A Method of Simulating Fluid Structure Interactions for Deformable Decelerators

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Vladimyr Mykhalo Gidzak

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

November, 2010

Acknowledgements

Great thanks goes to my advisor Graham Candler, who gave me a chance to prove myself in graduate school, to Ioannis Nompelis who fueled my interest in the simulation sciences, and all of the colleagues from whom I harvested much useful knowledge and expertise.

Dedication

To my sweet wife Bonnie.

Abstract

A method is developed for performing simulations that contain fluid-structure interactions between deployable decelerators and a high speed compressible flow. The problem of coupling together multiple physical systems is examined with discussion of the strength of coupling for various methods. A non-monolithic strongly coupled option is presented for fluid-structure systems based on grid deformation. A class of algebraic grid deformation methods is then presented with examples of increasing complexity. The strength of the fluid-structure coupling is validated against two analytic problems, chosen to test the time dependent behavior of structure on fluid interactions, and of fluid on structure interruptions. A one-dimentional material heating model is also validated against experimental data. Results are provided for simulations of a wind tunnel scale disk-gap-band parachute with comparison to experimental data. Finally, a simulation is performed on a flight scale tension cone decelerator, with examination of time-dependent material stress, and heating.

Contents

List of Figures

Chapter 1

Introduction

1.1 Ballistic Entry

When designing a craft for atmospheric entry, one of the most important factors is the ballistic coefficient. The ballistic coefficient is the mass of the craft divided by its coefficient of drag and cross-sectional area.

$$BC = \frac{Mass}{C_{Drag} Area}$$

By being able to control the ballistic coefficient during design, we can effectively have some control over the types of atmospheric entry trajectories that can be flown. The mass is dictated by the mission requirements of the craft, and the drag coefficient depends on the shape which is partially dictated by the space restrictions of packing the craft on top of a rocket.

The exposed area, however, can be adjusted by the deployment of deformable inflatable structures. The archetypical example of this is a parachute. The parachute can be packed into a small volume and then deployed when the craft is required to decelerate, at which time the parachute inflates, creating a much larger total area exposed to fast moving fluid.

However, the simulation of inflatable systems under aerodynamic load is difficult. Unlike rigid craft, any structure that can be packed into a small volume, is deformable, and can change shape when exposed to a high velocity flow. Most flow solvers are

created with the assumption that the geometry of the object to be simulated will remain constant throughout the duration of the simulation. This is not the case for decelerators, which by design are required to be deformable.

1.2 Fluid Structure Interactions

The phenomena in which a fluid flow changes the geometry of its own boundaries are called Fluid Structure Interactions (FSI). There are significant differences in the methods used to study fluid structure interaction across flow regimes, as the tools needed to study each regime vary widely.

Work in this field spans the incompressible regime with applications such as biological flow [1] in which finite-element methods are used for both computational fluid dynamic (CFD) and structural solvers. Subsonic aerodynamic airfoil flutter [2] has been studied using Arbitrary-Lagrangian-Eulerian (ALE) finite-volume method for simple deformations.

Recent developments in low Mach number supersonic flows involving parachute deployment have been made using methods where a structural mesh passes through a CFD mesh. Lingard [3] accomplishes this by the use of an ALE method to enforce flow conditions in the regions of the CFD domain as determined by the position of the structural mesh. Resolution is improved in regions near the structural mesh by clustering CFD grid cell via an advection of the CFD mesh. Pantano [4] also tracks a structural mesh through the CFD domain but use Cartesian adaptive mesh refinement to ensure high resolution of the domain near the interface, and a cut cell method for enforcing boundary conditions.

In high Mach number flows, it becomes necessary to use computational grids (or meshes) that are body fitted and highly refined so that one can resolve high enthalpy boundary layers. Due to the requirement of having an exponential increase in grid density normal to wall boundaries, it is not possible to achieve adequate resolution by advecting cells through the domain from an initial coarse mesh, nor by simply blocking out Cartesian cells to accommodate an arbitrary geometry. The high level of grid refinement also makes the implementation of certain common spring-based [5] grid deformation strategies used in subsonic aeroelastic problems unpractical [2].

It is the goal here to extend the simulation of fluid structure interactions to the high Mach number regime where the use of body fitted grids is needed.

1.3 Outline of this Work

In this work, a method will be developed that is able to carry out high fidelity, time accurate simulations of deformable decelerators. This requires the coupling together of a solver able to solve the fluid physics with one able to simulate the dynamics of a structural system. Also required is a method for modifying the space discretization, or grid, used by the flow solver.

Chapter 2 examines the methodology of coupling solvers for physical simulations and provides the framework with which fluid-structure interactions can be modeled.

Chapter 3 presents a class of algebraic grid deformation methods for use with FSI simulations which are adaptable and expandable to a wide range of needs.

Chapter 4 presents validation of the interaction terms between the flow solver and the structural membrane solver, as well as for a 1-D heating model. It is the view expressed herein that the validations of the individual components of a coupled system is sufficient for the validation of the whole as long as the cross terms between each component are updated at every time iteration, and that the time step used is reflective of the physics in both systems.

Chapter 5 contains simulations of a supersonic disk-gap-band (DGB) parachute with comparison to wind tunnel data. It is concluded here that the performance of disk-gap-band parachutes at supersonic speed is dictated by the interaction of three different phenomena; the cycling of canopy pressure, large-scale motion off axis, and loss of exposed area.

Chapter 6 looks a simulations on a tension cone inflatable decelerator at flight scale. The derivation of a tension shell shape is explained, and the time dependent material stresses experienced in the tension cone are examined.

Some general observations will then be made on the nature of simulating phenomena that include fluid structure interactions, with comments regarding the direction in which future developments may lead.

Chapter 2

Framework for Fluid Structure Interactions

2.1 Introduction

Whenever it is desirable to combine multiple physical simulations together, some form of coupling is needed. Occasionally this can be done analytically when forming the governing equations prior to implementing them in a simulation. An example of this is the way that conservation of mass, momentum, energy, an equation of state, and the Stokes hypothesis on viscosity can be combined to form the Navier-Stokes equations. More often, when we talk of a simulating a coupled system, we are referring to two or more separate physical systems that are not analytically combined, but influence each other via cross-terms. The strength of the coupling depends on how the cross-terms are treated.

There exists different ways of coupling different physical systems, to varying degrees of accuracy and requiring a range of complexity in their implementation. The best method to use may depend on the application. An overview is provided here of different levels of coupling systems, with the advantages, and difficulties of each explained.

As background, an overview is given on how a discretized system of equations can be integrated forward in time, and how such formulations can be modified to fit in a coupled framework.

The methods of coupling that will be examined here are a *Monolithic* formulation in

which all governing equations are combined into a single massive system of equations, an *In-Step* formulation in which each system is solved independently but integrated forward together in time, *Weakly Coupled* in which the systems are not integrated forward together in time, and *Iterative* methods in which the goal is a final equilibrium solution, without the resolution of time dependent behavior.

Finally, an In-Step coupling method relying on grid deformation will be presented for the simulation of high-speed time dependent fluid-structural interactions.

2.2 Formulation of Coupled Systems

2.2.1 Implicit Formulation of a System of Equations

Systems of equation can be solved in either an explicit, or an implicit formulation. In an explicit formulation, the change in the state variables The changes in the state variables, U, are linearized as:

$$\Delta U_i = A_{i,j} U_j \Delta t + \Delta U_0$$

where $A_{i,j}$ is the rate of change of state variable U_i per unit time with respect to variable U_j. ΔU_0 is the component change to the state variables that does not depend on the state variables themselves.

In an explicit formulation, the values of U used to evaluate ΔU are taken from the beginning of each time iteration. In an implicit formulation, the values for U at the end of the time iteration are used to evaluate ΔU. Absorbing Δt into $A_{i,j}$, we get;

Explicit : $\Delta U_i^n = A_{i,j} U_j^n$

Implicit : $\Delta U_i^n = A_{i,j} U_j^{n+1} = A_{i,j}(\Delta U_j^n) + \Delta U_{i\,explicit}^n$

In the explicit formulation, ΔU is evaluated by the product $A_{i,j} U_j^n$.

In the implicit formulation, the system of equations,

$$[A_{i,j} - I_{i,j}]\Delta U^n = -\Delta U_{i\,explicit}^n,$$

must be solved. The implicit system is more computationally expensive per iteration than the explicit scheme, but allows for much larger time steps which make up for the additional cost.

2.2.2 Monolithically Coupled

In a monolithically coupled system of equations, two or more systems are combined into a single large system of equations, and solved simultaneously.

$$\begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$$

This is the most useful when both systems have a similar nature and use the same solution method. In this form of coupling, both systems can easily be made implicit with respect to the others state variables.

$$\Delta U_1^n = fcn\left[U_1^{n+1}, U_2^{n+1}\right]$$
$$\Delta U_2^n = fcn\left[U_1^{n+1}, U_2^{n+1}\right]$$

Monolithically coupled systems have found use in the simulation of biological flows where both the incompressible fluid and the structure solvers use a finite element formulation [1]. The inclusion of finite rate chemistry in a reacting flow solver, while not traditionally named as such, can also be considered a monolithic coupling of the flow equations with the chemistry equations. When chemistry is coupled into a flow solver, it is more effective to combine its state vector directly with that of the flow solver. Instead of having the two systems kept separate;

$$U_{chem} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \cdots \\ e_{chem} \end{bmatrix} \qquad U_{flow} = \begin{bmatrix} \rho \\ \rho\vec{u} \\ e_{flow} \end{bmatrix}$$

we combine the state vectors for each cell into

$$U_{combined} = \begin{bmatrix} \rho_1 \\ \cdots \\ \rho\vec{u} \\ e_{flow}+e_{chem} \end{bmatrix}$$

Implementing a monolithically coupled system requires the solver to be written with coupling in mind from the outset. This increases the difficulty in implementation, and often requires using the same solution method for both systems.

2.2.3 In-Step Coupling

In cases where an existing solver is to be coupled with a second, strong coupling can still be achieved by embedding the secondary system as a sub-step of the primary solver. In this form, the primary system can be made implicit with respect to both its own state variables, and those of the secondary system, while the secondary system is only implicit to its own variables.

$$\Delta U_1^n = fcn\left[U_1^{n+1}, U_2^{n+1}\right]$$
$$\Delta U_2^n = fcn\left[U_1^{n}, U_2^{n+1}\right]$$

This method still provides strong coupling between two systems, as the cross terms between each system are still updated at every iteration, using the most recent available information. This method allows for each system to be solved by methods best suited the physics of the each system. The method of simulating fluid-structure interactions via grid deformation used here employs this type of coupling.

2.2.4 Weakly Coupled

The term weakly coupled, is applied to a range of configurations in which each system is implicit only to its own state variables. In some implementations, the influence of one system on the other may not be updated at every iteration.

$$\Delta U_1^n = fcn\left[U_1^{n+1}, U_2^{m}\right]$$
$$\Delta U_2^n = fcn\left[U_1^{m}, U_2^{n+1}\right]$$

with $m \leq n$.

This type of coupling can be useful for phenomena where the inter-system interactions are small and slowly changing. Solving a weakly coupled system will require less modifications to existing software than a monolithic scheme, or other strongly coupled method. This method is only a viable option when the two systems evolve at very disparate time scales. It is applicable when the cross terms either do not change from iteration to iteration, or change very little. Even then, it should be used with caution, with attention paid to how the coupling may distort the end result.

2.2.5 Iterative and Uncoupled

A set of systems should be considered uncoupled when the interaction terms between the two systems are no longer present.

$$\Delta U_1^n = fcn\left[U_1^{n+1}\right]$$
$$\Delta U_2^n = fcn\left[U_2^{n+1}\right]$$

Even though there is no direct communication between the two systems of equations, each system can be periodically restarted using boundary conditions extracted from the other solver. Solving such a system requires little to no modifications to pre-existing solvers. While this method cannot be used to solve time dependent problems, it is quite useful in iterative design optimization [6].

2.3 FSI via Mesh Deformation

In this work, structural interactions are coupled to a finite volume compressible flow solver. At each iteration, the structural solver receives surface forces from the flow solver and return surface displacements. The surface displacements are then used to deform the flow solver mesh.

With proper boundary conditions and adjustments to the flux evaluations, the mesh deformation is sufficient for coupling information from the structural solver back into the flow solver.

2.4 Solver Overview

2.4.1 Finite Volume Solver

The flow solver used is a modified version of US3D [7], [8], which is an unstructured implementation of the DPLR method [9].

In the finite volume formulation the state variables are conserved quantities, in intensive form, stored in cells. A comprehensive formulation can be found in [10]. The portion of the state vector for each cell is

$$U = \begin{bmatrix} \rho \\ \rho\vec{u} \\ e \end{bmatrix}$$

Where ρ is the mass density, $\rho\vec{u}$ is the momentum density, and e is the energy density. The variables change by transport in and out of cells via fluxes evaluated at the cell faces. The change in the state variables is in conservation law form as

$$\frac{\partial U}{\partial t} + \bigtriangledown \cdot \vec{F} = 0$$

This can be expressed for each cell as a sum over that cell's faces, divided by the cell's volume

$$\frac{\partial U}{\partial t} = \frac{1}{Vol} \sum_{faces} \vec{F} \cdot \hat{n}\, Area$$

Where $\hat{n}\, Area$ are the normal vectors of the faces, pointing out of the cell, multiplied by that face's area. The fluxes have both an inviscid and a viscous component. The inviscid component, in vector form is

$$\vec{F}_{invisc} \cdot \hat{n} = \begin{bmatrix} \rho\vec{(u} \cdot \hat{n}) \\ \rho\vec{u}\vec{(u} \cdot \hat{n}) + P\hat{n} \\ (e+P)(\vec{u} \cdot \hat{n}) \end{bmatrix}$$

while the viscous component is

$$\vec{F}_{visc} \cdot \hat{n} = \begin{bmatrix} 0 \\ \vec{\bar{\tau}} \cdot \hat{n} \\ (\vec{\bar{\tau}} \cdot \vec{u}) \cdot \hat{n} + \vec{q} \cdot \hat{n} \end{bmatrix}$$

Where $\vec{\bar{\tau}}$ is the shear stress tensor, and $\vec{q}$ is the three dimensional Fourier heat conduction.

Inviscid fluxes are split into left and right traveling components using Steger-Warming flux splitting. The flux splitting works by solving the eigenvalue problem on the linearized fluxes.

$$\vec{F} \cdot \hat{n} = \frac{\partial(\vec{F} \cdot \hat{n})}{\partial U} U = [A]\, U = [M^{-1}] \Lambda\, [M]\, U$$

Here the fluxes are linearized with respect to conserved variables giving a matrix, $[A]$ that is referred to as the Jacobian. Λ is a diagonal matrix containing the eigenvalues of A. Here, the eigenvalues are the characteristic speeds at which information propagates across the face. Matrices M and M^{-1} transform U to and from characteristic variables.

Based on the sign of the entries of Λ, values for U are taken from either side of the face. The insures quantities that cross the face are computed using values from the proper side.

Boundary conditions are enforced using ghost cells. Ghost cell are cells that are introduced into the domain on the exterior side of a boundary face, with conserved quantities set such that the correct value will be obtained when interpolated to that boundary face from either side.

2.4.2 Structural Membrane Solver

The structural solver used herein is a simple explicit, finite difference membrane code which solves the dynamic equations across membrane elements subject to surface loading and tractions due to linear elasticity.

The membrane is represented as a collection of quadrilateral elements defined by the grid points at their corners. The material properties of Young's modulus (E), Poisson ratio (ν), density (ρ_{mem}) and thickness (h) can be set for each membrane element.

The mass of each element is lumped at its corner nodes, where the displacement and velocity of the structure are defined.

At each iteration, the stress tensor $\sigma_{i,j}$ of each element is calculated as

$$\sigma_{i,j} = 2\mu\varepsilon_{i,j} + \lambda\varepsilon_{k,k}\delta_{i,j}$$

where μ and λ are the first and second Lamé parameters, and $\epsilon_{i,j}$ is the strain tensor within that element.

From the strain tensor the tractions are calculated at each corner of the element. The net force applied to each node is then the sum of the tractions from each of its associated elements, as well as a quarter of the surface load of each of its elements.

This formulation gives the position, momentum, and forces all at the nodes. It is a simple matter then to apply Newton's second law to the nodes, and integrate the system in time.

A damping force can be included for stability as

$$\vec{F}_{damp} = -m_{node}\vec{v}_{node}\frac{1}{t_{damp}}$$

where t_{damp} is a time scale at which you wish the momentum in the structure to decay.

It should be noted that since the traction forces are only in-plane, the membrane solver can be susceptible to out of plane oscillatory errors that can creep in as the solution progresses. Since the velocity of the oscillating nodes changes sign each iteration, the damping term can have an amplifying effect on these errors.

In the development and testing of this solver, it has been found that the introduction of a local low-pass filter to the nodal velocity has the effect of limiting these oscillations while still maintaining the fidelity of the simulation. It also reduces some of the need for the damping term.

The time integration is done via a Runge-Kutta 2 method. The stable time step is computed via the speed of translational wave propagation across the structure.

$$\Delta t_{max} = \frac{\ell}{2C_{mem}}$$

Where ℓ is the length scale in a membrane element, and C_{mem} is computed as

$$C_{mem} = \sqrt{\frac{E}{2\rho_{mem}(1+\nu)}}$$

2.4.3 Modifications for a Deforming Mesh

On wall boundaries, the ghost cells are set so that the interpolated velocity will be zero at the wall face. If the wall is moving this must be changed so that the velocity at the wall face will be that of the wall as determined from the structural solver.

$$\vec{v}_{ghost} = 2\,\vec{v}_{face} - \vec{v}_{cell}$$

When applying a no-slip boundary condition via a ghost cell, it is usually the case that one wishes the velocity at the wall to be zero, in which case the momentum in the ghost cell is the mirror of that in the domain. This is not the case when the wall has a non-zero velocity, and so one must take care that the energy set in the ghost cell is not

just copied from the cell within the domain, but is also adjusted for differences in the amount of kinetic energy on either side of the face.

During a mesh deformation, cells may experience changes in their volume. To maintain conservation of the total quantities of mass, momentum and energy, the intensive quantities with each cell must change as the volume changes.

The correction to each cell due to volumetric changes is:

$$\Delta U_{Vol} = U_o \left(1 - \frac{\Delta Vol}{Vol_f}\right)$$

Where U_o are the intensive variables before the volume correction, and Vol_f is the final volume of the cell.

Since the cell faces can now be moving with their own velocity relative to the laboratory frame in which all variables are stored, there will need to be a change in evaluating the speed at which information is propagating across them.

Instead of evaluating the fluxes as $Flux = C^{-1}\Lambda\, C\, U$, the characteristic speeds Λ must now be shifted by the speed of the face in its normal direction.

$$Flux = C^{-1}\left(\Lambda - I\vec{v}_{face}\cdot\hat{n}\right) C\, U.$$

It must be emphasized that the shift in the characteristic speeds caused by the velocity of the cell faces is the only change made to the inviscid fluxes. Conserved variable remain at all times in the laboratory frame, as do any primitive or characteristic variables used in calculating the fluxes.

The value of $\vec{v}_{face}$ used in the inviscid flux adjustment depends on the coupling method used. In monolithic and weakly coupled systems, the value from the last structural iteration is used, n for monolithic, $m \leq n$ for weakly coupled. However, in a coupling scheme where the structural solver is a sub-step of the flow solver, values for $\vec{v}_{face}$ are available for both n and $n+1$ time iterations. In this case, the integrated value across the time step is used, as it gives a flux Jacobian that is more accurate in time. If we assume that the acceleration of the velocity of the cell face is constant throughout the time step, then;

$$\int_{n}^{n+1} A_{i,j}^{n} - I_{i,j}\vec{v}_{face}(t) \cdot \hat{n} dt = \left(A_{i,j}^{n} - I_{i,j}\vec{v}_{face_{avg}} \cdot \hat{n}\right) \Delta t$$

Since no flux vector splitting is done for the viscous fluxes, no modifications need be made to how they are evaluated. However, the operators used to calculate the gradients required for the viscous fluxes must be updated at each iteration so that the current geometry and distances between cells are always used.

Chapter 3

Algebraic Grid Deformation

3.1 Grid Dependence on a Solution

In a CFD simulation, the flow field is discretized across a grid, or mesh, that spans the domain of interest. Because of this discretization, the quality of the solution that can be obtained depends highly on the quality of the grid.

In an Arbitrary-Lagrangian-Eulerian FSI method, the boundary between structure and fluid can be enforced independent of the boundary of the grid. For this reason, refinement of the grid by clustering points around an interface is necessary for proper resolution of the shape of that interface [11] [12]. In ALE methods, as an interface moves through the domain, grid points must be redistributed so that sufficient resolution can be maintained [13].

In Finite Volume methods, the position of the fluid-structure interface is enforced explicitly by the boundaries of the CFD grid. When the structure moves or changes shape, the boundaries of the grid must move with it. Interior regions of the grid must be able deform to fit within the displaced boundaries. One way this can be done is by re-discretizing the interior and interpolating the solution to the new grid. This is unfavorable for two reasons; re-gridding can be computationally expensive, and interpolating the solution results in a loss of resolution.

An alternative method is to deform the grid by maintaining the same connectivity and displacing the interior grid points in a coherent manner. When the effects of using a moving grid are properly implemented into the flow solver, this method maintains the

quality of the solution, and can be much faster that re-gridding the domain.

3.2 Grid Deformation

Grid deformation can allow for minor to moderate changes in geometry. For significant changes to the geometries of interest, such as the folding of surfaces, large compression of interior regions, or changes to topological structure of the grid, re-gridding becomes necessary. Barring certain failure modes, major geometry changes of these kind are beyond the realm of the normal operation for the devices that are of interest here.

There are different ways of deforming a grid. One of the more ubiquitous methods use for grid deformation is done by representing the grid as a collection of springs [5]. As the boundaries of the mesh change position, the network of springs adjusts to an equilibrium state providing the new grid. However, this method is know to be inadequate in grid regions that contain highly refined wall spacing over convex geometries [15]. In many instances, the converged solution of the spring system results in the crossing of parallel grid lines and an unusable grid.

A more sophisticated method is to represent the grid cells as elastic solids [14]. This ensures that grid lines do not cross over, but is computationally expensive as it requires the solution of a large non-linear system of equations.

Despite an innate bias in favor of grid deformation based on physical systems, there is no reason that a grid deformation scheme needs to be physically meaningful.

Radial basis functions [16] have been put to use to achieve a fast, analytic method for deforming CFD grids. This method uses a subset of surface points to define the motion of a grid boundary. The motion of all other grid points are calculated as a linear combination of displacements calculated from the surface points. This method has proven effective for certain grid deformations, but fails completely for others.

The radial basis function method fails in instances where the displacements of individual points are not coherently related within the coordinate system that is used to represent them. An example of this is the radial expansion of an object when the displacement of grid points is represented in a Cartesian coordinate system. Similarly, this method fails if one translates an object in Cartesian space while representing the grid points in a cylindrical coordinate system. Since inflatable devices can undergo both

expansion and translation, the normal radial basis methods cannot be used.

It would seem that all methods of grid deformations tend to suffer from being either prohibitively computationally expensive, or completely fail for the types of problems that are of interest to the simulation of inflatable, deformable decelerators.

Instead, an approach for creating problem specific algebraic grid deformations will be presented below that sufficiently meets the needs of simulating decelerator fluid structure interactions.

3.3 Benefits of an Algebraic Method

The deformation of a grid at a given instance is completely defined by the displacement of its grid points. In an algebraic method for grid deformation, the displacement of every grid point is calculated analytically from a finite set of parameters. The set of parameters used can vary between applications, but must account for the grid point's position relative to important features in the grid, and the how those features are allowed to change throughout the simulation. By parameterizing displacements with respect to a point's position in space, one can remove any reliance on the connectivity between grid points.

This provides three desirable qualities:

- Mesh independence – A specified algebraic grid deformation should work equally well on a variety of space discretizations. If one wishes to change the computational mesh for a given geometry, the same grid deformation strategy can still be used, independent of whether the mesh is structured, unstructured, or which type of grid elements are used.

- Reproducibility – This follows from mesh independence. It can be assured that if a given deformation works well when tested outside of the simulation, then it will work well once incorporated into the full fluid-structure simulation.

- Speed – Because the displacement of each point is obtained in an analytic manner, there is no additional computational cost incurred solving a large system of equations for the grid deformation. The low computational overhead for the deformation allows it to be implemented at every flow time step without significantly

slowing down the simulation.

The disadvantages to an algebraic method are twofold. The first is that input from the user may be required to parameterized the spatial domain, with *a priori* knowledge of how the object is expected to move and deform. The second disadvantage is a loss of generality in the range of deformations that can be accommodated. This loss of generality is not unique to algebraic methods. Some deformations, such as objects breaking apart as well as other severe changes in geometry can only be accomplished by re-meshing, and are beyond the scope of this work.

Parameters relating to a grid points position can be chosen by a variety of means. Bellow is a selection of deformations of interest, with appropriate parameterizations.

3.4 Rigid Body Motion

If an object is only undergoing rigid body motion (RBM) within the discretized domain, a very simple parameterization can be used.

3.4.1 Domain Parameterization

It is desirable for points near the object to move with the object and for most of grid stretching to be occur in areas away from grid boundaries. To this end, a parameterization of space around the object can be chosen which depends on a points distance from the center of the object.

$$\vec{r} = \vec{x}_{point} - \vec{x}_{object}$$

With the total distance being $d = |\vec{r}|$.

3.4.2 Deformation Parameterization

A rigid body motion centered on an object has both translational and rotational components;

$$\Delta\vec{x} = \vec{\delta}_t + \vec{\delta}_r$$

Where $\vec{\delta}_t$ is the translation component of the displacement, and $\vec{\delta}_r$ is the rotational component. The rotational component is computed with a rotational matrix, A, as:

$$\vec{\delta}_r = \vec{\vec{A}} \cdot \vec{r} - \vec{r}$$

To control how a grid point is displaced for a given deformation, we must combine the parameters in a functional form.

$$\Delta\vec{x}_{point} = f_1(d)\vec{\delta}_t + f_2(d)\vec{\delta}_r$$

Where $f_1(d)$ and $f_2(d)$ are functions that control the effects that the rigid body motion has on points based on the points distance from the object.

The decomposition of deformations into translated and rotated components is the foundation of the grid deformation methods used herein. The method for arriving what those components are for each grid point depends on how sophisticated one's method of controlling the deformation is.

3.4.3 Numerical Formulation

Parameters $\vec{r}$ and d are easily calculated from the grid point's initial position.

