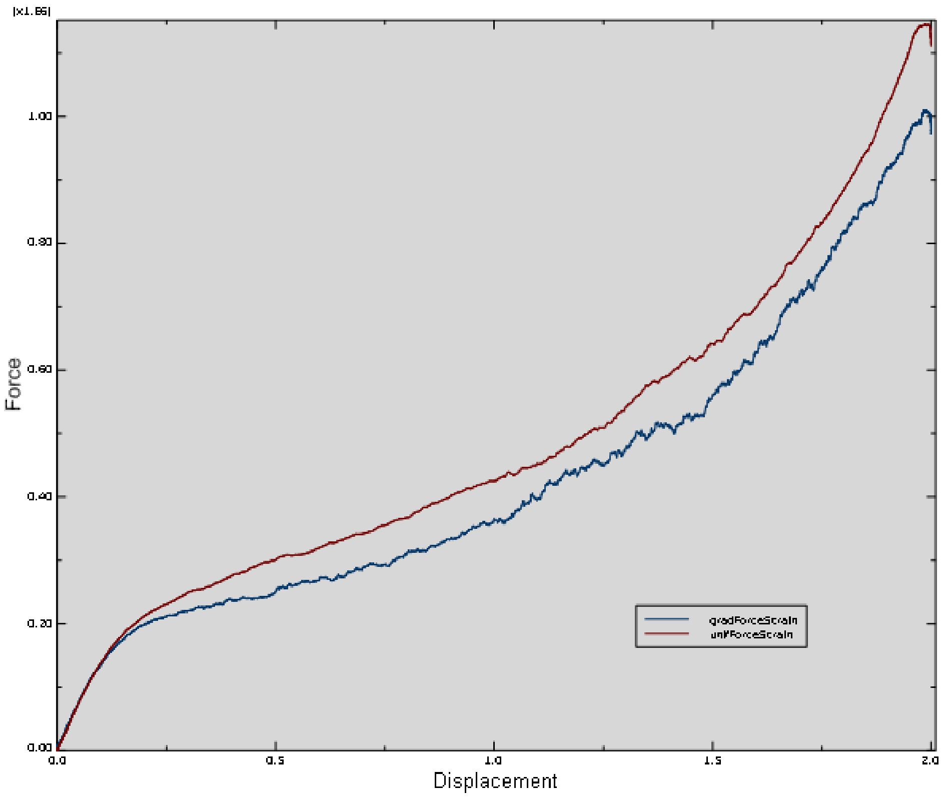


Figure 20. Reaction force of uniform and changing porosity specimens

To once again ensure the analysis was completed without dynamic effects being introduced the same comparison of the internal energies with the kinetic energies was conducted and was seen to look the same as Figure 17, and is therefore omitted for brevity.

Initially both compaction behaviors of the specimens with uniform and changing porosity seem very similar, but upon studying the magnitudes of the reaction forces some differences can then be realized. Although the specimen with a changing porosity has the same number of pores and roughly the same number of beam elements, it can produce a smaller reaction force when compared to the specimen with uniformly distributed pores which is depicted in Figure 20. Yet another way of interpreting how much energy has been absorbed by the structure is by a comparison of the strain energies throughout the analysis, which is shown in Figure 21. The uniform specimen is depicted in dark green and exhibits the expected result of being able to absorb more energy due it exhibiting a higher reaction force.

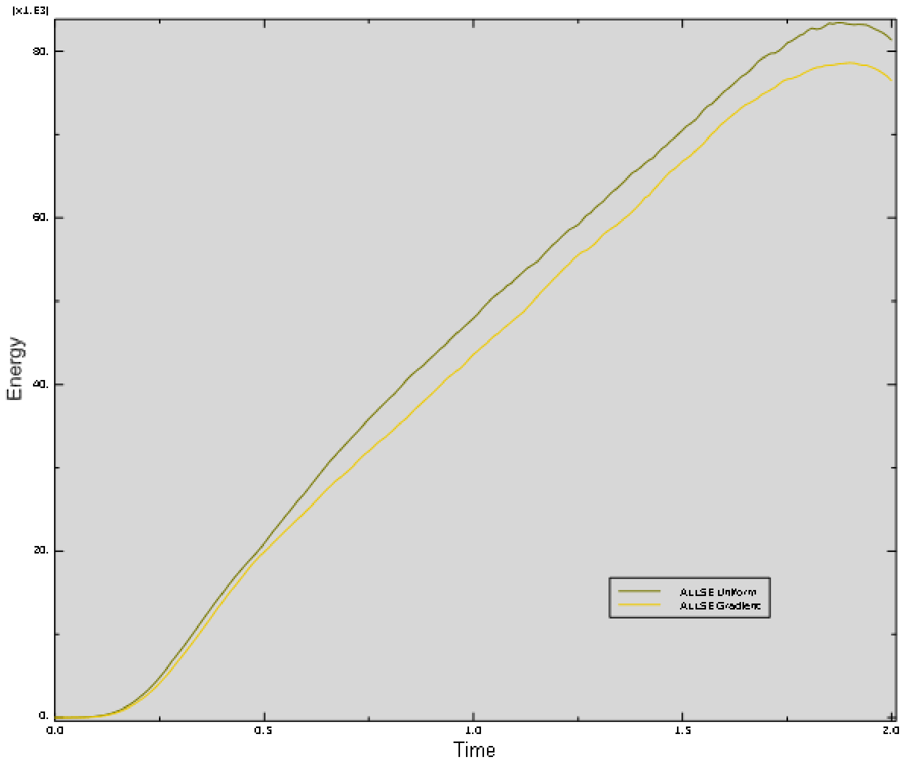


Figure 21. Strain energy comparison for the two specimens

To exaggerate the changing porosity, another specimen was modeled with 32,618 beams but a much larger shift in the porosity when compared to the previous model, and is shown in Figure 22 followed by 6 frames showing the compaction of the material using the same boundary conditions as before, shown in Figure 23.

