Feasibility of a Self-Erecting Shelter with an Inflatable-Fabric-Arch-Supported Roof and Rigid Walls

Jay Wegner
University of Maine, jay.wegner@maine.edu

Follow this and additional works at: https://digitalcommons.library.umaine.edu/etd

Recommended Citation
https://digitalcommons.library.umaine.edu/etd/3114
FEASIBILITY OF A SELF-ERECTING SHELTER WITH AN INFLATABLE-FABRIC-ARCH-SUPPORTED ROOF AND RIGID WALLS

By

Jay Wegner

B.S. University of Maine, 2017

A THESIS

Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science (in Civil Engineering)

The Graduate School
The University of Maine
August 2019

Advisory Committee:

William G. Davids, Professor of Civil Engineering, Advisor
Edwin Nagy, Lecturer of Civil Engineering
Roberto Lopez-Anido, Professor of Civil Engineering
FEASIBILITY OF A SELF-ERECTING SHELTER WITH AN INFLATABLE FABRIC

ARCH-SUPPORTED ROOF AND RIGID WALLS

By Jay Wegner

Thesis Advisor: Dr. William G. Davids

An Abstract of the Thesis Presented
in Partial Fulfillment of the Requirements for the
Degree of Master of Science
in Civil Engineering
August 2019

Rapidly erected shelters are often required due to military needs, natural disasters, and humanitarian crises. Creating shelters quickly can be difficult due to the amount of labor required for erection. One popular class of rapidly erected shelters consists of parallel, inflatable fabric arches that spring from the ground and are covered with tent fabric. However, while quickly erected, these structures require large capacity air compressors, lack the protection provided by a rigid-walled shelter, and usable interior space is compromised by the shape of the arches. The focus of this research is the exploration of a hybrid inflatable-rigid wall structure that will overcome some or all of these issues. A central question is whether inflatable arches can be used to drive erection of such a structure as part of a mechanism. Semi-circular, inflatable arches were clamped to create the necessary arch span and inflated to heights below their full rise, and the vertical force required to resist additional arch rise was measured as a function of inflation pressure to assess the lifting capacity of one arch. Testing revealed that inflatable arches have a high tendency to bend out-of-plane and the shape that they take prior to inflating to their full rise can significantly effect their lifting capacity. Additional tests were conducted to assess the ability of a single arch to erect folding, exterior rigid walls. These tests showed that a single arch was able to erect a structure with a folding wall panel on each side weighing 420 N at a modest inflation pressure of
about 110 kPa, well below the normal operating arch pressure of 344 kPa. Finite-element analyses of the inflated arches were conducted using nonlinear inflatable beam theory to explore the load capacity of the erected structure and assess its gravity load capacity relative to a conventional, soft-walled arch shelter. These analyses indicate that the arch structure with rigid walls has a gravity load capacity slightly higher than that of a conventional soft-walled arch for inflation pressures ranging from 137 kPa to 344 kPa. Future research on this structure should focus on refining estimates of arch lifting capacity during inflation, exploration of alternative forms of hybrid rigid-airbeam structures, and refining finite element models used to determine the inflated structure’s capacity.
ACKNOWLEDGEMENTS

Before we begin, I would like to thank individuals who have helped me through my time conducting this research. Firstly and most of all I would like to thank Dr. Bill Davids for the opportunity to pursue a graduate degree here at the University of Maine. His guidance and immense patience throughout the process was invaluable and I cannot thank him enough. I hope the findings of this research at least aid you in becoming rich and famous.

If it were not for the US Combat Capabilities Development Command Soldier Center, this research may not have happened. I appreciate their support and interest in airbeam supported shelters. I would also like to thank the University of Maine Graduate School for their aid in housing me in Stodder Hall for the past year as a Graduate Community Coordinator.

I would also like to thank Scott Tomlinson for his guidance in the operations at the Advance Structures and Composites Center. Also Kevin Brayley for his very informative and well documented thesis. Additionally the ASCC lab operations and instrumentation team were very helpful in the securing of testing space and the setting up of experiments.

I am very grateful for the professors of the Civil and Environmental Engineering Department. I have found many new role models. Their enthusiasm and teachings will be with me throughout my career.

I would also like to thank my father Dale for his occasional trips up here to go explore some of the most remote parts of Maine be it the Goose Rocks light house, Kokadjo, and many others. And my mother Gail for going on my Dad’s trips to the middle of nowhere as well as her moral support and encouragement.

I would also like to thank my colleagues PianPian Chen, Anthony Verzoni, and Adam Letourneau for their moral support and office banter. Finally I would like to thank the following undergraduate research assistants for their help in the conducting and manufacturing of experiments: Joshua Clark, Mitch St. Peter, Mikayla Rodgers, and Jordan Yoder.
# TABLE OF CONTENTS

ACKNOWLEDGEMENTS ................................................................................................................... v

LIST OF TABLES ................................................................................................................................. x

LIST OF FIGURES ............................................................................................................................... xi

CHAPTER 1 INTRODUCTION .................................................................................................................. 1

1.1. History of Inflatable Arch-Supported Structures ........................................................................ 1

1.2. Thesis Objectives and Tasks .................................................................................................... 2

CHAPTER 2 REVIEW OF CURRENT SOFT-WALL AND RIGID WALL TECHNOLOGIES .................. 3

2.1. Container-Based Shelters ......................................................................................................... 3

2.1.1. Tactical Two-Sided Expandable Shelter ............................................................................. 4

2.2. Soft-Wall Shelters ..................................................................................................................... 5

2.2.1. CAMSS 18TAC35 ............................................................................................................... 6

2.2.2. HDT Global Model 2021 (TEMPER Shelter) ..................................................................... 6

2.3. Review of Existing Literature .................................................................................................. 8

2.4. General HRAB Concept .......................................................................................................... 9

CHAPTER 3 PRELIMINARY TESTING TO ASSESS AIRBEAM ERECTION CAPACITY ............ 10

3.1. Overview .................................................................................................................................... 10

3.2. Dynamic Inflation Tests .......................................................................................................... 12

3.2.1. Test Set-Up .......................................................................................................................... 12

3.2.2. Results .................................................................................................................................. 16
3.3. Static Inflation Tests .................................................................................................................. 20

3.3.1. Pinned Test Set-Up .................................................................................................................. 21

3.3.2. Pinned Results ....................................................................................................................... 22

3.3.3. Roller Test Set-Up .................................................................................................................. 24

3.3.4. Roller Results ........................................................................................................................ 27

3.4. Summary .................................................................................................................................... 30

CHAPTER 4 DESIGN OF FULL-SCALE MOCK-UP AND TESTING TO PROVE CONCEPT .................. 31

4.1. Development of HRAB Concept ............................................................................................... 31

4.2. HRAB Concept tests ................................................................................................................ 33

4.2.1. Test Set-Up ............................................................................................................................ 33

4.2.2. Instrumentation and Test Protocol ........................................................................................ 38

4.2.3. HRAB Concept Test Results ................................................................................................ 41

4.3. Summary .................................................................................................................................... 48

CHAPTER 5 DETAILED ANALYSIS OF HYBRID AIRBEAM SHELTER ............................................ 49

5.1. Overview of Analysis Methods ................................................................................................ 49

5.2. Extension of Existing Non-Linear FE Techniques in PAA to Simulate the HRAB Structure .... 53

5.2.1. Element Verifications ............................................................................................................ 58

5.3. Generation of the HRAB Model ............................................................................................... 73

5.3.1. Element Properties ............................................................................................................... 73

5.3.1.1. Roof Properties .................................................................................................................. 73
BIOGRAPHY OF THE AUTHOR

226
LIST OF TABLES

Table 1: Mean and Coefficient of Variation of Lift Values ................................................................. 19
Table 2: Mean and Coefficient of Variation of Static Inflation Test: Roller ........................................ 29
Table 3: HRAB Experiment Strut Properties ....................................................................................... 36
Table 4: HRAB Wall Panel Properties ............................................................................................... 38
Table 5: HRAB Wall Panel Pressures ................................................................................................. 47
Table 6: HRAB Wall Element Properties ........................................................................................... 74
Table 7: HRAB Strut Element Properties ............................................................................................ 75
Table 8: HRAB Airbeam Element Properties ...................................................................................... 76
Table 9: HRAB Airbeam Element Axial and Shear Moduli Properties ............................................... 76
Table 10: HRAB Parametric Study Parameters .................................................................................. 82
Table 11: HRAB Guy Angle and Pretension Parametric Study Results: Model Gravity Load Capacities in Pascals .......................................................................................................................... 84
Table 12: HRAB Parametric Study Parameters .................................................................................. 86
Table 13: Equilibrium Values of HRAB West Wall, 275 kPa, 200 N Guy PT, HZ ............................... 90
Table 14: Moments Applied to HRAB West Wall, 275 kPa, 200 N Guy PT, HZ ............................... 90
Table 15: HRAB & TEMPER Load Capacities .................................................................................... 91
Table 16: Load Capacity Increase of HRAB Compared to TEMPER, Percent .................................... 92
Table 17: HRAB & TEMPER Load Capacities with Roof Distributed Load ......................................... 94
Table 18: Load Capacity Increase of HRAB with Roof Distributed Load Compared to TEMPER, Percent ........................................................................................................................................... 94
LIST OF FIGURES

Figure 1: Tent Extendable Modular PERsonnel shelter (Brayley, 2011) .................................................. 1
Figure 2. Shelter, tactical, expandable, two-side (“The United States Army | Natick,” n.d.) ...................... 4
Figure 3. Airframe diagram (Kabche et al., 2011) .............................................................................. 5
Figure 4. Air Supported TEMPER shelter (U.S. Army Natick Soldier RD&E Center, 2017) .............. 7
Figure 5. MBPS system around TEMPER Shelter (Tino, 2011) .............................................................. 7
Figure 6: Airbeam Pin Connection ..................................................................................................... 11
Figure 7: HRAB Force Test Pin Connection Clamp .............................................................................. 11
Figure 8: HRAB Weight Hanger Test Setup ......................................................................................... 13
Figure 9: Airbeam Folds ..................................................................................................................... 13
Figure 10: Airbeam Prior to Inflation ................................................................................................. 14
Figure 11: Airbeam Bending Out-of-Plane .......................................................................................... 15
Figure 12: Static & Dynamic Airbeam Restraining Force Load Cell. .................................................. 16
Figure 13: Dynamic Inflation Test 600 N #5, Load vs. Pressure ......................................................... 17
Figure 14: Dynamic Inflation Test 600 N #5, Load vs. Pressure 138 kPa to 150 kPa ......................... 17
Figure 15: Dynamic Inflation Test 600 N #5, Apex Rise vs. Pressure .................................................. 18
Figure 16: Airbeam Lift Load vs Airbeam Internal Pressure ............................................................... 19
Figure 17: Dynamic Inflation Test 0 N #0, Final Shape ..................................................................... 20
Figure 18: Static Inflation Restraining Device ..................................................................................... 21
Figure 19: Static Inflation Test of M11870-8 ........................................................................................ 22
Figure 20: Airbeam Restraining Force vs. Rise at 137 kPa ................................................................ 23
Figure 21: Typical M11162-3 Shape .................................................................................................. 24
Figure 22: Typical M11870-8 Shape .................................................................................................. 24
Figure 23: Triangular Final Shape of M11870-8 at 1155 mm .............................................................. 24
Figure 24: Airbeam Roller Connection (X is out of the page) ................................................................. 26
Figure 25: Static Inflation Test with Roller Supports ................................................................. 26
Figure 26: Static Inflation Test: Roller, Tension Tie Force vs Airbeam Pressure ................................... 27
Figure 27: Static Inflation Test: Roller, Airbeam Restraining Force vs Airbeam Pressure ................. 28
Figure 28: Static Inflation Test: Roller, Tension Tie and Restraining Force at 137 kPa......................... 29
Figure 29: Static Inflation Test: Roller, Tension Tie and Restraining Force at 206 kPa....................... 30
Figure 30: HRAB Concept Cross-Section ............................................................................................ 31
Figure 31: HRAB Concept Side View (guy lines not shown) .............................................................. 32
Figure 32: Hybrid Rigid Airbeam Concept Deployment Process (guy lines not shown for clarity)....... 33
Figure 33: HRAB Experimental Model ............................................................................................ 34
Figure 34: HRAB Experimental Model Components ........................................................................... 34
Figure 35: HRAB Pin Connection ....................................................................................................... 35
Figure 36: HRAB Strut and Roof Rope Connection ........................................................................... 36
Figure 37: HRAB Scale Model Wall Panel ......................................................................................... 37
Figure 38: HRAB Experimental Model As-Built Dimensions .............................................................. 39
Figure 39: HRAB Experimental Model Instrumentation Diagram ....................................................... 39
Figure 40: HRAB Experimental Model 16° and 0° wall panel openings ........................................ 40
Figure 41: HRAB Folded wall starting angle dimensions .................................................................. 40
Figure 42: HRAB Model Deflated ...................................................................................................... 41
Figure 43: HRAB Experimental Model Asymmetric Wall Panel Opening ........................................... 41
Figure 44: HRAB Experimental Model Symmetric Wall Panel Opening ........................................... 42
Figure 45: HRAB Experimental Model Pop Through to Time 1 ...................................................... 42
Figure 46: 8° Initial Opening, Time 2 ................................................................................................. 43
Figure 47: 8° Initial Opening, Time 3 ................................................................................................. 43
Figure 48: 8" Initial Opening, Internal Forces in HRAB Components vs. Internal Pressure ........................................ 44
Figure 49: 8" Initial Opening, Wall Panel Angle vs. Internal Pressure ........................................................................... 45
Figure 50: 16° Initial Opening #2, Time 1 ...................................................................................................................... 46
Figure 51: 16° Initial Opening #2, Forces in HRAB Components vs. Internal Pressure ............................................ 46
Figure 52: HRAB Concept Wall Panel Opening Pressure Results ...................................................................................... 47
Figure 53: Braided and Strapped Airbeam Cross Section ............................................................................................... 49
Figure 54: Airbeam Wrinkling ........................................................................................................................................ 51
Figure 55: HRAB Structure Element Locations ................................................................................................................... 53
Figure 56: Three Noded Quadratic Beam Element ............................................................................................................. 54
Figure 57: Large Deformation Equilibrium Check for Tension Only Elements .............................................................. 58
Figure 58: Wall & Strut Element Verification: Simply Supported Beam, Model and Reactions ............................. 59
Figure 59: Wall & Strut Element Verification: Pinned Beam, Model and Reactions ......................................................... 60
Figure 60: Wall & Strut Element Verification: Pinned Beam, Member Equilibrium ......................................................... 61
Figure 61: Wall & Strut Element Verification: Pinned Beam, Model Displacement ......................................................... 61
Figure 62: Wall & Strut Element Verification: Pinned Beam, Internal Member Forces ...................................................... 62
Figure 63: Wall & Strut Element Verification: Pinned Beam, Model Member Forces ......................................................... 63
Figure 64: Wall & Strut Element Verification: Simple Beam at 45°, Model and Reactions ........................................... 64
Figure 65: Wall & Strut Element Verification: Simple Beam at 45°, Member Equilibrium ........................................... 64
Figure 66: Wall & Strut Element Verification: Simple Beam at 45°, Model Displacements ........................................... 65
Figure 67: Wall & Strut Element Verification: Simple Beam at 45°, Model Member Forces ........................................... 66
Figure 68: Wall & Strut Element Verification: Fixed-Hinged Beam: Model and Reactions ........................................... 67
Figure 69: Wall & Strut Element Verification: Fixed-Hinged Beam at 45°, Member Equilibrium .................................... 68
Figure 70: Wall & Strut Element Verification: Fixed-Hinged Beam, Model Displacement ............................................ 68
Figure 71: Wall & Strut Element Verification: Fixed-Hinged Beam, Model Member Forces ......................................... 69
Figure 72: Roof Element Verification: Fixed Beam, Model and Load Discretization ............................................. 70
Figure 73: Roof Element Verification: Fixed Beam, Model Reaction Forces ...................................................... 70
Figure 74: Roof Element Verification: Fixed Beam, Member Equilibrium ......................................................... 71
Figure 75: Roof Element Verification: Fixed Beam, Model Displacements ......................................................... 71
Figure 76: Roof Element Verification: Fixed Beam, Model Member Forces ....................................................... 72
Figure 77: HRAB Model Strut .............................................................................................................................. 74
Figure 78: HRAB Discretized Model .................................................................................................................. 77
Figure 79: TEMPER Discretized Model ............................................................................................................. 78
Figure 80: Concentration of Roof Load ........................................................................................................... 79
Figure 81: Load Fraction vs. Apex Displacement, Airbeam, 206 kPa, 0 N Guy Pretension ................................... 81
Figure 82: Applied Load vs. Guy Angle, HRAB No Hz Load ............................................................................... 83
Figure 83: Guy Angle vs. Peak Applied Load, HRAB with Hz Load ............................................................... 83
Figure 84: HRAB 275 kPa, 200 N Guy PT, HZ .................................................................................................... 87
Figure 85: TEMPER 275 kPa, 200 N Guy PT, HZ ................................................................................................ 87
Figure 86: Equilibrium of HRAB Structure, 275 kPa, 200 N Guy PT, HZ ........................................................... 88
Figure 87: Equilibrium of HRAB West Wall, 275 kPa, 200 N Guy PT, HZ ........................................................... 89
Figure 88: Horizontally Loaded Models .......................................................................................................... 92
Figure 89: Non-Horizontally Loaded Models .................................................................................................... 93
Figure 90: HRAB Discretized Model with Roof Distributed Loads .......................................................... 93
Figure 91: 137 kPa, 0 N Guy Pretension, 45° Guy Angle, HZ Load ................................................................. 105
Figure 92: 137 kPa, 200 N Guy Pretension, 45° Guy Angle, HZ Load ............................................................. 106
Figure 93: 206 kPa, 0 N Guy Pretension, 45° Guy Angle, HZ Load ................................................................. 107
Figure 94: 206 kPa, 200 N Guy Pretension, 45° Guy Angle, HZ Load ............................................................. 108
Figure 95: 275 kPa, 0 N Guy Pretension, 45° Guy Angle, HZ Load ................................................................. 109
Figure 96: 275 kPa, 200 N Guy Pretension, 45° Guy Angle, HZ Load ................................................................. 110
Figure 97: 344 kPa, 0 N Guy Pretension, 45° Guy Angle, HZ Load ................................................................. 111
Figure 98: 344 kPa, 200 N Guy Pretension, 45° Guy Angle, HZ Load ................................................................. 112
Figure 99: 137 kPa, 0 N Guy Pretension, 45° Guy Angle, No HZ Load ............................................................... 113
Figure 100: 137 kPa, 200 N Guy Pretension, 45° Guy Angle, No HZ Load ............................................................. 114
Figure 101: 206 kPa, 0 N Guy Pretension, 45° Guy Angle, No HZ Load ............................................................. 115
Figure 102: 206 kPa, 200 N Guy Pretension, 45° Guy Angle, No HZ Load ............................................................. 116
Figure 103: 275 kPa, 0 N Guy Pretension, 45° Guy Angle, No HZ Load ............................................................... 117
Figure 104: 275 kPa, 200 N Guy Pretension, 45° Guy Angle, No HZ Load ............................................................. 118
Figure 105: 344 kPa, 0 N Guy Pretension, 45° Guy Angle, No HZ Load ............................................................... 119
Figure 106: 344 kPa, 200 N Guy Pretension, 45° Guy Angle, No HZ Load ............................................................. 120
1.1. History of Inflatable Arch-Supported Structures

Inflatable arch-supported structures are commonly used for temporary shelters in military and disaster-relief applications. These structures serve to cover the inhabitants and are currently required to withstand snow and wind loads when fully erected. The inflated fabric arches, often called airbeams, are the main load-carrying members, and consist of an impermeable air bladder surrounded by either a braided or woven fabric. Braided airbeams include reinforcing straps on their circumference. An example of this shelter is the Tent Extendable Modular PERsonnel (TEMPER) which distributes the applied load through a canvass cover to the airbeams.

Figure 1: Tent Extendable Modular PERsonnel shelter (Brayley, 2011) The study of air inflated beams was originally undertaken by Stein and Hedgepeth (Stein and Hedgepeth, 1961) where theory of elasticity was used to develop load-deflection relationships. Since then, research has progressed in the development of finite element models of inflated fabric beams (Davids, 2007), and the experimental testing and numerical simulation of inflated fabric arches (Brayley et al., 2012; Davids, 2009). While airbeams are capable of lifting the enclosing tent fabric and attachments, their ability to lift additional loads has not been assessed. Additionally, tents have inherently low resistance to blast and ballistic loads due their use of fabric covering.
1.2. Thesis Objectives and Tasks

The objectives of this research were to: 1) Quantify the maximum lifting force of a TEMPER tent airbeam during inflation; 2) Develop a preliminary design of an airbeam-supported tent with integral rigid walls to form a Hybrid Rigid-wall AirBeam (HRAB) structure, and experimentally assess its ability to erect itself by airbeam inflation; and 3) Quantify the gravity load capacity of an HRAB relative to a conventional, stand-alone TEMPER tent airbeam. Four major tasks were undertaken to achieve these objectives.

1. Currently available shelter types were reviewed to define advantages and shortcomings of common shelter systems. (Chapter 2).

2. The maximum lifting capacity of the TEMPER airbeam was quantified using both restrained static and dynamic lift experiments to assess the ability of an airbeam to drive erection of a larger structure (Chapter 3).

3. Based on the information and data in Chapters 2 and 3, a self-erecting HRAB structural concept is proposed that will increase useable internal structure volume and allow for simple up-armoring without applying additional load on the airbeams. Full scale model tests of the HRAB concept structure were conducted to verify self-erection (Chapter 4).

4. Materially and geometrically nonlinear, post-erection beam-based finite-element analyses of both a stand-alone airbeam and a Hybrid Rigid-Walled Airbeam (HRAB) were conducted to assess their relative gravity load capacities (Chapter 5).
CHAPTER 2

REVIEW OF CURRENT SOFT-WALL AND RIGID-WALL TECHNOLOGIES

To define a baseline and determine the current state-of-the-art in shelter technology, a literature review was conducted. This review involved targeted searches on the internet, analysis of peer-reviewed published literature, and an examination of available military, commercial, and recreational tents. Tents found ranged in sizes from large event structures to small backpacking shelters. Existing shelters are grouped into two categories in this review: container-based rigid wall shelters, and soft-wall shelters.

2.1. Container-Based Shelters

The container-based shelter class of shelters, commonly referred to as rigid-wall shelters, comes with predetermined dimensions usually in the form of standard shipping containers as defined by the International Organization for Standardization (ISO). These shelters often come fully assembled allowing them to be operational in a minimal amount of time. Some shelters come ready to use; however, they are often the size of a standard ISO container and the usable area equals the packed dimensions. Others are designed such that the longer walls are able to slide outward or fold down, thereby nearly tripling the usable area while maintain the packed ISO dimensions (U.S. Army Natick Soldier RD&E Center, 2017). Rigid-wall shelters can offer enhanced ballistic and blast resistance compared with hybrid and soft-wall shelters through their usage of flat surfaces for walls. However, container-based shelters often cannot be packed into smaller volumes causing them to require a large storage space. In addition, the parts or all of the container are often made of metal, making their weight per footprint area larger when compared to soft-wall and hybrid shelters of similar sizes. More modern rigid wall shelters are made of composite materials allowing them to be lighter (The Will-Burt Company, 2016) over the older foam and beam, honeycomb core, and steel shelters.
2.1.1. Tactical Two-Sided Expandable Shelter

The Shelter, Tactical, Expandable, Two Side, shown in Figure 2, is an example of an expandable rigid wall shelter. The shelter is transported in its packed ISO dimensions. Then at its destination, its longer walls are deployed, nearly tripling its usable interior area over a non-expandable ISO rigid wall shelter. The shelter is fully equipped with a breaker panel and receptacles for powering devices, as well as removable panels for environmental control unit (ECU) ducting and complexing between other shelters. (U.S. Army Natick Soldier RD&E Center, 2017)

- **Design Benefits:**
  - Rigid walls
  - Expandable shelter in terms of width
  - ISO shipping compatible

- **Opportunities for Improvement:**
  - Self Weight, 6900 lbf (30.6 kN) (U.S. Army Natick Soldier RD&E Center, 2017)
  - Low expansion ratio (area packed to area expanded)

![Figure 2. Shelter, tactical, expandable, two-side ("The United States Army | Natick," n.d.)](image)
2.2. Soft-Wall Shelters

Soft-wall shelters “include air supported and frame supported fabric structures that are transported and then erected or assembled on site” (Joint Committee On Tactical Shelters, 2012). This class includes airbeam shelters and metal framed tents. Unlike traditional fabric on rigid frame shelters, airbeam-supported shelters such as the TEMPER tent utilize an ‘airframe’. An airframe is a frame that is made of airbeams which are connected with tent fabric or other airbeams. Tent fabric is attached during manufacturing to the airframe to create the shelter. This fabric combined with guy lines (see Figure 1 and Figure 3 Error! Reference source not found. for guy line placement), provide resistance to side-sway buckling and out of plane motion of the airbeams.

Some soft-wall shelters utilize folding frames that can include interior liners, while other shelters do not fold and must be erected by attaching each piece of the rigid frame together. There are also shelters that utilize composite materials in the rigid frame of the shelter such as the Carbon Multi-Purpose Shelter CBZ 1200/300/496, which reduces weight over their steel framed counterparts (HTS Roder Rapid Deployment Shelters, n.d.). Two widely-used examples of softwall shelters, one rigid-framed and one air-supported, are reviewed in more detail below to highlight advantages and disadvantages of each.
2.2.1. CAMSS 18TAC35

The CAMSS 18TAC35 is a modern version of the Tent, Extendable, Modular, Personnel shelter which has been in use since 1984 (Carr et al., 1984). The CAMSS 18TAC35 has a quick set up time and nearly the same square footage of 630 ft$^2$. The 18TAC35 shelter utilizes a 1-piece folding frame, which eliminates the need for extra tools and decreases set up time (CAMSS Shelters, 2016) over the comparable frame supported Tent, Extendable, Modular, Personnel (U.S. Army Natick Soldier RD&E Center, 2017).

- **Design Benefits:**
  - Low areal weight
  - Tent Pole technology can create a structure able to withstand soft-wall shelter tests outlined in TOP-10-2-175 (CAMSS Shelters, n.d.)

- **Opportunities for Improvement:**
  - Requires 4 persons to set up
  - No simple way to add ballistic protection

2.2.2. HDT Global Model 2021 (TEMPER Shelter)

The Airbeam-supported Tent Extendable Modular PERsonnel (here on referred to as the TEMPER shelter) (U.S. Army Natick Soldier RD&E Center, 2017) has been chosen by the US Army for use in the Force Provider base camp model (HDT Global, 2016), and a picture of the shelter can be found in Figure 4. The TEMPER tent can be ordered in 17 different variations (HDT Global, 2016) to fit operational need. Shelters can connect to other TEMPER shelters, Tricon containers, ISO containers, vehicles, and air lock interfaces through vestibules. Additionally personnel and cargo doors can be installed on any side. Shelters include an external environmental control unit (ECU), duct openings, a removable lightweight thermal liner with an air distribution plenum, and hanging straps for lights and other accessories (HDT Global, 2016). Airbeams are interconnected so that inflation is uniform. However, each beam is designed as an isolated pressure tube so if one beam develops a leak, only the punctured beam will deflate (Tom Artes, 2014). Currently, in order to add ballistic protection to the shelter an external wall of Modular
Ballistic Protection system panels must be propped up by supports around the exterior of the shelter, shown in Figure 5.

- **Design Benefits:**
  - Low areal weight of useable space
  - Utilizes airbeam technology allowing for quick set up times for the erected size, 2 persons 10 minutes (HDT Global, 2016)
  - Can be erected with personnel, no vehicles
  - Currently widely used

- **Opportunities for Improvement:**
  - Trailer mounted air compressor required
  - Can be punctured
  - Too heavy to be lifted by personnel only (Tom Artes, 2014)
  - No simple way to add ballistic protection
  - Usable internal volume limited by curved arch walls.

![Figure 4. Air Supported TEMPER shelter (U.S. Army Natick Soldier RD&E Center, 2017)](image)

![Figure 5. MBPS system around TEMPER Shelter (Tino, 2011)](image)

The operating pressure of the airbeams is typically 345 kPa creating a structural element that is capable of carrying significant vertical loads before wrinkling (Brayley, 2011). The large volume of air and high
pressure require large compressors to inflate the airframe in a reasonable amount of time. The benefit of airframe shelters is once the shelter is laid out and staked down, personnel are free to attend to other tasks while the shelter inflates.

Airbeam shelters currently range in size from 4.6 m x 4.6 m to 12.2 m x 42.7 m. The airbeams used in HDT Global shelters range from 6 m diameter semicircles to 12.1 m diameter semicircles (HDT Global, 2015, 2014). The benefits of the TEMPER airbeam shelter have led it to currently be used in the U.S Army Force Provider Base Camp (Gourley, 2015).

2.3. Review of Existing Literature

Prior research on inflatable arches and beams as well as the theory behind their behavior has been well documented (Brayley, 2011; Clapp, 2017). However these arches and beams were tested and analyzed in their fully inflated shapes to assess their load-carrying capacity as a function of parameters such as inflation pressure or fabric braid angle. Less research regarding the behavior of airbeams has been focused on modeling their geometry during inflation. Lampani and Gaudenzi (2010) modeled the behavior of coiled airbeams during their inflation. Zhan et al. (2014) modeled both the inflation behavior of a coiled tube and the inflation behavior of tube folded along its length in Z pattern. Zhou et al. (2019) numerically analyzed the effect of openings in pressurized structures and the effect on the structure’s time to collapse.

The research most similar to the study of airbeams as tools to perform work during inflation can be found in the field of soft robotics. Singh et al. (2018) studied the constriction forces generated by a coiled airbeam around a cylindrical object. Their experiments studied airbeams made of a latex elastomer tube surrounded by cotton fibers with braid angles of 33 and 27.5 degrees bonded to the tube with rubber cement and liquid latex. The results showed a linear increase in constricting force with an increase in internal airbeam pressure. Most recently Felt (2019) conducted experiments on a folded
airbeam in the shape of an accordion and their resulting bending moments, calculated from the internal tension in the airbeam and the airbeam’s curvature, generated from being inflated.

2.4. General HRAB Concept

This brief review has highlighted the advantages and shortcomings of currently-deployed soft and rigid wall military shelters. The short deployment times with minimal personnel requirements make TEMPER airbeam tents very attractive in many situations. However, their inherent lack of blast and ballistic protection coupled with reduced internal volume due to the arch shape can be shortcomings in some scenarios. In contrast, rigid-wall shelters provide more protection and useable internal volume for a given footprint, but this comes at the price of more complex deployment and increased weight. This thesis explores the development of a Hybrid Rigid AirBeam (HRAB) shelter that combines the benefits of both rigid and soft-walled shelters. As a pre-cursor to the development of the final HRAB concept explored in Chapters 4 and 5, Chapter 3 examines the lifting ability of an airbeam during inflation to assess its ability to drive a lifting mechanism.
CHAPTER 3

PRELIMINARY TESTING TO ASSESS AIRBEAM ERECTION CAPACITY

3.1. Overview

Initial experimental tests were undertaken to assess the ability of an airbeam to drive erection of a larger, HRAB structure. Two sets of tests were performed, both of which focused on the magnitude of lifting force an airbeam produces while being inflated. Although the geometry of the airbeams used in these tests did not represent the geometry of the airbeam used in the HRAB concept, the results still quantify the amount of force the airbeam is able to generate during inflation. The first round of tests were dynamic, and involved using the airbeam to lift weights suspended from its apex. The second round of tests were static, where the airbeam apex was restrained to a range of heights while inflation pressure was increased, and the apex restraining force was recorded. The airbeams used in these tests were nominally 6096 mm major diameter semi-circular arches with a 254 mm cross-sectional diameter made by Vertigo Inc. and the same arches used by Brayley (2011) in defining the buckling capacity of an inflated, strapped arch. Brayley observed the initial creation of the hinges in his experiments. It was expected to see much greater wrinkling, and therefore hinges, in the airbeams in the experiments reported here.

For these tests the airbeam was pinned at both ends to replicate the field condition where the airbeam is pinned to the ground with stakes. Pin connections restrict movement in the XY (horizontal) plane and Z (vertical) direction but allow rotation about the Y-axis. The airbeam was pinned at one end using the footing shown in Figure 6. This footing used two rods to pinch the airbeam just in front of the aluminum clamp on the airbeam. In this clamp, the wrinkling of the airbeam fabric allowed the rotation of the arch. The other end of the airbeam was clamped using steel pipes, see Figure 7, to shorten the arc-length of the airbeam to induce an apex rise of 1829 mm measured from the base of the arch. This reduced the span of the arch to 5588 mm. The reduced airbeam span was chosen based on an initial HRAB concept.
that ultimately proved infeasible. In this clamp, the arch was able to rotate through the rise with the rotation of one of the steel pipes held in place with U-bolts on either side of the pipe. Both footings were anchored to a strong floor with steel rods to prevent movement in the X direction. The steel cables seen in the images were used purely as a safety measure.

Figure 6: Airbeam Pin Connection

Figure 7: HRAB Force Test Pin Connection Clamp
3.2. Dynamic Inflation Tests

To observe and quantify the behavior of the airbeam during inflation under load, it was used to lift a series of weights. The amount of weight was increased until the airbeam could no longer lift the applied weight prior to reaching a target pressure.

3.2.1. Test Set-Up

The components of this test are identified in Figure 8 and will be referenced throughout this section. The airbeam was inflated until erection of the airbeam was achieved while attached to a 378 N steel hanger carrying additional steel plates weighing 43 N, 111 N, 154 N, and 222 N. A cable connected the weight hanger to the airbeam through two pulleys so that a vertical gravity load could be applied without having the set of weights directly below the airbeam. The cable length was set such that the center of the airbeam’s diameter would rise to 749 mm before the cable was taut. The amount of steel plates was increased until the airbeam could no longer lift the hanger and weights at the target pressure of 137 kPa. During these test the airbeam was braced against out-of-plane bending by using six wooden braces, shown in Figure 8. To mitigate the possibility of the arch catching on the pulley system on the ground, neoprene sheets were laid across obstructions under the arch apex, however these obstacles would still sometimes impede the movement of the apex. The face of the braces that came in contact with the airbeam were lined with high-density polyethylene (HDPE) to reduce friction between the airbeam and the brace.
Figure 8: HRAB Weight Hanger Test Setup

During testing it was found that without out-of-plane bracing, the airbeam would collapse or move out of plane prior to inflating. This action is at least partially prevented in the TEMPER shelter through the use of the tent fabric to tie the arches together and guy lines. Folds in the airbeam, see Figure 9, were introduced at the brace points to ensure that 1) the apex of the airbeam started at the midspan, and 2) that the airbeam would rise and not bend out of plane. Prior to every test, these folds would be induced symmetrically about the arch apex and the airbeam would be aligned as shown in Figure 11.

Figure 9: Airbeam Folds
It was observed that these fold locations were natural hinges at low inflation pressures (less than 35 kPa) and contributed to the airbeam’s tendency to bend out-of-plane. This type of failure would start from a very low inflation pressure as the airbeam starts to move from the internal bladder filling. This shifting would sometimes cause the beam to move laterally and become caught on the wooden braces, at which point the test was halted, the airbeam deflated, and the test restarted. An example of this behavior is shown in Figure 11. If the airbeam avoided this behavior, the arch would slowly rise until the apex was loaded.
Figure 11: Airbeam Bending Out-of-Plane

A load cell was placed in line with the cable and was attached directly under the airbeam’s apex with a lifting strap and shackle (Figure 12) to measure the lifting force exerted by the airbeam. The lifting load was distributed through padding consisting of two 152 mm wide lifting straps placed side by side along the length of the arch to prevent hinging in the beam. This padding was placed on top of the arch apex and held in place with duct tape. The total weight of the padding and rigging connecting the load cell to the airbeam was 150.7 N.
3.2.2. Results

The airbeam used during this test (serial number M11870-8) was able to lift a maximum applied load of 600 N at a pressure of 148 kPa. This 600 N weight includes the weight of the hanger, and the steel plates. When the airbeam suddenly lifted the weight, there were large dynamic effects, observed in the inline load cell data as a result of bouncing of the hanger (Figure 13). This bouncing is driven by the sudden stop of the airbeam rise while the hanger has upward momentum, and the subsequent drop of the hanger and reloading of the airbeam. It can also be seen that once the airbeam erects, there is a decrease in pressure as the internal volume rapidly increases which leads to a decrease in lifting load. The airbeam behaves as a damped oscillator during this process. In its fully erect position, the load cell read 704 N (Figure 14), which included the hanger, applied weights, and self-weight of some components of the pulley system. During this test the position of the arch apex was measured using the string pot shown in Figure 8. An example of the measurements gathered from each test is shown in Figure 15. The movement prior to the initial rise of the airbeam is the in-plane, X direction motion as the airbeam is moving during low internal pressures. Once the lifting cable is taut the change in position of
the airbeam apex is constant until the airbeam lifts the hanger. This straight-line distance was measured from the midspan of the airbeam to the underside of the arch apex shown in Figure 8.

**Figure 13: Dynamic Inflation Test 600 N #5, Load vs. Pressure**

**Figure 14: Dynamic Inflation Test 600 N #5, Load vs. Pressure 138 kPa to 150 kPa**
Figure 15: Dynamic Inflation Test 600 N #5, Apex Rise vs. Pressure

A summary plot of the five different applied weights showing the load at which the airbeam lifted the applied weight versus its internal pressure is shown in Figure 16. The applied weight to the airbeam includes the hanger weight and the added steel plate weight. In theory, the airbeam would lift the applied load when the lift force of the airbeam reached the weight of the applied load. However even when accounting for the rigging weight of the pulley system (84 N) or the weight of the padding and load cell (150 N), the load at which the airbeam lifted the applied load did not equal the weight of the hanger and plates (Table 1). Therefore, it is clear that there is inherent friction in the pulley system attached to the hanger causing the lifting values to be higher than the total weight of the applied load.

The results show a linear increase with initial lifting load and pressure. Once the airbeam produced enough lifting force to overcome the load and friction in the system, the airbeam rose at an increasing rate until it reached its full height. During this test, the airbeam took the shape prior to lifting shown in the upper left of Figure 16 (for a larger image see Figure 17) during every test except one as noted on the figure. The large difference in inflation pressure for the unique geometry shown in the lower right of Figure 16 versus the geometry shown in the upper left with a similar lifting force (~720 to 750 N)
indicates that the required inflation pressure is sensitive to the initial geometry of the structure. The results show a linear increase in the magnitude of weight that was lifted with airbeam pressure ignoring the one outlier test.