Functions f_1 and f_2 depend on the size of the object, and the size of the grid. It is desirable that points within a certain radius move with the object exactly, so $f_1 = f_2 = 1$ for $d < d_{min}$. It is also required that points past a certain distance not to be displaced at all, or $f_1 = f_2 = 0$ for $d > d_{max}$. A good choice is for the translational displacement to decrease linearly away from the region around the object, so choose:

$$f_1(d) = 1 - \frac{(d_{max} - d)}{(d_{max} - d_{min})}$$

Rotational displacements increase linearly with the distance from the center of rotation. To counter this effect, and to maintain an even grid deformation, f_2 is required to decrease faster that f_1. This can be obtained by choosing:

$$f_2(d) = f_1^2(d)$$

3.5 Spline Fit

For rigid body motions where one desires more control over the grid deformations, or to consider the displacement of two bodies relative to each other, it can be useful to use

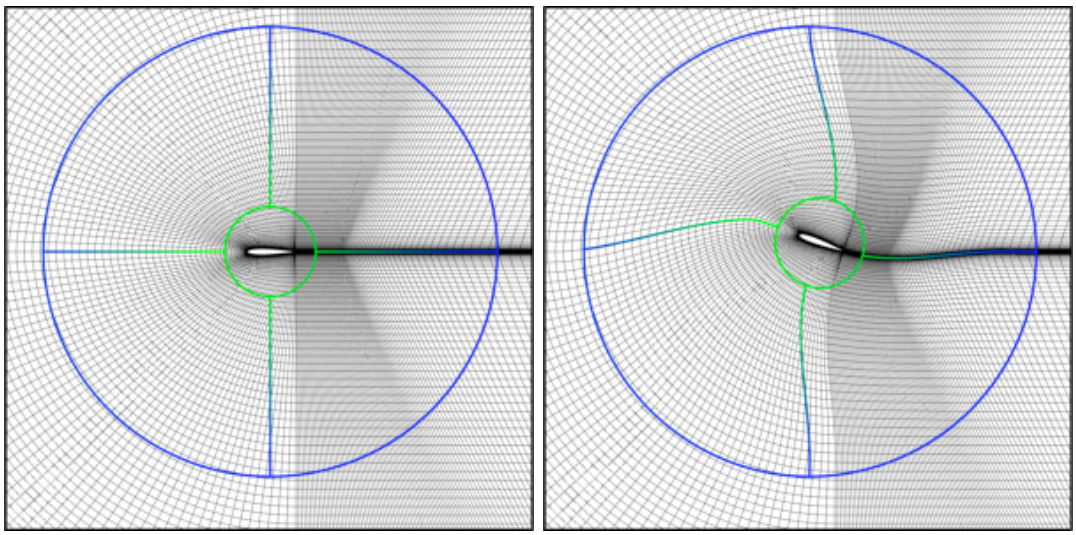

Figure 3.1: Airfoil given an angle of attack and translated to the upper right. Left: initial grid, right: deformed grid.

a multi-part spline curve to drive the deformation. A spline is a smooth curve that is defined as a piecewise set of polynomial curves. By using splines a smooth curve can be obtained by specifying control points that the curve is to pass though and the desired slope at those control points.

3.5.1 Domain Parameterization

A spline curve is placed from one end of the domain to the other, passing through each object of interest. For each position along the spline, a plane can be defined that intersects and is orthogonal (or nearly orthogonal) to the spline at that point. As long as the spline remains smooth, any point within the domain can be found on one of these planes. Smoothness is therefore a requirement in choosing the initial spline. Given an initial spline, the domain is then parameterized by the position along the spline, s, at which a plane containing the grid point intersects with the spline, and by the vector distance, r, from that intersection point.

If the initial spline is not a straight line, there may be some redundancies in the parameterization, i.e. a point may lay on more than one spline-defined plane. If this is the case, then the amount of the domain that can reliably deformed is limited to within

the radius of curvature of the initial spline.

3.5.2 Deformation Parameterization

As an object is displaced, the control points immediately fore and aft of it are moved with the object via a Rigid Body motion. This will change the shape of the spline. As the spline changes shape, the planes that intersect the spline will be made to move, and rotate with the intersection point that they are associated with. The spline control points are therefore the parameters that control the deformations, and set via each object's rigid body motion.

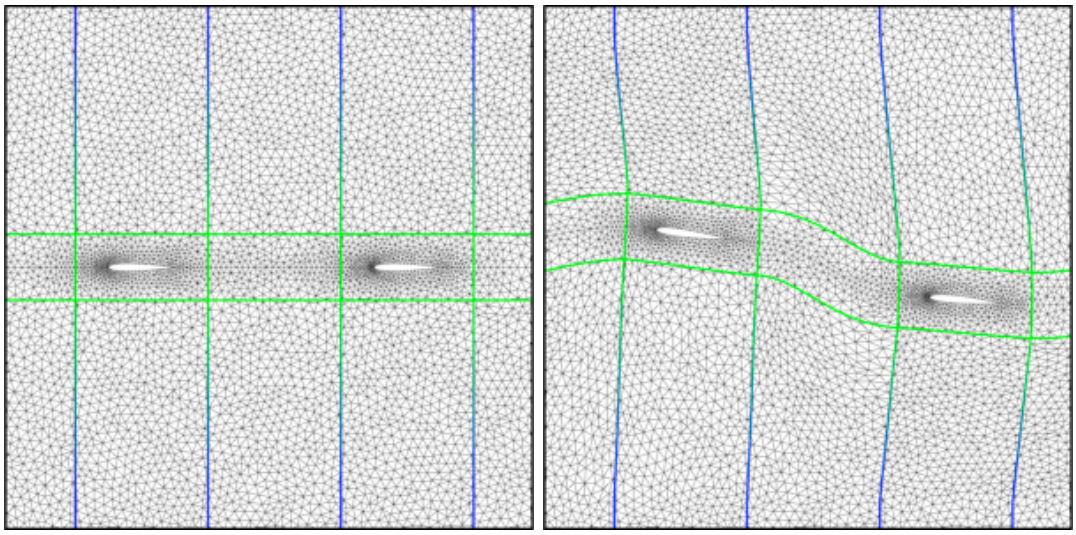

Figure 3.2: Two airfoils displaced with a spline passing though them.

3.5.3 Numerical Formulation

Each grid point must be associated with a point along the spline. This can be most easily be done by finding the point on the spline that has a minimum distance to the grid point. If the initial spline is a straight line, this is a trivial matter.

The displacement of grid points are calculated from the change in shape of the guiding spline.

$$\Delta\vec{x}_{point} = f_1(d)\vec{\delta}_{spline} + f_2(d)\vec{\delta}_r$$

Where $\vec{\delta}_{spline}$ is the displacement of the spline point with which the grid point is associated. $\vec{\delta}_r$ is the rotational displacement caused by the change in the angle of the spline at the intersection point, $f_1(d)$ and $f_2(d)$ are similar die-off functions to those defined for a rigid body motion.

3.6 Embedded Control Surface

The above methods only provide large scale, coarse grid deformations. For finer deformations it is necessary to be able to connect the domain parameterization directly to the geometry of an object's surface. This can be accomplished by replacing the large scale rigid body motions of the previous methods with a continuous set of RBMs defined on a control surface. Grid point displacements are calculated from an integral of the surface deformation with a distance based weighting function.

3.6.1 Domain Parameterization

For each grid point, $\vec{x}_i$, and surface point, $\vec{s}_j$, we have $\vec{r}_{i,j} = \vec{x}_i - \vec{s}_j$, and $d_{i,j} = |\vec{r}_{i,j}|$.

3.6.2 Deformation Parameterization

As in the previous methods, the displacement of a given grid point has both a translational and a rotational component. $\Delta\vec{x} = \vec{\delta}_t + \vec{\delta}_r$. In this deformation method the components are evaluated as an integral over the control surface. For each grid point,

$$\vec{\delta}_t = \int_{surface} f_1(d_{i,j})\, \vec{\delta}_t(s_j)\, dArea$$

$$\vec{\delta}_r = \int_{surface} f_2(d_{i,j})\ [A(s_j) \cdot \vec{r}_{i,j} - \vec{r}_{i,j}]\ dArea$$

Functions f_1 and f_2 must be normalized over the surface;

$$\int_{surface} f_{\{1,2\}}(d)\, dArea = 1$$

and have asymptotic behavior such that;

$$\lim_{d\to 0} f_{\{1,2\}}(d) = \infty, \quad \lim_{d\to\infty} f_{\{1,2\}}(d) = 0$$

This asymptotic behavior ensures that the integrals for grid points at or near the surface are dominated by the contributions of the part of the surface that they are closest to.

3.6.3 Numerical Formulation

The quality, and cost effectiveness of this method depends on the discretization of the surface integral. Restricting the integral to only the closest area of the surface can greatly reduce the number of operations that must be computed per grid point for a given control surface.

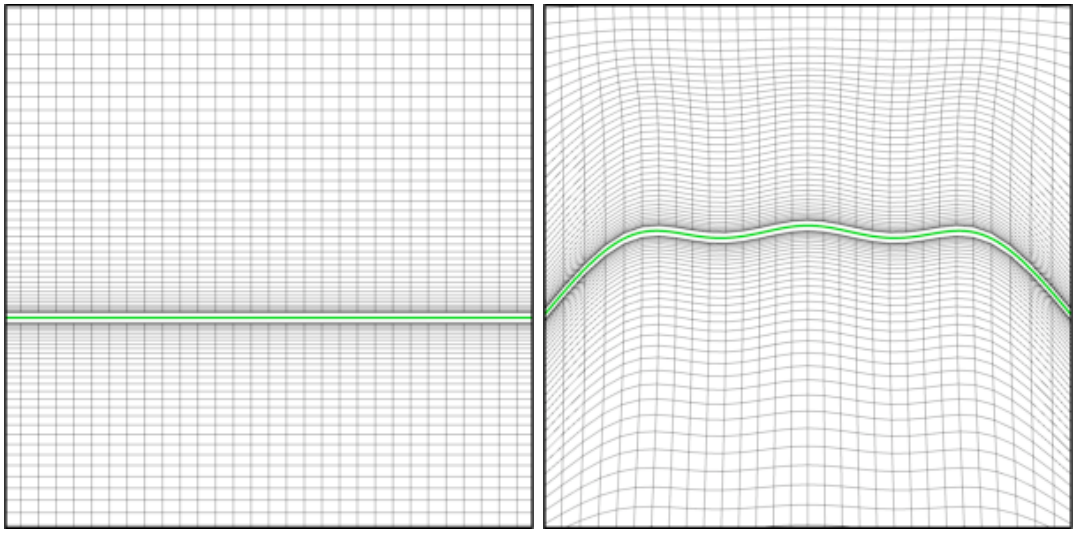

Figure 3.3: Surface driven grid deformation.

3.7 Shaping

There may be an area in the domain where one wishes to preserve the grid without deformation, such as areas where shock alignment has been done, or areas away from the deforming body. In these case it is desirable to have the deformations decrease to zero in some buffer region. This can achieved by introducing a shaping weighting

function that smoothly varies from 0 in regions where no deformation is desired, to 1 where the grid is free to deform according to which ever scheme is used.

These shaping functions would be similar to the weighting functions introduced for rigid body motion, and would be applied after the nominal deformation is calculated. It should be cautioned that the distance over which the shaping function changes from 1.0 to 0.0 will affect how much deformation can be accommodated without grid folding. The shaping regions should therefore be made to as gradual as possible.

Chapter 4

Validation

4.1 Introduction

Three validation cases are presented to test the modifications applied to the flow solver, the behavior of the structural solver, and a heating model. The first test is the flow caused by an accelerating piston. The flow in this 1-D problem is driven completely by the moving cell faces and the wall boundary condition. Correspondence between the analytical solution for flow properties and those computed by the flow solver is to check that the boundary conditions are properly applied and that the modifications to the inviscid flux evaluation are correct. The second test is a test of the time-dependent response of the structural solver to loading from the flow solver. The combination of validation of both fluid side response and structural side response should provide confidence in the simulation of complex combined interactions. The third validation test is of a 1-D heat transfer model for the time-dependent simulation of the structural temperature, and is compared to a wind tunnel test of aero-thermal heating on a candidate material for use on a high altitude inflatable decelerator.

4.2 Constant Acceleration Piston

The effects caused by the structural system on the fluid system is fully defined by a moving boundary condition. If the flow solver can be shown to be able to handle a wall boundary that is accelerating towards and away from the fluid, then this can

be considered as validation of the changes made to the flow solver to accommodate deforming geometries. As a validation case, consider the flow ahead of a piston that is either accelerating into, or away from a compressible fluid.

4.2.1 Analytic Solution

The analytical solution [17] for the velocity ahead/behind of a piston undergoing constant acceleration is:

$$u(x,t) = -\frac{1}{\gamma}(c_0 - \frac{\gamma+1}{2}a_P t) + \sqrt{\frac{1}{\gamma^2}(c_0 - \frac{\gamma+1}{2}a_P t)^2 - \frac{2}{\gamma}a_P(x - c_0 t)},$$

with the local sound speed given by:

$$c(x,t) = c_0 + \frac{\gamma+1}{2}u.$$

Where c_0 is the initial speed of sound of the undisturbed fluid, and a_P is the acceleration of the piston. These equations assume isentropic compression, and are valid up to shock formation.

4.2.2 Results

The acceleration of the piston is represented in the simulation by moving the entire computational domain with the piston. This has the advantage of simultaneously testing the implementation of the boundary condition, and verifying that no non-physical effects are introduced when the domain is moved through a stationary fluid.

Plots comparing the analytical solution to that obtained via simulation are provided in figures 4.1 and 4.2, with velocity and sound speed shown at 0.5 second time intervals.

In each case the line plots for both the simulation and analytic solutions lay on top of each other. The only noticeable difference is a slight diffusion seen in the simulation where the pressure front comes into contact with the undisturbed fluid. This is caused by the motion of the front across many cell faces as to solution progresses, and is an unavoidable effect of discretization.

It can be concluded by the overlapping of the profiles obtained via analytic and simulation methods, for velocity and sound speed, under both the compressing, and receding piston cases, that the changes to the flow solver are sufficient to correctly represent any

time dependency to the wall boundary conditions, as dictated by the structural solver.

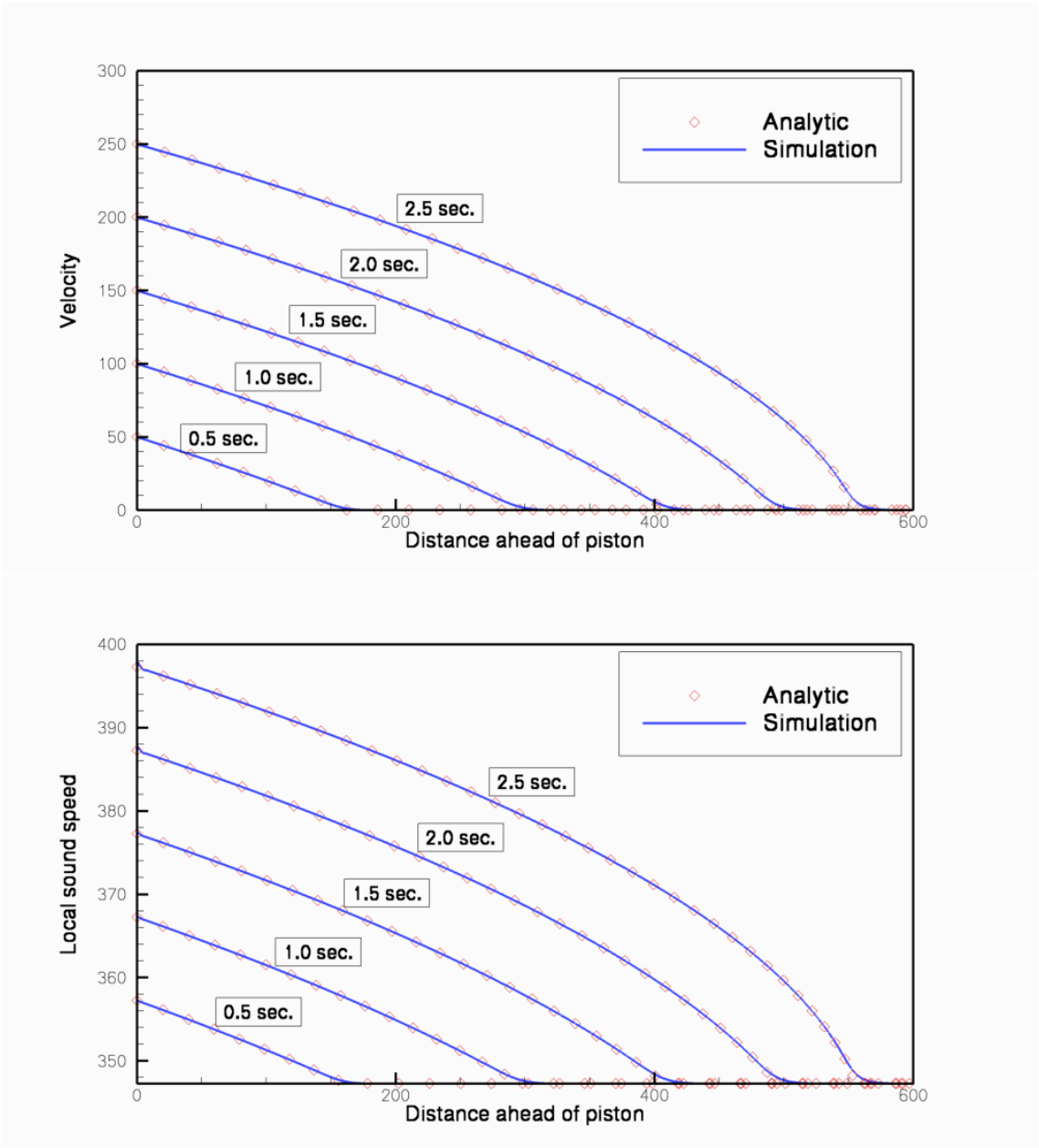

Figure 4.1: Velocity and speed of sound curves ahead of a compressing piston at multiple time intervals

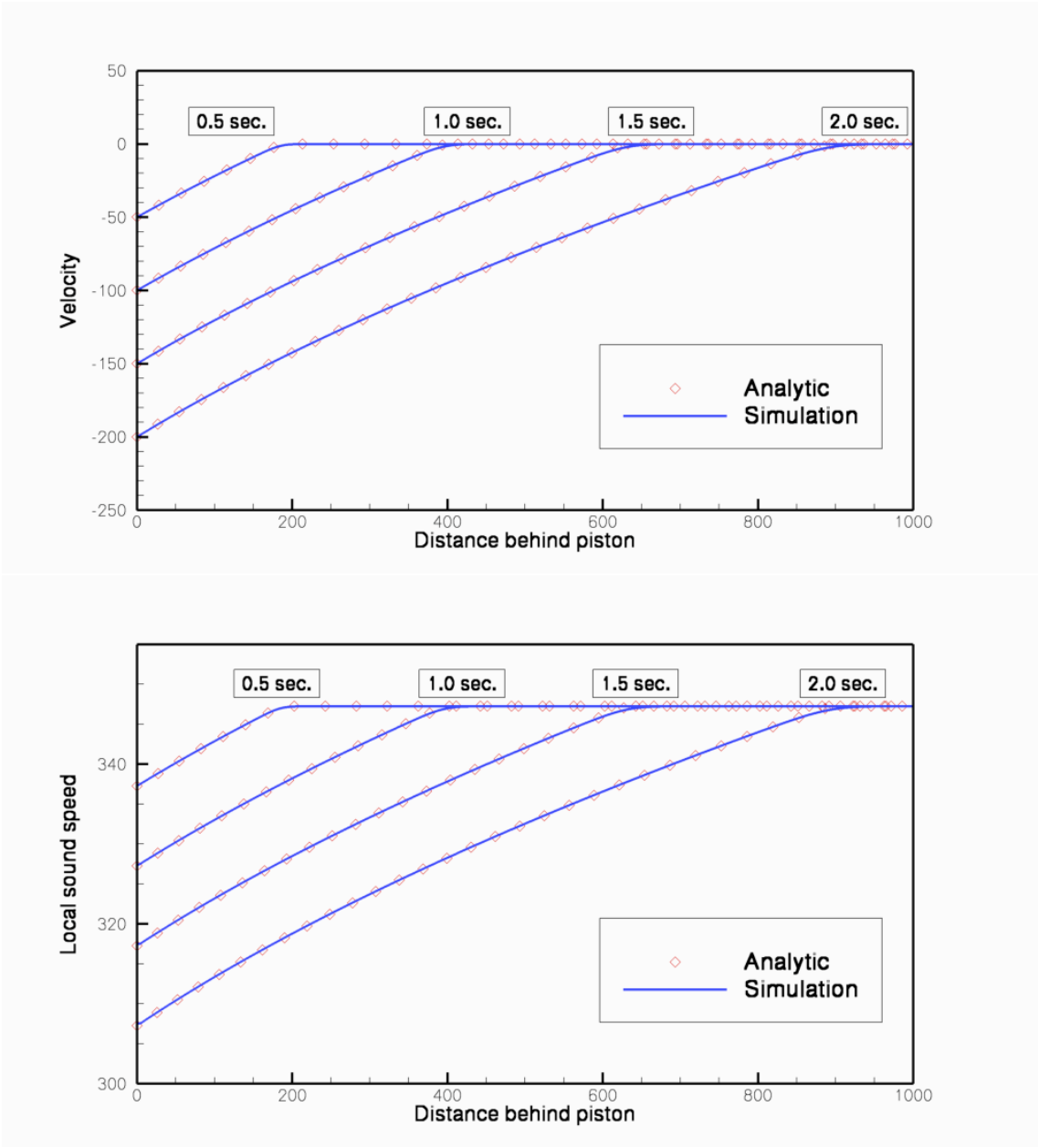

Figure 4.2: Velocity and speed of sound curves behind a receding piston at multiple time intervals

4.3 Inflation of an Elastic Membrane

The effects of the fluid on the structure lie in the forces applied to the structure by the fluid, as well as aero-thermal heating. To validate the time dependent behavior of the structural solver under changing force loads, the solver's behavior is compared to an analytic solution for inflating elastic cylinder. Some assumptions, such as acoustic waves coming off of the structure, must be made to arrive at a form for which a solution may be obtained.

4.3.1 Analytic Solution

Consider a cylinder made of an elastic membrane of initial, unstrained radius r_0 and thickness h. Under load caused by a difference of pressure, ΔP, between the inner and outer surfaces, the radius of the cylinder will expand due to the pressure load until the pressure difference is countered by the traction force due to strain. The change in the radial momentum of the cylinder, per unit length, and radian angle, is given by

$$\rho_{mem} h \frac{d^2 r}{dt^2} = \Delta P r - E h \frac{r - r_0}{r_0}.$$

Where ρ_{mem} is the density of the membrane at its unstrained configuration and E is the Young's modulus of the material. When the cylinder is released from its unstressed initial position, it will over-expand to a maximum radius until the elastic tractions have countered the momentum gained during expansion, at which point the cylinder will contract back to the unstressed radius. The cylinder oscillates between these points, giving an amplitude and a frequency to compare to the solver. For a constant pressure load irrespective of the change in radius, the pressure is a $\Delta P = P_{in} - P_{out}$. For an internal pressure that varies with the radius of the cylinder, we assume isentropic expansion and compression to get

$$\Delta P = P_{in_0} \left(\frac{r_0}{r} \right)^{2\gamma} - P_{out}$$

The constant pressure case reduces to that of a simple harmonic oscillator, where the radial dependent pressure case is more complicated. For simplicity, solutions for both are obtained via simple numerical integration in time, with sufficiently small time steps.

The solutions are then compared to the average radius of the the cylinder elements obtained via the structural solver.

4.3.2 Results

The structural solver was run with a damping time of $t_{damp} = 0.4sec.$, and was able to correctly capture both the frequency of the oscillations, as well as the amplitude. As the solver progressed, the damping caused the amplitude of the oscillations to gradually decrease, bringing the cylinder towards an equilibrium position where the elastic traction forces are balanced by the pressure difference.

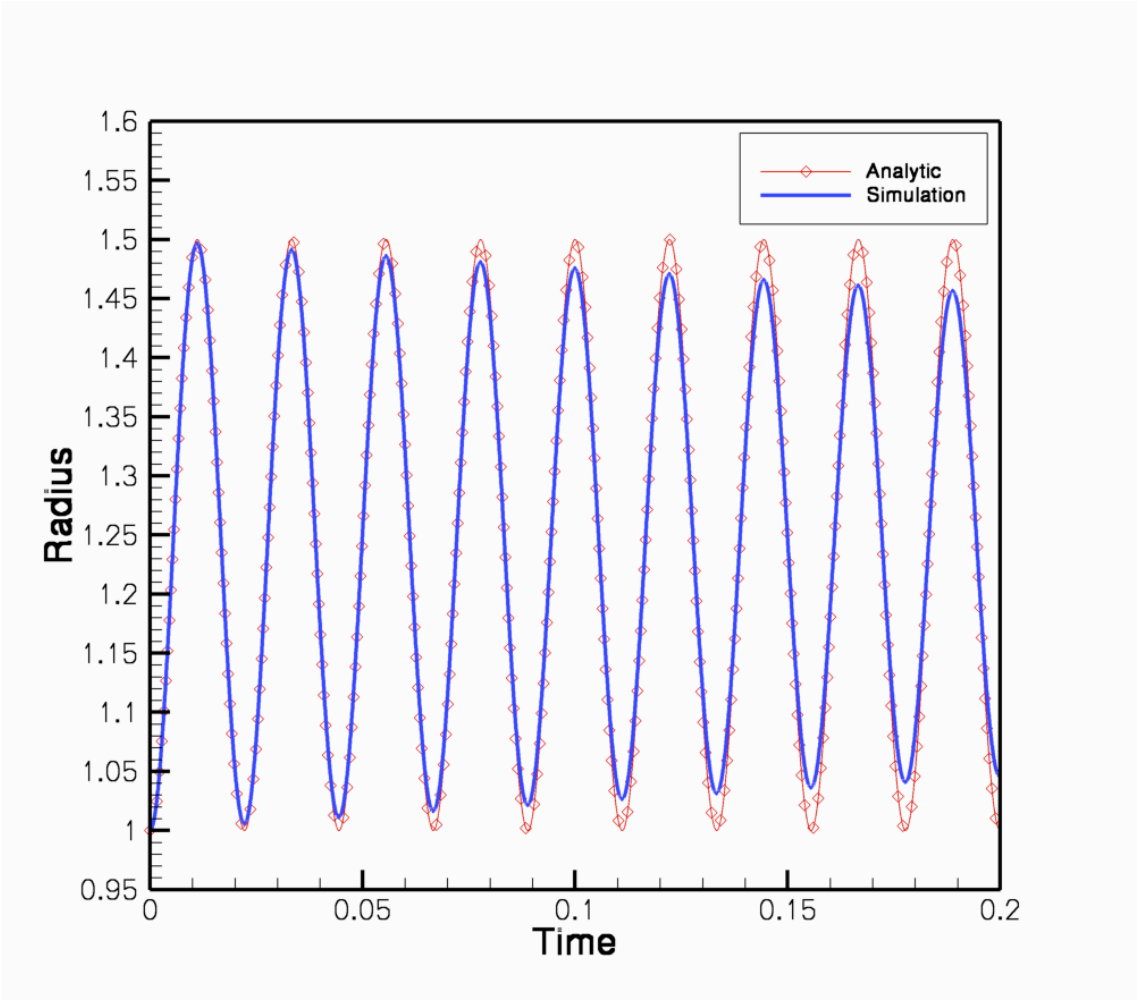

Figure 4.3: Radius of an oscillating cylinder under a constant pressure difference with respect to time.

