WestVirginiaUniversity
THE RESEARCH REPOSITORY @ WVU

Graduate Theses, Dissertations, and Problem Reports

2004

Influence of rearrangement of actin cytoskeleton on the overall material properties of ATDC5 cells during chondrogenesis

Madhavi V. S. Ayyalasomayajula
West Virginia University

Follow this and additional works at: https://researchrepository.wvu.edu/etd

Influence Of Rearrangement Of Actin Cytoskeleton On The Overall Material Properties Of ATDC5 Cells During Chondrogenesis

Madhavi V S Ayyalasomayajula

Thesis submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science

in

Mechanical Engineering

Nilay Mukherjee, Ph.D., Chair

Jacky C Prucz, Ph.D.

Thomas M Damiani, Ph.D.

Department of Mechanical and Aerospace Engineering
Morgantown, West Virginia
2004

Abstract

Influence Of Rearrangement Of Actin Cytoskeleton On The Overall Material Properties Of ATDC5 Cells During Chondrogenesis

Madhavi V S Ayyalasomayajula

Articular cartilage is a load-bearing tissue that helps in lubrication and load distribution in diarthroidal joints. It is made up of highly specialized cells called chondrocytes that are responsible for its growth and maintenance. Mechanical loading and cell-matrix interactions have been shown to influence the biological behavior of chondrocytes. Studies show that when chondrocytes are cultured in 2D, they lose their phenotype. This change is reversed when the cells are embedded in a 3D culture system such as in agarose gel. Previous experiments in our lab showed that when prechodrocytic cells were embedded in agarose, chondrogenesis was triggered in these cells. This work attempts to understand the mechanical implications of this behavior. It also provides an estimate of the sensitivity of the apparatus that would be required to carry out compression loading experiments of gel-cell constructs that can distinguish between pre-chondrocytes and chondrocytes.

ATDC5 cells were cultured in 3D for 12 days and fluorescent images of the actin cytoskeletal structure of the cells were obtained. A major reorientation of the cytoskeletal structure (from stress-fiber to punctate appearance) was observed. This data was then used to construct a Finite Element model of cell. The cell was considered as a composite matrix and the actin as the embedded fibers. Radial deformations of the cell (difference in the corresponding undeformed and deformed radii of the cell at a particular angle w.r.t. loading axis) were calculated for 5 sets of experimental data. For each set, 2D FE models of cell (along with actin) embedded in a 0.5mm x 0.5mm agarose gel were created (using Plane Stress assumptions) for days 1, 5, 8 & 12 of 3D culture. Uniaxial compressive loading (4.84KPa) was applied on one edge of the gel, with the other edge fixed. The radial deformations of the cell were calculated and related to its mechanical properties. The model predicted that, as a result of the rearrangement of the actin, the cell became more compliant and less anisotropic during chondrogenesis, though this difference was not statistically significant when the 15µ diameter cells were embedded in 500µ cubic gels. However, as the gel size was reduced to 20µ, the RUTL (Ratio of radial deformations in the Transverse direction to that in Loading direction) was significantly different between day1 and day 12 (chondrocyte) cells. The difference in force required to compress a day1 vs a day 12 cell was 103nN. This information can be used to build MEMS based devices that can distinguish between pre-chondrocytes and chondrocytes.

This thesis is dedicated to Amma and Pithasri

Acknowledgements

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1   Introduction

1.1 Articular Cartilage:

Articular Cartilage is a load bearing tissue that lubricates the diarthrodial joints. It primarily consists of the Extra Cellular Matrix (ECM) that is made up of highly specialized cells (Chondrocytes). The ECM is made up of water, proteoglycans, collagens and other proteins, which provide the tissue with its distinctive structure and mechanical properties.

The structure of articular cartilage varies throughout its depth and can be divided into four zones (Figure 1).

The top or the 'Superficial zone' acts as the gliding surface. It contains collagen fibrils and chondrocytes that are elongated along the axis parallel to the surface.

The middle or 'Transitional zone' consists of more rounded chondrocytes and randomly oriented collagen fibers. This zone plays a significant role in the shear properties of the articular cartilage.

The spherical chondrocytes and collagen fibers are oriented perpendicular to the joint surface in the 'Deep zone'.

The bottommost layer is the 'Zone of Calcified Cartilage'. It is a cartilaginous matrix that separates the subchondral bone from the hyaline cartilage [1].

Articular cartilage is permeable and allows water to flow through if a very high-pressure gradient is applied. Its low permeability helps in reducing the stress on the solid matrix when subjected to high loads. Cartilage is bi-phasic and viscoelastic and is always subjected to dynamic compressive loading. Upon compression,

cartilage stiffens in shear and becomes less dissipative. In tension, the superficial zone of cartilage is much stiffer than the middle zones.
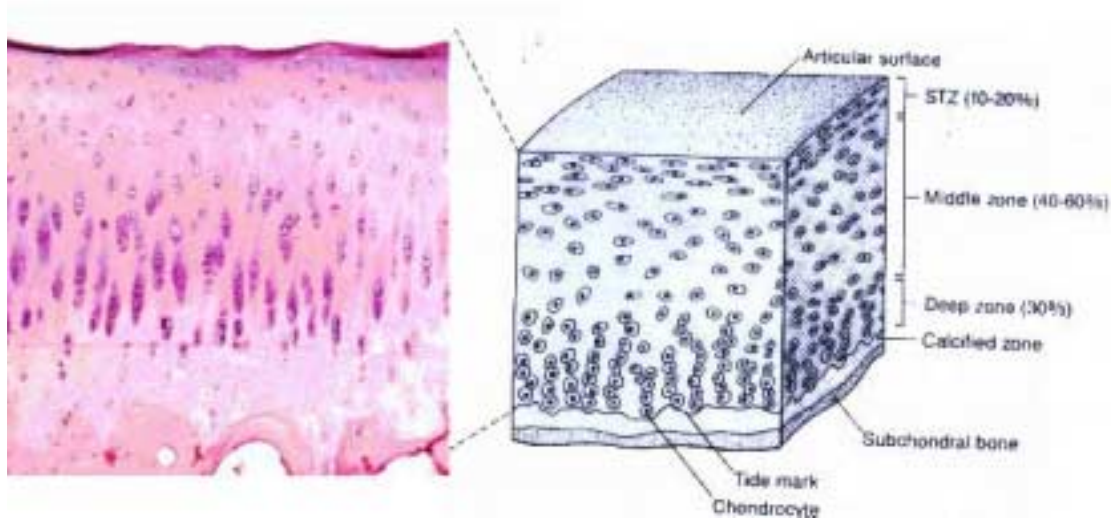
Figure 1 A histologic section of cartilage with a schematic of chondrocyte organization along its depth *(Dahweb.engr.ucdavis.edu/dahweb/126site/bone.htm).*

Articular Cartilage is composed of 65% to 80% water, 10% to 20% collagen and 5% proteoglycans. It contains a sparse population of chondrocytes (about 10%) that are responsible for its formation and maintenance. Articular cartilage has no blood vessels or neural network. Chondrocytes respond to environmental stimuli (like growth factors, mechanical loads etc.,) to maintain the tissue growth and regeneration.

1.2 Osteoarthritis:

Osteoarthritis is a disease in which the articular cartilage loses its ability to regenerate and wears down. Articular cartilage is subjected to repetitive loading due to movement of the joints. Erosion of cartilage causes higher friction in the joints causing severe pain and swelling. After the loss of the cartilage layer, Osteoblasts (bone building cells) try to make bone in the cavities created, causing small bone

spurs and roughening of the bone edges. This causes further pain in the joints (Figure 2).
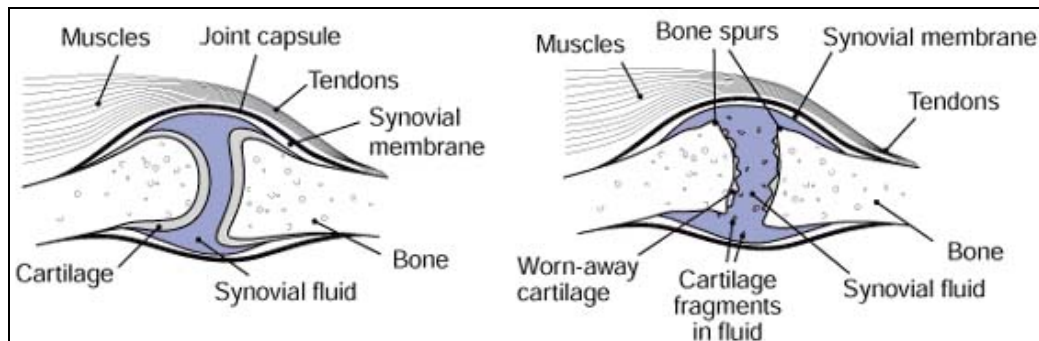
Figure 2 The picture in the left shows a good joint, the one in the right shows the diseased joint. *(http://www.niams.nih.gov/hi/topics/arthritis/oahandout.htm#1)*

Occurrence of this disease is correlated with the age of the subject [2]. With age, functional and mechanical properties of the cartilage deteriorate and hence regeneration rate of the worn-out cartilage decreases. Literature shows that articular cartilage degeneration depends on the behavior of the chondrocytes [3] and their inability to maintain the balance between cartilage formation and regeneration [4] [5]. The behavior of chondrocytes is affected by mechanical loading, Cell – Extra Cellular Matrix (ECM) interactions and chemical stimuli. In this work, we attempt to look at the effect of Cell – ECM interactions on cartilage phenotype and mechanical properties.

1.3 Tissue Engineering:

Tissue Engineering is the practice of implementing engineering and biological principles to devise new strategies for the regeneration of diseased tissues. Arthritis is one of the most common diseases among the people of the United States of America [6]. About 18.7% of people aged 18 or above have arthritic symptoms, out of which

57.8% are women. Osteoarthritis is a major arthritic disease. Hence the tissue engineering approaches to repair damaged articular cartilage are important.

In tissue engineering, living cells are cultured in 3D environment provided by a scaffold material. The cell – scaffold construct is introduced into the patients at the site of degenerated tissue to provide an outset to the regeneration process[7]. The surrounding tissues and cells present in the patient's body carry out further repair and regeneration. The main complication in cartilage tissue engineering is that its outcome is variable and in many cases fibrocartilage with inadequate mechanical properties is formed. It is thus necessary to study the mechanisms of the process in order to manipulate the repair process to our advantage.

1.4 Mechanical Loading Affects Biological Behavior Of Cells:

The fact that mechanical loading affects the biological behavior of cells is well known. It has been shown that when a dynamic pressure loading is applied on the periosteal explants, chondrogenesis is enhanced resulting in a faster repair of the cartilage [8]. The interaction of growth factors and chondrocytes has been shown to increase under a cyclic compressive loading [9]. A significant boost in the differentiation of mesenchymal cells due to presence of compressive loading has also been shown [10]. However, the mechanisms through which cells respond to the mechanical loading are still being figured out.

The mechanism through which cells respond to load is called mechanotransduction. The cell membrane contains special type of proteins called the "Receptors" that are capable of binding to some specific ligands (Figure 3). Some believe that the deformation due to loading causes bio-molecules in the cell to

physically come together and bind, thus starting a biological signaling cascade that results in a cellular response. It is also believed that the cells respond to mechanical stimuli by reorganization of their structure and by building up new proteins that help in tissue regeneration.
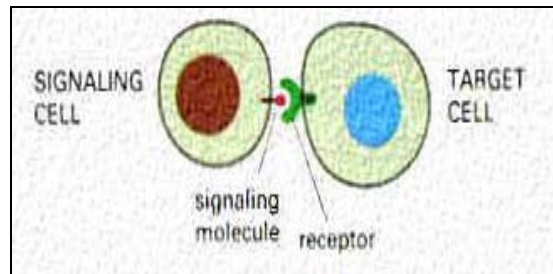
Figure 3 The figure shows the signaling process in two cells. The signaling molecule of the signaling cell binds to the receptor of the target cell to pass a signal [11]

Other theories implicate fluid flow that occurs in and around the cells as a result of loading [12]. Fluid flow has been shown to cause clustering of proteins in the cell membrane, which could result in a signaling cascade [13, 14]. Fluid flow can also affect bio-transport processes in and around the cell. The biological effect of loading can be cell division, differentiation (cells start to produce specialized proteins), inactivation or apoptosis (programmed cell death). Thus, studying cellular response to mechanical loading becomes critical as a first step to initiating strategies for maintaining tissue health or regenerating worn out tissue.

1.5 Cell – Extra Cellular Matrix (ECM) Interactions

Cells can sense and deform in response to external load on the tissue through proteins in the cell membrane called integrins, which connect cells to the extracellular tissue material [15], [16] (Figure 4)[16]. When a change occurs in the mechanical force distribution in the integrins, a deformation is caused in the structural elements of the cell, which may cause the receptors and ligands to bind together, passing a

signal from the cell exterior to its interior. Since the cells do not have a neural network, the Cell – ECM interactions play a major role in establishing the connection between interior and exterior of the cell.
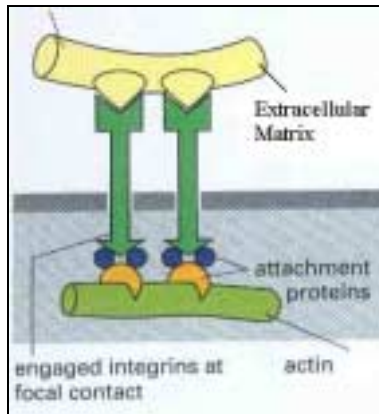
Figure 4 Trans Membrane molecules like Integrins, connect the ECM to the internal elements of the cell [11].

1.6 Concepts of Cell Biology:

Proper functioning of human body depends on an innumerable number of cells working together in an organized manner. Each human organ is a complex suite of tissues, which in turn are made up of cells. Cells are simple units of living matter – tiny compartments filled with aqueous solution of chemicals, which are bounded by a membrane.

It is well known that cells that populate tissues, like articular cartilage and bone produce the components of Extra Cellular Matrix (ECM) that surrounds the cells. Articular Cartilage is a porous, permeable, soft matrix that cushions the movement of bones. Bone is a composite material that provides structural integrity to our body. In both these tissues, the cells are embedded in the ECM and these cells generate the ECM.

The cells are responsible for the breakdown and turnover of worn out tissue. These processes are controlled by interplay of various factors like genetic factors, special biological signaling molecules like growth factors and mechanical loading. These cells are believed to sense and respond to mechanical stimuli through manufacture, arrangement and turnover of matrix proteins that make up the tissue[17].

Different cells like Epithelial cells, Connective tissue cells, Nervous tissue cells, Muscle cells, Blood cells, Germ cells, Sensory cells etc., exist to perform different functions. Various cell functions are achieved through the coordinated actions of many proteins. Under normal conditions cells either proliferate, differentiate or die.

Proliferation: This is a process in which the cells replicate through cell division. Nucleus of the cell divides into two, and finally two similar cells are produced.

Differentiation: This is a process in which the cells specialize to perform certain functions. For example, the cells that populate cartilage are called Chondrocytes and those that populate bone are called Osteons. Mesenchymal cells differentiate to form either chondrocytes or osteons.

Apoptosis: This is programmed cell death. In this process, the nucleus shrinks and the rest of the cell material fragments into small pieces that are disposed of by neighboring cells. Through this process, the cell number is maintained in the organism, thereby reducing the risk of excess cell growth.

1.7 Structural Elements of Cell:

Tissues of the human body are subjected to mechanical stresses during normal function. For example, arteries and veins, which carry blood to and from the heart, are subjected to shear stress due to the blood flow. Bone, a major structural component of the human body, is subjected to compression and bending. Articular cartilage is subjected primarily to compression [18]. The cells in these tissues act as mechanical sensors that react to the applied loading. The structural elements of a cell consist of a cell membrane and cytoskeletal architecture that consists of the actin filaments, intermediate filaments and Microtubules.