![Figure 16: Airbeam Lift Load vs Airbeam Internal Pressure](image-url)

**Table 1: Mean and Coefficient of Variation of Lift Values**

<table>
<thead>
<tr>
<th>Weight of Hanger and Plates (N)</th>
<th>Mean Lift Load (N)</th>
<th>CoV of Lift Load (%)</th>
<th>Mean Lift Pressure (kPa)</th>
<th>CoV of Lift Pressure (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>378</td>
<td>607</td>
<td>2.65</td>
<td>116</td>
<td>1.47</td>
</tr>
<tr>
<td>421</td>
<td>670</td>
<td>1.82</td>
<td>124</td>
<td>1.64</td>
</tr>
<tr>
<td>489</td>
<td>745</td>
<td>1.42</td>
<td>133</td>
<td>1.46</td>
</tr>
<tr>
<td>532</td>
<td>808</td>
<td>1.56</td>
<td>139</td>
<td>0.87</td>
</tr>
<tr>
<td>600</td>
<td>903</td>
<td>0.92</td>
<td>148</td>
<td>0.42</td>
</tr>
</tbody>
</table>

*Does not include the outlier shown above in Figure 16*
This test showed that the airbeam was able to lift 600 N plus the weight of the padding and rigging weight from a deflated position at about 148 kPa. It was also found that the arch inherently tends to bend out of plane prior to inflating. This specific arch assumed the same shape during every test except for one. This shape cannot be assumed to be typical for every arch as post testing inspection found that the coating on the outside of the airbeam was more malleable than the stiffer coating on the portions of the airbeam that did not have a hinge during inflation implying that an arch’s outer coating condition effects the lifting shape of the arch.

### 3.3. Static Inflation Tests

To further the understanding of the feasibility of using an airbeam to drive an erection mechanism, two sets of tests were conducted with a fixed apex height less than the final erected apex height. One set of tests used pinned connections described in Section 3.2, and the other used roller boundary conditions described later. The goal of these tests was to define the vertical force produced by the arch during
inflation in the absence of dynamic effects and the horizontal force required to restrain the arch bases during the lift as a function of inflation pressure.

3.3.1. Pinned Test Set-Up

This arch geometry was the same as the one tested in Section 3.2, with an apex rise of 1828 mm. Two nominally identical airbeams were tested: M11162-3 was tested three times through the full rise and M11870-8 was tested once with pinned connections and once with roller connections. The configuration of the airbeam with the pinned ends was the same as the one presented in Section 3.2, although without the weight hanger. The arch apex was restrained with a winch which was secured to the floor with a steel rod and angle. A sheet of neoprene was placed atop the winch with a hole for the cable to pass through, shown in Figure 18, to reduce wear on the airbeam and to reduce the chance of the airbeam’s motion being impeded by the obstacle. During alignment of the airbeam and introduction of the folds shown in Figure 10, the apex was placed atop of the winch. During inflation the sheet was moved and reset before the next test.

Figure 18: Static Inflation Restraining Device

For the pinned boundary condition tests, the winch cable was attached to the airbeam through lifting straps with padding to help distribute the load at the midpoint of the arch as described in Section 3.2.
The same load cell and connection mechanism shown in Figure 12 were used to measure the restraining force needed to prevent the airbeam from rising further. The range of heights that were tested started at the same lower bound as the one described in Section 3.2 of 749 mm. From there the height was increased in 76 mm increments.

3.3.2. Pinned Results

During testing, the same out of plane bending was observed as mentioned earlier in Section 3.2. The airbeam would move in the X direction of the beam until the arch rose to its set height. From there the load in the winch cable increased at a constant rate shown below in Figure 19. A large shock load is shown in the data when the arch rises to its set height. This shock load generally increases with the increase of the restrained height. During testing the arches would rise to the height set by the restraining cable before the internal pressure reached 70 kPa.

![Graph](image1)

**Figure 19: Static Inflation Test of M11870-8**

The maximum restraining force, shown below in Figure 20, was recorded at an internal pressure of 137 kPa. The results indicate inconsistency in the maximum restraining force, which can be attributed to
different geometric shapes the airbeam took during inflation. Near the full rise of the airbeam in round 2 and round 3 for M11162-3, the maximum restraining force decreased. This may have been due to the airbeam being too close to its fully inflated rise to generate an accurate load reading.

**Figure 20: Airbeam Restraining Force vs. Rise at 137 kPa**

The observed geometric airbeam shapes during testing indicated that the airbeams do not rise symmetrically. The typical final shapes of M11162-3 and M11870-8 were approximately mirror images of each other seen in Figure 21 and Figure 22 respectively, and the slope of the airbeam shape near midspan was opposite between the two airbeams (see red lines on Figure 21 and Figure 22). During a test of M11870-8 at a restricted height of 1155 mm, a triangular shape was achieved and maintained through the inflation to 137 kPa (Figure 23). This shape caused a sharp increase of load lifting capacity that can be seen at a lift height of about 930 mm in Figure 20. It is unclear what caused this shape to be taken and maintained.
3.3.3. Roller Test Set-Up

In the second test, M11870-8 was supported by roller connections with a tension tie between supports to prevent spreading of the arch bases shown in Figure 24. The roller connection was made by placing a
steel rod atop of two HDPE rails. For these tests, a winch cable was attached to the underside of the arch apex and set to a fixed height measured vertically from the centerline of the airbeam to the tension tie beneath it. The connection method to the airbeam apex was similar to that described in section 3.2, except a 950 mm wide canvass sling was used to distribute the restraining force instead of the padding. In these tests both ends were clamped with wood 4x4s, see Figure 25, with equal lengths of excess airbeam folded up on to the arch. Gaps in the wooden clamps were set to 1 inch on both ends to allow the full airbeam to inflate. A rope was attached to each of the wooden clamps and set to shorten the span of the arch by 381mm from its pinned span of 5588 mm. Concrete blocks were placed on either side of the footing to prevent the arch bases from moving out of plane, but did not interfere with the sliding of the arch base. Two load cells were used in this test, one to measure the restraining force of the tension tie shown in Figure 24, and the other to measure the restraining force of the arch apex.
The airbeam was inflated from a deflated position while recording internal pressure and restraining force in the winch cable. The heights of the apex started at 749 mm and increased by 152 mm up to the full rise. During testing out-of-plane bending was observed in some tests, which can be attributed to twists or misalignment in the deflated arch fabric. Six triangular braces were placed on both sides of the airbeam to prevent the airbeam from deflecting excessively out-of-plane during inflation. This bracing type is the same as described in section 3.2.
3.3.4. Roller Results

For these tests the maximum inflation pressure was increased to 206 kPa to gather more data on the airbeam outside of the target pressure range. As expected, the restraining force in the winch cable and the tension force in the tension tie increased linearly with pressure. The horizontal force in the tension tie as a function of inflation pressure is shown in Figure 26, for discrete rises up to 1809 mm. In this figure there is a noticeable jump in tension force between the 1504 mm and 1352 mm rises. It is at this rise where the airbeam no longer has any hinges along its arch. This figure shows that an arch with no hinges exerts more force horizontally on the bases if the arch is not allowed to take its full geometric shape. Figure 27 shows the force required to restrain the airbeam at each rise. As seen in the pinned tests, an increase in rise yields an increase in restraining force shown.

![Figure 26: Static Inflation Test: Roller, Tension Tie Force vs Airbeam Pressure](image_url)
Summary plots of the tests are shown in Figure 28 and Figure 29 for the restraining force of the airbeam and force in the tension tie at 137 kPa and 206 kPa respectively. Table 2 presents the values used in Figure 28 and Figure 29 with their mean and COV for the maximum tension tie values.
Figure 28: Static Inflation Test: Roller, Tension Tie and Restraining Force at 137 kPa

Table 2: Mean and Coefficient of Variation of Static Inflation Test: Roller

<table>
<thead>
<tr>
<th>Rise (mm)</th>
<th>Cable Tension</th>
<th>137 kPa</th>
<th>206 kPa</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Airbeam N</td>
<td>Tension Tie N</td>
<td>Airbeam N</td>
</tr>
<tr>
<td>742.9</td>
<td>483.5</td>
<td>330.5</td>
<td>859.7</td>
</tr>
<tr>
<td>742.9</td>
<td>488.9</td>
<td>313.2</td>
<td>851.8</td>
</tr>
<tr>
<td>895.3</td>
<td>551.7</td>
<td>295.7</td>
<td>962.2</td>
</tr>
<tr>
<td>1048</td>
<td>608.1</td>
<td>289.3</td>
<td>1062</td>
</tr>
<tr>
<td>1200</td>
<td>711.9</td>
<td>263.4</td>
<td>1251</td>
</tr>
<tr>
<td>1353</td>
<td>880.7</td>
<td>330.5</td>
<td>1512</td>
</tr>
<tr>
<td>1505</td>
<td>1074</td>
<td>473.8</td>
<td>1837</td>
</tr>
<tr>
<td>1657</td>
<td>1281</td>
<td>505.7</td>
<td>2114</td>
</tr>
<tr>
<td>1810</td>
<td>1373</td>
<td>510.2</td>
<td>2238</td>
</tr>
<tr>
<td>Mean</td>
<td>-</td>
<td>368</td>
<td>-</td>
</tr>
<tr>
<td>COV</td>
<td>-</td>
<td>27%</td>
<td>-</td>
</tr>
</tbody>
</table>
3.4. Summary

From the results gathered from the Dynamic and Static inflation tests in Sections 3.2 and 3.3 respectively, inflated arches were found to be able to lift loads much larger than their own self weight as long as the arch was allowed to rise up to a portion of it full rise. Once the airbeam has begun exerting a force on an applied load or restraining device, that force increases linearly with an increase in pressure. Additionally it was found that the each arch has a shape that it prefers to take during inflation, and this shape has a large effect on its lifting capacity. Once the arch has inflated, it acts as a damped oscillator if lifting a free weight and the slight decrease in pressure causes a slight decrease in load capacity. It was observed that during the inflation process the airbeam has a high tendency to move and bend out of plane, or inflate horizontally instead of vertically if not properly braced. Finally, it was observed that the horizontal force exerted by the arch bases generally remains constant with the increase of a restrained arch height.
CHAPTER 4

DESIGN OF FULL-SCALE MOCK-UP AND TESTING TO PROVE CONCEPT

4.1. Development of HRAB Concept

The testing detailed in Chapter 3 indicates that while an airbeam can generate significant lifting forces during inflation, it cannot be relied upon to consistently rise in a symmetric manner. Further, the dynamic tests showed that sudden erection can lead to significant inertial loads. Therefore, the simple HRAB concept structure shown in Figure 30 and Figure 31 is proposed here that uses the airbeam to lift side walls with a straightforward mechanism that does not require significant forces applied at the airbeam apex.

![HRAB Concept Cross-Section](image)

**Figure 30: HRAB Concept Cross-Section**
The HRAB concept combines the benefits of rigid and soft-wall shelters. The rigid walls of the shelter can be lightweight, sandwich panel thermoplastic composites that will include simple locking mechanisms to rigidize the middle hinge once the shelter has been erected. The roof of the shelter consists of airbeam arches that spring from the ground and provide a light yet strong structure. These two traits are expected to reduce the areal weight of the shelter relative to existing rigid-walled shelters. Once the shelter is erected, blast or ballistic resistant panels could then be hung on the thermoplastic sandwich panels, or incorporated directly in the wall panels as needed for a specific application. In contrast with current airbeam tent technology, the additional weight of this protection will not be borne by the airbeams or require separate supporting structures (Horak, 2009). The HRAB also increases the usable area of the shelter. MIL-STD-1472G defines the 95th percentile male as 73.9in, the TEMPER shelter. The airbeams are expected to drive erection of the structure, including lifting the walls. In this concept, the airbeams are supported on the ground, and the folding walls are erected as the airbeam pushes directly on them during inflation. The fabric roof serves as a tension member to tie the airbeam to the top of the walls. The roof, diagonal compression struts, and guy lines stabilize the walls during erection and post erection. Finally, the concept is modular, allowing larger structures to be erected by connecting
individual units. Multiple segments can be shipped in an ISO 1C container, allowing construction of shelters of different size using a single shipping unit. The erection process, shown in Figure 32, starts with segments being placed on the ground and connected to each other. In this step the airbeam is folded around the closed hinged wall panel. Then the inflation of the airbeams begins, causing the airbeam to lift the folding wall panels. Once the airbeam reaches the required pressure to erect the walls, the structure assumes its final shape.

![Figure 32: Hybrid Rigid Airbeam Concept Deployment Process (guy lines not shown for clarity)](image)

4.2. HRAB Concept tests

To verify this concepts feasibility, an experiment was created to test the airbeams ability to lift itself and the folding rigid walls in the concept. A model of the structure was created in the Advance Structure’s and Composites Center’s laboratory utilizing a TEMPER airbeam and wooden folding walls.

4.2.1. Test Set-Up

To test the airbeam’s ability to drive erection, the mock-up shown in Figure 33 and Figure 34 was constructed to mimic the mechanism as shown in Figure 30. The HRAB wall panels and struts were made of wood 2x4s. Figure 34 shows labeled components on the walls that will be referenced in this section. The west wall refers to the wall in the left of the figures, and east refers to the wall on the right of the figures.
The airbeam in the scale model was the same used in section 3.2 and section 3.3, a full 6096 mm major diameter semi-circular airbeam (serial number: M11870-8) attached to the ground with a pin connection near the base of the wall panel (Figure 35). The connection was achieved by removing the plastic rod that is inside of the aluminum clamp installed on the end of the airbeam from the manufacturer. This rod was replaced with a 16 mm diameter steel shaft that went through the aluminum clamp into two bearings on either side of the aluminum clamp, secured to the floor. A
wooden floor of oriented strand board was placed on top of the concrete lab floor at each end of the structure as well as under the airbeam. This floor was placed to allow the movement and placement of various test components (string pots, supports, wall panel and airbeam anchors) with relative ease.

![Figure 35: HRAB Pin Connection](image)

Two guy lines were attached to the top of the wall panel and were secured to the ground 1603 mm and 1625 mm from the base of the west and east walls respectively. The roof fabric was mimicked with one continuous piece of rope. The rope was attached to the top of one wall panel and connected to the tangent point on the airbeam, over the airbeam apex, attached at the second airbeam tangent point, and down to the other wall panel. The compression strut was attached at the top of the wall panel and airbeam through a pinned connection, and oriented perpendicular to the airbeam as seen in Figure 36. The strut connection to the wall panel was made with two bearings which held a shaft. Eye bolts in the strut and the rope were both attached to this shaft. The strut was attached to the airbeam with an NRS strap which was held in place with duct tape. Weights of the struts are shown below in Table 3. During the erection process, the NRS strap would move slightly from its original position. Before the next test, the NRS strap would be repositioned. The theoretical length of the strut is 653 mm and 674 mm for the west and east struts respectively.
The walls were made of 2x4 studs and constructed using steel connecting components and screws. The hinge in the wall was created by creating two halves of the wall connected in the middle with four door hinges. These hinges were also used on the bottom of the lower panel to connect it to the wooden floor and maintain the hinged connection. The hinges were lubricated with lightweight oil to decrease the resistance of the hinge. The wooden pad beneath the whole structure was secured to the lab floor using steel rods and plates. To prevent over-extension of the panels once they reach a vertical state, cantilevered portions of 2x4 were attached to the upper wall panel. Once the wall panel was vertical, the cantilevers would be prevented from over-extending by contacting the vertical lower wall shown in Figure 37.

<table>
<thead>
<tr>
<th>Strut</th>
<th>Total Length (mm)</th>
<th>Weight, No Load Cell (N)</th>
<th>Weight, with Load Cell (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>East</td>
<td>635</td>
<td>17.2</td>
<td>18.1</td>
</tr>
<tr>
<td>West</td>
<td>654</td>
<td>17.9</td>
<td>18.2</td>
</tr>
</tbody>
</table>
Wall panel weights and heights are shown below in Table 4 with an estimated weight for sandwich composite panel which would be used in the final structure. More specifics of this sandwich composite panel can be found in section 5.3.1.2. Figure 38 shows an as-built dimensioned drawing of the whole model. The wall panel height differences and shortage are attributed to manufacturing error. The height of the wall hinge was measured from the base of the wall to the center of the door hinge connecting the upper and lower panels. The upper panel weight includes the bearings and shaft which the strut and roof rope attach to at the top of the panel and the middle hinges. The lower panel weight includes the lower hinges and a latch to lock the wall panels together after they become vertical.
### Table 4: HRAB Wall Panel Properties

<table>
<thead>
<tr>
<th>Wall</th>
<th>Total Height (mm)</th>
<th>Height of Hinge (mm)</th>
<th>Lower Panel Weight (N)</th>
<th>Upper Panel Weight (N)</th>
<th>Equivalent Composite Panel Weight (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>East</td>
<td>2157</td>
<td>1098</td>
<td>173</td>
<td>238</td>
<td>90.3</td>
</tr>
<tr>
<td>West</td>
<td>2111</td>
<td>1057</td>
<td>167</td>
<td>253</td>
<td></td>
</tr>
</tbody>
</table>

#### 4.2.2. Instrumentation and Test Protocol

The model was instrumented with a load cells, inclinometers, and string pots shown below in Figure 39. A string pot was placed beneath the airbeam apex and fixed to the lab floor to track the rise of the apex. String pots were also placed to track movement of the top corner of the walls by locating one string pot on the floor beneath each wall to track its vertical rise. A second string pot was placed on a brace 1146 mm and 898 mm from the wall base horizontally and 2064 mm and 2082 mm up vertically for the west and east walls respectively. Video was also taken during each test to visually track the motion of the airbeam and walls during the erection sequence as well. The inclinometers were attached to the wall panels with screws and were oriented to only read angles in the plane of Figure 39. The load cells were located in-line with the components they were measuring. For the roof element, the rope was separated into two segments and each segment was tied to one end of the load cell. This same method was used for the load cells located in the guy lines. The strut load cells were attached to the bearing on the top of the wall panel on each side with an eye bolt and secured to the strut with a threaded rod on the other. During testing, over time the knots connecting the roof load cell to the rope would slip, lengthening the roof rope. This rope would also slip off of the top of the airbeam occasionally and would need to be re-centered. All load cells were verified in tension only and their compression (negative) readings should not be taken as accurate.
**Figure 38: HRAB Experimental Model As-Built Dimensions**

**Figure 39: HRAB Experimental Model Instrumentation Diagram**
The structure was tested with initial wall panel openings of 0, 4, 8, and 16 degrees (Figure 40). These angles were chosen to, in theory, reduce the required internal pressure to lift the wall panels. The angle was measured from the horizontal through the hinges to each wall panel as shown in Figure 41. These angles were created with the wooden supports seen in Figure 40. Supports for the lower panel were attached to the floor while the supports for the upper panel were attached to the lower panel. The wall angle was measured using the installed inclinometers on each wall panel.

**Figure 40:** HRAB Experimental Model 16° and 0° wall panel openings

The procedure for test was as follows:

1. Prop wall panels to initial desired wall panel opening
2. Lay airbeam flat with apex centered on apex string pot (Figure 42).

**Figure 41:** HRAB Folded wall starting angle dimensions
Figure 42: HRAB Model Deflated

3. Turn on the 275 kPa air supply
4. Begin recording data
5. Turn on air to airbeam
6. Stop air to airbeam once structure is erected.

4.2.3. HRAB Concept Test Results

Two wall opening scenarios occurred during inflation. In the first scenario, one wall panel would open first between 65 kPa and 104 kPa as shown in Figure 43. The internal pressure would then drop due to the slight increase in the airbeam’s internal volume, and the second wall panel would open once the pressure rose again to between 61 kPa and 102 kPa. In a few instances during an asymmetrical wall panel opening, the closed wall panel rose almost immediately after the first side opened due to dynamic effects caused by the rapid opening of the first panel. This opening is labeled as the second scenario, and would often happen when both wall panels were partially opened as shown in Figure 44.

Figure 43: HRAB Experimental Model Asymmetric Wall Panel Opening
Figure 44: HRAB Experimental Model Symmetric Wall Panel Opening

In the beginning of a test the airbeam would behave very similar to its behavior observed in Chapter 3. There would be shifting toward the west or east wall panel until the apex was restrained by the roof rope on either side. During this time the airbeam could also begin to bend out of plane. As the internal pressure increased the airbeam would begin loading the components in the structure as shown by Figure 48. The wall panels would also start to slowly rotate about the hinge connecting the lower wall panel to the floor. The airbeam would then pop through from its beginning state into time 1 as shown in Figure 45
From here the airbeam would continue loading the components until the force in the roof is great enough to pull the top of a wall panel inward, causing the structure to enter time 2 as shown in Figure 46. The structure would again start loading components until the walls begin to rotate allowing the airbeam legs to push against the middle hinge of the walls causing either both walls, or just one wall to open as shown in Figure 47.

Figure 46: 8° Initial Opening, Time 2

Figure 47: 8° Initial Opening, Time 3

Annotated plots of forces in the east and west struts, the roof load cell located on the west side, and on one of the guy lines located on the east wall were generated for all of the tests that were run. Figure 48 and Figure 49 show the internal forces measured and inclinometer measurements versus pressure for
the first test with an initial opening of 8 degrees. In both annotated figures, point one indicates the pop through of the airbeam from its beginning state (Figure 45), point two shows the west upper panel opening as shown in (Figure 46), and point three indicates the east wall panel opening fully opening as shown in (Figure 47). Large dynamic forces were observed in the data during times where there is large motion of the airbeam from one state to another.

Figure 48: 8°Initial Opening, Internal Forces in HRAB Components vs. Internal Pressure
In the cases of scenario two where both wall panels opening simultaneously, the large jumps in the data would occur at the same time. This simultaneous erection would often require that both panels be slightly open as shown in Figure 50. Figure 51 shows the associated component force versus pressure for the 16° initial wall panel opening #2 test shown in Figure 50. Time 1 shows the initial dynamic effects of both wall’s top panel opening (Figure 50) and time 2 shows the subsequent dynamic effect from both wall panels fully opening.
The difference in initial wall panel opening did not effect the internal pressure required to open the wall panels as shown in Figure 52. This wall panel opening pressure represents the pressure at which the wall panel became fully open. There were two outliers during the 4 degree tests, one of which led to large enough dynamic forces to break the lower wall panel leading to its replacement. After the wall panel
was repaired, the results matched the other data near the 89 kPa inflation pressure. During testing, the east wall erected fully first 75% of the time in erection scenarios one and two. This value does not include the outlier tests shown in Figure 52 at a four degree initial wall panel opening angle where the west wall did not open until nearly 130 kPa. The only tests where the west panel erected first were four of the 16 degree tests. For reference the airbeam arches tested in Chapter 3 all reached their final inflate shape at pressures less than 70 kPa while the wall opening pressures in Table 5 range from 61 kPa to 104 kPa. This cannot be taken as a direct comparison, however, since the airbeams are loaded in different configurations.

![Figure 52: HRAB Concept Wall Panel Opening Pressure Results](image)

**Table 5: HRAB Wall Panel Pressures**

<table>
<thead>
<tr>
<th>Wall Starting Angle</th>
<th>No. of Tests</th>
<th>East Wall Mean Opening Pressure</th>
<th>CoV of Opening Pressure</th>
<th>West Wall Mean Opening Pressure</th>
<th>CoV of Opening Pressure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Degrees</td>
<td>-</td>
<td>kPa</td>
<td>%</td>
<td>kPa</td>
<td>%</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>85.6</td>
<td>19.4</td>
<td>80.6</td>
<td>8.4</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>90.1</td>
<td>3.1</td>
<td>90.8</td>
<td>2.2</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>91.2</td>
<td>6.1</td>
<td>92.4</td>
<td>6.1</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>90.1</td>
<td>2.5</td>
<td>94.9</td>
<td>6.5</td>
</tr>
</tbody>
</table>
4.3. Summary

The tests reported in this chapter have shown that the airbeam is able to erect a folding wall structure from a deflated position. During erection, the wall panels erect rapidly, causing dynamic effects and impact forces in the wall panels. Further work will be needed to quantify the forces exerted in the wall panels during erection from the stops needed to prevent their over-rotation and from the pressure of the airbeam pinching the middle wall hinge prior to opening of the panels. While there was a wide distribution of the inflation pressure required for erection at an initial wall opening of 16 degrees, the structure successfully self-erected at all initial wall openings between 0 and 16 degrees in a pressure range of 61 kPa to 104 kPa. The structure also favored erecting the east wall before the west wall during 75% of the conducted tests. While this is likely an artifact of inherent lack of symmetry in the arch and/or HRAB mock-up, unsymmetric erection is expected to be the norm for any HRAB structure.

It is also important to quantify the time required to erect the HRAB. An initial estimate of a ‘best case’ scenario can be made based on assembly of the lab mock-up tested here. Erecting one 2438 mm long segment is estimated to require two people, four minutes to anchor wall segments and guy lines, and six minutes to inflate the airbeam to 137 kPa with a compressor. Therefore, for a four-segment structure, a reasonable estimate is 16 minutes for setup and 24 minutes to inflate the airbeams assuming sequential erection by two people. The inclusion of roof fabric over the structure may cause difficulties in laying out the deflated shelter which was not accounted for in the mock up. Additional time will be required for post-erection steps, including attachment of adjacent segments. Further, inflation time will depend on air compressor capacity.
CHAPTER 5

DETAILED ANALYSIS OF HYBRID AIRBEAM SHELTER

This chapter examines the load-carrying capacity of an HRAB structure using nonlinear finite-element analysis. The capacity of the HRAB structure is assessed through comparison of an equivalent TEMPER tent airbeam having the same arch geometry and supporting the same tributary width of structure.

5.1. Overview of Analysis Methods

The gravity load capacity of the HRAB structure was estimated using PressArchAnalysis (Davids and Clapp, 2013), a finite element package that analyzes inflated fabric beams and arches utilizing specially formulated three-noded Timoshenko beam elements to capture the behavior of woven and braided arches and beams. In an inflated fabric beam, inflation pressure pre-tensions the fabric, which gives a beam or arch the ability to carry external loads that cause compressive stress. In a braided, strapped arch used in a TEMPER tent or HRAB structure, the inflation pressure primarily pre-tensions external reinforcing straps that are bonded to the braided tube shown in the cross-section of a braided tube in Figure 53.

![Figure 53: Braided and Strapped Airbeam Cross Section](image)

Reinforcing Strap

Airbeam Braided Fabric
Different braid angles of the fabric tube give it different axial and shear stiffness values (Brayley, 2011). However, the primary source of flexural and axial stiffness and bending capacity are the external reinforcing straps. Brayley (2011) summarizes the work of others which shows that the level of prestress in the straps produced by inflation is a function of the braid angle as defined in Equation 1. Here, $P'$ is the pressure resultant in the straps generated by the total pressure resultant $P$ in the airbeam tube, where $\theta$ is the braid angle of the airbeam fabric measured from the longitudinal axis of the beam.

$$
\frac{P'}{P} = 1 - 2\cot^2(\beta) \quad \text{Equation 1}
$$

Using mechanics of materials concepts, the stress in the straps can be found by summing the stresses exerted in the straps due to the pressure resultant and the stress exerted by flexure from applied moments shown in Equation 2. In Equation 2, $\sigma_s$ is the stress in the straps, $M$ is the applied moment, $y$ is the distance from the neutral axis to the extreme fiber of a strap, $I$ is the cross-sectional moment of inertia of all reinforcing straps about a horizontal line drawn through the centroid of the cross-section in Figure 53, and $A_s$ is the cross-sectional area of one strap.

$$
\sigma_s = \frac{\pm My}{I} + \frac{P'}{\sum A_s} \quad \text{Equation 2}
$$

As an airbeam is loaded to produce bending, the net tensile force in the straps on the compression face is reduced. The load at which the net tensile force in the straps becomes zero is called the wrinkling load. For the simple case of pure bending, setting $\sigma_s = 0$ and solving Equation 2 for $M$ yields Equation 3 the wrinkling moment $M_w$ of the inflated member.

$$
M_w = \frac{P' I}{y \sum A_s} \quad \text{Equation 3}
$$

An arch will generally experience combined compression and bending. An applied compression force $F$ will reduce $M_w$ below the value predicted by Equation 3. This reduction in $M_w$ can be determined by calculating the reduction in strap prestress due to $F$. 50
Testing by Brayley et al. (2012) showed that a typical straight, braided strapped beam loaded in bending could carry about 15% additional load beyond wrinkling. After the onset of wrinkling, the bending response of a beam or arch becomes nonlinear and the volume of the section begins to reduce as shown in Figure 54, which illustrates pinching of the upper half of the section from the applied load of the green tarp.

![Airbeam Wrinkling](image)

**Figure 54: Airbeam Wrinkling**

The three-noded airbeam element in PressArchAnalysis was originally derived as a shear-deformable, flexure-only element that captured the materially non-linear effects of the inflated beam as a woven fabric wrinkled on the compression edge (Davids, 2007). Wrinkling is simulated as an effective material nonlinearity using moment-curvature analysis. Work done by pressure resulting from cross-sectional volume reduction due to shear and bending deformations is also simulated. Davids and Zhang (2008) then extended the element to include P-Δ effects from axial loading. Large deformations were included to capture geometric nonlinearities (Davids, 2009) necessary for simulating the in-plane response of arches. Most recently, Brayley et al. (2012) added the ability to include braided and strapped arches. Brayley et al (2012) developed expressions to determine the prestress force carried by the straps, the stress in the straps due to flexure and internal pressure loads, and the wrinkling moment in terms of the prestress force (Brayley et al., 2012).
PressArchAnalysis is able to model an arch with a variety of boundary conditions. The solver includes two-noded, axial, tension-only elements which are used for a tension tie to connect the bases of the roller supported arch together as well as for guy lines. The guy lines attach to the sides of the arch models to restrict sidesway motion of the arch. To run an analysis and view results, the user can either use the pre-compiled executable with a graphical user interface (GUI) which offers definition of arch properties, loading parameters, as well as tension only element locations and properties. Or, for more customization and batch analyses, the user must run the solver from within MATLAB using a driver function.

In the first analysis step, the self-weight of the airbeam structure is applied and the specified pretension in all tension-only elements is determined using a Newton solver. Once the Newton solver has reached a solution within an acceptable error, additional loads are applied incrementally using a user-defined load step. A Newton solver is employed for this load stepping until non-convergence, after which a cylindrical arc-length solver uses the load as a variable allowing for post-buckling response to be simulated where applied load decreases with increasing displacements (Davids, 2009).
5.2. Extension of Existing Non-Linear FE Techniques in PAA to Simulate the HRAB Structure

To use PressArchAnalysis in modelling the HRAB structure, new elements were added to the software. In addition to the currently available airbeam and tension only elements, wall, roof, and strut elements were added with some wall and strut elements having internal hinges. Figure 55 shows the different element types in the HRAB model. The arrows on the strut and wall elements show the orientation of the elements with an internal hinge at the end node.

<table>
<thead>
<tr>
<th>Element</th>
<th>Color</th>
<th>Hinged Element Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airbeam</td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td>Guy/Tension Tie</td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td>Wall</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Strut</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roof</td>
<td></td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Figure 55: HRAB Structure Element Locations**

The wall and strut elements are based on the current three-noded Timoshenko beam elements that are used for the airbeam elements but have constant flexural rigidity regardless of applied load, since they are expected to behave linearly elastically. The current three-noded element used in PressArchAnalysis is shown below in Figure 56. The stiffness matrix (derived later) has the degrees of freedom ordered by the node numbers shown in Figure 56.
Figure 56: Three Noded Quadratic Beam Element

To model the hinge connections in the wall and strut elements correctly, the basic three-noded Timoshenko beam element was modified to include an internal hinge at one end. The stiffness matrix of the original element without a hinge is shown below in Equation 4. The degrees of freedom in the rows and columns are ordered $[u_1 \ v_1 \ \theta_1 \ u_2 \ v_2 \ \theta_2 \ u_3 \ v_3 \ \theta_3]$. The rotation $\theta_3$ does not apply to this element, since end rotations are assumed to vary linearly along the element, but is included as a placeholder so the element has the same number of degrees of freedom per node.

$$
K = \begin{bmatrix}
\frac{7EA}{3L} & 0 & 0 & \frac{EA}{3L} & 0 & 0 & -\frac{8EA}{3L} & 0 & 0 \\
0 & \frac{7GA}{3L} & \frac{5GA}{6} & 0 & \frac{GA}{3L} & \frac{GA}{6} & 0 & -\frac{8GA}{3L} & 0 \\
0 & \frac{5GA}{6} & \frac{EI}{L} + \frac{GAL}{3} & 0 & -\frac{GA}{6} & \frac{EI}{L} + \frac{GAL}{6} & 0 & -\frac{3GA}{2L} \\
\frac{EA}{3L} & 0 & 0 & \frac{7EA}{3L} & 0 & 0 & -\frac{8EA}{3L} & 0 & 0 \\
0 & \frac{GA}{3L} & -\frac{GA}{6} & 0 & \frac{7GA}{3L} & -\frac{5GA}{6} & 0 & -\frac{8GA}{3L} & 0 \\
0 & \frac{GA}{6} & -\frac{EI}{L} + \frac{GAL}{6} & 0 & -\frac{5GA}{6} & \frac{EI}{L} + \frac{GAL}{3} & 0 & -\frac{2GA}{3L} \\
-\frac{8EA}{3L} & 0 & 0 & -\frac{8EA}{3L} & 0 & 0 & \frac{16EA}{3L} & 0 & 0 \\
0 & -\frac{8GA}{3L} & -\frac{2GA}{3} & 0 & -\frac{8GA}{3L} & \frac{2GA}{3} & 0 & \frac{16GA}{3L} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

Equation 4

To add the internal hinge to this element at the end node, a linear constraint must be added to the element at the end node’s rotational degree of freedom. This is done by setting $M_2 = 0$ as shown below in Equation 5 where row 6 of $K$ has been multiplied by the nodal displacement vector $U$ to give $M_2$. 

$$
M_2 = 0
$$
\[ M_2 = \begin{bmatrix} 0 & \frac{GA}{6}v_1 & -\frac{5GA}{6}v_2 & 0 & \frac{2GA}{3}v_3 \end{bmatrix} \begin{bmatrix} 0 \quad \frac{E_1}{L} + \frac{GAL}{6} \quad \frac{E_1}{L} + \frac{GAL}{3} \quad 0 \quad \frac{2GA}{3}v_3 \end{bmatrix} \begin{bmatrix} \theta_1 \quad \theta_2 \quad 0 \quad 0 \end{bmatrix} = 0 \]  

Equation 5

Solving Equation 5 for \( \theta_2 \) yields Equation 6.

\[ \theta_2 = \begin{bmatrix} 0 & \frac{GA}{6}v_1 & -\frac{5GA}{6}v_2 & 0 & \frac{2GA}{3}v_3 \end{bmatrix} \begin{bmatrix} 0 \quad \frac{E_1}{L} + \frac{GAL}{6} \quad \frac{E_1}{L} + \frac{GAL}{3} \quad 0 \quad \frac{2GA}{3}v_3 \end{bmatrix} = 0 \]  

Equation 6

A constraint matrix, \( R \), can then be generated from the coefficients of the displacement variables in Equation 6. The full constraint matrix with the \( \theta_2 \) column removed is shown in Equation 7.

\[ R = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \]  

Equation 7

The constraint matrix \( R \) is then applied to the stiffness matrix \( K \) through Equation 8. This gives a stiffness matrix, \( K^* \), where the \( \theta_2 \) row and column have been removed.

\[ K^* = R^T K R \]  

Equation 8

Mapping \( K^* \) from its 8x8 state to the full stiffness matrix yields the stiffness matrix of the three-noded Timoshenko beam element with an internal end at the end node shown in Equation 9 on page 57. In this stiffness matrix, the diagonal entry corresponding to \( \theta_2 \) was set to 1 to eliminate the possibility of a singular matrix being created while solving the model in PressArchAnalysis. This is also the same reason there is a 1 in diagonal corresponding to \( \theta_3 \).

This new element was applied at both ends of the struts as well as the bottoms of the wall panels. This ensured that no moment was transferred between the wall and airbeam at the bases, the strut and wall.
at the top of the wall, and the strut and airbeam similar to the physical model in section 4.2. In the
solver, the stiffness matrix shown in Equation 9 was not used, rather Equation 8 was used to calculate
the stiffness matrix to eliminate errors in the hard-wiring of the stiffness matrix. Roof elements were
chosen to be tension only elements to replicate the properties of a fabric, which cannot carry
compressive stress. All tension only elements in the solver are linear elastic and are made tension only
by being assigned a near-zero axial rigidity (EA) if they experience a compressive strain. Before adding
the wall, strut and roof elements to the PressArchAnalysis software, simple models were run in
PressArchAnalysis to verify the elements were working correctly as described next.
\[
K = \begin{bmatrix}
\frac{7EA}{3L} & 0 & 0 & EA & 0 & 0 & -8EA & 0 & 0 \\
0 & \frac{GA(9GAL^2 + 28EI)}{4L(GAL^2 + 3EI)} & \frac{3GA(GAL^2 + 4EI)}{4(GAL^2 + 3EI)} & 0 & \frac{GA(3GAL^2 + 4EI)}{4(GAL^2 + 3EI)} & 0 & 0 & -\frac{GA(3GAL^2 + 8EI)}{L(GAL^2 + 3EI)} & 0 \\
0 & \frac{3GA(GAL^2 + 4EI)}{4(GAL^2 + 3EI)} & \frac{GAL(GAL^2 + 12EI)}{4(GAL^2 + 3EI)} & 0 & \frac{GA(12EI - GAL^2)}{L(GAL^2 + 3EI)} & 0 & 0 & -\frac{GAL^2 L^2}{GAL^2 + 3EI} & 0 \\
\frac{EA}{3L} & 0 & 0 & \frac{7EA}{3L} & 0 & 0 & -8EA & 0 & 0 \\
0 & \frac{GA(3GAL^2 + 4EI)}{4L(GAL^2 + 3EI)} & -\frac{GA(12EI - GAL^2)}{L(GAL^2 + 3EI)} & 0 & \frac{GA(GAL^2 + 28EI)}{4L(GAL^2 + 3EI)} & 0 & 0 & -\frac{GA(GAL^2 + 8EI)}{L(GAL^2 + 3EI)} & 0 \\
0 & 0 & 0 & 0 & \frac{GA(3GAL^2 + 8EI)}{L(GAL^2 + 3EI)} & -GA^2 L^2 & 0 & 0 & 0 \\
-\frac{8EA}{3L} & 0 & 0 & -\frac{8EA}{3L} & 0 & 0 & 1 & 0 & 0 \\
0 & -\frac{GA(3GAL^2 + 8EI)}{L(GAL^2 + 3EI)} & -\frac{GA^2 L^2}{GAL^2 + 3EI} & 0 & \frac{GA(GAL^2 + 8EI)}{L(GAL^2 + 3EI)} & 0 & 0 & 4GA(GAL^2 + 4EI) & 0 \\
0 & -\frac{GA^2 L^2}{GAL^2 + 3EI} & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

Equation 9
5.2.1. Element Verifications

Simple models were generated for the three noded Timoshenko beam elements to be used for the wall, strut, and roof elements. In these models large deformation, geometrically non-linear analyses were conducted and verified with equilibrium checks. First, global horizontal and vertical equilibrium of the applied loads and reactions were checked. Second, the support reactions were summed with the member end forces at the supports while incorporating the large deformation of the model. In these equilibrium checks the deformation of the structure was taken into account as shown by Figure 57, where tension and shear forces in the element acted along and perpendicular to the orientation of the deformed element.