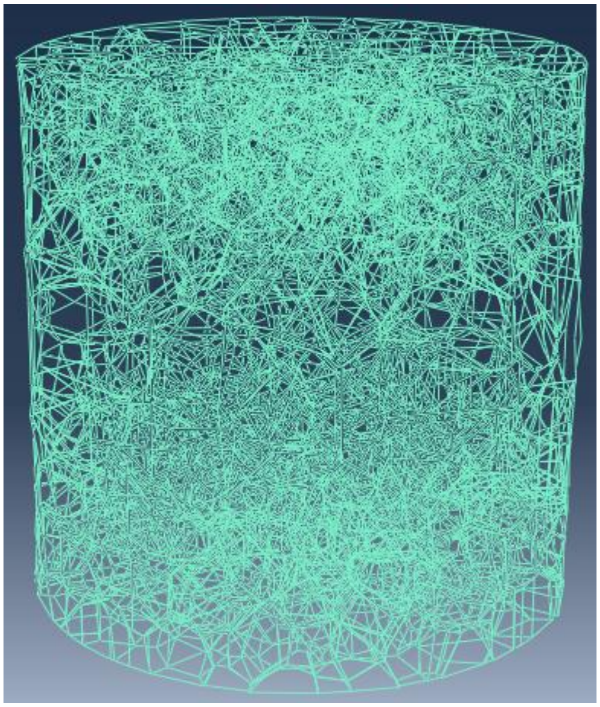


Figure 22. Specimen with changing porosity and 32,618 beams

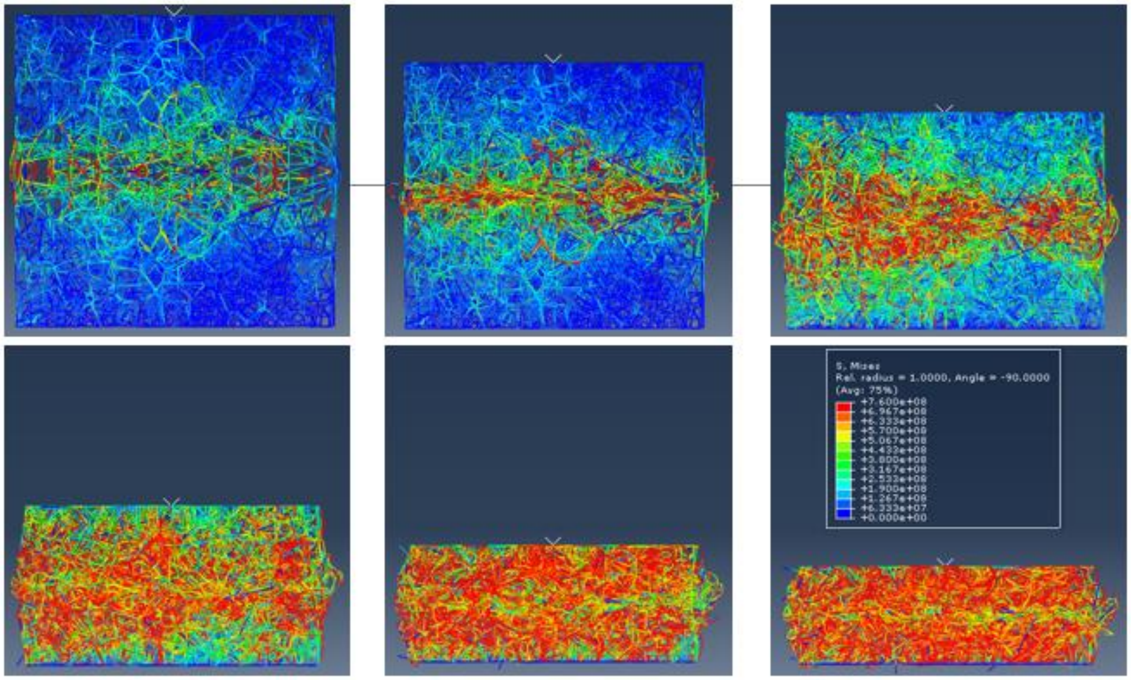


Figure 23. Compression of computational model with 32,618 beams

As was seen twice already, the kinetic energy compared to the internal strain energy of the model was seen to be of negligible value and the graph showing this was once again omitted for brevity. The effect of the changing porosity can then be seen much more clearly in this model. The first material to yield and undergo plastic deformation is located at the center of the specimen which has the least amount of "effective" cross sectional area to resist the loads applied. It therefore enters the plateau regime, which is exaggerated in this analysis because of the exaggeration in change in porosity of the model, and the beams seem to be almost in "free-fall" waiting to reach the densification regime. Upon reaching the densification regime, the reaction force then begins to rise again throughout the end of the analysis, as expected. As an added portion of the analysis, the specimen was then unloaded and loaded again to the same amount of strain the reaction force was tracked producing something that begins to look a lot like the compaction behavior of the pomelo peel, as is shown in Figure 24.

