



UNIVERSITY OF CALIFORNIA
LOS ANGELES

Computationally Solving Nonlinear Membranes with Plane Stress Condition

Author:
David VASKO

Professor:
William KLUG

August 19, 2015

Contents

1	Abstract	3
2	Theory	3
2.1	Membranes	3
2.2	Neo-Hookean Model	3
2.3	Deformation Gradient	4
2.4	Constitutive Laws	4
2.4.1	Adjusting neo-Hookean Terms to the Correct Frame	5
3	Incremental Displacement	5
4	Numerical Methods	5
4.1	Overview	5
4.2	Shape Functions	6
4.3	Quadrature	6
4.4	Three Point Differentiation	7
4.5	Newton-Rhapson	7
4.6	Meshing	8
4.7	Assembly	10
5	Results and Discussions	11
5.1	Overview	11
5.2	Single Element	11
5.2.1	Consistency	11
5.3	Whole Mesh	12
5.3.1	Consistency	12
5.4	Flat Plate under Equibiaxial Stretching	14
5.5	Flat Plate under Transverse Load	15
5.5.1	Linear Elements	15
5.5.2	Quadratic Elements	18
5.5.3	Flat Plate under Transverse Load: Review	21
5.6	Spherical Shell with Internal Pressure	22
5.6.1	Spherical Shell with Internal Pressure: Review	34
6	Conclusion	34
7	Source Code Listing	35
8	References	36

List of Figures

1	From Zienkiewicz and Taylor, Volume 1, Chapter 5: Gaussian quadrature rules for one and three point rules.	7
2	Sample Meshes	9
3	Reference and Random Configurations: Single Element	11
4	Elemental Consistency	12
5	Meshwise Consistency	13
6	Analytical vs Numerical Solution: Equibiaxial Stretching	14
7	Flat Plate Transverse Loading: Deformation Plot 1x	15
8	Flat Plate Transverse Loading: Deformation Plot 4x	16
9	Flat Plate Transverse Loading: Convergence of Linear Elements	17
10	Flat Plate Transverse Loading: Deformation Plot 1x	18
11	Flat Plate Transverse Loading: Deformation Plot 4x	19
12	Flat Plate Transverse Loading: Convergence of Quadratic Elements	20
13	Internal Pressure: 1x quadratic Mesh Quadratic Deformation Plot	22
14	Stretch Ratio: 1x Quadratic Mesh Deformation Plot	23
15	Internal Pressure: 2x Mesh Quadratic Deformation Plot	24
16	Stretch Ratio: 2x Mesh Quadratic Deformation	25
17	Internal Pressure: 3x quadratic Mesh Quadratic Deformation Plot	26
18	Stretch Ratio: 3x Quadratic Mesh Deformation Plot	27
19	Internal Pressure: 1x Linear Mesh Quadratic Deformation Plot	28
20	Stretch Ratio: 1x Linear Mesh Deformation Plot	29
21	Internal Pressure: 4x Linear Mesh Quadratic Deformation Plot	30
22	Stretch Ratio: 4x Linear Mesh Deformation Plot	31
23	Max Pressure: 2x Mesh Linear	32
24	Max Pressure: 2x Mesh quadratic	32
25	Exact vs Numerical Internal Pressure	33

1 Abstract

The purpose of this report is to outline the development of a nonlinear solver for membranes under the plane stress condition. Using the approximations of membranes and plane stress both simplify the complexity of the algorithm greatly, but still give accurate results for physical geometries that resemble the membrane approximation. The developed algorithm will be checked for consistency, as well as accuracy. Consistency will be checked by a three point differentiation scheme and accuracy will be checked by comparing numerical results to exact analytical ones. Geometries that will be simulated are flat plates and spherical shells, which both can be treated as membranes.

2 Theory

This section presents the definition of membranes, the basics of neo-hookean materials, how these material can be formulated mathematically with constitutive laws, and what plane stress means.

2.1 Membranes

For a finite element to be considered a membrane element it must satisfy the following:

- The thickness of the element is very small relative to length or width.
- The element observes the plane stress condition, meaning there is no stresses normal to the thickness.
- The element does not hold or transmit any moments

While these restrictions may seem unrealistic at first, there are tones of physical geometries that are accurately approximated by these conditions. For example a balloon, a the sheeting of a tent, or a flat plate. The two geometries analyzed in this report are flat plates and spherical shells

2.2 Neo-Hookean Model

The neo-Hookean hyperelastic model is a nonlinear relationship between stresses and strain similar to that of Hooke's law. It is often used to predict the nonlinear stress-strain relationship of materials that are subject to large deformations. While small deformations will behave almost linearly, the neo-Hookean model predicts a plateauing effect on stress. The equations for a neo-Hookean material are outlined below. First is the strain energy, which is a fundamental property in stress-strain analysis that is the foundation for how internal forces and stiffnesses are calculated. The strain energy can be expressed as

$$w(\mathbf{F}) = \frac{\lambda_0}{2} \ln^2(J) - \mu_0 \ln(J) + \frac{\mu_0}{2} (I_1 - 3)$$

where

$$\mathbf{C} = \mathbf{F}^T \mathbf{F}, \quad J = \det(\mathbf{F}), \quad I_1 = \text{tr}(\mathbf{C})$$

Where \mathbf{F} is the deformation gradient, μ_0 is the shear modulus, and λ_0 is the bulk modulus. The strain energy $w(\mathbf{F})$ can be differentiated with respect to the deformation gradient \mathbf{F} to arrive at the following equation for 1st Piola-Kirchoff Stress

$$P_{iJ} = (\lambda_0 \ln(J) - \mu_0) F_{Ji}^{-1} + \mu_0 F_{iJ}$$

and 1st Piola-Kirchoff Stress can further be differentiated with respect to \mathbf{F} to arrive at the Lagrangian tangent moduli for the material.

$$C_{iJkL} = \lambda_0 F_{Ji}^{-1} F_{Lk}^{-1} + \mu_0 \delta_{ik} \delta_{JL} - [\lambda_0 \ln(J) - \mu_0] F_{Jk}^{-1} F_{Li}^{-1}$$

Where δ is the Kronecker delta which is defined as

$$\delta_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

These are the basic equations that describe the material behavior, however most of the time deformation gradient is not constant over a mesh therefore these equations must be put into a different form. Additionally, in order to solve for displacements we need equations for force and stiffness, not stress and tangent moduli. Therefore we must do a rigorous variational derivation to get equations for force and stiffness.

2.3 Deformation Gradient

The first step in developing constitutive laws is defining a reference and deformed configuration. The reference configuration is your initial sample or mesh, and the deformed configuration is the current state of the sample or mesh. When there is no deformation the two configurations are the same. Once these configurations are defined the deformation gradient can be found by calculating the curvilinear tangent vectors.

$$\mathbf{F} = \mathbf{g}_i \otimes \mathbf{G}^i$$

Where \mathbf{g}_i are the deformed tangent vectors, and \mathbf{G}^i are the reference dual vectors. The tangent vectors are calculated as follows

$$\mathbf{g}_i = \frac{\partial \mathbf{r}(\theta_i)}{\partial \theta_i}$$

where $\mathbf{r}(\theta_i)$ is the position vector \mathbf{r} as a function of the curvilinear components θ^1 , θ^2 , θ^3 , and θ^i is just a specific curvilinear coordinate. This will be very easy to implement on a discretized mesh using shape functions. Shape functions will be introduced in the Numerical Methods section.

2.4 Constitutive Laws

After the rigorous derivation as shown in [1], the final equations for strain energy, internal force and stiffness are given below.

$$\begin{aligned} W &= \int_{\Omega_0} w dV \\ F_{ia}^{int} &= \frac{\partial W}{\partial x_{ia}} = \int_{\Omega_0} n_i^\alpha N_{a,\alpha} \sqrt{A} d\theta^1 d\theta^2 \\ K_{iakb} &= \frac{\partial F_{ia}^{int}}{\partial x_{kb}} = \int_{\Omega_0} \left[2\tilde{C}^{\alpha\beta\mu\nu} (\mathbf{a}_\beta \otimes \mathbf{a}_\nu)_{ik} + \tau^{\alpha\beta} \delta_\beta^\nu \delta_{ik} \right] N_{a,\alpha} N_{b,\mu} \sqrt{A} d\theta^1 d\theta^2 \end{aligned}$$

additionally F_{ia}^{ext} with a constant distributed load f_i will be needed

$$F_{ia}^{ext} = \int_{\Omega_0} f_i N_a \sqrt{A} d\theta^1 d\theta^2$$

where \mathbf{a} is the in plane curvilinear tangent vectors and \sqrt{A} is the ratio of the reference element compared to that of the standard isosceles parametric triangle. The term $\tilde{C}^{\alpha\beta\mu\nu}$ will be laid out in the next sub section as well as τ^{ij} .

2.4.1 Adjusting neo-Hookean Terms to the Correct Frame

The first step is take the tangent moduli from the neo-Hookean model C_{iJkL} and convert it to the adjusted lab frame moduli C_{IJKL} .

$$C_{IJKL} = \frac{1}{2} (F_{Ii}^{-1} F_{Kk}^{-1} C_{iJkL} - \delta_{ik} S_{JL})$$

where $\mathbf{S} = \mathbf{F}^{-1}\mathbf{P}$. Now that we have the moduli explicitly in the lab frame we need to transform it to the curvilinear frame C^{ijkl} as follows

$$C^{ijkl} = (G^i)_I (G^j)_J C_{IJKL} (G^k)_K (G^l)_L$$

And finally to look at the 2D material geometric stiffness

$$\tilde{C}^{\alpha\beta\mu\nu} = C^{\alpha\beta\mu\nu} - \frac{C^{\alpha\beta 33} C^{33\mu\nu}}{C^{3333}}$$

We also need the stresses in the 2D curvilinear frame. The first step for this is to convert \mathbf{P} into \mathbf{T}

$$\mathbf{T} = \mathbf{P} \mathbf{F}^T$$

and to convert it into the curvilinear frame we must do the same sort of transformation we did for the tangent moduli.

$$\tau^{ij} = (g^i)_I (T)_{IJ} (G^j)_J$$

And now these terms are ready to be plugged into the constitutive laws outlined above.

3 Incremental Displacement

The final most important step of the process is taking residual forces on the nodes and translating that into an incremental displacement update, \mathbf{u} . The equation for this is given as

$$\mathbf{K} \mathbf{u} = -\mathbf{r}$$

where \mathbf{r} is the residual force

$$\mathbf{r} = \mathbf{F}_{ia}^{int} - \mathbf{F}_{ia}^{ext}$$

Notice that this is linear and looks very similar to the original Hooks Law. To use this linear approximation the 'guessed' deformed configuration used to calculate \mathbf{K} and \mathbf{r} , must be sufficiently close to the true deformed configuration for the numerical method to converge. This fact is why you must do incremental loading in nonlinear solvers.

4 Numerical Methods

4.1 Overview

The simulations in the report were developed using MATLAB R2014A Student Version. The general process of the computational methods used is outlined by the following steps. First, a working model for individual linear and quadratic triangular elements was developed. This means that the constitutive laws were applied to a single element and the strain energy, internal force and stiffness were calculated for each node. The integration was done using the Gaussian quadrature rules.

Consistency was then checked in all the calculated values using a three point central difference scheme. Once a single element was consistent, the process of meshing began. I wrote my own meshing program which breaks surfaces down in to strips and makes triangles out of each strip while keeping track of a connectivity list. After a functional mesh, I iterate through each element of the mesh calculating its nodal forces and stiffnesses and add them to the global force and stiffness arrays. Consistency was then checked for the global mesh using another three point difference scheme for each mesh node. Once assemble is consistent, you can solve for the incremental displacement using residual force and stiffness. Finally when the residual force reaches a specified near zero tolerance, the deformation has converged.