1.7.1 Cell Membrane:

The cell membrane is a bi-lipid layer that forms the outer wall of the cell. This helps in maintaining structure and integrity of the cell. The thickness of the cell membrane is about 5 nm [11]. The elastic modulus of the membrane in red blood cells (Figure 5) was found to be 0.1 MPa [19].
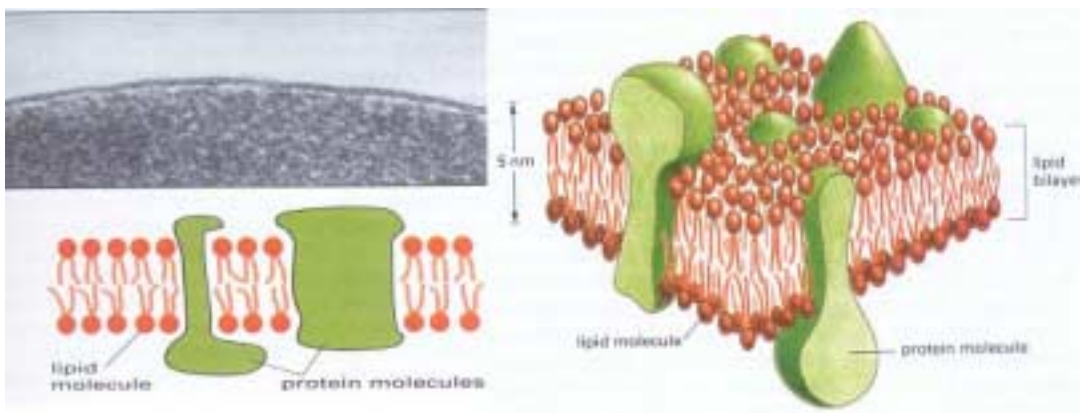
Figure 5 Cell membrane of human red blood cell (Schematic 2D and 3D views) [11]

All the cells consist of proteins that either bind the membrane to the cytoskeleton or ECM; or send signals to the interior of the cell in response to the

environmental stimuli. The cell membrane mainly takes up tensile loads, due to the presence of internal fluid pressure.

1.7.2 Cytoskeletal Elements:

Actin: The actin filaments are long, thin fibers, about 8 nm in diameter. They are attached to the plasma membrane and help in cell motility. As shown in the picture (Figure 6), they form bundles or extensive 2D and 3D network inside the cell. Gittes et al.,[20] found the flexural rigidity of actin to be 7.3E-26 Nm^2. They are flexible structures that are highly concentrated near the cell membrane. The actin fibers are primarily subjected to tension.
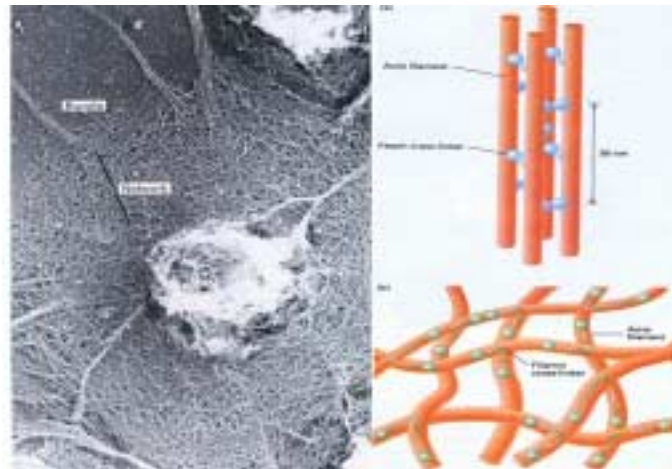
Figure 6 Actin Filaments form either an extensive network or bundles inside the cell.

Microtubules: Microtubules are hollow cylinder like structures with varying lengths, about 25nm in external diameter and 2.7nm thickness [11]. They are made of subunits called tubulin, which are arranged to form hollow tube like structures and are thought to resist compressive loading [21]. The flexural rigidity of microtubules was found out to be 2.1E-23 Nm^2 and hence the young's modulus was calculated to be 1.2 GPa. They are 1000 times stiffer in bending than actin.
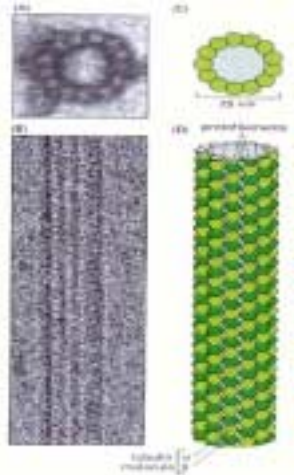
Figure 7 Electron Micrograph and Schematic of the cross section of Microtubules. 13 molecules called tubulins, combine to form a microtubule [11]

Intermediate Filaments: They are tough, stable fibers with diameters varying from 8 – 10 nm (Figure 8). The Intermediate filaments either stretch across the cytoplasm or form a network below the inner nuclear membrane. They can withstand high stresses and provide cells with mechanical strength. These filaments can endure larger stretching forces than actin or microtubules.
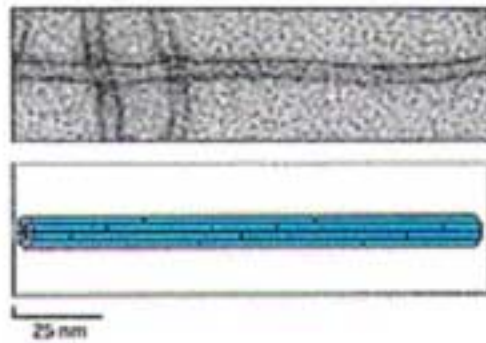
Figure 8 Intermediate Filaments [11]

1.8 Cell Mechanics:

Literature shows that many attempts were made to study cellular behavior under loading, subjecting the cells to either tension (by growing cells on a membrane and stretching the membrane) or shear (by flowing fluid over the cells [22]), but not

in compression. For chondrocytes compression is the primary loading due to repetitive motion of the joints.

A review of the literature suggests that intracellular fluid pressure and cytoskeletal architecture can both influence the compressive properties of cell. Osmotic studies suggest the presence of an intracellular pressure. In these studies, the volume of the cell changes in response to a change in solute concentration of the extracellular liquid. Smith et al. [23], have generated models of a yeast cell, which can be represented by a permeable, thin-walled, pressurized, liquid-filled sphere.

Studies by Ingber and others highlight the importance of pre-stress in cytoskeletal stability of a cell [24]. They have showed experimentally that microtubules bear compression, and actin filaments are in tension. Their model considers the cell as pre-stressed microtubules around which actin filaments are wound around like a cable. This model can predict many different cellular responses to load. However, it does not take into account fluid pressure of the cell. It has been showed by Guilak et al. [25] that variation in actin cytoskeleton has a strong influence on the visco elastic properties of the chondrocytes.

A common structural aspect of the cellular response is that they alter their cytoskeletal structure when subjected to loading. It is interesting to see if mechanical properties and/or cell response to load change when the cytoskeletal structure changes and can that change be correlated to biological phenotype.

Chapter 2 Rationale

It is believed that during normal development in chicken embryo, the joints are subjected to compressive loading through contraction of muscles [26]. Interestingly, if joint loading through muscle contraction is abolished by toxins, then malformation of the joint and cartilage occur. During development, extensive reorganization of cellular cytoskeleton is known to occur in the cells that eventually make cartilage [27]. Also, mature chondrocytes appear to have different mechanical properties compared to other cells from preliminary cellular experiments [17, 28, 29]. It is interesting to know if developing chondrocytes respond to external load via cytoskeletal reorganization and if this process is important in the formation of healthy cartilage.

Existing setups to study cellular behavior under loading, subject the cells to either tension (by growing cells on a membrane and stretching the membrane) or shear (by flowing fluid over the cells [22]), but not compression (important for chondrocytes). To address this problem, a simple procedure to study the behavior of ATDC5 cells (a prechondrogenic cell line) under compression has been devised. ATDC5 cells were embedded in 1% agarose gel and compressed to a total strain of 15% with a 10 min loading per day, for 3 days. These gel-cell constructs were cultured for 13 days and their behavior was analyzed. The cells manufacture special proteins called proteoglycans during chondrogensis. It was observed that the proteoglycan content in the gel-cell construct, increased with the days of culture in 3D environment (Figure 9). It was important to know if the proteoglycan content increased due to cell-cell contact or increase in protein release by each cell. A 'C'

program was developed to check for the cell clustering in cell-gel constructs (Appendix 1). It was observed that 3D culturing increased the proteoglycan content in the absence of cell-cell contact.
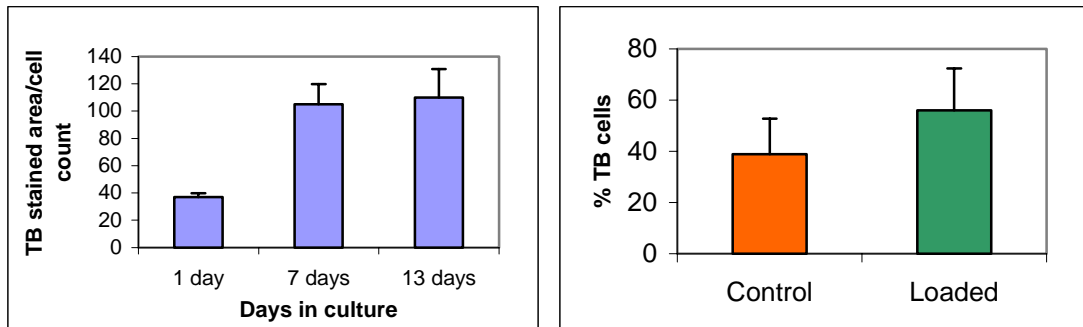
Figure 9 The gel-cell constructs were stained with Toluidine Blue (TB) to detect chondrogenesis. In the left chart, the TB stained area/cell count increased with the number of days in culture, showing an increase of chondrogenesis in ATDC5 cells due to culture in 3D. The chart in the right shows an increase in cell count/unit area due to mechanical loading in the gel-cell constructs.

Also, pre-chondrocytes were cultured in 3D and their actin cytoskeletons were observed on different days of culture from Days 1 to 12 (Refer to Appendix 6).
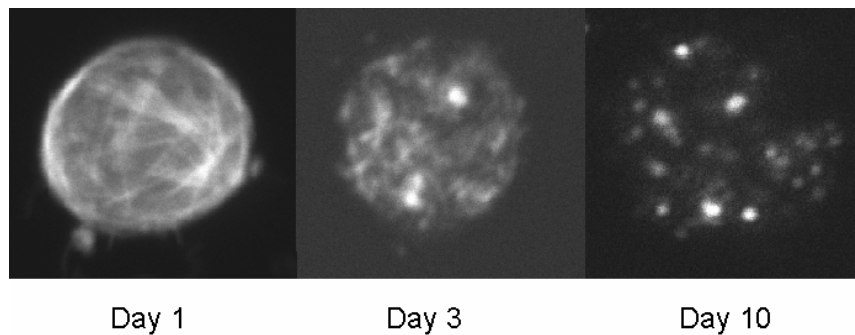
Figure 10 The cells were stained for actin and images of the largest cell cross-section were obtained on Day 1, Day 3 and Day 10 of the 3D culture period using a confocal microscope. The pictures show a reorganization of the actin filaments in the cells from Day 1 (stress-fiber appearance) to Day 10 (punctate appearance).

A clear reorganization of the actin filaments with the days in 3D culture can be observed (Figure 10). We hypothesize that differentiation state of chondrocytes can be correlated with cytoskeletal structure, and hence a measurable difference in the response of the cell to mechanical load. We propose a combination of experimental and Finite Element modeling approaches.

Chapter 3   Preliminary Experimental Approach

3.1 Introduction:

As a first step in determining a measurable property of the cell, a system was developed to measure the stiffness of the cell. In this approach, the cells were embedded in a gel and the gel-cell constructs were compressed to 25% strain, to obtain deformations of the cell. These deformations were then curve-fit using a standard elasticity solution to obtain the Young's modulus of the cell.

3.2 Materials and Methods:

Elasticity solution for a spherical inclusion inside an infinite medium was developed by Goodier [30] (Refer to Appendix 2). If the properties of the spherical inclusion (E_t, υ_t), properties of the medium (E_b, υ_b) and the load applied (T N/mm^2) are known; the radial and tangential deformations at any point inside the medium can be calculated using the following set of equations.

$$U_r^b = -\frac{A}{r^2} - \frac{3B}{r^4} + \left(\frac{5-4\upsilon_b}{1-2\upsilon_b}\right)\frac{2C}{3r^3} + \left[\frac{-9B}{r^4} + \left(\frac{5-4\upsilon_b}{1-2\upsilon_b}\right)\frac{C}{r^2}\right]\cos 2\theta \; + \frac{Tr}{2E}[(1-\upsilon_b)+(1+\upsilon_b)\cos 2\theta]$$

$$U_\theta^b = -\left[\frac{6B}{r^4} + \frac{2C}{r^2}\right]\sin(2\theta) - \frac{Tr}{2E}(1+\upsilon_b)\sin 2\theta$$

Where A,B,C are functions of mechanical properties of inclusion and medium. Similarly, there are separate sets of equations to calculate the radial and tangential deformations at any point inside the inclusion (Refer to Appendix 2).

If we consider spherical inclusion to be the cell and medium to be the gel, the radial deformation (U_r) can be calculated in terms of modulus of elasticity and Poisson's ratio of cell and gel; the applied stress; the radius of the cell and the angle

w.r.t loading axis. All of these except stiffness of the cell can be determined from separate experiments. We can thus obtain the stiffness of the cell through curve-fitting techniques.

3.2.1 Gel Properties:

Initially, a gel with appropriate properties had to be determined. Two standard gels for embedding, agarose and alginate, were tested for this purpose. A procedure was developed to prepare 1.2%, 2% and 3% agarose/alginate gels and compress them in a custom-devised loading apparatus (Refer to Appendix 3). The gels were compressed to a total of 25% strain in four loading stages using a stepper motor. Agarose and alginate are visco-elastic materials and hence each loading stage was held for 10 min to allow the gels to stress relax and equilibrate. Stress relaxation was observed in both agarose and alginate, as expected. The data for deformations and applied loads for each successive loading were acquired using Labtech Software (*LABTECH NOTEBOOK™, MA*). Load–Time and Displacement–Time curves were obtained using Microsoft Excel (Figure 11). Two different users carried out the testing, to check for repeatability and user dependency of the results.
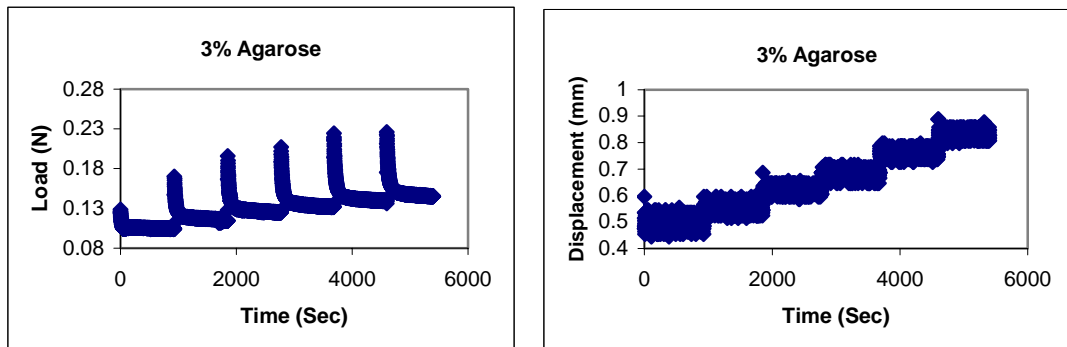
Figure 11 The left and right graphs show the Load–Time and Displacement–Time variation for 3mm cubes of 3% Agarose. A visco-elastic behavior of the agarose can be observed here.