![Figure 57: Large Deformation Equilibrium Check for Tension Only Elements](image)

The first step in the verification of the elements was to ensure that the element stiffness functions were functioning correctly by running a small deformation analysis of a simply supported beam and checking the results against small deformation beam bending equations. This was done by analyzing a simply supported beam model subjected to the point load of 100 N at the midspan as shown in Figure 58. Global equilibrium was checked using the output reactions from PAA (Equation 10 through Equation 12). The results from the analysis were then checked against small deformation beam bending equations which included the effect of shear deformation in the midspan displacement (Equation 13 and Equation 14).
Figure 58: Wall & Strut Element Verification: Simply Supported Beam, Model and Reactions

\[
\sum F_x = 0 \\
\sum F_y = 0 \\
50N + 50N - 100N = 0N \\
\sum M = 0
\]

Theoretical Moment at Midspan = Moment at Midspan from PAA

\[
\frac{PL}{4} = 76200 \text{ N} \times \text{mm} \\
76200 \text{ N} \times \text{mm} = 76200 \text{ N} \times \text{mm}
\]

Equation 13

Theoretical Deflection at Midspan = Deflection at Midspan from PAA

\[
\frac{PL^3}{48EI} + \frac{0.25PL}{GA_s} = 0.2250\text{mm} \\
0.2056\text{mm} + 0.019\text{mm} = 0.2230\text{mm} \\
0.2246\text{mm} = 0.2250\text{mm}
\]

Equation 14

The next model analyzed was a horizontal beam with pinned boundary conditions at both ends as shown in Figure 59. This model was subjected to a 200 kN vertical force at the midspan. This load produced a mid-span deflection of 315 mm, which is about 10% of the beam span. Global equilibrium was checked using the reactions in Figure 59, in the vertical and horizontal directions as well as...
moments with Equation 15 through Equation 17. In the equations presented, the ≈ sign symbolizes that the value on the left of the sign is near equal to the value on the right side of the sign. The nature of numerical modeling yields results that are near the exact solution.

Equilibrium between the pinned boundary condition and the deflected member was checked by summing the member forces exerted on the support and the reactions at that boundary condition.

Figure 60 shows the reactions at one of the pinned ends of the model with the member forces tension (T), shear (V), and moment (M) exerted on the boundary condition. Angles θ and α were 72.7 and 17.2 degrees respectively. The angle α was found by taking the inverse tangent of the y coordinate divided by
the x coordinate of the third node in the deflected shape of the model shown in Figure 61. The angle θ is simply 90 degrees minus α.

![Diagram showing forces and moments](image)

**Figure 60: Wall & Strut Element Verification: Pinned Beam, Member Equilibrium**

**Figure 61: Wall & Strut Element Verification: Pinned Beam, Model Displacement**

Equation 18 through Equation 20 show the equilibrium equations used to check the forces and moments at the boundary condition.
\[ \sum F_x = 0 \]
\[ T \cos(\alpha) - V \cos(\theta) - R_x = 0 \]
\[ 119.7607kN \cos(17.283^\circ) - 67.4648kN \cos(72.717^\circ) - 94.3088kN = 0.002kN \]  
Equation 18

\[ \sum F_y = 0 \]
\[ -T \sin(\alpha) - V \sin(\theta) + R_y = 0kN \]
\[ -119.7607kN \sin(17.283^\circ) - 67.4648kN \sin(72.717^\circ) + 99.9998kN = 0.002kN \]
Equation 19

\[ \sum M = 0 \]
Equation 20

Internal equilibrium was also checked by cutting the model just before the load application point and summing the internal forces at the midspan and the left boundary condition as shown in Figure 62.

Global equilibrium was checked using Equation 21 through Equation 23. The member forces for the whole model are shown in Figure 63. These equilibrium equations take into account the slight angle of the element connecting to the midspan node (\(N_{161}\)).

![Figure 62: Wall & Strut Element Verification: Pinned Beam, Internal Member Forces](image)

\[ \sum F_x = 0 \]
\[ R_x + T_m \cos(\alpha_m) - V_m \sin(\alpha_m) = 0 \]
\[ -94.30kN + 94.92kN \cos(0.356^\circ) - 99.04kN \sin(0.356^\circ) = -0.002kN \]
Equation 21

\[ \sum F_y = 0 \]
\[ R_y - T_m \sin(\alpha_m) - V_m \cos(\alpha_m) = 0 \]
\[ 99.99kN - 94.92kN \sin(0.356^\circ) - 99.04kN \cos(0.356^\circ) = 0.367kN \]
Equation 22
\[ \sum M = 0 \]

\[ T_m \sin(\alpha_m) \cdot \frac{L}{2} - M_m - T_m \cos(\alpha_m) \cdot \Delta_m + V_m \sin(\alpha_m) \cdot N_{159X} + V_m \cos(\alpha_m) \cdot N_{159Y} = 0 \]

\[ 94.92kN \sin(0.356\degree) \cdot 1524mm - 119600kN \cdot mm - 94.92kN \cos(0.356\degree) \cdot 315.5mm + 99.04kN \sin(0.356\degree) \cdot 1504mm + 99.04kN \cos(0.356\degree) \cdot 315.4mm = 0.506kN \cdot mm \]

**Figure 63:** Wall & Strut Element Verification: Pinned Beam, Model Member Forces

The second model that was analyzed was the same as shown in Figure 59, except oriented at a 45 degree angle and pinned at one end with a roller at the other as shown in Figure 64. This model was subjected to a 10 kN vertical force at the roller support. Global equilibrium was checked in the vertical and horizontal directions with Equation 24 through Equation 26 using the reactions in Figure 64.
Figure 64: Wall & Strut Element Verification: Simple Beam at 45°, Model and Reactions

\[ \sum F_x = 0 \]
\[ -6.6982kN + 6.6982kN = 0 \]  
Equation 24

\[ \sum F_y = 0 \]
\[ -10.0001kN + 10.0001kN = 0 \]  
Equation 25

\[ \sum M = 0 \]  
Equation 26

Equilibrium between the pinned boundary condition and the deflected member were checked by summing the member forces exerted on the boundary condition and the reactions at that boundary condition. Figure 65 shows the reactions at the pinned end of the model with the member forces, T and V, exerted on the boundary condition. Angles \( \theta \) and \( \alpha \) were 33.8 and 56.1 degrees respectively found using the deflection diagram in Figure 66.

Figure 65: Wall & Strut Element Verification: Simple Beam at 45°, Member Equilibrium
Figure 66: Wall & Strut Element Verification: Simple Beam at 45°, Model Displacements

Equation 27 through Equation 29 shows the equilibrium equations used to check the forces and moments at the boundary condition. A simple elongation check was performed using Equation 30. The member forces for the whole model are shown in Figure 67. The element did produce small moments at the ends of the model, however these moments are insignificant to the applied moment on the model.

\[
\sum F_X = 0 \\
T \cos(\alpha) + V \cos(\theta) - R_X = 0 \\
12.0361kN \cos(56.168°) + 0.172kN \cos(33.832°) - 6.6982kN = 0.003kN
\]

Equation 27

\[
\sum F_Y = 0 \\
T \sin(\alpha) - V \sin(\theta) + R_Y = 0 \\
12.0361kN \sin(56.168°) - 0.172kN \sin(33.832°) - 10.0001kN = -0.002kN
\]

Equation 28

\[
\sum M = 0 \\
Theoretical Elongation = Model Elongation \\
\frac{PL_0}{AE} = L - L_0 \\
\frac{12.0361kN \times 14.142mm}{6452mm^2 \times 6.895MPa} = 17.969mm - 14.142mm = 3.827mm
\]

Equation 30
The next model that was analyzed employed the three-noded Timoshenko beam element with an internal hinge at an end node. The same model was created as the pinned beam in Figure 59, however the internal hinged element was placed such that the hinge was located at the start and end of the beam with fixed boundary conditions at either end. These boundary conditions were chosen such that the results could be compared to the pinned beam model shown previously to verify the functionality of the hinged elements. All other elements used in this model were the regular, linear elastic three-noded beam elements. A 200 kN point load was applied at the midspan as shown in Figure 68. This model produced nearly identical reactions and displacements as the previous, nominally identical model, verifying the correct formulation of the hinged element.

**Figure 67: Wall & Strut Element Verification: Simple Beam at 45°, Model Member Forces**
Figure 68: Wall & Strut Element Verification: Fixed-Hinged Beam: Model and Reactions

\[ \sum F_x = 0 \]
\[ -95.94kN + 95.94kN = 0N \]  \hspace{1cm} \text{Equation 31}

\[ \sum F_y = 0 \]
\[ 100.0kN + 100.0kN - 200kN = 0N \]  \hspace{1cm} \text{Equation 32}

\[ \sum M = 0 \]  \hspace{1cm} \text{Equation 33}

Equilibrium between the pinned boundary condition and the deflected member were checked by summing the member forces exerted on the boundary condition and the reactions at that boundary condition. This load produced a mid-span deflection of 314 mm, which is about 10\% of the beam span. Figure 69 shows the reactions at the left fixed end of the model with the member forces, T, V, and M exerted on the boundary condition. Angles \( \theta \) and \( \alpha \) were 72.7 and 17.2 degrees respectively found using the deflection diagram in Figure 70.
Figure 69: Wall & Strut Element Verification: Fixed-Hinged Beam at 45°, Member Equilibrium

The same equilibrium checks were performed as shown in Equation 18 through Equation 20. The member forces for the whole model are shown in Figure 71 which closely match that of the pinned beam.
Finally the roof tension only elements were modeled to verify their functionality. These elements are different from the other tension only elements in PressArchAnalysis because they are used as loaded members rather than connecting members. The behavior of these elements was verified in PressArchAnalysis through a horizontal beam model loaded by a 10 N/mm distributed vertical load with fixed-fixed boundary conditions shown in Figure 72. The fixed-fixed boundary conditions were chosen to verify that the elements were not able to carry moment since this element was intended to be tension only. The distributed load was simulated with a series of point loads applied to all nodes that were not restrained by a boundary condition. Load that was applied directly to a node restrained by a boundary condition was not included in the reactions of the model.
The same equilibrium checks were performed as mentioned in the previous element verifications.

Global equilibrium was checked by summing the reactions in the horizontal direction, and the reactions and applied load in the vertical direction. The boundary conditions caused the sum of the applied load to equal 9 kN. Figure 73 shows the reactions on the model. The tension only nature of the element was verified as no moment reactions were generated. Equation 34 through Equation 36 show the equilibrium equations used to verify global equilibrium of the model.

Equilibrium between the fixed boundary condition and the deflected member were checked by summing the member forces exerted on the boundary condition and the reactions at that boundary condition. Figure 74 shows the reactions at the left boundary condition of the model with the member
force T exerted on the boundary condition. Angle $\alpha$ was $56.8^\circ$ found using the deflection diagram in Figure 75.

![Figure 74: Roof Element Verification: Fixed Beam, Member Equilibrium](image)

Large deformation equilibrium was checked using Equation 37 through Equation 39. The elements did not generate any moment or shear forces as shown in Figure 76.

\[
\sum F_X = 0
\]

\[
T \cos(\alpha) - R_x = 0
\]

\[
5.3756kN \cos(56.848^\circ) - 2.9388kN \approx 0
\]
\[ \sum F_y = 0 \]
\[ -T \sin(\alpha) + R_y = 0 \]
\[ -5.3756kN \sin(56.848°) + 4.5kN \approx 0 \]
\[ \sum M = 0 \]

Equation 38

Equation 39

Figure 76: Roof Element Verification: Fixed Beam, Model Member Forces

These models showed that PressArchAnalysis was able to run simple models with additional element types added to its library. The new three-noded Timoshenko beam element with hinge was verified to function correctly and the tension only roof element were verified to be able to carry load. The properties of these elements based on the anticipated HRAB structure are estimated in the following section.
5.3. Generation of the HRAB Model

5.3.1. Element Properties

With the verification of the functionality of the wall, strut, and roof elements completed, representative properties of the elements needed to be determined. Properties of these elements were based on reasonable assumptions from current materials used in similar structures, properties of the components used in the testing in Chapter 4, or reasonable assumptions for known materials. The overall length of structure tributary to a single arch was assumed to be 2438 mm, the standard panel product length that was used in the physical proof-of-concept tests in Chapter 4.

5.3.1.1. Roof Properties

The roof fabric was assumed to be a vinyl coated, nylon woven fabric. It was indicated that modern shelters use this kind of fabric (Matthews, 2016), and the Natick Soldier Research Center has tested vinyl coated nylon fabrics in the past and reported properties (Sebring et al., 1969, p. 11). The fabric tested by Sebring had an areal weight of 6.085 Pa, which is slightly higher than the current fabric used on some modern shelters (CAMSS Shelters, n.d.) but very close to others (HTS Roder Rapid Deployment Shelters, n.d.). Using the ultimate strength and strain percentage of the fabric provided by Sebring in table 2 on page 18, a secant modulus of 234 N/mm was computed, reported in units of Newtons per unit width of fabric. Sebring showed that the axial stiffness of the vinyl coated fabric is nearly linear and does not experience a large plastic deformation range allowing the use of a secant modulus for the whole range to be an acceptable assumption.

5.3.1.2. Wall Properties

The wall panels were idealized as composite sandwich panels that span the width of the segment (2.4 m). The sandwich panels were chosen to have a [0, 90] laminate face sheets of Polystrand™ IE 5842 and a 1 inch thick core of GURIT® G-PET™ FR 75. The material properties of Polystrand™ IE 5842 used in determining the wall panel properties were experimentally determined in previous research (Seigars,
The foam core was assumed to be an isotropic material. Its material properties were provided by its corresponding material data sheet (Gurit, 2015). Global modulus of elasticity and shear modulus were determined through Classical Lamination Theory (Barbero, 2018). The laminate moduli that were found apply to an equivalent orthotropic plate subjected only to in-plane loads. The geometric properties of the panels, moment of inertia and cross-sectional area, were based off the dimensions of the panels accounting for their width. The sandwich composite panel was estimated to weigh 90 N for a 1.06 m by 2.4 m panel yielding an areal weight of 34.7 Pa. The values used in the model are provided in Table 6.

**Table 6: HRAB Wall Element Properties**

<table>
<thead>
<tr>
<th>Self-Weight</th>
<th>Modulus of Elasticity</th>
<th>Shear Modulus</th>
<th>Moment of Inertia</th>
<th>Cross-Sectional Area</th>
<th>Height</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pa</td>
<td>MPa</td>
<td>MPa</td>
<td>Mm^4</td>
<td>mm^2</td>
<td>mm</td>
</tr>
<tr>
<td>34.7</td>
<td>786</td>
<td>81</td>
<td>3329851</td>
<td>61935</td>
<td>2133</td>
</tr>
</tbody>
</table>

**5.3.1.3. Strut Properties**

The strut properties in the model were based on the properties of the wooden strut used in the scale model. The strut consisted of two 2x4s secured together with screws shown below in Figure 77.

![HRAB Model Strut](image)

**Figure 77: HRAB Model Strut**

The species of the wood was assumed to be Spruce-Pine-Fir, the wood was stud grade. Geometric properties were found using values for X-X axis (edgewise) bending of two 2x4s. The modulus of elasticity used was the reference design value (American Wood Council, 2017, p. 37). The shear modulus of the wood was found using the elastic ratio for $G_{LT}/E_L$ provided by the US Forest Service for Douglas
Fir with approximately 12\% moisture content (Forest Products Laboratory (U.S.), 2010, pp. 5–2). Values for spruce pine fir was not reported hence Douglas Fir was used instead. This value was near the low end of the spectrum for the different pine, spruce, and fir species. This is not a critical value because the member is an axial loaded element and not subjected to shear. The strut weights used in the experimental model of the HRAB model were not used because it was assumed that the production version of the HRAB shelter would utilize a lighter, more optimized component. The self-weight of the strut was therefore chosen to have the same areal weight as the sandwich composite wall panels (33.5 Pa) but with a 304 mm width yielding a self-weight of 0.01 N/mm along the length of the strut. However while running the HRAB PressArchAnalysis model, numerical issues arose using this self-weight with a 206 kPa internal pressure and 200 N guy pre-tension at 45 degrees at model. It was changed to 0.011 N/mm which allowed the model to run. The values used in the model are provided in Table 7.

<table>
<thead>
<tr>
<th>Self-Weight</th>
<th>Modulus of Elasticity</th>
<th>Shear Modulus</th>
<th>Moment of Inertia</th>
<th>Cross-Sectional Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/mm</td>
<td>MPa</td>
<td>MPa</td>
<td>mm²</td>
<td>mm²</td>
</tr>
<tr>
<td>0.011</td>
<td>8274</td>
<td>645.3</td>
<td>4461480</td>
<td>6774</td>
</tr>
</tbody>
</table>

5.3.1.4. Airbeam Properties

The properties of the airbeam were based on the specimens that were used for testing in Chapter 3 and 4. The self-weight of the airbeam was found by weighing specimen M11870-8 and dividing its total weight by the total length of the beam. The cross-sectional diameter was measured to be 254 mm (Brayley, 2011, p. 75). The axial fabric modulus (E) and fabric shear modulus (G) varied depending on the internal pressure of the airbeam. A range of E values was reported by Brayley for a polyurethane coated, 75° bias angle fabric in table 6.45 (Brayley, 2011, p. 214). A range of G values was reported by Brayley for a polyurethane coated, 75° bias angle fabric in Table 3 (Brayley et al., 2012, p. 56). Linear interpolation was used to find the E and G values not reported by Brayley. The fabric bias angle of the
beam was measured to be 75 degrees. The strap modulus was determined experimentally (Brayley, 2011, p. 240). The angle of the straps provided was measured counterclockwise from the horizontal axis of the beam to the center of the strap. The thickness of the straps was 3.17 mm (Brayley, 2011, p. 120).

A summary of the values discussed is presented in Table 8.

**Table 8: HRAB Airbeam Element Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arch Radius (mm)</td>
<td>3048</td>
</tr>
<tr>
<td>Arch Spacing (mm)</td>
<td>2438</td>
</tr>
<tr>
<td>Cross-Sectional Diameter (mm)</td>
<td>254</td>
</tr>
<tr>
<td>Fabric Bias Angle</td>
<td>75</td>
</tr>
<tr>
<td>Strap Modulus (N/mm)</td>
<td>13351</td>
</tr>
<tr>
<td>Strap Locations (degrees)</td>
<td>65, 115, 245, 295</td>
</tr>
<tr>
<td>Strap Width (mm)</td>
<td>25.4</td>
</tr>
<tr>
<td>Strap Thickness (mm)</td>
<td>3.17</td>
</tr>
<tr>
<td>Self-Weight (N/mm)</td>
<td>0.016</td>
</tr>
</tbody>
</table>

**Table 9: HRAB Airbeam Element Axial and Shear Moduli Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Internal Airbeam Pressure (kPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>137</td>
</tr>
<tr>
<td>Effective Axial Fabric Modulus E (N/mm)</td>
<td>43</td>
</tr>
<tr>
<td>Effective Shear Fabric Modulus G (N/mm)</td>
<td>263</td>
</tr>
</tbody>
</table>

**5.3.2. Finite-Element Meshes**

The meshing of the HRAB structure in PressArchAnalysis was set to mimic the ideal HRAB structure with a 3048 mm radius arch. The walls were pinned to the ground at the same location as the arch bases, which were also pinned. Hinged elements were used at the base of the wall with the hinge located at the boundaries. The struts were connected perpendicularly to the arch from the top of the wall. The struts had hinges at both ends, with one hinge connected to the airbeam and the other to the top of the wall. The roof was attached tangentially from the top of the wall to the airbeam. With the small moment
of inertia in guy elements, as well as the model verification, it was ensured that the ends of the roof elements would act as hinges at the connections to the wall and airbeam. Guys were attached from the top of the walls to their bases 2133 mm horizontally from the base of the walls. This guy anchor location was chosen to generate a $45^\circ$ angle between the guy and the ground which is explained in section 5.5.

The airbeam was chosen to utilize 60 evenly spaced elements along the arc length of the arch. It was shown by Davids that 60 elements are sufficient for convergence of an arch model subjected to complex gravity load patterns (Davids, 2009). The roof and strut elements were chosen to have similar element length as those of the airbeam elements. The wall elements had element lengths twice that of the airbeam elements. Meshing of the strut and wall components could have been coarser since these elements were only exposed to axial loading and buckling of these members was not a concern due to small loads. The meshing of the HRAB structure is shown below in Figure 78. The red vertical lines on the roof of the structure represent the magnitude of the force applied at that node compared to the other applied forces. Since the magnitude of the distributed load on the airbeam is much smaller than that of the point loads generated by the roof, the arch arrow length is small compared to that of the roof point load lines.

![HRAB Discretized Model](image)

**Figure 78: HRAB Discretized Model**
The baseline for comparison of these models was the TEMPER shelter. Figure 79 shows the equivalent TEMPER model that the shelter was compared to in the parametric tests described in section 5.6. In modeling the TEMPER shelter, the same number of equal length airbeam elements was used. Additionally the fabric properties of the airbeam were changed with pressure. In summary, if the HRAB components (wall, struts, roof elements) were removed, uniform gravity load re-distributed, and the guy lines were placed on the airbeam at a y coordinate of 2133 mm the resulting shelter would be that of the TEMPER.

![Figure 79: TEMPER Discretized Model]

5.3.3. HRAB Loading

The HRAB model was subjected to a uniform gravity load as well as the dead loads provided in section 5.3.1. The dead loads and live loads were assigned to the start and end nodes of each of the elements based on tributary width and the horizontal projection of each element. This projected load was halved and placed at the start and end nodes of each element, including the tension-only roof elements and airbeam elements above the roof. Initial testing with a distributed load applied to the tension-only roof elements yielded convergence issues. Therefore half of the total load tributary to the roof elements was applied to the attachment points on the airbeam and the tops of the walls. This method of load
Application is shown in Figure 80, where the length of the arrow represents the magnitude of the force applied compared to the other applied forces. Since the magnitude of the distributed load on the airbeam is much smaller than that of the point loads generated by the roof, arch arrow length is small compared to that of the roof point load lines. Further refinement of the model was conducted to apply the roof loads to the roof elements. Results from these models are presented at the end of section 5.6.1.

**Figure 80: Concentration of Roof Load**

The dead weight of the fabric was included in the distributed gravity load. Pretension values for the guy lines were found through parametric studies of the arch which are discussed in section 5.5. The chosen values were 0 N and 200 N. Pretension for the roof elements was found through summing the forces at the top of the wall panel. It was found that 58% of the guy pretension goes into the roof elements with the current geometry of the roof and wall elements while the guy line is at 45° with respect to the ground. Therefore if the guy line is pretensioned to 100 N, the pretension in the roof elements would be 58 N. The tension force generated in the roof membrane by the gravity load in the roof was not accounted for by placing the roof loads at the roof’s attachment points to the structure.
Two loading scenarios were examined. The first was the structure under only gravity loads. However, such a loading scenario may not be realistic, since all structures have some small lateral load or lack of symmetry that will induce lateral movement and can generate instability. Therefore a perturbing force of 0.01% of the total applied gravity load was applied to the apex of the arch to simulate small, unintended horizontal loads or structure imperfections. This small perturbing force can be seen in Figure 80 at the apex of the arch. The second loading scenario included 10% of the gravity load at each node applied at the same node as a horizontal force. This second loading case was constructed to assess the significance of larger horizontal loads on stability of the structure. All horizontal loads are artificial, and intended to perturb the structure to model arch geometric imperfections, since perfect initial geometry was assumed in all analyses.

5.4. PressArchAnalysis Solution Procedure

During the analysis of a model, PressArchAnalysis incorporates both a dead load and live load solution stage. The dead load stage includes solving for the dead loads and guy pre-tension with a Newton solver that changes the strain in the guy lines until the specified guy pretension is achieved. In the second load stage, externally applied loads on the structure are applied in user-specified equal increments. A Newton solver is used initially to incrementally increase load until the structure becomes unstable, which generally corresponds to a decrease in capacity. An arc-length solver is then automatically employed to find forces and displacements after the structure has reached its peak load. An example of the ranges that the solvers operate in is shown below in Figure 81.
Figure 81: Load Fraction vs. Apex Displacement, Airbeam, 206 kPa, 0 N Guy Pretension

One difference between the dead load and live load stage is the use of load increments. The dead load stage applies all of the dead load and guy pretensions at once whereas the live load stage gradually applies the live load until an instability is detected. This single lump application of the dead loads and guy pretensions can cause certain models to fail to converge before the application of external loads as discussed in the next section.

5.5. Parametric Study of Guy Pre-Tension & Angle

Before models were used to determine capacity, the effect of the pretension in the guy lines and their angle of attachment to the arch were studied. The parameters for the study are shown below in Table 10. Both the HRAB structure with and without the 10% applied horizontal load were tested for each combination of the parameters.
Table 10: HRAB Parametric Study Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal Airbeam Pressure (kPa)</td>
<td>137, 206, 275, 344</td>
</tr>
<tr>
<td>Guy Pretension (N)</td>
<td>0, 100, 200, 300, 400</td>
</tr>
<tr>
<td>Guy Angle with ground (degrees)</td>
<td>20, 30, 40, 50, 60, 70</td>
</tr>
</tbody>
</table>

For these models, each component’s self-weight was set to 0 and only a uniform external gravity load was applied. The vertical load was distributed over the structure as shown previously in Figure 78. The external gravity load was increased until the structure could not carry additional load due to softening or instability. In general, the models could be run to larger displacements with decreasing load.

Figure 82 shows the results for the HRAB model where no horizontal load was applied and Figure 83 shows the results for when the 10% horizontal load is applied to vertically loaded elements. The missing data points indicate where the model was unable to solve the dead load and guy pretension on the structure. These results indicate that the angles of the guy lines do not significantly affect the gravity load capacity of the structure. Table 11 shows the data points gathered during the parametric study.

Each point in the table represents the peak load in Pascals that the model was able to carry. The cells with an X indicate that the model was unable to solve the dead load and guy pretension on the structure. Both the horizontally loaded and non-horizontally loaded models failed for the exact same parameters. It can also be seen that the models with zero guy pretension were always able to converge.

The results indicate that a shallow guy angle will cause the model to fail to converge at guy pretensions between 100 N and 300 N.
Figure 82: Applied Load vs. Guy Angle, HRAB No Hz Load

Figure 83: Guy Angle vs. Peak Applied Load, HRAB with Hz Load
Table 11: HRAB Guy Angle and Pretension Parametric Study Results: Model Gravity

Load Capacities in Pascals

<table>
<thead>
<tr>
<th>Pretension (N)</th>
<th>Guy Angle (Degrees)</th>
<th>Pressure (kPa)</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No Horizontal Load</td>
<td>137</td>
<td>206</td>
<td>275</td>
<td>344</td>
<td>Horizontal Load</td>
<td>137</td>
</tr>
<tr>
<td>0</td>
<td>20</td>
<td>224</td>
<td>334</td>
<td>439</td>
<td>537</td>
<td>192</td>
<td>276</td>
<td>361</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>224</td>
<td>334</td>
<td>439</td>
<td>537</td>
<td>197</td>
<td>278</td>
<td>363</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>224</td>
<td>334</td>
<td>438</td>
<td>537</td>
<td>197</td>
<td>278</td>
<td>363</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>224</td>
<td>334</td>
<td>438</td>
<td>537</td>
<td>193</td>
<td>276</td>
<td>361</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>224</td>
<td>334</td>
<td>439</td>
<td>537</td>
<td>186</td>
<td>274</td>
<td>358</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>224</td>
<td>334</td>
<td>439</td>
<td>537</td>
<td>183</td>
<td>270</td>
<td>353</td>
</tr>
<tr>
<td>100</td>
<td>20</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>225</td>
<td>334</td>
<td>X</td>
<td>536</td>
<td>201</td>
<td>282</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>225</td>
<td>X</td>
<td>438</td>
<td>536</td>
<td>198</td>
<td>X</td>
<td>366</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>225</td>
<td>334</td>
<td>439</td>
<td>536</td>
<td>189</td>
<td>278</td>
<td>361</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>225</td>
<td>334</td>
<td>438</td>
<td>536</td>
<td>185</td>
<td>273</td>
<td>355</td>
</tr>
<tr>
<td>200</td>
<td>20</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>225</td>
<td>334</td>
<td>438</td>
<td>X</td>
<td>203</td>
<td>286</td>
<td>372</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>225</td>
<td>334</td>
<td>X</td>
<td>536</td>
<td>201</td>
<td>284</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>225</td>
<td>334</td>
<td>439</td>
<td>536</td>
<td>193</td>
<td>281</td>
<td>365</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>224</td>
<td>334</td>
<td>438</td>
<td>537</td>
<td>188</td>
<td>276</td>
<td>358</td>
</tr>
<tr>
<td>300</td>
<td>20</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>226</td>
<td>X</td>
<td>438</td>
<td>X</td>
<td>205</td>
<td>X</td>
<td>375</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>225</td>
<td>334</td>
<td>438</td>
<td>536</td>
<td>202</td>
<td>288</td>
<td>373</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>225</td>
<td>334</td>
<td>439</td>
<td>536</td>
<td>196</td>
<td>284</td>
<td>368</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>225</td>
<td>X</td>
<td>439</td>
<td>537</td>
<td>191</td>
<td>X</td>
<td>361</td>
</tr>
<tr>
<td>400</td>
<td>20</td>
<td>224</td>
<td>331</td>
<td>437</td>
<td>536</td>
<td>212</td>
<td>298</td>
<td>383</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>227</td>
<td>334</td>
<td>437</td>
<td>536</td>
<td>211</td>
<td>296</td>
<td>381</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>227</td>
<td>334</td>
<td>438</td>
<td>536</td>
<td>208</td>
<td>294</td>
<td>380</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>226</td>
<td>X</td>
<td>438</td>
<td>536</td>
<td>205</td>
<td>X</td>
<td>376</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>X</td>
<td>334</td>
<td>438</td>
<td>X</td>
<td>X</td>
<td>287</td>
<td>372</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>223</td>
<td>334</td>
<td>439</td>
<td>537</td>
<td>193</td>
<td>281</td>
<td>364</td>
</tr>
</tbody>
</table>
Overall, these results indicate that guy line inclination does not significantly effect structure capacity. Therefore the guy line inclination was fixed at the intermediate value of 45 degrees. Additionally, these results show that the pretension in the guy lines does not have an impact on the peak load the structure can carry. Hence the maximum pretension in the guy lines was set to 200 N, which is a reasonable pretension that a person should be able to produce using a ratcheting guy line as used in the TEMPER shelter (Tom Artes, 2014).

5.6. Parametric Study of the HRAB Model

A parametric study of the HRAB model was established using the findings of section 5.5 to define parameters. The angle of the guy lines was set to 45 degrees with their pretension at 0 N and 200 N. The internal pressure of the airbeam was varied from 137 kPa to 344 kPa in 69 kPa increments (Table 12). The models were run using an optimization routine to determine the maximum load the structure was able to carry. This was achieved by using the bisection method to vary the gravity load the model was subjected to. This gravity load included the applied gravity load as well as the self-weight of the roof fabric. It was verified during these analyses that these models did not fail in the dead load and guy pretension stage. Therefore it was known that the model would either carry the applied load of that iteration or fail in global buckling. The bisection method started the loads at just the self-weight of the fabric and the self-weight of the fabric plus 1000 Pa which would fail the 344 kPa models in global buckling. From there the routine varied the load until the peak load achieved by the model reached at least 98% of the applied load specified in the input files. This was done by taking the average of the previous loads and setting that result to either the upper or lower bound of the load depending on whether that average load was able to be carried by the model or not. The new upper and lower bounds were then used to find a new average load. This ensured that at least 98% of the self-weight was successfully applied to the model. A 10% horizontal load was also applied to all vertically loaded nodes.
as described in section 5.3.3. As mentioned in section 5.3.1.4, the properties of the airbeam are pressure dependent and were changed for each pressure.

### Table 12: HRAB Parametric Study Parameters

<table>
<thead>
<tr>
<th>Property</th>
<th>Value(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal Airbeam Pressure (kPa)</td>
<td>137, 206, 275, 344</td>
</tr>
<tr>
<td>Guy Angle (degrees)</td>
<td>45</td>
</tr>
<tr>
<td>Guy Pre-Tension (N)</td>
<td>0, 200</td>
</tr>
<tr>
<td>Horizontal Load Percentage (%)</td>
<td>0, 10</td>
</tr>
</tbody>
</table>

This study of the HRAB structure was compared to the equivalent TEMPER structure for each set of parameters as shown previously in Figure 78 and Figure 79. The same load was applied to the airbeam structure as the HRAB structure. The guy lines in the TEMPER structure were placed at the wall height of the HRAB structure (2133 mm) on the airbeam. This maintained the 45 degree angle the guy line created with the horizontal as well as attaching to the TEMPER structure at the same height as the HRAB structure. Opportunities for improvement are discussed in section 5.7.

**5.6.1. Results**

Summary plots of results for each structure for each combination of parameters were created to show the relevant information regarding global structure displacement, load capacity vs. airbeam apex displacement, airbeam moment vs. x location, and airbeam moment to wrinkle moment ratio vs. x location. Plots for all parameter combinations are included in Appendix A for the HRAB and TEMPER arch analyses. Figure 84 and Figure 85 are provided below for a 275 kPa internal pressure with 200 N guy-pretension with a 10% horizontal load applied to all vertically loaded nodes. In all HRAB models, the vertical reaction forces of the walls are in tension implying that the airbeam is pushing on the strut elements subjecting them to compressive forces. The strut then in turn exerts a vertical force up on the wall subjecting it to tension. This is discussed further after the next page.
Figure 84: HRAB 275 kPa, 200 N Guy PT, HZ

Figure 85: TEMPER 275 kPa, 200 N Guy PT, HZ
Below in Figure 86 the reactions are shown for the HRAB model shown in Figure 84. In Figure 86 the reactions at the supports under the walls and airbeam show two separate numbers for the two separate boundary conditions at the wall (second number) and airbeam (first number) in the vertical and horizontal direction. As mentioned in section 5.3.2, the walls and airbeam have boundary conditions at the same geometric location. At the top of the airbeam the sum of the horizontal and vertical gravity loads are shown with the total self-weight of the structure shown at the very top. The direction of the reaction arrow corresponds to the resulting reaction force.

\[ \sum F_X = 0 \]
\[ 1.339kN - 0.5567kN - 0.0078kN - 0.0656kN - 1.195kN + 0.4854kN \approx 0kN \]  
Equation 40

\[ \sum F_Y = 0 \]
\[ 4.113kN + 3.051kN - 0.1888kN - 0.4296kN - 1.1161kN - 4.855kN - 0.6046kN = -0.0750kN \]  
Equation 41

\[ \sum M = 0 \]  
Equation 42

Figure 86: Equilibrium of HRAB Structure, 275 kPa, 200 N Guy PT, HZ
Equilibrium was checked on the east wall only subjected to the applied load. Figure 87 shows a diagram of the forces acting on the wall panel and their respective directions in the deformed state. These forces correspond to the model shown in Figure 84. The reactions, forces, and displacements correspond the final load the model was subjected to. Equation 43 through Equation 45 show the equilibrium equations used to check the diagram in Figure 87. Moment equilibrium is taken as clockwise positive in Equation 45. The resultant moment may appear large, however it is only 8.6% of the smallest major contributor to the moment in the system shown in Table 14.

![Equilibrium Diagram](image)

*Figure 87: Equilibrium of HRAB West Wall, 275 kPa, 200 N Guy PT, HZ*
Table 13: Equilibrium Values of HRAB West Wall, 275 kPa, 200 N Guy PT, HZ

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_R, \theta_R$</td>
<td>3348 N, 12.13°</td>
</tr>
<tr>
<td>$T_S, \theta_S$</td>
<td>-2677 N, 36.60°</td>
</tr>
<tr>
<td>$T_G, \theta_G$</td>
<td>1666 N, 44.19°</td>
</tr>
<tr>
<td>$W_W, \theta_W$</td>
<td>166.1 N, 1.61°</td>
</tr>
<tr>
<td>$R_X$</td>
<td>-7.85 N</td>
</tr>
<tr>
<td>$R_Y$</td>
<td>-188.8 N</td>
</tr>
<tr>
<td>$F$</td>
<td>-778.7 N</td>
</tr>
<tr>
<td>$(x, h)$</td>
<td>(59.9 mm, 2132 mm)</td>
</tr>
</tbody>
</table>

\[
\sum F_X = 0
\]

\[
R_X + T_R \cos(\theta_R) + T_S \cos(\theta_S) - T_G \cos(\theta_G) = 0
\]

\[
-0.0078kN + 3.348kN \cos(12.13°) - 2.677kN \cos(36.60°) - 1.666kN \cos(44.19°) = -0.0787kN
\]

\[
\sum F_Y = 0
\]

\[
R_Y + T_R \sin(\theta_R) - T_S \sin(\theta_S) - T_G \sin(\theta_G) - W_W - F = 0
\]

\[
-0.1888kN + 3.348kN \sin(12.13°) + 2.677kN \sin(36.60°) - 1.666kN \sin(44.19°) - 0.1661kN - 0.7787N = 0.0047kN
\]

\[
\sum M = 0
\]

\[
M_G + M_S + M_R + M_W + M_F = 0
\]

\[
(-T_G \cos(\theta_G) \cdot \cos(\theta_W)) + T_G \sin(\theta_G) \sin(\theta_W))h + (T_S \cos(\theta_S) \cdot \cos(\theta_W)) + T_S \sin(\theta_S) \sin(\theta_W))h + (T_R \cos(\theta_R) \cdot \cos(\theta_W) - T_R \sin(\theta_R) \sin(\theta_W))h + +W_W \frac{h}{2} \sin(\theta_W) + F \cdot h \sin(\theta_W) = 0
\]

\[
(1.666kN \cos(44.19°) \cdot \cos(1.61°) + 1.666kN \sin(44.19°) \sin(1.61°))2132mm + (2.677kN \cos(36.60°) \cdot \cos(1.61°) - 2.677kN \sin(36.60°) \sin(1.61°))2132mm + (3.348kN \cos(12.13°) \cdot \cos(1.61°) - 3.348kN \sin(12.13°) \sin(1.61°))2132mm + +166.1 \frac{2132mm}{2} \sin(1.61°) + 0.7787kN \cdot \sin(1.61°) = -214.2kN \cdot mm
\]

Table 14: Moments Applied to HRAB West Wall, 275 kPa, 200 N Guy PT, HZ
The results of the parametric study are summarized below in Table 15. The values in the table are the maximum load capacity, within 2%, of the structure not including the self-weight of the fabric. Table 16 shows the percent increase of load capacity of the HRAB shelter over the TEMPER shelter. It is shown that for all pressures and guy pretensions the HRAB shelter offers a slight increase in load capacity. Figure 88 and Figure 89 graphically show the increase of structure load capacity with pressure. Additionally it can be seen that the guy pretension slightly increases the capacity of both structures when subjected to a horizontal load.