4.2 Shape Functions

Shape functions are functions that interpolate solutions between known discrete values. A shape function takes in parametric coordinates (θ_1, θ_2) as inputs and returns and returns the value for the designed shape function and its derivative. If our parametric domain is the standard isosceles triangle, where the parametric coordinates are given as (r, s) , we have

$$0 \leq r \leq 1$$

$$0 \leq s \leq r$$

Shape functions are an integral part of finite analysis, and a good shape function possess several important qualities. The first of these qualities is that the shape functions themselves will sum to one at each point (r, s) . This is called a partition of unity:

$$\sum_a N_a(r, s) = 1$$

Similarly their derivatives must form a partition on nullity,

$$\sum_a N_{a,\alpha}(r, s) = 0$$

Another important quality is C^0 – *completeness* which simply means that the shape functions can interpolate a random polynomial exactly

$$p(\theta) = \sum_{a=1}^N p_a N_a(\theta)$$

where $p(\theta)$ is the exact solution and p_a is the polynomial evaluated at node a , and N_a is the shape function associated with that node.

4.3 Quadrature

Another important concept in Finite Element Analysis is quadrature. The Gaussian quadrature rules shown in the following figure give the positions and weights of the points needed to exactly integrate over an element.

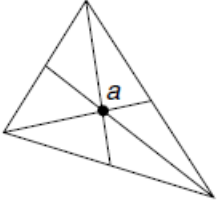
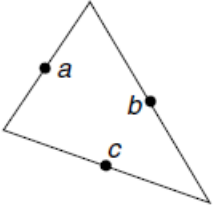
Order	Figure	Error	Points	Triangular coordinates	Weights
Linear		$R = O(h^2)$	a	$\frac{1}{3}, \frac{1}{3}, \frac{1}{3}$	1
Quadratic		$R = O(h^3)$	a b c	$\frac{1}{2}, \frac{1}{2}, 0$ $\frac{1}{2}, 0, \frac{1}{2}$ $0, \frac{1}{2}, \frac{1}{2}$	$\frac{1}{3}$ $\frac{1}{3}$ $\frac{1}{3}$

Figure 1: From Zienkiewicz and Taylor, Volume 1, Chapter 5: Gaussian quadrature rules for one and three point rules.

This is an awesome fact that with just a few discrete points you can exactly integrate over a continuous element. The more points used the equation order that is integrable over is higher.

4.4 Three Point Differentiation

The formula for the general 3-point difference scheme is shown below.

$$f'(a) \approx \frac{f(a+h) - f(a-h)}{2h} \approx f'_h(a)$$

where the error is given by,

$$\text{Error} = f'_h(a) - f'(a) = O(h^2)$$

When this scheme is applied to matrices such as internal force, each node must be perturbed each direction in order to generate the numerical approximation.

4.5 Newton-Rhapson

The simplest computational method for solving implicit nonlinear equations is using the Newton-Rhapson method. This method solves for the zeros of a function by repeated linearization. It is incredibly accurate given that the implicit function is smooth near zero, and the initial guess is "close" to the true value. Where "close" depends on the function itself. Newton-Rhapson is used extensively in my finite element solver so the steps for how it works are important to know.

1. Define a function $f(x)$ such that it equals zero.
2. Compute the derivative of $f(x)$ with respect to x .
3. Use the Newton update equation to iteratively solve for the zeros of the function. By setting a tolerance for convergence you can get within machine precision of the true value. The Newton update equation is shown below.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

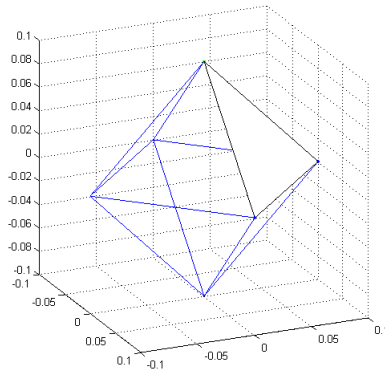
Where the subscript n just represents what incremental step it is in.

This process is mainly used for enforcing the plane stress condition by changing the out of plane stretch λ and driving the out of plane stress, T^{33} term, to go to zero. Written out this is

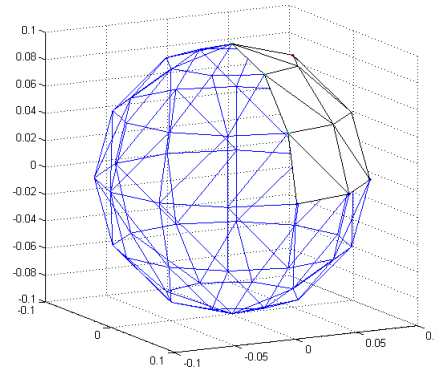
$$\lambda_{n+1} = \lambda_n - \frac{\tau^{33}}{2\lambda_n C^{3333}}$$

4.6 Meshing

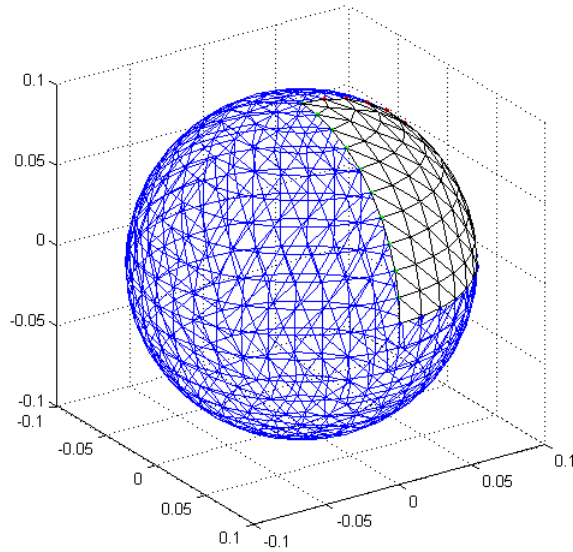
The meshing process starts by dividing the surface to be meshed into several equal thickness strips. Then each strip is individually divided into triangles. Its a fairly simple process, however the only complication comes in when successive rows have a different number of elements. If this happens you can no longer simply alternate between nodes of adjacent rows to create triangles; you must compensate for the difference in strip size by periodically using multiple nodes on the larger strip. Also keeping tract of connectivity is difficult but is of the utmost importance. A connectivity list is a list whose length is equal to the number of elements in a mesh. Each element of the connectivity list tells what nodes of the global node list make up a particular element. This is crucial for the assembly phase. The following figure will show three meshes for a spherical shell with increasing mesh resolution.



(a) 1x Mesh - 1 element



(b) 3x Mesh - 13 elements



(c) 10x Mesh - 130 elements

Figure 2: This figure shows a 1x, 3x and 10x mesh for a spherical shell. While the 1x mesh may not look exactly like a sphere its advantage is that it has very few elements and can be computed very fast. The 10x mesh, while it looks pretty is not worth the extra computational time. A 3x mesh is probably the best compromise.

Note that most of the mesh is shown in blue wireframe, but a 1/8 portion of the mesh is shown as white patches with black edges. This is because a sphere is eight way symmetric, therefore if you know what happens to this 1/8 portion under loads, you now what happens to the whole sphere. Always take advantage of symmetry when you can in computational problems because it increases the density of elements you can solve for. Every simulation in this report utilizes some facet of symmetry to simplify the model.

4.7 Assembly

Once a successful mesh has been generated, with a valid connectivity list, assembly is simple. You iterate through the connectivity list, pulling out each element individually as you go, calculating the forces and stiffnesses. Before you move on to the next element you update the global forces array and stiffness array using the connectivity which tells you exactly how node a of the current triangle maps to node A of the mesh. Shown mathematically for stiffness,

$$k_{iAkB}^m = k_{iAkB}^m + k_{iakb}^e$$

where k^m stands for the global stiffness matrix (m for mesh) and k^e stands for the elemental stiffness matrix. The relationship between A and a is given by the connectivity list. If the connectivity list is given a function name $TRI(n)$ you would determine A from a like follows

$$TRI(a) = A;$$

5 Results and Discussions

5.1 Overview

The results section will open by presenting solutions for single element cases before moving on to meshes, and ultimately applied problems. The nonlinear finite element solver developed in this paper is consistent accurate, but due to numerous nested for loops and while loops, tends to be rather slow. That being said it is still a very valuable resource in designing geometries that will encounter large deformations.

5.2 Single Element

A single element can be linear or quadratic and can deform in three principle directions at any of its nodes. Below is a figure showing both a linear and a quadratic element with the tangent bases superimposed.

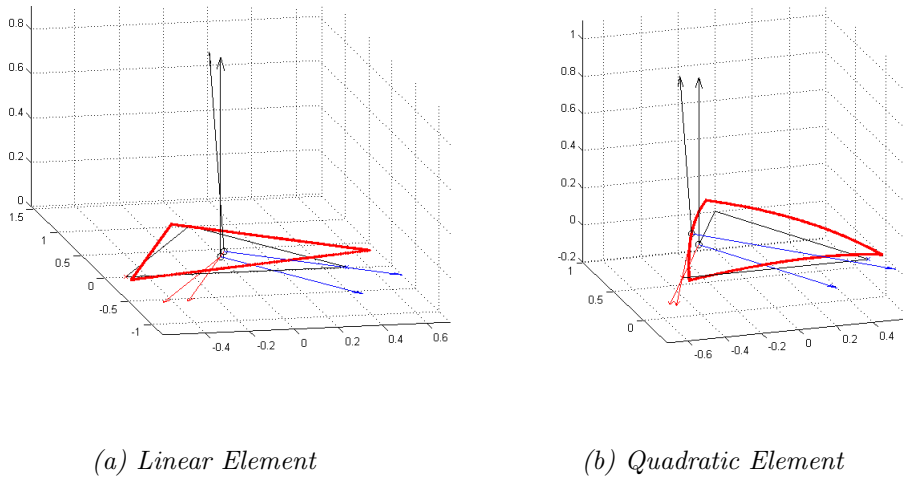
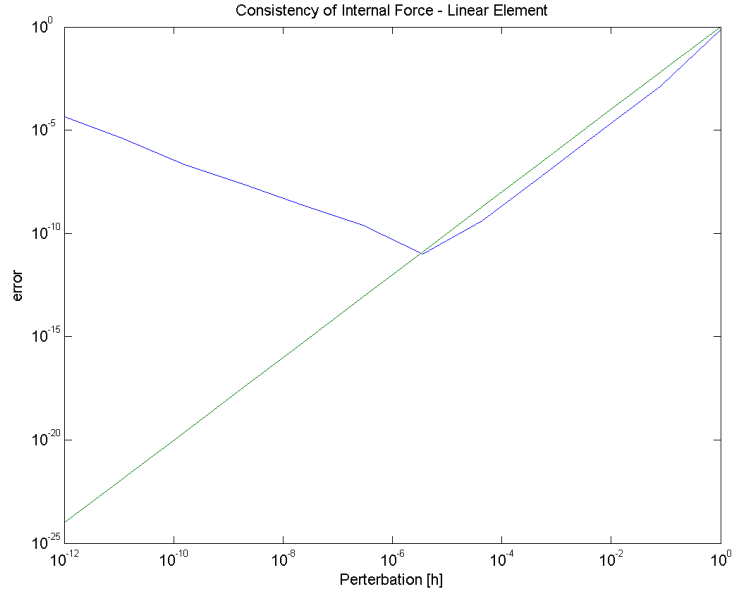


Figure 3: This figure shows the random deformation in three directions of (a) a single linear element and (b) a single quadratic element. A set of tangent bases is shown for one gauss point

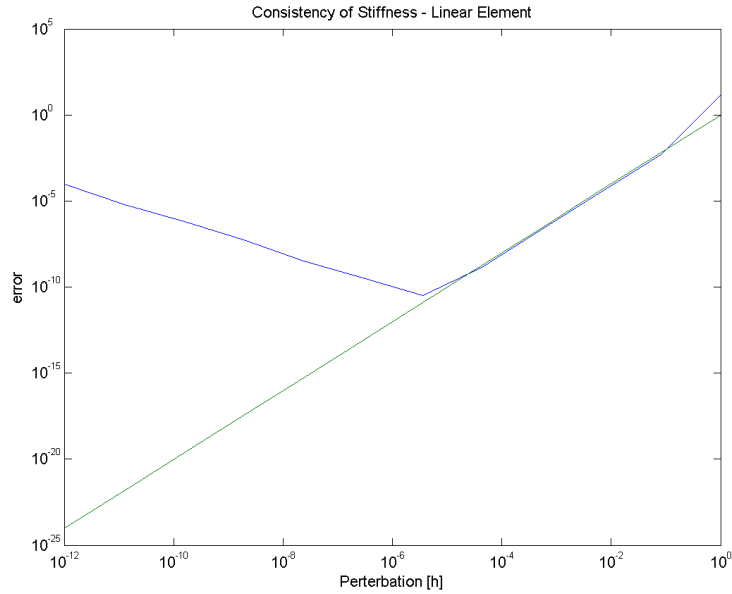
This figure is a good self check to makes sure things are working correctly on the element level. You can see the stiffness of linear elements, and the flexibility of quadratic ones. By looking specifically at the basis you can tell that they are tangent to the shape functions at their quadrature point, as they should be.