The applied loads and displacements obtained from such curves were used to calculate the Stress–Strain values, from which modulus of the gel could be obtained.

$$Stress = \frac{Applied.Load}{Area.of.the.Loaded.face.of.Gel}\left(\frac{N}{mm^2}\right)$$

$$Strain = \frac{Change.in.Length}{Original.Length.of.the.Gel}$$

$$Stiffness = \frac{Stress}{Strain}\left(\frac{N}{mm^2}\right)$$

The Stress–Strain graph obtained for the above case of 3% agarose gel is as shown in Figure 12.
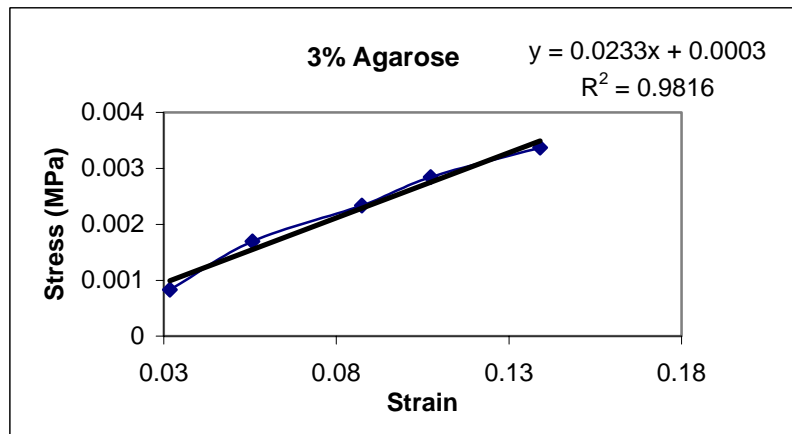
Figure 12 The Stress–Strain curve for a 3mm 3% agarose gel. A linear trend line is added. The stiffness of the gel obtained is 23.3 Kpa, R^2 value, which indicates goodness of fit (0 – bad fit, 1 – perfect fit) is 0.9816

There was a wide variation in the stiffness values obtained from the experiments. This may be due to the variation of experimental procedures, or user dependency of the experiments. A 10secs refrigeration of the gel, before the compression, caused 50% variation in the gel stiffness compared to gelling at room temperature. Our results lie in the range of values obtained from the literature (Figure 13) [5, 31, 32]. Stiffness of the gels increased with gel concentration and stiffness values were repeatable implying that our experimental set-up is precise. The standard

deviation values were very high for the 3mm cubic gels (6-30), but the discs had low standard deviation values (<4). The experimentation procedure shows that the 3 mm cubes were cut to their original dimensions, this could have affect on the gel properties (like permeability) to a certain amount.
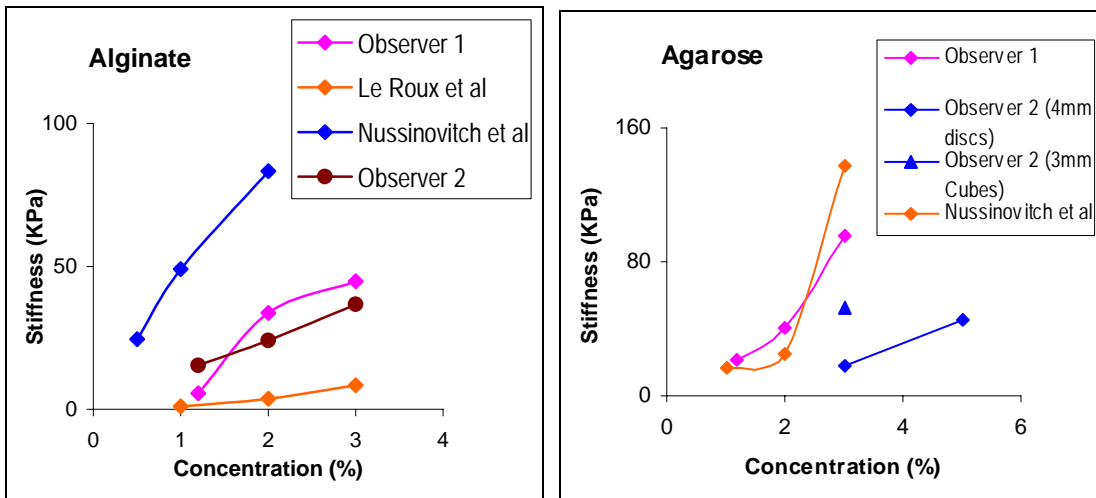
Figure 13 The stiffness values obtained for alginate and agarose gels. As can be observed, there is a wide variation in the stiffness values obtained, though the values lie in the range of other researchers.

Accuracy of the determined cell properties depends on the precision of gel properties. From the experiments we understand that, the gel properties in such experiments cannot be obtained from the literature. The gel properties highly vary and hence every gel needs to be tested for its properties to obtain exact cell properties.

3.2.2 Applied Stress:

A stepper motor was used to apply load on the gel. Four to five loading steps are applied in succession until the gel is compressed to 25% strain. A small pre-load (0.01N) was applied to all the gels. The face area of the gel was calculated from the gel dimensions (refer to Appendix 3). The load readings were obtained using Labtech Software (Data acquisition card – metrabyte–16A, *LABTECH NOTEBOOK™, MA*) and the applied load was calculated by subtracting the pre-load from the current load

reading. This applied load when divided by the area of the face of the cube, which takes up the load, gives the applied stress on the gel. This value is equivalent to the applied stress T in Goodier's elasticity solution (Refer To Appendix 2).

3.2.3 Radial Deformation of Cell:

Other researchers in our laboratory performed gel-cell compression experiments. Gel-cell constructs for the compression experiments were prepared using a standard procedure. The cells were stained using CMFDA stain (this stains the whole cell in fluorescent green, Excitation wavelength – 492nm, Emission Wavelength – 517nm) and placed in a 0.5mm cubic gel. Largest cross sectional images of the undeformed and deformed cells were obtained under the confocal microscope. The images obtained were post processed. The x-axis is the loading axis. The black lines show the limits of thresholded images (Figure 14).
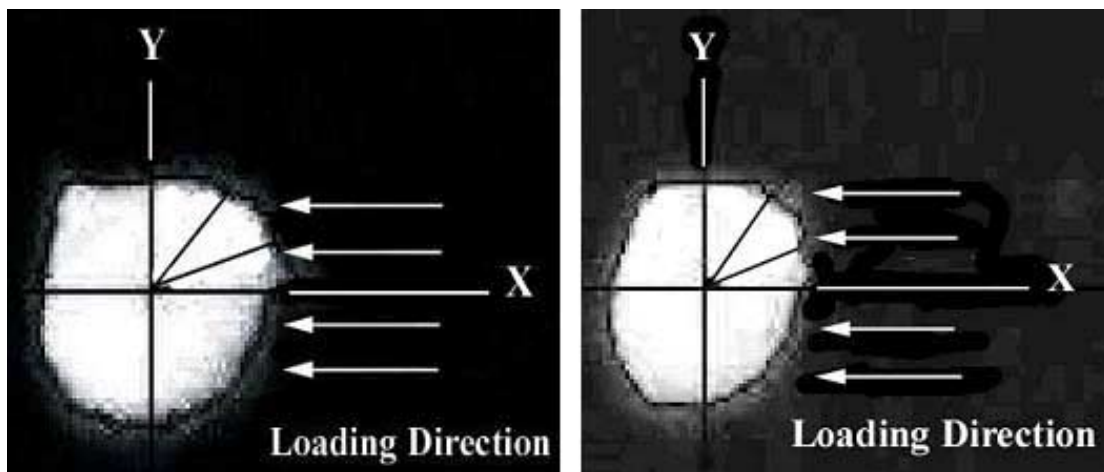
Figure 14 The stained cell in undeformed and deformed states. We can see that the size of the cell reduced in the x-direction when a compressive load is applied.

Radius of the cell is calculated at different angles (.) w.r.t loading axis, at these cross sections, for both compressed and uncompressed cells. Difference in radii of the uncompressed and compressed cells (U_r mm) is then plotted against the angle w.r.t to the loading axis (X axis). The cell became thinner in the direction of loading (where

18

U_r is positive) and longer in the direction perpendicular to loading (where U_r is negative).
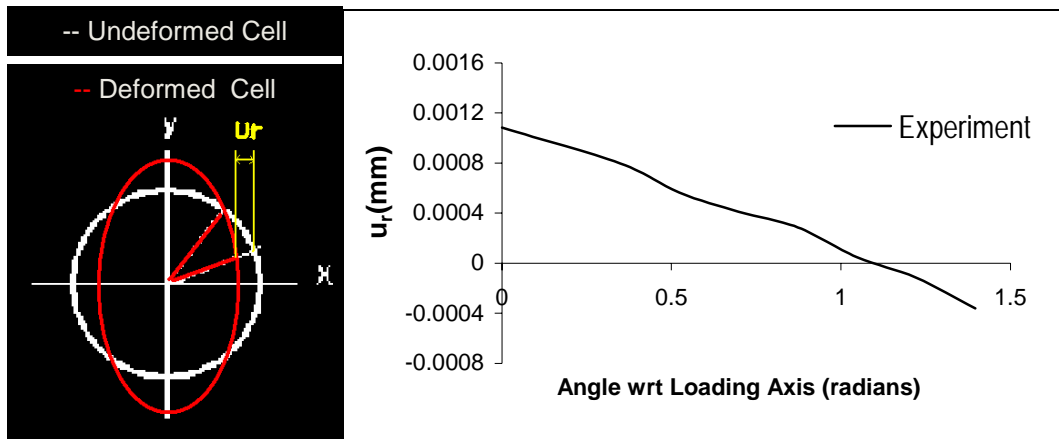
Figure 15 The picture shows the variation of Ur with angle (θ). We can see that Ur is positive in the direction of loading and negative in the direction perpendicular to loading.

3.3 Stiffness of Cell:

Goodier's elasticity solution provides theoretical values that could be predicted for a case of a cell in gel. From the above experiments, the material properties of the gel, applied stress and radial deformations of the cell could be calculated. Hence in Goodier's equations for the radial deformations of the cell at its outer edge, the only unknowns would be the material properties of the cell (Refer to Appendix 2). Cell was considered to be incompressible (v_t = 0.5). Thus stiffness of the cell could be calculated by curve fitting the experimental values with the theoretical values from Goodier's closed form solution. A maple code was written to calculate the radial deformations of the cell in terms of stiffness of the cell (E_t) using Goodier's equations (Refer to Appendix 4). This equation was entered into Excel and curve-fit with the experimental values. As we can see in the following case, the best fit was obtained at E_t = 390 Pa. This value lies in the range of the estimates of cell modulus performed by other groups.
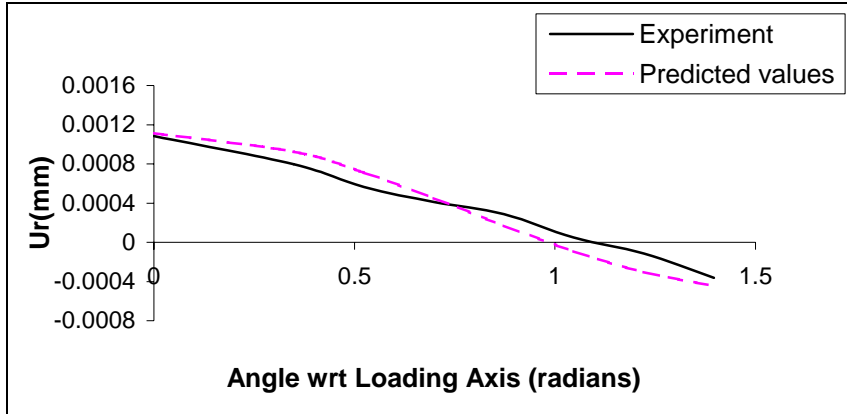
Figure 16 The graph shows the best fit of theoretical values with those of the experimental values. This fit was obtained at a cell stiffness value (Et) of 390 Pa.

3.4 Parametric Analysis:

We can clearly see that the value of cell stiffness (E_t) obtained, depends on the stiffness of the gel (E_b) and the applied stress (T). A parametric analysis was done to check the dependence of U_r on E_b, T and Et. In all the cases, $v_t = v_b = 0.499$ and radius of the cell r = 0.004764 mm.

3.4.1 Variation of U_r with E_b (Stiffness of the gel):

For determining the effect of E_b on U_r, the following values were substituted in the Goodier's equations.

E_t = 0.000391 MPa and

T = 0.0072 N/mm^2

Now U_r is a function of E_b and angle θ and thus U_r was calculated for different values of E_b ranging from 26.6 Kpa to 79.8 Kpa and θ values ranging from 0 to 1.57 radians (0° to 90°). From Figure 17 it can be observed that even a small variation in the stiffness of gel (E_b) causes a large deviation in the radial deformation.
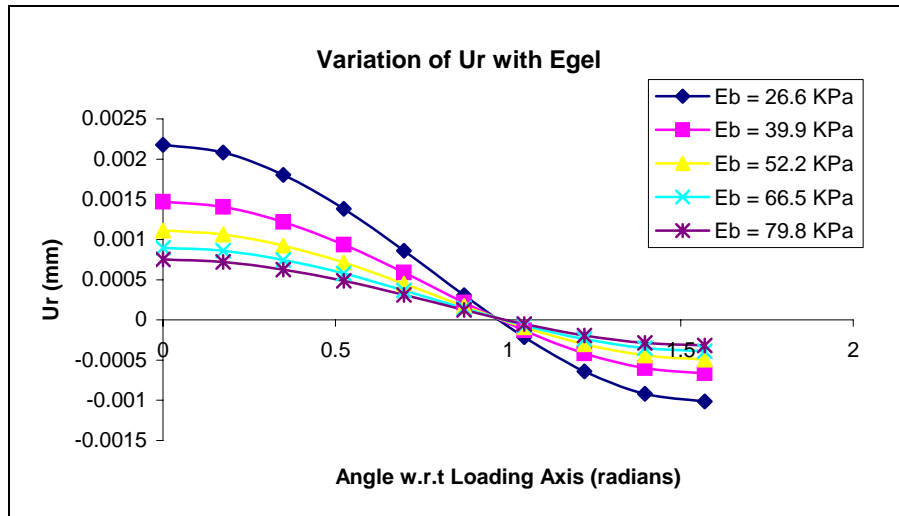
Figure 17 The variation of Ur for different values of Eb. It can be clearly observed that even a slight variation in Eb causes a huge change in the radial deformation

3.4.2 Variation of U_r with T (Applied Stress):

The applied load affects deformation of the cell-gel construct, but the important issue that needs to be observed here is the sensitivity. This analysis predicts the variation of U_r induced with a small deviation in the applied load. For this purpose, U_r was calculated for T values ranging from $0.0048 N/mm^2$ to $0.0144 N/mm^2$ by keeping the values $E_b = 0.000391$ MPa and $E_t = 0.07980$ MPa constant.
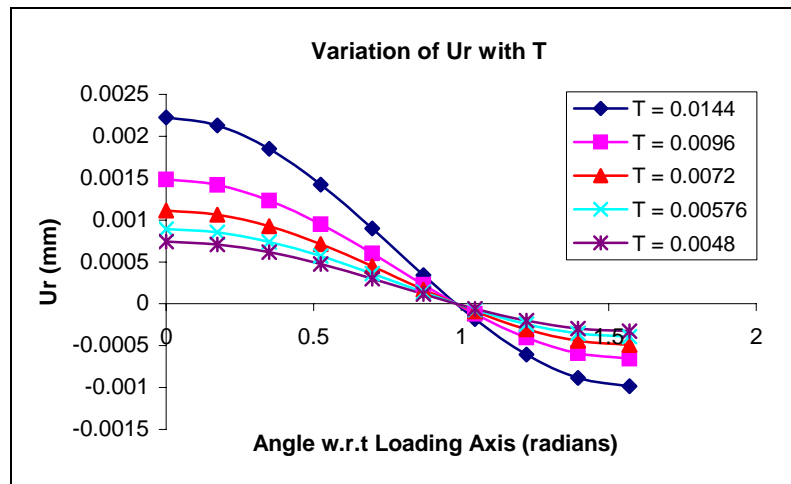
Figure 18 Variation of Ur with applied load T. It can be observed that the Ur obtained is highly sensitive to T.