The effect of the damping was more pronounced in the constant pressure case than in the variable pressure case. In both cases, the frequency of the oscillations, as well as the maximum amplitude that the radius reaches where captured with negligible error. The radius can be seen repeatedly to pass through the equilibrium point at each oscillation at the same time as the analytic solution.

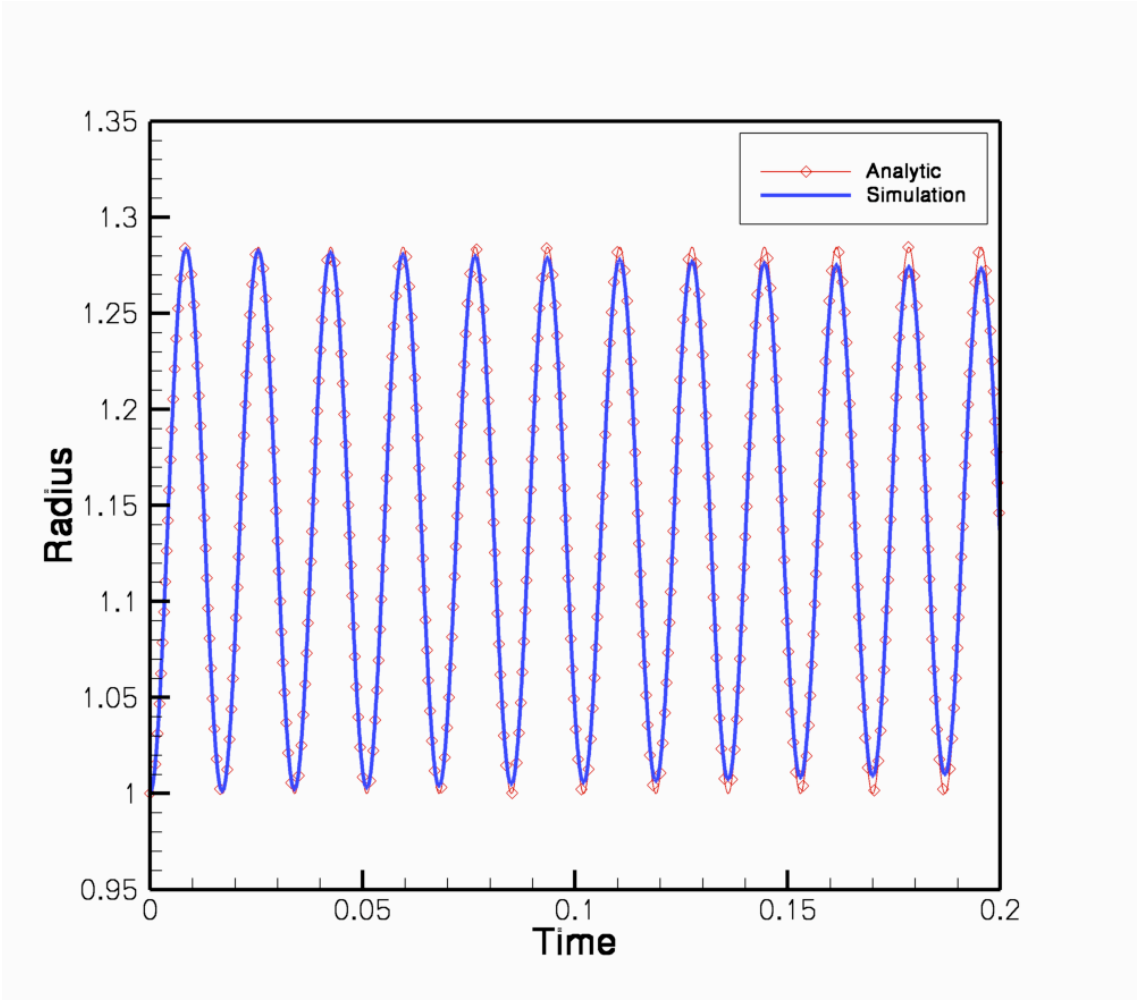

Figure 4.4: Radius of an oscillating cylinder under a variable pressure difference with respect to time.

By necessity, the changes in pressure due to the formation of acoustic waves at the cylinder's surface have been neglected, as they would introduce more complicated terms to the governing differential equations, such as a dependence on the acceleration and velocity of the radius instead of just the magnitude. However, the ability of the structural solver to correctly capture the time dependent dynamics of this simplified model suggests that as long as the simulations progress at a time step that reflects

the minimum of the stable time steps os the fluid and the structural systems, then the coupling system should also be time accurate.

4.4 1-D Heating

It is possible that the material used in an inflatable decelerator may increase in temperature during deployment to a point where there is a change in the material properties or a high chance of failure. To this end, the capability is required to include aero-thermal effects from the fluid on the structure.

4.4.1 Heat Equation

For each structural element, a temperature profile is maintained through the thickness of the material. The temperature at each point within the material is updated via the one dimensional heat equation;

$$\frac{dT(x)}{dt} = \frac{1}{\rho_{mem} c_p} \frac{dq(x)}{dx}$$

where ρ_{mem} is the density of the membrane material, and c_p is the specific heat of that material. Multiplied together, they give a volumetric specific heat. The internal heat flux rate, $q(x)$, is calculated from the temperature gradient as;

$$q(x) = -k \frac{dT(x)}{dx}$$

with k as the heat transfer coefficient for the material. ρ_{mem}, c_p, and k can all varies through the thickness of the material, providing the ability to model material layups. When modeling a material layup, it may be necessary to consider non-ideal heat conduction between layers. This is accomplished by inserting a contact resistance at the boundary between layers. At a such a boundary, there is a discontinuity in the temperature, with the heat flux using the form;

$$q_{cont}(x) = -\kappa_{cont} \left(T(x^+) - T(x^-) \right)$$

where κ_{cont} is a constant of conductivity, and $T(x^{+/-})$ is the temperature on either side of the discontinuity.

The temperature profile is driven via the heat flux at the top and bottom surfaces of the structural element, as provided to the 1-D heating model by the flow solver. Radiative loss, and gain can also be included via an adjustment of the boundary heat flux.

4.4.2 Experiment

As part of the PAIDAE [18] program at the NASA Langley Research Center, Del Corso et al. [19], [20], performed a set of wind tunnel experiments to test the survivability of various material layups when subjected to aero-thermal heating.

As a validation case, the heat transfer rates from one of the layup tests are modeled using the 1-D heat transfer model. The resulting intra-layer temperatures are compared between the simulation and when was recorded by the thermocouples in the experiment.

The layup under consideration, layup 2 in [19], consists of a top layer of Nextel, followed by a Pyrogel layer, and then two thin Kapton layers.

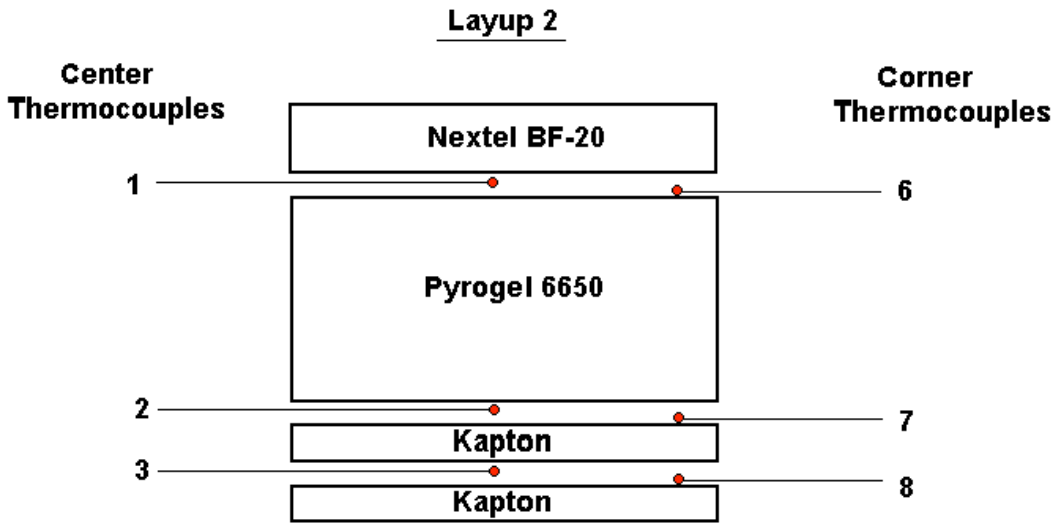

Figure 4.5: Material layup for validation of 1-D heat transfer model. Only the center thermocouples 1, 2, 3, are compared.

The applied heat flux to the top surface was 3.1 W/cm^2, at 300 K. As the temperature of the top surface increased, the applied heat flux is scaled down linearly as

$$q_{in} = 3.1 \left(\frac{1750K - T_{top}}{1750K - 300K} \right) W/cm^2$$

The heat load was applied for 90 seconds during which time the sample increased in temperature. After the thermal load was removed, heat continued to pass through the sample and out the back surface.

4.4.3 Simulation Results

Though it is claimed in [19] that the only loss of heat through the back face is due to radiation to ambient, this is obviously an underestimate. It is clear from figures 4.6 and 4.7 that the majority of cooling of the material occurred via conduction out of the bottom surface as the top surface remains hotter at all times than the lower surfaces, thus heat could only be traveling from top to bottom.

Both simulations with and without the contact resistance included accurately captured the changes in temperature between the top and bottom layers of the layup. When assuming perfect conduction between layers (Figure 4.6), the 1-D heat transfer model was not able to distinguish differences between the temperatures across the different Kapton layers. Inclusion of contact resistance did allow for differences in the temperature across the Kapton layers to be resolved. This suggest that the thermal resistance across those layers is almost entirely due to the imperfect thermal contact.

Figure 4.8 shows the temperature across the thickness of the layup after 90 seconds of heating. Note the drop in the temperature at each of the contact discontinuities.

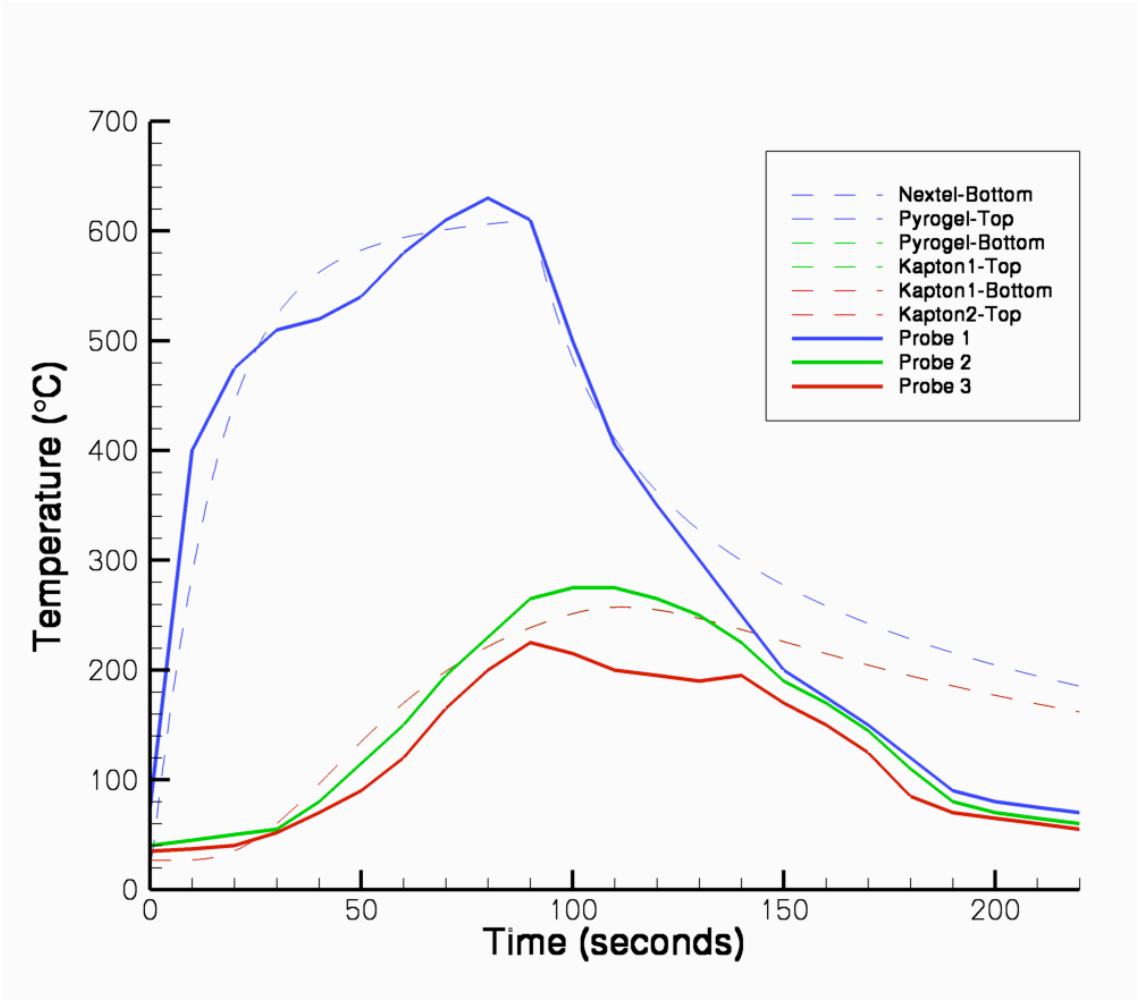

Figure 4.6: Temperature as a function of time at different points within the material layup, with perfect conduction between layers. Solid lines are from the experiment, dashed lines are from the simulation.

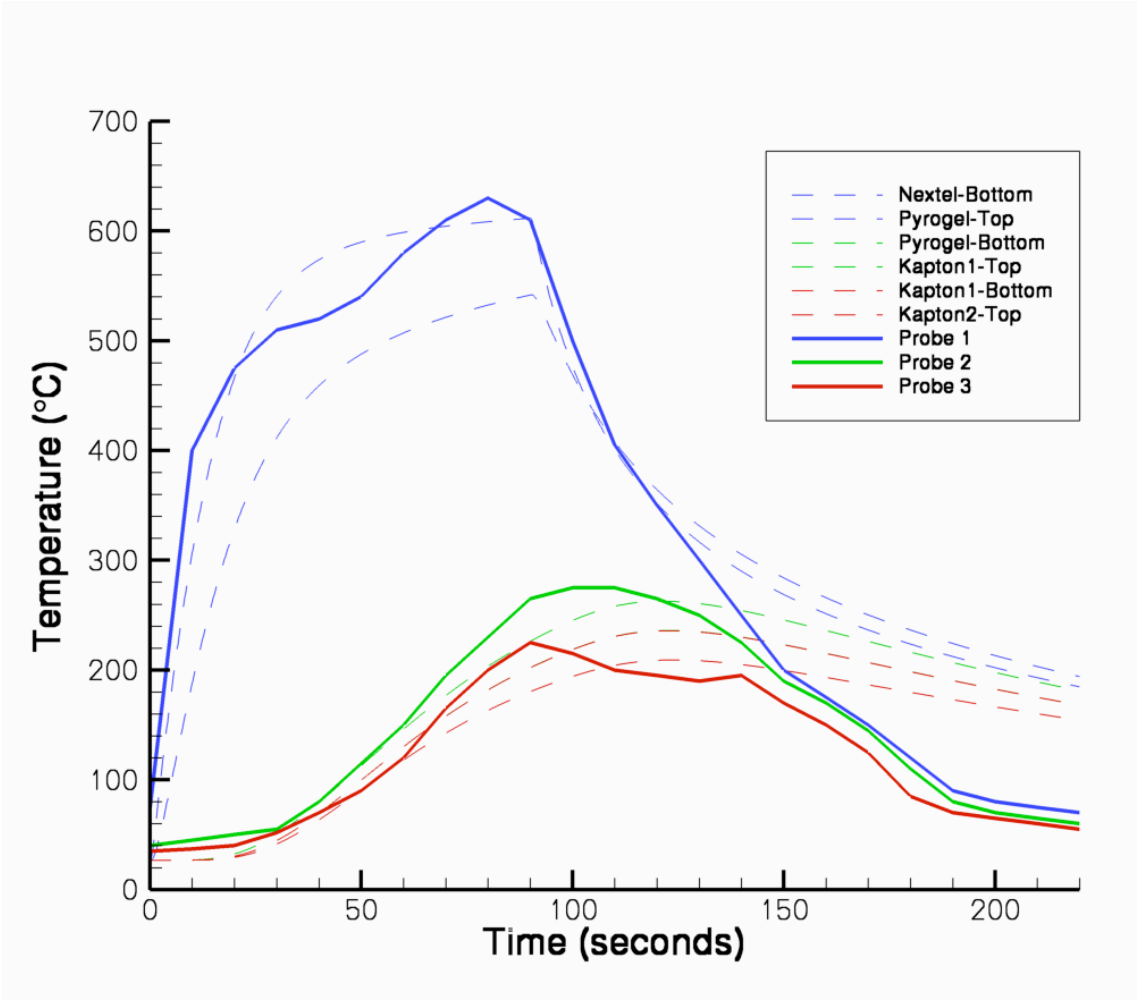

Figure 4.7: Temperature as a function of time at different points within the material layup, with contact resistance between layers. Solid lines are from the experiment, dashed lines are from the simulation.

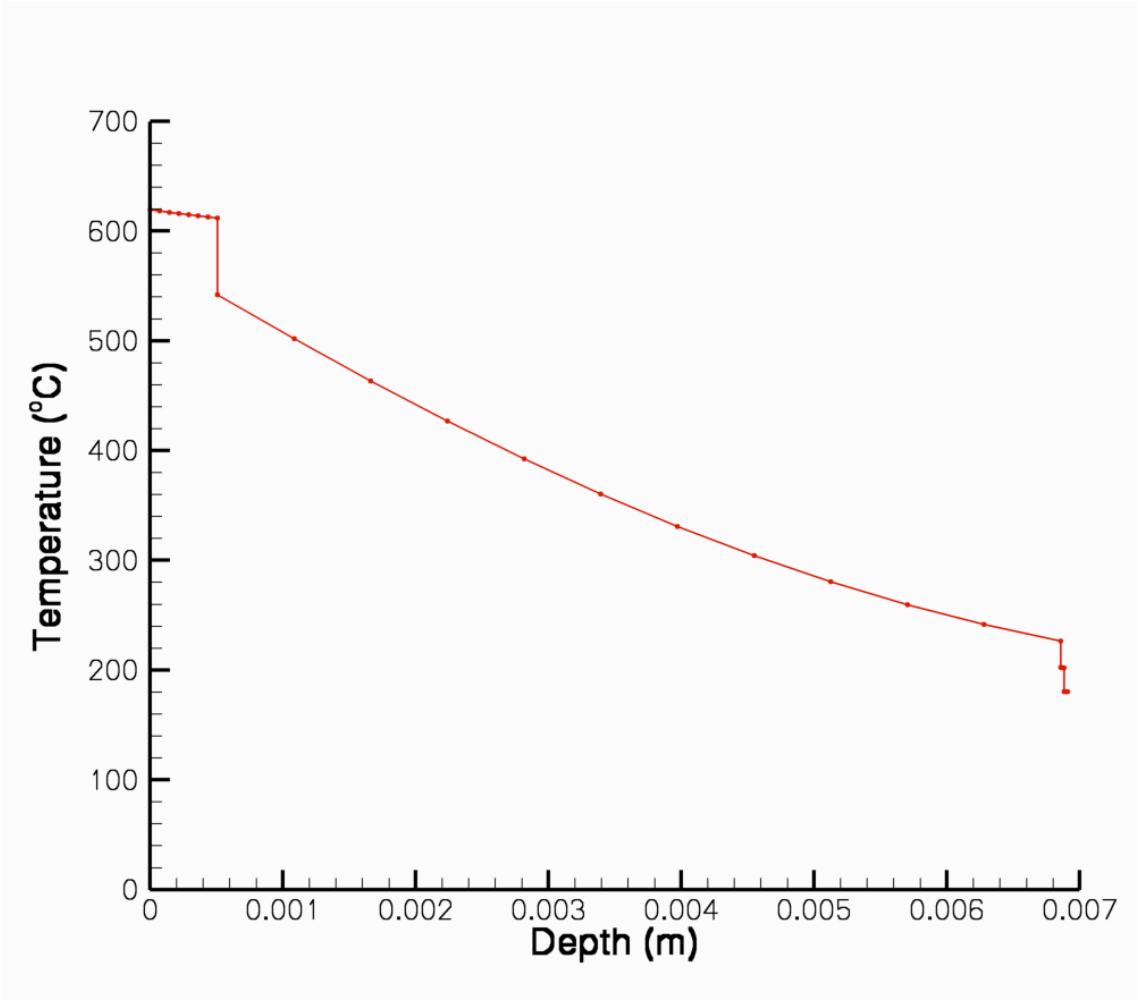

Figure 4.8: Temperature as a function of distance into the material layup, with contact resistance between layers.

Chapter 5

Disk-Gap-Band Parachute

Disk-Gap-Band (DGB) Parachutes begin to undergo highly unsteady dynamics and violent forces at speeds of the order of Mach 2 [21]. At these speeds the pull angle of the parachute oscillates off axis and the canopy can experiences partial collapse, decreasing its effective area. The predominant cause of this chaotic reaction to the flow is due to a pressurization cycle [22] that occurs in the bow shock ahead of the canopy. This pressurization cycle is caused by the shock expanding into locally subsonic regions in the wake of the main craft, then collapsing back onto the canopy.

5.1 MSL Parachute Validation

A DGB parachute is to be used as part of the entry-descent-landing phase of the upcoming NASA Mars Science Laboratory (MSL) mission. Because entry conditions for MSL are to be faster, at higher altitude, and with a heavier payload than previous Mars missions, extensive testing was done to qualify the parachute for the mission [23], [24], [25], [26], including simulation studies [27], [28], some results of which are included here.

5.1.1 Compression Cycle

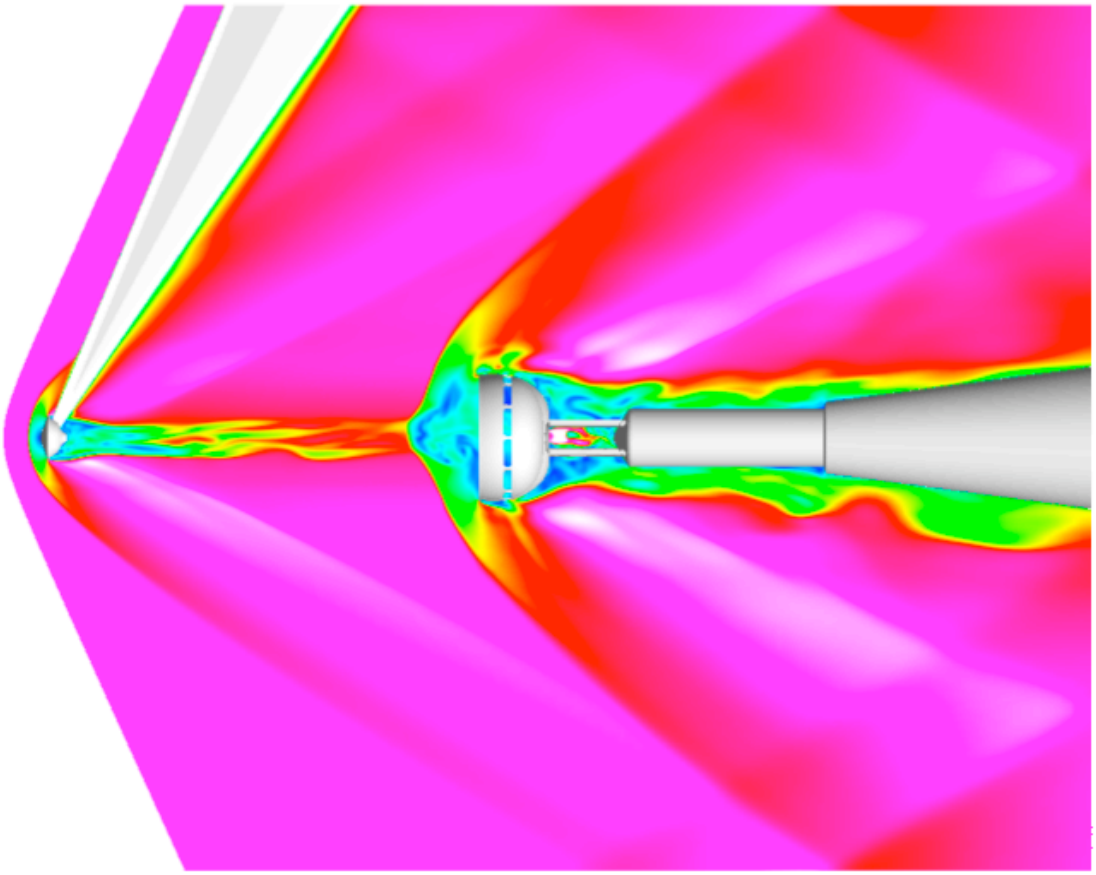

Figure 5.1: Flow structure of capsule-canopy configuration in center plane of 3-D flow field, colored by Mach number.[22]

Figure 5.1 shows Mach contours from a simulation of rigid canopy wind tunnel test that was performed in the NASA Ames 9x7 tunnel [22]. One can see packets of subsonic flow, being shed by the capsule and proceeding downstream into the bow shock that forms ahead of the canopy. The bow shock is able to grow forward into these subsonic packets, only to collapse back onto the canopy.