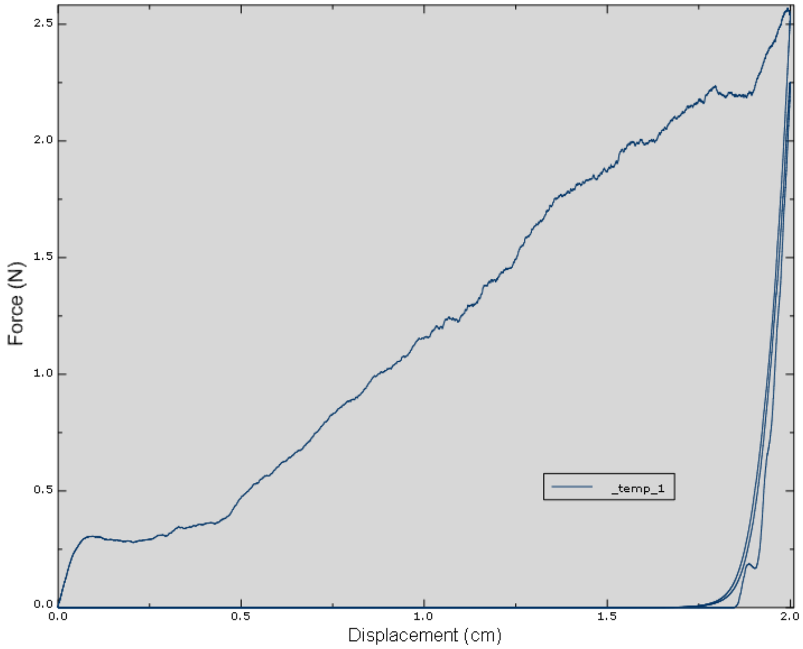


Figure 24. Compressive response of specimen with 32,618 beams

4.2 Impact analysis

Once the proper mass scaling was determined from the quasi-static analysis, that same amount of scaling was carried over to the dynamic analysis in which the only thing that was changed were the boundary conditions applied to the structure. The focus was to place the specimen in some real-world conditions subjecting to a drop from roughly 5 feet, and determining its deformation once it impacts the ground. To accomplish this, all of the elements in the specimen were given a predefined field value of 5.47 meters per second (which was converted to centimeters per second for the units to remain consistent in the model). The specimen was then placed 2.1 * 10^{-5} meters from a rigid plate that would be replicating the ground that the specimen would be impacting, with the plate being fully fixed in all displacements and rotations. Figure 25 depicts a specimen with 82,355 beam elements with the prescribed velocity shown with the orange arrows, and the floor in this case being the solid black line above the specimen.

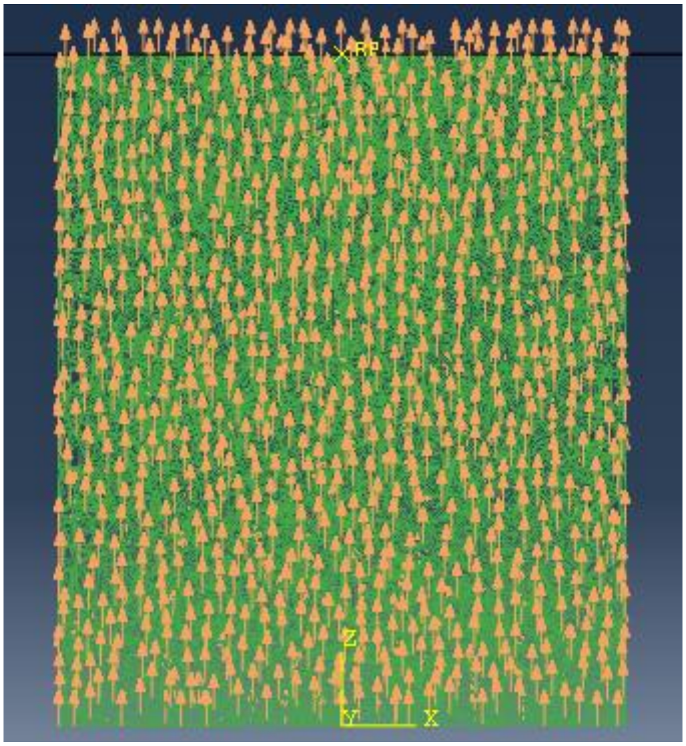


Figure 25. Boundary conditions for impact analysis of specimen with 82,335 beam elements and changing porosity

Due to impacts generally taking place over very short spans of time, the analysis for this case was not as computationally taxing as was the prior quasi-static analysis. Rather, the entire impact was seen to take place over roughly a mere 5.5 * 10^{-4} seconds, although attention does need to be paid to ensure the boundary conditions are properly defined and there is not excessive element distortion taking place. Figure 26 below captures the deformation and stress distributions that takes place during this time span.