3.4.3 Variation of U_r with E_t (Stiffness of Cell):

Similarly, the variation of U_r with E_t was observed by keeping E_b and T constant at 0.07980 MPa and 0.0072 N/mm^2 respectively. The values of E_t were varied from 3.91Pa to 3910 Pa and U_r was calculated for each value using Goodier's solution. It was observed that even a large variation of stiffness of the cell (E_t) produced small differences in U_r. Cell and gel with equal moduli values, may give better results, but too low concentrations of gels, cause them to break down.
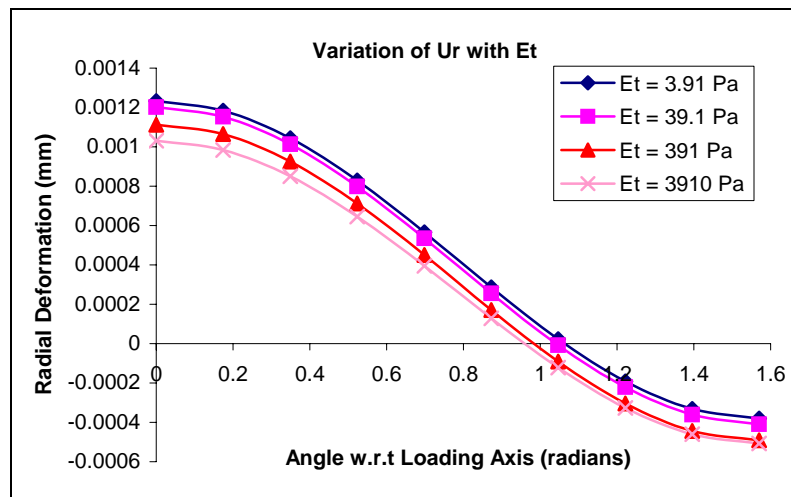
Figure 19 The graph shows variation of Ur with stiffness of cell (Et). It can be observed that even a 1000 fold increase in Et, causes only a minor variation in Ur values.

3.4.4 Discussion:

As mentioned earlier, the cell stiffness value (E_t) can be derived by curve fitting. The U_r values derived from the closed form solution play a pivotal role in determining cell stiffness. But with the parametric analysis, we have deduced that U_r is highly sensitive to the Stiffness of the gel and applied load, and hence even a slight error in the readings could lead to erroneous results. In order to overcome this problem, we decided to reform the experimental procedure by using a more sensitive load cell.

3.5 Micro-gel Experiments – Limitations:

A more sensitive load cell (0-2gm) was imparted in the improvised version of the experimental set-up. A smaller gel (0.5μm cube) was intended to increase the sensitivity of the system, and also help in obtaining better images under the confocal microscope. The gel formation and experimentation procedures are listed in Appendix 5. Initial attempts on this set-up were to calculate the stiffness of 3% agarose gel, but this procedure faced a lot of hurdles.

Water: The gel dries up under normal conditions and hence it was necessary to maintain a certain amount of water to keep it moist. During every load step, about 40μl of distilled water was placed on the gel. This addition of water caused a drift in the original loadings due to sensitivity of the load cell.

Light: To measure the tiny dimensions of the gel, a micropipette was used along with two micrometers. This measurement was difficult to be obtained with naked eye and hence the whole experimental procedure was carried out under a microscope. A light was required through out this operation, which produced a lot of heat. This heat from light caused some drift in load readings.

Air: In order to drive away the heat produced by the light, a small fan was introduced in the set-up. Introduction of the fan nearly nullified the effect of the light, but it was not completely helpful. Different fans and speed have been tried for this purpose.

Noise: The sensitivity of the load cell caused it to catch even the smallest vibrations in the equipment room. Various experiments were carried out to find the place in our laboratory with the least vibrations.

The testing conditions and results for the above factors are listed below:

For Air:
Initial Load cell reading is noted for 60 sec
Air is blown on to the Loading pin and load cell reading is noted upto 440 sec
Air is shut down and load cell reading is noted for 500 sec
For Water:
Load cell reading was noted for 60 sec
Water is added on the slide and reading is noted for 940 sec
Graph is plotted between Load and time
For Light:
Load cell reading is noted for 60 sec
Light is switched ON and Load cell reading is noted for 1000 sec
Graph is plotted between load and time
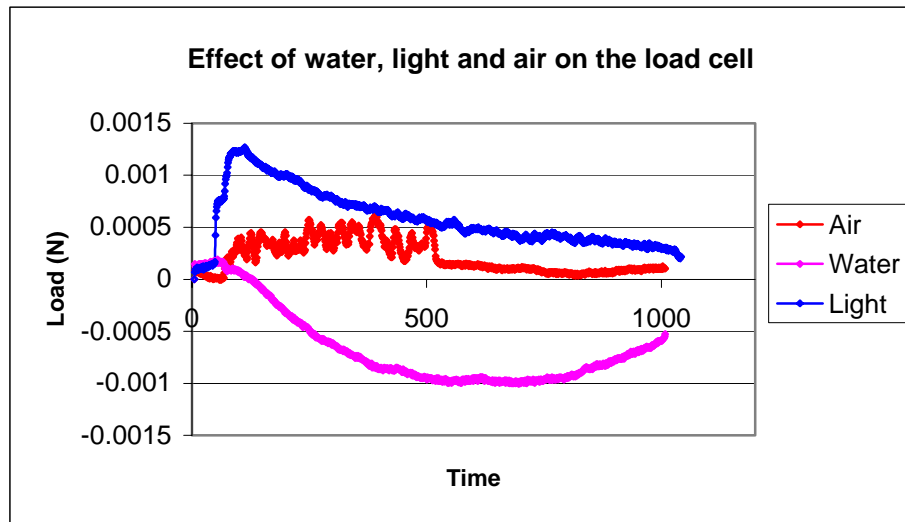
Figure 20 The chart shows the drift caused in load readings due to the presence of water, air and light on the compression apparatus.

The drift caused in load readings due to these factors was considerable. In all the experiments carried out on this apparatus, maximum load applied on the gel using the load cell was not more than 0.002 N. It can be clearly observed from Figure 20 that the drift caused due to these factors is about 0.001 N, hence estimated error in the load cell readings would be 50%, which is extremely high. Due to these limitations, this experimental approach was not pursued further at this point. We propose a new approach for quantifying the effect of structural variation of cytoskeletal organization on the cells' ability to resist mechanical loading.

Chapter 4　　Revised Approach

4.1 Introduction:

We have seen earlier that there was a remarkable difference in the cell cytoskeletal architecture over the days in 3D culture. Objective of this new approach was to define a simpler procedure for finding a response of the cell to loading that would be measurably different between cells with different cytoskeletal architecture. In this approach, the cells were placed in 3D culture for a period of 12 days and the structure of their actin filaments was observed on days 1, 5, 8 and 12 in 3D culture. Cross-sectional images of the cell were obtained using fluorescent microscopy and these images were further utilized to build a Finite Element model of the cell.

4.2 Materials and Methods:

Images of cell cross-sections were obtained at different days of culture. For this purpose, a standard cell staining protocol was followed. 5µl (at a concentration of 10^6cells/ml) of ATDC5 cells were added to 1ml of 2% agarose solution. This mixture was then injected into a custom-made mold to make 8 gels of size 1.2mm x 5mm x 0.5mm (width x Length x thickness) each. The gel-cell constructs were placed in 2 well slides and culture media was added (a total of 8 gels were placed in four 2 well slides). These slides were then placed in an incubator at 37^oC and 5% CO_2 (*SHEL.LAB, Sheldon Manufacturing Inc, Cornelius, Oregon*), to maintain proper pH, humidity and temperature required for cell growth. The culture media was changed every 2 days to replenish the nutrients.

One slide was removed on each of days 1, 5, 8 and 12 and the gel-cell constructs were stained for actin using a standardized protocol (Appendix 6). These stained cells were then observed under the confocal microscope for fluorescent imaging. The cell was observed at 400x magnification and zoomed to 6.7 times (this zoom factor produced the best image of the cell) its original size. The pinhole diameter was maintained between 2μ to 3μ and the images of the largest cross-section of the cell were obtained at 488nm wavelength. Figure 21 shows the variation of the actin structure over days 1 to 12 in 3D culture.
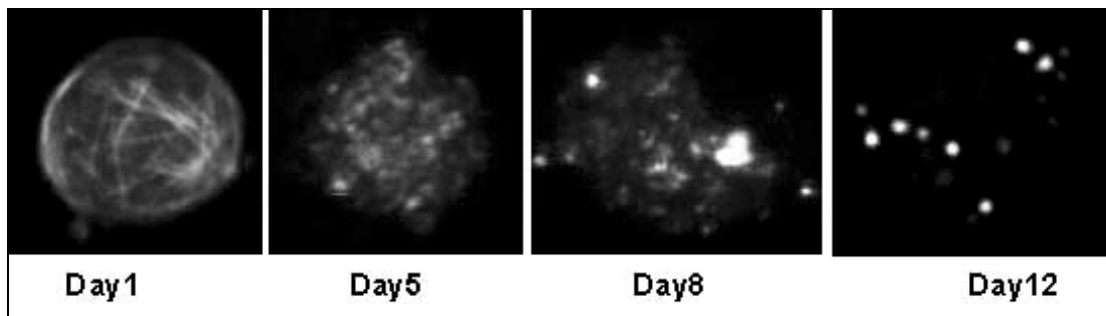
Figure 21 The pictures show a reorganization of the actin fibers in ATDC5 cells from Day1 to Day12

When ATDC5 cells were cultured on a plate (2D) the actin cytoskeleton had a stress fiber like appearance. Under such conditions, the cells do not get stimulated by mechanical loading to make proteoglycans. However, when they were cultured in agarose gel and mechanically stimulated to produce proteoglycans, we notice a marked redistribution (from fiber to rounded) of the actin cytoskeletal structure over time. It is interesting to see if the reorganization of the cytoskeleton is integral to the process of making cartilage. It makes intuitive sense, that a properly arranged cytoskeleton would be able to properly interpret the mechanical signals that control the biological process of making cartilage.

4.3 Post processing of Cell Cross-sectional Images:

OPTIMAS software was used to post process the cell images. Actin cytoskeleton can be represented in the form of lines (in case of Day1) or areas (in case of Day 12). The end coordinates of the lines, or the end coordinates of the bounding region of interest of the areas can be obtained from OPTIMAS in Microsoft excel format.
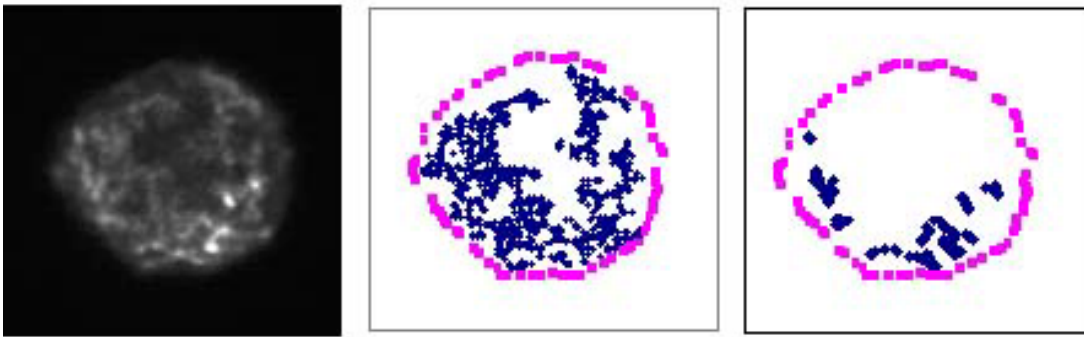
Figure 22 The cell cross-section (left) and the points representing the cell image obtained from OPTIMAS (center) are shown. The right picture shows the points collected from OPTIMAS at a high threshold value (170), it can be observed that the threshold plays an important role in the reconstruction of cell image.

The points obtained from OPTIMAS are highly dependant on the threshold applied. At a higher threshold, less number of points are obtained, similarly at a lower threshold higher number of points are obtained. In Figure 22, the right image shows the points obtained at one single threshold value (170, threshold 0 – Black, threshold 255 – White), it can be observed that these points are not an accurate representation of the cell cross-sectional image. Hence determining the exact threshold value is an important aspect in the reconstruction process. In order to avoid the problems of threshold, the points of the image at various threshold values were obtained and combined. The center picture in Figure 22 shows the combined points obtained at

different threshold values. This captures many of the key features of the cytoskeletal organization.

The point coordinates were obtained in microns, by appropriately calibrating the scale of the image. In OPTIMAS, the image screen width is 28cm. When converted to micrometers for 400x magnification, the screen width obtained is 232.2μm. Since the zoom factor used is 6.7, actual screen width would be 232.2/6.7 i.e, 34.65μm. These points were then combined to reconstruct the cell model with its embedded actin filaments. The data collection sets in OPTIMAS provide the points representing the cell cross-sectional image in Microsoft excel format.

4.4 Composite Model:

As we can see from the cell cross-sectional images, the actin filaments are either arranged as fibers (Day 1) or bundled up (Day 12) in the cell. They closely resemble a composite matrix with its embedded fibers. Hence for our analysis, the cell was considered to be a composite material with its cytoskeletal elements as the fibers embedded in the cell. Thus properties of the cell depend on the properties of fibers (actin) and cell matrix (all other components of the cell)

$$E = E_f V_f + E_m V_m$$

Where E is the Elastic modulus of the whole cell

E_m, E_f are the Elastic moduli of matrix and fibers respectively;

V_m, V_f are the Volume fractions of matrix and fibers respectively.

We know that the elastic modulus of the composite depends on the orientation of its fibers. In the above case, E will be maximum if all the fibers are oriented in the longitudinal direction, and minimum when all the fibers are oriented in the transverse

direction. Thus in this model, the stiffness, volume fraction and orientation of the actin fibers play an important role in determining the cell properties.

4.4.1 Embedded Element Concept:

In the process of creating 2D Finite element model of a cell, the cell was considered to be a continuum element with embedded truss elements (actin). The Embedded element concept of ABAQUS 6.2 was chosen for modeling purposes. Actin fibers were considered as Truss/2D plane Stress Elements embedded in a Cell that is the host element. For 2D modeling, the types of embedded elements-in-host elements provided by ABAQUS are solid-in-solid and Truss-in-Solid. Figure 23 shows a representation of a truss element embedded in a plane stress element.
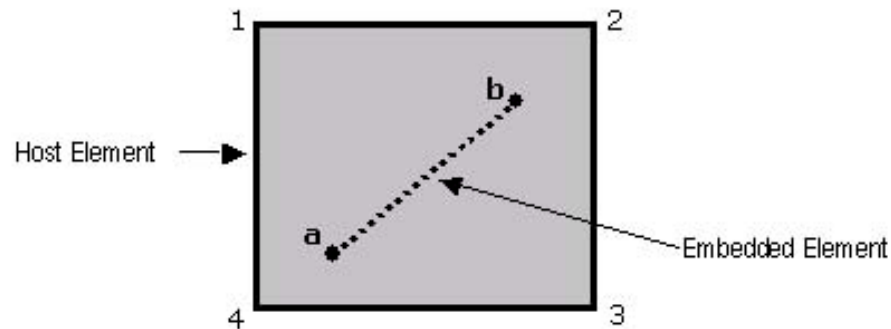
Figure 23 The picture is a representation of the embedded element concept in ABAQUS. 'ab' is the element embedded in host '1234'.