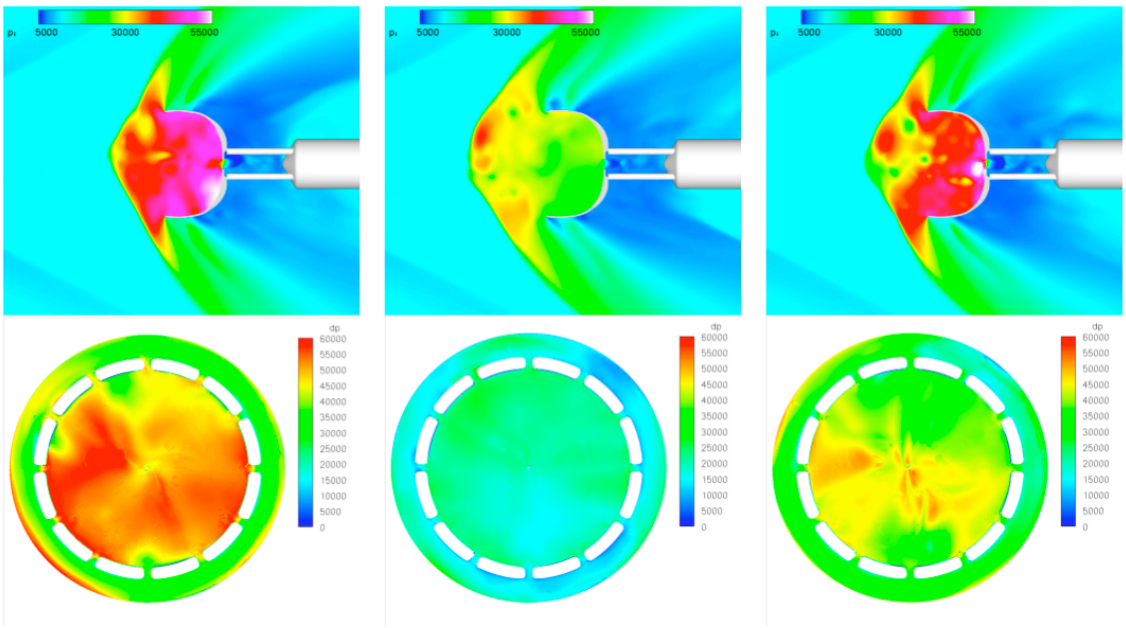

Figure 5.2: Image series depicting a parachute pressurization cycle. Top images show pressure contours through domain symmetry plane. Bottom images show an exploded polar view of the pressure differential across the canopy surface.[22]

Figure 5.2 shows the changes in the pressure of the fluid captured in the canopy as the growing and collapsing of the canopy bow shock continues. As the shock moves into the capsules wake, streamlines are diverted around the highly pressurized canopy. This diverts fluid away from the canopy, lowering the pressure within, and collapsing the shock. The canopy is able to then re-pressurize, growing the shock back out, and continuing the cycle.

5.1.2 Experiments

Sub-scale, supersonic wind tunnel experiments where performed at three Mach numbers with a flexible canopy. The runs that are examined in comparison to simulations are those where the canopy was constrained at zero angle of attack and those where it was unconstrained and free to move.

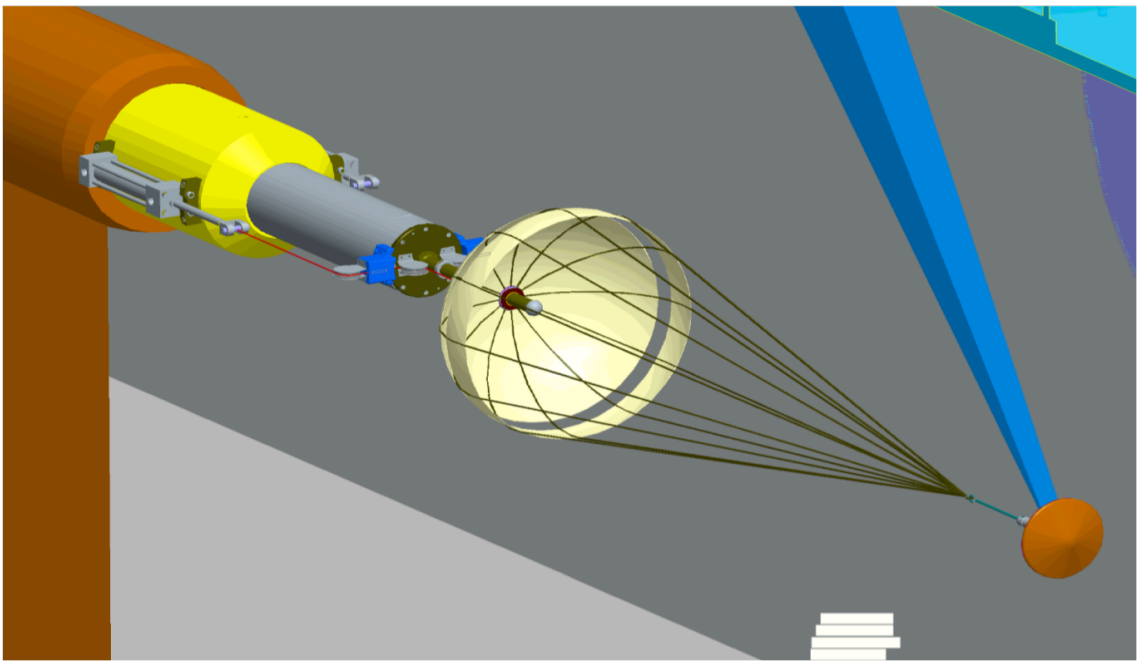

Figure 5.3: A illustration of the experimental setup of MSL capsule with a constrained flexible parachute in the NASA Glenn 10x10 tunnel.[26]

The supersonic wind tunnel tests where conducted in the 10x10 tunnel at NASA Glenn for the purpose of providing design feedback as well as validation data for CFD [26]. Figure 5.3 shows a diagram of the setup in the Glenn 10x10 tunnel. Two classes of tests are examined involving a flexible canopy. The first set has the canopy constrained to a zero degree pull angle, and allowed to deform in place. The second set allows the parachute, unconstrained motion. The conditions for both sets of simulations are given in Table 5.1.

Mach number	2.0	2.2	2.5
Re ($\times 10^6$)	0.77	1.02	1.23
Q (kPa)	17.4	18.6	19.9
$D_0(m)$	0.813	0.813	0.813

Table 5.1: The run conditions for the flexible canopy wind tunnel tests.

5.1.3 Simulation

The parachute was modeled as a static shape that could undergo motion in two degrees of freedom, corresponding to yaw, and pitch pull angles. Pressure and viscous forces are integrated across the canopy surface, and used to update the canopy's velocity and pull angle.

$$I\ddot{\vec{\omega}}_{pull} = \vec{r}_{pivot} \times \vec{F}_{canopy},$$

where $\vec{F}_{canopy}$ is the aerodynamic force on the canopy, $\vec{r}_{pivot}$ is the distance vector between the center of mass of the canopy, and the point at which it is connected to the capsule, and $I\ddot{\vec{\omega}}_{pull}$ is the second time derivative of the pull angle times moment of inertial about the pivot point. At each flow time step, $\ddot{\vec{\omega}}_{pull}$ is calculated and used to update the velocity and position of the canopy.

The grid deformation used was that described in Section 3.3 with the stationary capsule acting as one anchor point to the guiding spline and the displaced canopy acting as the other. An image of the deformed grid with the canopy off axis is shown in Figure 5.4 .

A key component to simulating the unsteady dynamics of the parachute is proper resolution of the wake region between the aeroshell and the parachute. If the eddies being shed by the capsule are not resolved sufficiently, the canopy pressurization cycle will not be captured, and there will be nothing to trigger, and drive the unsteady behavior.

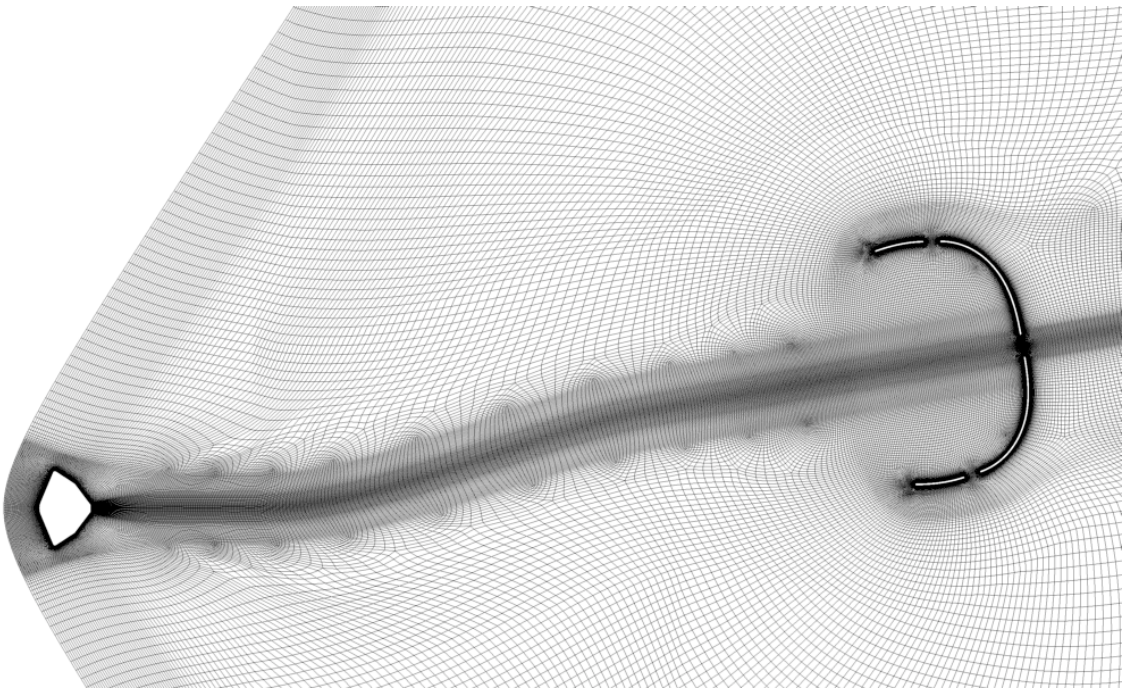

Figure 5.4: An example of the spline based deformation of the parachute grid.

5.1.4 Results

The results from computational simulations of the flexible test case are consistent with the measurements from the wind tunnel test performed at the Glenn 10x10 tunnel, and with the expected trend found from Viking sub-scale tests [29]. Plots of the coefficients of drag for the constrained simulated and experiment runs are compared in Figure 5.6. Table 5.2 provides numerical comparison of drag coefficients as well as the standard deviation in the instantaneous drag. Peak and minimum drag measurements as a percentage of the RMS drag are also provided.

	C_D	Std Dev (%RMS)	Max Drag (%RMS)	Min Drag (% RMS)
Mach 2.00 Experiment	0.464	12.1	158	19
Mach 2.00 Simulation	0.472	19.8	172	30
Mach 2.18 Experiment	0.449	12.3	159	31
Mach 2.2 Simulation	0.427	25.3	205	33
Mach 2.47 Experiment	0.386	18.9	217	7.9
Mach 2.50 Experiment	0.390	26.6	202	33

Table 5.2: Drag coefficients for constrained parachute test.

The drag force in both experimental and simulated flows has a very high degree of variation at frequencies characteristic of the over- and under-pressurization cycles observed in the rigid model tests. These variations had slightly larger amplitude in the simulation, the standard deviation of instantaneous drag measurements in the simulation being 20-25 % of RMS drag, and those of the experiment 12-20 % of RMS drag. Both experiments and simulations yielded spikes in drag with amplitudes near or in excess of 200 % of the mean value. The amplitudes of these spikes in drag are significant, as they may be a factor leading to parachute failure.

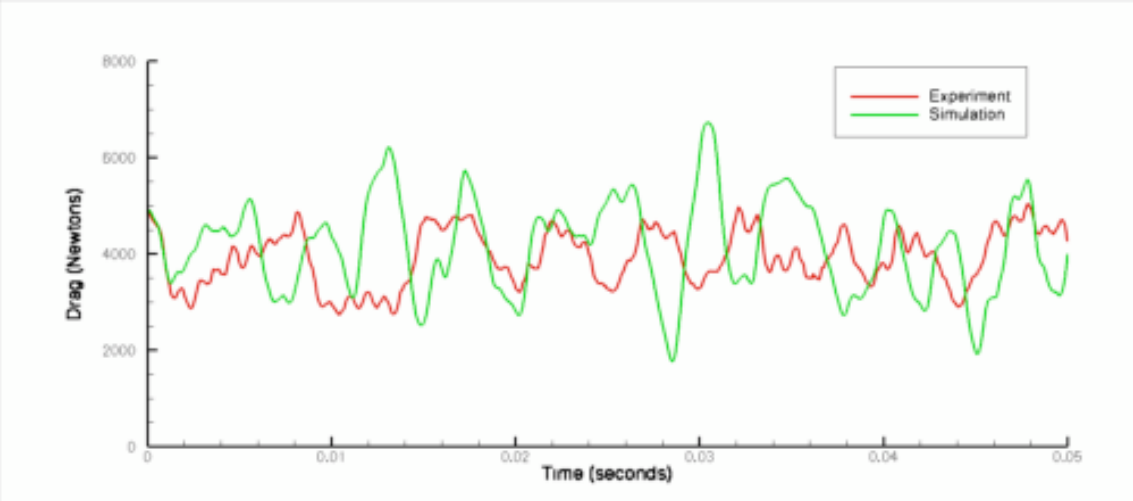

Figure 5.5: Time variations in the Coefficients of Drag from the unconstrained wind tunnel tests.

Figure 5.5 compares drag variations from the experimental and simulated tests of the constrained model at Mach 2.0 over an interval of 0.05 seconds. The large scale drag variations in both the experimental data and the simulated values have similar durations, however the peaks from the experiment are often interrupted, or truncated by shorter period oscillations. These shorter period oscillations are likely due to the deformation of the flexible canopy. An ability of the flexible canopy to spill out pressure buildups earlier, or by degrees throughout the pressurization cycles would explain this difference in the severity of variation between experiment and simulation.

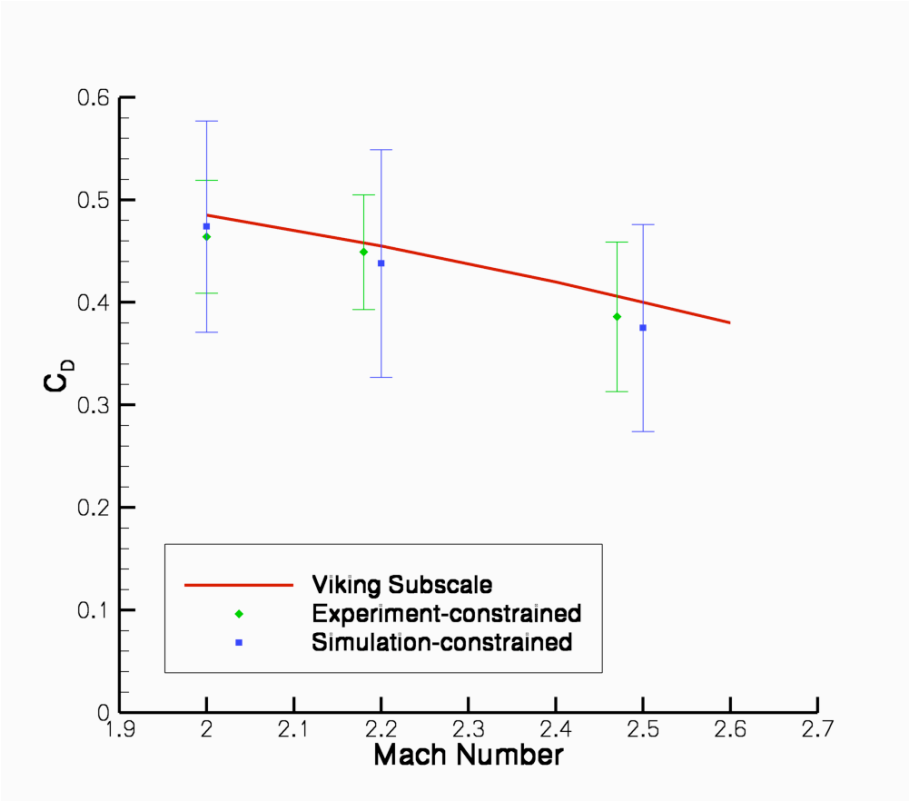

Figure 5.6: Coefficients of drag.

The experimentally determined coefficients of drag for the unconstrained tests are found to vary over the duration of experiment due to the rapid degradation and failure of the parachute that occurred in the wind tunnel. Because of this, only the early, high drag experimental values will be used for comparison with the simulation. Drag measurements in the unconstrained simulation were recorded over the first second following the parachute being released from a position of zero pull angle.

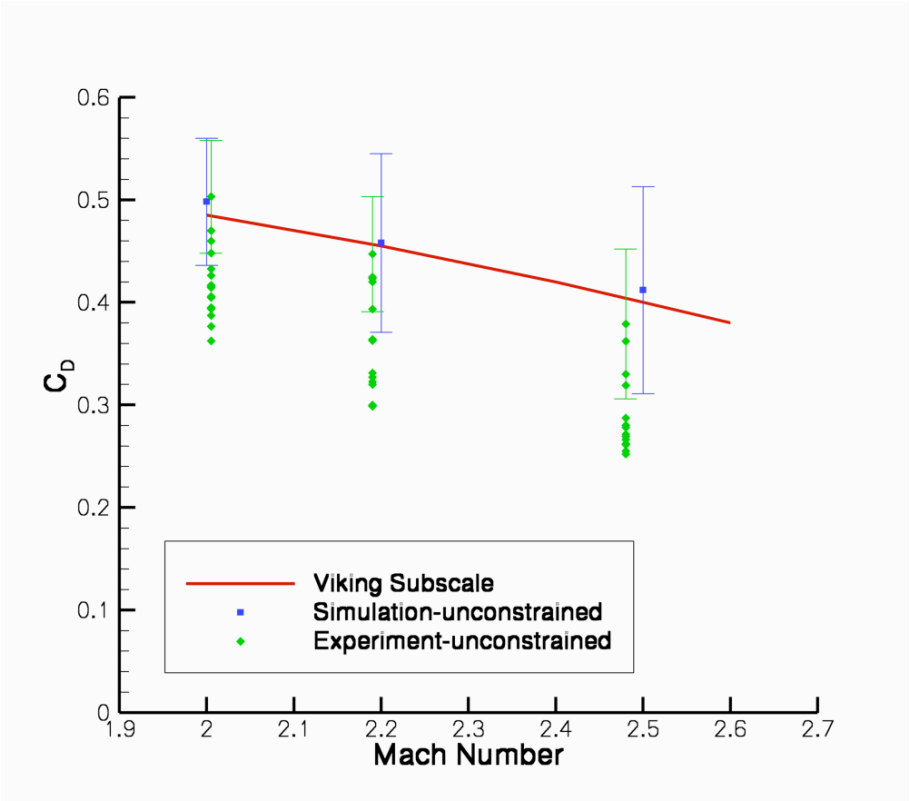

Figure 5.7: Coefficients of drag.

Figure 5.7 shows drag coefficients from the simulations match closely to the trend line from the Viking sub-scale measurements.

Drag coefficients for the unconstrained simulation are consistently higher than for the unconstrained simulation, as should be expected since the momentum of the flow increases as one moves out of the capsule's wake. However, this drag increase is only present in the experiment for the Mach 2.0 case. The higher Mach number coefficients also diverge from the Viking trend line. In these cases, the experiment saw sever reduction of the parachutes area due to deflation.

There are two different time scales at play which contribute to the variations observed in the operational drag of the parachute. The first is the short period pressurization cycles already described that occur at time scales on the order of 0.005 seconds. The second is due to the large-scale dynamic motion of the parachute, with time scales

on the order of 0.06 seconds.

5.1.5 Conclusions

By allowing the parachute to move off axis in response to forces from the flow field, it is found that the large scale motions of the canopy occur at a time scale that is an order of magnitude greater than the drag variations due to the growth and collapse of the parachutes bow shock. The simulations show that the RMS drag increases when the parachute is allowed to move partially out from behind the capsules wake. The parachute pull angles in the simulation reached amplitudes in the range of 2-3 degrees, which are consistent with what was observed in Viking flight tests [21]. It can then be assumed that the canopy should not be expected to vary away from an equilibrium position by pull angle magnitudes greater than single digit degrees when in flight. All coefficients of drag calculated in the simulations matched those obtained in the wind tunnel test, however the Mach 2.0 unconstrained case matched the experiments to a greater accuracy than the Mach 2.2 and 2.5 cases. Further simulations including full deformation of the canopy are required to determine for certain that it is an increased loss of area not being captured in these simulation that is responsible in the difference in accuracy between the Mach 2.0 and higher Mach number simulations.

The large time scale of the parachutes motion results in an increased computational cost to simulations. Simulations must be run with time steps that are small enough to resolve the turbulent wake behind the capsule, yet for a long enough duration to capture the parachutes dynamics

Chapter 6

Tension Cone

6.1 Introduction

The main disadvantage of a supersonic parachute comes from its interactions with a turbulent wake, as seen in the preceding chapter. This disadvantage is not present in an inflatable decelerator that does not lie in the wake. A tension cone is an example of such a decelerator.

6.2 Tension Cone IAD

The tension cone concept was developed in the 1960's [30]. Tension cones, like other inflatable decelerators including parachutes and ballutes, increases the amount of area exposed to the flow with minimal additional mass to the craft. This has the effect of lowering the ballistic coefficient of a re-entry vehicle and increasing its deceleration. The tension cone is comprised of a large inflated torus that is connected to the craft via a tension shell. The torus provides structural support [31] at the outer radius of the tension shell, the inflation of which deploys the decelerator. The key aspect of the tension shell is that its shape is such that it experiences only tensile forces when under load. Since there are no compressive forces to cause wrinkling or folding of the tension shell, the tension cone acts rigid when at equilibrium [32].

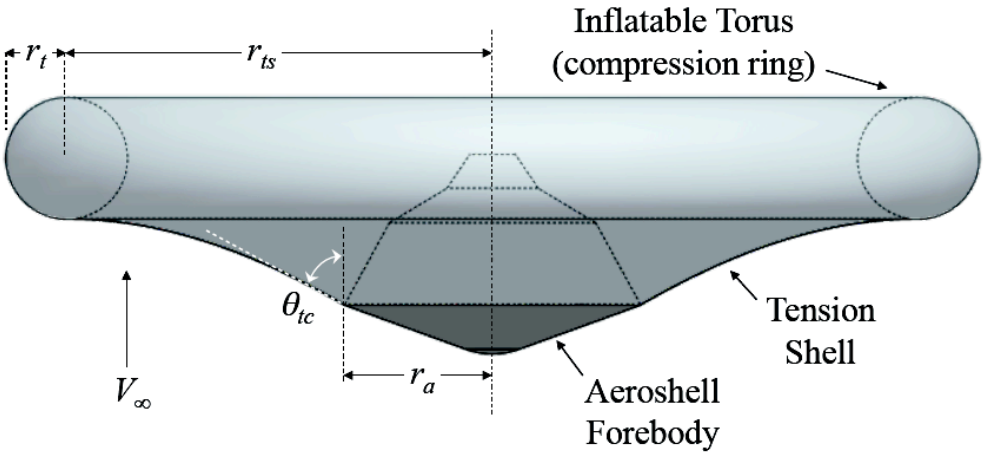

Figure 6.1: Diagram of a tension cone decelerator.[32]

6.2.1 Shape of Tension Shell

The shape of the tension shell is derived by enforcing a relation between the meridian stress and circumferential stress and applying a force balance.

The balance of in-plane stresses give a relation between the curve meridial angle, ϕ, the radius, r, and the material stresses σ_ϕ and σ_ϕ as

$$\frac{d(r\sigma_\phi)}{d\phi} - r_1\sigma_\theta \cos\phi = 0$$

A balance of the force normal to the surface requires consideration of the local radius of curvature, giving

$$\frac{\sigma_\phi}{r_1} + \frac{\sigma_\theta}{r_2} = \frac{P}{h}$$

for material of thickness h and the radii of curvature, r_1 and r_2, for ϕ and θ as

$$\frac{1}{r_1} = \cos\phi\frac{d\phi}{dr} = \frac{d\sin\phi}{dr},$$
$$\frac{1}{r_2} = \frac{\sin\phi}{r}.$$

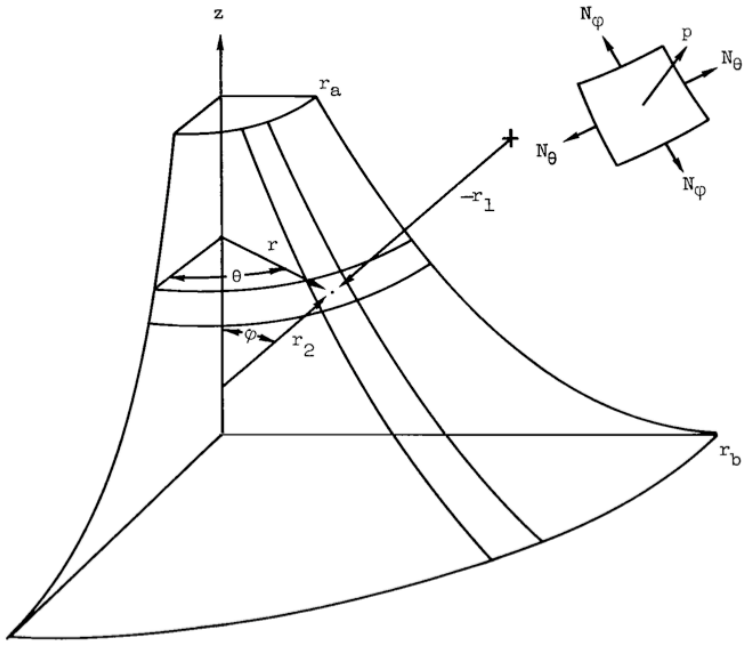

Figure 6.2: Coordinates used in derivation of tension shell shape. Meridial and circumferential stresses are labeled in labeled in this diagram as N_ϕ and N_θ instead of σ_ϕ and σ_θ. [30]

If one enforces a direct relation between the circumferential and meridian stresses,

$$\sigma_\theta = \alpha\sigma_\phi,$$

for some constant α, then the above equations reduce to

$$\frac{d\sin\phi}{dr} + \frac{\alpha}{r}\sin\phi - \frac{P}{h\sigma_o}\left(\frac{r}{r_0}\right)^{1-\alpha} = 0,$$

in which σ_o, is the stress at a reference radius r_o, and can be treated as a design parameter along with α. For a constant α, the stress σ_ϕ should be maximum at the point where the tension shell connects to craft, where the tension shell radius is minimum, and the full drag created by the tension cone is transfered to the rigid body of the decelerating vehicle. α need not always be a constant. In [30], a catenary tension shell shape is also derived for the purpose of being able to describe the shape of the tension

shell in terms of elementary functions instead of a collection of points, in which case α is not constant across the entire radius.

One possible concern is that the shape of the tension cone given by the above derivation is the shape under stress. If this stressed shape is used to construct the tension cone, the material stresses under load may be vary slightly from those expected, or assumed in the derivation. It should be possible, however to reconstruct the constructed shape from the stressed shape.

6.3 Flight Scale

The tension cone examined here is sized for an MSL like mission [33], for entry into the Martian atmosphere at 17 km, at Mach 5 attached to a 70 degree sphere cone capsule. The flight scale tension cone has an inner radius of 2.25 m, an outer radius of 6.23 m, and a torus radius of 0.89 m.

The thermal material properties used where extracted from [34], with structural properties chosen in the range characteristic of those used in [32]. Since the material used for this type of decelerator is often a rubberized fabric of some sort, and thus a non-homogeneous composite, structural properties must be tested for any actual craft. The exact material properties to be used on a tension cone of this scale and flight conditions are not available at the time of this writing, so approximate values are used instead. The properties used are listed in tables 6.1 and 6.2.

