6-2014

Computational Model of Pore Collapse and Densification in Aerogels under Compression

Lutao Xie
Union College - Schenectady, NY

Computational Model of Pore Collapse and Densification in Aerogels under Compression

By

Lutao Xie

* * * * * * * * *

Submitted in partial fulfillment

of the requirements for

Honors in the Department of Mechanical Engineering

UNION COLLEGE

June, 2014

ABSTRACT

ADVISOR: William Keat

Aerogels are solid, porous and light materials that are 90-99% air by volume, with particularly small pore sizes and large specific surface areas. According to previous studies, silica aerogels have appeared to be typical fractal materials. Its microstructure can be described as a fractal network in the length scale 10-1000Å, which is considered to be the result of an aggregation mechanism. To model the behavior of this material, a non-linear finite element code was developed to determine the sequence in which elements fail under compressive load for different starting pore distributions. The 2D geometry of the brittle silica lattice was represented by a single strand of bar elements interconnected by transverse beam elements. As pores collapsed, broken elements were replaced by non-linear contact springs to efficiently model fragmentation of the lattice. Results agreed with the expectation.

Table of Contents

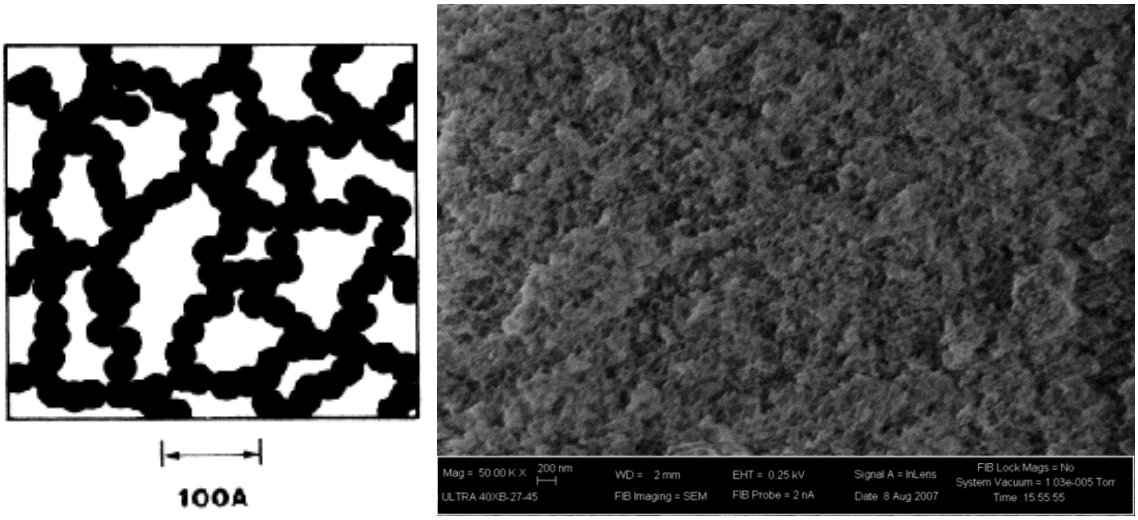
1 Introduction

1.1 Background

Aerogels are solid, porous and light weight materials that are 90-99% air by volume, with particularly small pore sizes and large specific surface areas. One common type of aerogel that has been significantly researched is silica aerogels which are highly porous ceramic materials derived from silica gel. It possesses a number of exceptional and unique physical properties including extremely low thermal and electrical conductivity, good optical transmission properties and can also be made super-hydrophobic [1]. The BET (Brunauer–Emmett–Teller) surface area of the silica aerogel made at Union using the RSCE (rapid supercritical extraction) process is usually about 550 - 760 m^2/g with a total pore volume ranging from 3.4 cm^3/g to 3.6 cm^3/g. Pore sizes mostly range from 10nm to 40nm in diameter. Pore distribution curves and stress-strain curves are commonly used to describe aerogels' mechanical properties. Both can be obtained by conducting experiments at the Union College aerogel lab [2].

According to previous studies, silica aerogels have appeared to be typical fractal materials. Its microstructure can be described as a fractal network in the length scale 10-1000Å, which is considered to be the result of an aggregation mechanism [3]. Fractal is a kind of self-similar geometric pattern that is repeated at ever smaller scales to produce irregular shapes and surfaces that cannot be represented by classical geometry [4]. Data from SAXS (Small-angle X-ray scattering) measurements in a previous study has suggested a reasonable structure of silica aerogel, shown schematically in Figure 1(a), where the porosity is most likely due to a random "jungle gym" or branched-polymer-like structure [5]. A SEM image of silica aerogel (shown in Figure 1(b)),to some extent,

indicates an agreement with the schematic diagram in Fig.1b, where the pores of the silica aerogel seem to be formed by a network of intersecting chains along with clusters.



(a) (b)

Figure 1. (a) Schematic diagram of the structure suggested for silica aerogel according to data from modified SAXS study[5]; (b) SEM image of silica aerogel made by Union College aerogel lab [6].

Pore distribution curves are used to determine the porosity of aerogels and can be obtained by conducting gas sorption tests. The plots indicate the relationship between pore size and the corresponding number of pores, especially the distribution of different pore sizes in aerogels. Gas sorption measurements are commonly used for determining the surface area and pore size distribution of different solid materials, which allows assessment of a wide range of pore sizes (from 0.35nm up to 100 nm). At the Union College aerogel lab, Micromeritics ASAP 2010 was used to run gas adsorption tests. The pore-size distribution curves of silica aerogel obtained from a gas sorption test are shown in Figure 2. The sample was crushed into pieces with sizes of 2 to 5 mm. Pore distribution curves were created corresponding to different equilibration times (5s,10s,20s and 40s) [7].

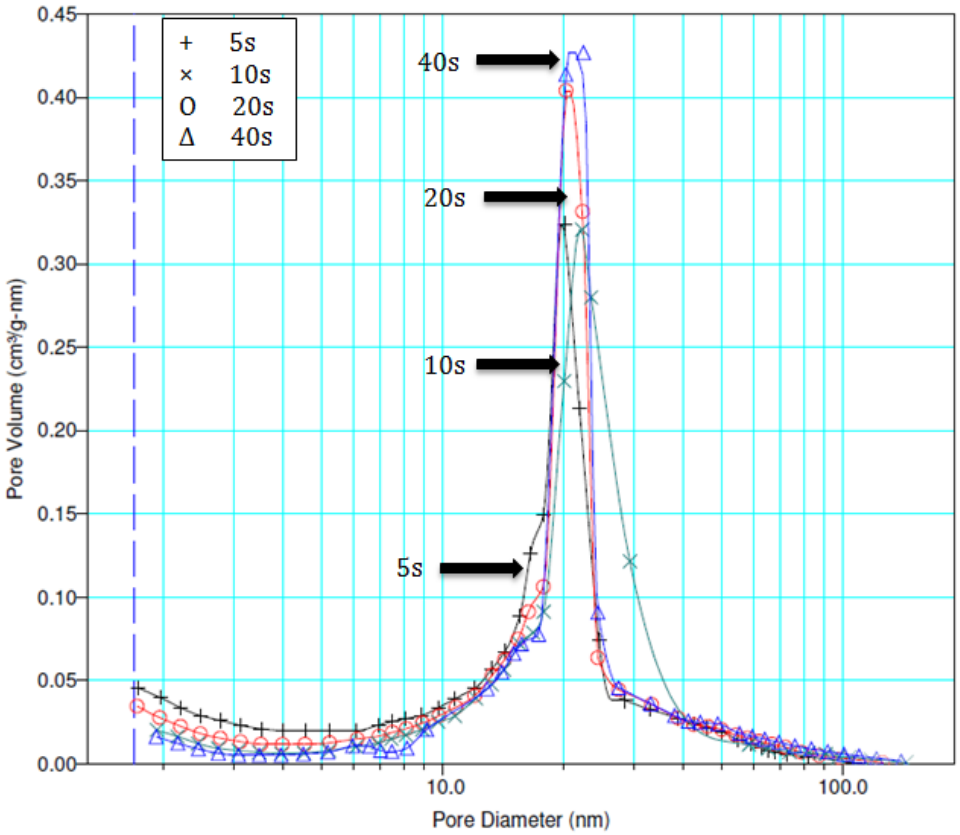


Figure 2. This plot shows the graph of the BJH desorption volume (cm^3/g-nm) as a function of the measured pore diameter (nm) for medium sized sample during the gas sorption test for silica aerogel. The corresponding symbols for the equilibration time 5s, 10s, 20s, and 40s are respectively "+", "×", "O" and "Δ" [7].

Another significant way to characterize mechanical properties of aerogels is through the stress-strain curve, also known as load-displacement curves. The diagram represents the relation between stress (force per area) and strain (ratio of deformation over the original length) in a given material in loading. To obtain such diagram, a tensile test either in compression or tension is conducted on a specimen of a material. A stress-strain curve was created by running a compression test on five randomly chosen samples at the Union College aerogel lab, shown in Figure 3. The stress-strain curve was initially linear with a very small slope which increases as the strain increases[2].

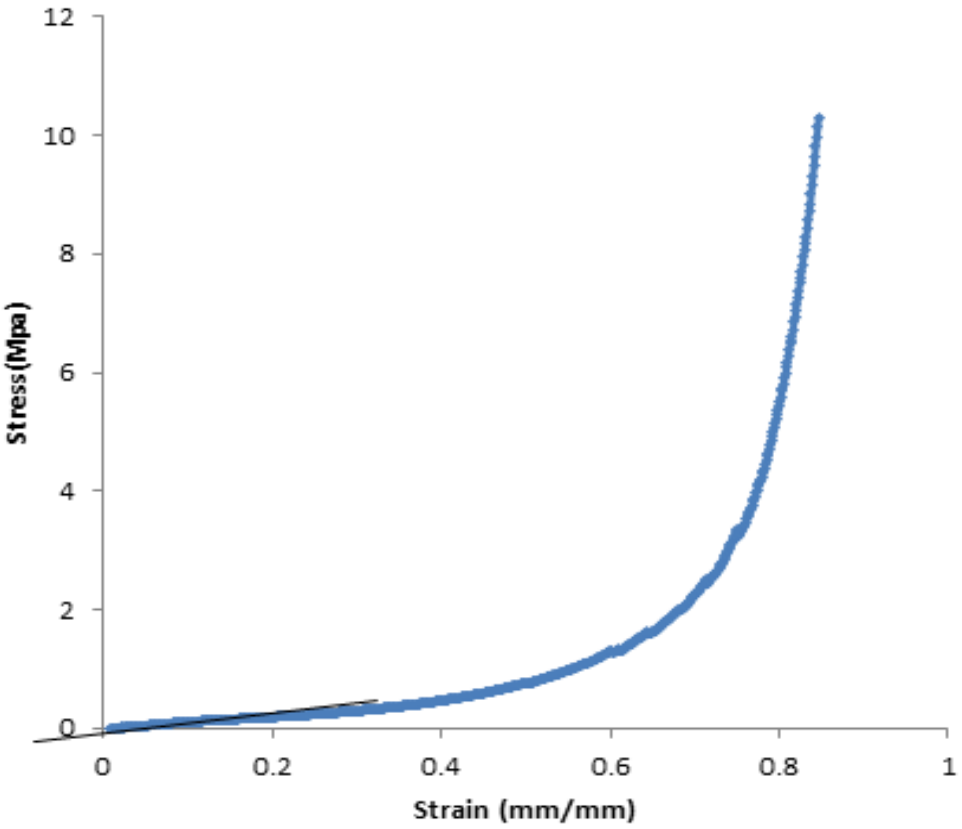


Figure 3. (b) Graph of stress as a function of strain from compression test data for silica aerogel. The original height of the sample was 19mm with a cross-sectional area of 388mm^2. A trend-line was added in the elastic deformation region to calculate the elastic modulus of this silica aerogel [2].

We believe that the process of how aerogels fail in compression can be summarized by the graph in Figure 4. As is shown, the aerogel will undergo linear elastic process when the test just starts. Normally this linear elastic period is very short since the aerogel is very fragile. We assume the chains forming the structure of aerogels can be modeled as beams. After the initial linear elastic behavior, the beams in the aerogel start to break, which results in pores collapsing. At high strains, the aerogel fragments are being crushed, or compacted, since most of the pores have collapsed [8].

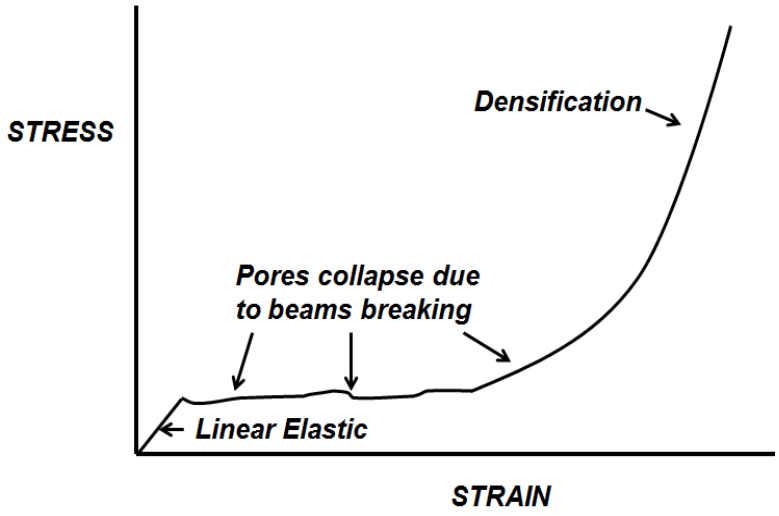


Figure 4. Stress-strain curve of silica aerogel in compression. When silica aerogels are subjected to small levels of uni-axial loading, the resulting stress-strain curve is initially linear. As the loading level increases, it becomes concave with increasing slope because of densification [8].