As shown, 'ab' is the embedded element, and '1234' is the host element. The nodes of the embedded element will have the Boundary conditions of the nearest node in the host element. In this case, all the BCs on node 4 of the plane stress elements will be applied to the node 'a' of the truss element. And similarly, the node 'b' will have the BCs of node 2. A small tolerance (0.01μm) can be specified for the embedded node, if it does not lie in the host element. Validation of this concept was the next step, before going into further Finite Element Modeling.

4.4.2 Validation:

Plane Stress Element:

A square 10x10 plane stress element was modeled in ABAQUS with the shown boundary conditions (See Table 1). One edge of the element was fully constrained (6 DOF) and a uniform pressure load of 0.2 N/mm^2 was applied on the other edge. Poisson's ratio of the element was 0.49.

Applied $\sigma = 0.2$ N / mm^2

$v_C = 0.49$

Length of plate = 10 mm

Width of plate = 10 mm

Initially the left edge was constrained and load was applied in the longitudinal direction. The stiffness obtained was 1.3115 N/mm^2. Then, the bottom edge of the element was constrained and the same load was applied in the transverse direction. The stiffness obtained in both the cases was equal.

One Truss Element embedded in the Continuum:

Then, a truss element was embedded into the 10x10 square plane stress element. The stiffness of the truss element is 1000 times more than that of the host element (actin properties are in the order of MPa, where as the gel properties in KPa), the cross-sectional area of the truss being 0.01 mm^2. Poisson's ratio remains the same for both [33] [34].

$\sigma = 0.2$ N / mm^2
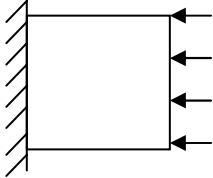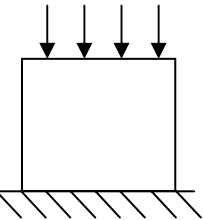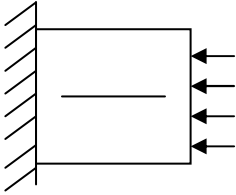
$v_T = 0.49$

$A_T = 0.01$ mm^2

$E_T = 1200 \text{ N} / \text{mm}^2$

$E_C = 1.2 \text{ N} / \text{mm}^2$

In the first case, load was applied in the longitudinal direction and E1 obtained was 1.5516 MPa. In the second case, load was applied in the transverse direction and E2 was 1.392 MPa. Table 1 shows that the set-up is stiffer in the longitudinal direction than in the transverse direction, as expected. Hence we can say that this method is valid to be used.

Load	ΔL	$\varepsilon = \Delta L / L$	$E = \sigma / \varepsilon$
	1.525	0.1525	1.3115
	1.525	0.1525	1.3115
	1.289	0.1289	1.5516
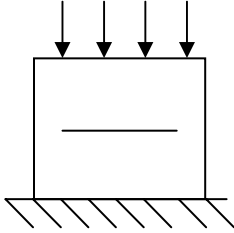
	1.437	0.1437	1.392

Table 1 Validation of Embedded element concept. Initially, in absence of embedded elements, the stiffness remains constant. When a truss element was embedded, the stiffness in longitudinal direction was greater than that in transverse direction as expected.

4.5 FE models – Isotropic and 0° Fibers:

Points on the cell outline and actin fiber end coordinates were obtained using OPTIMAS, from confocal images. A finite element model was generated using such images. A FORTRAN Code (Appendix 7) was written to create a program file (macro) to display the outline coordinates of the cell on I-DEAS screen. Then in I-DEAS, these points were used to create cell surface in 2D. A 500μ x 500μ gel (plane stress) was created around the cell. The gel and cell were then meshed using 2D plane stress (CPS4) elements. This model was then exported to ABAQUS.

Another FORTRAN Code (Appendix 8) was written to create an ABAQUS input file containing the embedded actin filaments. This program converts actin end point coordinates (from OPTIMAS) data to Finite Element format. The end coordinates were inserted as nodes and the fibers as 2D Truss (T3D2) elements. The two input files (one from I-DEAS and another from FORTRAN) were combined to give the final ABAQUS input file. This file generates the complete model with gel, cell and embedded actin fibers.

In order to observe the behavioral trend of actin fibers under compressive loading, two gel-cell models were created (See Figure 24).
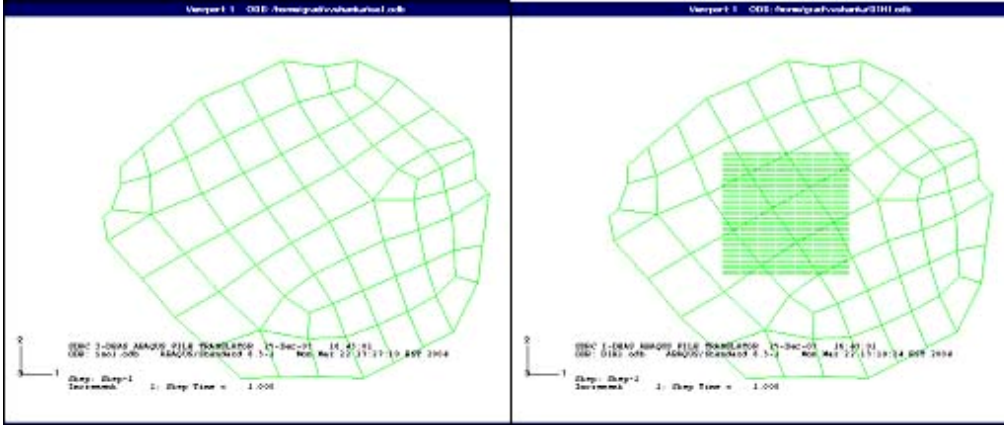
Figure 24 The image on the left shows the isotropic model and that on the right has embedded actin fibers in horizontal direction.

In one model, cell was considered as an isotropic continuum, without any embedded fibers (E_{cell} = 390 Pa), and in the other model, all the actin filaments were assumed to be in the horizontal direction (the total length of the filaments and the number of filaments were measured from the cell images, using OPTIMAS; E_{cell} = 385 Pa, E_{actin} = 363 MPa). A unixial compressive loading was applied on the gel-cell constructs in the global X-direction, to observe the radial deformations of the cell in both cases. As expected the deformations of the gel-cell construct with embedded horizontal fibers were lower than those with no fibers.

4.6 Curve Fitting an Ellipse:

In reality the cell outlines are not always smooth and round. One of the cell outlines, as obtained from OPTIMAS, can be observed in Figure 25. In order to nullify the effects of any discrepancies in the cell outline on the radial deformations obtained, an ellipse was fit to the cell outline. A MATLAB program was generated for this purpose. For any given set of points the program tries to fit a curve

$$y = \overset{+}{-} \left\{ \frac{b}{a} \sqrt{a^2 - (x - x_i)^2} + y_i \right\}$$

33

Where (x_i, y_i) are the coordinates of the centroid. From this equation, x values can be obtained from OPTIMAS and the corresponding y values can be calculated. The best ellipse is fitted by varying the values of 'a' and 'b' which are the semi major and minor axes of the ellipse respectively.
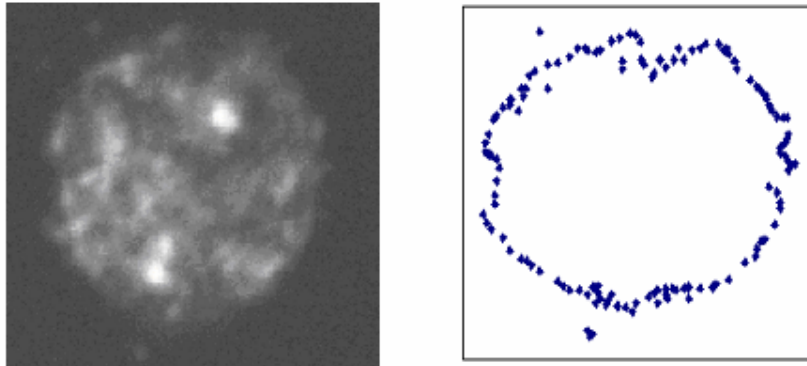
Figure 25 The pictures show the confocal image of the cell cross section, and the outline points obtained from OPTIMAS.

4.6.1 Centroid:

The ellipse fit depends on the centroid (x_i, y_i) of the cell. As discussed earlier, the cell cross-section is irregular. Hence a FORTRAN code was written to compute the centroid of the cell cross-section. Area of the cell was divided into small triangles and the centroid is calculated as the weighted sum of the centroids of individual triangular areas. The following algorithm shows the procedure for calculating the centroid (Appendix 9):

1. Enter one vertex (x_1, y_1) (=(0,0) say).

2. Take two consecutive points (out of a total of n points) on the cell outline. Read them as (x_2, y_2) and (x_3, y_3).

3. Calculate the centroid (x_{c1}, y_{c1}) of the triangle formed by (x_1, y_1), (x_2, y_2) and (x_3, y_3)

as $x_{c1} = \dfrac{x_1 + x_2 + x_3}{3}$ and $y_{c1} = \dfrac{y_1 + y_2 + y_3}{3}$

4. Area of the triangle can be calculated as

$$a1 = \frac{[(x_2 - x_1) * (y_3 - y_1)] - [(x_3 - x_1) * (y_2 - y_1)]}{2}$$

5. Calculate the area (a_2) and centroid (x_{c2}, y_{c2}) for the next consecutive set of outline points. Repeat for all the outline points using (x_1, y_1) as a common vertex for all sets of points.

6. Calculate the sum of areas $a = a_1 + a_2 + ... + a_n$

7. The Centroid of the irregular cell cross-section can be obtained as,

$$x_c = \frac{1}{a}\left(\sum_{i=1}^{n} a_i x_i\right) \text{ and } y_c = \frac{1}{a}\left(\sum_{i=1}^{n} a_i y_i\right)$$

Once the centroid is calculated, it can be entered into the MATLAB code to obtain the best ellipse fit.

4.6.2 Ellipse:

A MATLAB code was generated to curve-fit an ellipse to the cell outline. The coordinates of deformed/undeformed cell are input to the MATLAB program to give 'a' and 'b' values for the best ellipse fit. The sequential algorithm for the code is as follows:

1. There are 2 unknown parameters 'a' and 'b', that are to be obtained from the equation, so a set of two random variables, which can be changed to obtain the best fit, is created.

2. The x and y coordinates of the cell outline obtained from OPTIMAS software are assigned to the variables 't' and 'y'.

3. A user-defined function expfun accepts the parameters 'a' and 'b' from the random set created in step 1. Input parameters also include 't' and 'y'. The variable Fitted_curve $= \dfrac{b}{a}\sqrt{a^2 - (x - x_i)^2} + y_i$ where (x_i, y_i) is the center of the Ellipse.

4. The error is the difference of original y values (obtained from OPTIMAS) and calculated y values Error_Vector = Fitted_curve − y

5. Minimizing the sum of squares of error fits the curve, i.e., sse needs to be minimized

   sse $=$ (Error_Vector)^2

6. 'fminsearch' (MATLAB inbuilt function) is used to minimize the above value. The options required for the search are set to default.

7. Obtained y values are plotted along with the original values.

8. 'a' and 'b' values are obtained on the MATLAB command window as an answer.

   Figure 26 shows the best-fit ellipse for a given set of cell outline data points. The bottom and top halves of the ellipse are fit separately and the average 'a' and 'b' values were used for fitting the complete ellipse (Appendix 9).
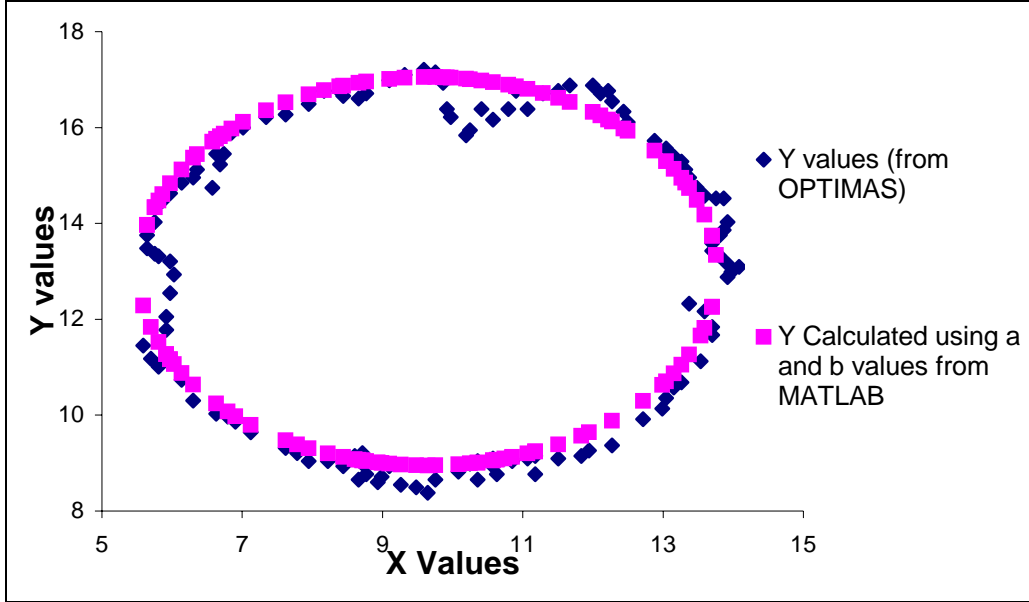
Figure 26 The best-fit ellipse for a given set of cell outline data points is shown in the above picture.

4.7 U$_r$ and RUTL:

As mentioned earlier, Compressive loading was applied to the models in the global x-direction and radial deformations were acquired. Deformed and undeformed cell outlines were fit with the ellipse. Since equation of the ellipse is known, the radius of the cell can be calculated at different angles (.) w.r.t loading axis, for both compressed and uncompressed cells.

We know that, $\dfrac{x^2}{a^2}+\dfrac{y^2}{b^2}=1$ is the equation of ellipse,

but $x = r \cos\theta$ and $y = r \sin\theta$, where r is in microns and θ is in radians.

Substituting in the equation of ellipse, we get

$$\frac{(r\cos.\ )^2}{a^2}+\frac{(r\sin.\ )^2}{b^2}=1$$

$$\Rightarrow r^2\left[\frac{(b\cos.\ )^2+(a\sin.\ )^2}{a^2 b^2}\right]=1$$

37

$$\Rightarrow r = \frac{ab}{\sqrt{(b\cos.\ )^2 + (a\sin.\ )^2}}$$

Hence from the above equation, radii of uncompressed and compressed cells can be obtained for different values of theta. The difference in radii of uncompressed and compressed cells (U_r mm) is then plotted against the angle w.r.t to the loading axis (X axis). The ratio A/B is termed to be the RUTL (Ratio of U_r value in Transverse direction to that in the Loading direction) of the cell. The variation of these two properties due to an applied compressive loading on the cell-gel construct was analyzed.