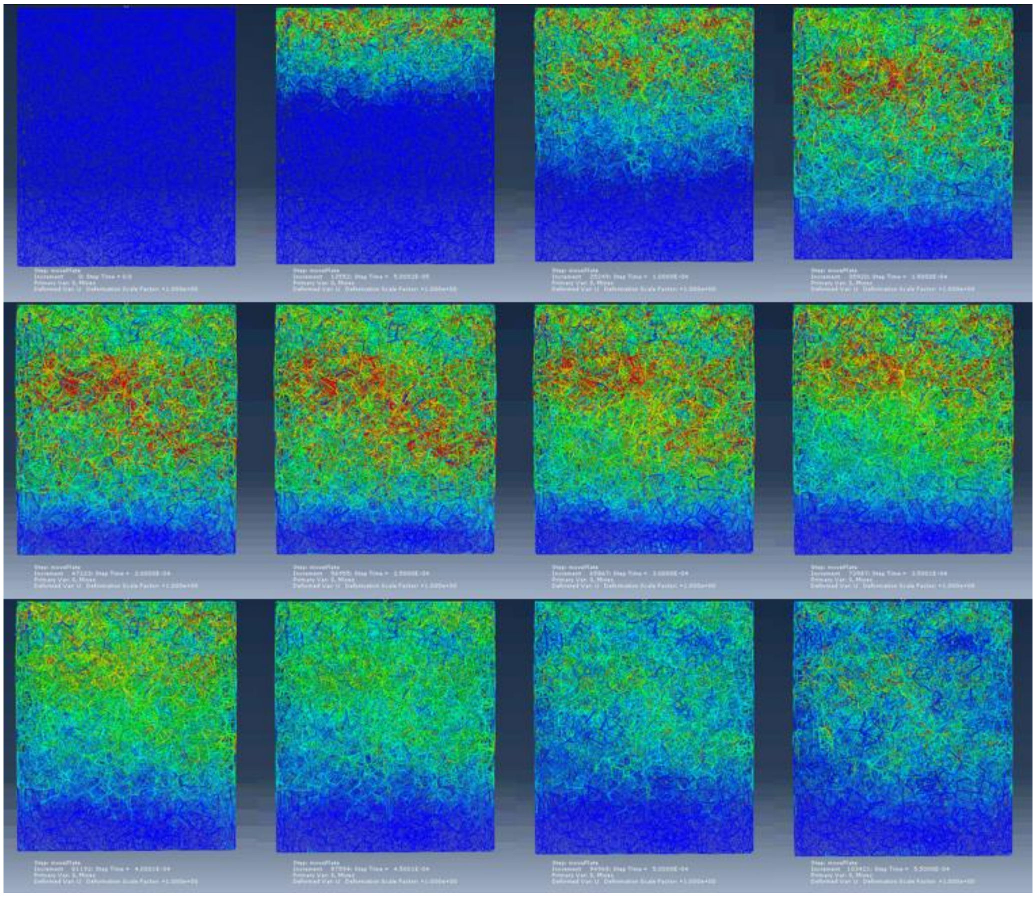


Figure 26. Stress distributions of specimen during impact (across 5.5 * 10^{-4} seconds)

As was done with the quasi-static analysis, a comparison of the strain energy and the kinetic energy of the structure was completed throughout the duration of the simulation in order to gain some more insight into how the structure is behaving and responding to the impact. The comparison is shown below in Figure 27, with an additional energy now included which is the plastic dissipation of the structure.