1.2 Alternative Computational Methods for Modeling Damage and Densification in Brittle Foams

To model the microstructural behavior of aerogels under compression, several significant challenges need to be conquered. First of all, a non-linear finite element analysis is required due to the fact that this system will be non-linear, which is different from a normal finite element analysis. Secondly, in the pores collapsing region, broken elements will come into contact with each other, which makes it most difficult to track all elements' motions. Even with the assumptions of the simplest and most idealized aerogel microstructures, to simulate such elements behavior is a grand challenge. Last but not the least, after the transition from the pore collapsing region into the densification region, broken fragments will fill up voids, the interaction of which is quite hard to predict. Similar problems and challenges have been discussed in previous studies as well, where

various modeling methods were introduced to simulate such microstructural behavior on brittle porous materials with common properties as aerogels.

One method used to model deformation of closed-cell foams is the Material Point Method (MPM), which was derived from the Particle-In-Cell Method (PIC), and for special cases, the MPM method can be modified as the General Interpolation Material Point (GIMP) method [9]. The MPM method was developed by Suslsky et al in 1994 and 1995 specifically for solving dynamic solid mechanics problems [10]. Regardless of the names and specific applications of the methods, they were all based on the same theory. The material body is discretized into material unit points that carry all material properties and states. The material points are to the studied material body is what the pixels are to an image [9]. A new grid will be generated each computational time step [9]. Material points are used to represent a material continuum and a fixed background mesh (grid), shown in Figure 5, is used for solving field equations. Note that the material points are not subjected to mesh entangling and are convected by the deformation of the solid throughout the background grid [10]. Because MPM does not depend on the use of a body-fixed mesh for computation, it provides an advantage in the simulation of some dynamic problems over finite element and meshless methods and moreover, is able to handle the modeling of realistic microstructures of foams and multiphase materials [10]. However, this method requires a large scale computation capability and does not include as much intelligence in modeling. A comparison of the stress-strain curve of foam compression results from MPM simulation and experimental measurements is shown in Figure 6. The curves showed some capability of simulating the features of stress-strain curve but did not match with the experimental measurements very well [11].

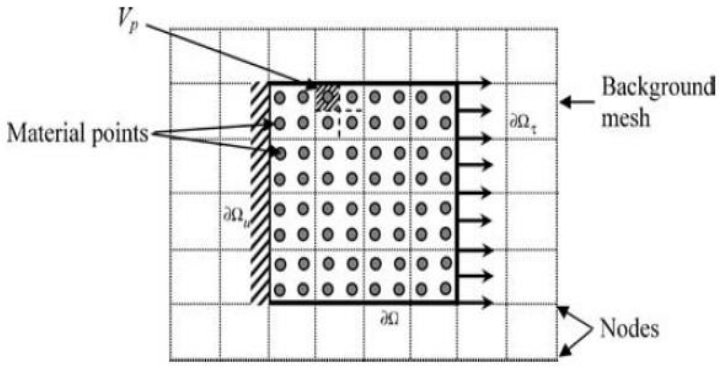


Figure 5. A schematic of MPM grid cells with material points [11].

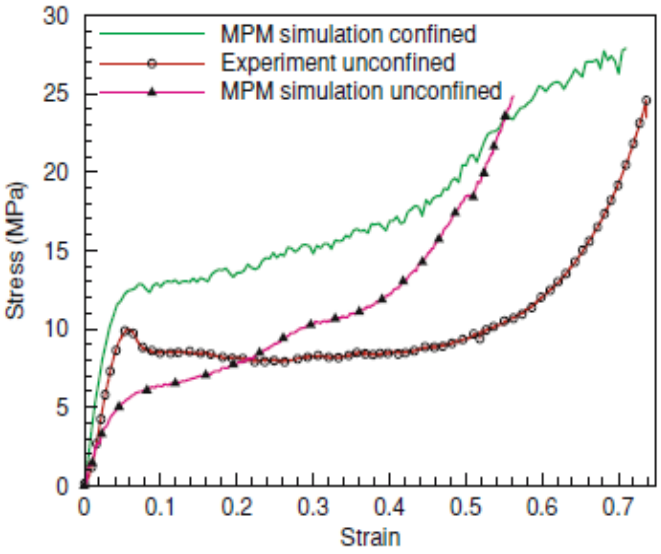


Figure 6. Comparison of stress-strain curve obtained from numerical simulation using MPM and experiment [11].

Another approach, discrete simulation technique combines discrete- and finite-element modeling techniques to simulate the compression of mixtures of both ductile and brittle particles. It was formalized by Cundall & Strack in 1979 to explore interactions in rock systems. Generally, with different mass, moment of inertia and contact properties, every particle is modeled separately [12]. In order to represent contact compliance and energy absorption, normal and tangential springs and dashpots are used at each point of contact, which can be either deformable or rigid. The contact forces at each instant are determined by the amount and rate of overlap between adjacent elements [12]. Based on

local equilibrium of direct neighbors, the total unbalanced forces and moment acting on each particle can be calculated and used to estimate each particle's acceleration, which is in turn used to compute new displacements at the end of the current time-step. The same process is repeated till forces and moments are incrementally approaching equilibrium [12]. A schematic of the interaction between two particles is shown in Figure 7. The boundary of each particle is modeled as an interface element with a halo of finite thickness (t), stiffness (k) and damping (c). This method allows finite displacements and rotations of discrete bodies, and is capable of identifying new contacts as the calculation automatically [12]. The advantage of this method is that it is able to predict material yield surfaces for the compaction of a mixture of different particles. However, adjustment on the model is required to improve the accuracy of the result [12].

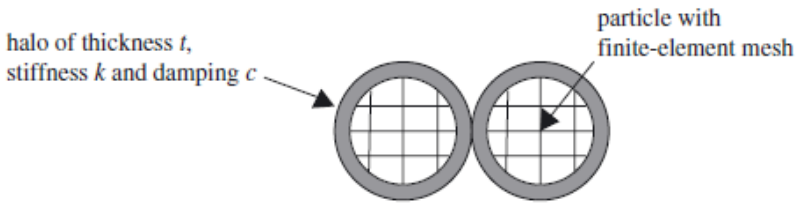


Figure 7. The schematic of the interaction between two particles [12].

The Kelvin cell foam model was developed to model the mechanical behavior of open cell foams under uniaxial compressive load. An example of the geometry of the Kelvin cell is shown in Figure 8, which is idealized to be periodic by adopting the regular, 14-sided polyhedron of Lord Kelvin [13]. The cell consists of equal length edges from 6 squares and 8 hexagons and the foam is assumed to be linear elastic with modulus E and Poisson's ratio υ.The ligaments are considered as Bernoulli-Euler beams with different cross-sections such as squares, circles, Plateau borders and equilateral tri-angles [13]. Moreover, the foam can be anisotropic and the effect of both axial and shear

deformation is taken into consideration in beam models. Since the microstructure is assumed to be very regular and periodic, the characteristic cell of the anisotropic Kelvin foams can be represented as in Figure 9(a). "The characteristic cell is discretized with finite elements in the nonlinear code ABAQUS using the B32, 3-noded quadratic space beam element" [12] and the elastic performance of the open cell foam can be predicted for this type of geometry. As for crushing response, the ligament contact is approximated by connecting the corner nodes of all vertical rhombi with springs, shown as Figure 9(b). Once the vertical distance between the two nodes exceeds a chosen limit, the springs will be activated [13]. This idealized Kelvin cell foam model has the ability to capture some features of the compressive responses well quantitatively and others qualitatively, yet a more accurate modeling of the irregularities in the cell morphology of actual foams is required for better predictions [13].The results of the simulation of crushing responses of a finite size foam micro-section for different values of contact variable is shown in Figure 10.

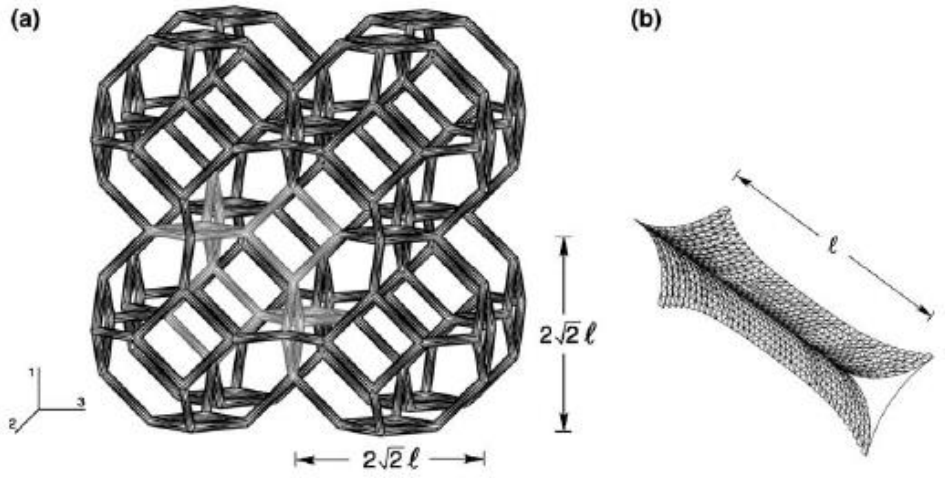


Figure 8. (a) Cluster of 14-sided Kelvin cells. (b) Geometry of foam ligaments [12].

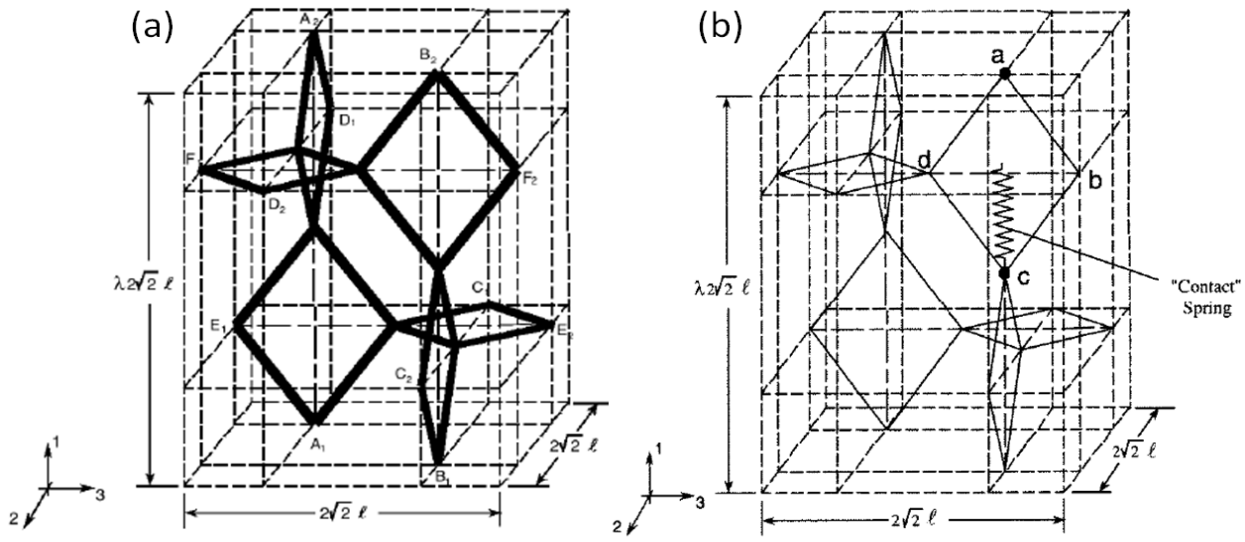


Figure 9. (a) The Kelvin foam characteristic cell. (b) Characteristic cell showing placement of a "contact" spring introduced in direction of loading in a typical vertical rhombus [13].

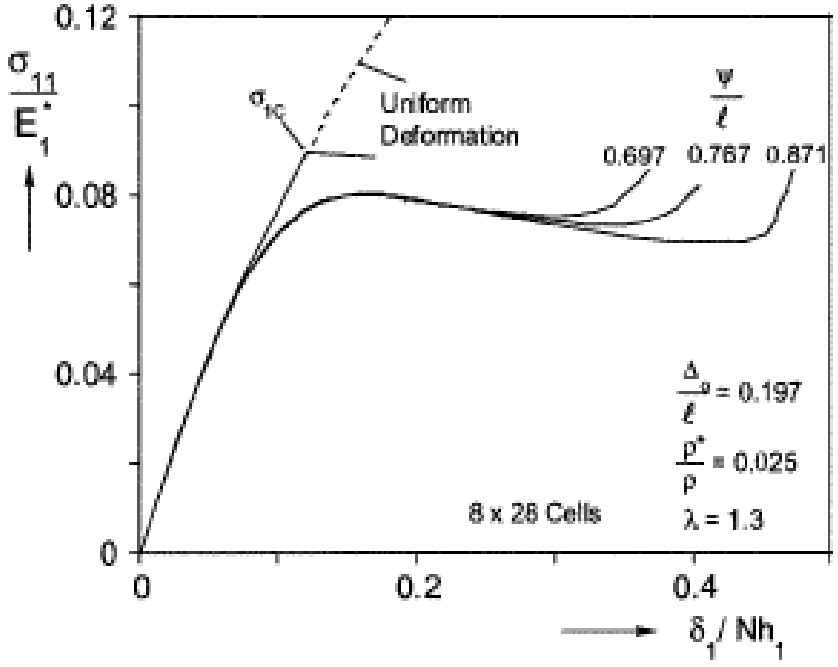


Figure 10. Crushing responses of a finite size foam microsection for different values of contact variable **Ψ**, which is the gap that must be closed first before the spring is activated.