5.2.1 Consistency

Running the constitutive laws on a single element both analytically and computationally yeilds the following relationship between error and perturbation value.



(a)



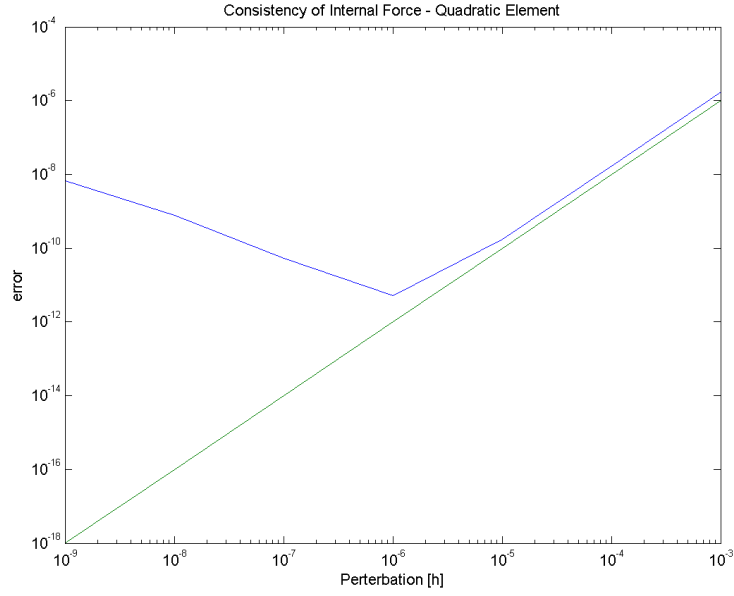
(b)

Figure 4: This figure shows the consistency of both internal force and stiffness for a linear element. The green line is $O(h^2)$ and the blue line is the error between numerical and analytical forms. I roughly captures the $O(h^2)$ trend. Quadratic elements are also consistent.

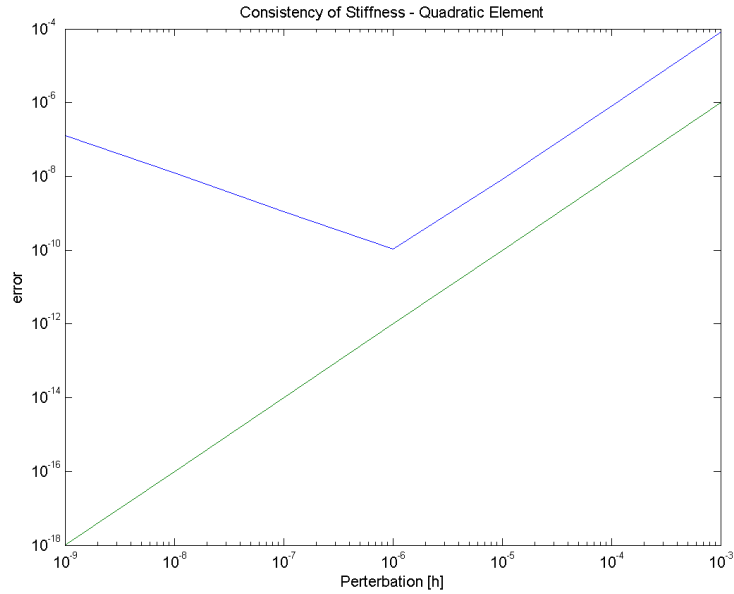
5.3 Whole Mesh

5.3.1 Consistency

Running the constitutive laws on the complete mesh analytically and computationally yields the following relationship between error and perturbation value.



(a)



(b)

Figure 5: This figure shows the consistency of both internal force and stiffness for a mesh made up of quadratic elements. The green line is $O(h^2)$ and the blue line is the error between numerical and analytical forms. I roughly captures the $O(h^2)$ trend. Meshes made of linear elements are also consistent.

Note that the relationship is also $O(h^2)$ Therefore the implementation for assembly is consistent. The minimum error occurs between 10^{-5} and 10^{-6} . With a smaller perturbation value than this rounding errors become important.

5.4 Flat Plate under Equibiaxial Stretching

When applying uniform stretching ratio to the two principle directions of a flat plate, the result is a constant deformation gradient. The stresses that arise can be computed from a converged mesh with prescribed edge displacements or directly using the original neo-Hookean hyperelastic model. Since both methods are derived from the same model, the results should be consistent. Additionally, every element in the mesh must have the same deformation gradient in order for the entire plate to have a constant deformation gradient.

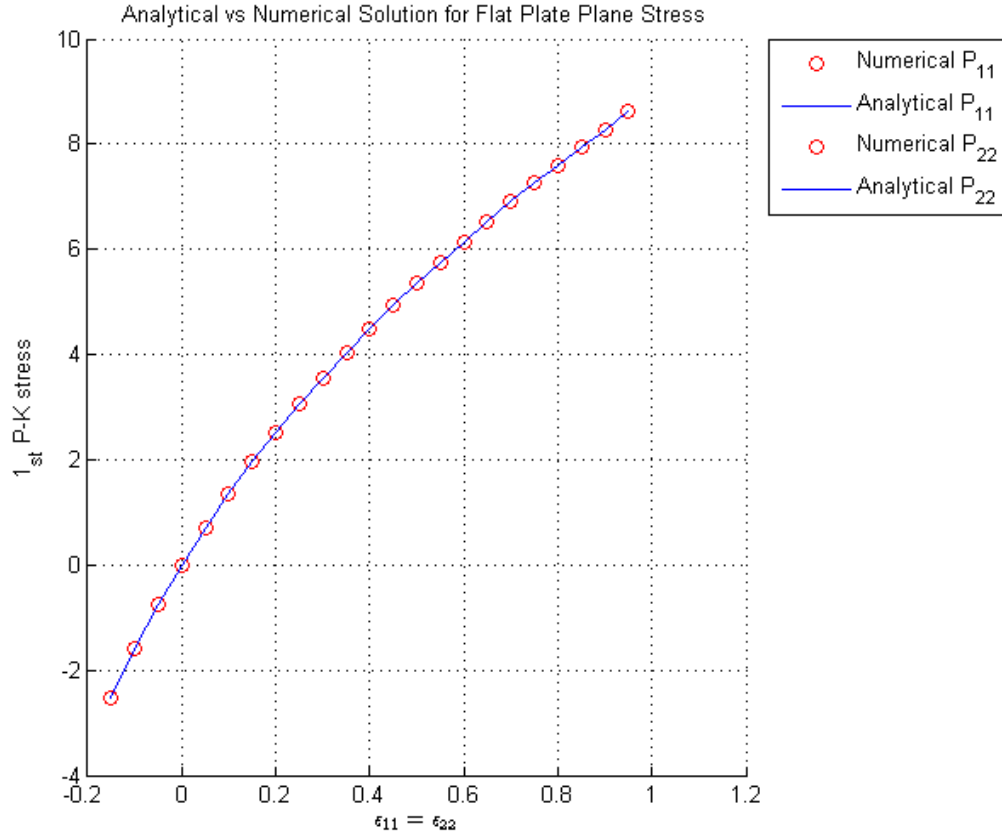


Figure 6: When a flat plate is stretched equally in two directions the deformation gradient is constant across all elements and the original neo-Hookean model can be used directly. This figure shows that both the analytical method and the numerical method using a mesh results in the same stresses in both the x and y directions. Note that the x and y stresses are the same which is what we expect for equibiaxial stretching.

The above figure shows virtually perfect consistency between the numerical and analytical methods. Also in this type of deformation linear and quadratic elements will produce the same result therefore figure 6 represents the results for both types of elements. Note that it was checked internally that deformation gradient was indeed constant across all elements.

5.5 Flat Plate under Transverse Load

5.5.1 Linear Elements

To begin this section i will present deflection graphs with 1x grid resolution and 4x grid resolution. This will give a good sense of how linear elements behave under applied load and how to appropriately use them in more complicated meshes. I will then show how the maximum deformation converges as mes resolution goes up.

1X Mesh

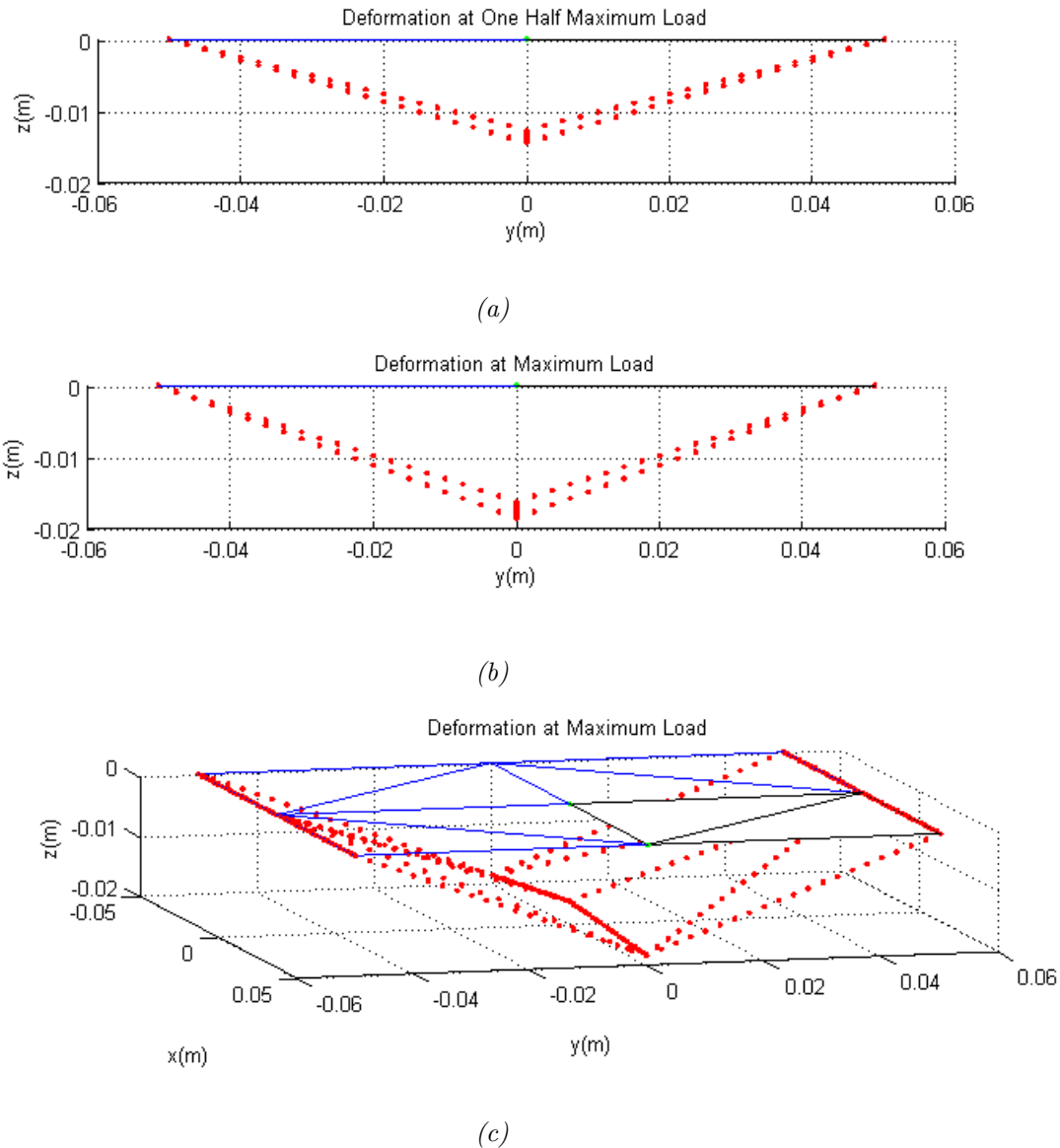
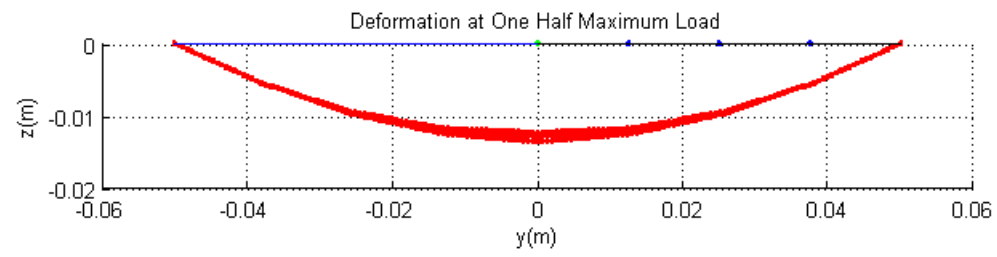
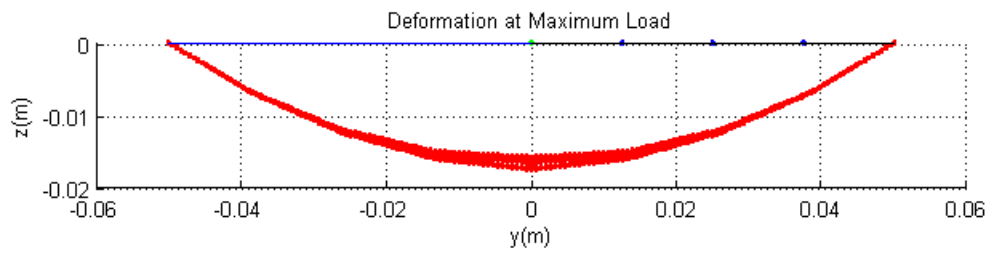