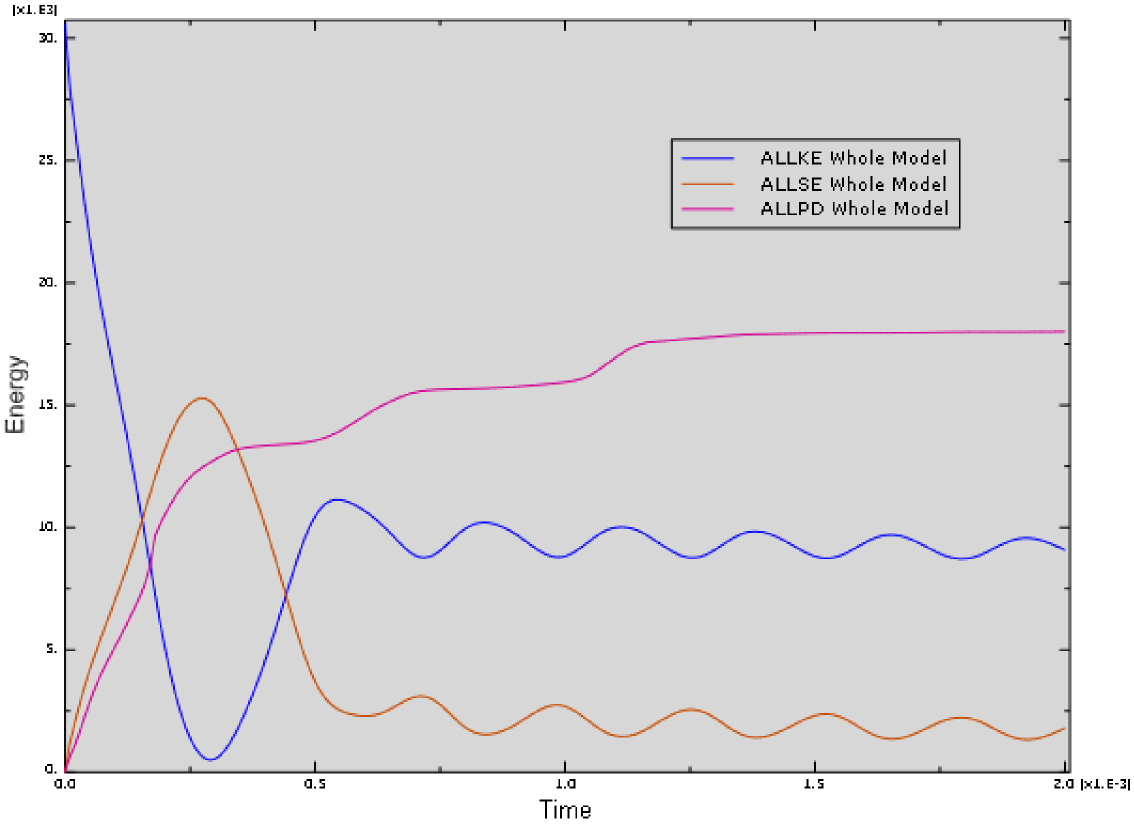


Figure 27. Comparison of energies during impact

5. CONCLUSIONS

It can then be seen that the changing porosity of the pomelo peel may play a more critical role in the mechanical response of the peel than was previously considered. Even without including a viscoelastic material model, a very similar trend can be seen to occur in which the first loading cycle dissipates a high amount of energy, and the second cycles then dissipates much less. This behavior can be attributed to a variety of reasons, some which being the plastic deformation of the structure during the loading cycle and another being the frictional dissipation between the beam elements during densification. A rather interesting aspect of these specimens is that they are almost immediately entering the densification stage, seemingly because of the limited free space that exists the pores. As mentioned previously, such a model containing hundreds if not thousands of unique contact patches would previously prove to be a rather difficult problem to solve with just the amount of computational resources required alone. However, utilizing Abaqus' beam to beam contact allows open cell foams to be modeled with relative ease compared to if one were to use three-dimensional solid elements or shell elements.

For the quasi static compressions, some rather interesting features were able to be exploited with a simple manipulation of the porosity density across the foam specimen, such as reducing the contact force along with reducing the strain energy that the structure absorbed although both the uniform and gradient specimens were under the application of the same boundary conditions. As for the cyclical loading, it was rather interesting to be able to replicate the response of the pomelo peel although the material properties were entirely different (Aluminum 6061-T6). The only location this difference in material properties can be denoted is at the very initial stages of the loading, in which there is a notable linear elastic regime that coincides with the elastic region of the aluminum material itself. This corresponds to typical compressions of cellular materials, in which the initial stages are the beam ligaments undergoing simple bending. It goes to say then, if the material properties were modified to replicate that of cellular material that constitutes the pomelo peel, this elastic regime would disappear and the loading cycle would then look even more similar to that of a pomelo peel.

The impact analyses were also interesting in the sense that they shed light on some very favorable characteristics of the gradient cellular material. When studying Figure 26, it was rather intriguing to see the shock from the impact propagate through the material but be arrested before reaching the other end (the bottom end). These results can be indicative of materials like this being favorable for applications involving high impact, shock, vibration, etc. Comparing Figure 26 with Figure 27, the kinetic energy seems to be turned into strain energy of the structure while some of it gets dissipated through plastic dissipation. After this initial impact stage, the strain energy then drops by some amount and seems to oscillate around some steady state value, while some of the strain energy is then converted back to kinetic energy as the specimen at the point is then bouncing off the ground and is no longer contact with the rigid plate. By inspection, the conversion of energies back and forth between strain and kinetic energies seems to ultimately dampen the impact that this structure has with the ground and, ideally, would also be serving the purpose of dampening the impact of whatever structure that resides on the other side of the foam specimen.

REFERENCES

1. Ashby, M.F. and R.M. Medalist, *The mechanical properties of cellular solids.* Metallurgical Transactions A, 1983. **14**(9): p. 1755-1769.

2. Thielen, M., T. Speck, and R. Seidel, *Viscoelasticity and compaction behaviour of the foam-like pomelo (Citrus maxima) peel.* Journal of Materials Science, 2013. **48**(9): p. 3469-3478.

3. Thielen, M., P. Schüler, and S.F. Fischer, *Fruit Walls and Nutshells as Inspiration for the Development of Novel Materials.* Laboratory Journal, 2013: p. 5.

4. Seidel, R., et al. *Impact resistance of hierarchically structured fruit walls and nut shells in view of biomimetic applications*. in *Proceedings of the 6th Plant Biomechanics Conference. French Guyana, France: ECOFOG, Cayenne*. 2009.

5. Thielen, M., et al., *Structure-function relationship of the foam-like pomelo peel (Citrus maxima)-an inspiration for the development of biomimetic damping materials with high energy dissipation.* Bioinspir Biomim, 2013. **8**(2): p. 025001.

6. Fischer, S.F., et al., *Pummelos as Concept Generators for Biomimetically Inspired Low Weight Structures with Excellent Damping Properties.* Advanced Engineering Materials, 2010. **12**(12): p. B658-B663.

7. Fischer, S.F., et al., *Production and properties of a precision-cast bio-inspired composite.* Journal of Materials Science, 2013. **49**(1): p. 43-51.

8. Seidel, R., et al., *Fruit walls and nut shells as an inspiration for the design of bio-inspired impact resistant hierarchically structured materials*, in *Design and Nature V*. 2010. p. 421-430.

9. Gaitanaros, S., S. Kyriakides, and A.M. Kraynik, *On the crushing response of random open-cell foams.* International Journal of Solids and Structures, 2012. **49**(19-20): p. 2733-2743.