Material	Thickness (m)	k $(\frac{J}{Ks})$	$C_P\rho$ $(\frac{J}{m^3K})$
Viton	0.001	1.02	1.23
Nomex	0.001	18.6	19.9
Viton	0.0005	0.813	0.813

Table 6.1: Thermal properties used for a Viton coated Nomex fabric. k is the heat transfer coefficient, and $C_P\rho$ is the volumetric heat capacity.

Young's modulus $(\frac{J}{m^3 s})$	Density $(\frac{kg}{m^3})$	Poisson's ratio	Total thickness (m)
0.30×10^6	2700	0.4	0.0025

Table 6.2: Structural properties used for the tension shell fabric.

6.4 Simulation

The flow field is initialized over the undeformed geometry with an adiabatic wall boundary condition. Once the flow field has reached a steady solution, the fluid-structure coupling and the 1-D heat transfer model are activated. The flow field is initialized with an adiabatic wall boundary condition so that a hot boundary layer forms. Once the 1-D heat transfer model is turned on, the boundary layer temperature quickly drops to energy leaving the fluid and entering the structure.

The fluid that is include of the inflated torus has sufficient mass compared to the free stream to make the torus sluggish in its dynamics. Because it is desirable for the tension cone to rapidly reach a near equilibrium state in the simulation, a small acceleration is applied to the torus for the first 0.1 second of FSI simulation.

The simulation is then allowed to proceed for 2.0 seconds of simulation time to gauge the magnitude of the stress load on the tension shell. During an atmospheric entry, 2.0 seconds would be sufficient time to reduce the speed from Mach 5 to Mach 4.0.

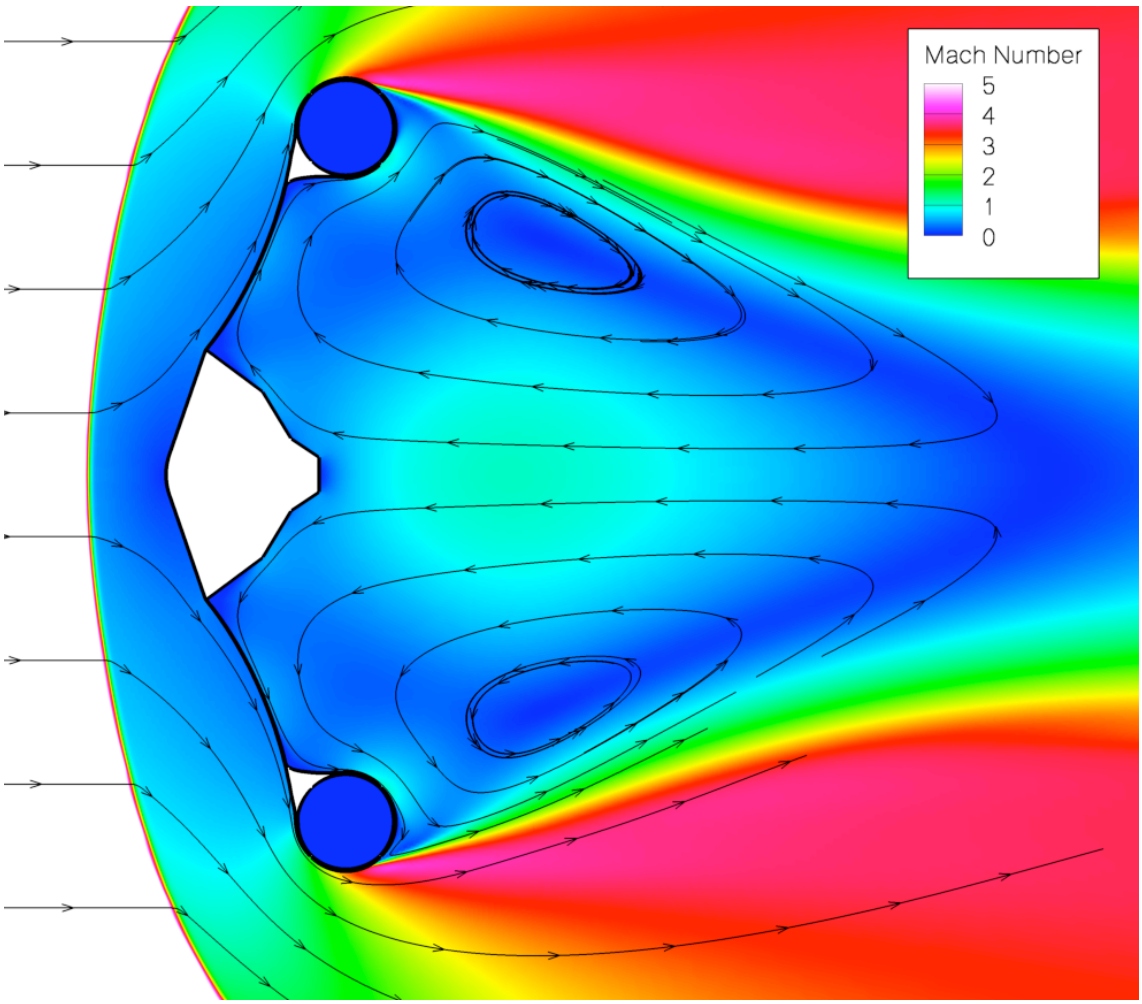

Figure 6.3: Flow field around a tension cone shape. Flood contours are Mach number.

Figure 6.3 shows the Mach number contours in the flow around the tension cone. Note the large subsonic recirculation region behind the vehicle. This recirculation region pulls fluid towards the aft side of the capsule.

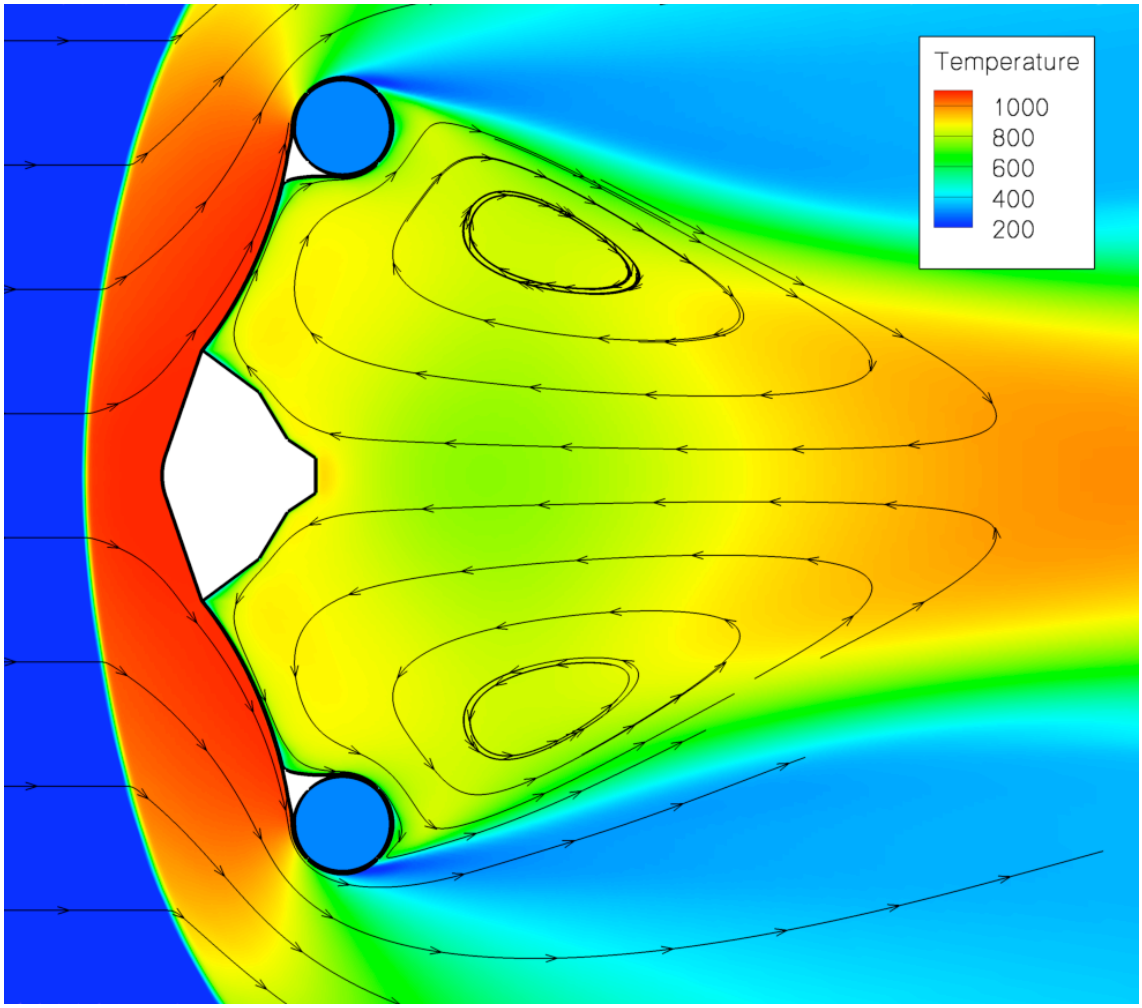

Figure 6.4: Flow field around a tension cone shape. Flood contours are temperature.

Figure 6.4 shows the temperature contours in the flow around the tension cone. Because the recirculation regions pulls fluid towards the capsule, aft heating may be a significant concern on the capsule. This may warrant future examination of the wake region using DES turbulence modeling with a low dissipation flux evaluation scheme to capture unsteady effects that are not resolved here [35].

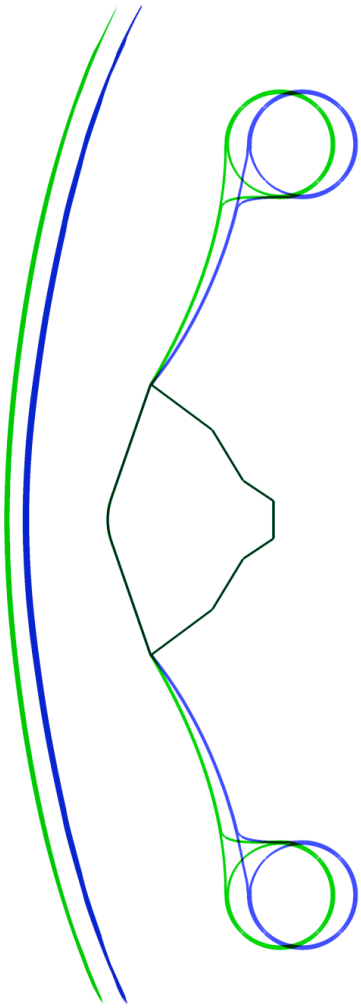

Figure 6.5: Change in geometry and shock position between constructed shape and deformed shape near equilibrium.

At equilibrium, the torus comes to a position of 0.38 m downstream of its initial position. Figure 6.5 shows the amount of deformation experienced, and the change in the shock position as a result of an effective narrowing of the total body.

6.4.1 Material Stress

It was found in the simulation that the material stress in the meridial direction was consistently larger than that in the circumferential direction This coincides with the shape for this tension shell being traced assuming an $\alpha < 1$. The meridial stress is also observed to be consistently monotonic, decreasing as the radius increases, as should be expected, since the material at the lower radius must support the drag created by the outer regions.

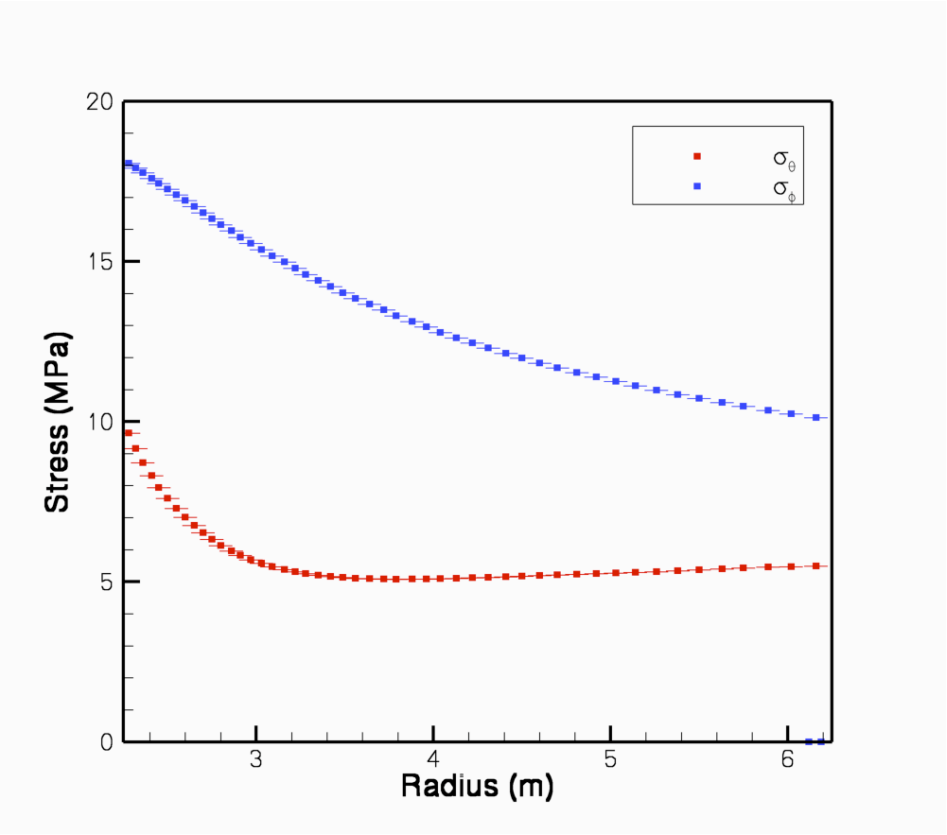

Figure 6.6: Meridian and circumferential stresses as a function of radius near equilibrium.

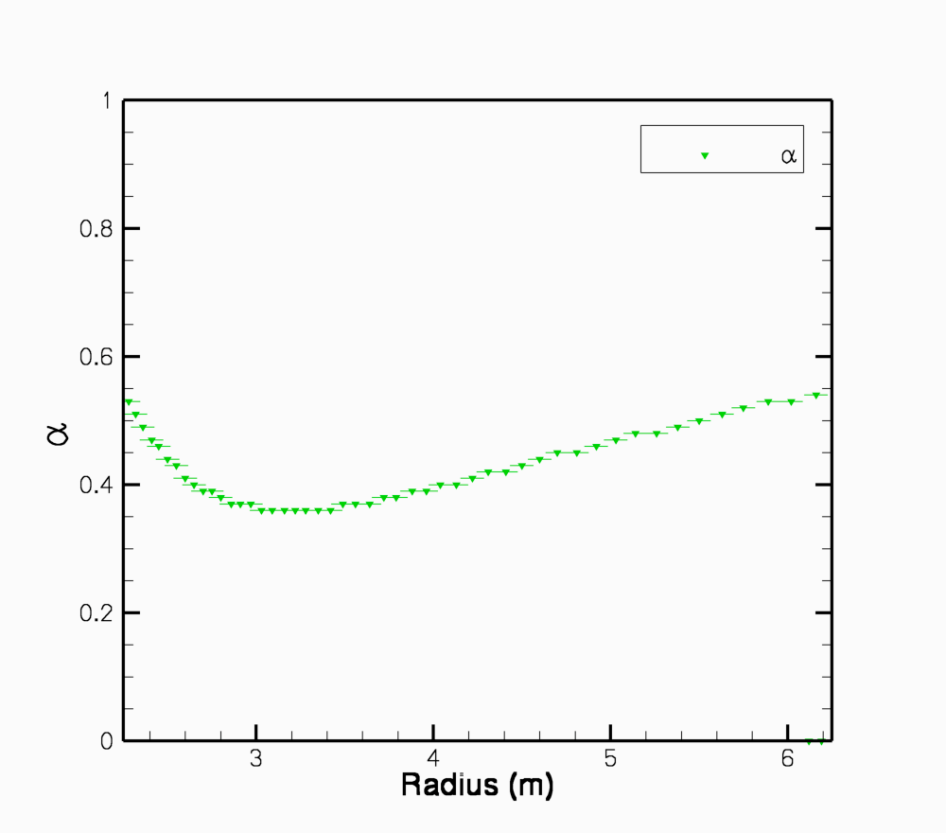

Figure 6.7: The ratio of circumferential to meridian stresses as a function of radius near equilibrium.

Looking at the ratio of circumferential to meridian stresses, we can gauge how far off of design the tension can behave when deployed under real aerodynamical loads. From the derivation, one should expect either a constant α, or an α that increases with the radius in the case of a catenary shape. However, from the simulation an α is found that decreases in the middle range of radii. A plot of α is shown in Figure 6.7. A history of the evolution of α is given in Appendix B.

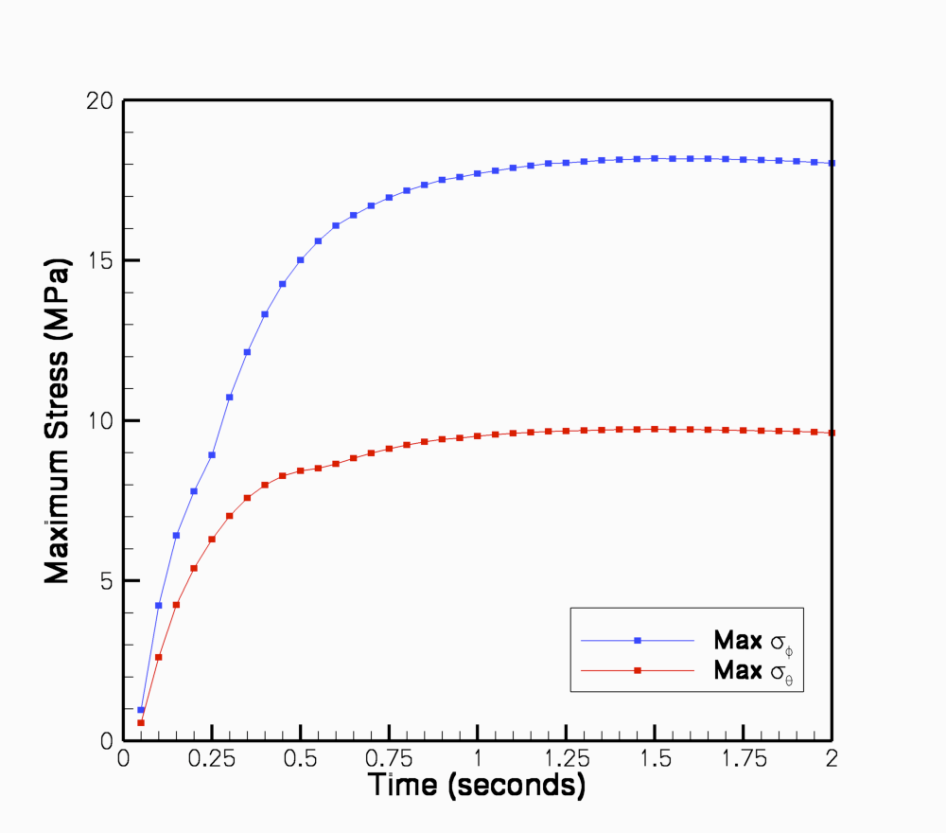

Figure 6.8: Maximum values for meridial and circumferential stresses at time intervals during loading.

Figures 6.8 and 6.9 show a time history of the maximum values for the stresses experienced by the tension shell and of the average value for α as the tension cone moves towards its equilibrium position. The error bars on Figure 6.9 are the standard deviation of α. The variations decrease as the torus reaches its equilibrium position, though they are still significant which shows off-design behavior.

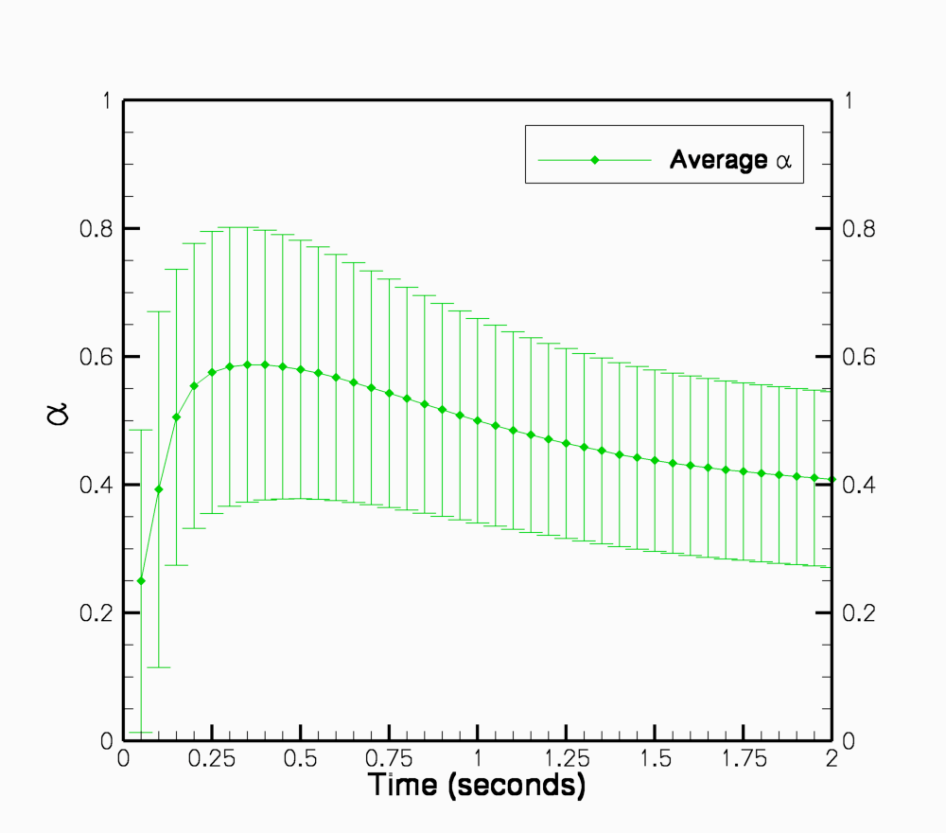

Figure 6.9: Average values for the ratio of circumferential and meridian stresses at time intervals during loading.

6.4.2 Material Heating

Under the examined flight condition, the amount of thermal energy available behind the shock is not sufficient to significantly increase the wall temperature of the tension cone. From the simulation, the density and temperature near wall at the connection point between the tension shell and capsule are $\sim 0.015\frac{kg}{m^3}$, and $\sim 1100K$. The volumetric specific heat for carbon dioxide at that density is only $0.0126\frac{kJ}{m^3K}$, where as for the Viton coating on the tension shell, the volumetric specific heat is $939\frac{kJ}{m^3K}$.

If the wall temperature is initially $300K$, then even if a 1 m thickness of stagnated gas were to come to thermal equilibrium with the wall, there would only be enough

energy to raise the temperature of the first 1mm of Viton coating 12.5 degrees.

It is therefore concluded that a tension cone of these dimensions, and materials should not be expected to experience any material failure due to either stresses or aerothermal heating during a Mach 5 deployment in a Martian environment.

Chapter 7

Conclusion and Discussion

7.1 Summary

It has been shown that a non-monolithically coupled fluid-structure solver is capable of providing time dependent results for simulations of deployable decelerator systems. It has also been demonstrated that such a tool can be useful in understanding the underlying phenomena that drive these systems. Even the most basic systems can lead to a full range of complex behavior when coupled together, a fact that is becoming more and more apparent with the development of computational methods. The availability of a tool for studying of fluid structure interactions should prove invaluable for the study of new and fascinating flow behavior.

7.2 Robustness

7.2.1 Grid Deformation

The range of deformation allowable is ultimately dependent on the robustness and generality of the method used to update the volume discretization of the flow solver. The methods outlined here are independent of the connectivity of the grid, yet are still limited to moderate deformations. It should be noted that any method of grid deformation will be inherently limited to how far grid density and alignment can be altered without significantly affecting the quality of the solution. Any situation that requires two surfaces coming into contact or splitting apart will require some re-gridding.

Re-gridding is computationally more expensive that grid deformation, and raises issues of fidelity when the solution is interpolated from the old grid to the new. The choice of method used to alter the CFD grid depends on the class of problems that one wishes to examine, and no single method can be guaranteed to work for all conceivable situations.

7.2.2 Structural Solver

The component that required the most attention to ensure time accurate behavior was the structural solver. The necessary addition of a local low-pass filter to the velocity is a testament to these difficulties. A finite-element solver was initially used for the coupling but was not able to meet the validation requirements for large deformations.

In this respect, the non-monolithic nature of the fluid-structure coupling definitely showed its advantages. Different structural solvers, or even grid deformation strategies, can be swapped out when needed. The same framework can be used for a structural membrane solver, a fully 3-D elastic solid solver, control surfaces, or thermal response model.

7.3 Open Questions

The time intervals required for the simulation of these decelerators is in the range of seconds. At these lengths of time, a decelerating craft will experience changes in its velocity as well as atmospheric composition as it descends. These effects should be included in a comprehensive look at the vehicle's performance. Also of interest would be the effects of being in a decelerating frame on both the fluid and the structure. An additional term appears in the momentum update when frame accelerations are included into the simulation. This may have an effect on flow phenomena, as well as on the dynamics of the membrane structure.

The amount of heating experienced in the tension cone simulation was not enough to significantly affect the wall temperatures. It would be of interest to use the 1-D heat transfer model on a tension cone that undergoes more substantial heating.

References

[1] Fries, T. P., "A Stabilized and Coupled Meshfree/Meshbased Methode for Fluid-Structure-Iteraction Problems," Dissertation, in *Braunschweiger Schriften zur Mechanik*(H, Antes, Ed.), Nr. 59-2005, Braunschweig, 2005.

[2] Yang, Z., and Mavriplis, D. J., Higher-Order Time-Integration Schemes for Aeroelastic Applications on Unstructured Meshes, *AIAA Journal*, Vol. 45, No. 1, 2007, pp. 138-150.

[3] Lingard, J. S., and Darley, G., Simulation of Parachute Fluid Structure Interaction in Supersonic Flow, *AIAA Paper 2005-1607*, 2005.

[4] Karagiozis, K., R. Kamakoti, and C. Pantano, "A Low Numerical Dissipation Immersed Interface Method for the Compressible Navier-Stokes Equations," *Journal of Computational Physics*, 229, 701-727, 2010.

[5] Degand, C., and Farhat, C., A Three-Dimensional Torsional Spring Analogy Method for Unstructured Dynamic Meshes, *Computers and Structures*, Vol. 80, pp. 305-316, 2002.

[6] Tanner, C., Cruz, J., Braun, R. J., Structural Verification and Modeling of a Tension Cone Inflatable Aerodynamic Decelerator, *AIAA Paper 2010-2830*, April 2010.

[7] Nompelis, I., Drayna, T., and Candler, G. V., A Parallel Unstructured Implicit Solver for Hypersonic Reacting Flow Simulation, *AIAA Paper 2005-4867*, June 2005.

[8] Druguet, M.-C., Candler, G. V., and Nompelis, I., Effects of Numerics on Navier-Stokes Computations of Hypersonic Double-Cone Flows, *AIAA Journal*, Vol. 43, No. 3, March 2005, pp. 616-623.