Figure 7: This figure shows (a) and (b) the deformation plot as seen from on the yz plane. (c) is a insometric view to give a better sense of the problem. Reference mesh is in blue, deformed is in red.

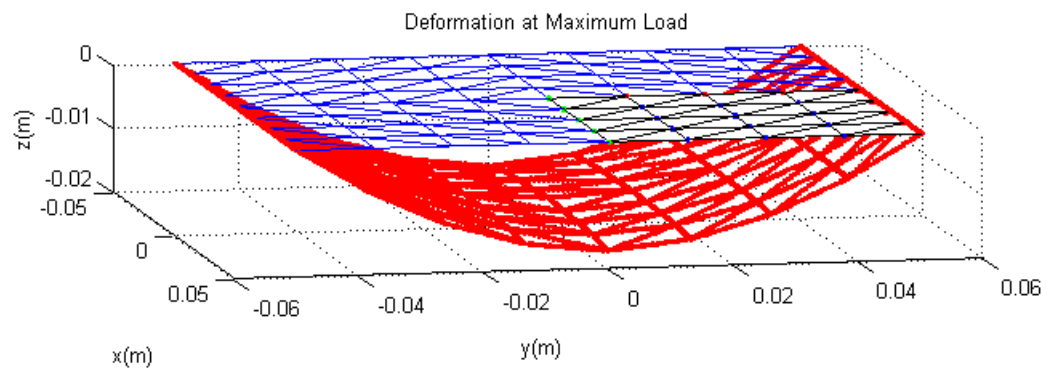
4X Mesh



(a)



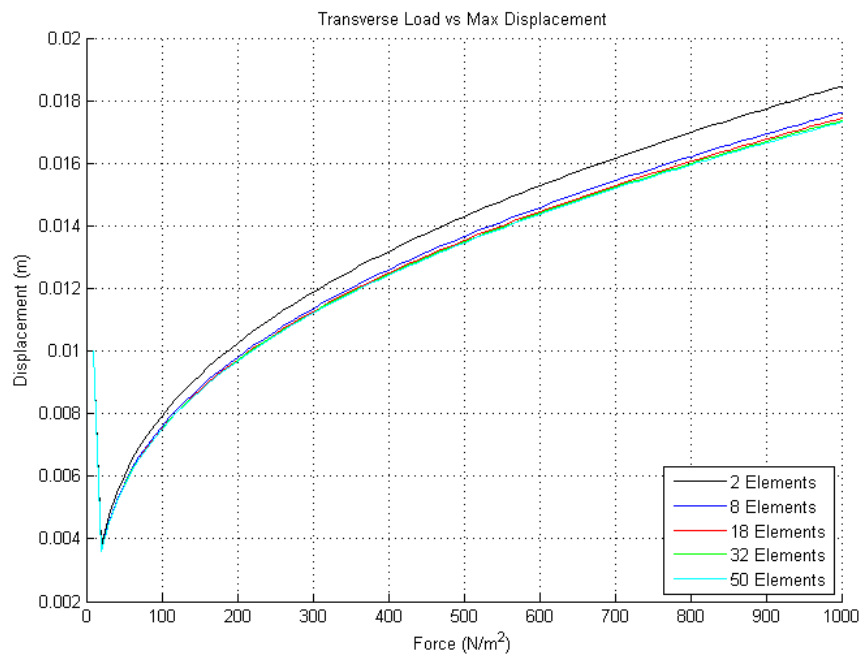
(b)



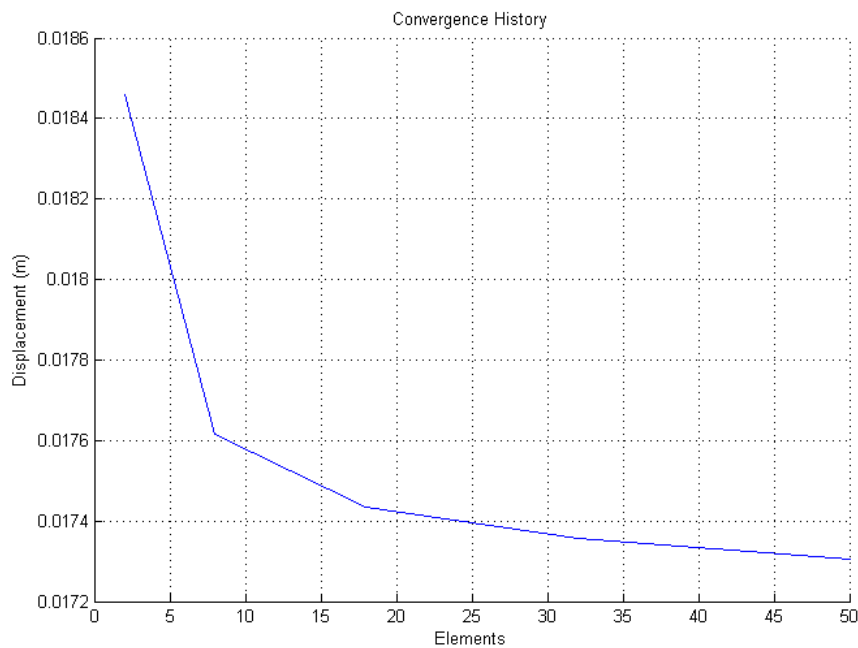
(c)

Figure 8: This figure shows (a) and (b) the deformation plot as seen from on the yz plane. (c) is an isometric view to give a better sense of the problem. Reference mesh is in blue, deformed is in red.

Convergence



(a)



(b)

Figure 9: (a) shows the displacement as a function of transverse load for several mesh refinements, while (b) just shows the maximum deformation as a function of elements used. This figure shows that as we increase the number of elements that make up our flat plate we are approaching a single value for the deformation when 10^3 N/m^2 are applied. This number is approximately 0.0173 m

5.5.2 Quadratic Elements

To begin this section i will present deflection graphs with 1x grid resolution and 4x grid resolution. This will give a good sense of how quadratic elements behave under applied load and how and when to use them. I will then show how the maximum deformation converges as mes resolution goes up.

1X Mesh

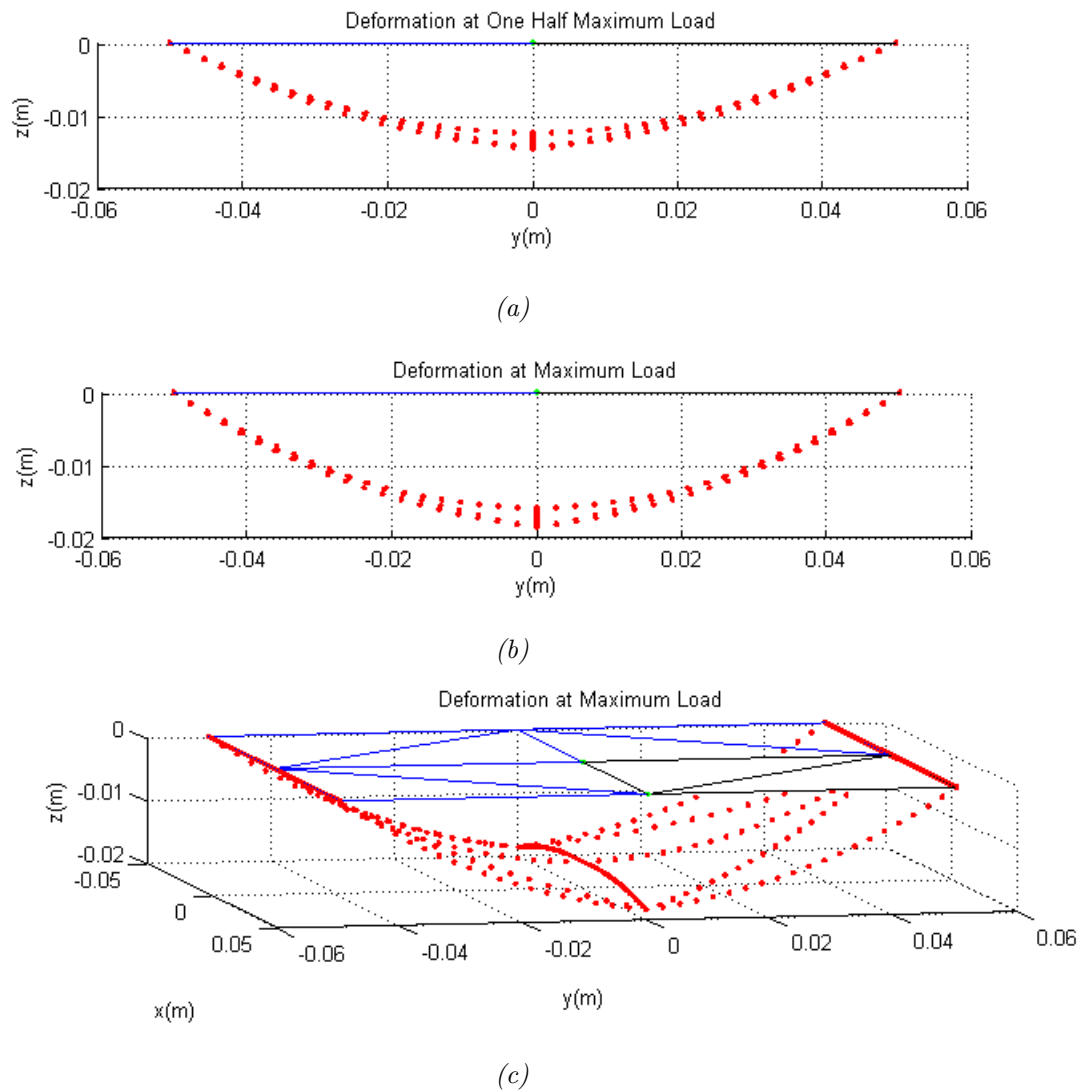


Figure 10: This figure shows (a) and (b) the deformation plot as seen from on the yz plane. (c) is a isometric view to give a better sense of the problem. Reference mesh is in blue, deformed is in red.