### Table 15: HRAB & TEMPER Load Capacities

<table>
<thead>
<tr>
<th>Airbeam Internal Pressure (kPa)</th>
<th>Vertical Load Only</th>
<th>Vertical + 10% Horizontal Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>137</td>
<td>205</td>
<td>275</td>
</tr>
<tr>
<td>HRAB Guy Pretension (N)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>224</td>
<td>334</td>
</tr>
<tr>
<td>200</td>
<td>224</td>
<td>333</td>
</tr>
<tr>
<td>TEMPER Guy Pretension (N)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>217</td>
<td>315</td>
</tr>
<tr>
<td>200</td>
<td>219</td>
<td>315</td>
</tr>
</tbody>
</table>
Table 16: Load Capacity Increase of HRAB Compared to TEMPER, Percent

<table>
<thead>
<tr>
<th>Airbeam Internal Pressure (kPa)</th>
<th>Vertical Load Only</th>
<th>Vertical + 10% Horizontal Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guy Pretension (N)</td>
<td>137</td>
<td>205</td>
</tr>
<tr>
<td>0</td>
<td>3.1</td>
<td>5.7</td>
</tr>
<tr>
<td>200</td>
<td>2.2</td>
<td>5.6</td>
</tr>
</tbody>
</table>

Figure 88: Horizontally Loaded Models
Figure 89: Non-Horizontally Loaded Models

Further refinement of the model allowed for distributed loads to be applied directly to the roof elements as shown below in Figure 90.

Figure 90: HRAB Discretized Model with Roof Distributed Loads
The same parametric study was conducted using the parameters in Table 12. During these analyses the peak load softening response was not observed. The load capacities for the same models presented in Table 15 are shown below for the model with the distributed roof loads in Table 17. Table 18 shows the percent increase or decrease of the distributed roof load model load capacity over the TEMPER shelter. These models showed similar results to that of the point loaded models showing that the assumption of using the point load did not have a large effect on the structure’s capacity.

Table 17: HRAB & TEMPER Load Capacities with Roof Distributed Load

<table>
<thead>
<tr>
<th>Airbeam Internal Pressure (kPa)</th>
<th>Vertical Load Only</th>
<th>Vertical + 10% Horizontal Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>137</td>
<td>207 201 218 220</td>
<td>192 198 194 198</td>
</tr>
<tr>
<td>205</td>
<td>230 231 236 236</td>
<td>287 294 274 280</td>
</tr>
<tr>
<td>275</td>
<td>265 256 276 274</td>
<td>375 383 344 350</td>
</tr>
<tr>
<td>344</td>
<td>291 281 341 361</td>
<td>457 464 405 412</td>
</tr>
</tbody>
</table>

Table 18: Load Capacity Increase of HRAB with Roof Distributed Load Compared to TEMPER, Percent

<table>
<thead>
<tr>
<th>Airbeam Internal Pressure (kPa)</th>
<th>Vertical Load Only</th>
<th>Vertical + 10% Horizontal Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>137</td>
<td>-4.9 -8.7</td>
<td>-1.0 -0.2</td>
</tr>
<tr>
<td>205</td>
<td>1.5 -0.9</td>
<td>4.4 4.7</td>
</tr>
<tr>
<td>275</td>
<td>3.5 1.8</td>
<td>8.6 8.8</td>
</tr>
<tr>
<td>344</td>
<td>-5.0 3.8</td>
<td>12.1 12.0</td>
</tr>
</tbody>
</table>
5.7. Summary

The HRAB structure is predicted to have a slight increase in gravity load capacity over the TEMPER structure. The reactions of the structure indicate the walls of the HRAB model are in tension which was not expected. The benefit of these results is amplified by the other qualitative gains the HRAB structure provides that do not relate directly to load carrying capacity, including the increase in useable interior volume and the addition of rigid walls which can facilitate up-armoring.

5.7.1. Model Limitations

Modeling of the HRAB structure showed that the model is sensitive to the input parameters that are used for the HRAB components. The results presented in this chapter are conservative with regards to the motion of the airbeam. In these models the motion of the airbeam leg near the wall was allowed to extend past the wall element. This allows the moment on that portion of the beam to increase past the wrinkling moment of the arch generating a hinge at that location. If this hinge was prevented from being formed it would push the hinge to generate in another location with a higher required load to generate that hinge hence increasing the shelter gravity load capacity.

The greatest improvement that could be made is the geometric limitation of the airbeam such that it is not able to extend past the wall elements at any given displacement. As shown by the displaced shape in Figure 75, the model predicts that the arch laterally displaces more than the wall as the structure experiences sidesway, which is unrealistic. In reality, the walls will confine the airbeam, which would in theory increase the capacity of the structure. However, failure modes of the wall panels would need to be explored to determine if the wall panels would be able to withstand the extra load exerted on them from the airbeam. Further refinement of the model with distributed roof loads is needed to ensure that post peak load softening response is achieved, since the current models were not able to capture post-peak softening with distributed roof loads.
CHAPTER 6

SUMMARY AND CONCLUSIONS

The objectives of this research were to:

1) Quantify the maximum lifting force of a TEMPER tent airbeam during inflation

2) Develop a preliminary design of an airbeam-supported tent with integral rigid walls to form a Hybrid Rigid-wall AirBeam (HRAB) structure, and experimentally assess its ability to erect itself by airbeam inflation; and

3) Quantify the gravity load capacity of an HRAB relative to a conventional, stand-alone TEMPER tent airbeam.

The main contribution of this work is the testing of airbeam arches from their deflated state to assess their ability to exert load and lift known weights, and the demonstration that airbeams can be used to erect a rigid-walled structure. The rest of this chapter summarizes the results and conclusions of the research undertaken to achieve these objectives, and contains recommendations for future research.

6.1. Summary of Airbeam Erection Capacity

Airbeams were tested at the University of Maine’s Advanced Structures and Composites Center in static restrained conditions. Two nominally identical 254 mm diameter, 6096 mm span braided and strapped airbeam arches were configured such that their full rise was 1828 mm. These arches were inflated to 137 kPa while restrained to rises between 749 mm and 1828 mm. These arches were tested with pin-pin boundary conditions and roller-roller boundary conditions with a horizontal tie connecting the two supports. While being inflated, the force required to restrain the arch was recorded. It was necessary to brace the arch against out of plane motion in order to ensure that the arch would rise vertically.

Arches were also tested dynamically to assess lifting capacity. During inflation, it was observed that the arches had a preferred shape that they would take while rising which was a near mirror image of each
other. This shape also had a large effect on the lifting capacity of the arch. However a general increase of restraining force was observed with the increase of arch rise. It was also observed that in a dynamic lifting scenario, the arch behaves as a damped oscillator post-erection and that the transition from a restrained state to an unrestrained state is very sudden, generating large dynamic forces.

6.2. Summary of HRAB Concept and Testing

A shelter concept was developed around the semi-circular airbeam arch utilizing folding sandwich composite wall panels. The shelter comes is envisioned to be made from 2.4 m long segments allowing for the shelter to be expanded or shortened based on need. The airbeam arch drives the erection of the shelter by exerting forces on the wall panels during inflation of the airbeam. An experiment was conducted to analyze the feasibility of using the inflation of the airbeam to drive the erection of the shelter. This experiment utilized the same airbeams that were used to determine the lifting capacity of an individual arch. The arch was pinned to the ground with the folding wall panels directly next to the arches footings. Wall panel frames were made of 2x4’s and attached to the airbeam through a wooden strut and rope representing roof fabric.

During inflation of the mechanism, one wall panel would open before the other, and in most tests the panels opened before the inflation pressure reached 110 kPa. Occasionally the second wall panel would open immediately after the first due to forces exerted in the mechanism from the opening of the first wall panel. Testing revealed that the arches must be braced against out of plane motion. It was also necessary to stiffen the fabric of the airbeam at the connection point with the strut as this point load on the airbeam would induce a hinge in the fabric. As with the dynamic arch inflation tests, the rise of the airbeam was very sudden generating large impact forces on the wall panels as well as the restraining guy lines.
6.3. Summary of HRAB Modeling

A finite element model of the HRAB shelter was generated using PressArchAnalysis, a nonlinear finite-element package for the analysis of inflated fabric arches and beams. New elements were added to PressArchAnalysis to simulate different components of the shelter. The shelter was subjected to a uniform gravity load which was applied to the horizontal projection of the element. Using realistic properties for the arches, walls and other components of the HRAB shelter, its capacity was determined for internal airbeam pressures of 137 kPa to 344 kPa. It was found that the gravity load capacity of the shelter increased nearly linearly with pressure, and is in general slightly larger than an equivalent TEMPER shelter.

6.4. Conclusions

The objective of quantifying the lifting capacity of a TEMPER arch was partially completed since arches rather than full semi-circular arches were used. However, it was shown that the airbeam arches are able to lift weights attached to their apex from a deflated state while braced out of plane that are significantly larger than the arch’s own self weight. A new shelter concept, Hybrid Rigid Airbeam (HRAB) shelter, was developed that incorporates the benefits of both rigid wall and soft wall shelters. This was done by adding rigid folding panels to the exterior of a semi-circular airbeam which is used to drive the erection of the panel through its own inflation. This mechanism was proved feasible through repeated experimental testing utilizing a full-scale version of the mechanism. The HRAB concept was then modeled using the finite element analysis software PressArchAnalysis, which was modified to include the components of the HRAB concept. Modelling of the concept showed that the shelter, in addition to its rigid walls and increase in interior volume, had a 0.9% to 9.1% larger load capacity than that of the equivalent TEMPER shelter.
6.5. Recommendations for Future work

During this research, topics arose that could be explored in future work to further the HRAB concept as well as the understanding of the airbeam arch load capacity. These topics are listed below:

- **Exploration of the different lifting shapes of airbeam arches and their effect.** The shape that the arch takes during inflation was shown to have a large effect on the lifting capacity. Different arch shapes were observed during this research with some showing much higher capacities than others. A study of these different shapes would further the understanding of their effect.

- **Testing of the lifting capacity of the airbeam arch in configurations other than midspan loading.** Determining the capacity of the airbeam arch under different loading scenarios could lead to other applications. These include lifting from two points rather than one such as at points at 1/3 arch length.

- **Testing of full semi-circular airbeam arch lifting capacity at midspan.** Tests should be conducted to determine the lifting capacity of the airbeam arch as arch tests presented in this paper were for reduced span arches.

- **Testing of airbeam arches and beams to determine their properties related to dynamic loads.** It was observed in Chapter 3 that the airbeam behaves as a damped oscillator once the airbeam reached its full rise. Further studies could be conducted to define the airbeam’s properties relating to free vibration and other load excitations.

- **Study the resistance of the airbeam to local point loads.** The connection between the strut and the airbeam in the HRAB structure should be further investigated. The point loads generated cause hinges in the airbeam. It should be investigated whether stiffening of the airbeam fabric is required to maintain functionality of the HRAB structure or how much reduction in capacity is expected with no additional stiffness applied to the fabric.
Further design of the HRAB concept. Further design of the HRAB concept is necessary in order to develop the shelter for use in the field. The connection of the airbeam and wall panel into a single unit that can be shipped, deployed and struck with a reasonable number of personnel in a short period of time must be explored. This will require that the sandwich composite wall panel design be optimized and that the panels incorporate a simple, reliable internal hinge and locking mechanism. This locking mechanism could be D-ring straps or latching clamps used in the experimental testing of Chapter 4. A method of safely and rapidly packing the entire structure must be developed. Further, if the HRAB is to be modular as envisioned here, a watertight and simple method of connecting adjacent segments must be developed.

Further testing of the HRAB concept with more representative conditions. Experimental testing of the HRAB concept can be furthered in ways that better represent its in-field conditions. This includes using sandwich composite wall panels, incorporating the connection of the wall panel and airbeam to the ground, adding a representative roof fabric, and testing of multiple, connected HRAB modules. It is also possible that the strut connecting the wall and airbeam is not needed, and this could be explored in these more representative tests.

Refinement of load application in the HRAB model in PAA. The PressArchAnalysis models used here assumed a uniform gravity load. However, a critical load condition for in-service TEMPER tents and other structures is snow loading, which is not uniform and varies with angle of the arch from horizontal. Because of this, the HRAB may be required to carry more snow load than a TEMPER tent, making the results of this study somewhat un-conservative. Additional analyses under more realistic snow load should be conducted to assess this. Additionally, the solver should be improved to allow snow load to be placed directly on the roof membrane.

Incorporate contact conditions in the HRAB PAA model. The studies reported here did not incorporate contact, and indeed the arch was predicted to pass through the wall near peak load
as the arch underwent sidesway. Incorporating contact conditions would prevent this. Similarly, if gravity loads are applied to the roof membrane, it will likely be more accurate to incorporate contact conditions between the roof and arch.

- **More accurate modeling of HRAB arch and component geometry.** PressArchAnalysis models all elements as connected at centerlines. While reasonable for a stand-alone arch, a more accurate approach would be to use rigid offsets to more accurately model the position of the struts, wall panels, roof membrane, and guy line relative to the CG of the arch cross-section.
BIBLIOGRAPHY


Tom Artes, 2014. 32FT Wide Airbeam.


Figure 91: 137 kPa, 0 N Guy Pretension, 45° Guy Angle, HZ Load
Figure 92: 137 kPa, 200 N Guy Pretension, 45° Guy Angle, HZ Load
Figure 93: 206 kPa, 0 N Guy Pretension, 45° Guy Angle, HZ Load
Figure 94: 206 kPa, 200 N Guy Pretension, 45° Guy Angle, HZ Load
Figure 95: 275 kPa, 0 N Guy Pretension, 45° Guy Angle, HZ Load
Figure 96: 275 kPa, 200 N Guy Pretension, 45° Guy Angle, HZ Load
Figure 97: 344 kPa, 0 N Guy Pretension, 45° Guy Angle, HZ Load
Figure 98: 344 kPa, 200 N Guy Pretension, 45° Guy Angle, HZ Load
Figure 99: 137 kPa, 0 N Guy Pretension, 45° Guy Angle, No HZ Load
Figure 100: 137 kPa, 200 N Guy Pretension, 45° Guy Angle, No HZ Load
Figure 101: 206 kPa, 0 N Guy Pretension, 45° Guy Angle, No HZ Load
Figure 102: 206 kPa, 200 N Guy Pretension, 45° Guy Angle, No HZ Load
Figure 103: 275 kPa, 0 N Guy Pretension, 45° Guy Angle, No HZ Load
Figure 104: 275 kPa, 200 N Guy Pretension, 45° Guy Angle, No HZ Load
Figure 105: 344 kPa, 0 N Guy Pretension, 45° Guy Angle, No HZ Load
Figure 106: 344 kPa, 200 N Guy Pretension, 45° Guy Angle, No HZ Load
APPENDIX B: HRAB ANALYSIS USER GUIDE AND CODE

HRAB analysis is an update to PressArchAnalysis which manipulates the meshing and analysis steps of PressArchAnalysis. To run, the user must add the code presented below to the ‘all source code’ folder of PressArchAnalysis and ensure that the ‘all source code’ folder is added to the Matlab path.

**Single Model**

Much of the process behind the code is discussed in chapter 5. Here the process of running a single model will be discussed.

1. Create a new folder which will hold all of the output files. Here this folder is call ‘Test Folder’
2. Use PressArchAnalysis to generate input files for the model, or use ones that have already been created. Place these files in the folder ‘Test Folder\Original Input Files’ in the ‘Test Folder’. Currently, the airbeam must have 60 elements and two guy lines. The .dat file and .paa must be renamed in the following format: internal airbeam pressure in psi, underscore, guy line pretension in Newtons, underscore, guy line angle with the horizontal in degrees, underscore, and spacing of the airbeams in feet.
   a. For example the following ‘30_100_70_8.dat’ indicates a model where the airbeam is inflated to 30 psi, the guy lines have a pretension of 100 N, the guy line create a 70° angle with the ground at their anchor points, and finally the airbeams have a spacing of 8 ft center to center.
3. Create an HRAB props file by copying an existing one and adjust the numerical values. This file must be name in similar fashion to the other input files. Continuing the example above, the HRAB props file would be named ‘HRAB_Props_30_100_70_8.dat’. This file must be in the ‘Test Folder’. The fields in the file are:
   a. Walls
      i. Self weight, N/mm along length of wall
      ii. Number of elements in each wall
      iii. Height, mm, total height of wall
      iv. Elastic Modulus, MPa
      v. Cross-sectional Area, mm²
      vi. Moment of Inertia, mm⁴
      vii. Shear Modulus, MPa
      viii. Uniform Gravity Load, N/mm along the length of the wall
   b. Roof
      i. Self-weight, oz/yd²
      ii. Number of elements in each roof component
      iii. Elastic Modulus, N/mm
      iv. Left roof attachment node. Airbeam node which the left roof attaches.
      v. Right roof attachment node. Airbeam node which the right roof attaches.
      vi. Uniform Gravity load, Pa
c. Strut
   i. Self-weight, oz/yd\(^2\)
   ii. Number of elements in each strut component
   iii. Elastic Modulus, MPa
   iv. Cross-sectional Area, mm\(^2\)
   v. Moment of Inertia, mm\(^4\)
   vi. Shear Modulus, MPa
   vii. Left roof attachment node. Airbeam node which the left strut attaches.
   viii. Right roof attachment node. Airbeam node which the right strut attaches.
   ix. Uniform Gravity load along the length of the strut, N/mm

d. Additional Model Parameters
   i. Horizontal load percentage applied to all gravity loaded nodes

4. Once these files have been created, set the MATLAB working folder to the ‘Test Folder’. Then run ‘HRAB_ANALYSIS.m’ with the input to the function being name of the input file. Continuing our example it would be HRAB_ANALYSIS(‘30_100_70_8’).

5. The analysis process will begin and generate the output files that PressArchAnalysis generates as well as
   a. Picture files of:
      i. The discretized model with the gravity loads applied to the nodes, not including self weight.
      ii. The deflections of the model for the last loadstep
      iii. Axial, Shear, and moment diagrams for all HRAB components

**Multiple Models**

HRAB analysis can also be run in a manner that compares the HRAB shelter to its equivalent TEMPER shelter. This may be easier to use for some users as no files need to be modified since the geometry of the TEMPER shelter is set and the other inputs are values that are changed in the driver function ‘HRAB_ANALYSIS_DRIVER’. The steps to run this function are shown below:

1. Create a folder, for example ‘Test Folder’ and place the input files ‘20psi_20ft.dat’, ‘20psi_20ft.paa’, and ‘HRAB_Props.dat’ in the ‘Test Folder’.
2. Open the ‘HRAB_ANALYSIS_DRIVER’ function and adjust the inputs in lines 8 through 11.
   a. HRAB, 1 or 0, indicates whether to apply the chosen parameters to a HRAB shelter
   b. Airbeam, 1 or 0, indicates whether to apply the chosen parameter to a TEMPER shelter
   c. Hz, 1 or 0, indicates whether to apply a horizontal load to the shelters.
   d. Optimize, 1 or 0, indicates whether to run the analysis in an optimizing manner such that the final load applied in the input files allows the shelter to reach a peak load fraction of 98%.

3. Adjust the analysis Parameters in lines 15 through 22. Each of these variables you are able to enter multiple values. The function will then create a model for each combination of parameters.
a. Param.PT, N, guy line pretension(s)
b. Param.P_name, psi, internal pressure(s) of the airbeam
c. Param.Angle, degrees, angle(s) of guy line with the ground at the anchor points of the guys
d. Param.Wall_height, mm, height(s) of walls
e. Param.Spacing_name, ft, center to center spacing(s) of airbeams
f. Loads_name, psf, gravity load(s) applied to the shelter(s). If the Optimize value is set to 1, multiple loads will not yield different results for each set of parameters.

4. With the parameters set, run ‘HRAB_Analysis_Driver’. This process uses the parallel computing tool box. If that is not installed, change all instances of parfor with for. The function will create separate folders for each applied load value as well as the horizontally loaded HRAB and TEMPER models.

5. When all models have been run, the driver function will go through and generate summary plots and check equilibrium for all models. Plot created for each model are:
   a. Image files of the following are created:
      i. Load Fraction vs Apex Displacement
      ii. Applied Load vs Apex Displacement
      iii. Internal Moment/Wrinkling Moment ratio along the arc length of the airbeam
      iv. Internal Moment along the arc length of the airbeam
      v. Summary image of the results which includes the above images, (i or ii is chosen depending on whether an optimization was run)

6. A comparison figure is created in the ‘Test Folder’ comparing the maximum load or load fraction achieved for each of the sets of models.

The following section contains all of the code used to run single and multiple models. It is listed alphabetically.
function [OK, fraction_incr] = arc_length_solver_HRAB(initial_displ,...
    initial_fraction, alpha);

global BASIC_PROPS
global NODES
global CONNECTIVITIES
global PROPS
global F
global TOTAL_F;
global DL_F;
global K
global K_BC
global U
global R
global BOUNDARIES
global M_kappa;
global MEMBER_FORCES;
global CAPACITY_DATA;
global STRESSES;
global GUYS;
global GUY_FORCES;
global REACTIONS;
global P_V_M;
global LARGE_DEF;
global UPDATED_NODES;
global SOLVER_TOLERANCE;

% we must initialize U here if not specified
ndof = size(NODES,1)*3;
if (size(initial_displ,1) < ndof)
    U = zeros(ndof,1);
else
    U = initial_displ;
end

old_updated_nodes = UPDATED_NODES;

% our convergence criterion
%tol = 5e-04;
error = 1;

count = 1;

% start arc length iteration
max_iter = 25;

warning('off');

% choose DOF as corresponding to max displ.
%ind = find(abs(initial_displ) == max(abs(initial_displ)));
% solve for initial load factor with Euler update
% form K_BC
assemble_stiff_HRAB;
apply_boundaries;
dummy_U = K_BC\TOTAL_F;
del_l = alpha*norm(initial_displ);
fraction_incr = del_l/(norm(dummy_U));
% get eigenvalue of symmetric part -- necessary for large curvatures due to
% the unsymmetric portion of K
[R,p] = chol(0.5*(K_BC + K_BC'));
if (p > 1e-3)
    fraction_incr = -fraction_incr;
end
del_U = fraction_incr*dummy_U;
U = initial_displ + del_U;
a = del_U'*del_U - del_l^2;
%initial_fraction
force_fraction = initial_fraction + fraction_incr;
F = DL_F + force_fraction*TOTAL_F;

% update coordinates if performing a large deformation analysis
if (LARGE_DEF == 1)
    update_nodes(U);
end
% compute member forces
curvature_error = compute_member_forces_HRAB;
% compute the residual force vector, R
compute_residual;
	error = norm(R)/norm(F);
error = 1;
s = sprintf('      error = %g', error);
disp(s);

SOLVER_TOLERANCE = 0.005;
while (error > SOLVER_TOLERANCE) && (count <= max_iter)

    % form K_BC
    assemble_stiff_HRAB;
    apply_boundaries;

    % form augmented system
    stiff = [K_BC -TOTAL_F;
             -2*del_U' 0];
    RHS = [R; a];

    old_R = R;

    % solve for LHS
\texttt{LHS = stiff}]*RHS; \\
\texttt{del\_U\_incr = LHS(1:ndof,1);} \\
\texttt{del\_fraction\_incr = LHS(ndof+1,1);} \\

\% increment the displacements, fraction increment, force fraction \\
\texttt{del\_U = del\_U + del\_U\_incr;} \\
\texttt{U = initial\_displ + del\_U;} \\
\texttt{fraction\_incr = fraction\_incr + del\_fraction\_incr;} \\
\texttt{force\_fraction = initial\_fraction + fraction\_incr;} \\
\texttt{F = DL\_F + force\_fraction*TOTAL\_F;} \\
\texttt{a = del\_U'*del\_U - del\_l^2;} \\

\% update coordinates if performing a large deformation analysis \\
\texttt{if (LARGE\_DEF == 1)} \\
\texttt{update\_nodes(U);} \\
\texttt{end} \\

\% compute member forces \\
\texttt{curvature\_error = compute\_member\_forces\_HRAB;} \\

\% compute the residual force vector, R \\
\texttt{compute\_residual;} \\

\% convergence is based on force equilibrium \\
\texttt{if (norm(F) > 1e-6)} \\
\texttt{error = norm(R)/norm(F);} \\
\texttt{else} \\
\texttt{\hspace{1em}error = norm(R)/norm(init\_R);} \\
\texttt{\hspace{1em}error = norm(R);} \\
\texttt{end} \\

\%if (count == 1) \\
\% \hspace{1em}init\_energy = del\_U\_incr'*old\_R; \\
\% \hspace{1em}error = 1; \\
\%else \\
\% \hspace{1em}error = del\_U\_incr'*old\_R/init\_energy; \\
\%end \\

\texttt{s = sprintf(' error = \%g', error);} \\
\texttt{s = sprintf(' error = \%g, curvature error = \%g', [error, curvature\_error]);} \\
\texttt{disp(s);} \\
\texttt{count = count + 1;} \\
\texttt{end} \\

\% check for change in sign of hz. displacement at apex, discard solution if \\
\% this is the case. \\
\%center\_node = (size(NODES, 1) - 1)/2 + 1 \\
\%old\_center\_displ = initial\_displ(center\_node*3 - 2) \\
\%new\_center\_displ = U(center\_node*3 - 2) \\
\%if (old\_center\_displ*new\_center\_displ < 0) \\
\% \hspace{1em}error = 2*SOLVER\_TOLERANCE; \\
\%end \\
\%pause
OK = 1;
woven = BASIC_PROPS(9);
max_displ = get_max_displ;

if (error > SOLVER_TOLERANCE) || isnan(error) % we have not converged, do not accept
    OK = 0;
    MEMBER_FORCES = zeros(size(CONNECTIVITIES,1),6);
    STRESSES = zeros(size(NODES,1),1);
    CAPACITY_DATA = zeros(size(NODES,1),3);
    message = sprintf(' Did not converge in %g iterations, error = %0.3g',
        count-1, error);
    disp(message);
    % here we must go back to our old node locations
    U = initial_displ;
    update_nodes(U);
else % converged, compute stresses, compute capacity data
    %max_wrinkle HRAB; % max tensile stress at each node, capacity data
    max_wrinkle = max(CAPACITY_DATA(:,3));
    %d = BASIC_PROPS(3);
    %if (max_wrinkle < d/1000)
    %    max_wrinkle = 0;
    %end
    %max_displ = get_max_displ;
    if (woven == 1)
        message = sprintf(' No. of iterations = %g, error = %0.3g, max. displ = %0.1f, max.
            wrinkle = %0.1f',
            count-1, error, max_displ, max_wrinkle);
        else
            message = sprintf(' No. of iterations = %g, error = %0.3g, max. displ = %0.1f',
                count-1, error, max_displ);
        end
        disp(message);
    end
warning('on');
end
function assemble_stiff_HRAB

% This function assembles the system stiffness matrix, K

global M_kappa
global NODES
global CONNECTIVITIES
global PROPS
global K
global F
global U
global BOUNDARIES
global MEMBER_FORCES
global GUYS
global LARGE_DEF
global GUY_FORCES
global LARGE_CURVATURE
global SPRING_SUPPORTS
global HRAB_WALL
global HRAB_STRUT
global HRAB_ELE_I

num_nodes = size(NODES,1);
K = sparse(num_nodes*3, num_nodes*3);

% get the number of elements
numels = size(CONNECTIVITIES,1);

% loop over all of the elements
for (i = 1:numels)
    elcon = CONNECTIVITIES(i,:);
    elprop = PROPS(i,:);
    eldispl = get_eldispl(elcon, elprop);

    % % get the element stiffness
    % el_stiff = element_stiffness_linear_axial(elprop, eldispl, ~MEMBER_FORCES(i,1));
    % get the element stiffness
    if HRAB_ELE_I(i) == 1 %Airbeam Element
        el_stiff = element_stiffness(elprop, eldispl, ~MEMBER_FORCES(i,1));
    elseif HRAB_ELE_I(i)==2 % Linear Elastic wall element
        EI = HRAB_WALL.E*HRAB_WALL.I;
        el_stiff = element_stiffness_linear_axial_HRAB(elprop, EI, ~MEMBER_FORCES(i,1));
    elseif HRAB_ELE_I(i)==3 % Linear Elastic Strut element
        EI = HRAB_STRUT.E*HRAB_STRUT.I;
        el_stiff = element_stiffness_linear_axial_HRAB(elprop, EI, ~MEMBER_FORCES(i,1));
    elseif HRAB_ELE_I(i)==4 %Linear Elastic wall element with hinge at the end
        EI = HRAB_WALL.E*HRAB_WALL.I;
        el_stiff = element_stiffness_linear_axial_hinge_end(elprop, EI, ~MEMBER_FORCES(i,1));
    elseif HRAB_ELE_I(i)==10 % Linear Elastic Wall Element with hinge at start
        EI = HRAB_WALL.E*HRAB_WALL.I;
        el_stiff = element_stiffness_linear_axial_hinge_start(elprop, EI, ~MEMBER_FORCES(i,1));
    elseif HRAB_ELE_I(i)==5 %Linear Elastic Strut element with hinge at the end
        EI = HRAB_STRUT.E*HRAB_STRUT.I;
        el_stiff = element_stiffness_linear_axial_hinge_start(elprop, EI, ~MEMBER_FORCES(i,1));
    else
        % handle other element types
    end

    % assemble the stiffness matrix, K
    K(elcon([1 2 3]), elcon([1 2 3])) = el_stiff;
end

end
EI = HRAB_STRUT.E*HRAB_STRUT.I;
el_stiff = element_stiffness_linear_axial_hinge_end(elprop, EI, -MEMBER_FORCES(i,1));
end

% include element geometric stiffness if small deformation analysis
if (LARGE_DEF == 0)
el_geom_K = element_geometric_stiffness(elprop, MEMBER_FORCES(i,1));
el_stiff = el_stiff - el_geom_K;
end

% transform
T = get_transform(elcon, elprop);
el_stiff = T'*el_stiff*T;

% now, for large deformations get additional geometric stiffness
if (LARGE_DEF == 1)
el_stiff = el_stiff + element_large_def_stiffness(elprop, elcon, ...MEMBER_FORCES(i,:));
end

% add the element stiffness to global K
add_element_k(elcon, el_stiff);
end

% loop over all guy lines
numguys = size(GUYS,1);
for (i = 1:numguys)
elprop = GUYS(i,:);
eldispl = get_guydispl(elprop);
T = generate_T_guybeam(elprop);
eldispl = T*eldispl;

% Now we must add the initial displacement -- make it a positive value
% at end node. This yields the guy pre-tensioning.
eldispl(4) = eldispl(4) + elprop(5);

% get the element stiffness
el_stiff = guy_stiffness(elprop, eldispl);
el_stiff = T'*el_stiff*T;

if (LARGE_DEF == 1) % geometric stiffness matrix for guys
    member_forces = GUY_FORCES(i,:);
el_stiff = el_stiff + guy_large_def_stiffness(elprop, member_forces);
end

% add the element stiffness to global K
add_element_k([elprop(1) elprop(2)], el_stiff);
end

% now, support springs
% now, add reactions at spring supports
num_springs = size(SPRING_SUPPORTS,1);
for (i = 1:num_springs)
    node = SPRING_SUPPORTS(i,1);
    kx = SPRING_SUPPORTS(i,2);
    ky = SPRING_SUPPORTS(i,3);
    x_dof = node*3 - 2;
    y_dof = node*3 - 1;
    K(x_dof, x_dof) = K(x_dof, x_dof) + kx;
    K(y_dof, y_dof) = K(y_dof, y_dof) + ky;
end
end
function curvature_error = compute_member_forces_HRAB

global BASIC_PROPS
global M_kappa
global NODES
global CONNECTIVITIES
global PROPS
global K
global K_BC
global F
global U
global BOUNDARIES
global MEMBER_FORCES
global GUYS
global GUY_FORCES
global LARGE_DEF
global UPDATED_NODES
global LARGE_CURVATURE
global HRAB_ELE_I
global HRAB_WALL
global HRAB_STRUT
global GUY_TYPE

% get the number of elements
numels = size(CONNECTIVITIES,1);

MEMBER_FORCES = zeros(numels, 10);

woven = BASIC_PROPS(9);

curvature_error = 0;

old_diff = 0;

% loop over all of the elements
for (i = 1:numels)
    elcon = CONNECTIVITIES(i,:);
    elprop = PROPS(i,:);

    % original, undeformed length used in curvature calculation
    Lo = elprop(3);

    % final length needed for large deflection analysis
    Ln = elprop(4);

    % we need the element displacements and strain
    % these next two routines work for large and small deformations
    el_displ = get_eldispl(elcon, elprop);
    T = get_transform(elcon, elprop);
    el_displ = T*el_displ;

    % compute member end axial and shear forces -- works for both large and
% small deformations
el_force = axial_shear_end_forces(elprop, el_displ);

% now, compute the end bending moments from curvature and shears
curvature = -el_displ(3)/Lo + el_displ(6)/Lo;

% compute exact curvature and error -- assumes constant shear
local_slope = (el_displ(5) - el_displ(2))/Lo;
exact_curvature = curvature/((1+local_slope^2)^1.5);
err = 0;
if (abs(exact_curvature) > 0)
    err = (curvature - exact_curvature)/exact_curvature;
end
diff = 3*local_slope^2/(1+local_slope^2)^0.5;
%curvature_error = max(abs(local_slope), abs(old_diff));
%old_diff = local_slope;
curvature_error = max(old_diff, diff);
old_diff = diff;

if (LARGE_CURVATURE == 1)
    small_curvature = curvature;
curvature = exact_curvature;
    y_bar = interp1(final_M_kappa(1,:,1), final_M_kappa(1,:,4), abs(curvature));
end

final_M_kappa = interp_M_kappa(-el_force(1));
if (woven == 1) & HRAB_ELE_I(i) == 1  % symmetric, use absolute values
    % interpolate M_kappa
    M = sign(curvature)*interp1(final_M_kappa(1,:,1), final_M_kappa(1,:,2), abs(curvature));
elseif woven == 2 & HRAB_ELE_I(i) == 1  % braided, potentially unsymmetric, -kappa implies +M
    %M = get_braided_moment(curvature, -el_force(1));
    %if (curvature > max(final_M_kappa(1,:,1))) || isnan(curvature)
    %    curvature = max(final_M_kappa(1,:,1));
    %end
    M = interp1(final_M_kappa(1,:,1), final_M_kappa(1,:,2), curvature, ...
        'linear', 'extrap');
elseif HRAB_ELE_I(i)==2  % Linear Elastic wall element
    EI = HRAB_WALL.E*HRAB_WALL.I;
    M = EI*curvature;
% Find stiffness matrix for use down below in determine member forces
    el_stiff = element_stiffness_linear_axial_HRAB(elprop, EI, -MEMBER_FORCES(i,1));
elseif HRAB_ELE_I(i)==3  %Linear Elastic strut element
    EI = HRAB_STRUT.E*HRAB_STRUT.I;
    M = EI*curvature;
    el_stiff = element_stiffness_linear_axial_HRAB(elprop, EI, -MEMBER_FORCES(i,1));
elseif HRAB_ELE_I(i)==4  % Linear Elastic Wall Element with hinge at end
    EI = HRAB_WALL.E*HRAB_WALL.I;
    M = EI*curvature;
    el_stiff = element_stiffness_linear_axial_hinge_end(elprop, EI, -MEMBER_FORCES(i,1));
elseif HRAB_ELE_I(i)==10  % Linear Elastic Wall Element with hinge at start
    EI = HRAB_WALL.E*HRAB_WALL.I;
    M = EI*curvature;
```matlab
el_stiff = element_stiffness_linear_axial_hinge_start(elprop, EI, MEMBER_FORCES(i,1));
elseif HRAB_ELE_I(i) == 5  % Linear Elastic Strut Element with hinge at end
    EI = HRAB_STRUT.E*HRAB_STRUT.I;
    M = EI*curvature;
    % Find stiffness matrix for use down below in determine member forces
    el_stiff = element_stiffness_linear_axial_hinge_end(elprop, EI, MEMBER_FORCES(i,1));
end

% balance moments and shears, taking previously computed moment as an
% average value and assuming a linear variation in moment
L = Lo;
V2 = el_force(5);  % shear at node 2
el_force(3,1) = -V2*L/2 - M;
el_force(6,1) = M - V2*L/2;

% Account for hinged elements, el_displ(6)=0 because hinge therefore
% curvature calculation is bad. Calculate el_force normally
% Displacements have already been converted to element coordinates on
% line 48
if HRAB_ELE_I(i) == 4 || HRAB_ELE_I(i) == 5 || HRAB_ELE_I(i) == 10  % Linear Elastic element with hinge at end
    el_stiff = element_stiffness_linear_axial_hinge_end_01(elprop, el_displ, MEMBER_FORCES(i,1));
    el_force = el_stiff*el_displ;  % in element coordinates
end

% now, we must include geometric stiffness effects in member forces if
% we are solving the small deformation problem
if (LARGE_DEF == 0)
    el_geom_stiff = element_geometric_stiffness(elprop, el_force(1));
    geom_force = -el_geom_stiff*el_displ;
    el_force = el_force + geom_force;
end

% now, assign to array
MEMBER_FORCES(i,1:9) = el_force';
% put the curvature in the last column
MEMBER_FORCES(i,10) = curvature;
end

% loop over all guy lines
numguys = size(GUYS,1);
GUY_FORCES = zeros(numguys,6);
for (i = 1:numguys)
    % GUYS = [node1, node2, l_o, EA, pre-tension, l_n];
    elprop = GUYS(i,:);
    Lo = elprop(3);

    % Works for large and small deflections.
    eldispl = get_guydispl(elprop);

    % Works for large and small deflections.
    T = generate_T_guy_beam(elprop);
```

eldispl = T*eldispl;

% Now we must add the initial displacement -- make it a positive value
% at end node. This yields the guy pre-tensioning.
eldispl(4) = eldispl(4) + elprop(5);

% Get the element stiffness. Works for large and small deflections
el_stiff = guy_stiffness(elprop, eldispl);

% Linearly elastic, even though tension-only. This line works for large
% and small deformations as far as shears and moments.
el_force = el_stiff*eldispl;

% now, compute axial force accouting for small or large deformations
if (LARGE_DEF == 1)
    delta_L = elprop(6) - elprop(3) + elprop(5);
eq = elprop(4);
    if (delta_L < 0) % it is in compression
        eq = eq*1e-8;
    end
    P = (delta_L/Lo)*eq;
el_force(1,1) = -P;
el_force(4,1) = P;
else
    % small deformations -- dirty trick that allows the element
    % stiffness matrix to use a larger EA than used to compute member
    % forces for compressive response
    delta_L = eldispl(4) - eldispl(1);
eq = elprop(4);
    if (delta_L < 0) % we are in compression, must modify eq
        eq = eq*1e-8;
    end
    P = (delta_L/Lo)*eq;
el_force(1,1) = -P;
el_force(4,1) = P;
end

% add to guy forces
GUY_FORCES(i,:) = el_force';
end
end
function [LC] = Driver_no_opti_parallel(HRAB)
%Driver script to run the HRAB analysis for every folder that is properly
%formatted with input files that were generated by the
%InputFileGenerator.m. If the parallel computing toolbox is not available
%replace the 'parfor' with a normal for loop

% Generate structure with info on the contents of the folder
cont=dir;
% Eliminate the two blank rows at the beginning of the structure
cont(1:2,:)=[];
% Create indices of the model folders
folders=[cont.isdir];
folder_index=find(folders==1)';

% Loop over each of the model folders and gather the final load fraction
% applied and the resulting displacement
parfor i=1:size(folder_index,1)
    home=cd(cont(folder_index(i)).name);
    if HRAB==1
        % Perform an HRAB Analysis
        [out]=HRAB_ANALYSIS(cont(folder_index(i)).name);
    else
        % Perform a PAA Analysis
        [out]=PAA_ANALYSIS(cont(folder_index(i)).name);
    end
    cd(home);
    LC(i,:)=out;
end
save('filesave.mat','LC')
end
function [] = Driver_parallel(HRAB,Hz)
% Driver script to run the HRAB analysis for every folder that is properly
% formatted with input files that were generated by the
% InputFileGenerator.m. This function uses the parallel computing toolbox.
% If that is not available, replace the parfor with a normal for loop.