[9] Wright, M. J., Bose, D., and Candler, G. V., A Data-Parallel Line Relaxation Method for the Navier-Stokes Equations, *AIAA Journal*, Vol. 36, No. 9, 1998, pp. 1603-1609.

[10] Hennighausen, K., "Fluid Mechanics of Microscale Flight," Dissertation, *University of Minnesota*, August 2001.

[11] Brackbill, J. U., An Adaptive Grid with Directional Control, *Journal of Computational Physics*, 108, 38-50, 1993

[12] Cao, W., Huang, W., and Russol, R. D., An r-Adaptive Finite Element Methode Based upon Moving Mesh PDEs, *Journal of Computational Physics*, 149, 221-244, 1994.

[13] Beckett, G., Mackenzie, J. A., and Robertson, M. L., A Moving Mesh Finite Element Mothod for the Solution of Two-Dimensional Stefan Problems, *Journal of Computational Physics*, 168, 500-518, 2001.

[14] Dwight, R.P., "Robust mesh deformation using the linear elasticity equations," *Proceedings of the Fourth International Conference on Computational Fluid Dynamics (ICCFD 4)*, Ghent, Belgium, 2006.

[15] Kholodar, D. B., Morton, S. A., and Cummings, R. M., "Deformation of Unstructured Viscous Grids," *43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005.

[16] de Boer, A. Van der School, M. S.,and Bijl, H., "Mesh deformation based on radial basis function interpolation," *Computers and Structures*, 85, 784795, June 2007.

[17] Tsangaris, S., and Pappou, Th., Analytical Solutions for the Unsteady Compressible Flow Equations Serving as Test Cases for the Verification of Numerical Scemes,, *Republished in RTO Technical Report 26*, October 2000, pp. 45-60.

[18] Player, Charles, PAIDAE Thermal Protection System Testing Final Report FY2008, *PAI-DAE-3.3-012*, 2008, NASA Langley Research Center.

[19] Del Corso, J., Bruce, W., Liles, K., and Hughes, S., Thermal Analysis and Testing of Candidate Materials for PAIDAE Inflatable Aeroshell. *AIAA Paper 2009-2925*, May 2009.

[20] Hughes, S., Ware, J., Del Corso, J., and Lugo, R., Deployable Aeroshell Flexible Thermal Protection System Testing. *AIAA Paper 2009-2926*, May 2009.

[21] D. Dickson, F. Hicks, F. Michel, J. Schlemmer, R. D. Moog, Balloon Launched Decelerator Test Program Post-Test Test Report, *TR-3720289*, September 1972.

[22] Barnhardt, M., Drayna, T., Nompelis, I., Candler, G.V., and Garrard,W., Detached Eddy Simulations of the MSL Parachute at Supersonic Conditions, *AIAA Paper 2007-2529*, 2007.

[23] Sengupta, A., et al., Overview of the Mars Science Laboratory Parachute Decelerator System, *19th AIAA Aerodynamic Decelerator Systems Technology Conference*, May 2007.

[24] Sengupta, A., et al., Supersonic Delta Qualification by Analysis Program for the Mars Science Laboratory Parachute Decelerator System, *AIAA Paper 2007-2542*, 2007.

[25] Sengupta, A., et al., Results from the Mars Science Laboratory Parachute Decelerator System Supersonic Qualification Program, *2008 IEEE Aerospace Conference*, March 2008.

[26] Sengupta, M. Wernet, J. Roeder, R. Kelsch, A. Witkowski, and T. Jones, "Supersonic Testing of 0.8 m Disk Gap Band Parachutes in the Wake of a 70 deg Sphere Cone Entry Vehicle," *Presented at the 20th AIAA ADS Conference*, May 2009, Seattle, WA.

[27] Gidzak, V., Barnhardt, M., Drayna, T., Nompelis, I., Candler, G.V., and Garrard,W., Simulation of Fluid-Structure Interactions of the Mars Science Laboratory Parachute, *AIAA Paper 2008-6910*, 2008.

[28] Gidzak, V., Barnhardt, M., Drayna, T., Nompelis, I., Candler, G.V., and Garrard,W., Comparison of Fluid-Structure Interactions of the Mars Science Laboratory Parachute with Wind Tunnel Tests, *AIAA Paper 2009-2971*, 2009.

[29] I. Jaremenko, S. Steinberg, and R. Faye-Petersen, Scale Model Test Results of the Viking Parachute System at Mach Numbers from 0.1 Through 2.6, *TR-3720181*, November 1971.

[30] Anderson, M. S., Robinson, J. C., Bush, H. G., Fralich, R. W., A Tension Shell Structure for Application To Entry Vehicles *NASA-TN-D-2675*, March 1965.

[31] Kyser, A. C., Deployment Mechanics for an Inflatable Tension-Cone Decelerator *NASA-CR-929*, November 1967.

[32] Clark, I. G., "Aerodynamic Design, Analysis, and Validation of a Supersonic Inflatable Decelerator," Dissertation, *Georgia Institute of Technology*, August 2009.

[33] Clark, I., Hutchings, A., Tanner, C., and Braun, R., Supersonic Inflatable Aerodynamic Decelerators For Use On Future Robotic Missions to Mars., *IEEEAC Paper 1419*, 2008.

[34] Paurote, G. L., Burgess, J. L., Thermal and Stress Analysis of an Attached Inflatable Decelerator (AID) Deployed in the Mars and Earth Atmospheres *NASA-CR-111920*, 1971.

[35] Subbaredy, P. and Candler, G. V., A Fully Discrete, Kinetic Energy Consistant Finite-Volume sceme for Compressible Flows, *Journal of Computational Physics*, 228, 1347-1364, 2009.

Appendix A

Shape of the Tension Shell

The x and radial coordinates that define the tension shell.

x	r	x	r	x	r	x	r	x	r
0.000	2.250	0.442	3.083	0.797	3.917	1.051	4.752	1.197	5.587
0.027	2.297	0.465	3.132	0.814	3.966	1.063	4.801	1.202	5.636
0.055	2.346	0.488	3.181	0.832	4.015	1.074	4.850	1.207	5.685
0.083	2.395	0.511	3.230	0.849	4.065	1.085	4.899	1.211	5.734
0.110	2.444	0.534	3.279	0.866	4.114	1.096	4.948	1.215	5.783
0.137	2.493	0.556	3.328	0.882	4.163	1.106	4.998	1.218	5.832
0.164	2.542	0.578	3.377	0.898	4.212	1.115	5.047	1.221	5.881
0.191	2.591	0.599	3.426	0.914	4.261	1.125	5.096	1.224	5.930
0.217	2.641	0.621	3.475	0.929	4.310	1.134	5.145	1.226	5.980
0.243	2.690	0.641	3.524	0.944	4.359	1.142	5.194	1.228	6.029
0.269	2.739	0.662	3.574	0.959	4.408	1.151	5.243	1.229	6.078
0.295	2.788	0.682	3.623	0.973	4.457	1.158	5.292	1.231	6.127
0.320	2.837	0.702	3.672	0.987	4.506	1.166	5.341	1.231	6.176
0.345	2.886	0.722	3.721	1.001	4.556	1.173	5.390	1.231	6.225
0.370	2.935	0.741	3.770	1.014	4.605	1.180	5.439		
0.394	2.984	0.760	3.819	1.027	4.654	1.186	5.489		
0.418	3.033	0.778	3.868	1.039	4.703	1.192	5.538		

Appendix B

Progression of Stress Distribution

It is important to see how the stress distribution changes as the tension cone is loaded. Circumferential and meridian stress distribution as a function of radius as the simulation progresses. Times are measured as the amount of time elapsed after the FSI component of the simulation is activated. The values of α for increasing radius are also plotted for these same time points in the simulation.

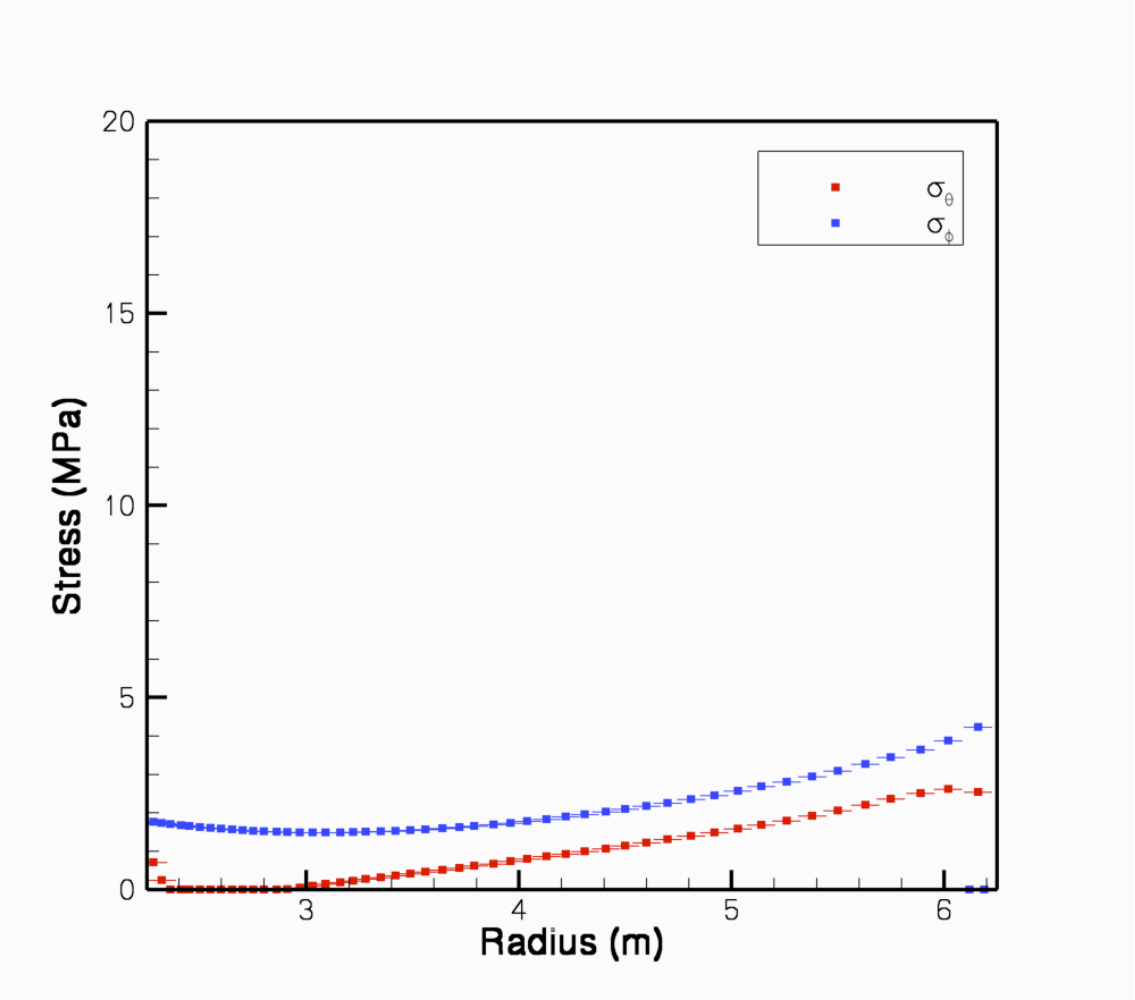

Figure B.1: Meridian and circumferential stresses at time = 0.1 second.

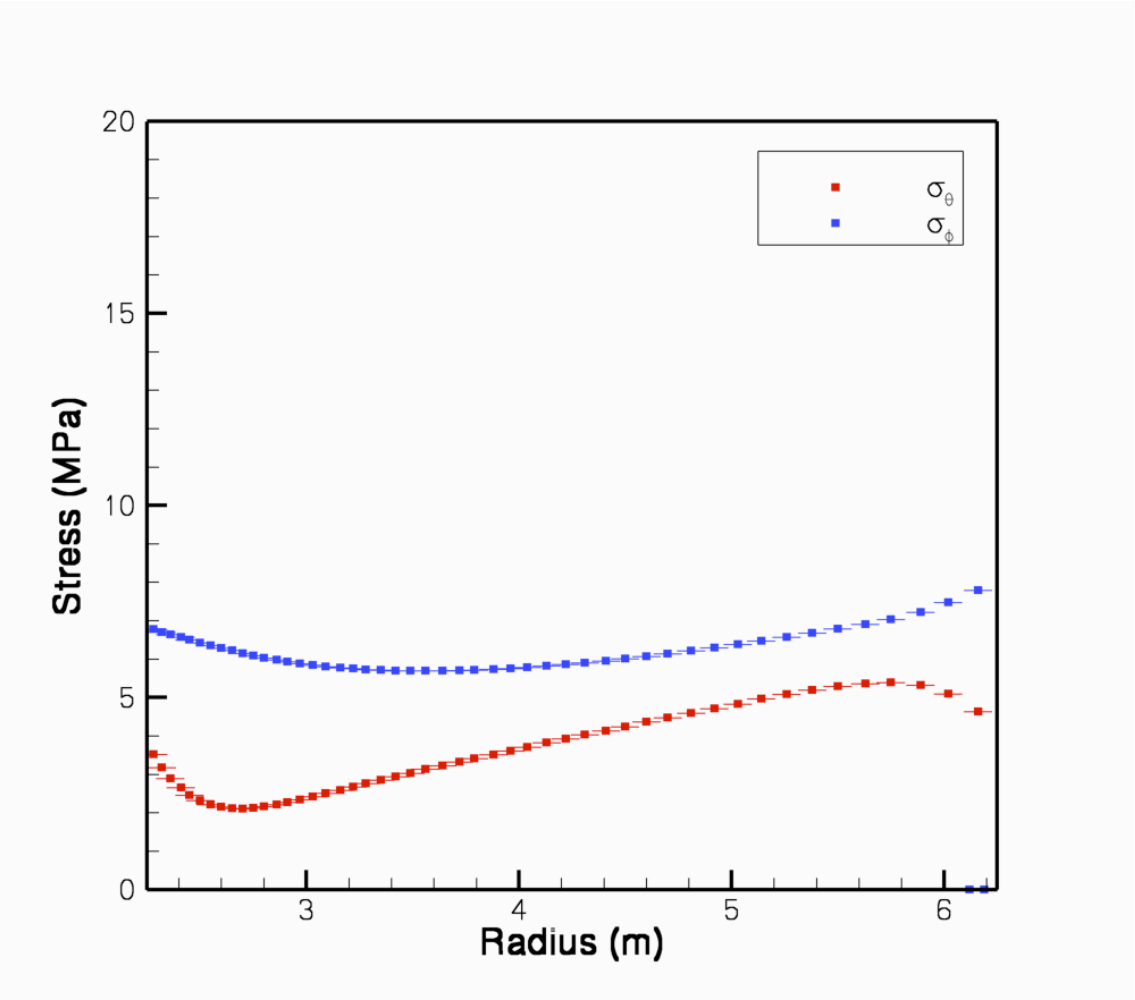

Figure B.2: Meridian and circumferential stresses at time = 0.2 second.

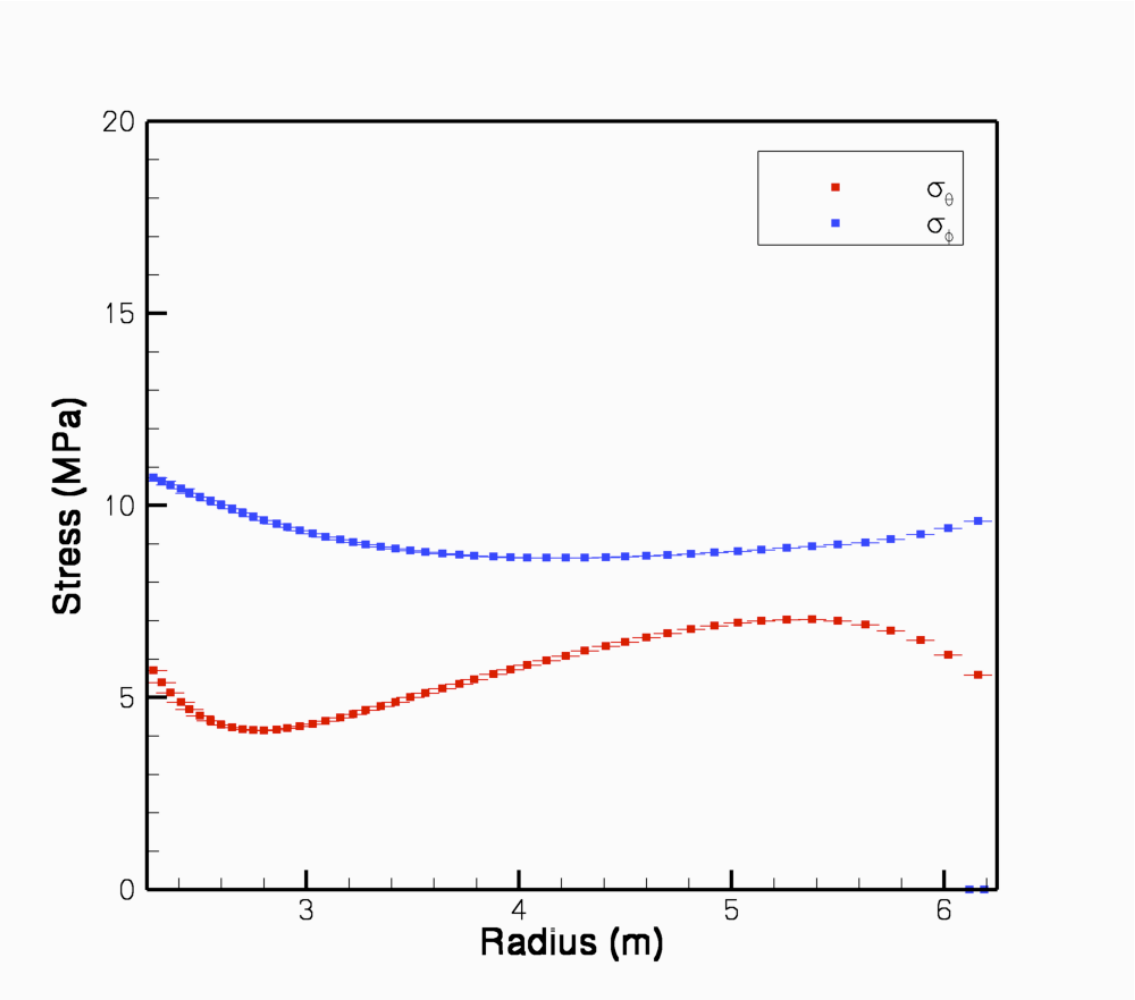

Figure B.3: Meridian and circumferential stresses at time = 0.3 second.

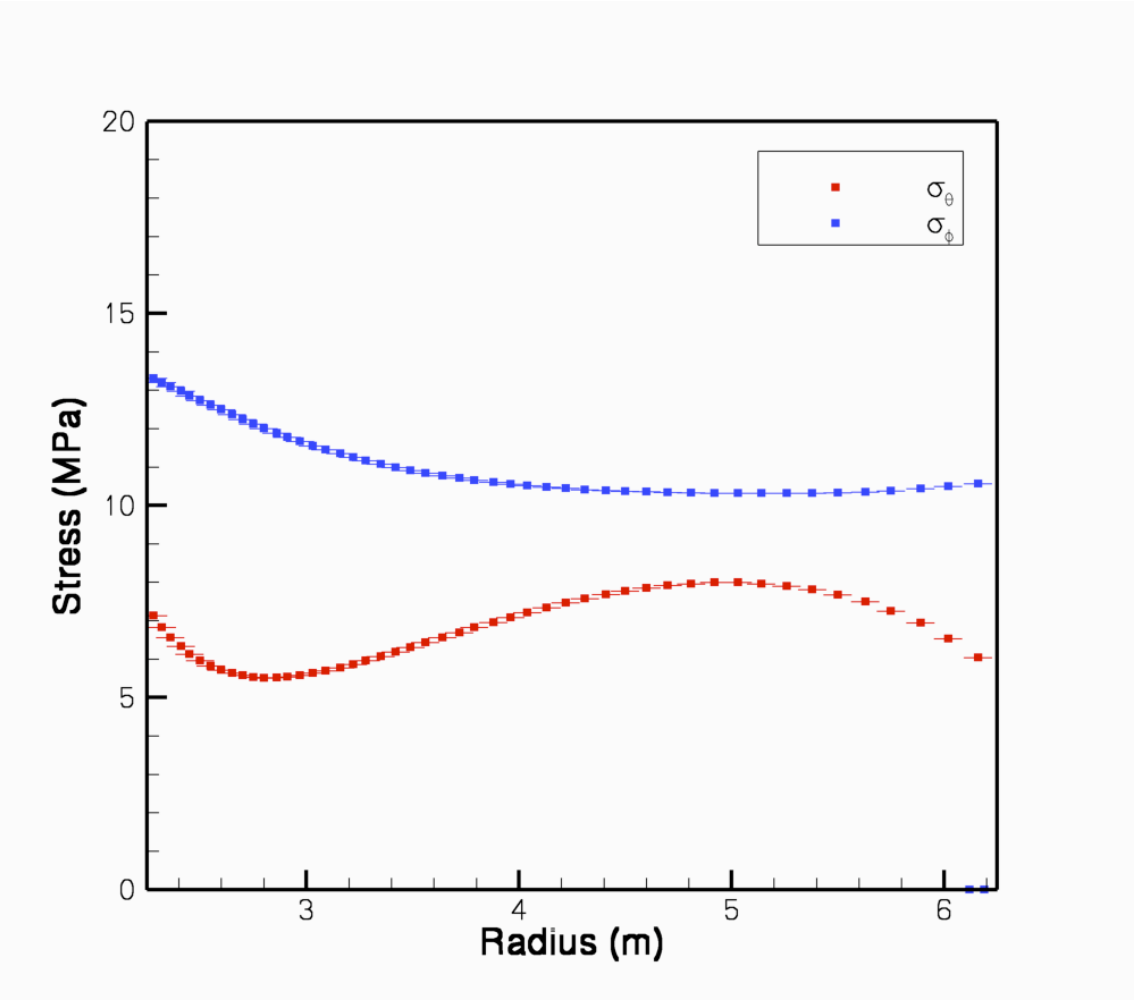

Figure B.4: Meridian and circumferential stresses at time = 0.4 second.

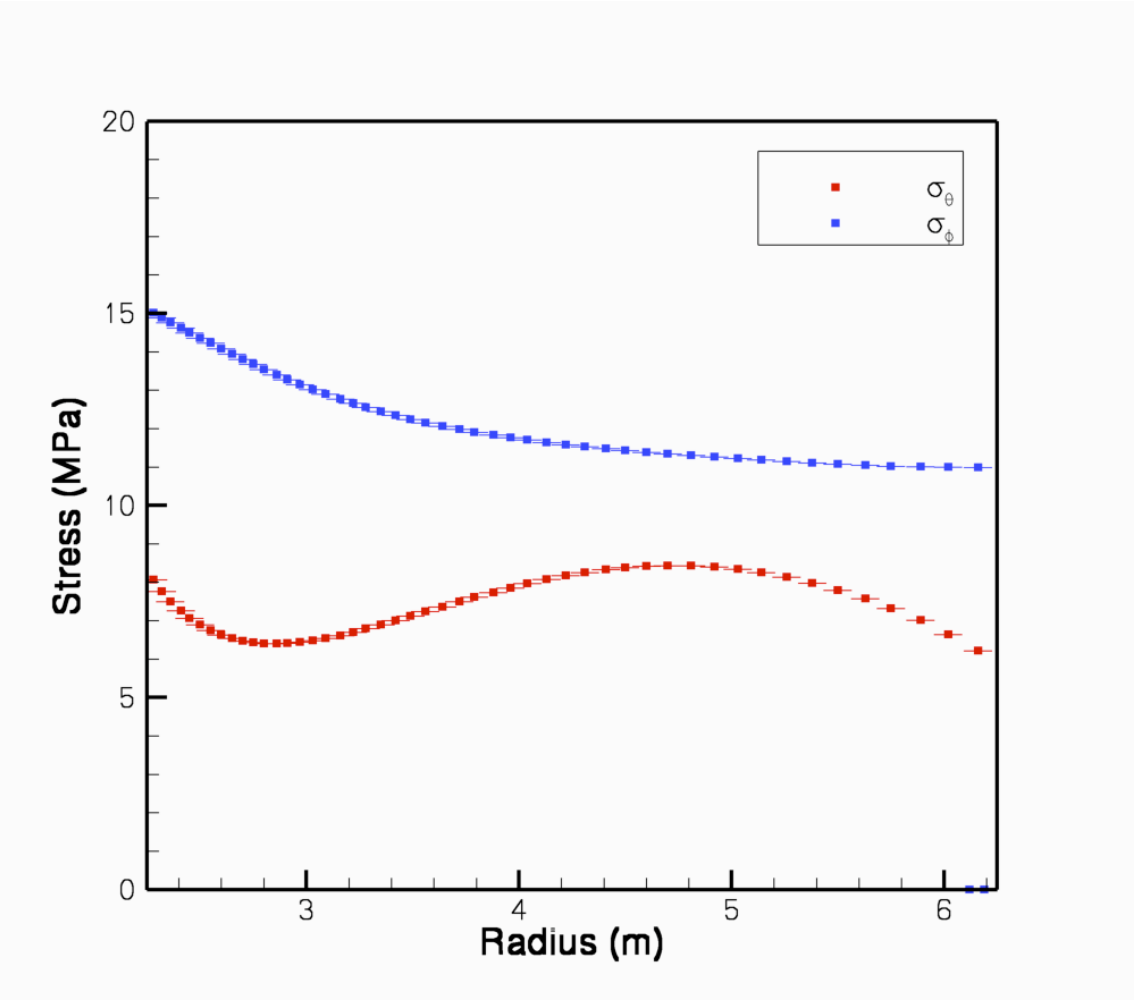

Figure B.5: Meridian and circumferential stresses at time = 0.5 second.