4X Mesh

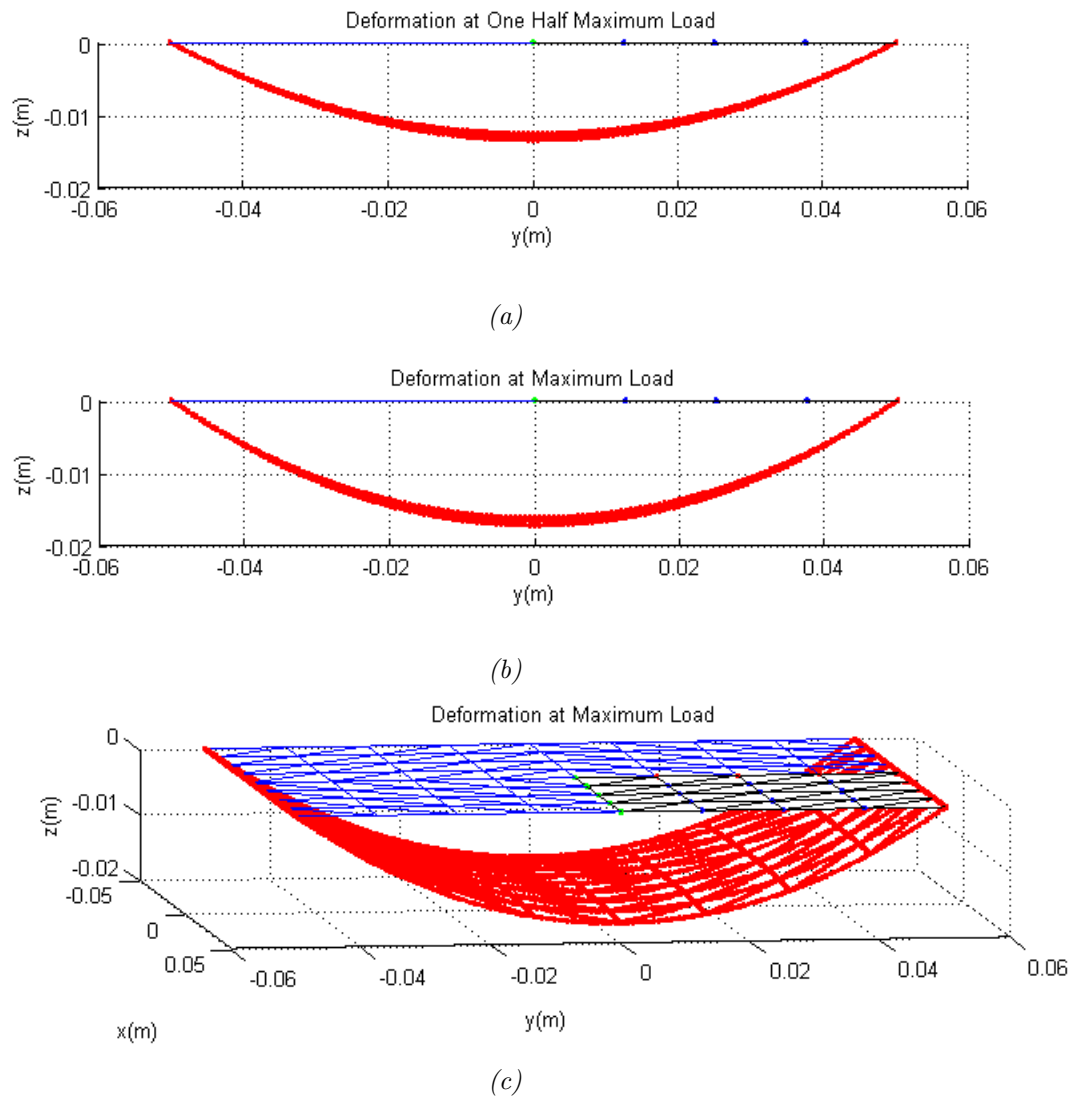
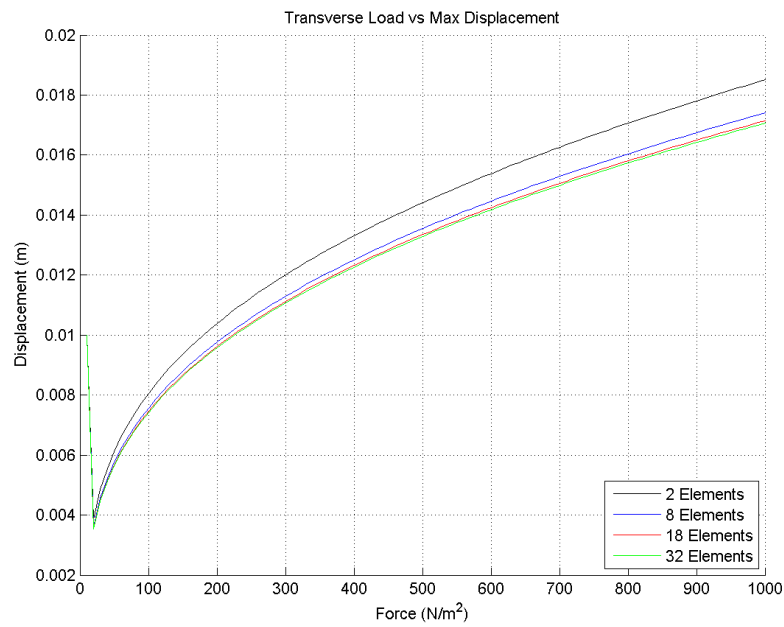
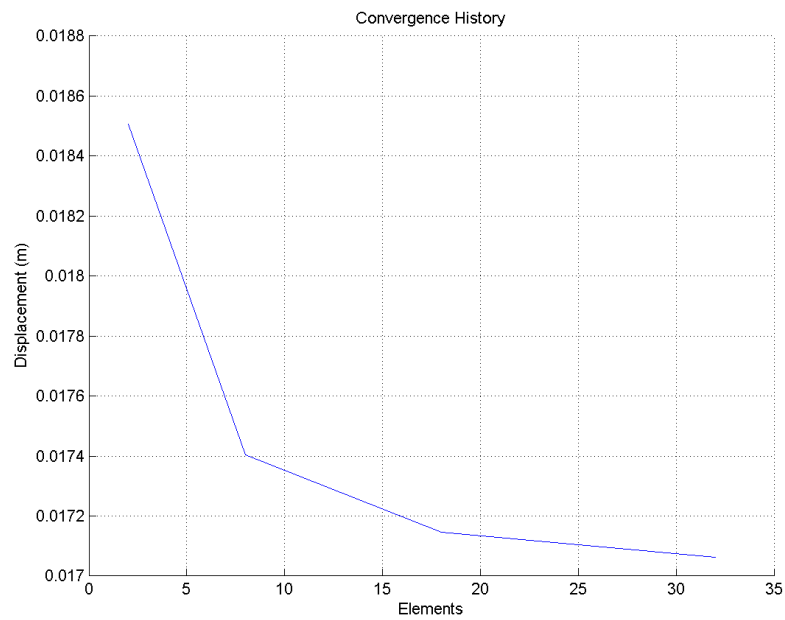


Figure 11: This figure shows (a) and (b) the deformation plot as seen from on the yz plane. (c) is a isometric view to give a better sense of the problem. Reference mesh is in blue, deformed is in red.

Convergence



(a)



(b)

Figure 12: (a) shows the displacement as a function of transverse load for several mesh refinements, while (b) just shows the maximum deformation as a function of elements used. This figure shows that as we increase the number of elements that make up our flat plate we are approaching a single value for the deformation when 10^3 N/m^2 are applied. This number is approximately 0.0171 m

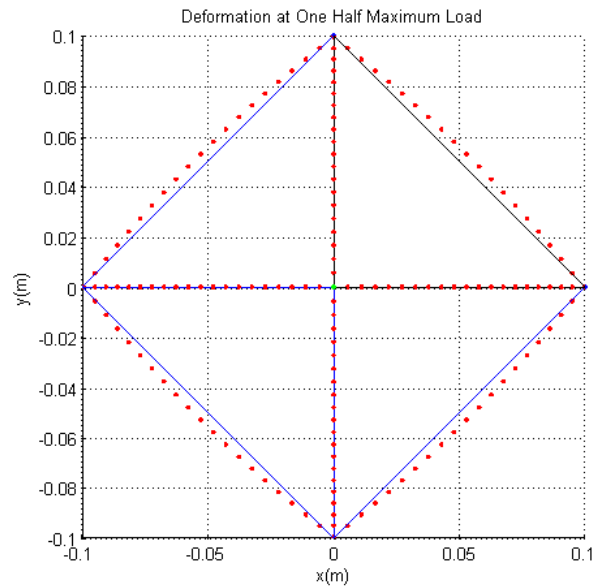
5.5.3 Flat Plate under Transverse Load: Review

Notice that while the refined meshes for both linear and quadratic elements converge to similar values, they are not the same. This is because of the nature of the two elements. Linear elements are stiff and will always have corners. Quadratic elements are softer and can round out creating a smoother deflection profile. Looking explicitly at the 1x grid, linear and quadratic elements give very different deformed shapes. However as resolution goes up the two plots eventually become indistinguishable. That being said, because the nature of a plate under transverse load will result in curvature, quadratic elements should be used. Additionally, since quadratic elements are better suited for this application they should be considered more accurate. Therefore the maximum deflection of this particular geometry under the specified loads was 1.71cm which is very reasonable given the scale of the problem.

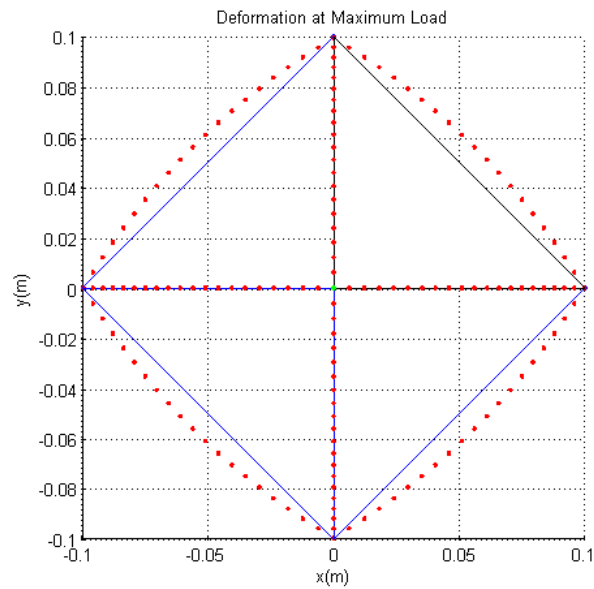
5.6 Spherical Shell with Internal Pressure

In this section several combinations of mesh resolution and force applied will be presented and discussed.

1X Quadratic Mesh 5000Pa



(a)



(b)

Figure 13: This figure shows a 1x quadratic mesh for the balloon under 2500Pa (a) and 5000Pa (b)