APPENDIX A: C++ CODE FOR VORONOI TESSELLATION OF SPACE

```
#include "voro++.hh"
#include <iostream>
#include <fstream>
#include <sstream>
#include <math.h>
#include <algorithm>
using namespace voro;

// Set up constants for the container geometry
const double x_min=-.5,x_max=.5;
const double y_min=-.5,y_max=.5;
const double z_min=0,z_max=.5;
double R = .5;
double centerStart = .2*(z_max-z_min);
double centerLength = .6*(z_max-z_min);
double topStart = .8*(z_max-z_min);
double topLength = .2*(z_max-z_min);
double bottomLength = .2*(z_max-z_min);

/* Setting a base value  for the number of particles that are
 going to be randomly introduced into the different sections of foam*/
const int particles=1000;

// Function that returns a random double between 0 and 1
double rnd() {
   return ((double)rand()/(RAND_MAX));
}

// Set the computational grid size
const int n_x=7,n_y=7,n_z=14;

int main() {
   //Creating a set of 10 cylindrical tessellations
   for(int k = 0; k < 1; k++){
      int i;
      double x, y, z;
      // Create a container with the geometry given above, and make it
      // non-periodic in each of the three coordinates. Allocate space for
      // eight particles within each computational block.
      container con(x_min,x_max,y_min,y_max,z_min,z_max,n_x,n_y,n_z,
               false,false,false,8);
      //Seeding the random number function
      srand(time(NULL));

      // Add a cylindrical wall to the container
      wall_cylinder cyl(0,0,0,0,0,1,.5);
      con.add_wall(cyl);

      //Randomly add points to the foam
      for(i=0;i<particles;i++) {
         double a = int(10000*rnd())/10000.0;
         double b = int(10000*rnd())/10000.0;

         //If b is less then a, swap the two values
         if(b<a) {
            double temp = a;
            a = b;
            b = temp;
         }

         //Placing the particles into the container
         x=b*R*cos(2*M_PI*a/b);
         y=b*R*sin(2*M_PI*a/b);
         z=rnd()*4;
         con.put(i,x,y,z);
      }


      // Output the particle positions in gnuplot format
      std::string fileName = "uniformPractice_3";
      fileName += std::to_string(k);
      con.draw_cells_gnuplot(fileName.c_str());
   }
}
```