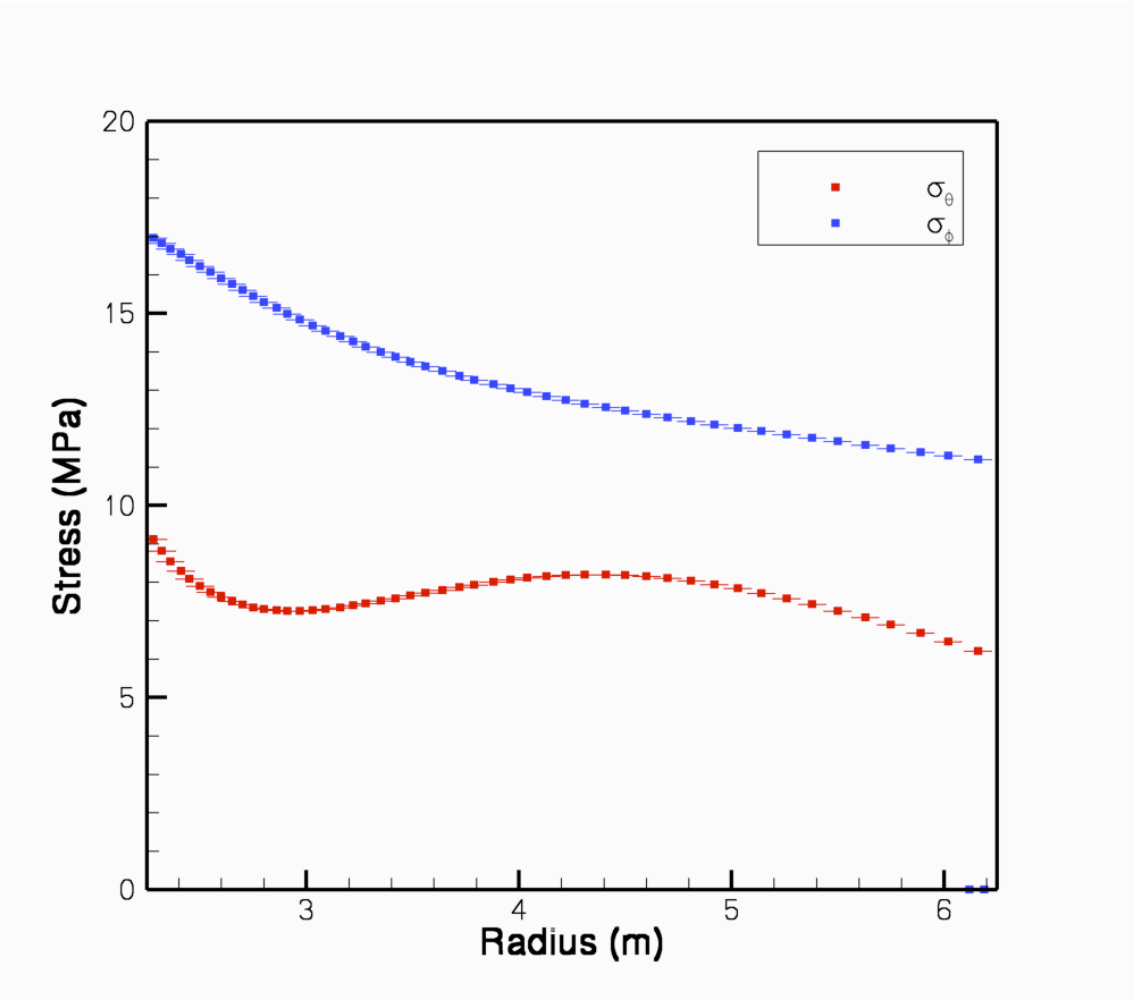

Figure B.6: Meridian and circumferential stresses at time = 0.75 second.

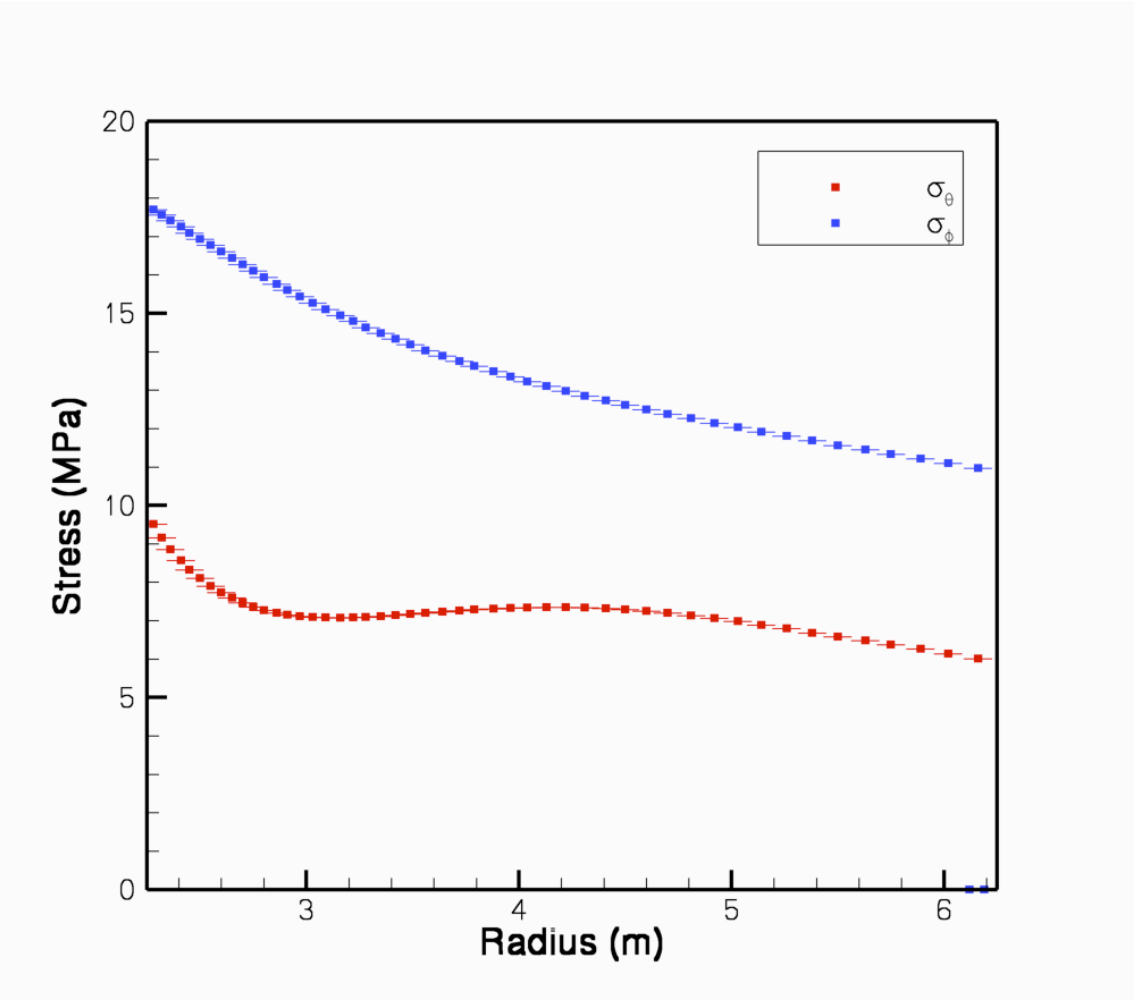

Figure B.7: Meridian and circumferential stresses at time = 1.0 second.

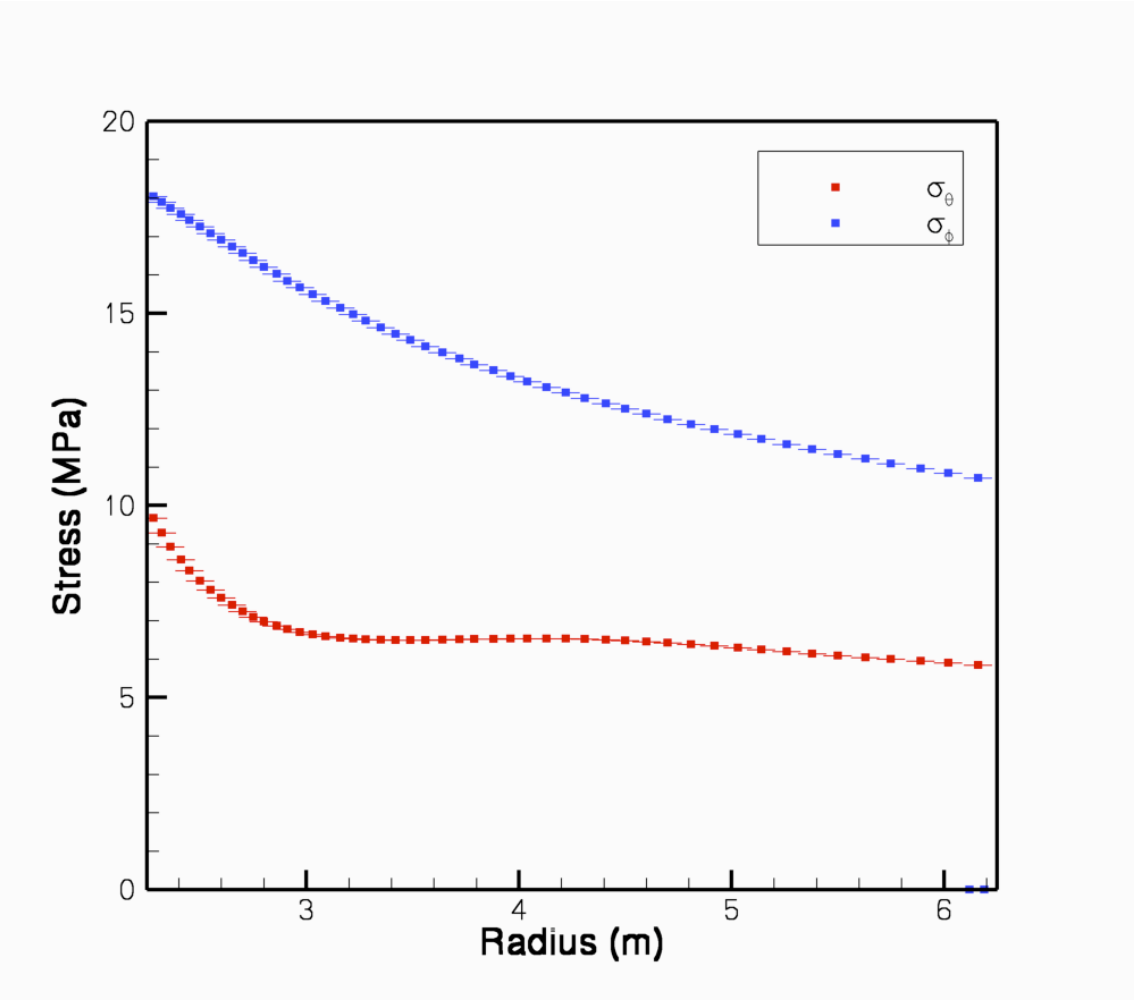

Figure B.8: Meridian and circumferential stresses at time = 1.25 second.

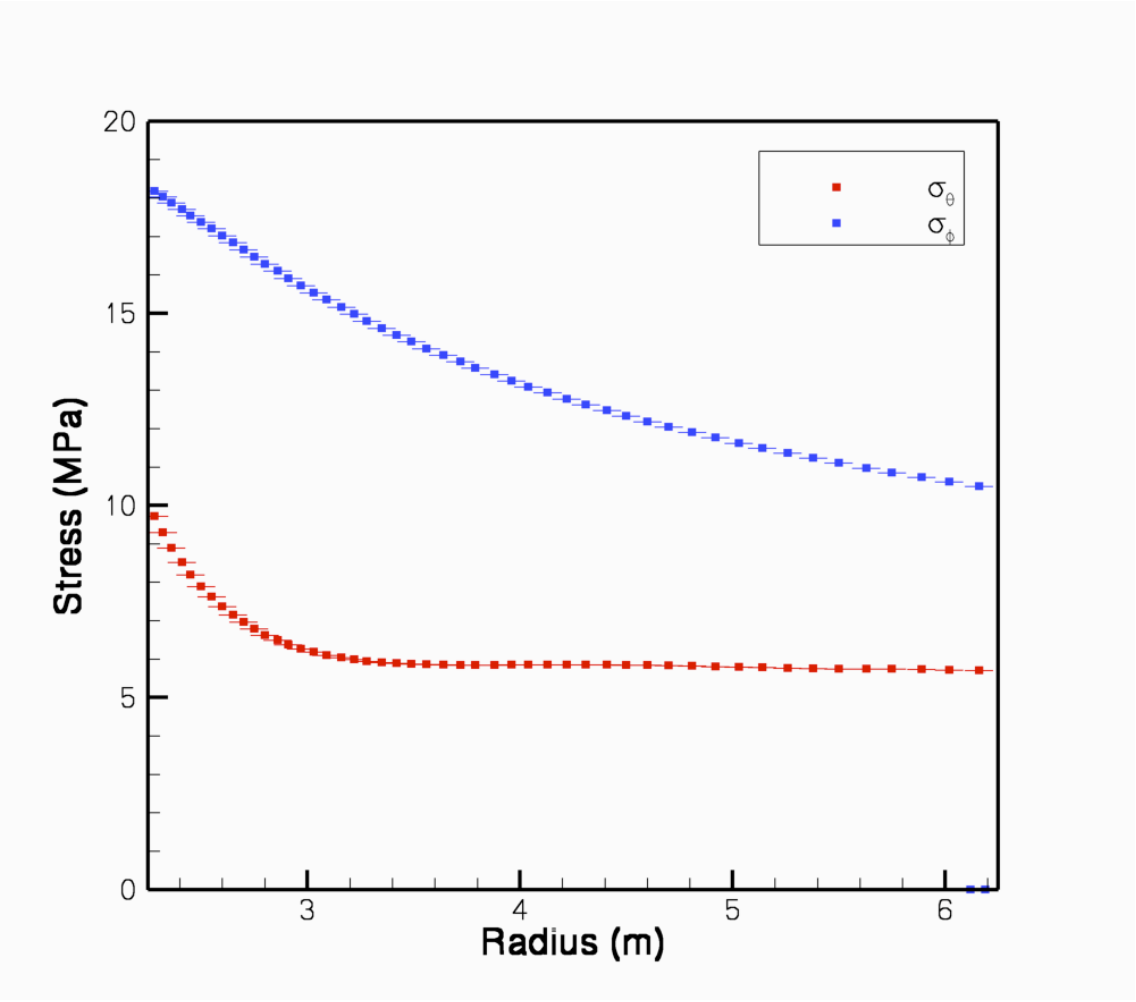

Figure B.9: Meridian and circumferential stresses at time = 1.5 second.

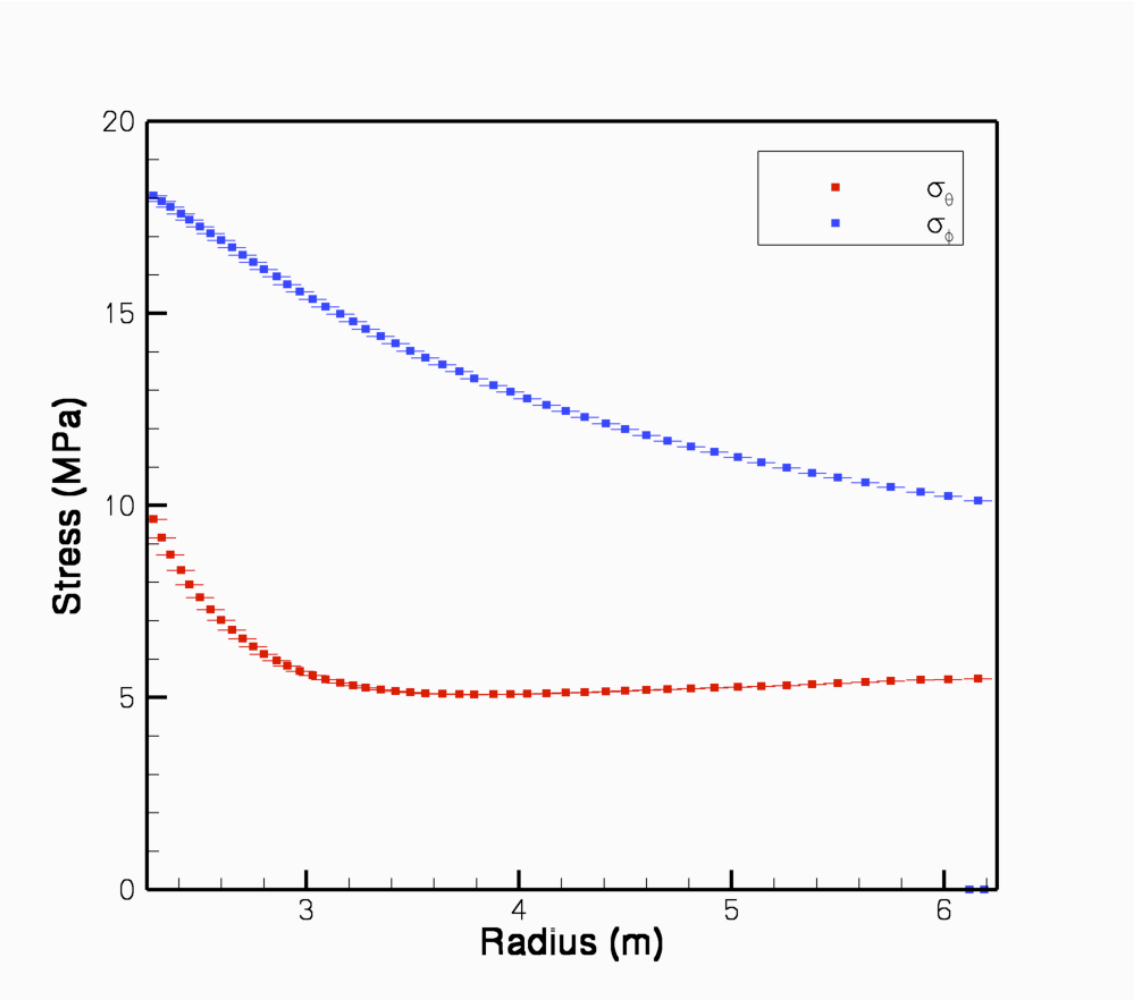

Figure B.10: Meridian and circumferential stresses at time = 2.0 second.

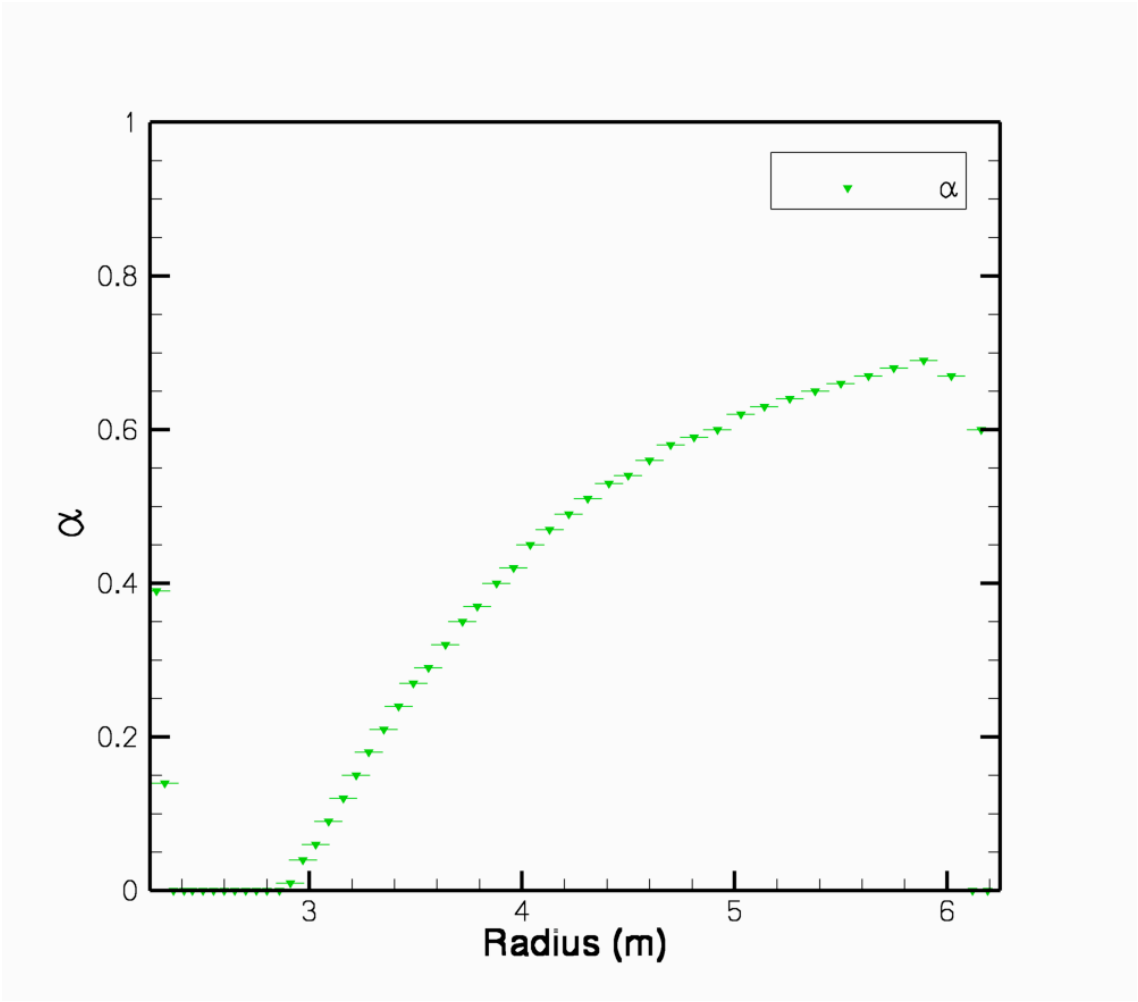

Figure B.11: The ration bewteen the meridian and circumferential stresses at time = 0.1 second.

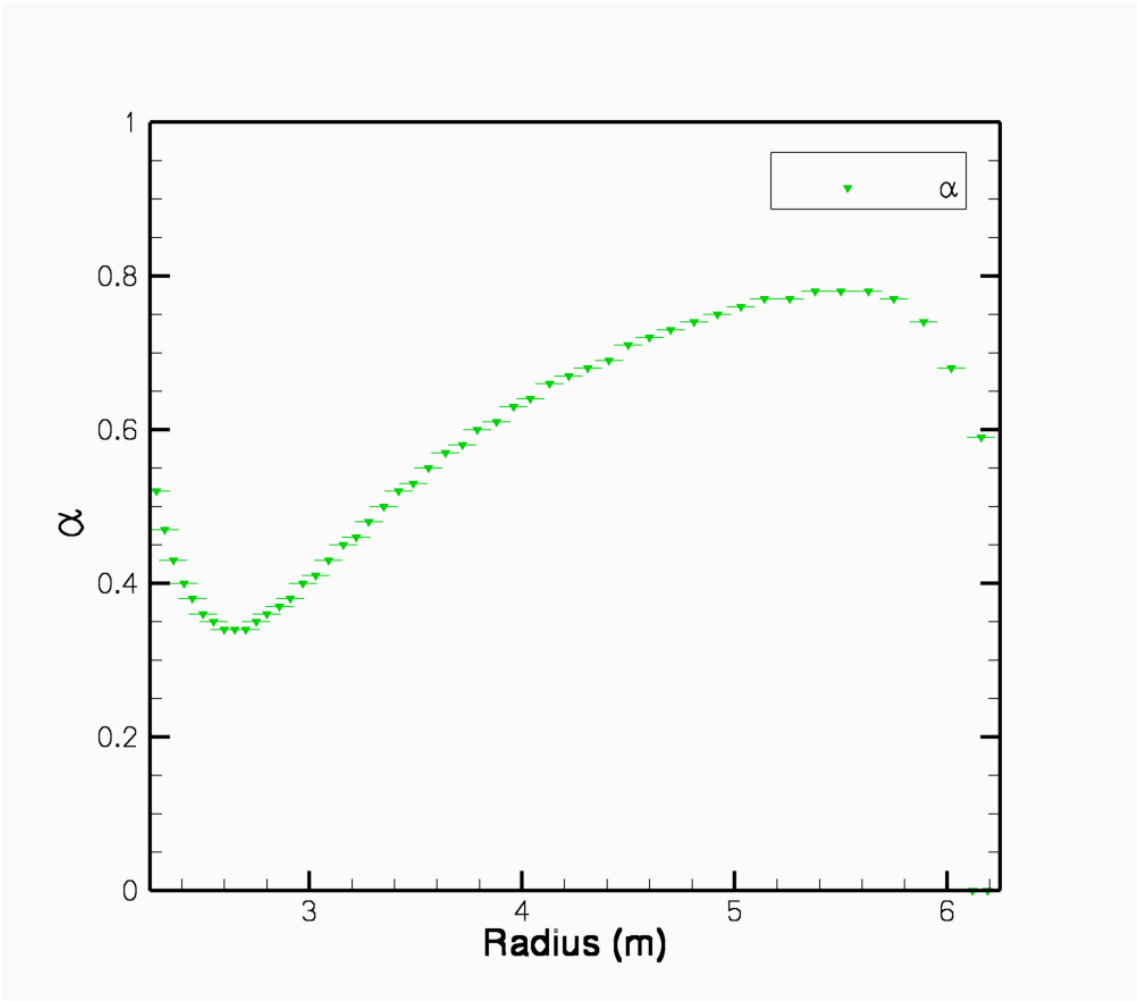

Figure B.12: Meridian and circumferential stresses at time = 0.2 second.

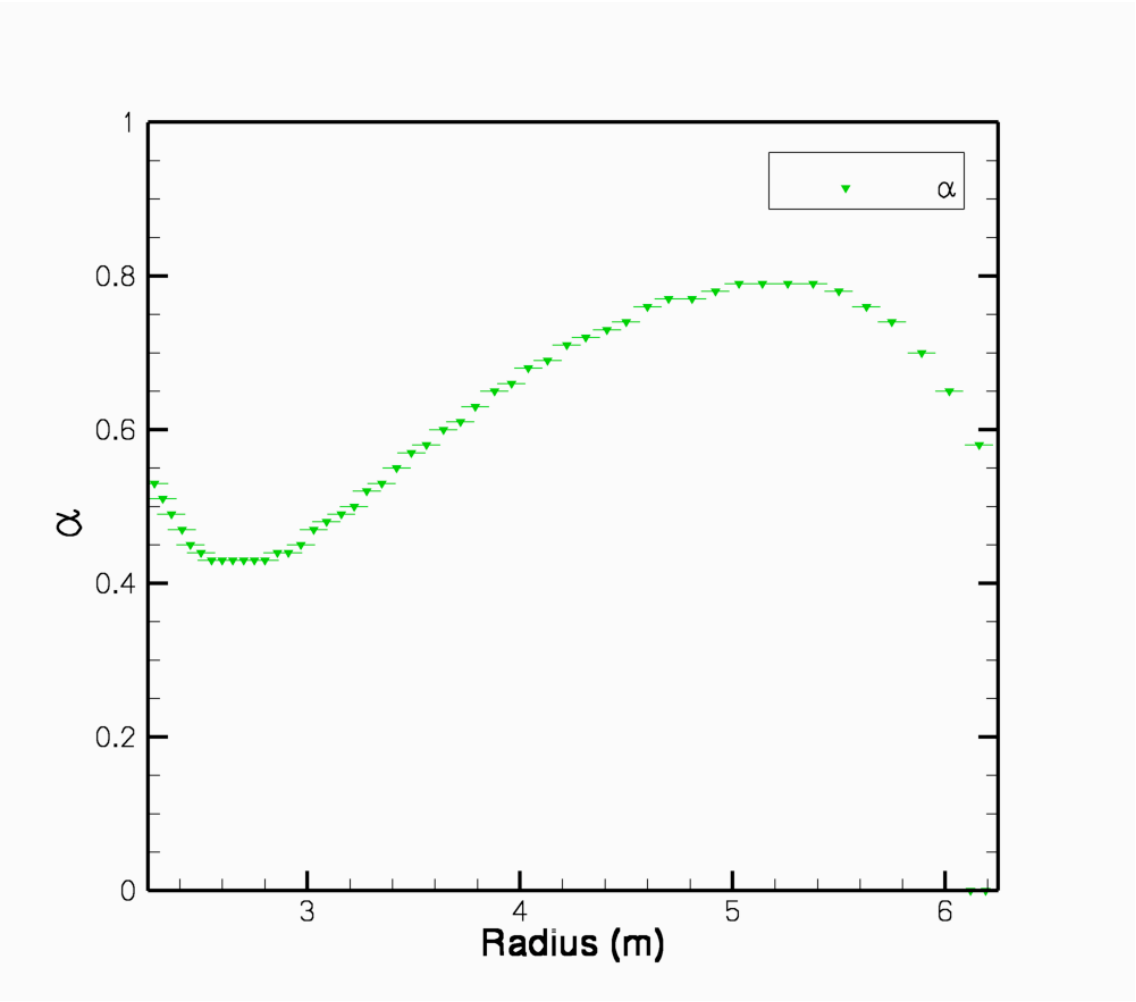

Figure B.13: The ration bewteen the meridian and circumferential stresses at time = 0.3 second.