% Generate structure with info on the contents of the folder
cont=dir;
% Eliminate the two blank rows at the beginning of the structure
cont(1:2,:)=[];
% Create indices of the model folders
folders=[cont.isdir];
folder_index=find(folders==1)';

Load=zeros(size(folder_index,1),2);
% Loop over each of the model folders and gather the final load fraction
% applied and the resulting displacement
parfor i=1:size(folder_index,1)
    home=cd(cont(folder_index(i)).name);
    % Run Load Optimization code
    [out]=Load_Optimize_BisectionMethod(HRAB,Hz,cont(folder_index(i)).name);
    Load(i)=out;
    cd(home);
end
save('filesave.mat','Load')
end
function K = element_stiffness_linear_axial_hinge_end(props, EI, axial_load)
% Modified by Jay Wegner 2018/12/12. This element is similar to Beam 189 in
% the ANSYS Element Reference Document.
% Element is a 3 noded Timoshenko Beam element with quadratic shape
% functions for local V displacement and linear shape functions for local
% theta rotations.

% This routine returns the tangent element stiffness matrix given:
% props = 3-element vector with EA, GA and L
% el_displ = 9-element vector of element displacements
% axial_load = element axial force, positive for tension
% M_kappa = matrix with curvatures, moments, EI

% global M_kappa;
% global BASIC_PROPS

% first, evaluate curvature at center of element, get corresponding EI
L = props(3);

% % This is a linear element.
% curvature = -el_displ(3)/L + el_displ(6)/L;
% % interpolate M_kappa
% final_M_kappa = interp_M_kappa(axial_load);
% % EI = interp1(final_M_kappa(1,:,1), final_M_kappa(1,:,3), abs(curvature));

% other necessary properties
EA = props(1);
GA = props(2);
% EI = input
% disp(EI)
% form element K
K(1,1) = 7*EA/(3*L);
K(1,4) = EA/(3*L);
K(1,7) = -8*EA/(3*L);
K(2,2) = 7*GA/(3*L);
K(2,3) = 5*GA/6;
K(2,5) = GA/(3*L);
K(2,6) = GA/6;
K(2,8) = -8*GA/(3*L);
K(3,2) = K(2,3);
K(3,3) = EI/L + GA*L/3;
K(3,5) = -GA/6;
K(3,6) = -EI/L + GA*L/6;
K(3,8) = -2*GA/3;
K(4,1) = K(1,4);
K(4,4) = 7*EA/(3*L);
\[ K(4,7) = -8*EA/(3*L); \]
\[ K(5,2) = K(2,5); \]
\[ K(5,3) = K(3,5); \]
\[ K(5,5) = 7*GA/(3*L); \]
\[ K(5,6) = -5*GA/6; \]
\[ K(5,8) = -8*GA/(3*L); \]
\[ K(6,2) = K(2,6); \]
\[ K(6,3) = K(3,6); \]
\[ K(6,5) = K(5,6); \]
\[ K(6,6) = EI/L + GA*L/3; \]
\[ K(6,8) = 2*GA/3; \]
\[ K(7,1) = K(1,7); \]
\[ K(7,4) = K(4,7); \]
\[ K(7,7) = 16*EA/(3*L); \]
\[ K(8,2) = K(2,8); \]
\[ K(8,3) = K(3,8); \]
\[ K(8,5) = K(5,8); \]
\[ K(8,6) = K(6,8); \]
\[ K(8,8) = 16*GA/(3*L); \]
\[ K(9,9) = 1; \]

\[ \text{sig} = EI/L + GA*L/3; \]
\[ R = [1 0 0 0 0 0 0 0; \]
\[ 0 1 0 0 0 0 0 0; \]
\[ 0 0 1 0 0 0 0 0; \]
\[ 0 0 0 1 0 0 0 0; \]
\[ 0 0 0 0 1 0 0 0; \]
\[ 0 0 0 0 0 1 0 0; \]
\[ 0 -GA/(6\text{sig}) (EI/L - GA*L/6)/\text{sig} 0 5*GA/(6\text{sig}) 0 -2*GA/(3\text{sig}) 0; \]
\[ 0 0 0 0 0 1 0 0; \]
\[ 0 0 0 0 0 0 1 0; \]
\[ 0 0 0 0 0 0 0 1]; \]
\[ \text{K}_r = R^T*K*R; \quad \% \text{this is an 8x8} \]

\% expand to 9x9 with one on diagonal in K(6,6)
\[ K = \text{zeros}(9,9); \]
\[ K(1:5, 1:5) = \text{K}_r(1:5, 1:5); \]
\[ K(1:5, 7:9) = \text{K}_r(1:5, 6:8); \]
\[ K(7:9, 1:5) = \text{K}_r(6:8, 1:5); \]
\[ K(7:9, 7:9) = \text{K}_r(6:8, 6:8); \]
\[ K(6,6) = 1; \]
\[ \text{end} \]
function K = element_stiffness_linear_axial_hinge_start(props, EI, axial_load)
% Modified by Jay Wegner 2018/12/12. This element is similar to Beam 189 in
% the ANSYS Element Reference Document.
% Element is a 3 noded Timoshenko Beam element with quadratic shape
% functions for local V displacement and linear shape functions for local
% theta rotations.

% This routine returns the tangent element stiffness matrix given:
% props = 3-element vector with EA, GA and L
% el_displ = 9-element vector of element displacements
% axial_load = element axial force, positive for tension
% M_kappa = matrix with curvatures, moments, EI

% global M_kappa;
% global BASIC_PROPS

% first, evaluate curvature at center of element, get corresponding EI
L = props(3);

% This is a linear element.
curvature = -el_displ(3)/L + el_displ(6)/L;
% % interpolate M_kappa
% final_M_kappa = interp_M_kappa(axial_load);
% % EI = interp1(final_M_kappa(1,:,1), final_M_kappa(1,:,3), abs(curvature));

% other necessary properties
EA = props(1);
GA = props(2);
% EI = input
% disp(EI)

% form element K
K(1,1) = 7*EA/(3*L);
K(1,4) = EA/(3*L);
K(1,7) = -8*EA/(3*L);
K(2,2) = 7*GA/(3*L);
K(2,3) = 5*GA/6;
K(2,5) = GA/(3*L);
K(2,6) = GA/6;
K(2,8) = -8*GA/(3*L);
K(3,2) = K(2,3);
K(3,3) = EI/L + GA*L/3;
K(3,5) = -GA/6;
K(3,6) = -EI/L + GA*L/6;
K(3,8) = -2*GA/3;
K(4,1) = K(1,4);
K(4,4) = 7*EA/(3*L);
\[ K(4,7) = -\frac{8EA}{3L}; \]
\[ K(5,2) = K(2,5); \]
\[ K(5,3) = K(3,5); \]
\[ K(5,5) = \frac{7GA}{3L}; \]
\[ K(5,6) = -\frac{5GA}{6}; \]
\[ K(5,8) = -\frac{8GA}{3L}; \]
\[ K(6,2) = K(2,6); \]
\[ K(6,3) = K(3,6); \]
\[ K(6,5) = K(5,6); \]
\[ K(6,6) = \frac{EI}{L} + \frac{GA*L}{3}; \]
\[ K(6,8) = \frac{2GA}{3}; \]
\[ K(7,1) = K(1,7); \]
\[ K(7,4) = K(4,7); \]
\[ K(7,7) = 16\frac{EA}{3L}; \]
\[ K(8,2) = K(2,8); \]
\[ K(8,3) = K(3,8); \]
\[ K(8,5) = K(5,8); \]
\[ K(8,6) = K(6,8); \]
\[ K(8,8) = 16\frac{GA}{3L}; \]
\[ K(9,9) = 1; \]

sig = \frac{EI}{L} + \frac{GA*L}{3};
R = [1 0 0 0 0 0 0 0; 0 1 0 0 0 0 0 0; 0 -\frac{5GA}{(6*sig)} 0 \frac{GA}{(6*sig)} (\frac{EI}{L} - \frac{GA*L}{6})/sig 0 2\frac{GA}{(3*sig)} 0; 0 0 1 0 0 0 0 0; 0 0 0 1 0 0 0 0; 0 0 0 0 1 0 0 0; 0 0 0 0 0 1 0; 0 0 0 0 0 0 1];
K_r = R'*K*R; \% this is an 8x8

% expand to 9x9 with one on diagonal in K(6,6)
K = zeros(9,9);
K_star_I=[1,2,4,5,6,7,8,9];
K(K_star_I,K_star_I)=K_r;
K(3,3) = 1;
end
function K = element_stiffness_linear_axial_HRAB(props, EI, axial_load)

% This routine returns the tangent element stiffness matrix given:
% props = 3-element vector with EA, GA and L
% el_displ = 9-element vector of element displacements
% axial_load = element axial force, positive for tension
% M_kappa = matrix with curvatures, moments, EI

% global M_kappa;

% first, evaluate curvature at center of element, get corresponding EI
L = props(3);
% curvature = -el_displ(3)/L + el_displ(6)/L;
% %
% % % interpolate M_kappa
% final_M_kappa = interp_M_kappa(axial_load);
% %
% EI = interp1(final_M_kappa(1,:,1), final_M_kappa(1,:,3), abs(curvature));

% I=1000;
% E=10000000;
% EI=E*I;
% EI=1.844210834973465E10;

% other necessary properties
EA = props(1);
GA = props(2);
% EI = BEAM_PROPS.E*BEAM_PROPS.I;
% disp(EI)

% disp(EI)
% form element K
K(1,1) = 7*EA/(3*L);
K(1,4) = EA/(3*L);
K(1,7) = -8*EA/(3*L);
K(2,2) = 7*GA/(3*L);
K(2,3) = 5*GA/6;
K(2,5) = GA/(3*L);
K(2,6) = GA/6;
K(2,8) = -8*GA/(3*L);
K(3,2) = K(2,3);
K(3,3) = EI/L + GA*L/3;
K(3,5) = -GA/6;
K(3,6) = -EI/L + GA*L/6;
K(3,8) = -2*GA/3;
K(4,1) = K(1,4);
K(4,4) = 7*EA/(3*L);
K(4,7) = -8*EA/(3*L);
\[ K(5,2) = K(2,5); \]
\[ K(5,3) = K(3,5); \]
\[ K(5,5) = \frac{7GA}{3L}; \]
\[ K(5,6) = -\frac{5GA}{6}; \]
\[ K(5,8) = -\frac{8GA}{3L}; \]
\[ K(6,2) = K(2,6); \]
\[ K(6,3) = K(3,6); \]
\[ K(6,5) = K(5,6); \]
\[ K(6,6) = \frac{EI}{L} + \frac{GA}{3L}; \]
\[ K(6,8) = \frac{2GA}{3}; \]
\[ K(7,1) = K(1,7); \]
\[ K(7,4) = K(4,7); \]
\[ K(7,7) = \frac{16EA}{3L}; \]
\[ K(8,2) = K(2,8); \]
\[ K(8,3) = K(3,8); \]
\[ K(8,5) = K(5,8); \]
\[ K(8,6) = K(6,8); \]
\[ K(8,8) = \frac{16GA}{3L}; \]
\[ K(9,9) = 1; \]
end
function [] = GatherLCs_HRAB_PAA_HZ_01(appload)
%Goes through the 'Solve Models' folder and gathers the name of the model,
%as well as the max load fraction.

close all

Models=dir;
Models(1:2,:)=[];
% appload=15; % load for models that you wish to analyze

a=0;
for i=1:size(Models,1)
    a=a+double(contains(Models(i).name,'outputs'));
end

% Check that the variables have been generated
if a==0
    % Not generated, generate the files
generate_mat_files
end

fig1=figure(1);
ax1=axes(fig1);
hold(ax1,’on’)

% Line Color indicates air pressure
linecolor={'r','g','b','c','m','y','k'};
% Marker Type indicates PreTension
marker={'o','+','*','x','s','d','A','v','>','<','p','h'};
% Line type indicates airbeam or HRAB
linetype={'-','--','-.'};

file=[‘HRAB Hz_’,num2str(appload),’_outputs.mat’];
HRAB_HZ=load(file);

HZ=1;
HRAB=1;

P=HRAB_HZ.P;
PT=HRAB_HZ.PT;
LF=HRAB_HZ.LF;

P_plot=unique(P);
PT_plot=unique(PT);
%

for i=1:size(P_plot,1)
    for j=1:size(PT_plot)
        for k=1:size(HRAB_HZ.Spacing)
            % Guy Pretension vs. Load Fraction
            X=P(P==PT_plot(j))*6.89;
            [X,I]=sort(X);
Y = LF(PT==PT_plot(j));
Y = Y(I);
p1 = plot(ax1, X, Y);
if isempty(p1)~=1
    p1.color = linecolor{i};
    p1.Marker = marker{j};
    p1.LineStyle = linetype{HRAB};
    p1.LineWidth = 2;
    p1.MarkerFaceColor = get(p1, 'Color');
    p1.DisplayName = ['Airbeam, PT= ', num2str(PT_plot(j)), ', N'];
end
end
end

Airbeam_HZ = load(['Airbeam Hz_', num2str(appload), '_outputs.mat']);

HZ = 1;
HRAB = 2;

P = Airbeam_HZ.P;
PT = Airbeam_HZ.PT;
LF = Airbeam_HZ.LF;

P_plot = unique(P);
PT_plot = unique(PT);

% Space_plot = unique(HRAB_HZ.Spacing);
% for i = 1:size(P_plot, 1)
%    for j = 1:size(PT_plot)
%        for k = 1:size(Space_plot, 1)
%            % Guy Pretension vs. Load Fraction
%            X = P(PT==PT_plot(j))*6.89;
%            [X, I] = sort(X);
%            Y = LF(PT==PT_plot(j));
%            Y = Y(I);
%            p1 = plot(ax1, X, Y);
%            if isempty(p1)~=1
%                p1.color = linecolor{i};
%                p1.Marker = marker{j};
%                p1.LineStyle = linetype{HRAB};
%                p1.LineWidth = 2;
%                p1.MarkerFaceColor = get(p1, 'Color');
%                p1.DisplayName = ['Airbeam, PT= ', num2str(PT_plot(j)), ', N'];
%            end
%        end
%    end
%end

xlabel(ax1, 'Internal Pressure (kPa)');
ylabel(ax1, 'Load Fraction');
legend(ax1, 'show', 'Location', 'eastoutside');
grid(ax1, 'on');
box(ax1,'on');

name=\['Internal Pressure vs LF_HZ'\];
print(fig1,name,\'-dmeta\'); %save open figures as explodable .emf files
print(fig1,name,\'-dpng\');
savefig(fig1,name); %save the figure as a Matlab .Fig file

close(fig1);

fig1=figure(1);
ax1=axes(fig1);
hold(ax1,'on')

HRAB_NoHZ=load(['HRAB No Hz_','num2str(appload),'_outputs.mat']);
HZ=2;
HRAB=1;

P=HRAB_NoHZ.P;
PT=HRAB_NoHZ.PT;
LF=HRAB_NoHZ.LF;

P_plot=unique(P);
PT_plot=unique(PT);

% Space_plot=unique(HRAB_HZ.Spacing);

% for i=1:size(P_plot,1)
% for j=1:size(PT_plot)
% for k=1:size(Space_plot,1)
%     % Guy Pretension vs. Load Fraction
%     X=P(PT==PT_plot(j))*6.89;
%     [X,I]=sort(X);
%     Y=LF(PT==PT_plot(j));
%     Y=Y(I);
%     p1=plot(ax1,X,Y);
%     if isempty(p1)~=1
%         p1.Color=linecolor{i};
%         p1.Marker=marker{j};
%         p1.LineStyle=linetype{HRAB};
%         p1.LineWidth=2;
%         p1.MarkerFaceColor=get(p1,'Color');
%         p1.DisplayName=['HRAB, PT= ',num2str(PT_plot(j)),' N'];
%     end
% end
% end

Airbeam_NoHZ=load(['Airbeam No Hz_','num2str(appload),'_outputs.mat']);
HZ=2;
HRAB=2;

P=Airbeam_NoHZ.P;
PT=Airbeam_NoHZ.PT;
LF=Airbeam_NoHZ.LF;
P_plot=unique(P);
PT_plot=unique(PT);

% Space_plot=unique(HRAB_HZ.Spacing);

% for i=1:size(P_plot,1)
%     for j=1:size(PT_plot)
%         for k=1:size(Space_plot,1)
%             Guy Pretension vs. Load Fraction
X=P(PT==PT_plot(j))*6.89;
    [X,I]=sort(X);
Y=LF(PT==PT_plot(j));
    Y=Y(I);
    p1=plot(ax1,X,Y);
    if isempty(p1)~=1
        p1.Color=linecolor{i};
        p1.Marker=marker{j};
        p1.LineStyle=linetype{HRAB};
        p1.LineWidth=2;
        p1.MarkerFaceColor=get(p1,'Color');
        p1.DisplayName=['Airbeam, PT= ',num2str(PT_plot(j)),' N'];
    end
%         end
%     end
% end

xlabel(ax1,'Internal Pressure (kPa)');
ylabel(ax1,'Load Fraction');
legend(ax1,'show','Location','eastoutside');
grid(ax1,'on');
box(ax1,'on');

name=['Internal Pressure vs LF_NOHZ'];
print(fig1,name,'-dmeta'); %save open figures as explodable .emf files
print(fig1,name,'-dpng');
savefig(fig1,name); %save the figure as a Matlab .Fig file
close(fig1);
function [] = GatherLCs_HRAB_PAA_HZ_01_opti(appload)
%Goes through the 'Solve Models' folder and gathers the name of the model,
%as well as the max load that that model was able to carry before
%collapsing

close all

Models=dir;
Models(1:2,:)=[];
% appload=15; % load for models that you wish to analyze

a=0;
for i=1:size(Models,1)
    a=a+double(contains(Models(i).name,'outputs'));
end

% Check that the variables have been generated
if a==0
    % Not generated, generate the files
    generate_mat_files_opti
end

fig1=figure(1);
ax1=axes(fig1);
hold(ax1,'on')

% Line Color indicates air pressure
linecolor={'r','g','b','c','m','y','k'};
% Marker Type indicates PreTension
marker={'o','+','*','x','s','d','^','v','>','<','p','h'};
% Line type indicates airbeam or HRAB
linetype={'-','--','-.'};

file=['HRAB Hz_',['../num2str(appload),'_outputs.mat'];
HRAB_HZ=load(file);

HZ=1;
HRAB=1;

P=HRAB_HZ.P;
PT=HRAB_HZ.PT;
Load=HRAB_HZ.Load; % N/m^2, This is the maximum load the model was able to
% acheive before buckling

P_plot=unique(P);
PT_plot=unique(PT);
% Space_plot=unique(HRAB_HZ.Spacing);

% for i=1:size(P_plot,1)
%     for j=1:size(PT_plot)
%         for k=1:size(Space_plot,1)
%         % Guy Pretension vs. Load Fraction
%     %end
% end
%end
X=P(PT==PT_plot(j))*6.89;
[X,I]=sort(X);
Y=Load(PT==PT_plot(j));
Y=Y(I);
p1=plot(ax1,X,Y);
if isempty(p1)==1
  p1.Color=linecolor{i};
p1.Marker=marker{j};
p1.LineStyle=linetype{HRAB};
p1.LineWidth=2;
  p1.MarkerFaceColor=get(p1,'Color');
p1.DisplayName=['Airbeam, PT= ',num2str(PT_plot(j)),' N'];
end
end
end

Airbeam_HZ=load(['Airbeam Hz_',num2str(appload),'_outputs.mat']);

HZ=1;
HRAB=2;
P=Airbeam_HZ.P;
PT=Airbeam_HZ.PT;
Load=Airbeam_HZ.Load;
P_plot=unique(P);
PT_plot=unique(PT);
% Space_plot=unique(HRAB_HZ.Spacing);

% for i=1:size(P_plot,1)
%   for j=1:size(PT_plot)
%     for k=1:size(Space_plot,1)
%       Guy Pretension vs. Load Fraction
X=P(PT==PT_plot(j))*6.89;
[X,I]=sort(X);
Y=Load(PT==PT_plot(j));
Y=Y(I);
p1=plot(ax1,X,Y);
if isempty(p1)==1
  p1.Color=linecolor{i};
p1.Marker=marker{j};
p1.LineStyle=linetype{HRAB};
p1.LineWidth=2;
  p1.MarkerFaceColor=get(p1,'Color');
p1.DisplayName=['Airbeam, PT= ',num2str(PT_plot(j)),' N'];
end
end
% end
% end

xlabel(ax1,'Internal Pressure (kPa)');</p>
ylabel(ax1,'Max Achieved Load (N/m^2)');</p>
legend(ax1,'show','Location','eastoutside');
grid(ax1,'on');
box(ax1,'on');

name=['Internal Pressure vs Load_HZ_',num2str(appload)];
print(fig1,name,-dmeta); %save open figures as explodable .emf files
print(fig1,name,-dpng);
savefig(fig1,name); %save the figure as a Matlab .Fig file

close(fig1);

fig1=figure(1);
ax1=axes(fig1);
hold(ax1,'on')

HRAB_NoHZ=load(['HRAB No Hz_','num2str(appload),'_outputs.mat']);
HZ=0;
HRAB=1;

P=HRAB_NoHZ.P;
PT=HRAB_NoHZ.PT;
Load=HRAB_NoHZ.Load;

P_plot=unique(P);
PT_plot=unique(PT);

%  Space_plot=unique(HRAB_HZ.Spacing);

%  for i=1:size(P_plot,1)
%    for j=1:size(PT_plot)
%      for k=1:size(Space_plot,1)
%        Guy Pretension vs. Load Fraction
%        X=P(PT==PT_plot(j))*6.89;
%        [X,I]=sort(X);
%        Y=Load(PT==PT_plot(j));
%        p1=plot(ax1,X,Y);
%        if isempty(p1)~=1
%          p1.color=linecolor{i};
%          p1.Marker=marker{j};
%          p1.LineStyle=linetype{HRAB};
%          p1.LineWidth=2;
%          p1.MarkerFaceColor=get(p1,'Color');
%          p1.DisplayName=['HRAB, PT= ',num2str(PT_plot(j)),', N'];
%        end
%      end
%    end
%  end

Airbeam_NoHZ=load(['Airbeam No Hz_','num2str(appload),'_outputs.mat']);
HZ=0;
HRAB=2;

P=Airbeam_NoHZ.P;
PT=Airbeam_NoHZ.PT;
Load=Airbeam_NoHZ.Load;

P_plot=unique(P);
PT_plot=unique(PT);

for i=1:size(P_plot,1)
    for j=1:size(PT_plot)
        for k=1:size(Space_plot,1)
            Guy Pretension vs. Load Fraction
            X=P(PT==PT_plot(j))*6.89;
            [X,I]=sort(X);
            Y=Load(PT==PT_plot(j));
            Y=Y(I);
            p1=plot(ax1,X,Y);
            if isempty(p1)==1
                p1.Color=LineColor{i};
                p1.Marker=marker{j};
                p1.LineStyle=linetype{HRAB};
                p1.LineWidth=2;
                p1.MarkerFaceColor=get(p1,'Color');
                p1.DisplayName=['Airbeam, PT= ',num2str(PT_plot(j)),' N'];
            end
        end
    end
end

xlabel(ax1,'Internal Pressure (kPa)');
ylabel(ax1,'Max Achieved Load (N/m^2)');
legend(ax1,'show','Location','eastoutside');
grid(ax1,'on');
box(ax1,'on');

name=['Internal Pressure vs Load_NoHZ_',num2str(appload)];
print(fig1,name,'-dmeta'); %save open figures as explodable .emf files
print(fig1,name,'-dpng');
savefig(fig1,name); %save the figure as a Matlab .fig file

close(fig1);

end
https://www.mathworks.com/products/matlab
function generate_els_nodes_HRAB;

global NODES;
global ELS_AT_NODES;
global LOCAL_NODES;
global CONNECTIVITIES;
global MIDDLE_NODES;
global GUYS

numnodes = size(NODES,1);
ELS_AT_NODES = zeros(numnodes,0);
LOCAL_NODES = zeros(numnodes,0);
numbers_at_nodes = zeros(numnodes,1);

numels = size(CONNECTIVITIES,1);
for (i = 1:numels);
    start_node = CONNECTIVITIES(i,1);
    end_node = CONNECTIVITIES(i,2);
    middle_node = CONNECTIVITIES(i,3);

    numbers_at_nodes([start_node, end_node, middle_node],1) = ... 
    numbers_at_nodes([start_node, end_node, middle_node],1) + [1 1 1]';

    num_start = numbers_at_nodes(start_node,1);
    num_end = numbers_at_nodes(end_node,1);
    num_middle = numbers_at_nodes(middle_node,1);

    ELS_AT_NODES(start_node, num_start) = i;
    ELS_AT_NODES(end_node, num_end) = i;
    ELS_AT_NODES(middle_node, num_middle) = i;

    LOCAL_NODES(start_node,num_start) = 1;
    LOCAL_NODES(end_node,num_end) = 2;
    LOCAL_NODES(middle_node,num_middle) = 3;
end

% Map the GUY elements (includes all guy lines and roof elements
for i = 1:size(GUYS,1)
    start_node = GUYS(i,1);
    end_node = GUYS(i,2);

    numbers_at_nodes([start_node, end_node],1) = ... 
    numbers_at_nodes([start_node, end_node],1) + [1 1]';

    num_start = numbers_at_nodes(start_node,1);
    num_end = numbers_at_nodes(end_node,1);

    count = numels+i;
    ELS_AT_NODES(start_node, num_start) = count;
    ELS_AT_NODES(end_node, num_end) = count;

    LOCAL_NODES(start_node,num_start) = 1;
end
LOCAL_NODES(end_node, num_end) = 2;

end
end
function [] = generate_HRAB_DL_HZ
% Generates and assigns the DL values to the DL_F global variable in PAA
% This code also adds X% of the vertical load (Dead and Applied) as a horizontal load to the
% airbeam nodes.

global MODEL_PARAMS_I
global LOAD_DATA_I
global NODES
global CONNECTIVITIES
global PROPS
global DL_F
global TOTAL_F
global HRAB_Airbeam
global HRAB_CONN_I
global HRAB_NODE_I
global HRAB_WALL
global HRAB_ROOF
global HRAB_STRUT
global GUYS
global LOAD_VECTOR_I
global ALL_LOADS_I

% Find element lengths and assign them to their respective spots in the
% PROPS global variable

% Increase the DL_F & TOTAL_F by the number of DOF that were added, The guy nodes
% are already in the DL_F so we need to subtract 2.
% Reset the matrix by multiplying by 0.
DL_F=[DL_F;zeros(3*(size(NODES,1)-size(HRAB_NODE_I.Airbeam,1)-2),1)];
TOTAL_F=[TOTAL_F;zeros(3*(size(NODES,1)-size(HRAB_NODE_I.Airbeam,1)-2),1)];

% Remove the vertical loads from the nodes that are covered by the roof
conn1=find(HRAB_ROOF.LA==CONNECTIVITIES(:,1));
conn2=find(HRAB_ROOF.RA==CONNECTIVITIES(:,2));
% Airbeam elements under the left roof fabric.
for i = [1:conn1-1,]
    % Remove vertical loads from the start nodes of the airbeam
    % element
    node1 = CONNECTIVITIES(i,1);
    start_dof = node1*3-1;
    TOTAL_F(start_dof,1) = 0;
end
% Airbeam elements under the right roof fabric.
for i = [conn2+1:60]
    % Remove vertical loads from the end nodes of the airbeam
    % element
    node2 = CONNECTIVITIES(i,2);
    end_dof = node2*3-1;
    TOTAL_F(end_dof,1) = 0;
end
% Re-find the load on the airbeam where the roof connects. This is
% because the load is not exactly half due to slope change. Do not
% need to change the DL_F value because the dead load on the
% airbeam element is the same. We will add the roof fabric to
% DL_F on the airbeam later.
    gravity = MODEL_PARAMS_I.arch_spacing*HRAB_ROOF.gravity*1/(1000^2); %generate line
load in N/mm

% Left roof attachment
% The element
    node1 = CONNECTIVITIES(conn1,1);
    node2 = CONNECTIVITIES(conn1,2);
    delx = NODES(node2,1) - NODES(node1,1);
    start_dof = node1*3 - 1;
% now the uniform gravity load that acts over x-projection -- this is
% treated as a transient load
% The following are for uniform gravity loads. I am assuming that there
% is only snow load applied to the top.
    the_load = gravity*delx;
    TOTAL_F(start_dof,1) = -the_load/2.0;
    TOTAL_F(start_dof-1,1) = the_load/2.0*MODEL_PARAMS_I.HZ;

% Right roof attachment
% The element
    node1 = CONNECTIVITIES(conn2,1);
    node2 = CONNECTIVITIES(conn2,2);
    delx = NODES(node2,1) - NODES(node1,1);
    end_dof = node2*3 - 1;
% now the uniform gravity load that acts over x-projection -- this is
% treated as a transient load
% The following are for uniform gravity loads. I am assuming that there
% is only snow load applied to the top.
    the_load = gravity*delx;
    TOTAL_F(end_dof,1) = -the_load/2.0;
    TOTAL_F(end_dof-1,1) = the_load/2.0*MODEL_PARAMS_I.HZ;

% Go through the new HRAB elements
for i = HRAB_CONN_I.Wall(1):size(CONNECTIVITIES,1)
% Assign the properties of each of the elements
    if HRAB_CONN_I.Wall(1)<=i && i<=HRAB_CONN_I.Wall(end)
        PROPS(i,1) = HRAB_WALL.E*HRAB_WALL.A;
        PROPS(i,2) = HRAB_WALL.G*HRAB_WALL.A/2;
        wt = HRAB_WALL.wt;
        gravity = HRAB_WALL.gravity;
    elseif HRAB_CONN_I.Strut(1)<=i && i<=HRAB_CONN_I.Strut(end)
        PROPS(i,1) = HRAB_STRUT.E*HRAB_STRUT.A;
        PROPS(i,2) = HRAB_STRUT.G*HRAB_STRUT.A/2;
        wt = HRAB_STRUT.wt;
        gravity = HRAB_STRUT.gravity;
    end
    delx = NODES(CONNECTIVITIES(i,2),1) - NODES(CONNECTIVITIES(i,1),1);
dely = NODES(CONNECTIVITIES(i,2),2) - NODES(CONNECTIVITIES(i,1),2);
PROPS(i,3) = sqrt(delx^2 + dely^2);

% now, make sure that center node lies on line from nodes 1 - 2
elcon = CONNECTIVITIES(i,:);
NODES(elcon(3),:) = (NODES(elcon(1,:)) + NODES(elcon(2,:),:))/2.0;

% Assigning Dead Loads
node1 = CONNECTIVITIES(i,1);
node2 = CONNECTIVITIES(i,2);
% first, the self-weight that acts over member length -- this is
% treated as an initial load
the_load = wt*PROPS(i,3);
start_dof = node1*3-1;
end_dof = node2*3-1;
DL_F(start_dof,1) = DL_F(start_dof,1) - the_load/2.0;
DL_F(end_dof,1) = DL_F(end_dof,1) - the_load/2.0;
% now the uniform gravity load that acts over x-projection -- this is
% treated as a transient load
% delx = NODES(CONNECTIVITIES(i,2),1) - NODES(CONNECTIVITIES(i,1),1);
% The following are for uniform gravity loads. I am assuming that there
% is only snow load applied to the top.
% it is treated as a distributed load
% the_load = gravity*delx;
TOTAL_F(start_dof,1) = TOTAL_F(start_dof,1) - the_load/2.0;
TOTAL_F(end_dof,1) = TOTAL_F(end_dof,1) - the_load/2.0;
end

% Loop over the roof element that are stored in GUYs variable and
% assign to the TOTAL_F variable
wt = HRAB_ROOF.wt;
gravity = MODEL_PARAMS_I.arch_spacing*HRAB_ROOF.gravity/(1000^2); %generate line load in N/mm
Distributed=0; %1= load the roof in a distributed load manner, 0= place the load on the roof
at the airbeam and wall only
if Distributed==1
    for i=4:size(GUYs,1)
% Assigning Dead Loads
    node1 = GUYs(i,1);
    node2 = GUYs(i,2);

delx = NODES(node2,1) - NODES(node1,1);
dely = NODES(node2,2) - NODES(node1,2);
L = sqrt(delx^2 + dely^2);

% first, the self-weight that acts over member length -- this is
% treated as an initial load
the_load = wt*L;
start_dof = node1*3-1;
end_dof = node2*3-1;
DL_F(start_dof,1) = DL_F(start_dof,1) - the_load/2.0;
DL_F(end_dof,1) = DL_F(end_dof,1) - the_load/2.0;
% now the uniform gravity load that acts over x-projection -- this is
% treated as a transient load
% The following are for uniform gravity loads. I am assuming that there
% is only snow load applied to the top.
the_load = gravity*delx;
the_hz_load = MODEL_PARAMS_I.HZ*the_load;
TOTAL_F(start_dof,1) = TOTAL_F(start_dof,1) - the_load/2.0;
TOTAL_F(start_dof-1,1) = TOTAL_F(start_dof-1,1) + the_hz_load/2.0;

TOTAL_F(end_dof,1) = TOTAL_F(end_dof,1) - the_load/2.0;
TOTAL_F(end_dof-1,1) = TOTAL_F(end_dof-1,1) + the_hz_load/2.0;
end
else

% Get roof del_x
del_x=NODES(HRAB_ROOF.LA,1)-0; % The left wall is set at X=0;
load=del_x/2*gravity; % Point load that will be added to the roof
hz_load = MODEL_PARAMS_I.HZ*load;

% attachment points and to walls.
% Add load to the neccesary DOF
% wall
Wall_L_top=CONNECTIVITIES(HRAB_CONN_I.Wall(HRAB_WALL.ele),2);
Wall_R_top=CONNECTIVITIES(HRAB_CONN_I.Wall(HRAB_WALL.ele*2),2);
% Left wall
start_dof = Wall_L_top*3-1;
TOTAL_F(start_dof,1) = TOTAL_F(start_dof,1) - load;
TOTAL_F(start_dof-1,1) = TOTAL_F(start_dof-1,1) + hz_load;

% Right Wall
start_dof = Wall_R_top*3-1;
TOTAL_F(start_dof,1) = TOTAL_F(start_dof,1) - load;
TOTAL_F(start_dof-1,1) = TOTAL_F(start_dof-1,1) + hz_load;
% Roof
Roof_L=HRAB_ROOF.LA;
Roof_R=HRAB_ROOF.RA;
% Roof Left attachment
start_dof = Roof_L*3-1;
TOTAL_F(start_dof,1) = TOTAL_F(start_dof,1) - load;
TOTAL_F(start_dof-1,1) = TOTAL_F(start_dof-1,1) + hz_load;
% Roof Right attachment
start_dof = Roof_R*3-1;
TOTAL_F(start_dof,1) = TOTAL_F(start_dof,1) - load;
TOTAL_F(start_dof-1,1) = TOTAL_F(start_dof-1,1) + hz_load;

end
% now, factor loads
DL_factor = LOAD_DATA_I.DL_factor;
SL_factor = LOAD_DATA_I.SL_factor;
WL_factor = LOAD_DATA_I.WL_factor;

% NOTE -- unif. DL included as separate item in input file
% currently TOTAL_F includes only snow load
LOAD_VECTOR_I = SL_factor*TOTAL_F;
% now, convert to global variable ALL_LOADS, numnodesx2 array for FE input
ALL_LOADS_I = zeros(size(NODES,1),2);
total_vert = 0;
for (i = 1:size(NODES,1))
    start_dof = i*3-2;
    end_dof = i*3-1;
    ALL_LOADS_I(i,:) = (LOAD_VECTOR_I(start_dof:end_dof))';
end
end
function [ ] = generate_HRAB_props()
%Generates PROPS related to the HRAB structure

global HRAB_WALL
% global HRAB_ROOF
global HRAB_STRUT
global PROPS
global HRAB_CONN_I
global NODES
global CONNECTIVITIES
	numel_wall=size(HRAB_CONN_I.Wall,1);
% numel_roof=size(HRAB_CONN_I.Roof,1);
numel_strut=size(HRAB_CONN_I.Strut,1);

temp=zeros(numel_wall+numel_strut,3);
PROPS=[PROPS;temp];

EA=HRAB_WALL.E*HRAB_WALL.A.*ones(numel_wall,1);
GA=HRAB_WALL.G*HRAB_WALL.A.*ones(numel_wall,1);
    delx = NODES(CONNECTIVITIES(HRAB_CONN_I.Wall(:,2),1) -
 NODES(CONNECTIVITIES(HRAB_CONN_I.Wall(:,1),1),1);
    dely = NODES(CONNECTIVITIES(HRAB_CONN_I.Wall(:,2),2) -
 NODES(CONNECTIVITIES(HRAB_CONN_I.Wall(:,1),2),2);
L = sqrt(delx.^2 + dely.^2);
PROPS(HRAB_CONN_I.Wall,:)=[EA,GA./2,L];

% EA=HRAB_ROOF.E*HRAB_ROOF.A.*ones(numel_roof,1);
% GA=HRAB_ROOF.G*HRAB_ROOF.A.*ones(numel_roof,1);
%     delx = NODES(CONNECTIVITIES(HRAB_CONN_I.Roof(:,2),1) -
% NODES(CONNECTIVITIES(HRAB_CONN_I.Roof(:,1),1),1);
%     dely = NODES(CONNECTIVITIES(HRAB_CONN_I.Roof(:,2),2) -
% NODES(CONNECTIVITIES(HRAB_CONN_I.Roof(:,1),2),2);
% L = sqrt(delx.^2 + dely.^2);
% PROPS(HRAB_CONN_I.Roof,:)=[EA,GA./2,L];

EA=HRAB_STRUT.E*HRAB_STRUT.A.*ones(numel_strut,1);
GA=HRAB_STRUT.G*HRAB_STRUT.A.*ones(numel_strut,1);
    delx = NODES(CONNECTIVITIES(HRAB_CONN_I.Strut(:,2),1) -
 NODES(CONNECTIVITIES(HRAB_CONN_I.Strut(:,1),1),1);
    dely = NODES(CONNECTIVITIES(HRAB_CONN_I.Strut(:,2),2) -
 NODES(CONNECTIVITIES(HRAB_CONN_I.Strut(:,1),2),2);
L = sqrt(delx.^2 + dely.^2);
PROPS(HRAB_CONN_I.Strut,:)=[EA,GA./2,L];
end
function [] = Generate_Load_File(load,HRAB,HZ)

%Generates the applied vertical loads in the .dat file. and re-writes the .dat file.