Stretch Ratio vs Internal Pressure

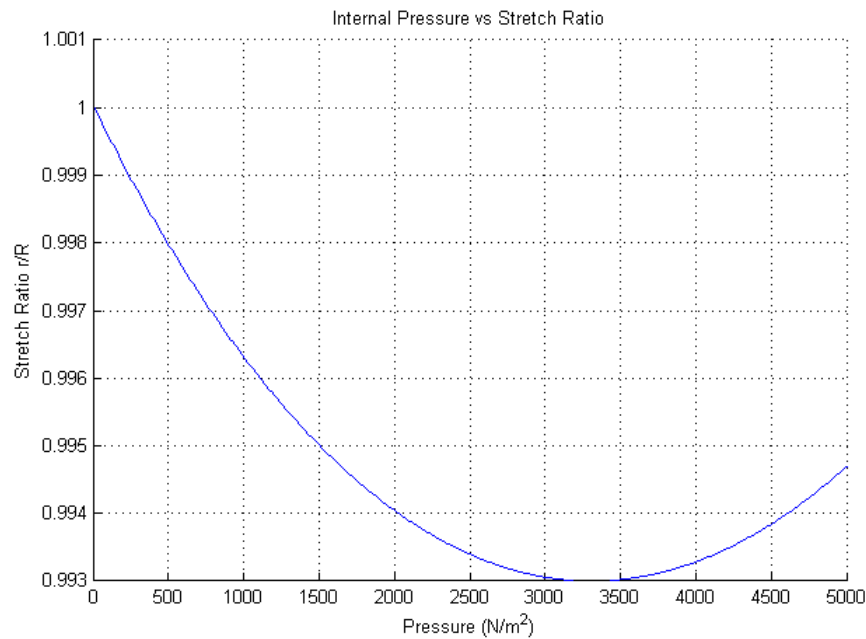
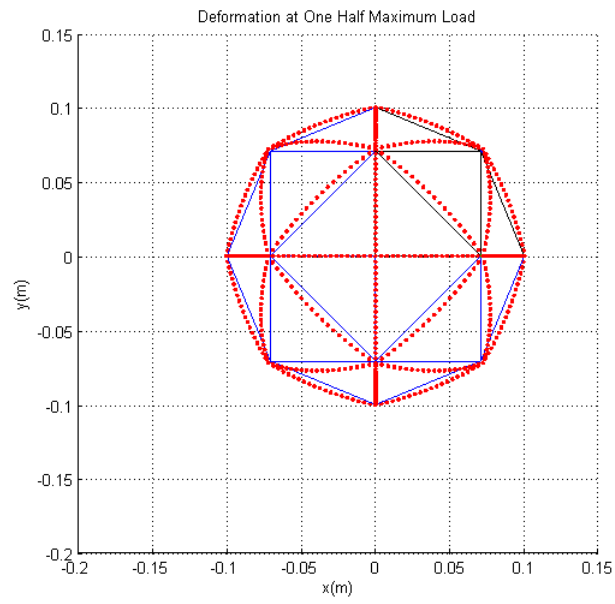
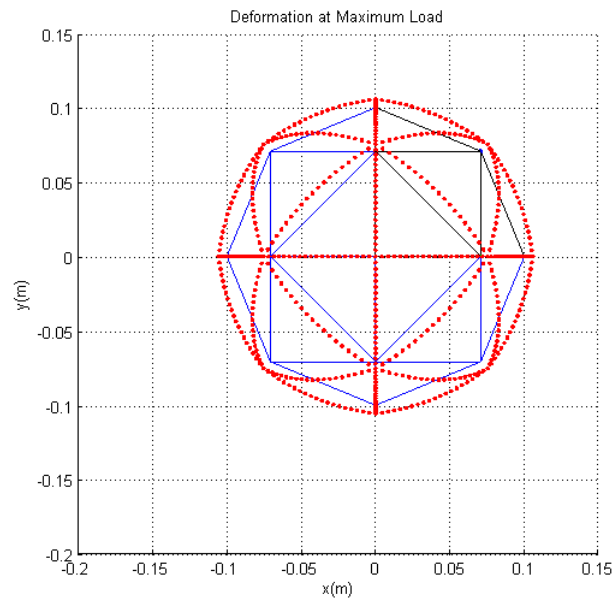


Figure 14: This figure shows the relationship between stretch length and applied pressure

Things to note about this configuration of simulation parameters: The mesh resolution of this configuration is way too low. It spends the whole simulation trying to round itself out, but since each element can only bend so much, it never truly becomes spherical. The one nice thing this graph shows is that because our initial mesh is a diamond, where the middle nodes are not actually on the true radius of the sphere, the max radius actually shrinks when a force is applied. This is a consequence of quadratic elements, because as the elements try to make themselves more round, they 'pull' in the sharp corners that were present in the initial mesh. Although it only decreases to 0.993 of the initial radius before starting to grow again, this tells us why the stretch ration for the 2x mesh had the 's' shape.

2X Quadratic Mesh 1000Pa

(a)



(b)

Figure 15: This figure shows a 2x quadratic mesh for the balloon under 500Pa (a) and 1000Pa (b)

Stretch Ratio vs Internal Pressure

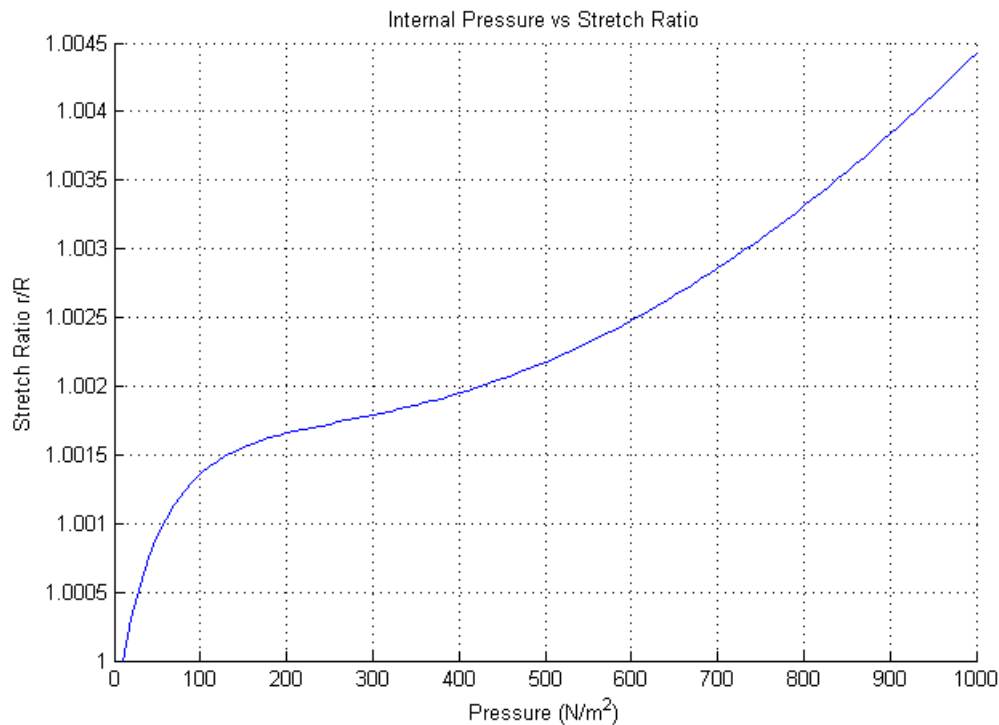


Figure 16: This figure shows the relationship between stretch length and applied pressure

Things to note about this configuration of simulation parameters: Note that because of the flexible nature of quadratic elements, even though the initial mesh is initialized to look like the linear mesh, only with extra nodes, after a few runs the middle nodes get expanded out and it looks like a smooth sphere. The most interesting trend here is in the pressure vs stretch ratio graph where we observe almost an 's' shape. Stretch ratio increases very fast initially, then slows and finally reaches a stable increase rate. This may be due to the fact that the mesh takes some time to stabilize the middle nodes that start off of the true radius of the element.

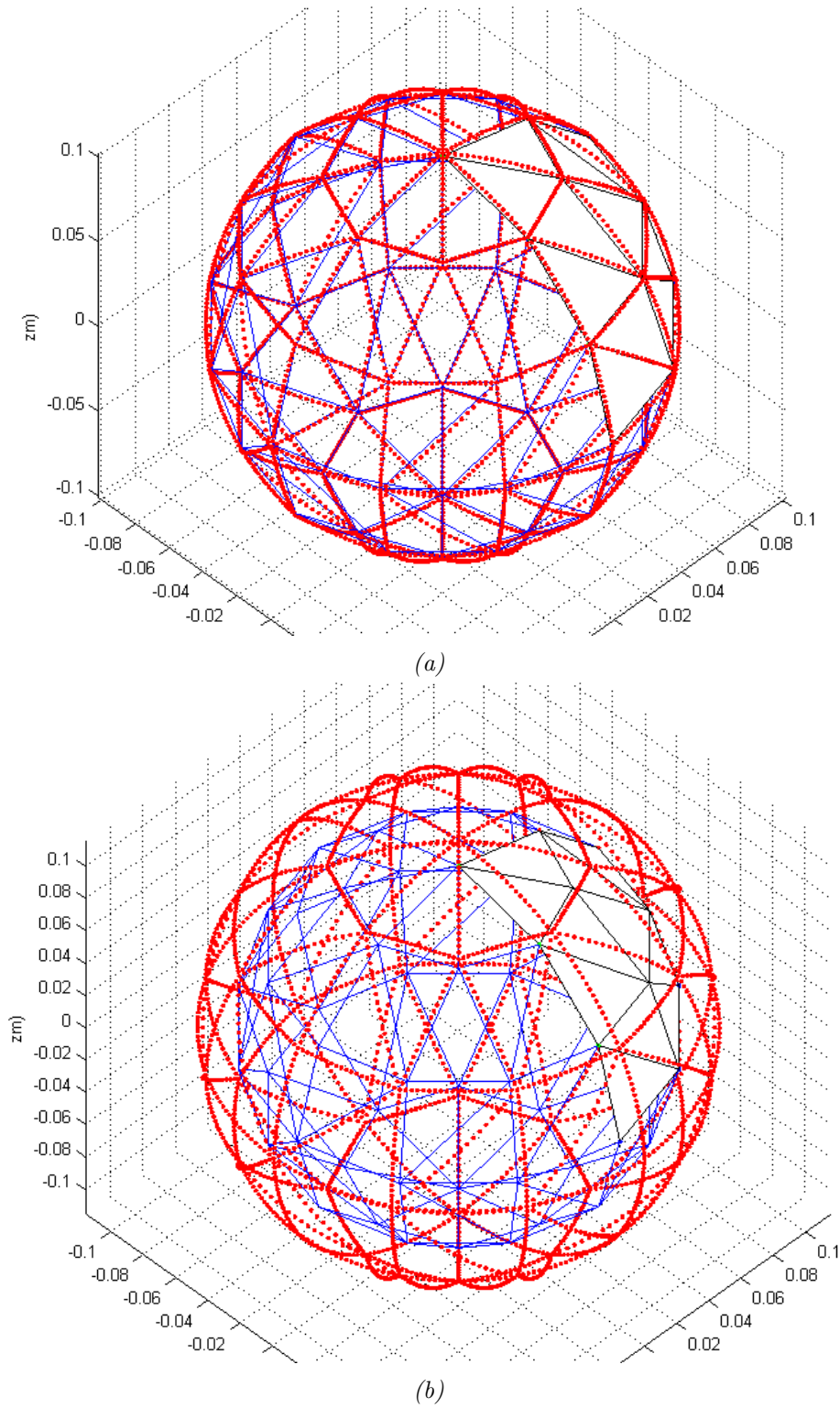
3X Quadratic Mesh 5000Pa

Figure 17: This figure shows a 1x quadratic mesh for the balloon under 1000Pa (a) and 5000Pa (b)