1.3 Specific objectives of this research

A single strand model will be developed in which each element is treated as a linear spring and the stiffness for each element can be calculated from the force-displacement relationship for an axial member under tension or compression. Global finite element equations will be employed to solve for nodal displacement, which in turn, will be used to find the corresponding stress of each element, as well as its factor of safety. Once an element from the strand breaks, it will be replaced by a non-linear spring to simulate contact between the broken elements. The advantage of this approximation is that it simplifies the modeling of fragmentation and contact problems as compared to other methods.

After this single strand model is completed, it will be implemented in two and three dimensions by connecting multiple strands together with beam elements. The strands and beams can be arranged in various ways to simulate different porous structures.

The simulation of the densification region of this model is highly dependent on the stiffness of the non-linear spring used to replace the broken elements. The choice of non-linear spring functions will affect this part of the stress-strain curve will be investigated.

Ultimately, this model will be used to simulate compression of silica aerogel by using the experimental data obtained from the aerogel lab. Comparison of experimental and simulated modeling results will be made and discussed. In addition, various mesh generators developed previously will be applied to this model to study the effects of clustering structures under loading.

2 1-D Single Strand Model

2.1 Model Review

A 1D model is developed in Figure 11(a) models a single strand of connected bar elements under compression. Each element is modeled as a linear spring which interacts with adjacent elements at the nodes. The bottom of the strand is supported by the ground while the top end is subject to a compressive force, which is applied incrementally. Each element experiences the same compressive load due to static equilibrium. The nodes are numbered as shown to track the axial displacements of each individual element while also tracking the movement of the grip. The stress on each element was determined using the governing finite element equations. Three types of structural failure modes were analyzed in order to identify the next element to fail [8]. When an element fails (equivalent to a pore collapsing), the length of the failed element is reset to close to zero with a large cross sectional area and the coordinates of the other nodes are updated to be consistent with this change (see Figure 11(b)). The result is that there will be a gap between the grip and strand. Eventually, as the grip is moved incrementally, the grip will catch up to the element and start compressing the strand again. This process is repeated till all elements break [8].

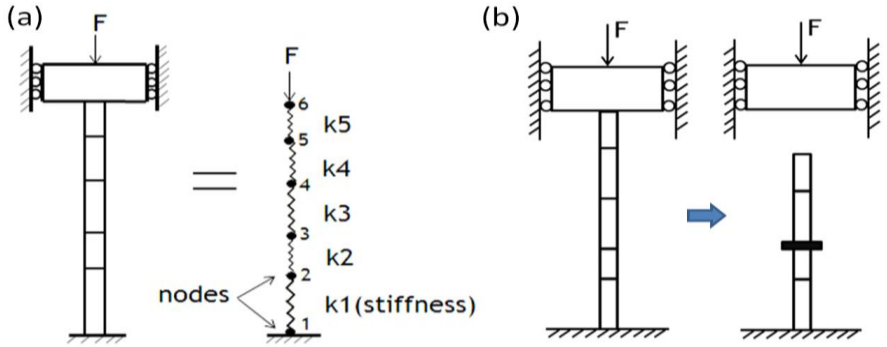


Figure 11. (a) An example of a strand with five elements and six nodes. The bottom is supported by the ground and a vertical force is applied at the top end of the strand. Each element is treated as a spring with its own stiffness. (b) A sketch showing the collapse of one element [16].

2.2 Pore Collapse Description [8]

Figure 12 shows a typical sketch of force versus grip displacement, ΔYgrip. The force increases until one element breaks; then the force drops to zero because of the gap between the grip and the end of strand. Once the grip catches up with the strand, the force will increase until another element breaks. The magnitude of the critical force for breaking each element gets bigger as elements break because the weakest elements break first.

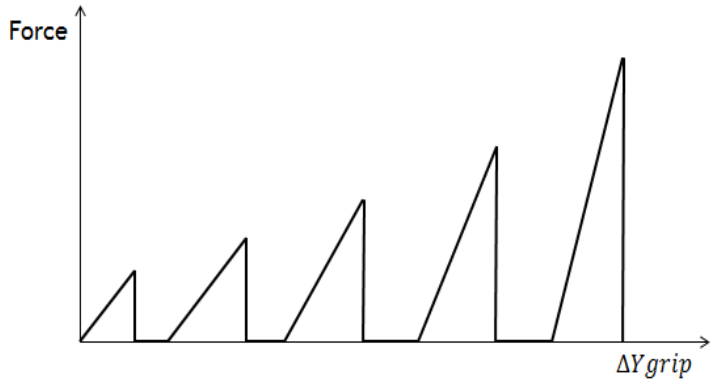


Figure 12. A sketch of force versus change in y-coordinate of the grip.

Three structural failure modes were considered when calculating the stress (σ) of each element: axial compression, Euler buckling for simply supported ends and buckling with initial curvature. Axial compression is due to the normal stress that the strand experiences from the vertical force. The corresponding equation is as follows:

$$\sigma = \frac{P}{A} \tag{1}$$

where P is the compressive load and A is the cross-sectional area of the element. The second mode is Euler buckling of simply supported columns, illustrated in Figure 13.

Figure 13. Euler buckling of imply supported column [15].

The critical buckling load, P_{cr}, is defined by:

$$\mathbf{P_{cr}} = \frac{\pi^2 EI}{L^2} \tag{2}$$

and the corresponding critical stress is

$$\sigma_{cr} = \frac{P}{A} = \frac{\pi^2 EI}{AL^2} \tag{3}$$

where E is the modulus of elasticity, I is the moment of inertia and A is the cross-sectional area of the element. The third mode is buckling of a simply supported column with initial curvature, shown in Figure 14.

Figure 14. Simply supported column with initial curvature [16].

The stress associated with this form of buckling is composed of two parts, one attributable to axial compression, the other to bending:

$$\boldsymbol{\sigma} = \left|\frac{\boldsymbol{P}}{\boldsymbol{A}}\right| + \left|\frac{\boldsymbol{MC}}{\boldsymbol{I}}\right| \tag{4}$$

where C is the radius of the circular cross-section, and the internal bending couple, M, at the mid-span is given by

$$\mathbf{M} = \frac{\boldsymbol{P \cdot e}}{\mathbf{1} - \frac{\boldsymbol{P}}{\boldsymbol{Pcr}}} \tag{5}$$

in which e is the eccentricity of the initial curvature and P_{cr} is the critical buckling defined in equation (2). For the first and third failure modes, the factor of safety is calculated using the following equation:

$$N_{1,3} = \frac{S_U}{\sigma} \tag{6}$$

where Su is the ultimate stress, and σ for compression and for buckling with initial curvature are defined by equations (1) and (4), respectively. For Euler buckling, the factor of safety is found by using:

$$N_2 = \frac{\sigma_{cr}}{\sigma} \tag{7}$$

where σ and σ_{cr} are defined by equations (1) and (3), respectively.

2.3 Densification Model

In order to better simulate the densification behavior of the single strand model under compression, two modifications were made. Firstly, the random parameter of the diameter was increased from 0.5 to 0.8, so that the cross-sectional areas of the elements could vary over a larger range. Secondly, the change in element length associated with pore collapse was reduced from 100% to 75% of the failed elements' original length. This enabled the broken elements to continue to deform under the compression load.

2.4 Relations to Pore Distribution Data

Element lengths were varied along the length of the strand in accordance with actual pore distribution data from a silica aerogel. The relationship between pore diameter, D, and the derivative of pore volume with respect to pore diameter, dV/dD, was known from the experimental data. The volume increment associated with each pore diameter was obtained by estimating the area under the curve of dV/dD vs. D, treating the pores as spheres. The number of pores corresponding to each pore diameter could then be found by dividing each volume increment by the volume of a single pore of

corresponding pore diameter. The element lengths were thus assigned so as to have the same magnitude and distribution relationship as the pore diameters [8].

2.5 Results

A single strand of aerogel was modeled using 100 elements. The lengths of the elements comprising the strand were varied in accordance with actual pore distribution data for silica aerogel. Other inputs were set as follows: Ltot (total length of the strand) =1.0e-06m, di (initial diameter) =3.49e+9m, E (elastic modulus of the aerogel) =7.3e+10m, dincr (Number of displacement increments in elastic range) =40, Ugripmax (Total displacement of grip) =-8.0e-07m, Su(Ultimate stress) =1.1e+8pa, fracl (Fraction of element length) =0.1, alpha (Densification factor) =0.9.

The graph of utots (downward displacement of the top end of the strand) vs. nincr (number of increments) is shown in Figure 15. The downward displacement of the top end of the strand starts at zero and gets larger as the elements break. The displacement associated with elastic deformation is much smaller than the displacement jumps due to pore collapse. As result, the slope of the curve during the elastic deformation appears almost horizontal in the graph. Once a pore collapses, utots jumps by an amount equal to approximately 75% of the original element length[8].

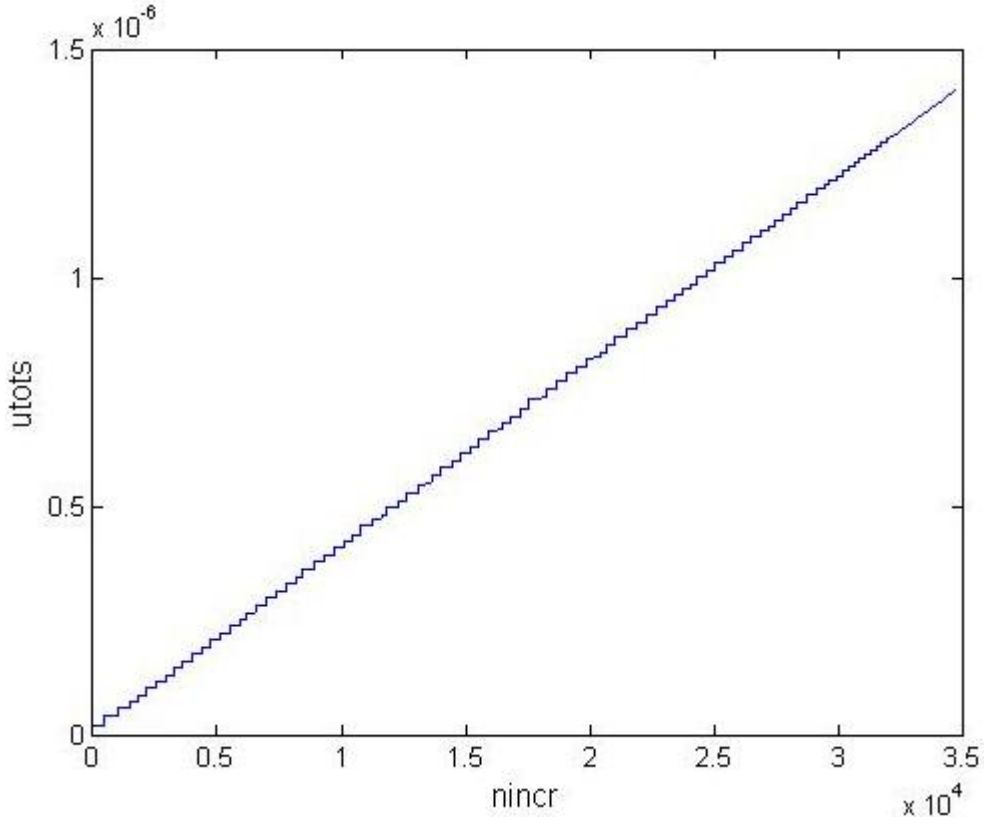


Figure 15. The graph of utots (downward displacement of the top end of the strand) vs. nincr (number of increment).

The graph of f (compressive force) vs. ΔYgrip (downward displacement of the end of grip) for a single strand of aerogel is shown in Figure 16. The force followed an increasing trend overall since the weakest element breaks first and a greater force will be required to break the next element. The force falls to zero after pore collapse because of the resulting gap between the grip and the strand. Once the grip catches up with the strand, the force will increase again [8].