% Read in load parameters from the PAA file.
fid=fopen(['"20psi_20ft.paa"'],'r');
% Read in the .dat file into a cell array
l = 1;
tline = fgetl(fid);
B{l,1} = tline;
while ischar(tline)
    l = l+1;
    tline = fgetl(fid);
    B{l,1} = tline;
end
fclose(fid);

% Rewrite the Current overhead load in the PAA file
B{93,1}=num2str(load);
% Write cell B into txt
fid = fopen(['"20psi_20ft.paa"'],'w');
for l = 1:numel(B)
    if B{l+1} == -1
        fprintf(fid,'%s', B{l});
        break
    else
        fprintf(fid,'%s
', B{l});
    end
end
fclose(fid); %close the file

% load=str2double(cell2mat(B(93))); % N/m^2, horizontal load applied
load=load/(1000^2);
% N/mm^2

% spacing=str2double(cell2mat(B(31))); %mm, spacing of airbeam

fid=fopen(['"20psi_20ft.dat"'],'r');
% Read in the .dat file into a cell array
l = 1;
tline = fgetl(fid);
A{l,1} = tline;
while ischar(tline)
    l = l+1;
    tline = fgetl(fid);
    A{l,1} = tline;
end
fclose(fid);

% Gather the nodes to find the del_x
for temp=38:158
    values=cell2mat(A(temp));
    space_ind=strfind(values,' ');
nodes(temp-37,1)=str2double(values(1:space_ind(1)-1)); % x location
nodes(temp-37,2)=str2double(values(space_ind(1)+1:end)); % y location
end

% Gather the elements to find the connectivities
F=zeros(121,2);
for temp=164:223
    i=temp-163;
    values=cell2mat(A(temp));
    space_ind=strfind(values,' ');
    conns(i,1)=str2double(values(1:space_ind(1)-1)); % start node
    conns(i,2)=str2double(values(space_ind(1)+1:space_ind(2)-1)); % end node
    conns(i,3)=str2double(values(space_ind(2)+1:end)); % middle node
end

% Apply the loads to the F variable, each row of F is a node
del_x=nodes(conns(i,2),1)-nodes(conns(i,1),1);
    nodal_load=-load*spacing*del_x/2; %N
    % Start Node
    F(conns(i,1),2)=nodal_load+F(conns(i,1),2);
    % End Node
    F(conns(i,2),2)=nodal_load+F(conns(i,2),2);
end

if HZ==1
    % Apply the horizontal load to the loads variable.
    F(:,1)=F(:,2).*-0.1;
end

% Write Loads
for temp=247:367
    values=cell2mat(A(temp));
    space_ind=strfind(values,' ');
    num1=str2double(values(1:space_ind(1)-1)); % node number
    values=[num2str(num1),' ',num2str(F(temp-246,1)),',',num2str(F(temp-246,2))];
    A{temp,1}=values;
end

% Write cell A into txt
fid = fopen(['20psi_20ft.dat'], 'w');
for l = 1: numel(A)
    if A{l+1} == -1
        fprintf(fid,'%s', A{l});
        break
    else
        fprintf(fid,'%s
', A{l});
    end
end
fclose(fid); %close the file

% If HRAB = 1, adjust the HRAB Props File
if HRAB==1
    fid=fopen(['HRAB Props.dat'],'r');
    % Read in the .dat file into a cell array
    l = 1;
tline = fgetl(fid);
A{l,1} = tline;
while ischar(tline)
    l = l+1;
    tline = fgetl(fid);
    A{l,1} = tline;
end
fclose(fid);

% Adjust the Roof Load Value
A{45,1}=num2str(load*(1000^2)); % N/m^2

% Write cell A into txt
fid = fopen(['HRAB Props.dat'], 'w');
for l = 1:numel(A)
    if A{l+1} == -1
        fprintf(fid,'%s', A{l});
        break
    else
        fprintf(fid,'%s
', A{l});
    end
end
fclose(fid); % close the file
function generate_loads_HRAB

% This routine generates the factored load vector corresponding to the
% current model.

global ANALYSIS_PARAMS_I;
global MODEL_PARAMS_I;
global MESH_DATA_I;
global LOAD_DATA_I;
global SNOW_DATA_I;
global WIND_DATA_I;
global ADD_DL_I;
global ADD_UNIF_DL_I;
global LOAD_VECTOR_I;
global ALL_LOADS_I;
global HRAB_NODE_I;
global HRAB_CONN_I;
global TOTAL_F;
global DL_F

% nodes and els
nodes = MODEL_PARAMS_I.nodes;
els = MODEL_PARAMS_I.els;

numnodes = size(nodes,1);
if (MODEL_PARAMS_I.include_guys == 1)
    numnodes = numnodes - 2;
    nodes = nodes([HRAB_NODE_I.Airbeam;HRAB_NODE_I.Wall;HRAB_NODE_I.Strut];,:);
end

numels = size(els,1);

LOAD_VECTOR_I = zeros(size(nodes,1)*3,1);
ALL_LOADS_I = zeros(numnodes,2);
dl_vector = zeros(size(nodes,1)*3,1);
sl_vector = zeros(size(nodes,1)*3,1);
w1_vector = zeros(size(nodes,1)*3,1);
add_load_vector = zeros(size(nodes,1)*3,1);

% For no horizontal load
%    generate_HRAB_DL_20190401_01
% For horizontal load on airbeam
%    generate_HRAB_DL_HZ
end
function generate_stresses_HRAB

% generates output
% STRESSES contains the max tensile stress for each node
% CAPACITY_DATA contains P/Pn and M/Mn for each member. M is taken as the
% largest value at either end of the member. Mn is interpolated from the
% moment-curvature relationship corresponding to P and the specified level of
% wrinkling, alpha. Pn is the pressure resultant.

clear STRESSES;

global BASIC_PROPS
global NODES
global CONNECTIVITIES
global PROPS
global F
global K
global K_BC
global U
global R
global BOUNDARIES
global M_kappa
global MEMBER_FORCES
global GUY_FORCES
global CAPACITY_DATA;
global STRESSES;
global ELS_AT_NODES;
global LOCAL_NODES;
global P_V_M;
global STRAP_DATA;
global BRAIDED_M_KAPPA;

linear = BASIC_PROPS(1);
E = BASIC_PROPS(4);
p = BASIC_PROPS(2);
r = BASIC_PROPS(3)/2;
alpha = BASIC_PROPS(6);
Pn = p*pi*r^2;
woven = BASIC_PROPS(9);
numels = size(MEMBER_FORCES,1);
umnodes = size(NODES,1);

% initialize to zeros of correct size
CAPACITY_DATA = zeros(numnodes,3);
STRESSES = zeros(numnodes,3);
P_V_M = zeros(numnodes,3);

for (i = 1:numnodes)

    % get list of elements connected to nodes, strip zeros
    the_connected_els = ELS_AT_NODES(i,:);
    the_local_nodes = LOCAL_NODES(i,:);
count = 1;
connected_els = zeros(0,0);
local_nodes = zeros(0,0);
for (j = 1:size(the_connected_els,2))
    if (the_connected_els(j) > 0)
        connected_els(count) = the_connected_els(j);
        local_nodes(count) = the_local_nodes(j);
        count = count + 1;
    end
end

P = 0;
M = 0;
V = 0;
um_connected = size(connected_els,2);

% check to be sure there are beam elements connected to this node
if (num_connected > 0)
    for (j = 1:size(connected_els,2))
        the_el = connected_els(j);
        the_node = local_nodes(j);
        % positive P is compression, positive M produces positive curvature
        % positive V is per beam sign convention
        if the_el<=size(CONNECTIVITIES,1) %The element is a 3 noded element
            if (the_node == 1) % starting node
                P = P + MEMBER_FORCES(the_el, 1);
                V = V + MEMBER_FORCES(the_el, 2);
                M = M - MEMBER_FORCES(the_el, 3);
            elseif (the_node == 2) % ending node
                P = P - MEMBER_FORCES(the_el, 4);
                V = V - MEMBER_FORCES(the_el, 5);
                M = M + MEMBER_FORCES(the_el, 6);
            else % This is a middle node, will not be connected to any other
                % element. Here, we take averages of end node values.
                P = (MEMBER_FORCES(the_el, 1) - MEMBER_FORCES(the_el, 4))/2;
                V = (MEMBER_FORCES(the_el, 2) - MEMBER_FORCES(the_el, 5))/2;
                M = (-MEMBER_FORCES(the_el, 3) + MEMBER_FORCES(the_el, 6))/2;
            end
        else %The element is a guy element
            if (the_node == 1) % starting node
                P = P + GUY_FORCES(the_el-numels, 1);
                V = V + GUY_FORCES(the_el-numels, 2);
                M = M - GUY_FORCES(the_el-numels, 3);
            elseif (the_node == 2) % ending node
                P = P - GUY_FORCES(the_el-numels, 4);
                V = V - GUY_FORCES(the_el-numels, 5);
                M = M + GUY_FORCES(the_el-numels, 6);
            end
        end
    end
% compute the average
P = P/num_connected;
V = V/num_connected;
M = M/num_connected;
P_V_M(i,1) = P;
P_V_M(i,2) = V;
P_V_M(i,3) = M;

if min(connected_els)<=60
    % use absolute values for woven
    if (woven == 1)
        M = abs(M);
    end

    final_M_kappa = interp_M_kappa(-P);

    % Interpolate the curvature from moment and the NA locations, compute
    % stress. Interpolated curvature is more accurate.
    kappa = interp1(final_M_kappa(1,:,2), final_M_kappa(1,:,1), M, ...
                    'linear', 'extrap');

    % Check for situation where M exceeds max. stored value,
    % re-compute kappa accordingly.
    red_incr = 0.001;
    red = 1.0 - red_incr;
    flag = 0;
    while (isnan(kappa))
        kappa = interp1(final_M_kappa(1,:,2), final_M_kappa(1,:,1), ...
                        red*M);
        red = red - red_incr;
        flag = 1;
    end
    if (flag == 1)
        red + red_incr
    end

if (woven == 1)
    if (linear == 1)
        STRESSES(i,1) = (Pn - P)/(2*pi*r) + M/(pi*r^2);
    else
        y_bar = interp1(final_M_kappa(1,:,2), final_M_kappa(1,:,4), M);
        eps_initial = final_M_kappa(1,1,6);
        eps = eps_initial + kappa*y_bar;
        STRESSES(i,1) = stress_strain(eps, E);
    end
else  % braided, strapped, interpolate max. strap force
    STRESSES(i,1) = interp1(final_M_kappa(1,:,1), ...
                        final_M_kappa(1,:,7), kappa);
    %if (isnan(kappa))
    %    disp('stop_kappa');
    %    kappa
    %    M
    %    STRESSES(i,1)
    %    pause
```matlab
%elseif (isnan(STRESSES(i,1)))
%    isnan(STRESSES(i,1))
%    disp('stop_stresses');
%    kappa
%    STRESSES(i,1)
%    final_M_kappa(1,:,1)
%    pause
%end
end

% interpolate the wrinkle depth
wrinkle_depth = interp1(final_M_kappa(1,:,1), final_M_kappa(1,:,5), kappa);

numpts = max(size(final_M_kappa));

% now, compute M/Mw accounting for axial load
linear = BASIC_PROPS(1);
%if (linear == 1)
%    Mw = 1e6*M;
%else
%    % note that for braided/strapped, -M implies -wrinkle depth for
%    % bookeeping purposes
%
if (woven == 1)
    CAPACITY_DATA(i,2) = abs(2*M)/((Pn-P)*r);
else
    % braided, strapped
    A = sum(STRAP_DATA(:,2)); % strap area
    if (M >= 0)
        Mw = BRAIDED_M_KAPPA(12) - P/A*BRAIDED_M_KAPPA(8);
    else
        Mw = BRAIDED_M_KAPPA(13) - P/A*BRAIDED_M_KAPPA(9);
    end
    CAPACITY_DATA(i,2) = abs(M)/Mw;
end

if (p > 1e-6)
    CAPACITY_DATA(i,1) = P/Pn;
else
    CAPACITY_DATA(i,1) = 0.0;
end

% now, take absolute value of wrinkle depth for output
CAPACITY_DATA(i,3) = abs(wrinkle_depth);
end
end
end
```

https://www.mathworks.com/products/matlab
function G = get_G_PT_HRAB(PT_init,guy_I)

global BASIC_PROPS;
global NODES;
global UPDATED_NODES;
global CONNECTIVITIES;
global BOUNDARIES;
global TOTAL_F;
global DL_F;
global F;
global M_kappa;
global U;
global MEMBER_FORCES;
global STRESSES;
global CAPACITY_DATA;
global GUYS

global GUY_FORCES

global REACTIONS

global LOCAL_NODES;
global STRAP_DATA;
global ELS_AT_NODES;
global ALL_RESULTS;
global P_V_M;
global LARGE_DEF;

num_steps = 2;
del_F = F/num_steps;
F = zeros(size(F));
last_good_U = zeros(size(del_F));

for (i = 1:num_steps)
    F = F + del_F;
    OK = newton_solver_HRAB(last_good_U, 1e-4, 20, 1, 0);
    last_good_U = U;
end

G = GUY_FORCES(guy_I,4) - PT_init;
end

https://www.mathworks.com/products/matlab
function [ ] = HRAB_2nodemesh( dat_input_file, plot )
% input_file is actually a .dat file

% Used for testing of the main solver
% clear global
global NODES
global CONNECTIVITIES
global GYUS
global MODEL_PARAMS_I
global BOUNDARIES
global HRAB_NODE_I
global HRAB_CONN_I
global HRAB_WALL
global HRAB_ROOF
global HRAB_STRUT
global HRAB_ELE_I
global BEAM_PROPS

% open main file for output
% dat_input_file=strrep(fe_input_file, '.paa', '.dat');
fname=dat_input_file(1:end-4);
fe_input_file=strrep(dat_input_file, '.dat', '.paa');
output_file=strrep(dat_input_file, '.dat', '.out');

load_model_HRAB(fe_input_file) % Loads the MODEL_PARAMS_I variable for plotting

% Generate index values for the airbeam
% Node_Airbeam_end=length(NODES)-2; % Node number for the last node on the airbeam
Airbeam_Conn=[1:size(CONNECTIVITIES,1)]; % Index of elements that correspond to the airbeam elements.
HRAB_ELE_I=ones(size(Airbeam_Conn,1),1); % Airbeam elements
Airbeam_I=[1:Node_Airbeam_end]; % Vector of indices to map Airbeam Nodes

% Strip the guy anchor nodes and save to place them at the end later
NODE_GUY=NODES(end-1:end,:);
NODES(end-1:end,:)=[]; MODEL_PARAMS_I.nodes=NODES;

% Number of elements in the wall, strut, and Roof Fabric
output_fname=strcat('HRAB_Props_',fname,'.out');
out_fid = fopen(output_fname, 'wt');

% read input data, set up model, echo input data
read_input_HRAB(['HRAB_Props_','fname','.dat'], out_fid);

Wall_n=2*HRAB_WALL.ele+1;
Roof_n=HRAB_ROOF.ele+1;
AB_fab_L=HRAB_ROOF.LA; % Node which the left fabric touches when running into the airbeam
AB_fab_R=HRAB_ROOF.RA; % Node which the right fabric touches when running into the airbeam

Strut_n=2*HRAB_STRUT.ele+1;
AB_strut_L=HRAB_STRUT.LA; % Node which the left strut touches the airbeam
AB_strut_R=HRAB_STRUT.RA; %node which the right strut touches the airbeam

% Wall Panels

% Nodes
Nodes_wall_X_L=zeros(Wall_n,1); %allocate space
Nodes_wall_X_R=ones(Wall_n,1).*NODES(121,1);
Nodes_wall_Y=linspace(0,HRAB_WALL.ht,Wall_n)'; %assumes the wall starts at the ground
Nodes_wall=[Nodes_wall_X_L,Nodes_wall_Y;
    Nodes_wall_X_R,Nodes_wall_Y]; %Gather nodal coordinates

% Remove the nodes that have already been defined.
% Nodes_wall(1,:)=[]; % Removes the node that overlaps at the base of the L Panel
% Nodes_wall(Wall_n,:)=[]; % Removes the node that overlaps at the base of the R panel
L=length(NODES)+1; % increase by one so that the wall has its own base point
L_wall_base=L; % Node number for the base of the left wall.
NODES=[NODES; % Add the wall panel nodes to the NODES variable
    Nodes_wall];

% Generate Index values for the wall nodes in the NODES global variable
% wall_I=[Airbeam_I(end)+1:Airbeam_I(end)+size(Nodes_wall,1)'];

% Connectivity for the wall panels
% Left Wall Panel
Wall_conn_L=zeros(HRAB_WALL.ele,3); %sets up the vector to hold the wall connectivites
Wall_conn_L(1,1:3)=[L,L+1,L+2];
for i=2:size(Wall_conn_L,1)
    temp=fliplr(Wall_conn_L(i-1,:));
    Wall_conn_L(i,:)=temp+[0,2,4];
end
% Wall_conn_L(1,1)=L+1; %set the first node of the wall element base to begining of the airbeam
% Swap columns 2 and 3 so that the array is [start end middle] node
wall_conn_L(:,[2 3])=Wall_conn_L(:,[3 2]);
% Swap the element start and end points so that the hinged end is at the base of the wall
Wall_conn_L(1,[1,2])=Wall_conn_L(1,[2,1]);
CONNECTIVITIES=[CONNECTIVITIES;Wall_conn_L];
% Create the indices for the elements
temp=ones(size(Wall_conn_L,1),1)*2; % Assign indices to wall elements
temp(1)=10; % Assign the indices to the hinged elements
HRAB_ELE_I=[HRAB_ELE_I,temp];
% Right Wall Panel
Wall_conn_R=zeros(HRAB_WALL.ele,3); %sets up the vector to hold the wall connectivites
L=L+Wall_n;
R_wall_base=L; % node number for the base of the right wall
Wall_conn_R(1,1:3)=[L,L+1,L+2];
for i=2:size(Wall_conn_R,1)
    temp=fliplr(Wall_conn_R(i-1,:));
    Wall_conn_R(i,:)=temp+[0,2,4];
end
% wall_conn_R(1,1)=Node_Airbeam_end; %set the first node of the wall element base to end of the airbeam
% Swap columns 2 and 3 so that the array is [start end middle] node
wall_conn_R(:,[2 3])=wall_conn_R(:,[3 2]);
% Swap the element start and end points so that the hinged end is at the base of the wall
wall_conn_R(1,[1,2])=wall_conn_R(1,[2,1]);
CONNECTIVITIES=[CONNECTIVITIES;wall_conn_R];
% Create the indices for the elements
temp=ones(size(wall_conn_R,1),1)*2; % Assign indices to wall elements
temp(1)=10; % Assign the indices to the hinged elements
HRAB_ELE_I=[HRAB_ELE_I;temp];

wall_top_R_node=length(NODES);
wall_top_L_node=L-1;
curr_ele=length(NODES);

% Roof Fabric
% Left Fabric
% Nodes
Nodes_fab_X_L=linspace(NODES(Wall_top_L_node,1),NODES(AB_fab_L,1),Roof_n)';
% Vector of X cordinates for the left roof fabric
Nodes_fab_Y_L=linspace(NODES(Wall_top_L_node,2),NODES(AB_fab_L,2),Roof_n)';
% Vector of Y cordinates for the left roof fabric
Nodes_fab_L=[Nodes_fab_X_L,Nodes_fab_Y_L];
Nodes_fab_L(1,:)=[];
Nodes_fab_L(end,:)=[];
NODES=[NODES;Nodes_fab_L];

%Connectivities, Will be added to the GUY matrix later
Conn_fab_L=zeros(HRAB_ROOF.ele,2); %sets up the vector to hold the guy connectivites
Conn_fab_L(1,:)=wall_top_L_node,curr_ele+1);
for i=2:size(Conn_fab_L,1)
    temp=fliplr(Conn_fab_L([i-1,:]));
    Conn_fab_L(i,:)=temp(1) temp(1)+[0 1];
end
% Conn_fab_L(1,1)=wall_top_L_node; % set the first node of the wall element base to end of the airbeam
Conn_fab_L(end)=AB_fab_L; % assigns end node to top of left wall panel
roof_conns=Conn_fab_L;
curr_ele=length(NODES);

% Right Fabric
% Nodes
Nodes_fab_X_R=linspace(NODES(AB_fab_R,1),NODES(Wall_top_R_node,1),Roof_n)';
% Vector of X cordinates for the right roof fabric
Nodes_fab_Y_R=linspace(NODES(AB_fab_R,2),NODES(Wall_top_R_node,2),Roof_n)';
% Vector of Y cordinates for the right roof fabric
Nodes_fab_R=[Nodes_fab_X_R,Nodes_fab_Y_R];
Nodes_fab_R(1,:)=[];
Nodes_fab_R(end,:)=[];
%Connectivities
Conn_fab_R=zeros(HRAB_ROOF.ele,2); %sets up the vector to hold the guy connectivites
Conn_fab_R(1,:)=[wall_top_R_node,curr_ele+1];
for i=2:size(Conn_fab_R,1)
    temp=fliplr(Conn_fab_R(i-1,:));
    Conn_fab_R(i,:)=[temp(1) temp(1)]+[0 1];
end
Conn_fab_R(1,1)=AB_fab_R; %assigns first node to the roof attachment point on the airbeam
Conn_fab_R(end)=wall_top_R_node; %set the last node of the top of the right wall
roof_conns=[roof_conns;Conn_fab_R];
NODES=[NODES;Nodes_fab_R];
curr_ele=length(NODES);

% Indecies of roof elements
    Roof_I=[Wall_I(end)+1:Wall_I(end)+size(Nodes_fab_R,1)*2]';
    Roof_CONN=CONNECTIVITIES(Roof_I,:);

Roof=struct('Nodes',[Nodes_fab_L;Nodes_fab_R],'Ele',[Conn_fab_L;Conn_fab_R],'Ele_n',HRAB_ROOF.ele);

% Struts
%   Left Strut
%      Nodes
    Nodes_strut_X_L=linspace(NODES(Wall_top_L_node,1),NODES(AB_strut_L,1),Strut_n)';
    % Vector of X coordinates for the left roof fabric
    Nodes_strut_Y_L=linspace(NODES(Wall_top_L_node,2),NODES(AB_strut_L,2),Strut_n)';
    % Vector of Y coordinates for the left roof fabric
    Nodes_strut_L=[Nodes_strut_X_L,Nodes_strut_Y_L];
    Nodes_strut_L(1,:)=[];
    Nodes_strut_L(end,:)=[];
    NODES=[NODES;Nodes_strut_L];
% Connectivities
    Conn_strut_L=zeros(HRAB_STRUT.ele,3); %sets up the vector to hold the guy connectivites
    Conn_strut_L(1,:)=[curr_ele,curr_ele+1,curr_ele+2];
for i=2:size(Conn_strut_L,1)
    temp=fliplr(Conn_strut_L(i-1,:));
    Conn_strut_L(i,:)=temp+[0,2,4];
end
Conn_strut_L(1,1)=wall_top_L_node; % Strut goes from the wall to the airbeam
Conn_strut_L(end)=AB_strut_L; %assigns end node to top of left wall panel
% Swap columns 2 and 3 so that the array is [start end middle] node
    Conn_strut_L(:,[2 3])=Conn_strut_L(:,[3 2]);
% Swap start and end nodes so that the linear elastic beam
% element with a hinge at the end node has the hinge at the top of the wall panel
    Conn_strut_L(1,[1,2])=Conn_strut_L(1,[2,1]);
CONNECTIVITIES=[CONNECTIVITIES;Conn_strut_L];
% Create Indicies For the elements
    temp=ones(size(Conn_strut_L,1),1)*3; % Assign linear elastic strut element
    temp(1)=5; temp(end)=5; % Assign the indices to the hinged elements
    HRAB_ELE_I=[HRAB_ELE_I;temp];
curr_ele=length(NODES);

% Right Strut
% Nodes
Nodes_strut_X_R=linspace(NODES(Wall_top_R_node,1),NODES(AB_strut_R,1),Strut_n)';
Nodes_strut_Y_R=linspace(NODES(Wall_top_R_node,2),NODES(AB_strut_R,2),Strut_n)';

% Vector of X coordinates for the right roof fabric
Nodes_strut_R=[Nodes_strut_X_R,Nodes_strut_Y_R];

% Connectivities
Conn_strut_R=zeros(HRAB_STRUT.ele,3); % sets up the vector to hold the guy connectivites

Conn_strut_R(1,:)=[curr_ele,curr_ele+1,curr_ele+2];

for i=2:size(Conn_strut_R,1)
    temp=fliplr(Conn_strut_R(i-1,:));
    Conn_strut_R(i,:)=temp+[0,2,4];
end

Conn_strut_R(1,1)=Wall_top_R_node; % set the first node of the wall element base to end of the airbeam
Conn_strut_R(end,3)=AB_strut_R; % assigns end node to top of right wall panel

% Swap columns 2 and 3 so that the array is [start end middle] node
Conn_strut_R(:,[2 3])=Conn_strut_R(:,[3 2]);

% Swap start and end nodes so that the linear elastic beam % element with a hinge at the end node has the hinge at the % top of the wall panel
Conn_strut_R(1,[1,2])=Conn_strut_R(1,[2,1]);

% Do not need to swap the other end because the end node % contacts the airbeam which is desired.

CONNECTIVITIES=[CONNECTIVITIES;Conn_strut_R];
NODES=[NODES;Nodes_strut_R];

% Create Indicies for the elements

curr_ele=length(NODES);

Strut_I=[Wall_I(end)+1:Wall_I(end)+size(Nodes_strut_R,1)*2]';

% Strut_CONN=CONNECTIVITIES(Strut_I,:);

BEAM_PROPS.E=HRAB_STRUT.E;
BEAM_PROPS.I=HRAB_STRUT.I;

% Guy Lines

numguys = size(GUYS,1);
% re-assign the guy nodes in the NODES and MODEL_PARAMS_I.nodes % variables
NODES(end+1:end+2,:)=NODE_GUY;

% [node 1, node 2, EA (N), initial Pretension (N)]

GUYS([1:2],2)=[Wall_top_L_node;Wall_top_R_node];
node_ct = size(NODES,1);
GUYS([1:2],1) = [node_ct-1,node_ct];  % re-assing the node index
GUY_PT = GUYS(1,end);  % N, guy initial pretension
for i = 1:numguys
    values = GUYS(i,:);
    % compute length
    delx = NODES(values(2),1) - NODES(values(1),1);
dely = NODES(values(2),2) - NODES(values(1),2);
l = sqrt(delx^2 + dely^2);
% [node 1, node 2, length (mm), EA(N), initial pre-tension force(N)]
% note that PT force is replaced by pre-stretch (disp)
% GUYS(i,:) = [values(1:2), l,values(3), values(4)];
GUYS(i,:) = [values(1:2), l,values(4), values(5)];
end

% Loop over the Roof components and assign to the GUY variable
% Pretension in roof elements is based off of guy line
% pretension and angle of the guy line
% alpha=45; % Angle of the guy with the ground
% theta=90-alpha; % Angle of the guy with the wall
% beta=58.891; % angle of the strut with the wall
% gamma=19.913; % angle of the roof with the horizontal
% A=[-sind(beta),-cosd(gamma); -cosd(beta),sind(gamma)];
% B=[-sind(theta);cosd(theta)];
% C=A\B;
% Roof_PT=C(1); % the percent of pretension of the guy in the roof

for i = 1:size(Roof.Ele,1)
    values = Roof.Ele(i,:);
    % compute length
    delx = NODES(values(2),1) - NODES(values(1),1);
dely = NODES(values(2),2) - NODES(values(1),2);
l = sqrt(delx^2 + dely^2);
% [node 1, node 2, length (mm), EA(N), initial pre-tension force(N)]
% If roof elements have their own pretension
% GUYs(numguys+i,:) = [values(1:2),l,HRAB_ROOF.EA,HRAB_ROOF.PT];
% If roof elements do not have their own pretension, their
% pretension is from the pretension of the guys. 58.8% is
% from a RISA 3D analysis
    GUYs(numguys+i,:) = [values(1:2),l,HRAB_ROOF.EA,GUY_PT*0.588];
end

MODEL_PARAMS_I.guy_nodes = GUYS([1 2],[1 2]);  % the guy connectivities

% Set the boundary conditions
BOUNDARIES = ones(size(NODES,1),3);  % Each node has 3 DOF, 1 indicates free DOF
BOUNDARIES(1:end-1,end,[1,2,3]) = [0,0,1;0,0,1];  % pin the ends of the guys
BOUNDARIES(CONNECTIVITIES(Airbeam_Conn(end),2,:),:) = [0,0,1];  % roller the right side
airbeam end
BOUNDARIES(1,:) = [0,0,1];  % roller the left side airbeam end
% Assign Pins to the base of the wall panels
BOUNDARIES(l_wall_base,:)=[0,0,1]; % pin the left wall panel
BOUNDARIES(r_wall_base,:)=[0,0,1]; % pin the right wall panel

% Assign outputs
MODEL_PARAMS_I.nodes=NODES;
MODEL_PARAMS_I.els=CONNECTIVITIES;

% For use in generate_loads_HRAB
HRAB_NODE_I=struct('Airbeam',Airbeam_I,'Wall',Wall_I,...
'Strut',Strut_I,'AB_fab',[AB_fab_L,AB_fab_R]);

% subtract two to negate the guy elements.
Wall_Conn=[size(Airbeam_Conn,1)+1:size(Airbeam_Conn,1)+HRAB_WALL.ele*2]';
Strut_Conn=[Wall_Conn(end)+1:Wall_Conn(end)+HRAB_STRUT.ele*2]';

HRAB_CONN_I=struct('Airbeam',Airbeam_Conn,'Wall',Wall_Conn,'Strut',Strut_Conn);

generate_HRAB_props
generate_loads_HRAB

% Plotting
if plot==1
    plot_HRAB_2node_mesh(NODES,CONNECTIVITIES,GUYS,1,'LIVE');
end

% first, saves the current data
% Save the .dat file
save_fe_input_HRAB(dat_input_file);
% Save the .paa file
save_model(fe_input_file); % saves current geometric inputs to input file
end
function [LC_results] = HRAB_ANALYSIS(FNAME)
% Runs an analysis of the HRAB structure through PressArchAnalysis
clear global
global PATH_NAME
global MODEL_NAME

copyfile(['Original Input Files\',FNAME,'.dat'])
copyfile(['Original Input Files\',FNAME,'.paa'])

PATH_NAME=strcat(pwd,'\');
MODEL_NAME=[FNAME,'.paa'];

% first, saveas the current data
fe_input_file = [FNAME,'.dat'];

% now, create FE input and output files
fe_output_file = strrep(fe_input_file, '.dat', '.out');
% tic
main_solver_HRAB(fe_input_file, fe_output_file);
% toc
HRAB_results_processer(FNAME);
LC_results=dllread([FNAME,'_loadsteps.out']);
LC_results=LC_results(end,:);
one=LC_results(end,1);
two=LC_results(end,2);
end
function [] = HRAB_Analysis_Driver()
% Driver script to run the analyseses to find the capacity of the
% structures under the specified parameters.
%Start in FolderName. In it are a .dat a .paa, and a HR

global_home=pwd;

HRAB=0; % Include the HRAB structure (1 = yes, 0 = no)
Airbeam=0; % Include the Airbeam structure (1 = yes, 0 = no)
HZ=1; % Include horizontal loading (1 = yes, 0 = no)
Optimize=1; % 1 = find the load on the structure which yields a LF=1.
% 0 = find the LF from the given load.

% Model parameters
Param.PT=[0 200]; %N, pretension in guy lines
Param.P_name=[20:10:50]; %psi, internal pressure of airbeam
Param.Angle=[45]; %deg, angle between the ground and the guy
Param.Wall_Height=2134; %mm, height of the wall
Param.Spacing_name=[8]; %ft, Arch spacing

roof_wt=18.3; % oz/yd^2
Loads_name=[15]; %psf
Loads=Loads_name+roof_wt*0.00694; % psf, range of global dead loads to
% apply to the top of the structure
Loads=Loads.*47.88; %N/m^2
if HRAB==1
  for i=1:length(Loads)
    foldername=[HRAB No Hz_,num2str(Loads_name(i))];
    mkdir(foldername);
    % Copy input files
    copyfile('HRAB Props.dat',foldername);
    copyfile('20psi_20ft.dat',foldername);
    copyfile('20psi_20ft.paa',foldername);
    back=cd(foldername);
    % Run InputFileGenerator to generate the required folders for the run.
    Generate_Load_File(Loads(i),HRAB,0); % Re-write .dat and .paa
    % files to incorporate the new loads
    InputFileGenerator_HZ(HRAB,0,Param); % Rewrite the .dat, .paa
    % HRAB.dat files to incorporate the pressure dependent
    % properties, horizontal forces, and guy anchor locations
    % Run analysis
    if Optimize==1
      Driver_parallel(HRAB,0); %Optimization Driver
    else
      Driver_no_opti_parallel(HRAB);
    end
    cd(back);
  end
  if Hz==1
    for i=1:length(Loads)
      % continuation of script
    end
  end
end
foldername=['HRAB Hz_',num2str(Loads_name(i))];
mkdir(foldername);

% Copy input files
    copyfile('HRAB Props.dat',foldername);
    copyfile('20psi_20ft.dat',foldername);
    copyfile('20psi_20ft.paa',foldername);
back=cd(foldername);
% Run InputFileGenerator to generate the required folders for the run.
    Generate_Load_File_01(Loads(i),HRAB,1);
    InputFileGenerator_HZ_20190603(HRAB,Hz,Param);

% Run analysis
    if Optimize==1
        Driver_parallel(HRAB,Hz);
    else
        Driver_no_opti_parallel(HRAB);
    end
    cd(back);
end

if Airbeam==1
    for i=1:length(Loads)
        foldername=['Airbeam No Hz_',num2str(Loads_name(i))];
        mkdir(foldername);
        % Copy input files
            copyfile('20psi_20ft.dat',foldername);
            copyfile('20psi_20ft.paa',foldername);
        back=cd(foldername);
        % Run InputFileGenerator to generate the required folders for the run.
            Generate_Load_File_01(Loads(i),0,0);
            InputFileGenerator_HZ_20190603(0,0,Param);

% Run analysis
            if Optimize==1
                Driver_parallel(0,0);
            else
                Driver_no_opti_parallel(0);
            end
        cd(back);
    end
end

if Hz==1
    for i=1:length(Loads)
        foldername=['Airbeam Hz_',num2str(Loads_name(i))];
        mkdir(foldername);
        % Copy input files
            copyfile('20psi_20ft.dat',foldername);
            copyfile('20psi_20ft.paa',foldername);
        back=cd(foldername);
        % Run InputFileGenerator to generate the required folders for the run.
            Generate_Load_File_01(Loads(i),0,1);
InputFileGenerator_HZ_20190603(0, Hz, Param);

% Run analysis
if Optimize==1
    Driver_parallel(0, 1);
else
    Driver_no_opti_parallel(0);
end
cd(back);
end
end

% Generate Summary Plots
if Optimize==1
    for i=1:size(Loads_name, 2)
        GatherLCs_HRAB_PAA_HZ_01_opti(Loads_name(i));
    end
else
    for i=1:size(Loads_name, 2)
        GatherLCs_HRAB_PAA_HZ_01(Loads_name(i));
    end
end
end
end

function [LF] = HRAB_Analysis_Opti(Load, FName, HRAB, Hz)
% HRAB Analysis Function to be used in optimization
% Move all input files into current folder
movefile(['Original Input Files\*'])
Optimize_Generate_Load_File(FName, Load, HRAB, Hz);
copyfile([FName, '.dat'], 'Original Input Files');
copyfile([FName, '.paa'], 'Original Input Files');
copyfile(['HRAB_Props_\', FName, '.dat'], 'Original Input Files');

[~]=HRAB_ANALYSIS(FName);
Lf=dlmread([FName, '_loadsteps.out']);
LF=max(Lf(:, 1));
end
function [max_deflect] = HRAB_results_processer(filename)
%UNTITLED Summary of this function goes here
% Detailed explanation goes here

global HRAB_WALL
global HRAB_ROOF
global HRAB_STRUT

close all

% Load in the HRAB props to determine the number of elements in each of the
% walls, roof, and struts for plotting the member forces.
[hour,minute,~]=hms(datetime('now'));
output_fnum=strcat('HRAB Props_',datestr(now,'yyyymmdd'),'_','num2str(hour)',num2str(minute)','.out');
out_fid = fopen(output_fnum,'wt');
read_input_HRAB(['HRAB Props_',filename,'.dat'],out_fid);
fclose('all'); % close the file

results=dlmread([filename,'_results.out']);
% [(1)node number, (2)x-direction nodal displacement, (3)y-direction nodal displacement,
% (4)axial load, (5)shear, (6)moment, (7)maximum fabric stress,
% (8)ratio of applied axial load to member pressure resultant,
% (9)wrinkle depth, (10)ratio of moment to wrinkling moment];

n=find(results(:,1)==1);
n=n(end);
full_load=results(n:end,:);
nodes=dlmread([filename,'_nodes.out']); %[Node, Delta X, Delta Y, Total Displ., Rotation]

axial = full_load(:,4);
shear = full_load(:,5);
moment = full_load(:,6);
% Displaced Shape
fig1=figure(1);
ax1=axes(fig1);
hold(ax1,'on')
xlabel(ax1,'X Location (mm)');
ylabel(ax1,'Y Location (mm)');
grid(ax1,'on');
box(ax1,'on');
% Original Shape
p1=scatter(ax1,nodes(:,2),nodes(:,3),'filled');
p1.Marker='o';
p1.LineWidth=2;
p1 MarkerFaceColor=get(p1,'MarkerFaceColor');
p1.DisplayName='Original Shape';
% Displaced Shape
max_deflect=min(nodes(:,3)+full_load(:,3));
p2=scatter(ax1,nodes(:,2)+full_load(:,2),nodes(:,3)+full_load(:,3),'filled');
p2.Marker='o';
% Forces on Members
% Airbeam
n_airbeam=121; %Number of airbeam nodes

fig2=figure(2);
ax21=subplot(2,2,1);
p21=plot(ax21,nodes(1:n_airbeam,2),axial(1:n_airbeam));
p21.Marker='o';
p21.LineWidth=2;
p21.MarkerFaceColor=get(p21,'Color');
xlabel(ax21,'X Location (mm)');
ylabel(ax21,'Axial (N)');
grid(ax21,'on');
box(ax21,'on');

ax22=subplot(2,2,2);
p22=plot(ax22,nodes(1:n_airbeam,2),shear(1:n_airbeam));
p22.Marker='o';
p22.LineWidth=2;
p22.MarkerFaceColor=get(p22,'Color');
xlabel(ax22,'X Location (mm)');
ylabel(ax22,'Shear (N)');
grid(ax22,'on');
box(ax22,'on');

ax23=subplot(2,2,[3,4]);
p23=plot(ax23,nodes(1:n_airbeam,2),moment(1:n_airbeam));
p23.Marker='o';
p23.LineWidth=2;
p23.MarkerFaceColor=get(p23,'Color');
xlabel(ax23,'X Location (mm)');
ylabel(ax23,'Moment (N*mm)');
grid(ax23,'on');
box(ax23,'on');

print(fig2,['Airbeam Member Forces'],'-dmeta'); %save open figures as explodable .emf files
print(fig2,['Airbeam Member Forces'],'-dpng');
savefig(fig2,['Airbeam Member Forces']); %save the figure as a Matlab .Fig file