Figure 27 shows Ur – θ curves obtained in both the cases. The deformation in the model with embedded horizontal elements is lower than that in the isotropic case, as expected. But the difference in the deformations is very small due to low volume fraction of the actin fibers. We know that the stiffness of composite depends on the stiffness and volume fraction of matrix and fibers

$E = E_m V_m + E_f V_f$ Where E_m, V_m are Stiffness and Volume Fraction of Matrix

And E_f, V_f are Stiffness and Volume Fraction of Fiber.
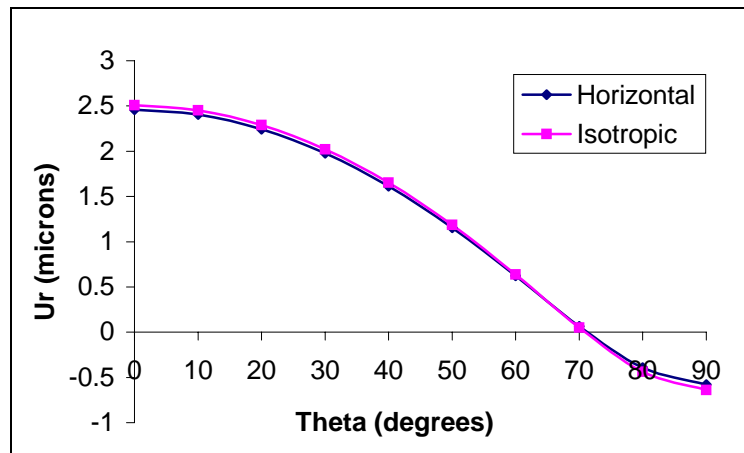
Figure 27 The graph shows variation of Ur – θ curves for Isotropic and Horizontal actin fibers in gel-cell model. Deformation in case of horizontal fibers is < that in isotropic case as expected.

In this case, the cell is the composite, actin are the fibers and the rest of the cell cytosol is the matrix. For a cell radius of 4μm, slice thickness of 2μm, the volume of cell (matrix) obtained was 100.53 cubic μm. Similarly for actin radius of 4nm and length 25.55μm, the volume of actin (fibers) obtained was 0.00128 cubic μm. Thus the volume of rest of the cell cytosol can be obtained as the difference in the volumes of cell matrix and fibers (100.528 cubic μm in this case). From the above, volume fraction of fibers can be calculated as the ratio of volume of actin fibers to the total volume (Volume of fibers/total composite volume), 0.000127 in this case. Volume fraction of fibers was obtained to be 0.9998. Thus even if the stiffness of the actin fibers is high (363 MPa), their affect on the cell deformation is low if the volume fraction of the actin is low. The RUTL obtained in the above case are 0.2352 for Horizontal fibers case and 0.2542 for isotropic case.

Chapter 5   Finite Element Approach

5.1 Introduction:

An important step in fulfilling the objective of this project was to reconstruct 2D model of the gel-cell construct with embedded actin filaments using Finite Element Packages I-DEAS and ABAQUS. Confocal images of 5 groups of cell cross-sections were obtained, each group containing 4 images (Days 1, 5, 8 and 12). All the 20 images were post processed using OPTIMAS software. 2D models of each image were created and analyzed under uniaxial compressive loading. U_r and RUTL were then calculated for each group. The 2D reconstruction process involved the following steps.

5.2 Cell Outline:

First step was to obtain the cell outline. Points representing the cell outline were obtained from OPTIMAS. The (X,Y) coordinates of these points were then stored in a data ('point.dat') file. A FORTRAN code was written to convert this data file into I-DEAS program file ('.prg' file/macro). This program when run in I-DEAS displays the cell outline points on the I-DEAS screen. The algorithm is as follows:

1. The input for this FORTRAN Code is a data file "point.dat" containing the (X,Y) coordinates of the cell outline.

2. Create a new blank program file "format.prg" to enter the required code for conversion of the data points into I-DEAS macro format.

3. Write the initial input file to open the **Master Modeler> Create points in 3D** options in I-DEAS. This is generally in the format "K : $ $ $ /ta MM" for Master Modeler and "K : $ /cr m3 p" for 3D points (Appendix 7)

4. Open the file "point.dat" containing the cell outline data points.

5. Use a Do Loop to read the (X,Y) coordinates from "point.dat" and write them to the "format.prg" file (in the format "K : X coordinate, Y coordinate")

6. Close the file "point.dat".

7. Enter the commands to end the program file "format.prg" in the I-DEAS macro format. ("K: $ mpos :; /F PR E" etc.)

8. Close the "format.prg" file.

9. If the FORTRAN code is compiled and run, it creates an I-DEAS program file "format.prg" to display the cell outline points on the I-DEAS screen.

5.3 Gel-Cell Construct:

The next step is to create the 2D model consisting of the cell and gel.

1. The cell outline points displayed on the I-DEAS screen are joined using **Master Modeler> Lines in 3D** option in I-DEAS. Now an outline is created for the cell model.

2. A rectangle of dimensions (500 x 500) is created around the cell using **Master Modeler> Create Rectangle** option in I-DEAS. Ensure that the cell is approximately at the center of the rectangle.

3. Make the rectangle into a surface by using **Master Modeler> Surface by Boundary** option in I-DEAS.

4. Extrude the cell outline by using the **Master Modeler> Extrude** option. Give the thickness to be a very small value (0.01) and use the **Partition** option in Extrude to make the cell and gel into different areas.

5. Now a 2D geometry of cell in gel is created.

6. Constrain the left end of the gel in all directions, by applying 0 displacement and rotation to all Degrees of freedom on the left edge of the geometry.

7. Apply a pressure load of 0.00484 MPa on the right edge of the geometry.

8. Assign material properties for the cell. E = 385 Pa, v = 0.499

9. Assign material properties for the gel. E = 26.7 KPa, v = 0.499

10. Mesh the cell with 2D plane stress (CPS4) elements, using a <u>fine mesh</u>. Group the nodes on the cell outline and specify the group name to be "OUTLINE"

11. Mesh the gel with 2D plane stress (CPS4) elements. Since the cell has a fine mesh, I-DEAS automatically generates a refined mesh at the gel-cell interface.
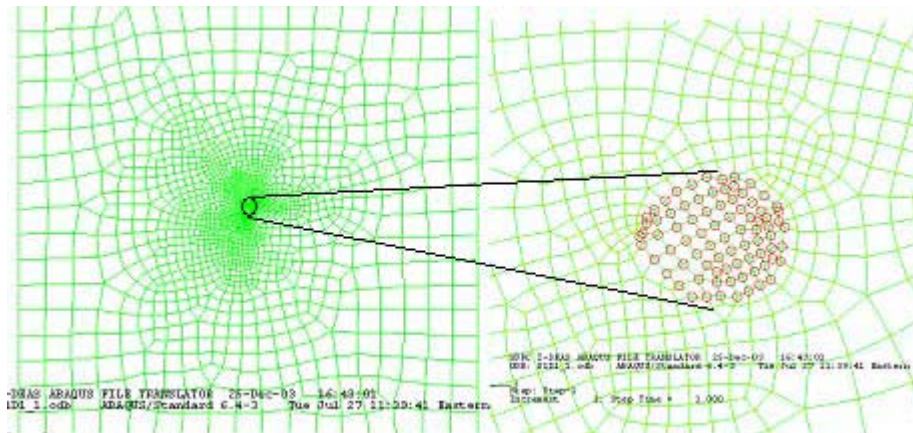
Figure 28 The left picture shows the cell-gel mesh. There is a fine mesh near the center of the gel where the cell is located. The right picture shows highlighted nodes of the cell in a gel.

12. Run the analysis and export the I-DEAS file to ABAQUS input file. Make sure that the nodeset "OUTLINE" is also written to the ABAQUS input file.

These steps create a 2D model of a cell-gel construct. One important point is to remember that the orientation of the nodes should be changed from counter clockwise to clockwise direction, after exporting to the ABAQUS input file.

5.4 Embedded elements:

The next step is to create actin filaments as embedded fibers in the cell outline. Actin fibers exhibit a change in their structure (from fiber to punctate appearance) from Day 1 to Day 12. Fiber-like structures were modeled as 2D truss elements (T3D2) and punctate structures were modeled as 2D 4-noded plane stress elements (CPS4). A FORTRAN Code was written to create the ABAQUS input file to incorporate these structures into the cell-gel model.

5.4.1 Truss Elements:

End coordinates of the lines representing actin filaments in the cell cross-section can be obtained from OPTIMAS. The purpose of this program is to read the end coordinates (x_1,y_1) and (x_2,y_2) of the data set obtained from OPTIMAS, convert them to nodes and join them as T3D2 Elements. This program can be used to reconstruct the lines representing the actin fibers, like those on Day1. The data set of the end coordinates should be stored in a file "linecoords.dat". The first two points in the data set correspond to one line in the image, the next two points make the second line and so on. If the file "linecoords.dat" is in the following format

1,	0.0,	0.0,	0.0
2,	1.0,	1.0,	0.0
3,	0.0,	1.0,	0.0
4,	-1.0,	1.0,	0.0
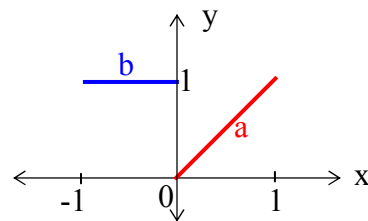
The first two points 1(0,0) and 2(1,1) create the line 'a' as shown in the above diagram. Similarly the points 3(0,1) and 4(-1,1) form the line 'b'. The following flow chart describes the FORTRAN code (See Appendix 8)

```
                    ┌─────────────────┐
                    │     Start       │
                    └─────────────────┘
                             │
                             ▼
         ┌──────────────────────────────────────────┐
         │   Open a new input file "opmodel.inp"     │
         └──────────────────────────────────────────┘
                             │
                             ▼
         ┌──────────────────────────────────────────┐
         │     Open the file "linecoords.dat"        │
         └──────────────────────────────────────────┘
                             │
                             ▼
         ┌──────────────────────────────────────────┐
         │        Set number of nodes n=0            │
         └──────────────────────────────────────────┘
                             │
                             ▼
         ┌──────────────────────────────────────────┐
         │          Do Loop, i = 1 to EOF            │
         └──────────────────────────────────────────┘
                             │
                             ▼
         ┌──────────────────────────────────────────┐
         │ Read the line end coordinates from        │
         │ "linecoords.dat" into an array            │
         │ Node Number, X – coord, Y – coord, Z – coord │
         └──────────────────────────────────────────┘
                             │
                             ▼
         ┌──────────────────────────────────────────┐
         │ Write the nodal coordinates to the input  │
         │ file in the following format              │
         │ Node number + 100000, X – coord, Y – coord, Z – Coord │
         └──────────────────────────────────────────┘
                             │
                             ▼
                    ┌─────────────────┐
                    │    n = n + 1    │
                    └─────────────────┘
                             │
                             ▼
                          (  A  )
```

```
                              ( A )
                                |
                                v
+-------------------------------------------------------------+
|  Generate T3D2 elements, Set number of elements ielno=1     |
+-------------------------------------------------------------+
                                |
                                v
+-------------------------------------------------------------+
|                    Do Loop, k = 1 to n                      |
+-------------------------------------------------------------+
                                |
                                v
                           /  k mod 2  \         False
                          <    ≠ 0      >  ------------->
                           \           /
                                |
                              True
                                |
                                v
+-------------------------------------------------------------+
|  Assign,   n1 = Node number (k)      n2 = Node number (k+1) |
+-------------------------------------------------------------+
                                |
                                v
+-------------------------------------------------------------+
|                   Write to input file,                      |
|        ielno + 100000, n1 +100000, n2 + 100000              |
+-------------------------------------------------------------+
                                |
                                v
                    +---------------------+
                    |  ielno = ielno + 1  |
                    +---------------------+
                                |
                                v
+-------------------------------------------------------------+
|  Enter the material properties of the actin in an ABAQUS    |
|  input file format and close all the files.                 |
+-------------------------------------------------------------+
                                |
                                v
                        (   Stop   )
```

Figure 29 The picture shows the reconstructed cell model (right) with embedded truss elements and the cell cross-section (left)

The above picture shows the cell model reconstructed from the cell cross-sectional image. Actin filaments are embedded as Truss elements into the cell (CPS4) elements. Embed Truss-in-Solid concept of ABAQUS was used for this purpose.

Auto meshing in I-DEAS created the node numbers and element numbers for the cell-gel model. Hence, the node and element numbers for the truss elements were incremented by 100000 (just an arbitrarily large number chosen assuming that the total elements in the gel will not exceed this value), in this program, to avoid repetition of the numbers. The output for this FORTRAN code is an ABAQUS input file. This file when combined with the input file for creating Gel-Cell construct, provides the final input file for creating the Cell-gel model with embedded actin fibers.

5.4.2 Solid Elements:

Actin fibers that have a punctate appearance (like those on Day 12) cannot be represented by 2D Truss elements. In this case, actin fibers were considered as 2D Plane stress (CPS4) elements. If we consider the punctate fibers to be an area on the cell cross-section, OPTIMAS provides the 2 corner end points of the rectangle that surrounds this area.
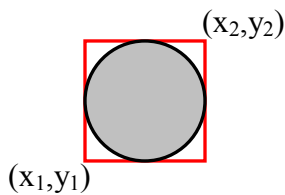
For example, if we consider the area to be a circle as shown, then OPTIMAS provides the coordinates (x_1,y_1) and (x_2,y_2) of the rectangle Bounding the Region of Interest (circle).

A FORTRAN Code was written to convert these end coordinates into 4-noded CPS4 elements. If we input the end coordinates (x_1,y_1) and (x_2,y_2) to the FORTRAN program, it creates the output file containing the 4 nodes, (x_1,y_1), (x_1,y_2), (x_2,y_2), and (x_2,y_1). Connecting these 4 nodes in the clockwise direction creates the 4-noded CPS4 element.

Input for this program is a file "solidcoords.dat" containing the end coordinates of all such rectangles. For example, if the file "solidcoords.dat" contains the following set of points,

1,	0.0,	0.0,	0.0
2,	1.0,	1.0,	0.0
3,	0.0,	1.0,	0.0
4,	-1.0,	-2.0,	0.0
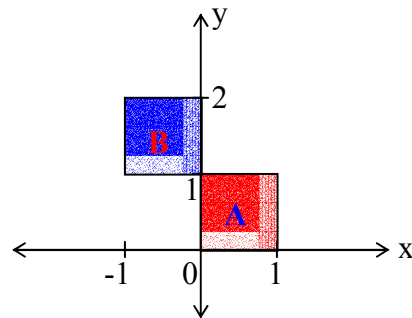
The area 'A' will be formed by the points 1 & 2 and the area 'B' by the points 3 & 4. In case of coincident areas, the smaller of the two areas was removed. 'A' and 'B' are two coincident areas, in this case, area 'A' is deleted. This coincidence was determined visually after the model is created.

The following flowchart describes the FORTRAN Code (See Appendix 7)


```
                    ┌──────────────┐
                    │    Start     │
                    └──────────────┘
                            │
        ┌───────────────────────────────────────────┐
        │   Open a new input file "opmodel.inp"      │
        └───────────────────────────────────────────┘
                            │
        ┌───────────────────────────────────────────┐
        │   Open the file "solidcoords.dat"          │
        └───────────────────────────────────────────┘
                            │
        ┌───────────────────────────────────────────┐
        │   Set number of nodes n=0                  │
        └───────────────────────────────────────────┘
                            │
        ┌───────────────────────────────────────────┐
        │   Do Loop, i = 1 to EOF                    │
        └───────────────────────────────────────────┘
                            │
        ┌───────────────────────────────────────────┐
        │ Read the line end coordinates from         │
        │ "solidcoords.dat" into an array            │
        │ Node Number, X – coord, Y – coord, Z – coord│
        └───────────────────────────────────────────┘
                            │
                    ┌──────────────┐
                    │  n = n + 1   │
                    └──────────────┘
                            │
        ┌───────────────────────────────────────────┐      (A)
        │  Write the nodal coordinates to the input file │
        └───────────────────────────────────────────┘
                            │
        ┌───────────────────────────────────────────┐      (B)
        │  Write the element data to the input file      │
        └───────────────────────────────────────────┘
                            │
        ┌───────────────────────────────────────────┐      (C)
        │  Enter material properties of actin and close all files │
        └───────────────────────────────────────────┘
                            │
                    ┌──────────────┐
                    │    Stop      │
                    └──────────────┘
```

```
                              ( A )
                                |
                                v
        +---------------------------------------------------+
        |    Generate 4 nodes from the given 2 end points   |
        +---------------------------------------------------+
                                |
                                v
        +---------------------------------------------------+
  +---->|              Do Loop, j = 1 to n                  |
  |     +---------------------------------------------------+
  |                             |
  |                             v
  |                        /         \
  |                       /  j mod 2  \      False
  |                      <    ≠ 0       >-----------> ( B )
  |                       \           /
  |                        \         /
  |                             | True
  |                             v
  |  +------------------------------------------------------------------+
  |  |                   Write to input file,                           |
  +--| n1 = Node number (j)+100000      n2 = Node number (j)+ 200000    |
     | x1 = xcoord(j)                   x2 = xcoord(j+1)                 |
     | y1 = ycoord(j)                   y2 = ycoord(j)                   |
     | n3 = Node number (j+1)+100000    n4 = Node number (j)+ 200000    |
     | x3 = xcoord(j+1)                 x4 = xcoord(j)                   |
     | y3 = ycoord(j+1)                 y4 = ycoord(j+1)                 |
     | in the format, Node Number, X – coord, Y – coord, Z – coord      |
     +------------------------------------------------------------------+
```

The above flowchart (Part A) describes the node generation part of the FORTRAN code for creating solid elements. Consecutive points are read from the input file and the four corner nodes for 2D CPS4 elements are created using the above logic.