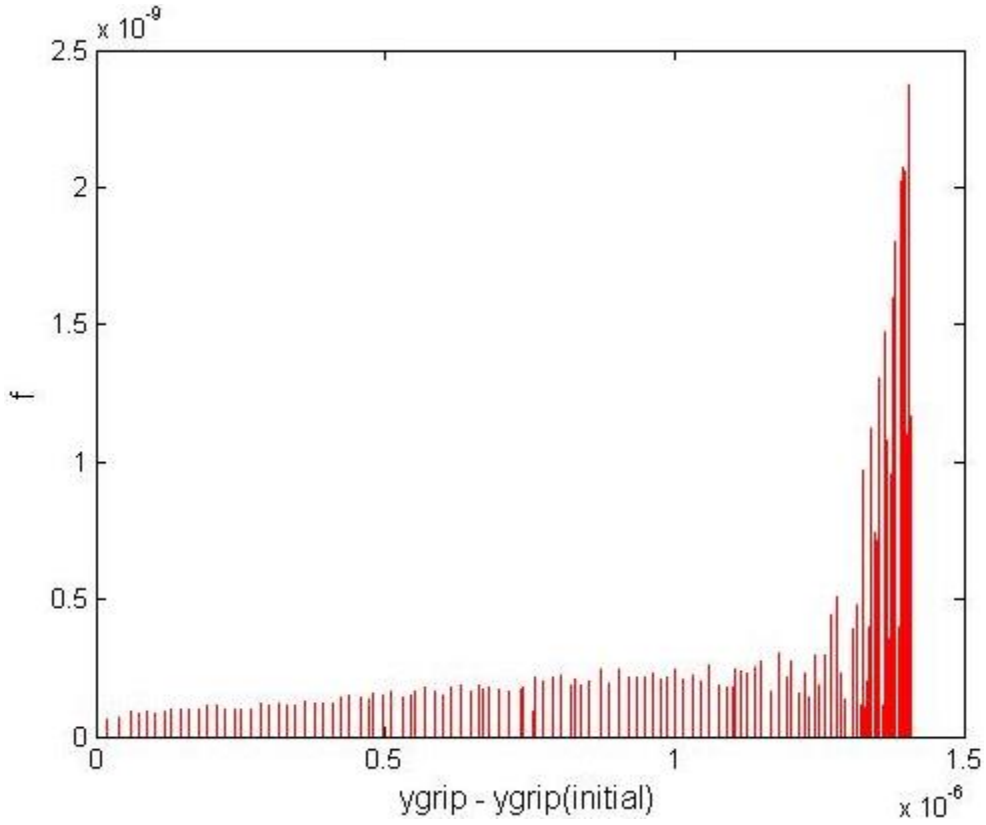


Figure 16. The graph of f (compressive force) vs. ΔYgrip (downward displacement of the end of grip).

A graph comparing the stress-strain curve created from the model and from the experiments is shown in Figure 17. The stress was approximated as the ratio of the force to the cross-sectional area. The low strain results appear to be similar in slope and magnitude, and the simulation curve in the high regime follows the increasing trend of the experimental results. However, there are still notable discrepancies between the curves, especially in the high strain regimes. Certainly the model geometry bears little resemblance to the actual aerogel [8].

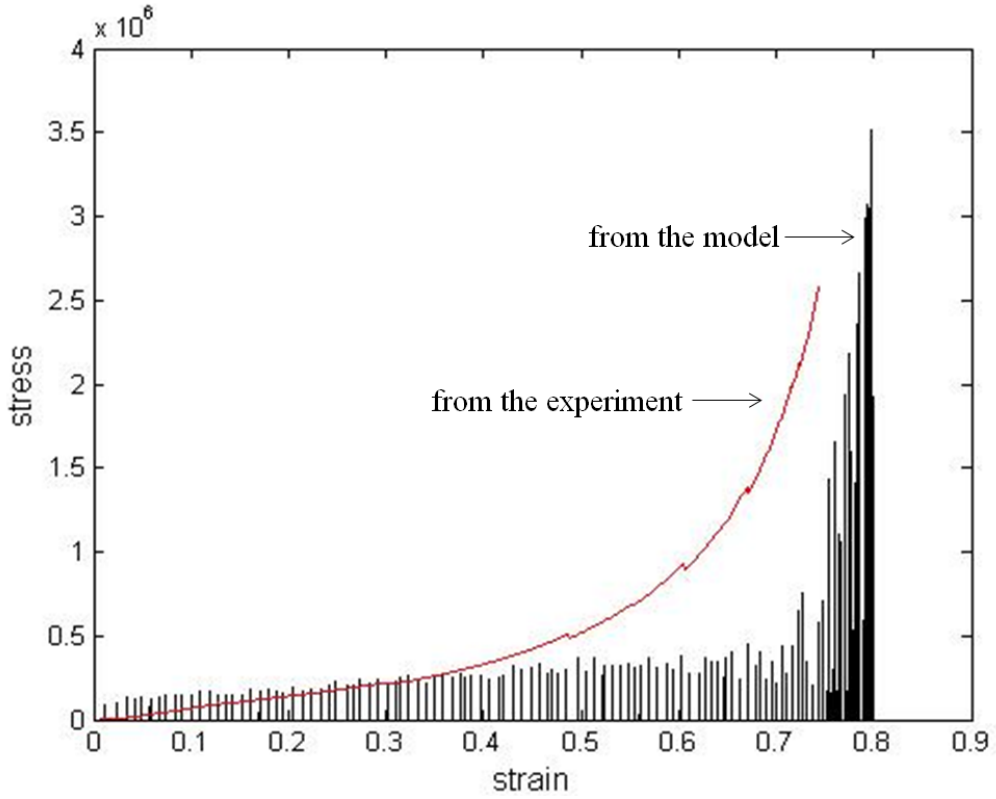


Figure 17. The graph of comparison of stress-strain curves generated by the model and from the experiment.

2.6 Conclusions and Discussions

The results from this one dimensional non-linear model of pore collapse under compression agree with our expectations. The graphs created are very reasonable and can be well-explained in terms of the pore collapsing process and the resulting single stand model has a number of properties in common with the actual aerogel. First, because the element lengths are consistent with the size and number of pores in the actual aerogel, the model is energy equivalent in the sense that the same number of elements is expected to break on average, thus releasing an equivalent amount of strain energy. Second, the model has the same bulk density as the actual aerogel and thus may be viewed as volume

equivalent. Third, the model is also structurally equivalent in that element lengths and cross-sectional areas are consistent with the experimentally measured pore distribution and surface area of the actual aerogel [8].

However, the stress-strain curve did not match with the experimental results perfectly because this one-dimensional model falls short in two important respects. First, due to the different geometry in our model, all elements are equally loaded, while in the actual aerogel, forces will be different due to angled elements and clustering etc. Second, because this model is only in one dimension, the simulation is as close as the real situation [8].

3 Single Strand Model with Transverse Beam Elements

3.1 Model Overview

As a first step towards development of two-and three-dimensional models, such as depicted in Figure 18, the single strand model, composed of bar elements, was enhanced through the addition of transverse beam elements. The beam elements were connected on one end to the nodes of the single strand model and on the other end to the ground. All beam and bar elements were treated as linear springs with the nodes of single strand constrained to move in the axial direction, as in the earlier model. A simple diagram is shown in Figure 19. A vertical load will be applied at the end of the strand as shown. The stress experienced in each beam and bar element will be determined by using governing finite element equations. During compression, bar element are subject to three possible failure modes as before and the beam elements are assumed to fail when the maximum bending stress in the element is equal to the ultimate strength. During crushing, failed beams are removed from the model while non-linear springs, capable of modeling contact between adjacent fragments, are used to replace the failed bar elements. The advantage of using the non-linear springs is that the bar elements are kept connected, which simplified the modeling of fragmentation and contact problems as introduced before. A preliminary 2D model with one strand and beams connected by nodes is planned to develop first.

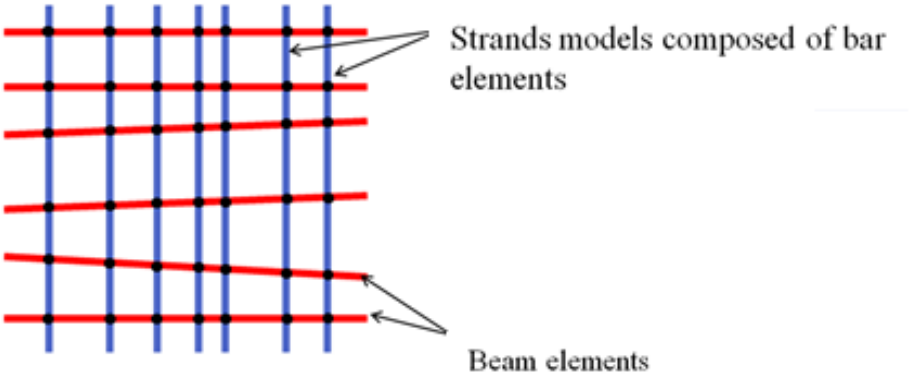


Figure 18. A brief schematic of the beam-bar structure of the 2D model.

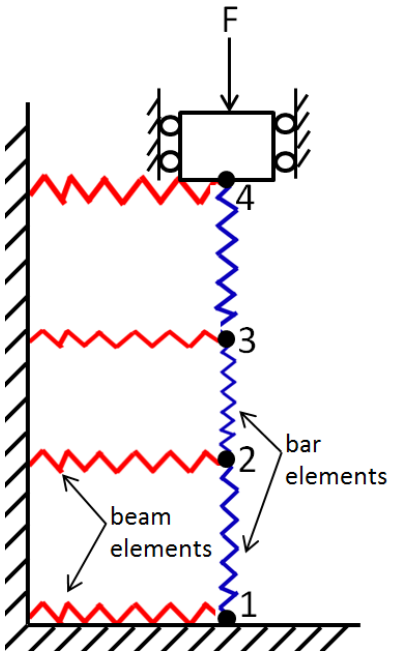


Figure 19. A diagram of the preliminary model with one strand and beams connected to supported wall.

3.2 Transverse Beam Element

The stiffness for a beam element is calculated from the force-displacement relationship for an axial member under tension or compression:

$$k = \frac{P}{\delta} = \frac{AE}{L} \tag{8}$$

where k represents element stiffness, P is the axial force, $\boldsymbol{\delta}$ is the change in length, A is the cross sectional area, E is the modulus of elasticity and L is the length of the element. The stiffness for a beam element was derived by superposition for the displacement boundary conditions represented in Figure 20.

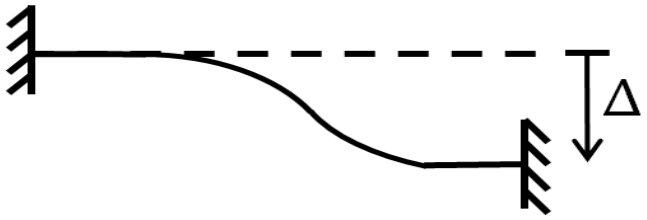


Figure 20. The superposition for the displacement boundary conditions of a beam element[16].

The resulting stiffness equation for the beam element is given by:

$$\boldsymbol{k} = \frac{\boldsymbol{P}}{\Delta} = \frac{\boldsymbol{12EI}}{\boldsymbol{L^3}} \tag{9}$$

where I represents the area moment of inertia.

The fail of beam elements is treated different from the bar elements, as shown in Figure 20. Figure 20, they will be bended due to the vertical force at the node. The moment can be calculated as

$$\boldsymbol{M} = -\frac{\boldsymbol{6EI\Delta}}{\boldsymbol{L^2}} \tag{10}$$

where Δ stands for the displacement at the end of the beam.Therefore the maximum stress can be found as

$$\boldsymbol{\sigma_{max}} = \left|\frac{\boldsymbol{MC}}{\boldsymbol{I}}\right| = \left|\frac{\boldsymbol{6E\Delta C}}{\boldsymbol{L^2}}\right| \tag{11}$$

where C is the diameter of the cross-sectional area of the beam element and the corresponding factor of safety is the radio of the ultimate stress and the maximum stress:

$$\mathbf{n_{beam}} = \frac{\sigma_u}{\sigma_{max}} \tag{12}$$

The factor of safety of beam elements will be calculated together with the bar elements and all the factors of safety will be compared together so as to find the minimum value. The broken beam element will be simply taken out from the model.

3.3 Non-linear Spring Element for Modeling Contact

The graph of the approximate functions used to interpret the stiffness of the non-linear spring is shown as Figure 21. It is composed of two linear functions, numbered in the figure as (1) and (2). The x-axis represents the separation between the two nodes and y-axis stands for the corresponding stress from the spring. The stress/force from the spring will increase largely when the distance between the two nodes is compressed to a certain level (less than t). The stress/force from the spring will be relatively small when the spring gets released (when the length of separation is greater than t). The stress will approach to zero when the spring is at its original length L.

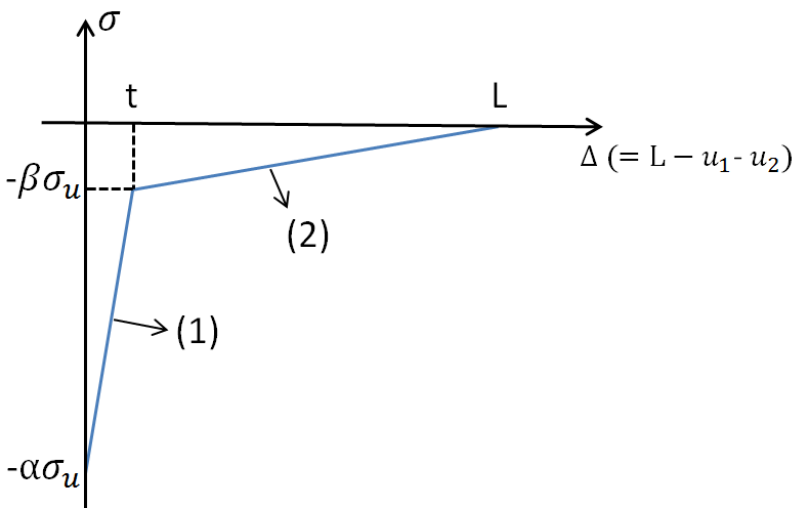


Figure 21. The graph of the approximate functions to interpret the stiffness of the non-linear spring.