Stretch Ratio vs Internal Pressure

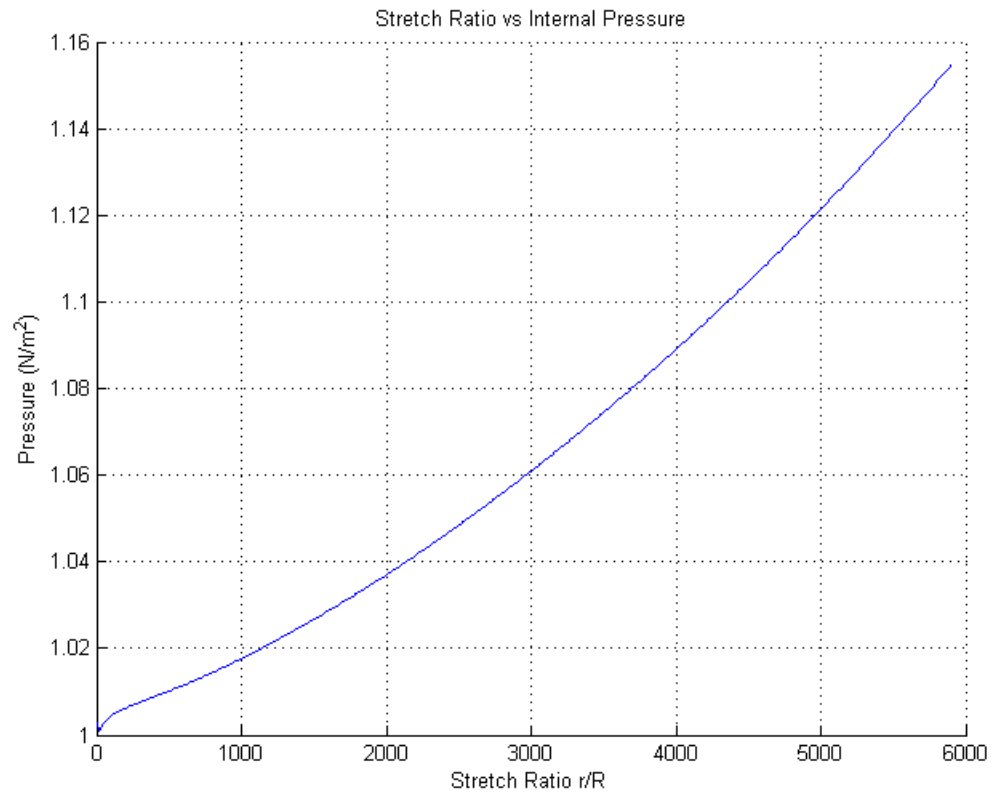
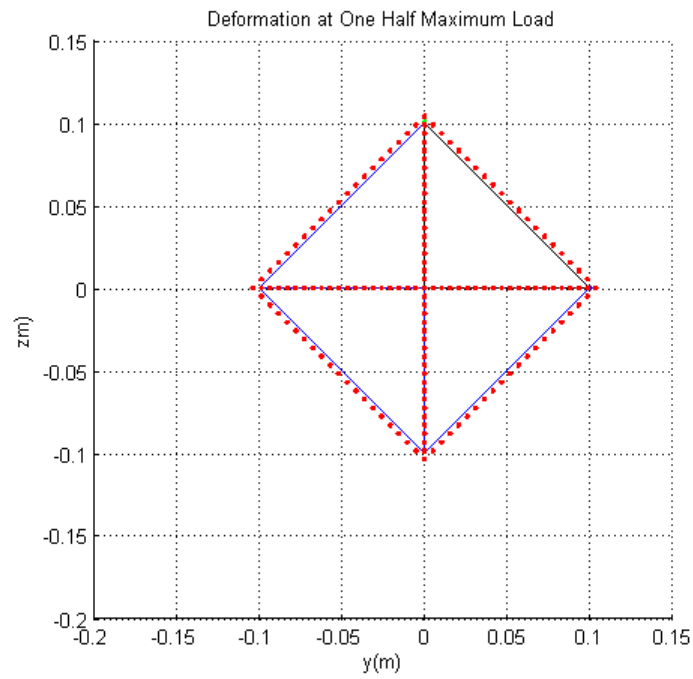
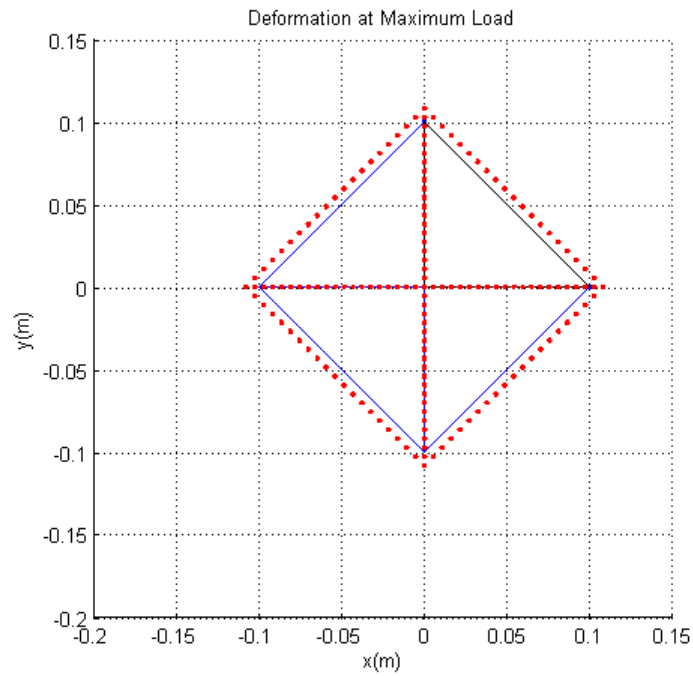


Figure 18: This figure shows the relationship between stretch length and applied pressure

Things to note about this configuration of simulation parameters: This is the first case where you can really see significant deformation in the deformation plot. The elements are a good size and this is the mesh I will use in a long scale run to see how it compares to the exact solution.

1X Linear Mesh 5000Pa

(a)



(b)

Figure 19: This figure shows a 1x linear mesh for the balloon under 2500Pa (a) and 5000Pa (b)

Stretch Ratio vs Internal Pressure

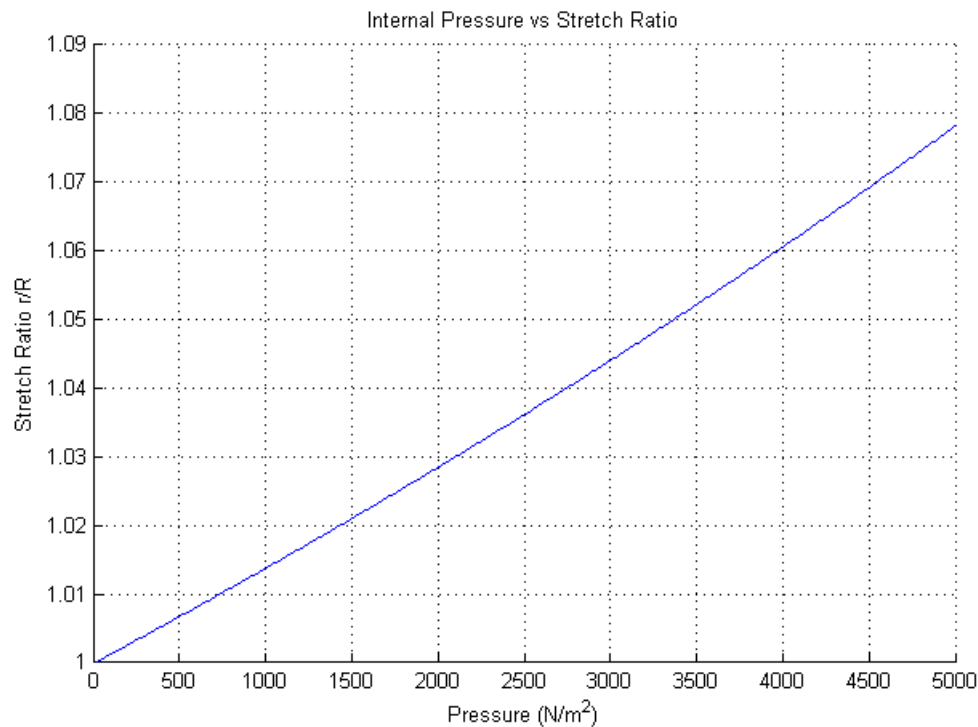
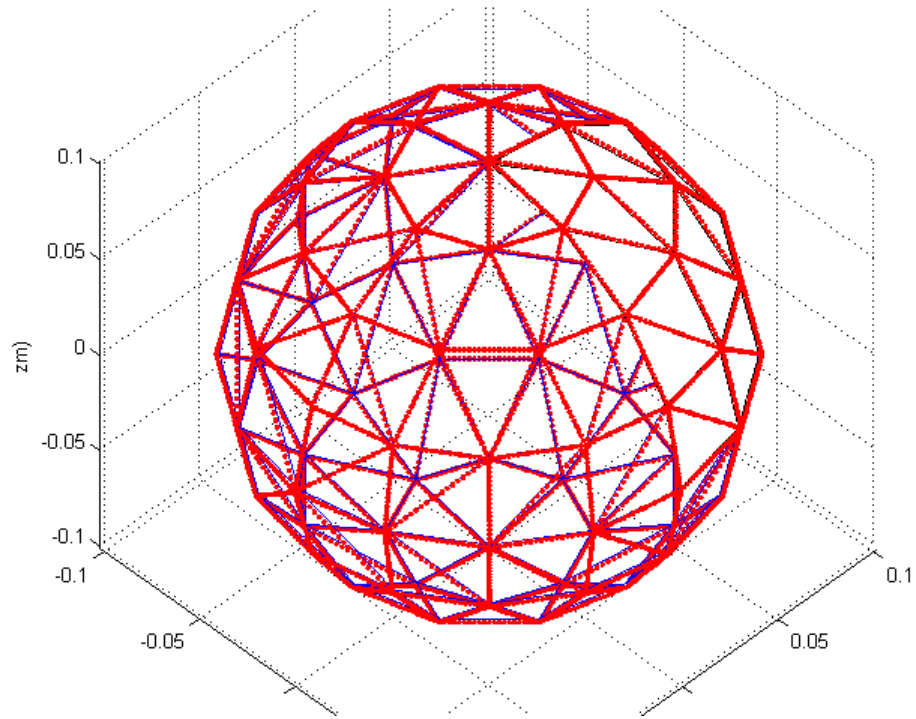
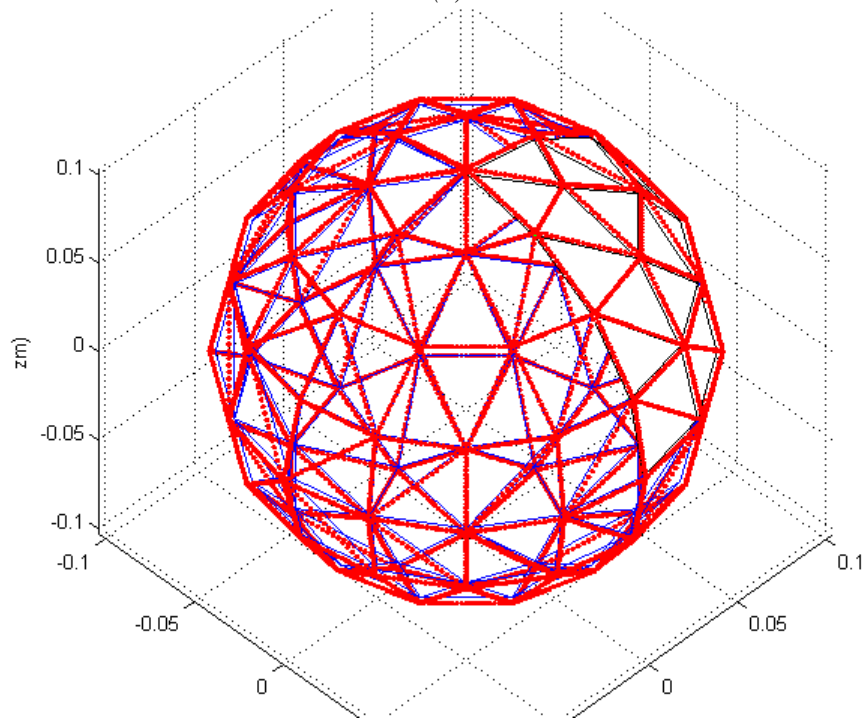


Figure 20: This figure shows the relationship between stretch length and applied pressure

Things to note about this configuration of simulation parameters: Linear elements are very stiff. This particular mesh is very rough in size and the initial mesh is more of a diamond than anything. However because linear elements are stiff, it stays a diamond for all time, the only thing each element can do is scale. Therefore, while you do get stretch, it will not be as accurate as the quadratic case. Looking at the stretch ratio graph we see a different trend than the quadratic case. This is again because of the stiffness of the elements. The next plot will show a mesh refinement to see the effect that has for linear elements.