% Walls
% West Wall
n_wall_W=n_airbeam+2*HRAB_WALL.ele+1; %Top node of the W wall
i_wall_W=[n_airbeam+1:n_wall_W];
fig3=figure(3);
ax31=subplot(2,3,[1 4]);
    p21=plot(ax31,axial(i_wall_W),nodes(i_wall_W,3));
    p21.Marker='o';
    p21.LineWidth=2;
    p21.MarkerFaceColor=get(p21,'Color');
    ylabel(ax31,'Y Location (mm)');
    xlabel(ax31,'Axial (N)');
    grid(ax31,'on');
    box(ax31,'on');
ax32=subplot(2,3,[2 5]);
    p22=plot(ax32,shear(i_wall_W),nodes(i_wall_W,3));
    p22.Marker='o';
    p22.LineWidth=2;
    p22.MarkerFaceColor=get(p22,'Color');
    ylabel(ax32,'Y Location (mm)');
    xlabel(ax32,'Shear (N)');
    grid(ax32,'on');
    box(ax32,'on');
ax33=subplot(2,3,[3,6]);
    p23=plot(ax33,moment(i_wall_W),nodes(i_wall_W,3));
    p23.Marker='o';
    p23.LineWidth=2;
    p23.MarkerFaceColor=get(p23,'Color');
    ylabel(ax33,'Y Location (mm)');
    xlabel(ax33,'Moment (N*mm)');
    grid(ax33,'on');
    box(ax33,'on');

print(fig3,['W Wall Member Forces','-dmeta']); %save open figures as explodable .emf files
print(fig3,['W Wall Member Forces','-dpng']);
savefig(fig3,['W Wall Member Forces']); %save the figure as a Matlab .Fig file

% East Wall
n_wall_E=n_wall_W+2*HRAB_WALL.ele+1; %Top node of the E wall
i_wall_E=[n_wall_W+1:n_wall_E];

fig4=figure(4);
ax41=subplot(2,3,[1 4]);
    p21=plot(ax41,axial(i_wall_E),nodes(i_wall_E,3));
    p21.Marker='o';
    p21.LineWidth=2;
    p21.MarkerFaceColor=get(p21,'Color');
    ylabel(ax41,'Y Location (mm)');
    xlabel(ax41,'Axial (N)');
    grid(ax41,'on');
    box(ax41,'on');
ax42=subplot(2,3,[2 5]);
    p22=plot(ax42,shear(i_wall_E),nodes(i_wall_E,3));
    p22.Marker='o';
    p22.LineWidth=2;
    p22.MarkerFaceColor=get(p22,'Color');
ylabel(ax42,'Y Location (mm)');
xlabel(ax42,'Shear (N)');
grid(ax42,'on');
box(ax42,'on');
ax43=subplot(2,3,[3,6]);
p23=plot(ax43,moment(i_wall_E),nodes(i_wall_E,3));
p23.Marker='o';
p23.LineWidth=2;
p23.MarkerFaceColor=get(p23,'Color');
ylabel(ax43,'Y Location (mm)');
xlabel(ax43,'Moment (N*mm)');
grid(ax43,'on');
box(ax43,'on');

print(fig4,['E Wall Member Forces'],'-dmeta'); %save open figures as explodable .emf files
print(fig4,['E Wall Member Forces'],'-dpng');
savefig(fig4,['E Wall Member Forces']); %save the figure as a Matlab .Fig file

% W Roof
n_roof_W=n_wall_E+HRAB_ROOF.ele-1; %End node of the W roof
i_roof_W=[n_wall_W,n_wall_E+1:n_roof_W,HRAB_ROOF.LA];

fig5=figure(5);
ax21=subplot(2,2,1);
p21=plot(ax21,nodes(i_roof_W,2),axial(i_roof_W));
p21.Marker='o';
p21.LineWidth=2;
p21.MarkerFaceColor=get(p21,'Color');
xlabel(ax21,'X Location (mm)');
ylabel(ax21,'Axial (N)');
grid(ax21,'on');
box(ax21,'on');
ax22=subplot(2,2,2);
p22=plot(ax22,nodes(i_roof_W,2),shear(i_roof_W));
p22.Marker='o';
p22.LineWidth=2;
p22.MarkerFaceColor=get(p22,'Color');
xlabel(ax22,'X Location (mm)');
ylabel(ax22,'Shear (N)');
grid(ax22,'on');
box(ax22,'on');
ax23=subplot(2,2,[3,4]);
p23=plot(ax23,nodes(i_roof_W,2),moment(i_roof_W));
p23.Marker='o';
p23.LineWidth=2;
p23.MarkerFaceColor=get(p23,'Color');
xlabel(ax23,'X Location (mm)');
ylabel(ax23,'Moment (N*mm)');
grid(ax23,'on');
box(ax23,'on');

print(fig5,['W Roof Member Forces'],'-dmeta'); %save open figures as explodable .emf
files

print(fig5,'W Roof Member Forces','-dpng');
savefig(fig5,'W Roof Member Forces'); %Save the figure as a Matlab .Fig file

% E Roof
n_roof_E=n_roof_W+HRAB_ROOF.ele-1; %End node of the W strut
i_roof_E=[HRAB_ROOF.RA,n_roof_W+1:n_roof_E,n_wall_E];

fig6=figure(6);
ax21=subplot(2,2,1);
p21=plot(ax21,nodes(i_roof_E,2),axial(i_roof_E));
p21.Marker='o';
p21.LineWidth=2;
p21.MarkerFaceColor=get(p21,'Color');
xlabel(ax21,'X Location (mm)');
ylabel(ax21,'Axial (N)');
grid(ax21,'on');
box(ax21,'on');
ax22=subplot(2,2,2);
p22=plot(ax22,nodes(i_roof_E,2),shear(i_roof_E));
p22.Marker='o';
p22.LineWidth=2;
p22.MarkerFaceColor=get(p22,'Color');
xlabel(ax22,'X Location (mm)');
ylabel(ax22,'Shear (N)');
grid(ax22,'on');
box(ax22,'on');
ax23=subplot(2,2,[3,4]);
p23=plot(ax23,nodes(i_roof_E,2),moment(i_roof_E));
p23.Marker='o';
p23.LineWidth=2;
p23.MarkerFaceColor=get(p23,'Color');
xlabel(ax23,'X Location (mm)');
ylabel(ax23,'Moment (N*mm)');
grid(ax23,'on');
box(ax23,'on');

print(fig6,'E Roof Member Forces','-dmeta'); %Save open figures as explodable .emf files

print(fig6,'E Roof Member Forces','-dpng');
savefig(fig6,'E Roof Member Forces'); %Save the figure as a Matlab .Fig file

% W Strut
n_strut_W=n_roof_E+2*HRAB_STRUT.ele-1; %End node of the W strut
i_strut_W=[n_wall_W,n_roof_E+1:n_strut_W,HRAB_STRUT.LA];

fig7=figure(7);
ax21=subplot(2,2,1);
p21=plot(ax21,nodes(i_strut_W,2),axial(i_strut_W));
p21.Marker='o';
p21.LineWidth=2;
p21.MarkerFaceColor=get(p21,'Color');
xlabel(ax21, 'X Location (mm)');
ylabel(ax21, 'Axial (N)');
grid(ax21, 'on');
box(ax21, 'on');

ax22 = subplot(2, 2, 2);
p22 = plot(ax22, nodes(i_strut_W, 2), shear(i_strut_W));
p22.Marker = 'o';
p22.LineWidth = 2;
p22.MarkerFaceColor = get(p22, 'Color');
xlabel(ax22, 'X Location (mm)');
ylabel(ax22, 'Shear (N)');
grid(ax22, 'on');
box(ax22, 'on');

ax23 = subplot(2, 2, [3, 4]);
p23 = plot(ax23, nodes(i_strut_W, 2), moment(i_strut_W));
p23.Marker = 'o';
p23.LineWidth = 2;
p23.MarkerFaceColor = get(p23, 'Color');
xlabel(ax23, 'X Location (mm)');
ylabel(ax23, 'Moment (N*mm)');
grid(ax23, 'on');
box(ax23, 'on');

print(fig7, ['W Strut Member Forces', '-dmeta']); %save open figures as explodable .emf files
print(fig7, ['W Strut Member Forces', '-dpng']);
savefig(fig7, ['W Strut Member Forces']); %save the figure as a Matlab .Fig file

% E Strut
n_strut_E = n_strut_W + 2*HRAB_STRUT.ele - 1; %End node of the W strut
i_strut_E = [n_wall_E, n_strut_W+1:n_strut_E, HRAB_STRUT.RA];

fig8 = figure(8);
ax21 = subplot(2, 2, 1);
p21 = plot(ax21, nodes(i_strut_E, 2), axial(i_strut_E));
p21.Marker = 'o';
p21.LineWidth = 2;
p21.MarkerFaceColor = get(p21, 'Color');
xlabel(ax21, 'X Location (mm)');
ylabel(ax21, 'Axial (N)');
grid(ax21, 'on');
box(ax21, 'on');

ax22 = subplot(2, 2, 2);
p22 = plot(ax22, nodes(i_strut_E, 2), shear(i_strut_E));
p22.Marker = 'o';
p22.LineWidth = 2;
p22.MarkerFaceColor = get(p22, 'Color');
xlabel(ax22, 'X Location (mm)');
ylabel(ax22, 'Shear (N)');
grid(ax22, 'on');
box(ax22, 'on');

ax23 = subplot(2, 2, [3, 4]);
p23 = plot(ax23, nodes(i_strut_E, 2), moment(i_strut_E));
p23.Marker='o';
p23.LineWidth=2;
p23.MarkerFaceColor=get(p23,'Color');
xlabel(ax23,'X Location (mm)');
ylabel(ax23,'Moment (N*mm)');
grid(ax23,'on');
box(ax23,'on');

print(fig8,['E Strut Member Forces','','-dmeta']); %save open figures as explodable .emf
print(fig8,['E Strut Member Forces','','-dpng']);
savefig(fig8,['E Strut Member Forces']); %save the figure as a Matlab .Fig file

close all
clear global
function [] = InputFileGenerator_HZ(HRAB,HZ,Params)
%Generates the input files for the PAA analysis for the different
%parameters

% SnowLoad_name=10; %psf, snow load, NOT LOOPED
%     SnowLoad=4.788e-5*SnowLoad_name; %MPa, snow load
PT=Params.PT; %N, pretension in guy lines
P_name=Params.P_name; %MPa, internal pressure of airbeam
P=P_name.*0.00689476; %MPa, internal pressure of airbeam
Angle=Params.Angle; %deg, angle between the ground and the guy
Wall_Height=Params.Wall_Height; %mm, height of the wall
    GUY_X=Wall_Height./tand(Angle); %mm, X position of guy anchor point for the angle
Spacing_name=Params.Spacing_name; %ft, Arch spacing
    Spacing=Spacing_name.*304.8; %mm

    LineLoad=SnowLoad.*Spacing;
% This code is to find the appropriate G values for the different
% pressures. These values are from Table 3 of Brayley's journal paper
G1=[210 263 304 339 369]; %N/mm
P1=[69 138 207 276 345]; %kPa
G=interp1(P1,G1,P.*1000,'linear','extrap'); %linearly extrapolates values outside the range

% This code is to find the appropriate E values for the different
% pressures, these values are from table 6.45 of Brayley's thesis
E1=[39 43 46 49 52]; %N/mm
P1=[69 138 207 276 345]; %kPa
E=interp1(P1,E1,P.*1000,'linear','extrap'); %linearly extrapolates values outside the range

for i=1:length(P)
    for j=1:length(PT)
        for m=1:length(Spacing)
            for k=1:length(GUY_X)
                if HRAB==1
                    % HRAB Props File
                FNAME_HRAB=['HRAB Props_','num2str(P_name(i)),'_',num2str(PT(j)),'_','num2str(Angle(k)),'_','num2str(Spacing_name(m)),'._',dat];
                copyfile('HRAB Props.dat',FNAME_HRAB);
                % Change the Arch Spacing and internal Pressure
                fid=fopen(FNAME_HRAB,'r');
                % Read in the .dat file into a cell array
                l = 1;
                tline = fgetl(fid);
                A{l,1} = tline;
                while ischar(tline)
                    l = l+1;
                    tline = fgetl(fid);
                    A{l,1} = tline;
                end
                fclose(fid);
            end
        end
    end
end
% Change the Roof Load spacing
    A{57,1} = [num2str(LineLoad(m))];
% Get roof attachment nodes for adjust the DAT file
% Loads later
    LA=str2double(A{39,1}); % Left roof airbeam attachment node
    RA=str2double(A{42,1}); % Right roof airbeam attachment node
if HZ==1
    A{77,1}=num2str(0.1);
else
    A{77,1}=num2str(0);
end
% Write cell A into txt
    fid = fopen(FNAME_HRAB, 'w');
    for l = 1:numel(A)
        if A{l+1} == -1
            fprintf(fid,'%s', A{l});
            break
        else
            fprintf(fid,'%s
', A{l});
        end
    end
    fclose(fid); %close the file
    clear A
end
% Dat File
    FNAME_dat=[num2str(P_name(i)),'_',num2str(PT(j)),'_',num2str(Angle(k)),'_',num2str(Spacing_name(m)),'.dat'];
copyfile('20psi_20ft.dat',FNAME_dat);
% Change the guy node location and internal Pressure
    fid=fopen(FNAME_dat,'r');
% Read in the .dat file into a cell array
    l = 1;
    tline = fgetl(fid);
    A{l,1} = tline;
    while ischar(tline)
        l = l+1;
        tline = fgetl(fid);
        A{l,1} = tline;
    end
    fclose(fid);
% Change the Internal Pressure
    A{9,1} = sprintf('%d',P(i));
% Change the Fabric Modulus (E)
    A{13,1} = sprintf('%d',E(i));
% Change the Fabric Shear Modulus (G)
    A{15,1} = sprintf('%d',G(i));
% Change the GUY Pretensions
    guyE=str2num(A{241,1}); % E Guy Props
    guyE(4)=PT(j);
    guyW=str2num(A{242,1}); % W Guy Props
    guyW(4)=PT(j);
    A{241,1} = [num2str(guyE(1)),' ',num2str(guyE(2)),' ',num2str(guyE(3)),' ',num2str(guyW(1)),' ',num2str(guyW(2)),' ',num2str(guyW(3)),' '];
',num2str(guyE(3)),',num2str(guyE(4))];
A{242,1} = [num2str(guyW(1)),',num2str(guyW(2)),'',num2str(guyW(3)),',num2str(guyW(4))];
% Change the GUY anchor location
span=str2num(A{242,1}); %mm
values=cell2mat(A(158));
space_ind=strfind(values,'');
span=str2double(values(1:space_ind(1)-1)); % span length
A{159,1} = [num2str(-GUY_X(k)),',num2str(0)];
A{160,1} = [num2str(GUY_X(k)+span),',num2str(0)];
if HZ==1
% Apply the horizontal load to the Total_F variable.
for temp=247:367
    values=cell2mat(A(temp));
    space_ind=strfind(values,'');
    num1=temp+1:space_ind(1)-1; % node number
    num2=temp+space_ind(1):space_ind(2)-1; % horizontal load
    num3=temp+space_ind(2):end; % vertical load
end
else % Apply a perturbing force at the apex of the arch
    % This force is a % of the total vertical applied load
    for temp=247:367
        values=cell2mat(A(temp));
        space_ind=strfind(values,'');
        num1=temp+1:space_ind(1)-1; % node number
        num2=temp+1:space_ind(2)-1; % horizontal load
        num3=temp+1:space_ind(2)+1:end; % vertical load
    end
end
% Apply perturbing force to apex of arch
num2(61,1)=sum(num3(:))*-0.01/100;
for l=1:length(num1)
    values=[num2str(num1(l,1)),',',num2str(num2(l,1)),',
            num2str(num3(l,1))];
    temp=246+l;
    A{temp,1}=values;
end

% Write cell A into txt
fid = fopen(FNAME_dat,'w');
for l = 1:numel(A)
    if A{l+1} == -1
        fprintf(fid,'%s', A{l});
        break
    else
        fprintf(fid,'%s
', A{l});
    end
end
fclose(fid); %close the file
clear A

% PAA file
FNAME_paa=[num2str(P_name(i)),'_',num2str(PT(j)),'_',num2str(Angle(k)),'_',num2str(Spacing_name(m)),'.paa'];
copyfile('20psi_20ft.paa',FNAME_paa);
% Change the Arch Spacing and internal Pressure
fid=fopen(FNAME_paa,'r');
% Read in the .dat file into a cell array
l = 1;
tline = fgetl(fid);
A{l,1} = tline;
while ischar(tline)
    l = l+1;
tline = fgetl(fid);
    A{l,1} = tline;
end
fclose(fid);
% Change the Internal Pressure
A{16,1} = sprintf('%d',P(i));
% Change the Fabric Modulus (E)
A{23,1} = sprintf('%d',E(i));
% Change the Fabric Shear Modulus (G)
A{25,1} = sprintf('%d',G(i));
% Change the Arch Spacing
A{31,1} = sprintf('%d',Spacing(m));
% Change the Guy Pretension
A{39,1} = sprintf('%d',PT(j));

% Write cell A into txt
fid = fopen(FNAME_paa, 'w');
for l = 1:numel(A)

189
if \( A[l+1] == -1 \)
    fprintf(fid, '%s', A[l]);
    break
else
    fprintf(fid, '%s
', A[l]);
end
fclose(fid); % close the file
clear A

% Make a directory and move the input files into it
mkdir(FNAME_dat(1:end-4));
    home=cd(FNAME_dat(1:end-4));
    mkdir('Original Input Files');
    cd(home);
    movefile(FNAME_dat,[FNAME_dat(1:end-4),'\Original Input Files']);
    movefile(FNAME_paa,[FNAME_dat(1:end-4),'\Original Input Files']);
    if HRAB==1
        copyfile(FNAME_HRAB,FNAME_dat(1:end-4));
        movefile(FNAME_HRAB,[FNAME_dat(1:end-4),'\Original Input Files']);
    end
end
function load_model_HRAB(file_name, all_handles)

global ANALYSIS_PARAMS_I;
global MODEL_PARAMS_I;
global MESH_DATA_I;
global LOAD_DATA_I;
global SNOW_DATA_I;
global WIND_DATA_I;
global ADD_DL_I;
global ADD_UNIF_DL_I;

fid = fopen(file_name);

% read basic input
values = get_line(fid);
ANALYSIS_PARAMS_I.linear = values;
values = get_line(fid);
ANALYSIS_PARAMS_I.beam = values;
values = get_line(fid);
ANALYSIS_PARAMS_I.num_steps = values;
values = get_line(fid);
ANALYSIS_PARAMS_I.diameter = values;
values = get_line(fid);
ANALYSIS_PARAMS_I.wrinkle_depth = values;
values = get_line(fid);
ANALYSIS_PARAMS_I.max_displ = values;
values = get_line(fid);
ANALYSIS_PARAMS_I.pressure = values;

% read save the model geometry, etc
values = get_line(fid);
MODEL_PARAMS_I.arch_type = values;
values = get_line(fid);
MODEL_PARAMS_I.fabric_E = values;
values = get_line(fid);
MODEL_PARAMS_I.fabric_G = values;
values = get_line(fid);
MODEL_PARAMS_I.arch_radius = values;
values = get_line(fid);
MODEL_PARAMS_I.beam_span = values;
values = get_line(fid);
MODEL_PARAMS_I.arch_spacing = values;
values = get_line(fid);
MODEL_PARAMS_I.include_guys = values;
values = get_line(fid);
MODEL_PARAMS_I.guy_location = values;
values = get_line(fid);
MODEL_PARAMS_I.guy_EA = values;
values = get_line(fid);
MODEL_PARAMS_I.guy_stretch = values;
values = get_line(fid);
MODEL_PARAMS_I.pinned = values;
values = get_line(fid);
MODEL_PARAMS_I.tie_modulus = values;
values = get_line(fid);
MODEL_PARAMS_I.include_straps = values;
values = get_line(fid);
MODEL_PARAMS_I.num_straps = values;
values = get_line(fid);
MODEL_PARAMS_I.strap_diameter = values;
values = get_line(fid);
MODEL_PARAMS_I.strap_E = values;
values = get_line(fid);
MODEL_PARAMS_I.bias_angle = values;
values = get_line(fid);
MODEL_PARAMS_I.strap_thickness = values;
num_straps = MODEL_PARAMS_I.num_straps;
for (i = 1:num_straps)
    values = get_line(fid);
    MODEL_PARAMS_I.angle_width(i,:) = values;
end
values = get_line(fid);
MODEL_PARAMS_I.mesh_type = values;
values = get_line(fid);
MODEL_PARAMS_I.numels = values;
values = get_line(fid);
MODEL_PARAMS_I.user_geometry = values;
values = get_line(fid);
MODEL_PARAMS_I.user_geometry_exists = values;
tline = fgetl(fid);
MODEL_PARAMS_I.control_points = tline;
if (MODEL_PARAMS_I.user_geometry == 1) && (MODEL_PARAMS_I.user_geometry_exists)
    the_control_points = load(MODEL_PARAMS_I.control_points);
    MODEL_PARAMS_I.the_control_points = the_control_points;
end

% generic load data
values = get_line(fid);
LOAD_DATA_I.DL_factor = values;
values = get_line(fid);
LOAD_DATA_I.SL_factor = values;
values = get_line(fid);
LOAD_DATA_I.WL_factor = values;
values = get_line(fid);
LOAD_DATA_I.self_wt = values;
values = get_line(fid);
LOAD_DATA_I.I_factor_snow = values;
values = get_line(fid);
LOAD_DATA_I.I_factor_wind = values;
values = get_line(fid);
LOAD_DATA_I.Ce = values;

% snow load data
values = get_line(fid);\nSNOW_DATA_I.balanced = values;
values = get_line(fid);
SNOW_DATA_I.Pg = values;
values = get_line(fid);
SNOW_DATA_I.Ct = values;

% wind load data
values = get_line(fid);
WIND_DATA_I.enclosed = values;
values = get_line(fid);
WIND_DATA_I.positive_pressure = values;
values = get_line(fid);
WIND_DATA_I.wind_speed = values;

% additional DL data
values = get_line(fid);
umloads = values;
for (i = 1:numloads)
    values = get_line(fid);
    ADD_DL_I.addload_x(i,1) = values(1);
    ADD_DL_I.addload_mag(i,1) = values(2);
end

add_unif_load = get_line(fid);
ADD_UNIF_DL_I.add_unif_load = add_unif_load;
values = get_line(fid);
if (values(1) < 0)
    values(1) = 0;
endif
if (values(2) > MODEL_PARAMS_I.beam_span)
    values(2) = MODEL_PARAMS_I.beam_span;
endif
ADD_UNIF_DL_I.add_unif_load_position(1) = values(1);
ADD_UNIF_DL_I.add_unif_load_position(2) = values(2);

fclose(fid);

% now, we must assign values to the GUI
% set_gui_values(all_handles);
function [FLoad, FLF] = Load_Optimize_BisectionMethod(HRAB, Hz, FName)

% Run Analysis to try and get a max LF close to 1
% FName = name of the folder that the test is running

Domain = [0.127, 20.127]*47.88; % N/m^2, domain over which the answer lies.

if HRAB == 1
    % Get initial applied load
    fid = fopen(['HRAB_Props_','FName', '.dat'], 'r');
    % Read in the .dat file into a cell array
    l = 1;
    tline = fgetl(fid);
    A{l,1} = tline;
    while ischar(tline)
        l = l+1;
        tline = fgetl(fid);
        A{l,1} = tline;
    end
    fclose(fid);
    Load(1,1) = str2double(A{48,1}); % N/m^2, includes roof weight

    % Run Analysis and get first LF
    % I know the
    % Run Analysis until we get close to 1. If go over decrease Load by
    % half
    a(1,1) = Domain(1);
    b(1,1) = Domain(2);

    y_a(1,1) = HRAB_Analysis_Opti(a(1,1), FName, 1, Hz);
    y_b(1,1) = HRAB_Analysis_Opti(b(1,1), FName, 1, Hz);
    m(1,1) = (a+b)/2;

    if y_a(1,1) == 0 % Model is unstable under dead load/ guy pretension
        Load = a;
        LF = y_a;
        FLoad = Load(end);
        FLF = LF(end);
        save('LF_Load.mat', 'LF', 'Load')
        return
    end

    i = 1;
    tolerance = 0.02;

    y_m(1,1) = HRAB_Analysis_Opti(m(1,1), FName, 1, Hz);
    LF(i,1) = y_m(1,1); Load(i,1) = m(1,1);
    while (1-y_m(i,1)) > tolerance || (1-y_m(i,1)) == 0
        i = i+1;
        if y_m(i-1,1) == 0
            a(i,1) = m(i-1,1);
            y_a(i,1) = y_m(i-1,1);

            b(i,1) = b(i-1,1);
        else
            a(i,1) = y_m(i-1,1);
            y_a(i,1) = y_m(i-1,1);

            b(i,1) = b(i-1,1);
        end
    end
end
\[ y_{b}(i,1) = y_{b}(i-1,1); \]
\[ \text{else} \]
\[ b(i,1) = m(i-1,1); \]
\[ y_{b}(i,1) = y_{m}(i-1,1); \]
\[ a(i,1) = a(i-1,1); \]
\[ y_{a}(i,1) = y_{a}(i-1,1); \]
\[ \text{end} \]
\[ m(i,1) = (a(i,1) + b(i,1))/2; \]
\[ y_{m}(i,1) = \text{HRAB\_Analysis\_Opti}(m(i,1), FName, 1, Hz); \]
\[ \text{if} \ y_{m}(i,1) > 1 \]
\[ y_{m}(i,1) = 1; \]
\[ \text{end} \]
\[ LF(i,1) = y_{m}(i,1); \]
\[ \text{Load}(i,1) = m(i,1); \]
\[ p_{\text{diff}} = (LF(i,1) - LF(i-1))/\text{mean}([LF(i), LF(i-1)]); \]
\[ \text{if} \ \text{abs}(p_{\text{diff}}) < 0.001 \ \&\& \ LF(i,1) > 0.9 \ \&\& \ LF(i,1) \neq 1 \]
\[ \text{break} \]
\[ \text{end} \]
\[ \text{end} \]
\[ \text{end} \]
\[ \text{if} \ \text{HRAB} = 0 \]
\[ \% \ \text{Get initial applied load} \]
\[ \text{fid} = \text{fopen}(['\text{Original Input Files/}', FName, '.paa'], 'r'); \]
\[ \% \ \text{Read in the .dat file into a cell array} \]
\[ l = 1; \]
\[ tline = \text{fgetl}(\text{fid}); \]
\[ A{l,1} = tline; \]
\[ \text{while} \ \text{ischar}(tline) \]
\[ l = l + 1; \]
\[ tline = \text{fgetl}(\text{fid}); \]
\[ A{l,1} = tline; \]
\[ \text{end} \]
\[ \text{fclose}(\text{fid}); \]
\[ \text{Load}(1,1) = \text{str2double}(A{93,1}); \%N/m^2, \text{includes roof weight} \]
\[ \% \ \text{Run Analysis and get first LF} \]
\[ \% \ \text{I know the} \]
\[ \% \ \text{Run Analysis until we get close to 1. If go over decrease Load by} \]
\[ \% \ \text{half} \]
\[ a(1,1) = \text{Domain}(1); \]
\[ b(1,1) = \text{Domain}(2); \]
\[ y_{a}(1,1) = \text{PAA\_Analysis\_Opti}(a(1,1), FName, 0, Hz); \]
\[ y_{b}(1,1) = \text{PAA\_Analysis\_Opti}(b(1,1), FName, 0, Hz); \]
\[ m(1,1) = (a(1,1) + b(1,1))/2; \]
\[ \text{if} \ y_{a}(1,1) = 0 \% \ \text{Model is unstable under dead load/ guy pretension} \]
\[ \text{Load} = a; \]
\[ \text{LF} = y_{a}; \]
\[ \text{FLoad} = \text{Load}(\text{end}); \]
\[ \text{FLF} = \text{LF}(\text{end}); \]
save('LF_Load.mat','LF','Load')
return
end
i=1;
tolerance=0.02;
y_m(1,1)=PAA_Analysis_Opti(m(1,1),FName,0,Hz);
LF(i,1)=y_m(1,1); Load(i,1)=m(1,1);
while (1-y_m(i,1))>tolerance || (1-y_m(i,1))==0
i=i+1;
if y_m(i-1,1)==1
  a(i,1)=m(i-1,1);
y_a(i,1)=y_m(i-1,1);
  b(i,1)=b(i-1,1);
y_b(i,1)=y_b(i-1,1);
else
  b(i,1)=m(i-1,1);
  y_b(i,1)=y_m(i-1,1);
  a(i,1)=a(i-1,1);
  y_a(i,1)=y_a(i-1,1);
end
m(i,1)=(a(i,1)+b(i,1))/2;
y_m(i,1)=PAA_Analysis_Opti(m(i,1),FName,0,Hz);
if y_m(i,1)>1
  y_m(i,1)=1;
end
LF(i,1)=y_m(i,1); Load(i,1)=m(i,1);
p_diff=(LF(i)-LF(i-1))/mean([LF(i),LF(i-1)]);
if abs(p_diff)<0.001 && LF(i,1)>0.9 && LF(i,1)~=1
  break
end
end
FLoad=Load(end);
FLF=LF(end);
save('LF_Load.mat','LF','Load')
end
function main_solver_HRAB(input_file, output_file, bar);

clear BASIC_PROPS;
clear STRAP_DATA;
clear NODES;
clear UPDATED_NODES;
clear CONNECTIVITIES;
clear BOUNDARIES;
clear TOTAL_F;
clear F;
clear M_kappa;
clear U;
clear MEMBER_FORCES;
clear STRESSES;
clear CAPACITY_DATA;
clear ELS_AT_NODES;
clear LOCAL_NODES;
clear ALL_RESULTS;
clear P_V_M;
clear LOAD_STEPS;
clear LARGE_DEF;
clear REACTIONS;
clear GUYS;
clear GUY_FORCES;
clear ARC_LENGTH;
clear LAST_GOOD_DELU;
clear LARGE_CURVATURE;
clear STRAP_MODULAR_RATIO;
clear SPRING_SUPPORTS;
clear PROPS;

global BASIC_PROPS;
global NODES;
global UPDATED_NODES;
global CONNECTIVITIES;
global BOUNDARIES;
global TOTAL_F;
global DL_F;
global F;
global M_kappa;
global U;
global MEMBER_FORCES;
global STRESSES;
global CAPACITY_DATA;
global GUYS
global GUY_FORCES
global REACTIONS
global LOCAL_NODES;
global STRAP_DATA;
global ELS_AT_NODES;
global ALL_RESULTS;
global P_V_M;
global LARGE_DEF;
global ARC_LENGTH;
global LAST_GOOD_DELU;
global LARGE_CURVATURE;
global STRAP_MODULAR_RATIO;
global SOLVER_TOLERANCE;
global SPRING_SUPPORTS;
global iterations

global n

GUYS = zeros(1,1);

SOLVER_TOLERANCE = 1e-3;

STRAP_MODULAR_RATIO = 0.0;
LARGE_CURVATURE = 0;
bar = waitbar(0, 'Reading Input Data, Defining Parameters');

% start with load stepping
ARC_LENGTH = 0;

% open main file for output
out_fid = fopen(output_file, 'wt');

% read input data, set up model, echo input data
read_input(input_file, out_fid);

% Run the HRAB script
HRAB_2nodemesh(input_file,1)
iterations=25;

% open other files for output formatted for easy plotting
[loadsteps_fid, results_fid, reaction_fid] = ...
open_files(output_file);

% generate ELS_AT_NODES, list of elements connected to each node and
% LOCAL_NODES, list of corresponding element local nodes
generate_els_nodes_HRAB;

linear = BASIC_PROPS(1);
p = BASIC_PROPS(2);
r = BASIC_PROPS(3)/2;
alpha = BASIC_PROPS(6);
bias_angle = BASIC_PROPS(13)*p*pi/180;

% generate pressure-dependent moment-curvature relationship
number_axial = 10;
crush_load = -0.99*p*pi*r^2;
woven = BASIC_PROPS(9);
if (woven == 2) % this beam is braided
    crush_load = crush_load + 0.99*2*p*pi*r^2*(cot(bias_angle))^2;
    braided_m_kappa_params;
end
max_tension = -crush_load;
force_range = [0.5*crush_load, 0.5*max_tension];
max_curvature = 0.006;
num_curvatures = 50;
generate_m_phi(max_curvature, num_curvatures, number_axial, force_range, 0);

% our stopping criterion
max_iter = 30*BASIC_PROPS(7); % maximum number of load steps
largest_allow_displ = BASIC_PROPS(8); % largest allowable displacement

P_OK = 1; % make sure that axial load does not exceed P
alpha_OK = 1; % wrinkle depth check
displ_OK = 1; % maximum displacement is 1 diameter
F_OK = 1; % applied load cannot exceed specified load
del_F_OK = 1; % load increment cannot get too small
% we also check that norm(del_F)/norm(TOTAL_F) > 0.001
iter = 1;

last_good_U = zeros(size(TOTAL_F));
U = zeros(size(TOTAL_F));
update_nodes(U);
last_fraction = 0;
numels = size(CONNECTIVITIES,1);

% update waitbar, solve for DL
waitbar(0, bar, 'Solving for Dead Load + Guy Pre-tension');
% starting force increment and initial load vector
F = DL_F;
initial_incr = 1.0/BASIC_PROPS(7);
force_fraction = 1.0/BASIC_PROPS(7);
fraction_incr = initial_incr;
orig_arc_incr = initial_incr;
arc_incr = orig_arc_incr;

s = sprintf('
 Solving for DL + guy pre-tension
');
disp(s);

solve_for_DL_HRAB;
% save results
accepted = 1;
LOAD_STEPS(accepted) = 0;
delta_V(accepted) = generate_volume;
save_results_HRAB(out_fid, results_fid, reaction_fid, 0, delta_V);

s = sprintf('
 Successfully solved for DL + guy pre-tension
');
disp(s);

s = sprintf('
 Now solving for Wind + Snow + Additional Loads
');
disp(s);

waitbar(0, bar, 'Fraction of Wind + Snow + Additional Loads Applied');
F = force_fraction*TOTAL_F + DL_F;
last_good_U = U;
newton_iter = 0;
not_solved = 0;
the_total_iter = 0;
arc_iter_good = 0;
max_displ = largest_allow_displ;

while ((iter <= max_iter) && (P_OK) && (alpha_OK) && (displ_OK) ...
    && (F_OK) && (del_F_OK) && (not_solved < 20))
    % try for a solution
    if (ARC_LENGTH == 1) % use the arc length solver
        [OK, fraction_incr] = arc_length_solver_HRAB(last_good_U,...
            last_fraction, arc_incr);
        % check for convergence with very large max displ -- happens
        new_max_displ = get_max_displ;
        if (new_max_displ > 2*max_displ)
            OK = 0;
        end
    end
    if (OK == 0) % go to a smaller arc increment
        arc_incr = 2/3*arc_incr;
        not_solved = not_solved + 1;
        arc_iter_good = 0;
    else % good, increment the force fraction
        force_fraction = force_fraction + fraction_incr;
        not_solved = 0;
        arc_iter_good = arc_iter_good + 1;
        if (arc_iter_good >= 5) && (fraction_incr < 0) % try to increase the arc_incr
            % THIS LINE INCREASES LOAD INCREMENT POST PEAK
            arc_incr = 1.5*arc_incr;
            arc_iter_good = 0;
        end
    end
else
    [OK, n] = newton_solver_HRAB(last_good_U, SOLVER_TOLERANCE, iterations, 0, 1);
    the_total_iter = the_total_iter + n;
    if (OK == 0) % increment number of times Newton's method failed
        newton_iter = newton_iter + 1;
    end
end
if (OK == 0) % we did not get a solution
    s = sprintf(' Solution not achieved');
    disp(s);
    U = last_good_U;
    %if (newton_iter > 0) % we will move to an arc-length solver
    % THIS LINE ALLOWS SUBDIVISION OF LOAD INCREMENT IF > 0
    if (newton_iter > 0) && (LARGE_DEF == 1) % we will move to an arc-length solver
        ARC_LENGTH = 1;
        force_fraction = last_fraction;
s = sprintf(' Using arc length solver');
disp(s);
else % we are still sub-dividing load steps
    s = sprintf(' Sub-dividing load step');
disp(s);
    fraction_incr = fraction_incr/2;
    force_fraction = last_fraction + fraction_incr;
    %orig_arc_incr = fraction_incr;
    %last_fraction = initial_incr;
end
else % we will accept the results, save them, check other criteria
    accepted = accepted + 1;
    ndof = max(size(U));
    apex_hz_displ(accepted) = U((numels+1)*3-2);
    apex_vert_displ(accepted) = -U((numels+1)*3-1);

    s = sprintf(' Successfully solved for load fraction = %g', force_fraction);
disp(s);
    waitbar(force_fraction);
    fract(accepted) = force_fraction;
    LOAD_STEPS(accepted) = force_fraction;
    last_good_U = U;

    max_axial = max(CAPACITY_DATA(:,1));
    max_wrinkle = max(CAPACITY_DATA(:,3));
    %max_displ = max(abs(U));
    x_displ1 = U(1:3:ndof);
    y_displ1 = U(2:3:ndof);
    max_displ1 = max(sqrt(x_displ1.^2 + y_displ1.^2));

    P_min = -max(MEMBER_FORCES(:,1));
    P_max = -min(MEMBER_FORCES(:,1));

    if isempty(max_axial)==1
        if (max_axial > 1.0) && (linear == 2) % we have exceeded axial pressure resultant
            P_OK = 0;
        end
    end

    if (max_wrinkle > alpha*(2*r)) % wrinkling criterion exceeded
        alpha_OK = 0;
    end

    if (max_displ1 >= largest_allow_displ1) % max. displacement too large
        displ_OK = 0;
    end

    % get volume change
    delta_V(accepted) = generate_volume;

    % save results
    save_results_HRAB(out_fid, results_fid, reaction_fid, ...
force_fraction, delta_V(accepted));

last_fraction = force_fraction;
if (ARC_LENGTH == 0) % here, we have to increment the force fraction
    force_fraction = force_fraction + fraction_incr;
end

if (force_fraction > 1.001) % we will exceed maximum specified load
    F_OK = 0;
end

if (newton_iter >= 15) || (arc_incr/orig_arc_incr < 0.01) % our load increment is too small
del_F_OK = 0;
end

% update pressure
pressure_result = pi*p*r^2;
% we did not solve on first load step, or this is the first load step
if (((OK == 0) && (iter == 1)) || (last_fraction < 1e-6))
    P_min = -0.99*pressure_result;
    P_max = -P_min;
elseif (abs(force_fraction - last_fraction) > 1e-6)
    P_min = min(P_min*force_fraction/last_fraction*1.3, P_min);
    if (P_min < -0.99*pressure_result)
        P_min = -0.99*pressure_result;
    end
    %P_min = max(P_min, -0.99*pressure_result);
    %P_min = P_min*force_fraction/last_fraction - 0.50*abs(P_min);
    P_max = max(P_max*force_fraction/last_fraction*1.3, P_max);
    %P_max = P_max*force_fraction/last_fraction + 0.50*abs(P_max);
end

if ((iter <= max_iter) && (P_OK) && (alpha_OK) && (displ_OK) && (F_OK) && (del_F_OK) && (not_solved < 20)) && isempty(CONNECTIVITIES)==1
    % here, get maximum curvature from last result, use to determine
    % range of curvatures for next iteration
    max_curvature = 3*get_max_curvature;
    %P_min
    %P_max
    generate_m_phi(max_curvature, num_curvatures, number_axial, [P_min, P_max], 0);
end