Functions (1) and (2) were defined as

$$\boldsymbol{\sigma_{(1)} = \frac{(\alpha-\beta)\sigma_u}{t}\Delta - \alpha\sigma_u} \tag{13}$$

$$\boldsymbol{\sigma_{(2)} = \frac{\beta\sigma_u}{L-t}\Delta - \frac{\beta\sigma_u}{L-t}L} \tag{14}$$

where Δ represents the length of separation of the two nodes; $\sigma_{(1)}$ and $\sigma_{(2)}$ stand for the axial stress for function (1) and (2); t is the diameter of silica strand; which also acts as the "threshold"; L is the initial element length; $\sigma_u$is the ultimate strength of the beam element; α and β are two positive constants that we defined as 664 (= E/Su+ β) and 0.1 respectively. The corresponding element stiffness matrix equation for this non-linear spring is defined as

$$[k]\{u\} + \{F_I\} = \{F\} \tag{15}$$

Where [k] stands for the stiffness of the spring, {u} is its nodal displacement, $\{F_I\}$ represents the oringinal stiffness and {F} is the resultant force from the spring.

Therefore, the relationship between nodal displacement and the force from the non-linear spring can be derived as

$$\begin{bmatrix} \frac{A(\alpha-\beta)\sigma_u}{t} & -\frac{A(\alpha-\beta)\sigma_u}{t} \\ -\frac{A(\alpha-\beta)\sigma_u}{t} & \frac{A(\alpha-\beta)\sigma_u}{t} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} + \begin{Bmatrix} -\frac{A(\alpha-\beta)\sigma_u L}{t} \\ \frac{A(\alpha-\beta)\sigma_u L}{t} \end{Bmatrix} = \begin{Bmatrix} F_1 \\ F_2 \end{Bmatrix} \tag{16}$$

$$\begin{bmatrix} \frac{A\beta\sigma_u}{L-t} & -\frac{A\beta\sigma_u}{L-t} \\ -\frac{A\beta\sigma_u}{L-t} & \frac{A\beta\sigma_u}{L-t} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{Bmatrix} F_1 \\ F_2 \end{Bmatrix} \tag{17}$$

where A stands for the cross-sectional area of the beam elements, u_i stands for the displacement at two nodes and F_i is the force at the two ends of the non-linear spring. They were used to modify the global stiffness matrix after bar elements broken.

3.4 Governing Finite Element Equations

Each element is originally considered as linear springs and has its own set of stiffness equations, which for element n may be expressed as:

$$\begin{bmatrix} k_n & -k_n \\ -k_n & k_n \end{bmatrix} \begin{Bmatrix} u_n \\ u_{n+1} \end{Bmatrix} = \begin{Bmatrix} f_n \\ f_{n+1} \end{Bmatrix} \tag{18}$$

where k is the element stiffness, u is the nodal displacement at each end and f represents the force applied at the ends. Element stiffness and extend loads for all elements were assembled into a global stiffness matrix and a global force vector respectively by using the direct stiffness method:

For the bar elements, the global stiffness matrix has the following structure.

$$[K]_{strands} = \begin{bmatrix} k_1 & -k_1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ -k_1 & k_1+k_2 & -k_2 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & -k_2 & k_2+k_3 & k_3 & 0 & \cdots & 0 & 0 & 0 & 0 \\ & & & & & \vdots & & & & \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & -k_{n-2} & k_{n-2}+k_{n-1} & -k_{n-1} \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & -k_{n-1} & k_{n-1} \end{bmatrix}$$

(19)

The global stiffness matrix for the beam elements is a diagonal matrix.

$$[\boldsymbol{K}]_{\boldsymbol{beams}} = \begin{bmatrix} k_{b1} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & k_{b2} & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & k_{b3} & 0 & \cdots & 0 & 0 & 0 \\ & & & & \vdots & & & \\ 0 & 0 & 0 & 0 & \cdots & 0 & k_{b(n-1)} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & k_{bn} \end{bmatrix} \quad (20)$$

The total global stiffness matrix is the sum of the global stiffness matrices for the strand and the transvers beams.

$$[\boldsymbol{K}]_{\boldsymbol{total}} = [\boldsymbol{K}]_{\boldsymbol{strands}} + [\boldsymbol{K}]_{\boldsymbol{beam}} \quad (21)$$

The corresponding global vectors of nodal displacement and force are defined below.

$$\{U\} = \begin{Bmatrix} \boldsymbol{u_1} \\ \boldsymbol{u_2} \\ \boldsymbol{u_3} \\ \vdots \\ \boldsymbol{u_n} \end{Bmatrix} \quad (22)$$

and

$$\{F\} = \begin{Bmatrix} \boldsymbol{f_1} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{f_n} \end{Bmatrix} \quad (23)$$

Before solving the equations, the following displacement boundary conditions were imposed by modifying [K] and {F}:

$$U_1 = 0 \tag{24}$$

$$U_N = \Delta Utot \tag{25}$$

where U_1 is the displacement of the node at the bottom end, U_N is the displacement of the N^{th} node located at the top end of the strand and $\Delta Utot$ is the controlled incremental displacement of the fixed grip, which will be a negative quantity.

The resulting global finite element equations, represented below in shorthand matrix form

$$[K]\{U\} = \{F\} \tag{26}$$

were solved simultaneously to obtain nodal displacements, {U}, using the following Matlab syntax

$$U = K\backslash F \tag{27}$$

The forces acting on the individual elements were found by back-substituting the known nodal displacements into the stiffness equations for element 1.Then element force was found by applying to element 1:

$$\begin{bmatrix} \boldsymbol{k_1} & \boldsymbol{-k_1} \\ \boldsymbol{-k_1} & \boldsymbol{k_1} \end{bmatrix} \begin{Bmatrix} \boldsymbol{u_1} \\ \boldsymbol{u_2} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{f_1} \\ \boldsymbol{f_2} \end{Bmatrix} \tag{28}$$

Note that f_1 and f_2 have the same magnitude but with opposite signs. Therefore, the stress in each element can be found by apply the equation:

$$\boldsymbol{\sigma_i} = \left|\frac{\boldsymbol{f_2}}{\boldsymbol{A_i}}\right| \tag{29}$$

where i refers to the ith element, f_2 is used for calculation and A represents the corresponding cross-sectional area of each element. This calculation was repeated for each element.

Moreover, the non-linear finite element equation for the non-linear spring used to replace the broken bar element can be expressed below in shorthand matrix form:

$$[K]\{U\} + \{F_I\} = \{F\} \tag{30}$$

where F_I stands for the original stiffness of the non-linear spring. Equilibrium iterations were performed to ensure the convergence of the system. The unbalanced load $\{\Psi\}$ was calculated as follows, and the system would reach to the equilibrium state when $\{\Psi\}$ reached to zero:

$$\{\Psi\} = \{F\} - ([K]\{U\} + \{F_I\}) \tag{31}$$

3.5 Non-linear Solution Algorithm

A flow chart of the non-linear solution algorithm is shown in Figure 22. Among the inputs that need to be defined at the top of the code are: the meshtype (the type of mesh), nele (total number of elements comprising a strand), nstrand (number of parallel strands), Ltot (the total length of each strand) etc. The complete list of inputs appears in the matlab script BeamBarNonLinear.m in the appendix.

Variation of element length within each strand is controlled by the input variable meshtype. The following three options are available: (1) elements with same constant

length, (2) length evenly randomized in a controlled range and (3) length corresponding to experimental pore distribution sizes.

The global stiffness matrix [K] and the incremental force vector {F} were assembled using the direct stiffness method. As introduced in the previous section 2.2.2, four types of global stiffness matrix were used in this model: $[K]_0$, $[K]_c$, $[K]_{BC}$, $[K]_{cbc}$. $[K]_0$ is the initial linear stiffness matrix which was saved in the beginning of the model. $[K]_c$ represents the current (non)-linear stiffness matrix which depends on the current values of the nodal displacements due to the non-linear springs inserted to the strand. $[K]_c$ equals $[K]_0$ when there is no bar element breaking. $[K]_{BC}$ and $[K]_{cbc}$ are derived from $[K]_0$ and $[K]_c$ respectively by applying boundary conditions and it is used to enforce current nodal displacement before and during the equilibrium iterations. The corresponding finite element equations were then solved simultaneously to determine incremental nodal displacements, $\{U\}$. Substitution of 'total' nodal displacements into the element stiffness equations for element 1 yields the force in the strand from which element stresses can be calculated.

Equilibrium iterations were performed before determining element stresses, the unbalanced load vector $\{\Psi\}$ was calculated to determine if the strand reached to an equilibrium state, and the current global stiffness matrix $[K]_c$ was modified till the system finally converged.

Three factors of safety were computed for each bar element, one for each potential failure mode: axial compression, Euler buckling, and buckling with initial curvature. The bending moment and the maximum stress of beam elements were found to

determine their factors of safety as well. The smallest of all these factors of safety identifies the critical failure theory for that element. The element with the smallest overall factor of safety is presumed to be next in line to fail. If the factor of safety of this critical element is found to be greater than one, the analysis for that load increment is deemed complete, the total force vector, {F}, is updated and the algorithm returns to the top to begin analysis of the next load step. If, on the other hand, the factor of safety is less than 1, the critical element is modeled as having failed. Broken beam elements and broken bar elements were treated differently in this model. If a beam element broke, it is effectively removed from the system. If, instead, a bar element broke, the broken element will be replaced by a non-linear spring, which will be used to simulate the densification of the strand. Since failure of the element corresponds to pore collapse, nodal values of position, total displacement and total force have to be adjusted accordingly. Iterations within the same load increment continue until the critical factor of safety rises above 1.

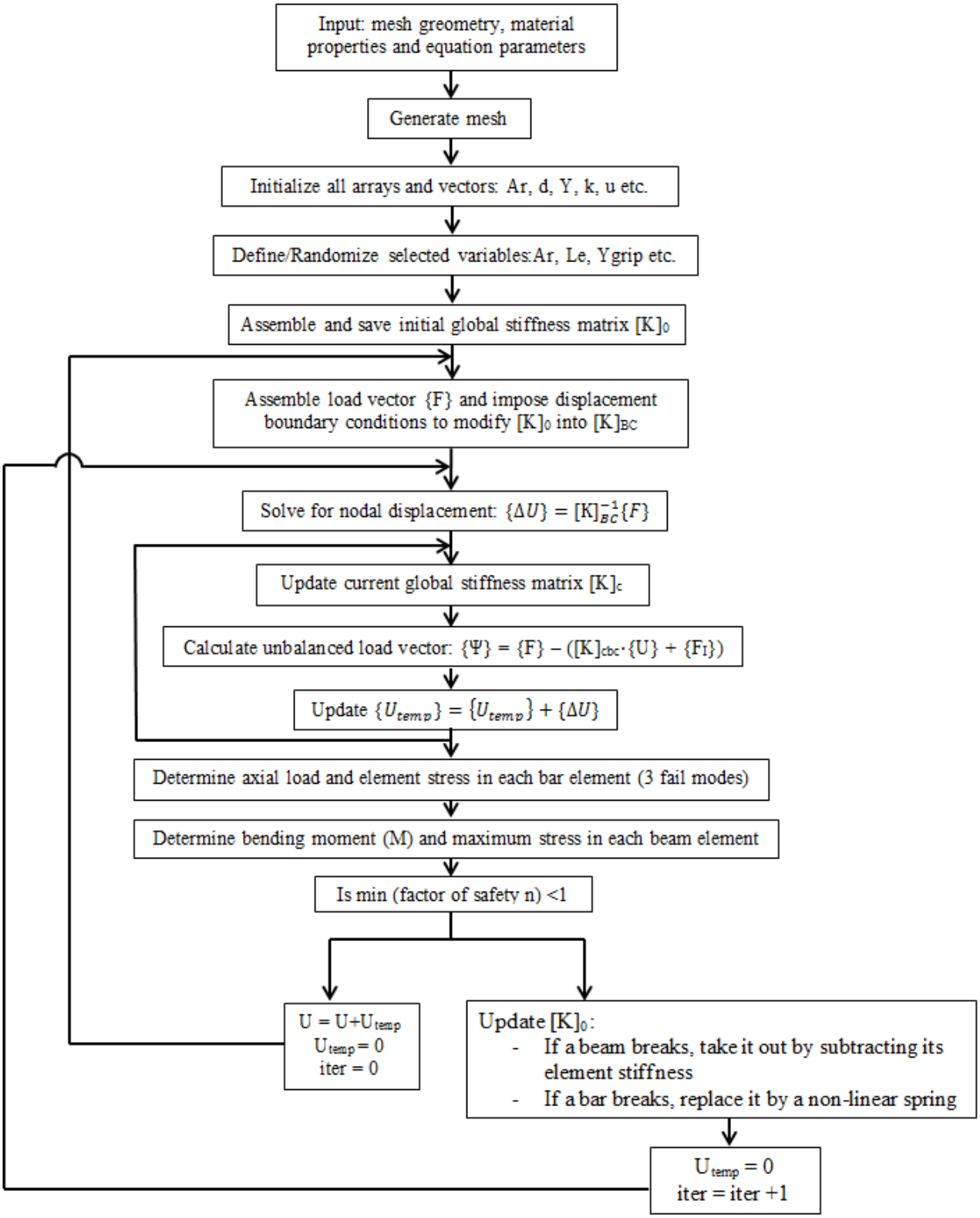


Figure 22. The flow chart of the non-linear model algorithm.