4X Linear Mesh 1000Pa

(a)



(b)

Figure 21: This figure shows a $4x$ linear mesh for the balloon under 500Pa (a) and 1000Pa (b)

Stretch Ratio vs Internal Pressure

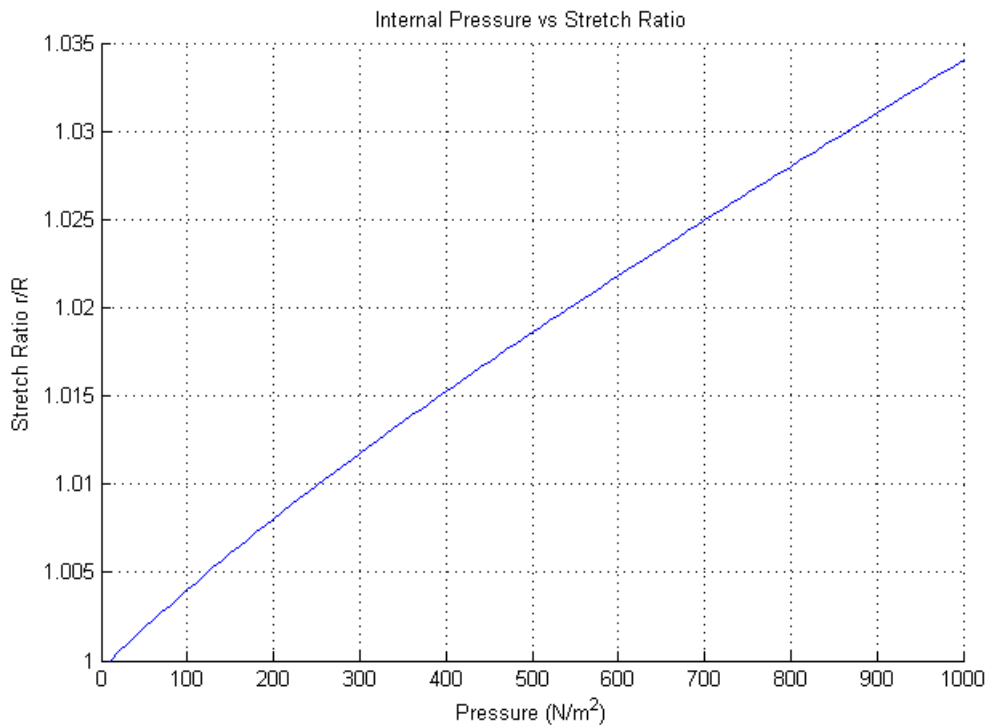


Figure 22: This figure shows the relationship between stretch length and applied pressure

Things to note about this configuration of simulation parameters: Linear elements are very stiff. Although this mesh is much more refined than the 1x it still has sharp corners. This is inevitable with linear elements. However, even with just stiff triangles, we observe a shape that looks essentially spherical. A fine enough mesh should approach the exact solution. Looking at stretch ratio, this configuration of parameter looks slightly different than the simple mesh, It almost looks like it is flattening out. This simulation needs to be run for longer to see what the long term trend would be.

Max Pressure

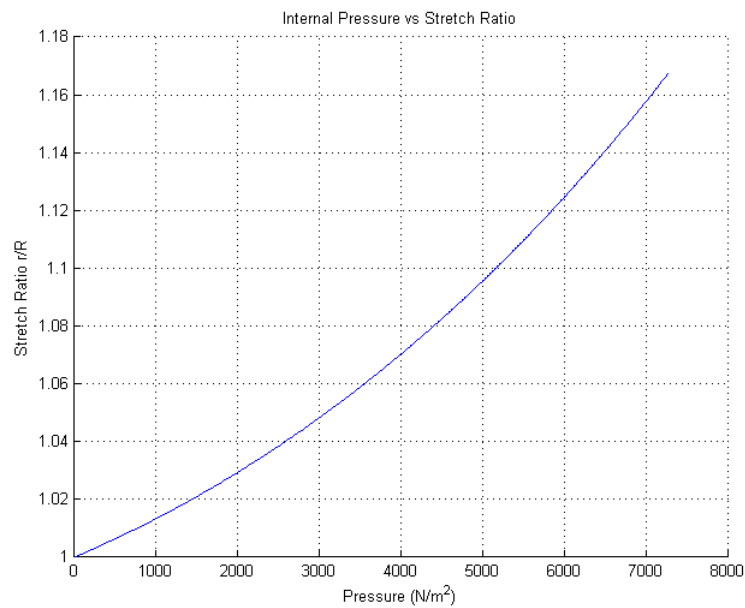


Figure 23: Max pressure for a Linear element on $2x$ mesh. This figure shows the relationship between stretch length and applied pressure, up to the point that the Newton solver could no longer converge. This occurred at $P = 7270\text{Pa}$

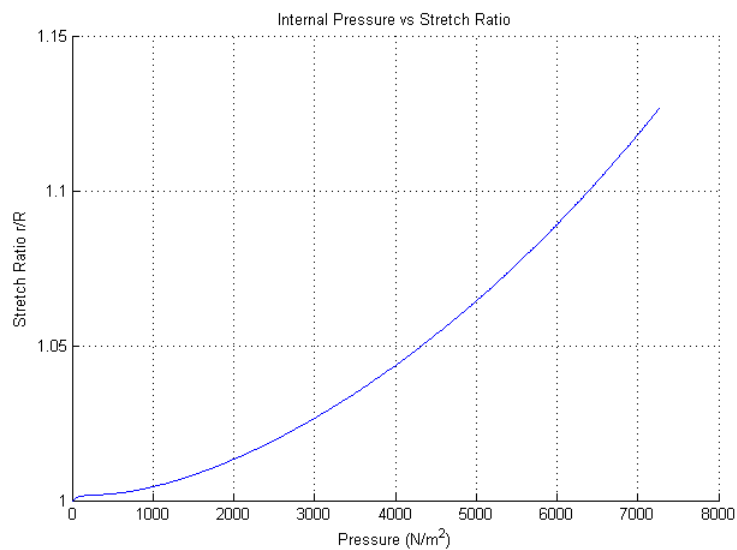


Figure 24: Max pressure for a quadratic element on $2x$ mesh. This figure shows the relationship between stretch length and applied pressure, up to the point that the Newton solver could no longer converge. This also occurred at $P = 7270\text{Pa}$

Comparison to Exact

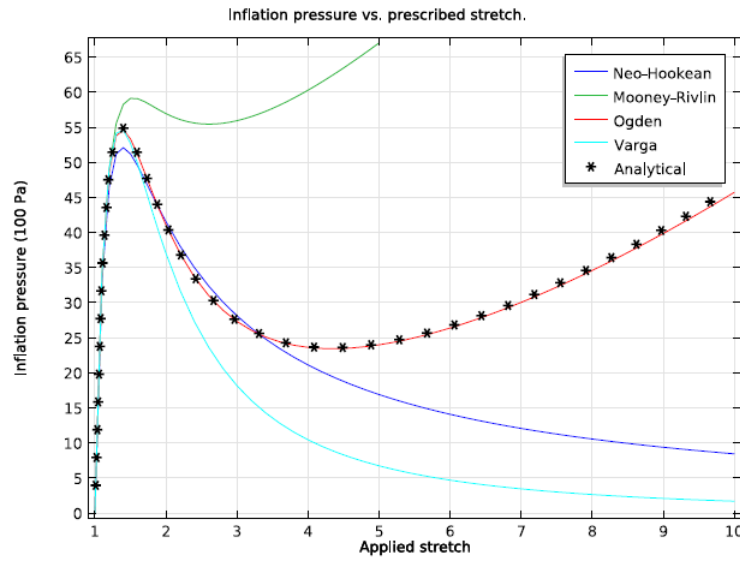
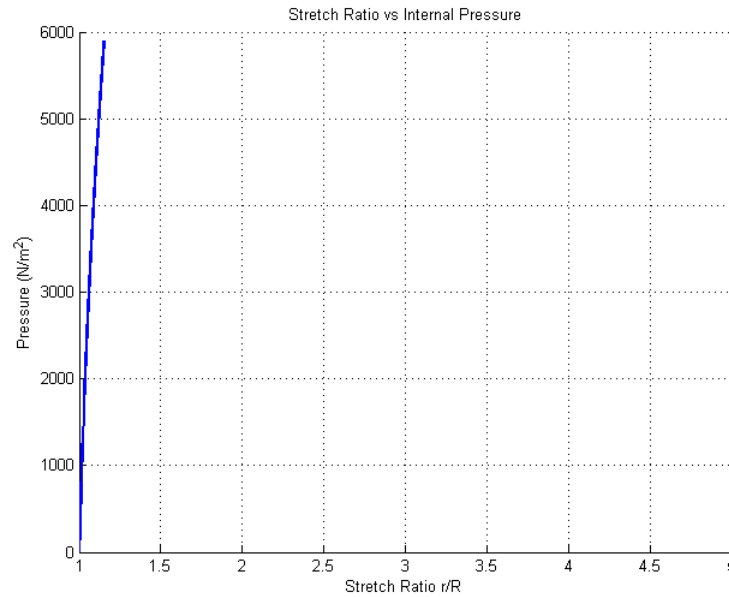


Figure 3: Computed inflation pressure as a function of circumferential stretch for different material models, compared to the analytical expression for Ogden material.

(a) This figure shows the exact solution as presented by [2]



(b) This figure shows the numerical solution computed with 3x quadratic mesh.

Figure 25: Exact vs Numerical Internal Pressure. While my solution has not had a chance to run fully, the part that has run looks like it is of similar functional form as the exact neoHookean model as provided by ComSol 2012. Given more time i would like to fully run my algorithm on the 3x quadratic mesh and see if it eventually levels off.

5.6.1 Spherical Shell with Internal Pressure: Review

Overall we see very different behavior when using linear vs quadratic elements. Spherical shell is very round and only quadratic elements can truly capture this roundness, so they should give better accuracy in this case. However, despite their differences, they both determined the same value for max pressure when the Newton-Raphson iterations would no longer converge. This value was found to be 7270Pa. The stretch ratios at this point was slightly different, however the consistency in max force adds validity to my method. While there are many possible choices of meshes and elements, using a 3x quadratic mesh looks like it models the exact solution fairly well as shown by the previous figure.

6 Conclusion

As seen by the results, membranes are a very practical and accurate way to analyze real systems in a simple manner. While the nonlinear finite element solver developed in this paper is relatively slow due to numerous nested for loops and while loops, it is still a consistent and accurate method. Given more time i would strive to improve the speed of this algorithm and add in dynamic and viscous terms. As it is however, the FEM i developed over the past few weeks is a valuable asset in analyzing simple geometries.

7 Source Code Listing

hw3main.m The main routine where everything else is called. This homework was broken down into several large functions. Due to time issues the spherical balloon under pressure was implemented here too.

assemble.m This takes a given reference and deformed mesh and computes the stiffness and forces for each element then puts them together in big global arrays.

clipper.m This function takes a stiffness matrix and resultant and cuts out the degrees of freedom that are not being looked at

consistency.m This function tests the consistency of either a single element or a whole mesh. Decides based on if it is passed a connectivity list.

enforcePS.m This function performs the newton iteration to drive Tau 33 to zero. It returns the transformed quantities from the neoHookean routine

gen_deform.m This was a helper function to help test things. It just creates certain types of deformation i ask it to.

get_duals.m This function takes in the given tangent vectors and returns the duals by using the inverse metric.

get_tans.m This function takes in the reference and deformed configurations and returns the tangent basis vectors

linTri_Na.m This is the linear shape function.

quadTri_Na.m This is the quadratic shape function.

neoHookean.m This is the fundamental hyperelastic model and the base of the code. It calculates P , C_{ijkl} and w . It returns T_{CIJKL} and w . It computes F internally from tangent basis.

neoHookeanOLD.m This was needed to compare the results for the equibiaxial problem. This would return the analytical values of P based on a specified H .

plate_iso_def.m This function will take in how much you want to stretch a plate and try to do it without incremental loading. Pretty limited use.

plate_iso_def.m This function however does incremental loading and can handle much larger deformations.

plate_mesh.m This function takes in a L W and mesh parameters and returns the mesh for a $1/4$ plate. I use symmetry in all solvers so only need $1/4$ mesh.

plot_tans.m Makes a nice visual to see that my tangent vectors and shape functions are working correctly.

PSNeoHookean.m This was also used to calculate analytical P during equibiaxial.

quad_mesh.m My original meshing function only do three pointed triangles. if i want to used quadratic elements i need 6 nodes therefore i add on on the midpoints and add the new connectivity to the original connectivity list as we as adding the new nodes to the nodes list.

quad_midpoints.m helper function for `gen_deform.m` needed because quadratic elements have more nodes than linear ones.

quatra_rule.m This function returns the quadrature rules we are using. Essentially a look up table

sphere_mesh.m one of my prides and joy from this assignment. Generate the mesh for any sphere with a specified mesh size.

TL_plate.m This one is the sub routine that handles transverse loading. Does incremental force.

transform_from_para.m This takes parametric coordinates and configurations and tells you where in Cartesian space that is.

unwrap_K.m This function simply makes K square by reshaping it.

WfK.m Another big work horse. Calculates the elemental f K and w .

8 References

- [1] Klug, William. "Lecture Notes MAE 261B" UCLA MAE Department. Winter 2015.
- [2] 'Inflation of a Spherical Rubber Balloon.' Comsol 2012.