APPENDIX B: PYTHON SCRIPT TO IMPORT BEAMS INTO ABAQUS

```
from abaqus import*
from abaqusConstants import *
from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from optimization import *
from job import *
from sketch import *
from visualization import *
from connectorBehavior import *
import csv

import math

#Function that finds the vector between two points
def vector(arg1, arg2):
  connectionVect = [0,0,0]
  for j in range(3):
    connectionVect[j] = round(arg2[j] - arg1[j],3)
  return connectionVect

#Function that finds the negative vector between two points
def vectorNeg(arg1, arg2):
  connectionVect = [0,0,0]
  for j in range(3):
    connectionVect[j] = -1*(round(arg2[j] - arg1[j],3))
  return connectionVect

#Function converts set of coords from string to float and rounds to 3 after the decimal
def stringToFloat(arg1):
  threeFloats = arg1.split()
  for j in range(len(threeFloats)):
    threeFloats[j] = round(float(threeFloats[j]),3)
  return threeFloats

#Function that gets the magnitude of a vector
```

```
def mag(arg1):
  return math.sqrt(arg1[0]**2 + arg1[1]**2 + arg1[2]**2)

#Function that counts number of beams below tolerance and returns their
#index in beamLengths, which is the same for currentCoords and nextCoords
def shortCount(beamLengths,tolerance):
  shortBeamCounter = 0
  shortBeamIndex = []
  for i in range(len(beamLengths)):
    if beamLengths[i] <= tolerance:
      shortBeamCounter = shortBeamCounter + 1
      shortBeamIndex.append(i)
  return shortBeamCounter, shortBeamIndex

#Function that copies list for given range, not including the upper element
def copy(inputList, lower, upper):
  listCopy = [None]*(upper-lower)
  for i in range(0, (upper-lower)):
    listCopy[i] = inputList[lower+i]
  return listCopy

#Function that finds the midpoint of a line
def mid(arg1,arg2):
  copy1 = arg1
  copy2 = arg2
  midLocation = [0,0,0]
  for i in range(0,3):
    midLocation[i] = .5*(copy1[i]+copy2[i])
  return midLocation

#Function that simply prints the information for the beams
def beamInfo():
    print "Length of beamLengths:", len(beamLengths)
    print "Number of beams in model:", len(beams)
    print "Shortest beam in model:", min(beamLengths)
    print "Longest beam in model:", max(beamLengths)
    shortBeamCount, shortBeamLocation = shortCount(beamLengths,tolerance)
    print "Number of beams below tolerance: ", shortBeamCount, "\n"

#Function the builds all the beam arrays
def beamBuild(inputString):
    for i in range(len(myString)-1):
        #If the next line is not blank, store the two points in space to be used later to
        #create a beam element between them
        if myString[i+1] != "" and myString[i] != "":
         #Storing the current line and calling stringToFloat function
         currentCoord = stringToFloat(myString[i])

         #Storing the next line and calling stringToFloat function
         nextCoord = stringToFloat(myString[i+1])

         #Checking if this vector is already in beams, if not then append
         if vector(currentCoord,nextCoord) not in beams and vectorNeg(currentCoord,nextCoord) not
in beams:
            currentCoords.append(myString[i])
            nextCoords.append(myString[i+1])
            beams.append(vector(currentCoord,nextCoord))
            beamLengths.append(mag(vector(currentCoord,nextCoord)))

#Function that sweeps the short beams from a structure
def sweep(tolerance,currentCoords,nextCoords,beams,beamlengths):
   shortBeamCount, shortBeamLocation = shortCount(beamLengths,tolerance)
   for i in range(0,shortBeamCount):
      changeIndex = shortBeamLocation[i]
      oldCurrent = stringToFloat(currentCoords[changeIndex-i])
      oldNext = stringToFloat(nextCoords[changeIndex-i])
      midPoint = mid(oldCurrent,oldNext)
      del currentCoords[changeIndex-i], nextCoords[changeIndex-i], beamLengths[changeIndex-i],
beams[changeIndex-i]
      for j in range(0,len(currentCoords)):
         if currentCoords[j] == oldCurrent:
            currentCoords[j] = midPoint
         elif nextCoords[j] == oldNext:
            nextCoords[j] = midPoint

#Function the creates the node list and writes to an output file
#f is the input file that is passed to the function
def nodeList(f):
   skips = 0;
   for i in range(2*len(currentCoords)):
      if i < len(currentCoords):
         if currentCoords[i] not in nodes.values():
            nodes[i+1-skips] = currentCoords[i]
         else:
            skips = skips+1
      elif i >= len(currentCoords):
         if nextCoords[i-len(currentCoords)] not in nodes.values():
            nodes[i+1-skips] = nextCoords[i-len(currentCoords)]
         else:
            skips = skips+1
```

```
#   for i in range(len(nodes)):
#       nodes[i+1] = nodes[i+1].split(" ")
#       currentCoords[i] = currentCoords[i].split(" ")
#       nextCoords[i] = nextCoords[i].split(" ")

    #Writing node list to an output file to match the abaqus input file format
    for i in range(len(nodes)):
        temp = nodes[i+1].split(" ")
        f.write("      " + str(i+1) + ",  " + temp[0] + ",  " + temp[1] + ",  " + temp[2] + "\n")

        #Adding nodes to bot or top foam sets if they are at the top or bottom
        if temp[2] == bot:
            botFoamNodes.append(i+1)
        elif temp[2] == top:
            topFoamNodes.append(i+1)

#Builds the element connectivity from the currentCoords and nextCoords arrays
#using the node number built from nodeList
def elementConnectivity(f):
    for i in range(len(beams)):
        f.write(str(i+1) + ", ")
        f.write(str(nodes.keys()[nodes.values().index(currentCoords[i])]) + ", ")
        f.write(str(nodes.keys()[nodes.values().index(nextCoords[i])]) + "\n")

        #If the element being added is either at the top or the bottom, add to top or bottom element 
sets
        currentCoordsTemp = currentCoords[i].split(" ")
        nextCoordsTemp = nextCoords[i].split(" ")
        if currentCoordsTemp[2] == bot and nextCoordsTemp[2] == bot:
            botFoamElems.append(i+1)
        elif currentCoordsTemp[2] == top and nextCoordsTemp[2] == top:
            topFoamElems.append(i+1)
    return i+1

#Function that write set information
def writeSet(inputSet):
    for i in range(len(inputSet)):
        if i%12 == 0 and i != 0:
            f.write("  " + str(inputSet[i]) + "\n")
        elif i == len(inputSet)-1:
            f.write("  " + str(inputSet[i]) + "\n")
        else:
            f.write("  " + str(inputSet[i]) + ",")
inputFile1 = open('twoSandCyl_Thielen_100cells_10.txt', 'r')
```

```
#Read entire input file and store as an array of strings separated by new lines (\n)
myString = inputFile1.read().split('\n')

#Array that will track beam vectors and beam lengths
beams = []
beamLengths = []

#Arrays to track currentCoords and nextCoords