3.6 Results

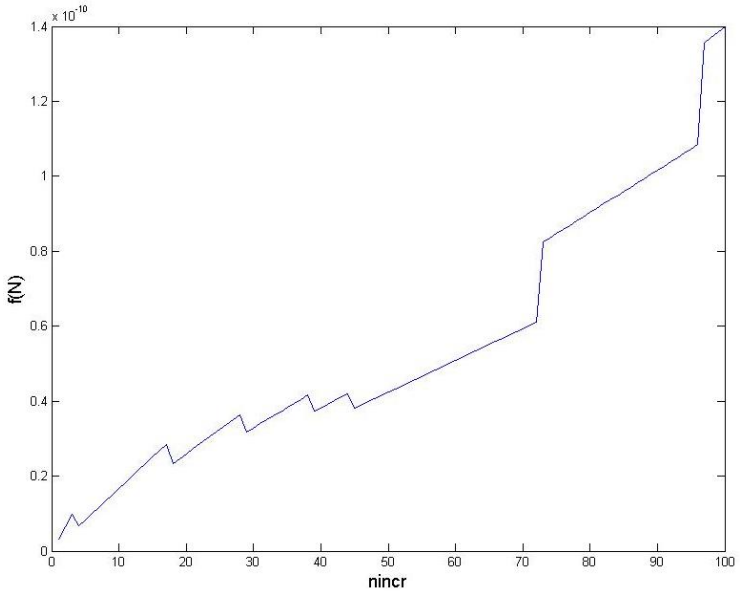
The code was run with six and ten bar elements. The length of each bar and beam element was set to be equal to each other. Other inputs were defined and shown in Table 1.

Table 1. The value of input variables.

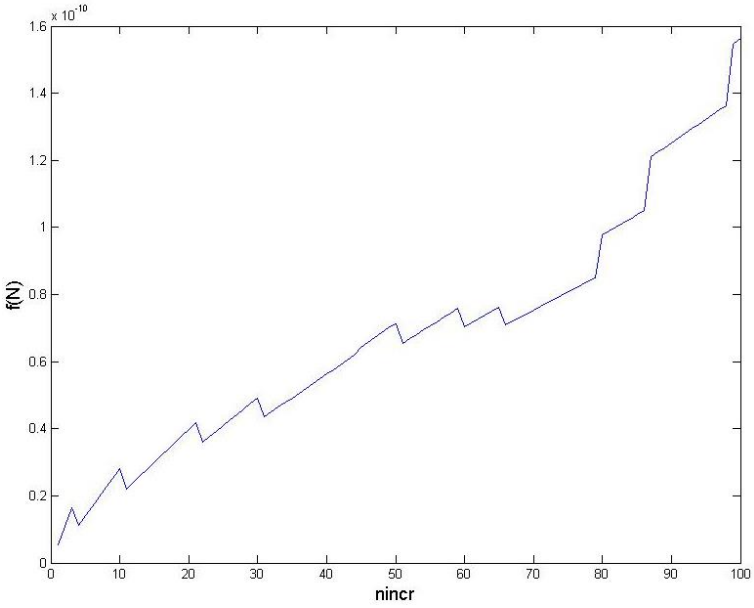
Ltot	1.00E-06	m	The total length of the strand
Di	3.49E+09	m	The initial diameter
E	7.30E+10	pa	The elastic modulus of the aerogel
nincr	100	-	The number of displacement increments in elastic range
Ugripmax	-8.00E-07	m	The total displacement of grip
Su	1.10E+08	pa	The ultimate stress
fracl	1.00E-01	-	The fraction of element length
Bnorm	9.00E-01	-	The fraction of the nodal force at the end to check convergence
Be	1.00E-01	-	The constant beta for non-linear spring stress function
Ae	6.64E+02	-	The constant alpha for non-linear spring stress function(=E/Su+Be)

The corresponding graphs of f (downward force applied at the top of the mesh) vs. nincr (number of increments) for six and ten bars are respectively shown in Figure 23 (a) and (b). The force builds up till one bar element breaks and then drops due to the fail of the element. Different from the previous single strand model, the force does not fall to zero (as shown in Figure 12) because the broken bar element is replaced by a non-linear contact spring. Instead, the system will go through the equilibrium iterations till it converges and the force increases again until another bar element breaks. Therefore, each peak shown in the graphs represents the force value that causes one bar element to fail. After all bar and beam elements break, only contact springs are left in the system and it just goes through the equilibrium iterations repeatedly. This causes the structure to gradually stiffen, after about 60 increments in Figure 23 (a) and 75 increments in Figure

23 (b). In addition, due to the limited number of increments, sometimes more than one bar breaks in one increment, which results in missing peaks in the graphs.



(a)



(b)

Figure 23. The graphs of f (downward force applied at the top of the geometry) vs. nincr (number of increments) with different number of bar elements: (a) six bars; (b) 10 bars.

To better understand how the force changes during the process of compression, the graph of f (downward force applied at the top of the mesh) vs. number of iterations for ten bar elements was generated, shown in Figure 24. Note that iterations were used to obtain a converged solution whereas increments relate to different positions of the grip. Because it is a plot of force versus number of iterations, compared to Figure 23 (b), it shows more details about how the force was changing. The three sharp peaks in Figure 24 indicate that during the equilibrium iterations, the corresponding non-linear spring stress function was actually switched to function (1) and then jumped back to function (2). The fact that the force dropped to a reasonable level shortly after the sharp peak demonstrates the effectiveness of equilibrium runs.

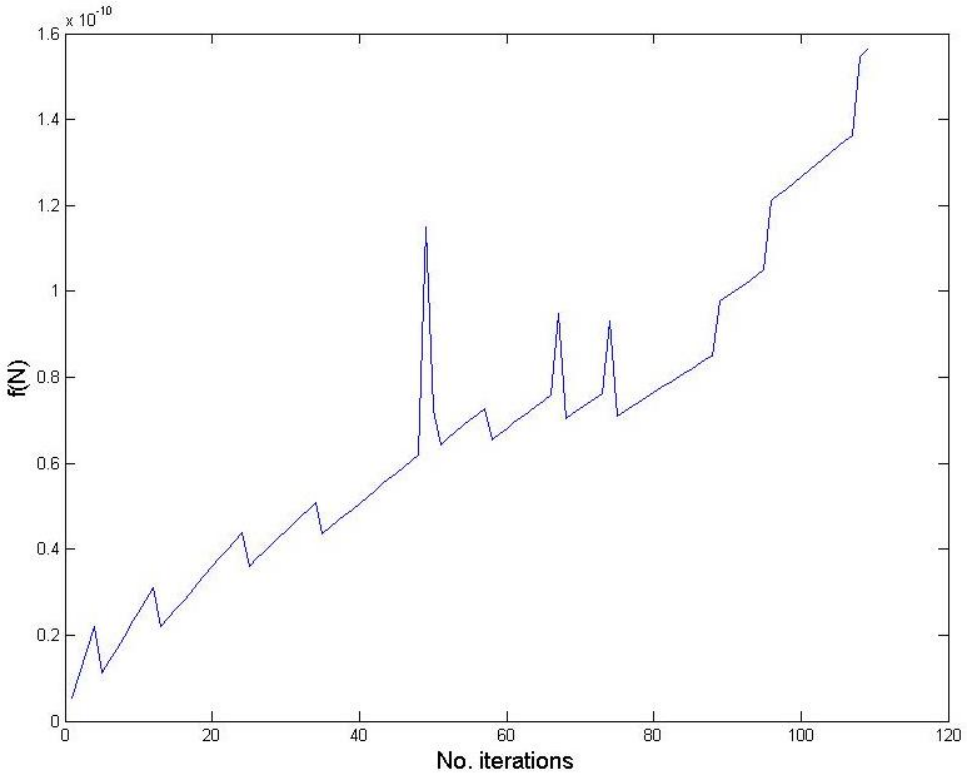


Figure 24. The graph of f (downward force applied at the top of the mesh) vs. No. Iterations (the corresponding number of iterations).

In order to check the convergence of the system after equilibrium iterations, a graph of element force versus number of iterations was created for the case of six bar elements, as shown in Figure 25. The element forces for all six bars were saved after each time the system converged. As shown in this graph, the plots of six bars overlap each other, which imply that, the magnitude of the forces in six bar elements were equal to each other as they should have been. The numerical results were double checked as well. This finding shows that the system is converged in a correct way. Note that this graph looks very similar to Figure 23 (a), because the element force is supposed to be the same in all bar elements, including the force applied at the top.

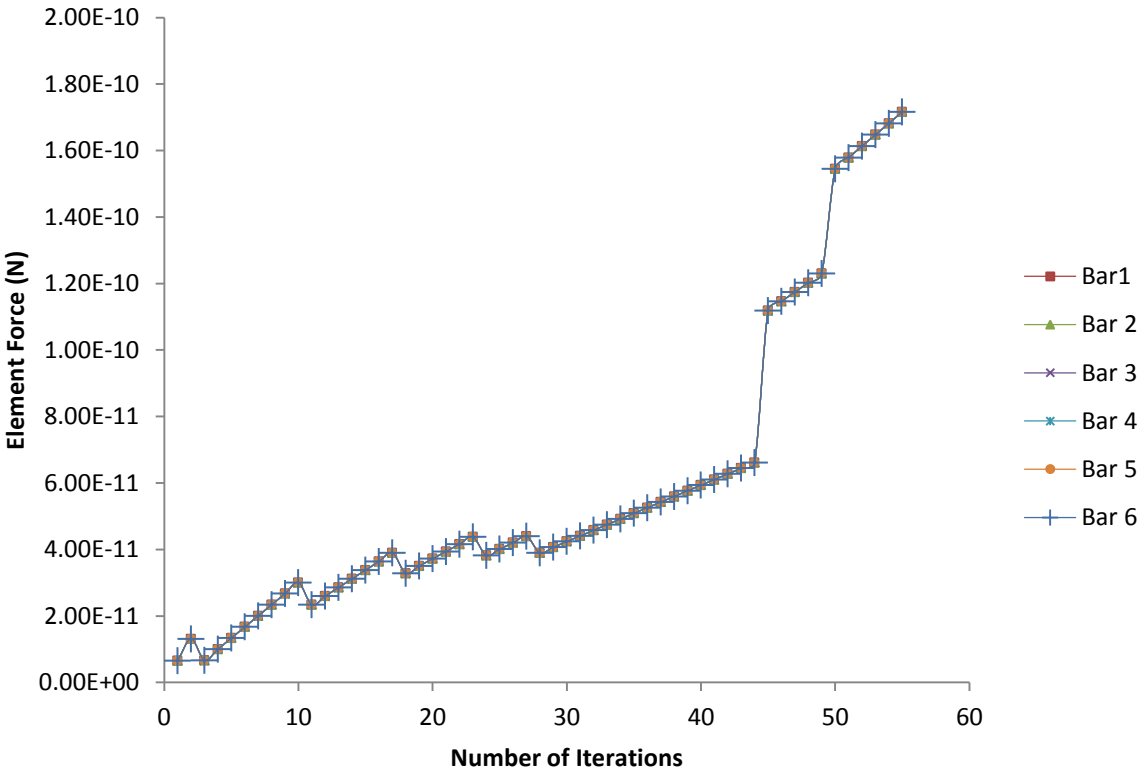


Figure 25. The graph of element force vs number of iterations for six bar elements. The element forces for different bars were shown in different shapes of plotted points.

It was found that all bar elements fail in the third failure mode (buckling with initial curvature), except the first bar, which failed due to the second failure mode (Euler buckling for simply supported ends). In addition, the beam elements do not have a significant impact on the results since one ends of the beam elements are fixed, which will cause most of the beam elements to fail before bar elements break. However, once multiple strands are applied to the model, the impact of the beam elements will affect the results over a wider range.

3.7 Conclusions and Discussions

The results obtained from this non-linear finite element model of pore collapse under compression were reasonable, which implies that the code was working in the expected way. The contact problem between broken elements and fragmentation were treated efficiently in this model by adding non-linear contact springs. The equilibrium iterations were performing correctly to ensure the convergence of the system. The biggest difference between this model and the previous single strand model is that the forces acting at the top do not fall to zero after bar elements break because of the addition of the contact springs. If the contact springs were to be made a lot softer, the two sets of results would begin to look similar. Different meshes can be applied to this model and by collecting these unit cells, a three dimensional model can be developed.

4 References

[1] Nikel, Ondrej, Ann M. Anderson, Mary K. Carroll, and William D. Keat. "Effect of Uni-axial Loading on the Nanostructure of Silica Aerogels." *Journal of Non Crystalline Solids* 357 (2011): 3176-183.Web. 6 Aug. 2012.

[2] Xie Lutao and William D. Keat " Modeling The Effect of Pore Distribution on the Strength of Silica Aerogel." *Summer Research Report*. Union College. 10 Aug. 2012.

[3] Marliere, C., T. Woignier, P. Dieudonne, J. Primera, M. Lamy., and J. Phalippou. "Two Fractal Structures in Aerogel." *Journal of Non-crystalline Solids* 285 (2001): 175-180. Web. 9 Aug. 2012

[4] "Fractal." *The Free Dictionary*. Farlex, n.d. Web. 9 Aug. 2012. <http://www.thefreedictionary.com/fractal>.

[5] Schaefer, Dale W., and Keith D. Keefer. "Structure of Random Porous Materials: Silica Aerogel." Physical Review Letters 56.20 (1986): 2199-202. Print.