F = force_fraction*TOTAL_F + DL_F;
iter = iter + 1;
end

% give some information on why analysis terminated
if (F_OK == 0)
    fprintf(out_fid, \n"MAXIMUM SPECIFIED LOAD REACHED\n");disp('MAXIMUM SPECIFIED LOAD REACHED');
End
if (P_OK == 0)
    fprintf(out_fid, 'n* MAXIMUM AXIAL COMPRESSION EXCEEDS PRESSURE RESULTANT\n');
disp('MAXIMUM AXIAL COMPRESSION EXCEEDS PRESSURE RESULTANT');
end
if (alpha_OK == 0)
    fprintf(out_fid, 'n* SPECIFIED WRINKLE DEPTH EXCEEDED\n');
disp('SPECIFIED WRINKLE DEPTH EXCEEDED');
end
if (displ_OK == 0)
    fprintf(out_fid, 'n* SPECIFIED DISPLACEMENT LIMIT EXCEEDED\n');
disp('SPECIFIED DISPLACEMENT LIMIT EXCEEDED');
end
if (iter > max_iter) || (del_F_OK == 0)
    fprintf(out_fid, 'n* UNABLE TO SOLVE WITHIN LOAD DIVISION TOLERANCE\n');
disp('UNABLE TO SOLVE WITHIN LOAD DIVISION TOLERANCE');
end
if (not_solved >= 20)
    fprintf(out_fid, 'n* MORE THAN TWENTY ARC-LENGTH SUBDIVISIONS\n');
disp('MORE THAN TWENTY ARC-LENGTH SUBDIVISIONS');
end
% save load step and volume change information
% Check if model ran, if not place a 0 in the loadstep.
    if exist('fract','var')~=1
        fract=0;
    end
for (i = 1:max(size(fract)))
    fprintf(loadsteps_fid, '%%10.8f  %%10.8f\n', [fract(i), delta_V(i)*100]);
end
% close the output files
fclose(out_fid);
fclose(loadsteps_fid);
fclose(results_fid);
fclose(reaction_fid);
% close the waitbar
close(bar);
end
function [ok,count] = newton_solver_HRAB(initial_displ, tol, max_iter, accept, ...
    show_results);

    global BASIC_PROPS
    global NODES
    global CONNECTIVITIES
    global PROPS
    global F
    global K
    global K_BC
    global U
    global R
    global BOUNDARIES
    global M_kappa;
    global MEMBER_FORCES;
    global CAPACITY_DATA;
    global STRESSES;
    global GUYS;
    global GUY_FORCES;
    global REACTIONS;
    global P_V_M;
    global LARGE_DEF;
    global UPDATED_NODES;

    % we must initialize U here if not specified
    ndof = size(NODES,1)*3;
    if (size(initial_displ,1) < ndof)
        U = zeros(ndof,1);
    else
        U = initial_displ;
    end

    old_updated_nodes = UPDATED_NODES;

    % compute member forces
    curvature_error = compute_member_forces_HRAB;

    % compute the residual force vector, R
    compute_residual;
    init_R = R;

    % our convergence criterion
    % tol = 1e-04;
    error = 1;

    count = 1;

    % start Newton iteration
    %max_iter = 50;
    NR_count = 1;
    NR_update = 1;
    first = 1;
warning('off');

while (error > tol) && (count <= max_iter)
    if (rem(NR_count, NR_update) == 0) || first
        % assemble the tangent stiffness matrix and apply B.C. to K
        assemble_stiff_HRAB;
        apply_boundaries;
        NR_count = 1;
        first = 0;
    end
    NR_count = NR_count + 1;
    % solve for displacement increment
    del_U = K_BC\R;
    old_R = R;
    % increment the displacements
    U = U + del_U;
    % update coordinates if performing a large deformation analysis
    if (LARGE_DEF == 1)
        update_nodes(U);
    end
    % compute member forces
    curvature_error = compute_member_forces_HRAB;
    % compute the residual force vector, R
    compute_residual;
    % convergence is based on force equilibrium
    if (norm(F) > 1e-2)
        error = norm(R)/norm(F);
    else
        %error = norm(R)/norm(init_R);
        error = norm(R);
    end
    %if (count == 1)
    %    init_energy = del_U'*old_R;
    %    error = 1;
    %else
    %    error = del_U'*old_R/init_energy;
    %end
    if (show_results == 1)
        s = sprintf(' error = %g', error);
        disp(s);
    end
    count = count + 1;
end
% new_NODES(:,1)=NODES(:,1)+U(1:3:end);
new_NODES(:,2)=NODES(:,2)+U(2:3:end);
plot_HRAB_2node_mesh(new_NODES,CONNECTIVITIES,GUYS,0)
end

OK = 1;
woven = BASIC_PROPS(9);

if ((error > tol) || isnan(error)) && (accept == 0) % we have not converged, do not accept
  OK = 0;
  MEMBER_FORCES = zeros(size(CONNECTIVITIES,1),6);
  STRESSES = zeros(size(NODES,1),1);  
  CAPACITY_DATA = zeros(size(NODES,1),3);
  if (show_results == 1)
    message = sprintf(' Did not converge in %g iterations, error = %0.3g',...
       count-1, error);
    disp(message);
  end

  % here we must go back to our old node locations
  U = initial_displ;
  if (LARGE_DEF == 1)
    update_nodes(U);
  end

  % compute member forces
  % curvature_error = compute_member_forces;

  % compute the residual force vector, R
  %compute_residual;
else % converged, compute stresses, compute capacity data
  generate_stresses_HRAB;  % max tensile stress at each node, capacity data
  max_wrinkle = max(CAPACITY_DATA(:,3));
  %d = BASIC_PROPS(3);
  %if (max_wrinkle < d/1000)
    % max_wrinkle = 0;
  %end
  max_displ = get_max_displ;
  if (woven == 1)
    message = sprintf(' No. of iterations = %g, error = %0.3g, max. displ = %0.1f, max.
       wrinkle = %0.1f',...
       count-1, error, max_displ, max_wrinkle); 
  else
    message = sprintf(' No. of iterations = %g, error = %0.3g, max. displ = %0.1f',...
       count-1, error, max_displ);
  end
  if (show_results == 1)
    disp(message);
  end
end

warning('on');
end
function [] = Optimize_Generate_Load_File(FName,load,HRAB,HZ)

%Generates the applied vertical loads in the .dat file. and re-writes the %dat file.

% Read in load parameters from the PAA file.
fid=fopen([FName,'.paa'], 'r');
% Read in the .dat file into a cell array
l = 1;
tline = fgetl(fid);
B{l,1} = tline;
while ischar(tline)
    l = l+1;
    tline = fgetl(fid);
    B{l,1} = tline;
end
fclose(fid);

% Rewrite the Current overhead load in the PAA file
B{93,1}=num2str(load);

% Write cell B into txt
fid = fopen([FName,'.paa'], 'w');
for l = 1:numel(B)
    if B{l+1} == -1
        fprintf(fid,'%s', B{l});
        break
    else
        fprintf(fid,'%s
', B{l});
    end
end
fclose(fid);

% Adjust the Roof Load Value

% load=str2double(cell2mat(B(93))); % N/m^2, horizontal load applied
load=load/(1000^2); %N/mm^2

% spacing=str2double(cell2mat(B(31))); %mm, spacing of airbeam

% HRAB PROPS File
% If HRAB = 1, adjust the HRAB Props File
if HRAB==1
    fid=fopen(['HRAB_Props_',FName,'.dat'], 'r');
    % Read in the .dat file into a cell array
    l = 1;
    tline = fgetl(fid);
    A{l,1} = tline;
    while ischar(tline)
        l = l+1;
        tline = fgetl(fid);
        A{l,1} = tline;
    end
    fclose(fid);

    % Adjust the Roof Load Value
A{45,1}=num2str(load*(1000^2)); %N/m^2

% Get roof attachment nodes for adjust the DAT file
% loads later
LA=str2double(A{39,1}); % Left roof airbeam attachment node
RA=str2double(A{42,1}); % Right roof airbeam attachment node

% Write cell A into txt
fid = fopen(['HRAB_Props_','FName,'.dat'], 'w');
for l = 1:numel(A)
    if A{l+1} == -1
        fprintf(fid, '%s', A{l});
        break
    else
        fprintf(fid, '%s
', A{l});
    end
end
fclose(fid); %close the file
end

% DAT File
fid=fopen([FName,'.dat'],'r');
% Read in the .dat file into a cell array
l = 1;
tline = fgetl(fid);
A{l,1} = tline;
while ischar(tline)
    l = l+1;
    tline = fgetl(fid);
    A{l,1} = tline;
end
fclose(fid);

% Gather the nodes to find the del_x
for temp=38:158
    values=cell2mat(A(temp));
    space_ind=strfind(values, ' ');
    nodes(temp-37,1)=str2double(values(1:space_ind(1)-1)); % x location
    nodes(temp-37,2)=str2double(values(space_ind(1)+1:end)); % y location
end

% Gather the elements to find the connectivities
F=zeros(121,2);
for temp=164:223
    i=temp-163;
    values=cell2mat(A(temp));
    space_ind=strfind(values, ' ');
    conns(i,1)=str2double(values(1:space_ind(1)-1)); % start node
    conns(i,2)=str2double(values(space_ind(1)+1:space_ind(2)-1)); % end node
    conns(i,3)=str2double(values(space_ind(2)+1:end)); % middle node
end

% Apply the loads to the F variable, each row of F is a node
del_x=nodes(conns(1,2),1)-nodes(conns(1,1),1);
nodal_load=load*spacing*del_x/2; \%N
% Start Node
F(conns(i,1),2)=nodal_load+F(conns(i,1),2);
% End Node
F(conns(i,2),2)=nodal_load+F(conns(i,2),2);
end

if HZ==1
% Apply the horizontal load to the loads variable.
F(:,1)=F(:,2).*-0.1;
else
F(61,1)=sum(F(:,2))*-0.01/100; \% Arch Perturbing force
end

if HRAB==1
% Delete the Horizontal loads beneath the roof.
F(1:LA-1,1:2)=F(1:LA-1,1:2)*0;
F(RA+1:end,1:2)=F(RA+1:end,1:2)*0;
end

% Write Loads
for temp=247:367
values=cell2mat(A(temp));
space_ind=strfind(values,' ');
num1=str2double(values(1:space_ind(1)-1)); \% node number
values=[num2str(num1),',',num2str(F(temp-246,1)),',',num2str(F(temp-246,2))];
A{temp,1}=values;
end

% Write cell A into txt
fid = fopen([FName,'.dat'], 'w');
for l = 1:numel(A)
if A{l+1} == -1
fprintf(fid,'%s', A{l});
break
else
fprintf(fid,'%s
', A{l});
end
end
fclose(fid); %close the file
end
function [LC_results] = PAA_ANALYSIS(FNAME)
% Runs an analysis of the airbeam structure through PressArchAnalysis
clear global
global PATH_NAME
global MODEL_NAME

copyfile(['Original Input Files\',FNAME,'.dat'])
copyfile(['Original Input Files\',FNAME,'.paa'])

PATH_NAME=strcat(pwd,'\');
MODEL_NAME=[FNAME,’.paa’];

% first, saveas the current data
fe_input_file = [FNAME,’.dat’];
% save_model(fe_input_file); % saves current geometric inputs to input file

% now, create FE input and output files
fe_output_file = strrep(fe_input_file, ’.dat’, ’.out’);
main_solver(fe_input_file, fe_output_file);
LC_results=dlmread([FNAME,’_loadsteps.out’]);
LC_results=LC_results(end,:);
end

function [LF] = PAA_Analysis_Opti(Load,FName,HRAB,Hz)
%HAB Analysis Function to be used in optimization
% Move all input files into current folder
  back=cd(‘Original Input Files’);
  Optimize_Generate_Load_File(FName,Load,HRAB,Hz);
  cd(back);
  [-]=PAA_ANALYSIS(FName);
  LF=dlmread([FName,’_loadsteps.out’]);
  LF=max(LF(:,1));
end
function [fig1] = plot_HRAB_2node_mesh(NODES,CONNECTIVITIES,GUYS,PL,TYPE)
% Plots mesh of the HRAB structure. If wish to plot loads enter 1 for PL value.

% Create matrices of the start, middle, and end nodes
start=NODES(CONNECTIVITIES(:,1),:);
middle=NODES(CONNECTIVITIES(:,3),:);
end_node=NODES(CONNECTIVITIES(:,2),:);

% close all
fig1=figure(1);
ax1=axes(fig1);
hold(ax1,'on')
s1=scatter(ax1,start(:,1),start(:,2),'filled','green');
s1.DisplayName='Start Node';
s11=scatter(ax1,NODES(GUYS(:,1),1),NODES(GUYS(:,1),2),'filled','green');
s11.HandleVisibility='off';
s2=scatter(ax1,end_node(:,1),end_node(:,2),'red');
s2.DisplayName='End Node';
s2.LineWidth=1.5;
s22=scatter(ax1,NODES(GUYS(:,2),1),NODES(GUYS(:,2),2),'red');
s22.LineWidth=1.5;
s22.HandleVisibility='off';
s3=scatter(ax1,middle(:,1),middle(:,2),'filled');
s3.DisplayName='Middle Node';
s3.MarkerFaceColor=[0.9290, 0.6940, 0.1250];

for i=1:length(start)
    X=[start(i,1);middle(i,1);end_node(i,1)];
    Y=[start(i,2);middle(i,2);end_node(i,2)];
    line(ax1,X,Y,'HandleVisibility','off')
end

for i=1:length(GUYS(:,1))
    line(ax1,NODES(GUYS(i,[1,2]),1),NODES(GUYS(i,[1,2]),2),'HandleVisibility','off')
end

legend(ax1,'show','Location','south');
axis(ax1,'equal');
xlabel('X coordinate (mm)');
ylabel('Y coordinate (mm)');
grid(ax1,'on')
box(ax1,'on');

if PL==1
    plot_loads_HRAB_GUYS(ax1,TYPE)
end

print(fig1,['Discretized Model','-dmeta']); % save open figures as explodable .emf files
print(fig1,['Discretized Model','-dpng']);
savefig(fig1,['Discretized Model']); % save the figure as a Matlab .Fig file
function plot_loads_HRAB_GUYS(ax,load)

    global ANALYSIS_PARAMS_I;  % struct with general analysis parameters
    global MODEL_PARAMS_I;  % struct with model geometry, etc.
    global LOAD_VECTOR_I;
    global MODEL_NAME;
    global HRAB_CONN_I
    global HRAB_NODE_I
    global DL_F
    global GUY'S
    global TOTAL_F

    if strcmp(load,'LIVE')
        load_vector = LOAD VECTOR_I;
    elseif strcmp(load,'DEAD')
        load_vector = DL_F(1:end-6);
    end

    the_nodes = MODEL_PARAMS_I.nodes;
    els = MODEL_PARAMS_I.els;

    all_handles = guihandles;
    axes(ax1);
    clf('reset');
    hold on;
    legend('hide')
    axis equal;
    model_name = strrep(MODEL_NAME, '_', '_');
    title(model_name);

    % establish the same plotting limits as used when plotting mesh
    beam = ANALYSIS_PARAMS_I.beam;
    arch_radius = MODEL_PARAMS_I.arch_radius;
    beam_span = MODEL_PARAMS_I.beam_span;
    radius = ANALYSIS_PARAMS_I.diameter/2;
    include_guys = MODEL_PARAMS_I.include_guys;
    user_geometry = MODEL_PARAMS_I.user_geometry;
    pinned = MODEL_PARAMS_I.pinned;

    % The code below adjusts the figure plot area. That is already done in the
    % Test Rig code
    if (user_geometry == 0)
        if (beam > 1e-4)
            span = beam_span;
            axis([-0.1*beam_span, 1.1*beam_span, -1.2*beam_span/5, 1.2*beam_span/5]);
        else
            axis([-0.1*beam_span, 1.1*beam_span, -1.2*beam_span, 1.2*beam_span]);
        end
    else
        % code for user geometry
    end

end
span = arch_radius;
y_span = arch_radius;
axis([-5*radius, (2*arch_radius+5*radius), -radius, max(1.1*arch_radius, ...
     arch_radius + 5*radius)]);
end
else
    numn = size(the_nodes,1);
y_span = max(the_nodes(:,2));
x_span = max(the_nodes(:,1));
    if (include_guys == 1)
        numn = numn-2;
        x_span = the_nodes(numn,1) - the_nodes(1,1);
    end
    span = max(the_nodes(:,2));
    arch_radius = max([max(the_nodes(:,2)), x_span/2]);
    axis([-5*radius, (x_span+5*radius), 0, max(1.1*y_span, ...
         y_span + 5*radius)]);
end

% establish maximum x-direction and y-direction loads, scale factor
vect_size = max(size(load_vector));
max_x_force = max(abs(load_vector(1:3:vector_size)));
max_y_force = max(abs(load_vector(2:3:vector_size)));

% maximum force plots as 4*cross-sectional radius
max_force = max(1, max(max_x_force, max_y_force));
scale = 4*radius/max_force;
numels = size(els,1);
arow_dim = radius/3;

% strip the middle node loads and add to end nodes, draw the arch CL
for (i = 1:numels)
    % modify loads
    middle_dof = [(els(i,3) - 1)*3 + 1, (els(i,3) - 1)*3 + 2];
    start_dof = [(els(i,1) - 1)*3 + 1, (els(i,1) - 1)*3 + 2];
    end_dof = [(els(i,2) - 1)*3 + 1, (els(i,2) - 1)*3 + 2];
    load_vector(start_dof) = load_vector(start_dof) + ...
                           load_vector(middle_dof)/2;
    load_vector(end_dof) = load_vector(end_dof) + ...
                        load_vector(middle_dof)/2;
    load_vector(middle_dof) = 0;

    % draw CL of arch
    x = [the_nodes(els(i,1),1), the_nodes(els(i,2),1)];
    y = [the_nodes(els(i,1),2), the_nodes(els(i,2),2)];
    h = line(x,y);
    set(h, 'color','k');
    set(h, 'LineStyle', '-');
    set(h, 'LineWidth', 1);

    % draw supports
    if (i == 1)
if (beam < 1e-4) % it is an arch
    if (MODEL_PARAMS_I.pinned > 1e-4) || ((pinned < 1e-4) && (include_guys < 1e-4)) %
        draw a pin
        fill([x(1), x(1)+0.5*radius, x(1)-0.5*radius], ...
             [y(1), y(1)-radius, y(1)-radius], 'b');
    else % roller
        circle_radius = 0.5*radius;
        center_x = x(2);
        center_y = y(2) - radius - 2*circle_radius;
        fill(center_x + cos([0:0.2*pi:2*pi])*circle_radius, ...
             center_y + sin([0:0.2*pi:2*pi])*circle_radius, ...
             'b');
    end
else % it is a beam -- pin
    %fill([x(1), x(1)+0.5*radius, x(1)-0.5*radius], ...
    % [y(1), y(1)-radius, y(1)-radius], 'b');
    fill([x(1), x(1)+0.5*radius, x(1)-0.5*radius], ...
         [y(1), y(1)-radius, y(1)-radius], 'b');
end
elseif (i == HRAB_CONN_I.Airbeam(end))
    if (beam < 1e-4) && (MODEL_PARAMS_I.pinned > 1e-4) % pinned arch
        fill([x(2), x(2)+0.5*radius, x(2)-0.5*radius], ...
             [y(2), y(2)-radius, y(2)-radius], 'b');
    else % draw a circle for a roller
        circle_radius = 0.5*radius;
        center_x = x(2);
        if (beam > 1e-4)
            center_y = y(2) - circle_radius;
        else
            center_y = y(2) - circle_radius;
        end
        fill(center_x + cos([0:0.2*pi:2*pi])*circle_radius, ...
             center_y + sin([0:0.2*pi:2*pi])*circle_radius, ...
             'b');
    end
end

% For the roof element guy lines
for i = 4:size(GUYS(:,1),1)
    % draw CL of arch
    x = [the_nodes(GUYS(i,1),1), the_nodes(GUYS(i,2),1)];
    y = [the_nodes(GUYS(i,1),2), the_nodes(GUYS(i,2),2)];
    h = line(x,y);
    set(h, 'Color','k');
    set(h, 'LineStyle', '-');
    set(h, 'LineWidth', 1);
end

numnodes = max(size(the_nodes));
if (MODEL_PARAMS_I.include_guys == 1)
    numnodes = numnodes - 2;
end
% numnodes = size([HRAB_NODE_I.Airbeam; HRAB_NODE_I.Wall; GUYS(4:end,1); HRAB_NODE_I.Strut],1);
end

xlim = max(the_nodes(:,1)) - min(the_nodes(:,1));
ylim = max(the_nodes(:,2)) - min(the_nodes(:,2));
for (i = 1:numnodes);
  x = the_nodes(i,1);
y = the_nodes(i,2);

  % plot x-direction load -- start at outside of arch
  x_load = load_vector((i-1)*3+1);
middle = xlim/2 + min(the_nodes(:,1));
  if (x >= middle)
    x1 = [x, x + abs(x_load)*scale];
  else
    x1 = [x - abs(x_load)*scale, x];
  end
  y1 = [y y];
h1 = line(x1,y1);

  % draw a little triangle showing direction
  if (abs(x_load) > 1e-6)
    yp = [y+arrow_dim, y, y-arrow_dim];
    if (x_load < 0) % negative
      if (x >= middle) % negative, on right
        xp = [x + 2*arrow_dim, x, x + 2*arrow_dim];
      else % negative, on left
        xp = [x1(2), x1(2) - 2*arrow_dim, x1(2)];
      end
    else % positive
      if (x >= middle) % positive, on right
        xp = [x1(2), x1(2) + 2*arrow_dim, x1(2)];
      else % positive, on left
        xp = [x - 2*arrow_dim, x, x - 2*arrow_dim];
      end
    end
    fill(xp,yp,[1 0 0]);
  end

  % plot y-direction load -- start at top of arch
  y_load = load_vector((i-1)*3+2);
y2 = [y, y + abs(y_load*scale)];
x2 = [x x];
h2 = line(x2,y2);
set(h2, 'Color', [1 0 0]);

  % draw a little triangle showing direction
  if (abs(y_load) > 1e-6)
    xp = [x, x+arrow_dim, x-arrow_dim];
    if (y_load < 0) % downward load
      yp = [y, y+2*arrow_dim, y+2*arrow_dim];
    else % upward load
      yp = [y2(2) + 2*arrow_dim, y2(2), y2(2)];
  end

end
end
fill(xp,yp,[1 0 0]);
end
end
end
function read_input_HRAB(fname, out_fid)

    global HRAB_WALL
    global HRAB_ROOF
    global HRAB_STRUT
    global MODEL_PARAMS_I

    fid = fopen(fname);

    % read inputs regarding the properties of the HRAB structure components
    % Walls
    values = get_line(fid);
    HRAB_WALL.wt = values;  % N/mm, self weight of the wall panels
    values = get_line(fid);
    HRAB_WALL.ele = values;  % number of elements
    values = get_line(fid);
    HRAB_WALL.ht = values;  % mm, height of the wall
    values = get_line(fid);
    HRAB_WALL.E = values;  % MPa, modulus of the wall
    values = get_line(fid);
    HRAB_WALL.A = values;  % mm^2, cross-sectional area of the wall
    values = get_line(fid);
    HRAB_WALL.I = values;  % mm^4, moment of inertia of the wall
    values = get_line(fid);
    HRAB_WALL.G = values;  % MPa, shear modulus of the wall element
    values = get_line(fid);
    HRAB_WALL.gravity = values;  % N/mm, uniform gravity load over the x projection

    % Roof
    values = get_line(fid);
    HRAB_ROOF.wt = values;  % oz/yd^2, self weight
    HRAB_ROOF.wt = HRAB_ROOF.wt*3.325E-7*MODEL_PARAMS_I.arch_spacing;  % N/mm, self weight along length of roof
    values = get_line(fid);
    HRAB_ROOF.ele = values;  % number of elements
    values = get_line(fid);
    HRAB_ROOF.E = values;  % N/mm, Elastic Modulus
    values = get_line(fid);
    HRAB_ROOF.LA = values;  % left roof attachment node to airbeam
    values = get_line(fid);
    HRAB_ROOF.RA = values;  % right roof attachment node to airbeam
    values = get_line(fid);
    HRAB_ROOF.gravity = values;  % N/m^2, Snow Load applied
    HRAB_ROOF.EA=HRAB_ROOF.E*MODEL_PARAMS_I.arch_spacing;

    % Strut
    values = get_line(fid);
    HRAB_STRUT.wt = values;  % N/mm, self weight
    values = get_line(fid);
    HRAB_STRUT.ele = values;  % number of elements
    values = get_line(fid);
    HRAB_STRUT.E = values;  % MPa, Elastic Modulus
values = get_line(fid);
HRAB_STRUT.A = values;  % mm^2, cross-sectional area of the strut
values = get_line(fid);
HRAB_STRUT.I = values;  % mm^4, moment of inertia of the strut
values = get_line(fid);
HRAB_STRUT.G = values;  % MPa, shear modulus of the wall element
values = get_line(fid);
HRAB_STRUT.LA = values;  % left roof attachment node to airbeam
values = get_line(fid);
HRAB_STRUT.RA = values;  % right roof attachment node to airbeam
values = get_line(fid);
HRAB_STRUT.gravity = values;  % N/mm, uniform gravity load over the x projection

% Additional Model Params
values = get_line(fid);
MODEL_PARAMS_I.HZ = values;  % percent of horizontal force applied to all gravity loads.
fclose(fid);

% Echo parameters in .out file
fprintf(out_fid, '* BEGIN HRAB PARAMS
');
fprintf(out_fid, '* Walls
');
fprintf(out_fid, '*     Wall Self weight, N/mm
');
fprintf(out_fid, '%g
', HRAB_WALL.wt);
fprintf(out_fid, '*     Wall Number of elements
');
fprintf(out_fid, '%g
', HRAB_WALL.ele);
fprintf(out_fid, '*     Wall Height, mm
');
fprintf(out_fid, '%g
', HRAB_WALL.ht);
fprintf(out_fid, '*     Elastic Modulus, MPa
');
fprintf(out_fid, '%g
', HRAB_WALL.E);
fprintf(out_fid, '*     Cross-Sectional Area, mm^2
');
fprintf(out_fid, '%g
', HRAB_WALL.A);
fprintf(out_fid, '*     Moment of Inertia, mm^4
');
fprintf(out_fid, '%g
', HRAB_WALL.I);
fprintf(out_fid, '*     Shear Modulus, MPa
');
fprintf(out_fid, '%g
', HRAB_WALL.G);
fprintf(out_fid, '*     Uniform Gravity Load, N/mm
');
fprintf(out_fid, '%g
', HRAB_WALL.gravity);
fprintf(out_fid, '* Roof
');
fprintf(out_fid, '*     Roof Self weight, N/mm
');
fprintf(out_fid, '%g
', HRAB_ROOF.wt);
fprintf(out_fid, '*     Number of elements
');
fprintf(out_fid, '%g
', HRAB_ROOF.ele);
fprintf(out_fid, '*     Elastic Modulus, N/mm
');
fprintf(out_fid, '%g
', HRAB_ROOF.E);
fprintf(out_fid, '*     Roof Left Attachment Node
');
fprintf(out_fid, '%g
', HRAB_ROOF.LA);
fprintf(out_fid, '*     Roof Right Attachment Node
');
fprintf(out_fid, '%g
', HRAB_ROOF.RA);
fprintf(out_fid, '*     Uniform Gravity Load, N/m^2
');
fprintf(out_fid, '%g
', HRAB_ROOF.gravity);
fprintf(out_fid, '* Strut
');
fprintf(out_fid, '*     Strut Self weight, N/mm
');
fprintf(out_fid, '%g
', HRAB_STRUT.wt);
fprintf(out_fid, '*     Number of elements
');
fprintf(out_fid, '%g
');
fprintf(out_fid, '%g
n', HRAB_STRUT.ele);
fprintf(out_fid, '*     Elastic Modulus, MPa
');
fprintf(out_fid, '%g
n', HRAB_STRUT.E);
fprintf(out_fid, '*     Cross-Sectional Area, mm^2
');
fprintf(out_fid, '%g
n', HRAB_STRUT.A);
fprintf(out_fid, '*     Moment of Inertia, mm^4
');
fprintf(out_fid, '%g
n', HRAB_STRUT.I);
fprintf(out_fid, '*     Shear Modulus, MPa, mm
');
fprintf(out_fid, '%g
n', HRAB_STRUT.G);
fprintf(out_fid, '*     Strut Left Attachment Node
');
fprintf(out_fid, '%g
n', HRAB_STRUT.LA);
fprintf(out_fid, '*     Strut Right Attachment Node
');
fprintf(out_fid, '%g
n', HRAB_STRUT.RA);
fprintf(out_fid, '*     Uniform Snow Load, N/m^2
');
fprintf(out_fid, '%g
n', HRAB_STRUT.gravity);
fprintf(out_fid, '* Additional Model Params
');
fprintf(out_fid, '*     Horizontal Load Percentage, decimal format
');
fprintf(out_fid, '%g
n', MODEL_PARAMS_I.HZ);
% we are done -- close file
fclose(out_fid); %close the echo parameter file
end
function save_results_HRAB(out_fid, results_fid, reaction_fid, ...
    force_fraction, delta_V);

global NODES;
global CONNECTIVITIES;
global TOTAL_F;
global U

global MEMBER_FORCES

global STRESSES

global CAPACITY_DATA

global GUYS

global GUY_FORCES

global REACTIONS

global P_V_M

% load step
fprintf(out_fid, '\n\n* RESULTS FOR LOAD FRACTION %g\n', force_fraction);

% print displacements
fprintf(out_fid, '\n\n* Nodal Displacements\n');
fprintf(out_fid, '\n Node Delta X Delta Y Total Displ. Rotation\n');
fprintf(out_fid, '\n (length) (length) (length) (radians)\n');
numnodes = size(NODES,1);
for (i = 1:numnodes)
    dof1 = (i-1)*3 + 1;
    dof2 = dof1 + 1;
    dof3 = dof2 + 1;
    total_displ = (U(dof1,1)^2 + U(dof2,1)^2)^0.5;
    fprintf(out_fid, '\%5i %8.3f %8.3f %8.3f  %8.6f\n', i, U(dof1,1), U(dof2,1), total_displ, U(dof3,1));
end

% print member forces
fprintf(out_fid, '\n\n* Member End Forces for Each Element\n');
fprintf(out_fid, '\n (force) (force) (force*length) (force) (force) (force*length)\n');
numels = size(MEMBER_FORCES(:,1));
for (i = 1:numels)
    forces = MEMBER_FORCES(i,1:9);
    fprintf(out_fid, '\%7i %8.2f %8.2f %11.2f %9.2f %8.2f %11.2f\n', i, ...
        forces(1), forces(2), forces(3), forces(4), forces(5), forces(6));
end

% print reactions
fprintf(out_fid, '\n\n* Support Reactions at Each Boundary Node\n');
fprintf(out_fid, '\n Node X-dir Y-dir Moment\n');
fprintf(out_fid, '\n (force) (force) (force*length)\n');
numrxns = size(REACTIONS,1);
for (i = 1:numrxns)
    rxn = REACTIONS(i,:);
    dof = size(rxn,2);
fprintf(out_fid, '%7i %12.2f %12.2f %12.2f
', rxn);

end

% print volume change
fprintf(out_fid, '
* Relative Volume Change (volume change/original volume) = %g
', delta_V);

% print stresses
fprintf(out_fid, '
* Max. Tensile Stress (force/length) and Wrinkle Depth at Each Node
(length):');
fprintf(out_fid, '
* (for strapped beam or arch, stress is maximum tensile strap force)
');
fprintf(out_fid, '
     Node Stress  Wrinkle Depth
');
numpts = size(STRESSES,1);
for (i = 1:numpts)
    fprintf(out_fid, '%7i %9.2f  %11.2f
', i, STRESSES(i), CAPACITY_DATA(i,3));
end

% print guy forces
fprintf(out_fid, '
* Forces in Guys -- positive is tension
');
fprintf(out_fid, '
 Element  Force
');
numguys = size(GUYS,1);
for (i = 1:numguys)
    guy_force = -GUY_FORCES(i,1);
    fprintf(out_fid, '%5i %10.3f
', i, guy_force);
end

% now, save results to specially formatted file for easy plotting
for (i = 1:size(P_V_M,1))
    % first, node number and nodal displacements
    fprintf(results_fid, '%5i %8.3f %8.3f ', [i, U((i-1)*3+1), U((i-1)*3+2)]);
    % now, P, V and M
    fprintf(results_fid, '%8.3f %8.3f %8.3f ', P_V_M(i,:));
    % now, stress for woven beams, max strap force for braided, strapped
    fprintf(results_fid, '%8.3f ', 0);
    % now, P/Pn, wrinkle depth, and wrinkling moment (braided only)
    fprintf(results_fid, '%8.3f %8.3f ', CAPACITY_DATA(i,1), CAPACITY_DATA(i,3), CAPACITY_DATA(i,2));
    fprintf(results_fid, '%8.3f %8.3f %8.3f
', 0,0,0);
end

% finally, reactions
for (i = 1:numrxns)
    rxn = REACTIONS(i,:);
    fprintf(reaction_fid, '%7i %12.2f %12.2f %12.2f
', rxn);
end
function solve_for_DL_HRAB

global BASIC_PROPS;
global NODES;
global UPDATED_NODES;
global CONNECTIVITIES;
global BOUNDARIES;
global TOTAL_F;
global DL_F;
global F;
global M_kappa;
global U;
global MEMBER_FORCES;
global STRESSES;
global CAPACITY_DATA;
global GUYS
global GUY_FORCES
global REACTIONS
global LOCAL_NODES;
global STRAP_DATA;
global ELS_AT_NODES;
global ALL_RESULTS;
global P_V_M;
global LARGE_DEF;

% GUYS = [node1, node2, l_o, EA, pre-tension, l_n];
um_guys = size(GUYS,1);
PT_init = 0;
if (num_guys > 1)
    guy_I_full=[1:2,4:size(GUYS,1)];
    num_roof=size(GUYS,1)-3;
    if num_guys > 3
        guy_I=[1,4];
    else
        guy_I=1;
    end
    PT_init = GUYS(guy_I,5);
    % initialize stretch to a small value
    GUYS(guy_I_full,5) = 0.001;
end

% no guys or no pre-tension, just one solve
% first, solve for DL only
num_steps = 2;
del_F = F/num_steps;
F = zeros(size(F));
last_good_U = zeros(size(del_F));

for (i = 1:num_steps)
    F = F + del_F;
    OK = newton_solver_HRAB(last_good_U, 1e-4, 20, 1, 0);
    last_good_U = U;
end
end

if (num_guys > 1) && (norm(PT_init) > 0.001) % we have PT guys

% compute DL slack which must be overcome to PT guy lines
slack = GUYS(guy_I,3) - GUYS(guy_I,6);

% estimate initial stretch to produce PT + slack
stretch = PT_init.*GUYS(guy_I,3)./GUYS(guy_I,4) + slack;

del_s = 0.1;
GUYS(guy_I_full,5) = [stretch(1);stretch(1);stretch(2).*ones(num_roof,1)];
G = get_G_PT_HRAB(PT_init,guy_I);
PT_err = norm(G)/norm(PT_init);

message1 = sprintf('Guy PT force = %g, estimated pre-stretch = %g, PT error = %0.3g', ...
    G(1) + PT_init(1), stretch(1), PT_err);
message2 = sprintf('Roof PT force = %g, estimated pre-stretch = %g', ...
    G(end) + PT_init(end), stretch(end));
disp(message1);
disp(message2);

% Newton iteration to solve
max_iter = 100;
iter = 1;
while (PT_err > 0.001) && (iter <= max_iter)
    GUYS(guy_I_full,5) = [stretch(1).*ones(2,1);stretch(2).*ones(num_roof,1)]+del_s;
    % Forward Approximation
    G_forward = get_G_PT_HRAB(PT_init,guy_I); % G(x_(i+1))
    % Find the Jacobian term
    dG_ds = (G_forward - G)./del_s;
    % Check if one of the terms is 0, if so exit loop, solution failed
    if sum(dG_ds==0)==1
        break
    end
    % Update Inputs
    stretch = stretch - G./dG_ds;
    GUYS(guy_I_full,5) = [stretch(1);stretch(1);stretch(2).*ones(num_roof,1)];
    % Find new G
    G = get_G_PT_HRAB(PT_init,guy_I);
    % Error for each guy pre-tension
    PT_err = norm(G)./norm(PT_init);
    iter = iter + 1;
    message1 = sprintf('Guy PT force = %g, estimated pre-stretch = %g, PT error = %0.3g', ...
        G(1) + PT_init(1), stretch(1), PT_err);
    message2 = sprintf('Roof PT force = %g, estimated pre-stretch = %g', ...
        G(end) + PT_init(end), stretch(end));
    message3 = sprintf('Iter %g / %g',iter,max_iter);
    disp(message1);
disp(message2);
disp(message3);

% If PT_err is significantly large, terminate
  if abs(PT_err) > 1000
    break
  end
end
BIOGRAPHY OF THE AUTHOR

Jay Wegner was born in Annapolis, Maryland on June 15, 1995 to Gail and Dale Wegner. He was raised in Davidsonville, Maryland attending St. Andrews UM Day School until 2008 where he developed a love for soccer and farming by being a part of the school’s inaugural Club Pollo which taught kids how to take care of farm animals.

He then attended high school on Maryland’s Eastern Shore at Gunston Day School in Centreville, Maryland. It was here he began to develop an interest in civil engineering, specifically bridge design, as he was bused across the Chesapeake Bay Bridge twice a day. He continued his soccer passion here while also starting the school’s inaugural near space balloon program and learning Spanish from his Columbian Spanish teacher and soccer coach. There he also found an unlikely friend which he carries with him today, theatre. During his time at Gunston, he acted and stage managed in three productions.

Graduating from high school an Eagle Scout in 2013, he then packed up and headed north in search of more snow and colder temperatures. Arriving at the University of Maine, he found welcoming friends and enthusiastic teachers who helped cultivate his interests as a civil engineer. He continued acting and playing soccer here and there, however the demands of school commanded his attention. Graduating with a Bachelor’s degree in Civil Engineering in 2017, he doubled down when presented with an offer to work under Dr. Davids at the Advance Structures and Composites Center. There he continued the work Dr. Davids and his previous graduate students had completed in the field of inflatable airbeams. While a graduate student, he played a large role in the guidance of the school’s steel bridge team, honing his skills as both a leader and design engineer.

These experiences have led him to what he is today, an athletic, theatrically inclined, unemployed civil engineer in training. Upon graduation Jay will return to his coop in Maryland (much to his mother’s dismay) where he will be searching for employment as a civil engineer. Jay is a candidate for the Master of Science degree in Civil Engineering from the University of Maine in August 2019.