currentCoords = []
nextCoords = []

#Dictionary to track nodes and corresponding coordinates
nodes = {}
bot = '0'
top = '3'
topFoamNodes = []
botFoamNodes = []
topFoamElems = []
botFoamElems = []

#Tolerance to be used when sweeping the structure
tolerance = .02

#Building arrays for starting and ending coordinates, beam vectors, and beam lengths
beamBuild(myString)

#Sweeping the structure for beams below tolerance structure
sweep(tolerance, currentCoords, nextCoords, beams, beamLengths)

#input parameters for the input file generation
jobName = "practiceJob"
modelName = "practiceModel"
partName = "practicePart"

#Writing the input file
f = open('Thielen10grad.inp','w')
f.write("*Heading\n")
f.write("** Job name: " + jobName + " Model name: " + modelName + "\n")
f.write("** Generated by: Abaqus/CAE 6.14-1\n")
f.write("*Preprint, echo=NO, model=NO, history=NO, contact=NO\n")
f.write("**\n** PARTS\n**\n*Part, name=" + partName + "\n")
f.write("*Node\n")
#--------------------------------------------------------------------#
#Need to import all the nodes here
```

```
#-----------------------------------------------------------------------#
nodeList(f)

f.write("*Element, type=B31\n")
#-----------------------------------------------------------------------#
#Need to import all the beams here
#-----------------------------------------------------------------------#
numbElements = elementConnectivity(f)

#This just seems to be the last node and element in a set
f.write("*Nset, nset=_PickedSet28081, internal, generate\n")
f.write("     1,  " + str(len(nodes)) + ",      1\n")
f.write("*Elset, elset=_PickedSet28081, internal, generate\n")
f.write("     1,  " + str(numbElements) + ",      1\n")
f.write("** Section: beamSection  Profile: beamProfileSolid\n")
f.write("*Beam Section, elset=_PickedSet28081, material=Aluminum_cm,
temperature=GRADIENTS, section=CIRC\n")
f.write("0.002\n")

#Not sure what this is
f.write("0.5,0.5,-1.\n")
f.write("*End Part\n")
f.write("**  \n")
f.write("*Part, name=plate\n")
f.write("*End Part\n")
f.write("**  \n")
f.write("**\n")
f.write("** ASSEMBLY\n")
f.write("**\n")
f.write("*Assembly, name=Assembly\n")
f.write("**  \n")
f.write("*Instance, name=Model-1-Part-1, part=" + partName + "\n")
f.write("*End Instance\n")
f.write("**  \n")
f.write("*Instance, name=plate-1, part=plate\n")
f.write("         0.,           0.,       3.0021\n")
f.write("         0.,           0.,       3.0021,           1.,           0.,       3.0021,           90.\n")
f.write("*Node\n")
#Not sure what this is
f.write("      1,           0.,           0.,           0.\n")
f.write("*Nset, nset=plate-1-RefPt_, internal\n")
f.write("1, \n")
f.write("*Surface, type=CYLINDER, name=topSurf\n")
#Not sure what this is
f.write("START,          2.5,           0.\n")
```

```
f.write(" LINE,        -2.5,          0.\n")
f.write("*Rigid Body, ref node=plate-1-RefPt_, analytical surface=topSurf\n")
f.write("*End Instance\n")
f.write("**  \n")
f.write("*Nset, nset=botFoam, instance=Model-1-Part-1\n")
#---------------------------------------------------------------------------#
#Writing all the nodes that are in botFoamNodes
writeSet(botFoamNodes)
#---------------------------------------------------------------------------#

f.write("*Elset, elset=botFoam, instance=Model-1-Part-1\n")
#---------------------------------------------------------------------------#
#Writing all the elements that are in botFoamElems
writeSet(botFoamElems)
#---------------------------------------------------------------------------#

f.write("*Nset, nset=plateRP, instance=plate-1\n")
f.write(" 1,\n")

f.write("*Nset, nset=topFoam, instance=Model-1-Part-1\n")
#---------------------------------------------------------------------------#
writeSet(topFoamNodes)
#---------------------------------------------------------------------------#

f.write("*Elset, elset=topFoam, instance=Model-1-Part-1\n")
#---------------------------------------------------------------------------#
writeSet(topFoamElems)
#---------------------------------------------------------------------------#

f.write("*End Assembly\n")
f.write("*Amplitude, name=cyclic, time=TOTAL TIME, definition=PERIODIC\n")
f.write("1,         1.5708,            0.,           0.5\n")
f.write("         -0.5,            0.\n")
f.write("*Amplitude, name=quasi, definition=SMOOTH STEP\n")
f.write("           0.,            0.,            2.,            1.\n")
f.write("** \n")
f.write("** MATERIALS\n")
f.write("** \n")
f.write("*Material, name=Aluminum_cm\n")
f.write("*Density\n")
f.write(" 2.7,\n")
f.write("*Elastic\n")
f.write(" 7e+11, 0.33\n")
f.write("*Plastic\n")
f.write(" 7.6e+08,0.\n")
```

```
f.write("** \n")
f.write("** INTERACTION PROPERTIES\n")
f.write("** \n")
f.write("*Surface Interaction, name=Fric\n")
f.write("*Friction\n")
f.write(" 0.3,\n")
f.write("*Surface Behavior, pressure-overclosure=HARD\n")
f.write("** ----------------------------------------------------------------\n")
f.write("** \n")
f.write("** STEP: movePlate\n")
f.write("** \n")
f.write("*Step, name=movePlate, nlgeom=YES\n")
f.write("*Dynamic, Explicit\n")
f.write(", 2.\n")
f.write("*Bulk Viscosity\n")
f.write("0.06, 1.2\n")
f.write("** Mass Scaling: Semi-Automatic\n")
f.write("**               Whole Model\n")
f.write("*Fixed Mass Scaling, dt=1e-05, type=below min\n")
f.write("** \n")
f.write("** BOUNDARY CONDITIONS\n")
f.write("** \n")
f.write("** Name: fixedBottom Type: Symmetry/Antisymmetry/Encastre\n")
f.write("*Boundary\n")
f.write("botFoam, ENCASTRE\n")
f.write("** Name: movePlate Type: Displacement/Rotation\n")
f.write("*Boundary, amplitude=cyclic\n")
f.write("plateRP, 3, 3, -2.\n")
f.write("** Name: stablePlate Type: Displacement/Rotation\n")
f.write("*Boundary\n")
f.write("plateRP, 1, 1\n")
f.write("plateRP, 2, 2\n")
f.write("plateRP, 4, 4\n")
f.write("plateRP, 5, 5\n")
f.write("plateRP, 6, 6\n")
f.write("** \n")
f.write("** INTERACTIONS\n")
f.write("** \n")
f.write("** Interaction: genContact\n")
f.write("*Contact, op=NEW\n")
f.write("*Contact Inclusions, ALL EXTERIOR\n")
f.write("*Contact Property Assignment\n")
f.write(" ,  , Fric\n")
f.write("** \n")
f.write("** OUTPUT REQUESTS\n")
```

```
f.write("** \n")
f.write("*Restart, write, number interval=1, time marks=NO\n")
f.write("** \n")
f.write("** FIELD OUTPUT: F-Output-1\n")
f.write("** \n")
f.write("*Output, field, variable=PRESELECT\n")
f.write("** \n")
f.write("** HISTORY OUTPUT: H-Output-1\n")
f.write("** \n")
f.write("*Output, history, variable=PRESELECT\n")
f.write("** \n")
f.write("** HISTORY OUTPUT: reactionForce\n")
f.write("** \n")
f.write("*Output, history, time interval=0.0004\n")
f.write("*Node Output, nset=plateRP\n")
f.write("RF3, U3\n")
f.write("*End Step\n")

f.close()
```