[6] Courtesy of Prof. Ann Anderson

[7] Xie Lutao and Ann M. Anderson " Effect of Sample Preparation and Gas Sorption Euilibration Time on Measurement of the Textural Properties of Silica Aerogel Material." *Sophomore Scholars Project Report*. Union College. 15 July. 2012.

[8] Xie Lutao and William D. Keat " Characterization of Microstructure from Pore Density and Load-displacement Data." *Summer Research Report*. Union College. 4 Aug.2013.

[9] Bardenhagen, S., A. Brydon, and J. Guilkey. "Insight into the Physics of Foam Densification via Numerical Simulation." Journal of the Mechanics and Physics of Solids 53.3 (2005): 597-617. Print.

[10] Brydon, A., S. Bardenhagen, E. Miller, and G. Seidler. "Simulation of the Densification of Real Open-celled Foam Microstructures." Journal of the Mechanics and Physics of Solids 53.12 (2005): 2638-660. Print.

[11] Daphalapurkar, N. P., J. C. Hanan, N. B. Phelps, H. Bale, and H. Lu. "Tomography and Simulation of Microstructure Evolution of a Closed-Cell Polymer Foam in Compression." *Mechanics of Advanced Materials and Structures* 15.8 (2008): 594-611. Print.

[12] Ransing, R. S., R. W. Lewis, and D. T. Gethin. "Using a Deformable Discrete-

Element Technique to Model the Compaction Behaviour of Mixed Ductile and Brittle Particulate Systems." *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 362.1822 (2004): 1867-884. Print.

[13] Gong, L., S. Kyriakides, and W. Jang. "Compressive Response of Open-cell Foams. Part I: Morphology and Elastic Properties." *International Journal of Solids and Structures*42.5-6 (2005): 1355-379. Print.

[14] Gong, L., and S. Kyriakides. "Compressive Response of Open Cell Foams Part II: Initiation and Evolution of Crushing." *International Journal of Solids and Structures* 42.5-6 (2005): 1381-399. Print.

[15] Beer, Ferdinand P., Jr. E. Russell Johnston, John T. DeWolf, and David F. Mazurek. Mechanics of Materials. 6th ed. New York: McGraw-Hill, 2010. Print.

[16] Courtesy of Prof. Keat William.

5 Appendix

5.1 OneDSingleStrandModel.m

```
%OneDSIngleStrandModel
%Lutao Xie
clc
clear
format compact
rng default

meshtype = input ('Enter the type of mesh:')%Input types of mesh
generators
nele = input('Enter the number of elements:')%The total number of
elements
%The number of strand
nstrand = 1
%The total length of axial member:10e-7 m
Ltot = 1.0000e-06
%diameter : 3.49e-9
di = 3.49e-9
%Youngs modulus:73e9
E = 7.3000e+10
%Number of displacement increments in elastic range:20
dincr = 40
%Total displacement(grip):2e-7
Ugripmax =-8.0000e-07
%Ultimate Stress:110e6
Su =110000000
%fraction of element length
fracl = 0.1
alpha = 0.9


%generate the mesh
nn = nele+1; % number of nodes
L = Ltot/nele; % calculate L for each element
beta=Ugripmax/Ltot;% parameter for the ratio of Ugripnax and Ltot

for i = 1:nstrand

if meshtype ==1
    Le = zeros(1,nele); %length of each element
    for nl = 1:nele
        Le(nl) = L;
    end
elseif meshtype ==2
    Le = zeros(1,nele); %length of each element
    R = input ('The randomness of Le:')
    for nl = 1:nele
        Le(nl)= L+(R*L*rand(1,1));
    end
elseif meshtype ==3
    PoreD = xlsread('PoreD','B2:B101');
```

```
    NPore = xlsread('PoreD','C2:C101');
    NP = round(nele*NPore/sum(NPore));
    Le = zeros(1,sum(NP)); %length of each element
    for nL = 1:length(NP)
        le = PoreD(nL)*ones(1,NP(nL));
        Le = [Le,le];
    end
    Le(find(Le==0))=[];
    Ltot = sum(Le);
    Ugripmax=beta*Ltot;
    nele = length(Le);
    nn = nele+1;
end

%zero all arrays and vectors
Ar = zeros(1,nele); %cross sectional area
d = zeros(1,nele); %element diameter
Y = zeros(1,nn); %y coordinate of each node
k = zeros(1,nele); %element stiffness
u = zeros(1,nn); %deformation of the lattice
urb = zeros(1,nn); %deformation due to pore collapse

for na = 1: nele
    d(na)= di+0.5*di*(rand(1,1)-0.5); %randomize diameter
    Ar(na)= pi*d(na)^2/4;
end

%calculating tavg
tsum = 0;
for tn = 1:nele
    t=4/3*pi()*(d(tn)/2)^3;
    tsum = tsum +t;
end
tavg = sqrt(tsum/Ltot);
tavg = 2.6e-8;

Y = [0,cumsum(Le)]; % calculate Y-coord for each node
Ygrip = Y(nn);

for kn = 1:nele
    k(kn) = Ar(kn)*E/Le(kn);%calculate element stiffness matrix [k]
    kd(kn) = 1/k(kn);
end
keq = 1/sum(kd);
Pcr = -min(Ar)*Su;
deltUgrip = Pcr/keq/dincr; %estimate initial end displacement increment

if i==1
    nincr = round(Ugripmax/deltUgrip); %calculate no. of disp.
increments
    fs = zeros(1,nincr);
    utots = zeros(1,nincr);
    ygrips = zeros(1,nincr);
else
    nincr =nincr;
end
```

```
FL = []; %force at the end
UTOT = []; %total deformation
YGRIP = []; %y coordinate of the grip

icount = 0;

B1 = 0;
B2 = 0;
B3 = 0;

for incr = 1:nincr

    iter = 1;
    while iter >0
    iter = 0;
    icount = icount+1;
%calculate element stiffness
        for kn = 1:nele
            k(kn) = Ar(kn)*E/Le(kn);
        end

%assemble element stiffness matrix [k] into global stiffness matrix K
    K = zeros(nn,nn);
            for nK = 2:nn-1
                K(nK,nK)=k(nK)+k(nK-1);
                K(nK-1,nK)= -k(nK-1);
                K(nK+1,nK)= -k(nK);
            end
    K(1,1)= k(1);
    K(2,1) = -k(1);
    K(nn,nn)= k(nele);
    K(nn-1,nn)= -k(nele);

%impose displacement boundary conditions
%modify [K]
    K(1,1)= K(1,1)*10e7;
    K(nn,nn)= K(nn,nn)*10e7;

%assemble {f}
    deltF = zeros(nn,1);

%define the value of the incremental displacement deltUend
    if Ygrip >= Y(nn)
        if Ygrip-Y(nn)<abs(deltUgrip)
            deltUend = -(abs(deltUgrip)-(Ygrip-Y(nn)));
        else
            deltUend = 0;
        end
    else
        deltUend = deltUgrip;
    end

    deltF(nn)= K(nn,nn)*(deltUend);
```

```
%solve for nodal disp.
    deltU = K\deltF;

%update Utemp
    Utemp = u + deltU';

%calculate axial load and element stress
N1 = [];
N2 = [];
N3 = [];
    for nf = 1:nele
        kl = k(nf)*[1 -1;-1,1]; %assemble k for one element
        ul = [Utemp(nf);Utemp(nf+1)]; %assemble l for one element
        fl = kl*ul; %axial load
        st = abs(fl(2))/Ar(nf); %element stress
        Pcr = pi()^3*E*(d(nf))^4/(64*Le(nf)^2); %critical buckling load
for simply supported column

        n1 = abs(Su/st); %element factor of safety
        n2 = abs(Pcr/Ar(nf)/st); %buckling

        e = fracl*Le(nf); %max eccentricity
        M = abs(fl(2))*e/(1-abs(fl(2))/Pcr); %moment
        stmaxcomp = abs(st)+abs(32*M/(pi()*d(nf)^3));%maximum
compresstion stress
        n3 = Su/stmaxcomp; %simply supported column with initial
curvature
        min12 = min(n1,n2);
        n(nf) = min(min12,n3);%find the minimum factor of safety
        N1 = [N1,n1];
        N2 = [N2,n2];
        N3 = [N3,n3];
    end

    if min(n) < 1
        B11 = any(min(n)==N1);
        B22 = any(min(n)==N2);
        B33 = any(min(n)==N3);
        if B11 == 1
            B1=B1+1;
        elseif B22 == 1
            B2=B2+1;
        else
            B3=B3+1;
        end % Counting No. of broken elements of each type of failure
mode
        ind = find(n==min(n)); %find the broken element
        for indrb = 1:ind     %define the displacement caused by pore
collapse
            disppore(indrb)= 0; %displacement of nodes under broken
element=0
        end
        for indY = (ind+1):nn
            disppore(indY) = -0.75*Le(ind); %displacement of nodes
above broken element=-Le(ind)
        end
```

```
        Y = Y+disppore; %update Y
        urb = urb +disppore; %update urb
        u = zeros(1,nn);
        Ar(ind) = Ar(ind)/(1-alpha); %update broken element - Ar
        Le(ind) = Le(ind)*(1-alpha); %update broken element - L

        if Ygrip-Y(nn) >= abs(deltUgrip) %Is Ygrip - Yend >= Ugrip?
            Ygrip = Ygrip + deltUgrip;
            Utot = u + urb;
            fl(2) = 0;
            iter = 0;
        else
            iter = iter +1;
        end
    else
        Ygrip = Ygrip + deltUgrip;
        u = Utemp;
        Utot = u + urb;
        iter = 0;
    end

    end %corresponds to while loop :iter>=0

    UTOT = [UTOT,abs(Utot(nn))];
    FL = [FL,abs(fl(2))];
    YGRIP = [YGRIP,Ygrip];
end %corresponds to for loop:incr = 1:nincr

fs = fs+FL;
utots = utots+UTOT;
end

fs = fs/nstrand;
utots = utots/nstrand;

Ygripin = Ltot*ones(1,length(YGRIP));
deltYgrip = abs(YGRIP-Ygripin);

newstrs = fs/(tavg^2);
newstrn = deltYgrip/Ltot;

expstrs = xlsread('PoreD','H2:H658');
expstrn = xlsread('PoreD','G2:G658');

figure;
plot(deltYgrip,fs,'r')
xlabel('ygrip - ygrip(initial)','fontsize',12);
ylabel('f','fontsize',12);
figure;
plot([1:nincr],utots,'b')
xlabel('nincr','fontsize',12);
ylabel('utots','fontsize',12);
figure;
if meshtype==3
```

```
    plot(newstrn,newstrs,'k')
    xlabel('strain','fontsize',12);
    ylabel('stress','fontsize',12);
    hold on
    plot(expstrn,expstrs,'r')
    hold off
else
    plot(newstrn,newstrs,'k')
    xlabel('strain','fontsize',12);
    ylabel('stress','fontsize',12);
end
```