The following flowchart (Part B) describes the element generation part of the FORTRAN Code. After the nodes are created, the nodes are arranged in clockwise direction to obtain the 2D CPS4 elements.
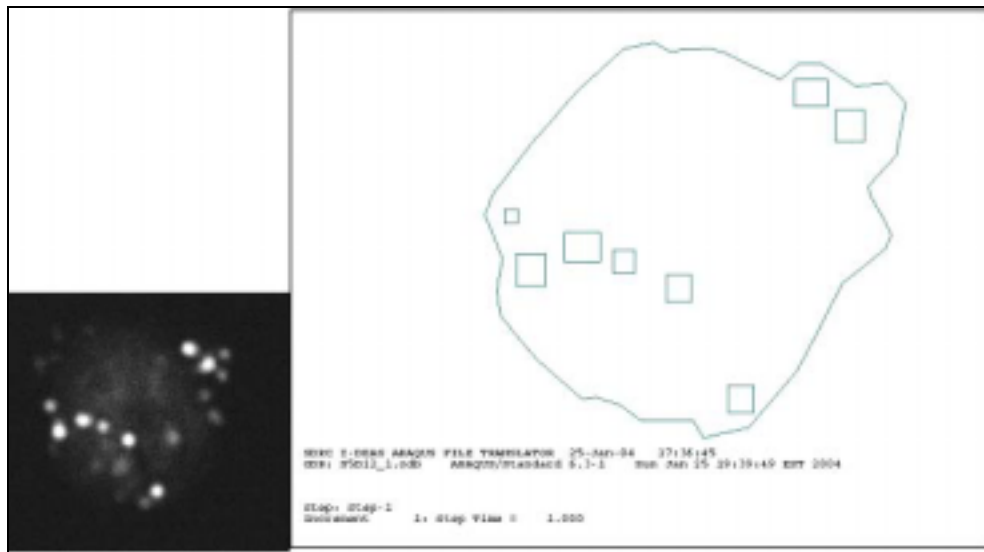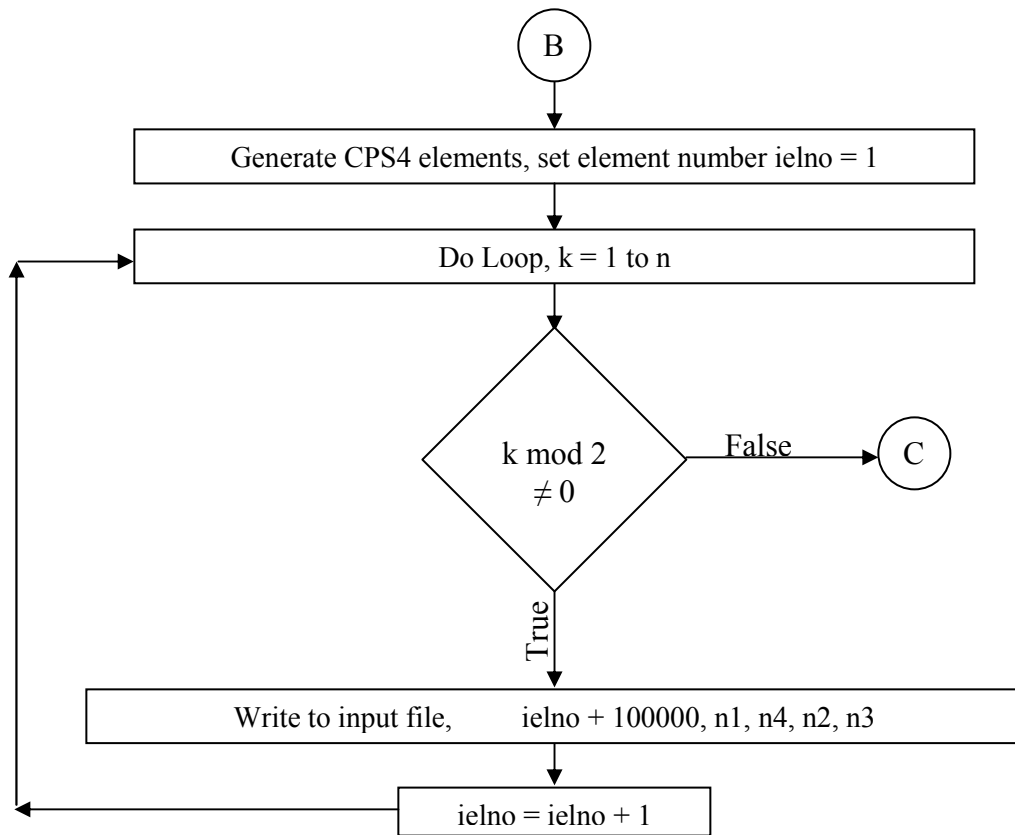
Figure 30 The picture shows the reconstructed cell model (right) with embedded CPS4 elements and the cell cross-section (left).

To avoid repetition of node/element numbers, the node numbers were incremented by values 100000 and 200000 (numbers chosen assuming that the gel elements will not exceed the value 100000). Figure 30 shows the cell model reconstructed from the cross-sectional image. The actin fibers that appear to be punctate in shape are observed as 2D CPS4 elements in the reconstructed model. The output file "opmodel.inp" when combined with the ABAQUS input file for creating Cell-gel model, gives the final file to create the whole model (cell-gel with embedded actin).

5.5 Boundary Conditions:

The input file is created to generate a cell-gel model in ABAQUS. The next step is to check the boundary conditions. As mentioned earlier (5.3), the left edge of the cell-gel model is constrained and a compressive loading was applied on the right edge.
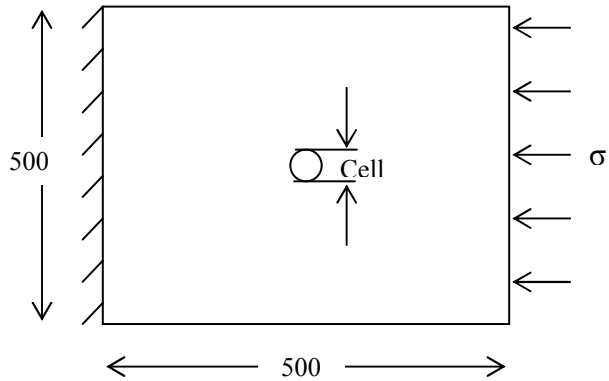
The following are the material properties and boundary conditions for the analyses:

Component	Elastic Modulus (MPa)	Poisson's Ratio (ν)	Dimensions
Actin	363	0.499	Length – from OPTIMAS Diameter $= 8 \times 10^{-3} \mu$
Cell	385×10^{-6}	0.499	From OPTIMAS
Gel	29.7×10^{-3}	0.499	$500\mu \times 500\mu$

Table 2 Properties of actin, cell and gel used for analyses.

Applied Load (= σ) = 0.00484 MPa (This value of stress produced a gel stiffness of 29.7 KPa experimentally)

The boundary conditions were maintained the same through out the analyses in all 5 groups. The gel and cell properties were derived from earlier experiments in our laboratory. Other researchers determined actin properties, and all the materials were considered to be incompressible. A no-slip condition was assumed between the cell and gel.

5.6 Deformations from ABAQUS output file:

Once the material properties are verified, the analyses are carried out in ABAQUS. The next step is to determine the deformations of the cell, due to the applied load. After the analysis is run in ABAQUS, the results (deformations, stresses, strains etc.,) are stored in a results file "*filename*.**fil**". The filename is the same for the input and results files. User subroutines were created, using FORTRAN, to extract the cell deformations from the ABAQUS results file.

The user subroutines created were compiled using the ABAQUS/Make command. This command compiles the user subroutine and creates an executable file that can be used to run using ABAQUS execution procedures. The syntax for the ABAQUS Make execution command is as follows:

abaqus make job = *filename*
    this command compiles the user subroutine and creates an executable file,

abaqus filename
    this command will run the execution procedure.

For obtaining the radial deformations of the cell ($U_r - \theta$ graph), it is sufficient if the deformations of the cell outline are obtained. In the step 12 of section 5.3, the

nodes of the cell outline were written as a node set "OUTLINE". Thus, if the deformed and undeformed coordinates of the node set "OUTLINE" are determined, the radial deformations of the cell can be obtained. Extracting the nodal deformations from the ABAQUS results file contains two stages. In the first stage, the nodes in the node set "OUTLINE" are extracted and in the second stage, the nodal deformed and undeformed coordinates are extracted.

5.6.1 Extracting the nodes on cell outline:

A user subroutine was written to extract node numbers of the nodes in the node set "OUTLINE". Input for this program is the ABAQUS results *filename* and the number of nodes in the node set (this number can be obtained from the ABAQUS input file). Output is a file "nsetno.dat" that contains the node numbers of the nodes in the node set "OUTLINE". ABAQUS results files can be accessed using specific numbers called Keys, which are outlined in the ABAQUS Standard User's Manual. Node numbers from a particular node set can be retrieved from the ABAQUS results file using the Key 1931. Attributes for this key are Node set name, First node in the node set, Second node in the node set, etc,.

JRRAY is an array containing all the data related to the node set. The first attribute in the array is the Record Length, which is the same for any record. The second attribute is the Record Type Key (JRRAY(1,2) =1931 in the above case). The third attribute is the node set name (OUTLINE in the above case), which is in the A8 format, i.e., 8 characters per name. (Appendix 10)

The following flowchart describes the user subroutine to extract the nodes in the node set "OUTLINE" from the **.fil** file.

```
                        ┌─────────────┐
                        │    Start    │
                        └─────────────┘
                               │
                               ▼
        ┌──────────────────────────────────────────────────┐
        │   Read the filename FNAME of the .fil file         │
        │ Read the number of nodes (nsetn) in the node set   │
        │                  "OUTLINE"                         │
        └──────────────────────────────────────────────────┘
                               │
                               ▼
        ┌──────────────────────────────────────────────────┐
        │        Open a new output file "nsetno.dat"         │
        └──────────────────────────────────────────────────┘
                               │
                               ▼
        ┌──────────────────────────────────────────────────┐
        │          Do Loop – To access the results file     │
        └──────────────────────────────────────────────────┘
                               │
                               ▼
        ┌──────────────────────────────────────────────────┐
        │  Do Loop – To access the KEY records from the .fil │
        │                      file                          │
        └──────────────────────────────────────────────────┘
```

Key = 1931 — True — Nset name = OUTLINE — True → A

False

False

Continue ← B

Continue

Close all files

Stop

```
                    ┌───┐
                    │ A │
                    └─┬─┘
                      │
                      ▼
    ┌─────────────────────────────────────────┐
    │       Set variables c = 2 and i = 5      │
    └─────────────────────┬───────────────────┘
                          │
                          ▼
    ┌─────────────────────────────────────────┐
    │ Write to output file, Node Number (1) = JRRAY(1, 4) │
    └─────────────────────┬───────────────────┘
                          │
                          ▼
    ┌─────────────────────────────────────────┐
    │      Do Loop, i = 5 to (nsetn + 3)       │
    └─────────────────────┬───────────────────┘
                          │
                          ▼
    ┌─────────────────────────────────────────┐
    │  Write to file, Node Number (c) = JRRAY(1, i) │
    └─────────────────────┬───────────────────┘
                          │
                          ▼
              ┌───────────────────┐
              │     c = c + 1     │
              └─────────┬─────────┘
                        │
                        ▼
                    ┌───┐
                    │ B │
                    └───┘
```