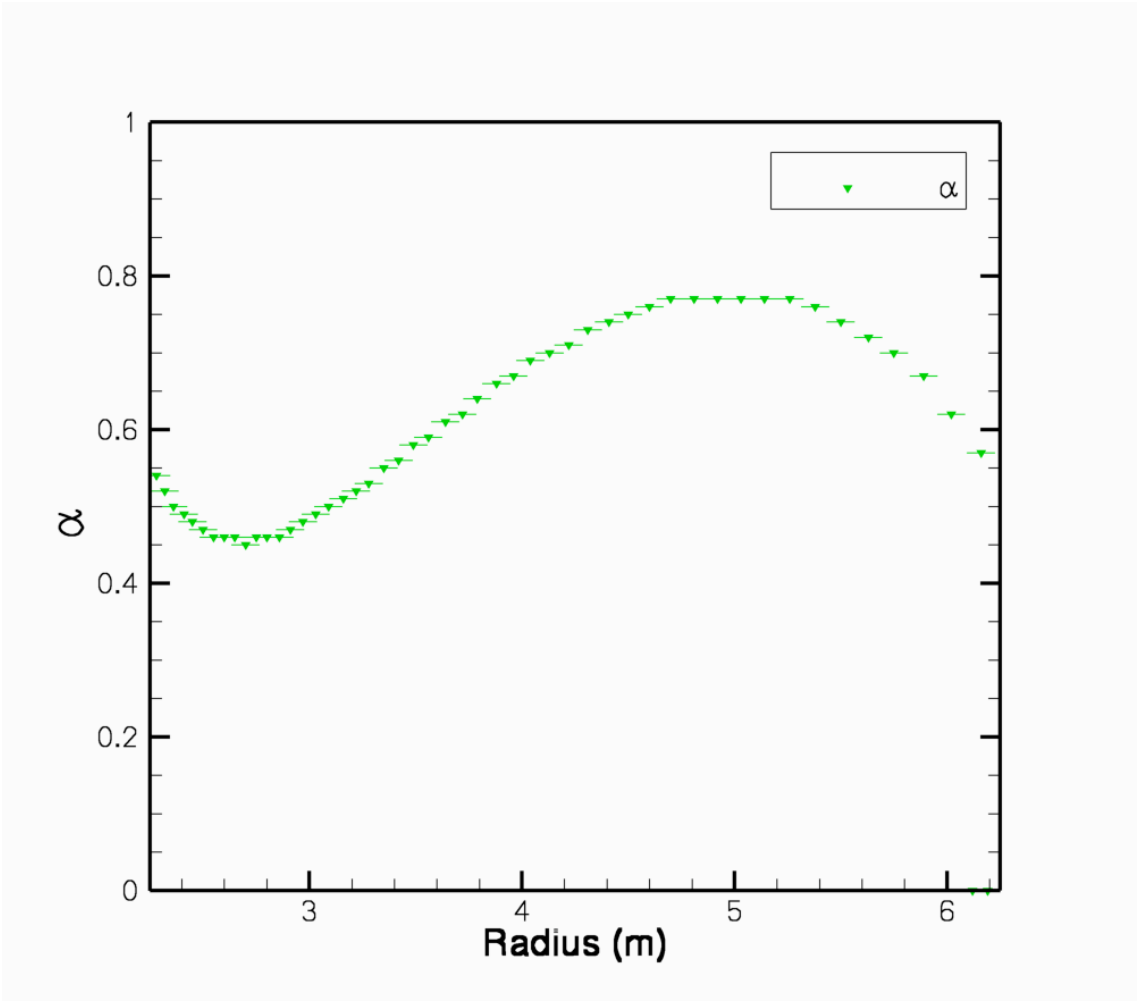

Figure B.14: The ration bewteen the meridian and circumferential stresses at time = 0.4 second.

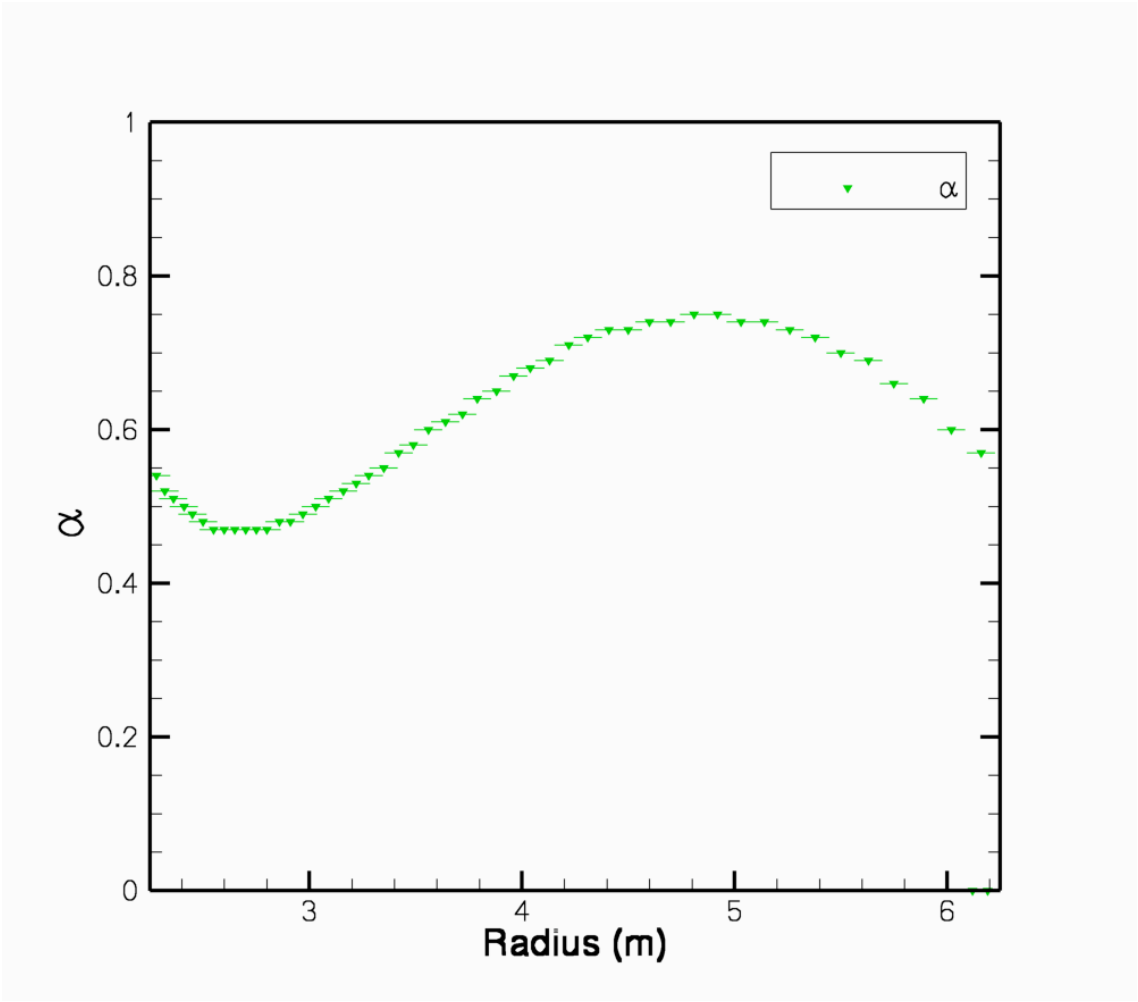

Figure B.15: The ration bewteen the meridian and circumferential stresses at time = 0.5 second.

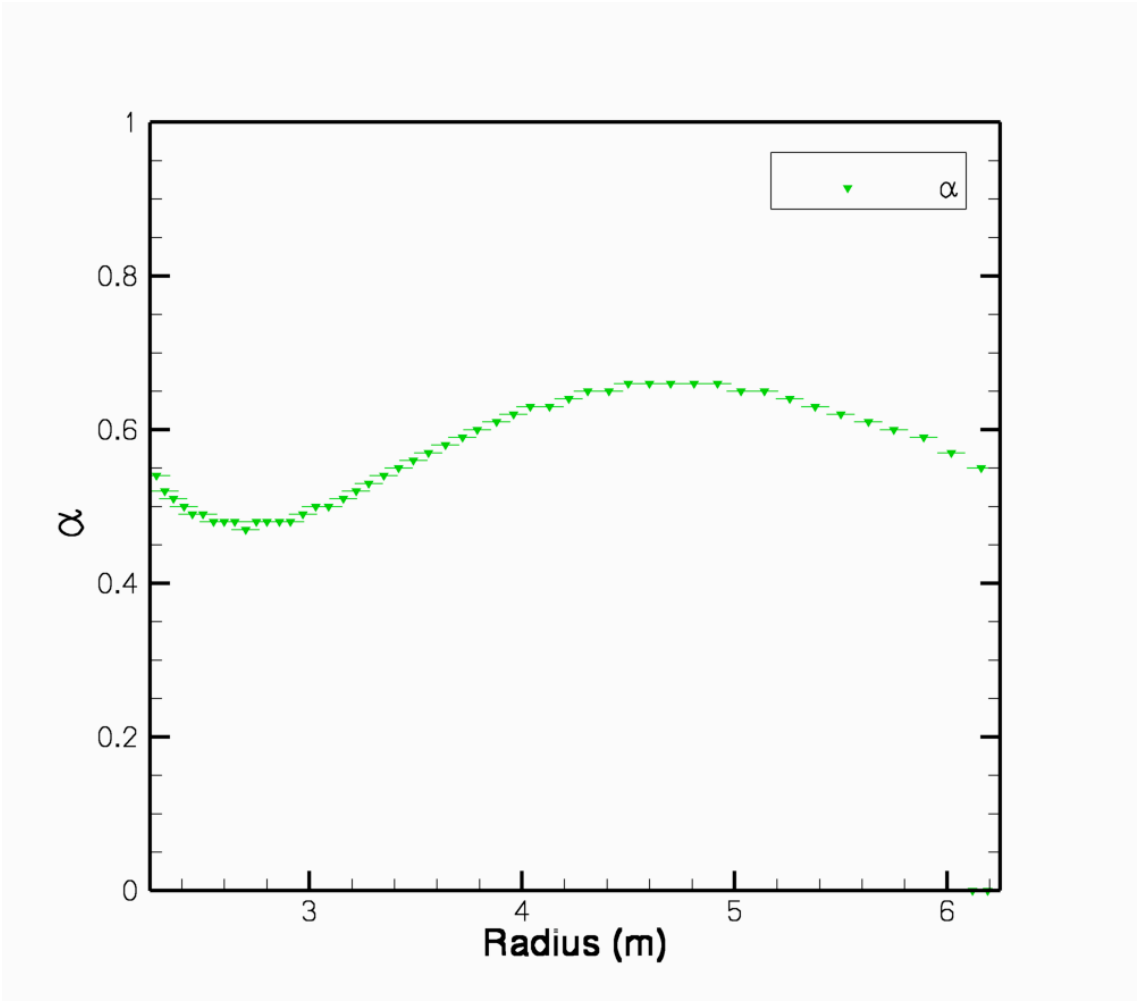

Figure B.16: The ration bewteen the meridian and circumferential stresses at time = 0.75 second.

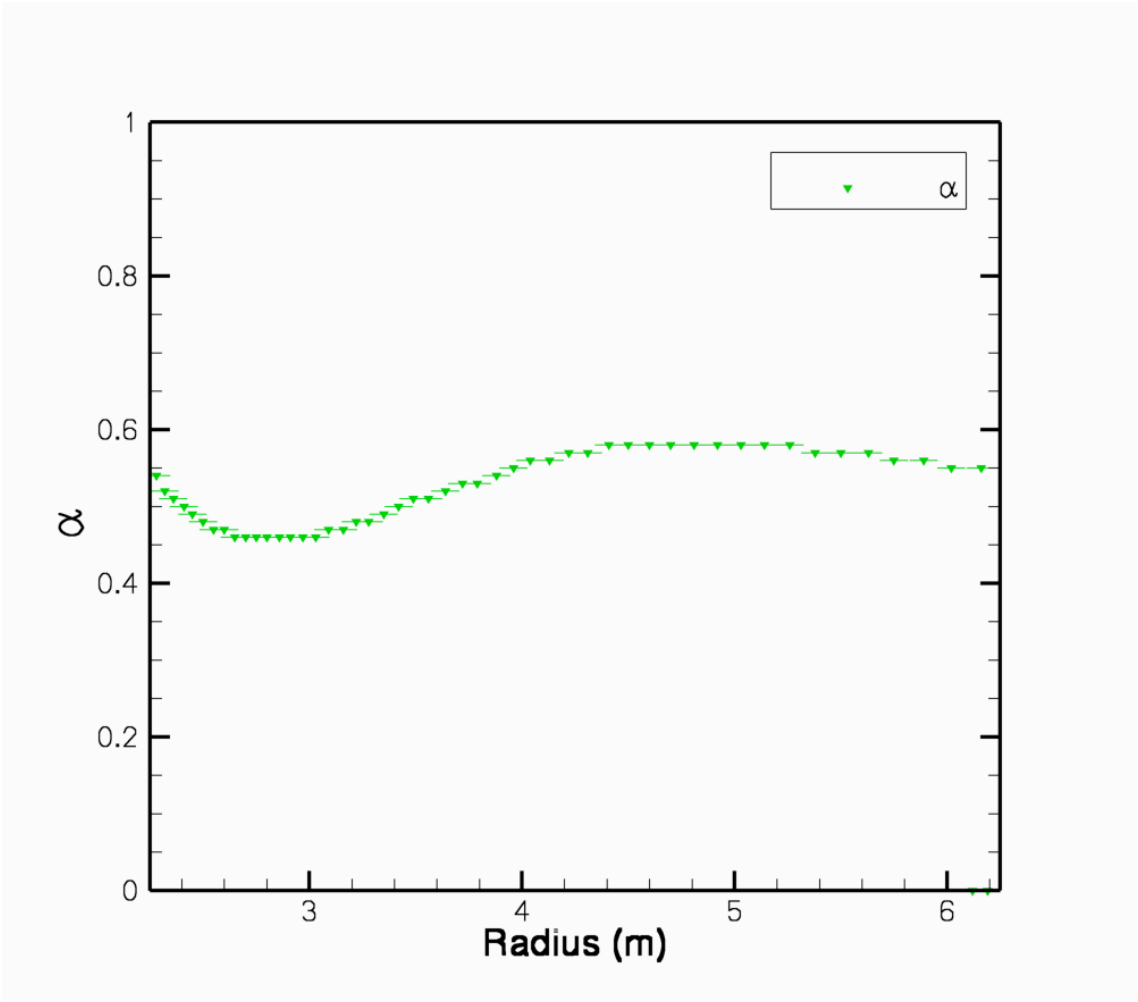

Figure B.17: The ration bewteen the meridian and circumferential stresses at time = 1.0 second.

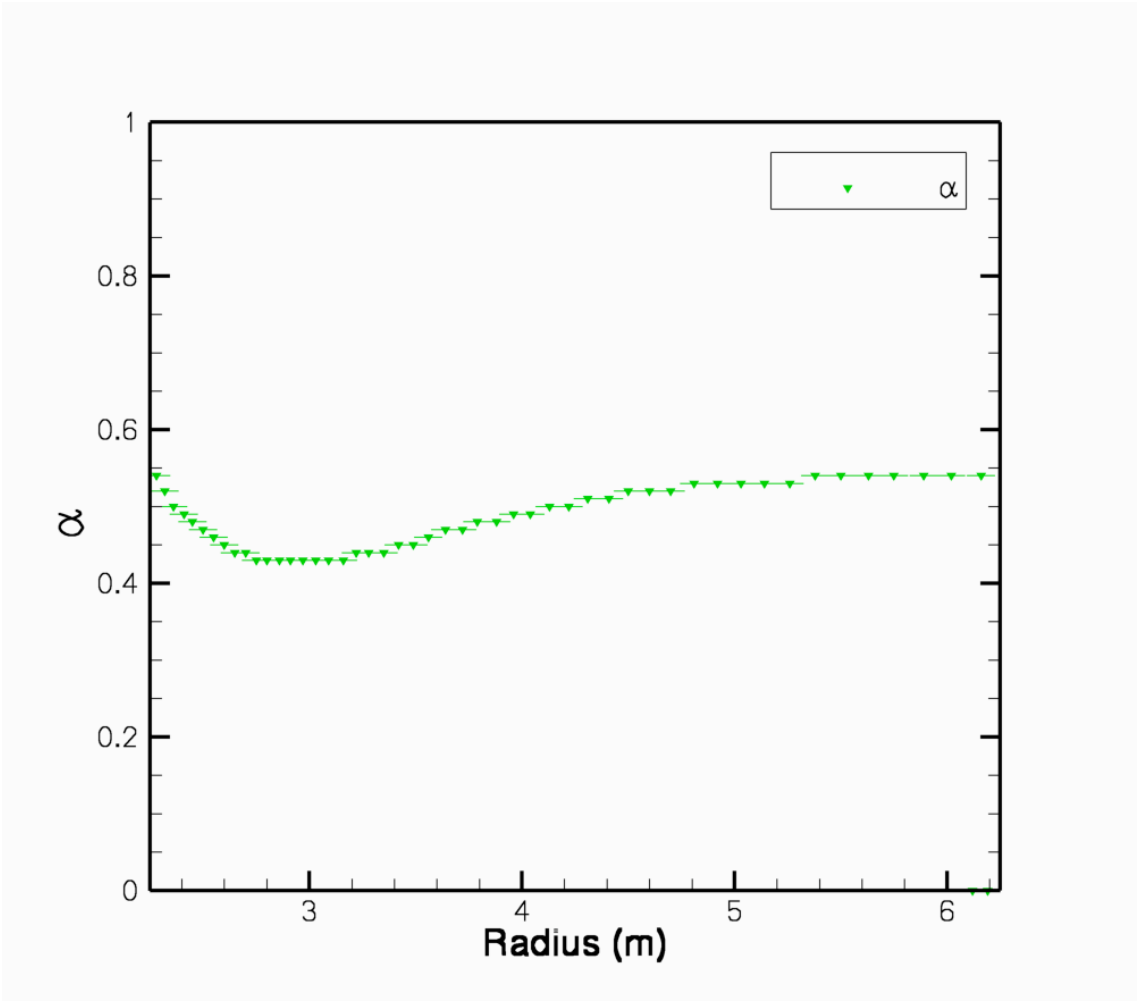

Figure B.18: The ration bewteen the meridian and circumferential stresses at time = 1.25 second.

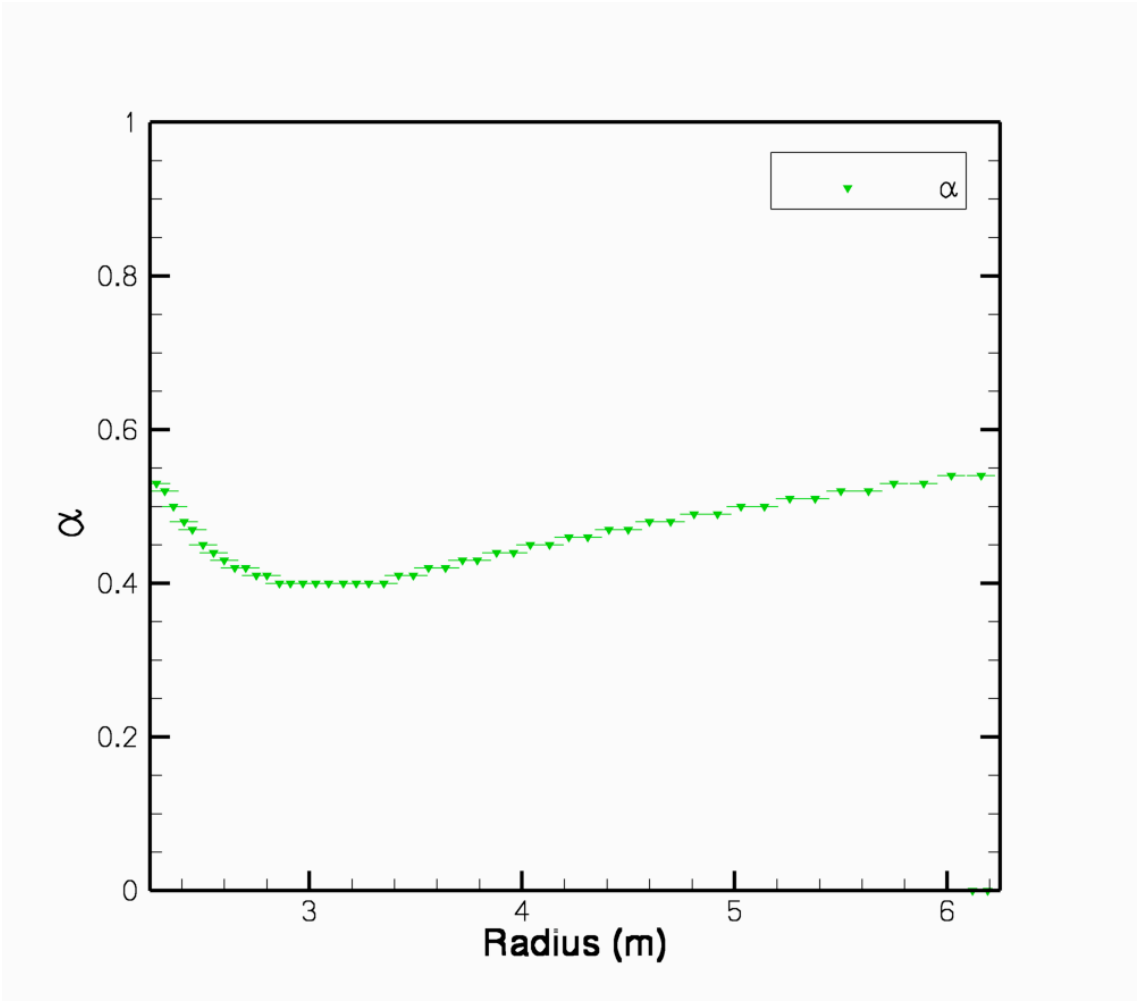

Figure B.19: The ration bewteen the meridian and circumferential stresses at time = 1.5 second.

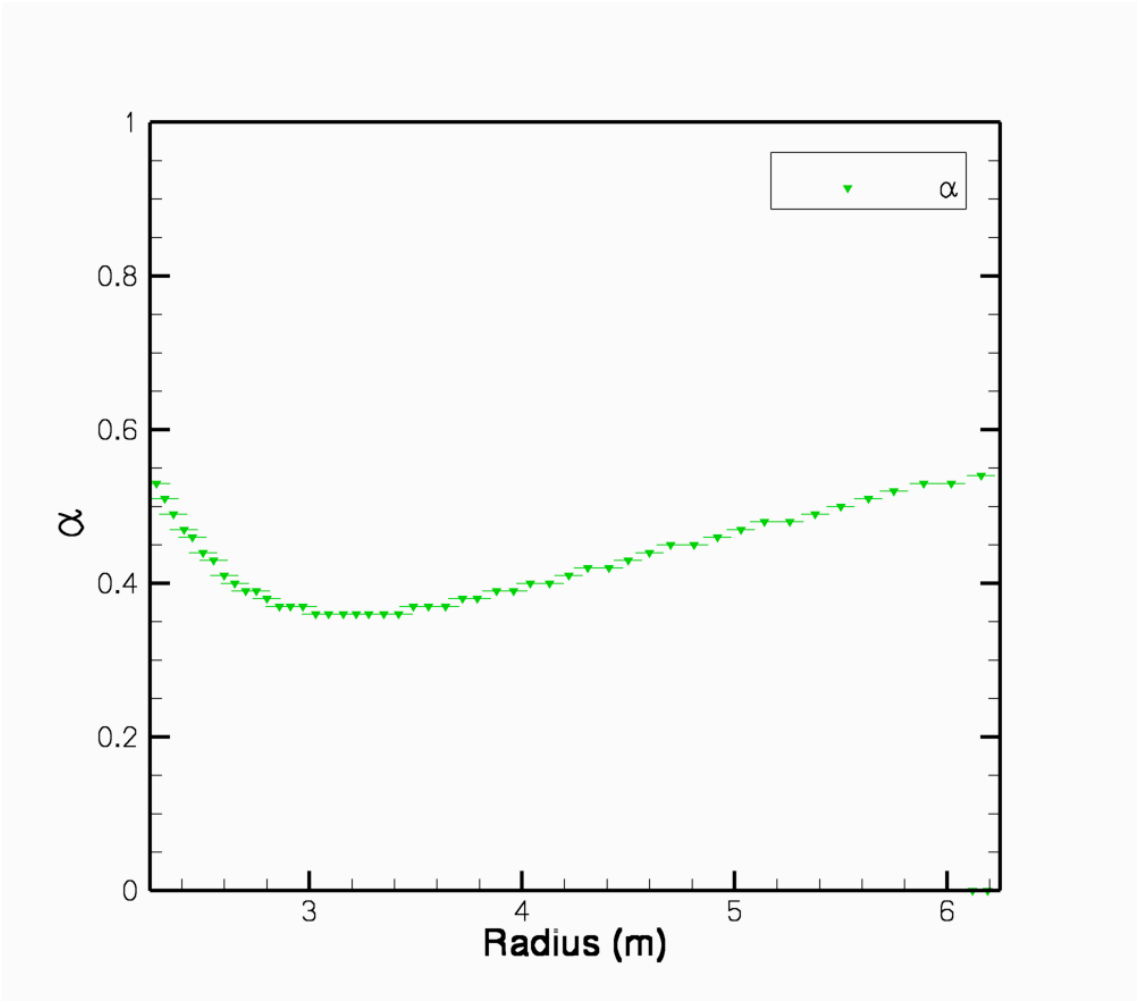

Figure B.20: The ration bewteen the meridian and circumferential stresses at time = 2.0 second.