5.2 SingleStrandModelWthBeam.m

```
%Non-linear Finite Element Code
%Lutao Xie
clc
clear
format compact
rng default

meshtype = input ('Enter the type of mesh:')%Input types of mesh
generators
nele = input('Enter the number of bar elements:')%The total number of
elements

%The number of strand
nstrand = 1
%The total length of axial member:10e-7 m
Ltot = 1.0000e-06
%diameter : 3.49e-9
di = 3.49e-9
%Youngs modulus:73e9
E = 7.3000e+10
%Number of displacement increments in elastic range:20
nincr = 100
%Total displacement(grip):2e-7
Ugripmax =-8.0000e-07
%Ultimate Stress:110e6
Su =11e7
%fraction of element length ( max eccentricity)
fracl = 0.1
%other parameters
%fraction of P (force) to check if psi converges
Bnorm = 0.05
%the constant beta for non-linear spring equation
Be = 0.1
%the constant alpha for non-linear spring equation
Ae = E/Su+Be
%initialize the index for broken elements
StrandInd = zeros(2*nele+1,1)

%generate mesh by entering mesh number
nn = nele+1; % number of nodes
```

```
L = Ltot/nele; % calculate L for each element
beta=Ugripmax/Ltot;% parameter for the ratio of Ugripmax and Ltot

if meshtype ==1 %same length for each element
    Le = zeros(1,nele); %length of each element
    for nl = 1:nele
        Le(nl) = L;
    end
elseif meshtype ==2 %randomized length of each element
    Le = zeros(1,nele); %length of each element
    R = input ('The randomness of Le:')
    for nl = 1:nele
        Le(nl)= L+(R*L*rand(1,1));
    end
elseif meshtype ==3 %lengths matching pore distribution data
    PoreD = xlsread('PoreD','B2:B101');
    NPore = xlsread('PoreD','C2:C101');
    NP = round(nele*NPore/sum(NPore));
    Le = zeros(1,sum(NP)); %length of each element
    for nL = 1:length(NP)
        le = PoreD(nL)*ones(1,NP(nL)); %matching pore distribution data
        Le = [Le,le];
    end
    Le(find(Le==0))=[]; %eliminate elements with length of zero
    Ltot = sum(Le); %calculate total length of the strand
    Ugripmax=beta*Ltot; %define Ugripmax
    nele = length(Le);  %define number of elements
    nn = nele+1; %define number of nodes
end

%zero all arrays and vectors
Ar = zeros(1,nele); %cross sectional area
d = zeros(1,nele); %element diameter
Y = zeros(1,nn); %y coordinate of each node
k = zeros(1,nele); %element stiffness
u = zeros(1,nn); %deformation of the lattice
StrandInd = zeros(2*nele+1,1);% StrandIndications: if an element breaks
or not
Utemp = zeros(1,nn); %temparory nodal displacement

%Define diam, Ar, Y-coord, deltUgrip
for na = 1: nele
    d(na)= di+0.5*di*(rand(1,1)-0.5); %randomize diameter
    Ar(na)= pi*d(na)^2/4; %calculate cross-sectional area
end
Y = [0,cumsum(Le)]; % calculate Y-coord for each node
Ygrip = Y(nn); % define y coordinate of grip
deltUgrip = Ugripmax/nincr; %movement of grip in each increment


%Assemble initial stiffness matrix [K0]

%Strand element: Kstrand
    for kn = 1:nele
        ks(kn) = Ar(kn)*E/Le(kn);%calculate element stiffness
    end
```

```
    Kstrand = zeros(nn,nn);%assemble global stiffness matrix for strand
            for nK = 2:nn-1
                Kstrand(nK,nK)=ks(nK)+ks(nK-1);
                Kstrand(nK-1,nK)= -ks(nK-1);
                Kstrand(nK+1,nK)= -ks(nK);
            end
    Kstrand(1,1)= ks(1);
    Kstrand(2,1) = -ks(1);
    Kstrand(nn,nn)= ks(nele);
    Kstrand(nn-1,nn)= -ks(nele);

%Beam element: Kbeam
%Define length and area of each beam element
Leb = zeros(1,nn);
for ml = 1:nn
    Leb(ml) = L; %length of beam from mesh type 1
end
for ma = 1: nn
    db(ma)= di+0.5*di*(rand(1,1)-0.5); %randomize diameter for beams
    Arb(ma)= pi*db(ma)^2/4; %calculate cross-sectional area of beams
end

for km = 1:nn
%     kb(km)=3*E*pi()*(db(km)/2)^4/Leb(km)^3; %beam element stiffness
      kb(km) = 0;
end

Kbeam = zeros(nn,nn);
for mK = 1:nn
    Kbeam(mK,mK)=kb(mK); %assemble global stiffness matrix for beams
end
K0 = Kstrand +Kbeam; %K0

% for loop + while loop
FL = []; %force at the end (for graphing)
UTOT = []; %total deformation (for graphing)
YGRIP = []; %y coordinate of the grip (for graphing)
FTOP = []; %force at the top (for graphing)
FL = [];%force acting at each element causing bar broke

icount = 0; % for checking while loop

B1 = 0; %for counting number of bars fail in type 1 mode
B2 = 0; %for counting number of bars fail in type 2 mode
B3 = 0; %for counting number of bars fail in type 3 mode
B4 = 0; %for counting number of beams

F1 = [];% element for before equilibrium iterations
F2 =[];% element for after equilibrium iterations
CEQ = [];% counting number of equilibrium iterations
FL = [];% force in each element in stress/factor of safety calculations

for incr = 1:nincr;
```

```
KSP1 = [];
KSP2 = [];

    iter = 1;
    while iter >0
    iter = 0;
    icount = icount+1; %counting the number of times it went through
while loop

%Assemble Kbc
%Impose boundary conditions
    Kbc = K0;
    Kbc(1,1)= K0(1,1)*10e7;
    Kbc(nn,nn)= K0(nn,nn)*10e7;
    Kbc;

%Assemble {f}
    deltF = zeros(nn,1);
    deltF(nn)= Kbc(nn,nn)*(deltUgrip);


%solve for nodal disp.
    deltu = Kbc\deltF;
    deltU = deltu'; %resize the vector from column form to row form
update Utemp ( 1)
    Utemp = Utemp + deltU;

% Grabbing element force before iteration
uele = u+Utemp;
f1 = [];
for nstr = 1:nele
    uc = [uele(nstr);uele(nstr+1)];
    kele = ks(nstr)*[1 -1;-1 1];
    f11 = kele*uc;
    f1 = [f1,f11];
end

%Calculate Psi and check convergence
%Define function and update kc
eqn1 = 0; %count for the times it went through eqn 1
counteq = 0;

%run equilibrium iterations

for eq = 1:2000

    for ii = 1:nele %going through the strand index vector
    if StrandInd == 0
        Kc = K0;
        FI = zeros(nn,1);
    else
        Kc = Kc;
    end
    Uupdate = u+Utemp;
```

```
    if StrandInd(ii) == 1.1 || StrandInd(ii) == 1.2 %find the broken
bar element

%     Uupdate = u+Utemp;
    delta = Le(ii)-Uupdate(ii)+Uupdate(ii+1);%determine delta
    kspring1 = Ar(ii)*(Ae-Be)*Su/d(ii);%the new stiffness from equation
1
    kspring2 = Ar(ii)*Be*Su/(Le(ii)-d(ii));%the new stiffness from
equation 2

    if delta <= 0.1*Le(ii)
        % Approach 2: directly edit Kc - equation 1
        if StrandInd(ii) == 1.1;
            Kc = Kc;
        elseif StrandInd(ii)== 1.2;
            Kc(ii,ii) = Kc(ii,ii)-kspring2+kspring1;
            Kc(ii,ii+1) = Kc(ii,ii+1)-(-kspring2)+(-kspring1);
            Kc(ii+1,ii) = Kc(ii+1,ii)-(-kspring2)+(-kspring1);
            Kc(ii+1,ii+1) = Kc(ii+1,ii+1)-kspring2+kspring1;
            StrandInd(ii) = 1.1;
        end
        % assemble force vector, matching equation 1
        fi = zeros(nn,1);
        fi(1) = -1;
        fi(nn) =1;
        FI =(Ar(ii)*(Ae-Be)*Su*Le(ii)/d(ii))* fi;
        eqn1 = eqn1 +1;
    else
        % Approach 2: directly edit Kc - equation 2
        if StrandInd(ii) == 1.2;
            Kc = Kc;
        elseif StrandInd(ii) == 1.1;
            Kc(ii,ii) = Kc(ii,ii)-kspring1+kspring2; %update stiffness
            Kc(ii,ii+1) = Kc(ii,ii+1)-(-kspring1)+(-kspring2); %update
stiffness
            Kc(ii+1,ii) = Kc(ii+1,ii)-(-kspring1)+(-kspring2); %update
stiffness
            Kc(ii+1,ii+1) = Kc(ii+1,ii+1)-kspring1+kspring2; %update
stiffness
            StrandInd(ii) = 1.2;
        end
        % assemble force vector
        FI = zeros(nn,1);
    end
    else
        Kc=Kc;
    end
    end %corresponds to ii = 1:nele

%Assemble p (nodal force)

    kc1 = Kc(1,1)*[1 -1;-1,1]; %assemble k for first element Use
element
    ueq1 = [Uupdate(1);Uupdate(2)]; %assemble l for one element
    p1 = kc1*ueq1; %axial load at tht bottom
```

```
    kcend = Kc(nn,nn)*[1 -1;-1,1]; %assemble k for first element
    ueqend = [Uupdate(nn-1);Uupdate(nn)]; %assemble l for one element
    pend = kcend*ueqend; %axial load at the top

    p = zeros (nn,1);
    p(1) = p1(1);
    p(nn) = pend(1);

%calculating psi (3)
    psi = p - (Kc*(Uupdate') + FI);
    psi(1) = 0;
    psi(end) = 0;

    Kcbc = Kc; %add boundary conditions to Kc
    Kcbc(1,1)= Kc(1,1)*10e7;
    Kcbc(nn,nn)= Kc(nn,nn)*10e7;

    deltU = Kcbc\psi;  %use K0 with boundary conditions ( switch to Kc?
)
    Utemp = Utemp + deltU' ; % update Utemp

end %corresponds to while eq > 0

% Grabbing element force after equilibrium iterations
uele2 = u+Utemp;
f2 = [];
for nstr = 1:nele
    uc2 = [uele2(nstr);uele2(nstr+1)];
    if StrandInd(nstr) == 1.1;
        kele2 = (Ar(nstr)*(Ae-Be)*Su/d(nstr))*[1 -1;-1 1];
    elseif StrandInd(nstr) == 1.2;
        kele2 = (Ar(nstr)*Be*Su/(Le(nstr)-d(nstr)))*[1 -1;-1 1];%the
new stiffness from equation 2
    else
    kele2 = ks(nstr)*[1 -1;-1 1];
    end
    f22 = kele2*uc2;
    f2 = [f2,f22];
end

F1 = [F1,f1];
F2 = [F2,f2];
CEQ = [CEQ,counteq];


%calculate axial load and element stress
N1 = [];
N2 = [];
N3 = [];

% finding force applied at the top
utot = u + Utemp;
if StrandInd(nele) == 1.1;
        ktop = (Ar(nele)*(Ae-Be)*Su/d(nele))*[1 -1;-1 1];
    elseif StrandInd(nele) == 1.2;
```

```
        ktop = (Ar(nele)*Be*Su/(Le(nele)-d(nele)))*[1 -1;-1 1];%the new
stiffness from equation 2
    else
    ktop = ks(nele)*[1 -1;-1 1];
    end
utop = [utot(nele);utot(nele+1)];
ftop = ktop*utop;

Fl = [];
Fll = [];
    for ns = 1:nele
        if StrandInd(ns) == 0
        kl = ks(ns)*[1 -1;-1,1]; %assemble k for one element
        ul = [utot(ns);utot(ns+1)]; %assemble ul for one element
        fl = kl*ul; %axial load
        st = abs(fl(2))/Ar(ns); %element stress
        Pcr = pi()^3*E*(d(ns))^4/(64*Le(ns)^2); %critical buckling load
for simply supported column
        st2 = Pcr/Ar(ns);

        n1 = abs(Su/st); %axial load
        n2 = abs(st2/st); %Euler buckling

        e = fracl*Le(ns); %max eccentricity
        M = abs(fl(2))*e/(1-abs(fl(2))/Pcr); %moment
        stmaxcomp = abs(st)+abs(32*M/(pi()*d(ns)^3));%maximum
compresstion stress
        n3 = Su/stmaxcomp; %simply supported column with initial
curvature
        else
            fl = 0;
            n1 = 2000;
            n2 = 2000;
            n3 = 2000;
        end
        nst(ns) = min([n1,n2,n3]);%find the minimum factor of safety
        N1 = [N1,n1];
        N2 = [N2,n2];
        N3 = [N3,n3];
        Fl = [Fl,fl(1)];
    end
  Fll = [Fll,Fl];

    for nb = 1:nn
        if StrandInd(nele+nb) == 0;
        Ib = pi()*db(nb)^4/64; %Ibeam
        Mb = abs(6*E*Ib*utot(nb)/Leb(nb)^2); %beam moment
        stb = abs(32*Mb/((db(nb))^3*pi()));%beam stress
        nbeam(nb) = Su/stb ;%beam n
        else
        nbeam(nb) = 1000;
        end
    end
    n = [nst,nbeam];

if min(n)<1
```

```
%    StrandInd = zeros(2*nele+1,1);
    B11 = any(min(n)==N1);
    B22 = any(min(n)==N2);
    B33 = any(min(n)==N3);
    B44 = any(min(n)==nbeam);
        if B11 == 1 %bar breaks, counting type 1
            B1=B1+1;
        elseif B22 == 1 %bar breaks, counting type 2
            B2=B2+1;
        elseif B33 == 1 %bar breaks, counting type 3
            B3=B3+1;
        else %beam breaks,counting number of beam elements break
            B4=B4+1;
        end

        if B44 == 1 %beam element breaks
            indbeam = find(n==min(n))-nele;
            Kc(indbeam,indbeam) = Kc(indbeam,indbeam) -
kb(indbeam);%update beam element (take out, kc)
            StrandInd(nele+indbeam) = 2; %Save the broken beam
element's index
        else %bar element breaks
            indbar = find(n==min(n));
            StrandInd(indbar) = 1.2; %Save the broken bar element's
index
        % update the broken bar element with the spring (equation 2);
directly edit Kc
        korg = Ar(indbar)*E/Le(indbar);%the original stiffness
        kspring = Ar(indbar)*Be*Su/(Le(indbar)-d(indbar));%the new
stiffness from the spring
        Kc(indbar,indbar) = Kc(indbar,indbar)-korg+kspring; %update
stiffness
        Kc(indbar,indbar+1) = Kc(indbar,indbar+1)-(-korg)+(-
kspring);%update stiffness
        Kc(indbar+1,indbar) = Kc(indbar+1,indbar)-(-korg)+(-
kspring);%update stiffness
        Kc(indbar+1,indbar+1) = Kc(indbar+1,indbar+1)-
korg+kspring;%update stiffness
        % assemble force vector
        FI = zeros(nn,1);
        end
        iter = iter+1;
        Utemp = zeros(1,nn);
else
    u = u+Utemp;
    Utemp = zeros(1,nn);
    iter = 0;
end

    if any(StrandInd == 0) == 0 % to check if all elements broke, if
so, end program
    break
    end

   % FTOP = [FTOP,abs(ftop(2))]; % Ftop vs. iterations
```

```
    end %while loop

    YGRIP = [YGRIP,Ygrip];
    FTOP = [FTOP,abs(ftop(2))]; % Ftop vs. nincr
    FL = [FL,Fll]; % force acting at each element that causing bar
broke
    if any(StrandInd == 0) == 0 % to check if all elements broke, if
so, end program
    break
    end

end %for incr

plot(FTOP,'b')
xlabel('nincr','fontsize',12);
ylabel('f(N)','fontsize',12);
```