5.6.2 Extracting nodal deformations of cell outline:

Node numbers on the cell OUTLINE are now used to extract the nodal deformed and undeformed coordinates by using another user subroutine. As mentioned earlier, all the data of a particular Key are accessible in the form of an array 'JRRAY'. The Key 1901 is used for extracting undeformed nodal coordinates and Key 101 for deformed coordinates. The output file ("nsetno.dat") of step 5.6.1 is used as the input file for this program. ABAQUS results *filename* is to be specified as another input. The output files are "ornode.dat" containing the undeformed nodal coordinates and "denode.dat" containing the deformed nodal coordinates (Appendix 11). The flowchart shows the logic used for this user subroutine (written in FORTRAN):

```
                        ┌──────────┐
                        │  Start   │
                        └──────────┘
                             │
                             ▼
        ┌─────────────────────────────────────────────────┐
        │     Read the filename FNAME of the .fil file     │
        └─────────────────────────────────────────────────┘
                             │
                             ▼
        ┌─────────────────────────────────────────────────┐
        │  Open the input file "nsetno.dat"and output      │
        │  files "ornode.dat" and "denode.dat"             │
        └─────────────────────────────────────────────────┘
                             │
                             ▼
        ┌─────────────────────────────────────────────────┐
        │ Read node numbers into an array NSRAY,           │
        │ number of nodes = nn                             │
        └─────────────────────────────────────────────────┘
                             │
                             ▼
        ┌─────────────────────────────────────────────────┐
        │ Do Loop – To access the KEY records from the     │
        │ .fil file                                        │
        └─────────────────────────────────────────────────┘
                             │
                             ▼
                         ╱Key ╲      True      ╭───╮
                        ╱ = 1931╲──────────────▶│ A │
                         ╲      ╱               ╰───╯
                          ╲    ╱
                        False
                             │
                             ▼
                         ╱Key ╲      True      ╭───╮
                        ╱ = 101 ╲─────────────▶│ B │
                         ╲      ╱               ╰───╯
                          ╲    ╱
                        False
                             │
                             ▼
                    ╭──────────────╮      ╭───╮
                    │   Continue   │◀─────│ C │
                    ╰──────────────╯      ╰───╯
                             │
                             ▼
        ┌─────────────────────────────────────────────────┐
        │                Close all files                   │
        └─────────────────────────────────────────────────┘
                             │
                             ▼
                        ┌──────────┐
                        │   Stop   │
                        └──────────┘
```
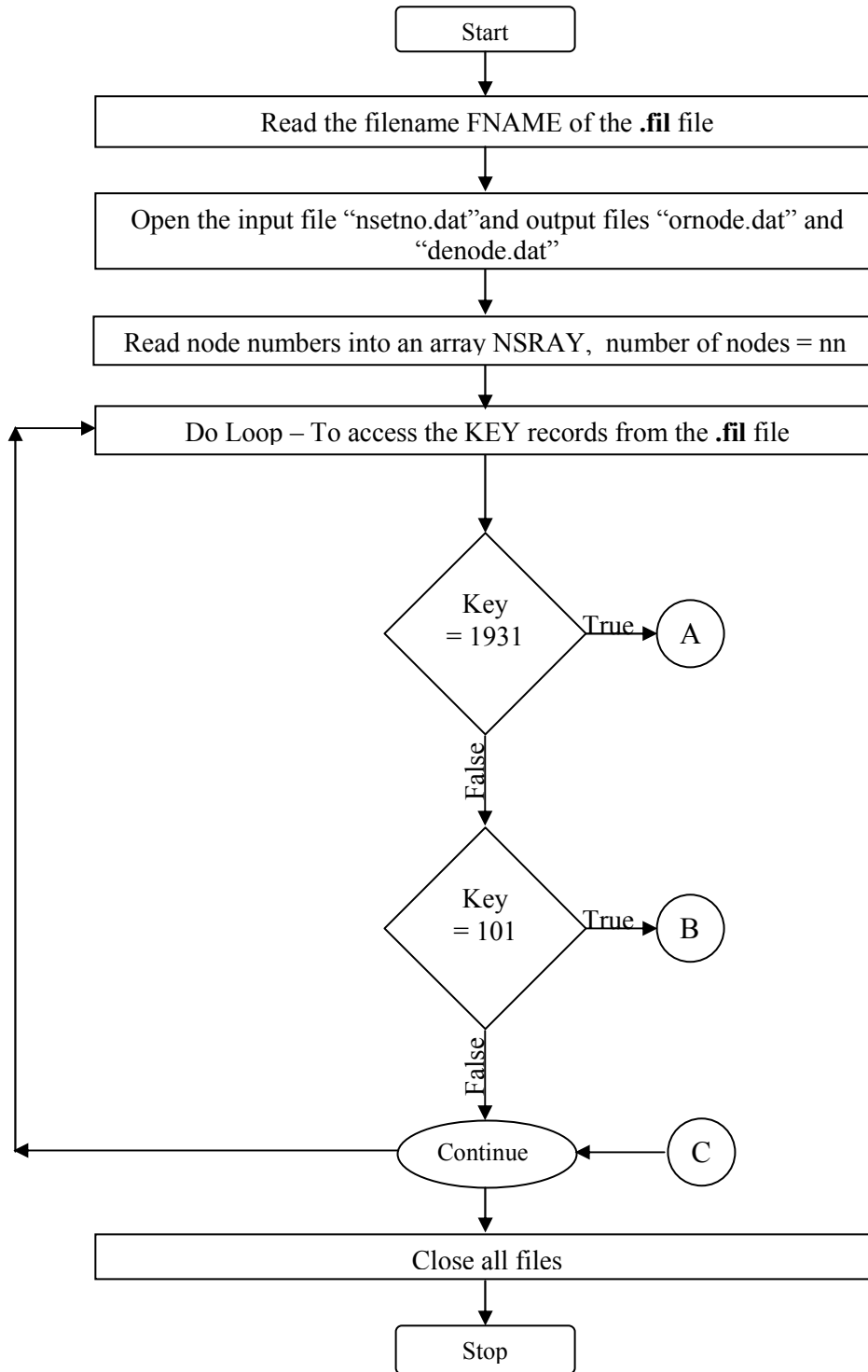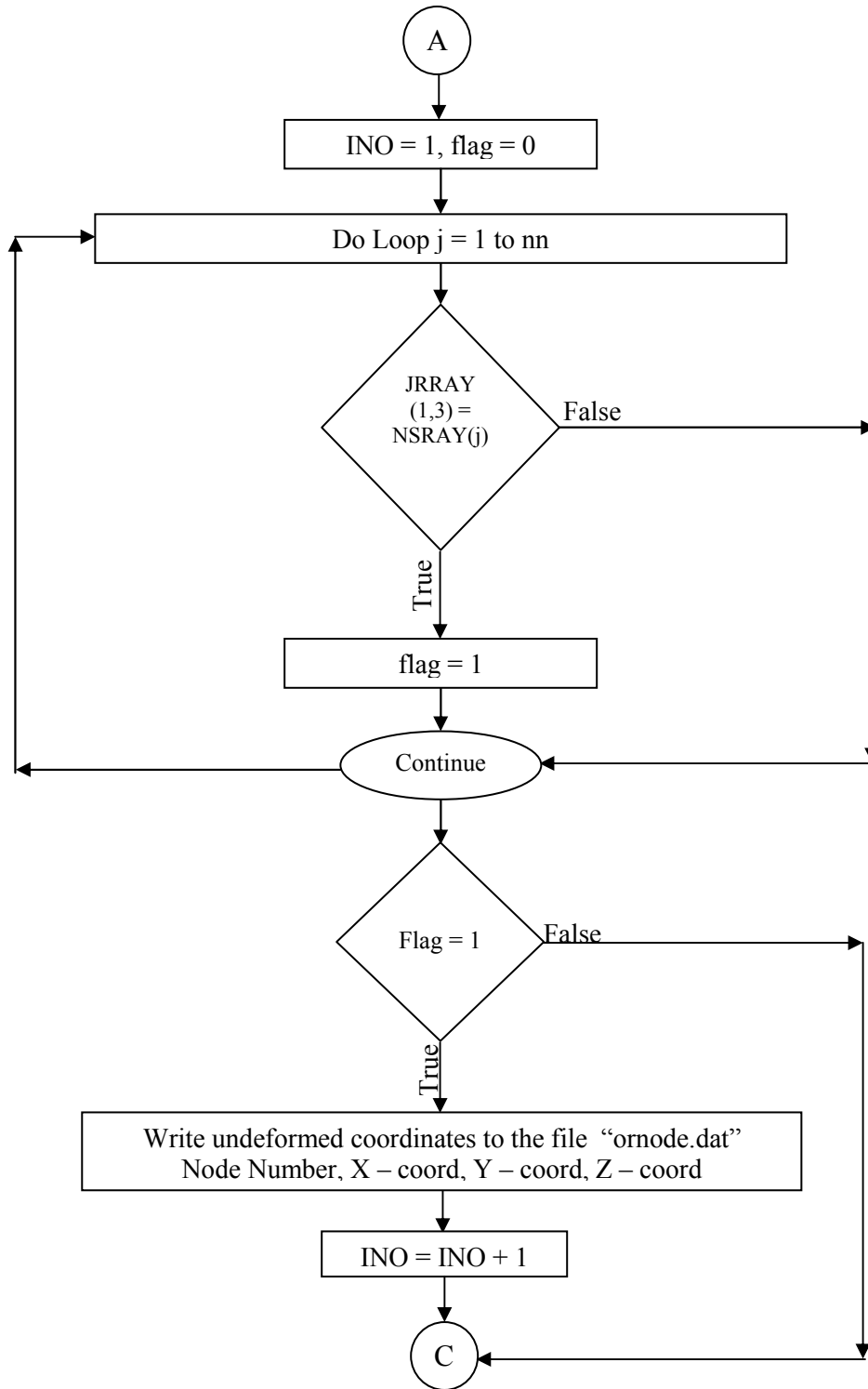
The above flowchart describes the user subroutine for extracting the undeformed nodal coordinates from the node set 'OUTLINE'.

```
                              ( B )
                                │
                                ▼
                    ┌───────────────────────┐
                    │  IDNO = 1, flag1 = 0   │
                    └───────────────────────┘
                                │
                                ▼
        ┌─────────────────────────────────────────────┐
   ┌───▶│            Do Loop k = 1 to nn               │
   │    └─────────────────────────────────────────────┘
   │                            │
   │                            ▼
   │                         ◇ JRRAY
   │                         (1,3) =      False
   │                          NSRAY  ─────────────────────────┐
   │                            (k) ◇                          │
   │                            │ True                         │
   │                            ▼                              │
   │                    ┌───────────────┐                     │
   │                    │   flag1 = 1    │                     │
   │                    └───────────────┘                     │
   │                            │                              │
   │◀──────────────(   Continue   )◀─────────────────────────┘
                                │
                                ▼
                             ◇ flag1    False
                               =1  ◇──────────────────────────┐
                                │ True                         │
                                ▼                              │
        ┌──────────────────────────────────────────────┐     │
        │ Write deformed coordinates to the file "denode.dat" │
        │  Node Number. X – coord. Y – coord. Z – coord  │     │
        └──────────────────────────────────────────────┘     │
                                │                              │
                                ▼                              │
                    ┌───────────────────┐                     │
                    │  IDNO = IDNO + 1   │                     │
                    └───────────────────┘                     │
                                │                              │
                                ▼                              │
                              ( C )◀──────────────────────────┘
```
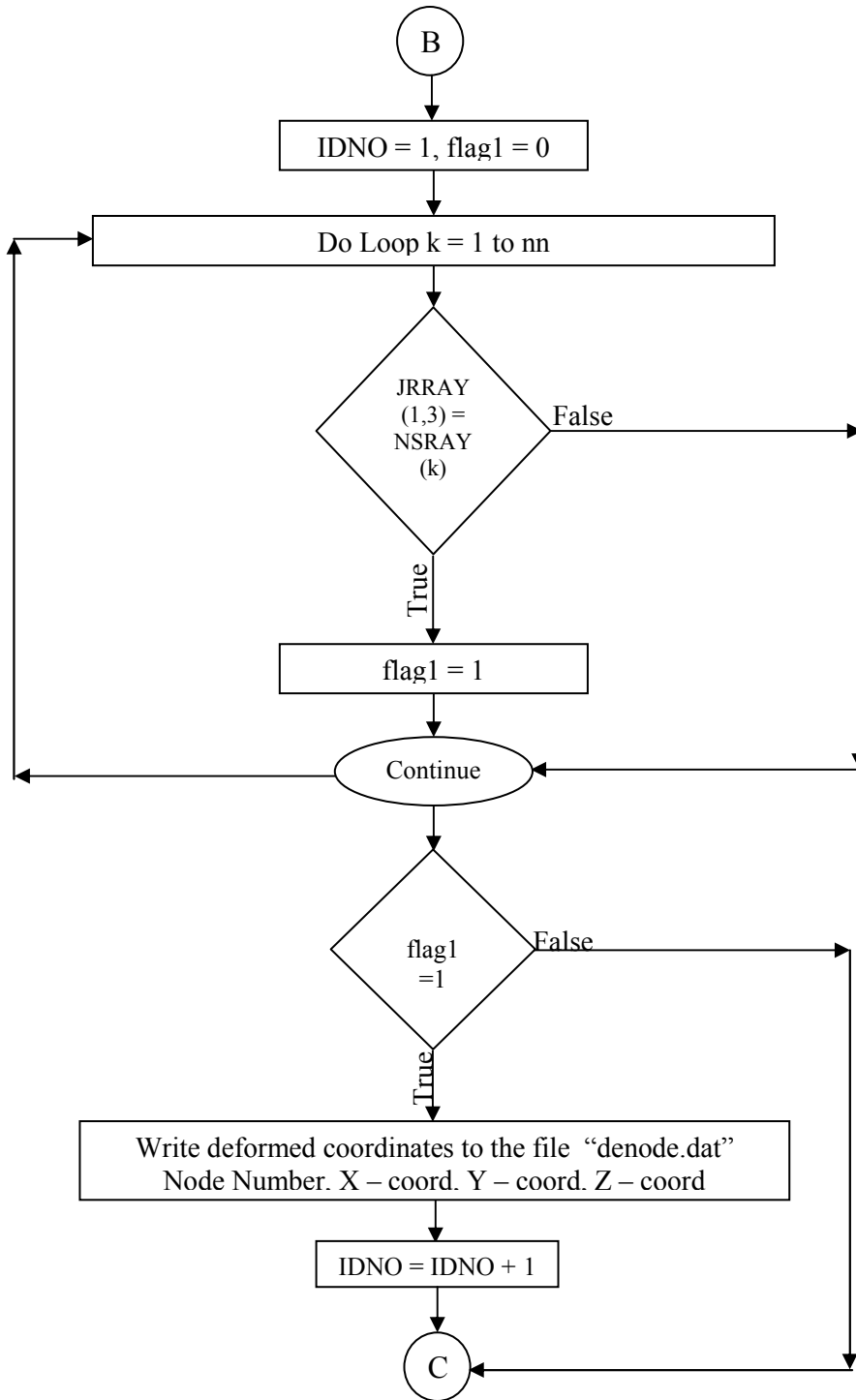
The above flowchart describes the part of user subroutine that explains the logic to extract the deformed nodal coordinates from the node set 'OUTLINE' and store them in the file "denode.dat".

5.7 Curve Fitting an Ellipse to obtain Ur values:

The centroid is calculated and an ellipse is fit to the undeformed coordinates obtained from the file "ornode.dat" (curve fitting procedure described in 4.6). This gives the undeformed nodal coordinates 'r' as a function of 'θ'. Similar procedure is adopted for the deformed nodal coordinates from the file "denode.dat". The radii of the undeformed and deformed cells are calculated at 10 different values of 'θ', ranging from 0° to 90°. The radial deformations of the cell are defined as the difference in the radii of their deformed and undeformed coordinates. Microsoft Excel was used to plot the radial deformations (U_r) against the theta values.

Chapter 6   Results and Discussion

6.1 U_r values

Experiments were carried out on five cell sets (each from Day1 to Day12) and their $U_r - \theta$ curves were obtained using Finite Element and curve fitting techniques discussed earlier. The mean values of deformations of all the sets for each day were plotted against the angle 'θ'. Figure 31 shows the variation of U_r on Day 1 for the five groups of experiments.
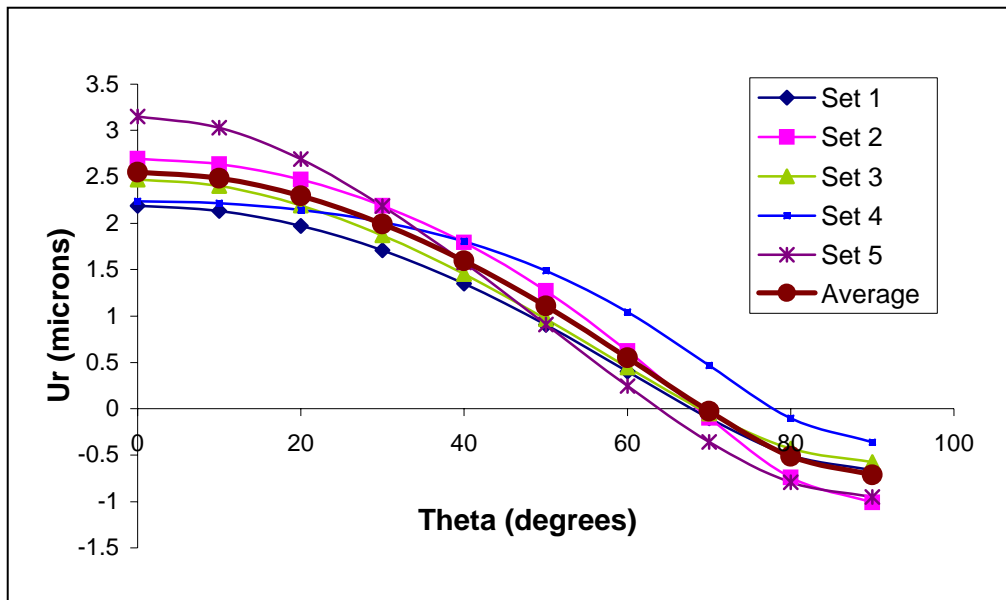
Figure 31 The graph shows Ur – θ graphs for different sets on Day 1. The average of all such curves was taken and plotted as shown.

As expected, the cell became thinner in the direction of loading (where U_r is positive) and longer in the direction perpendicular to loading (where U_r is negative). Similar graphs were obtained for all the days 1, 5, 8 and 12. The mean values of U_r when plotted with angle 'θ', demonstrated a structural variation from Day 1 to Day 12.
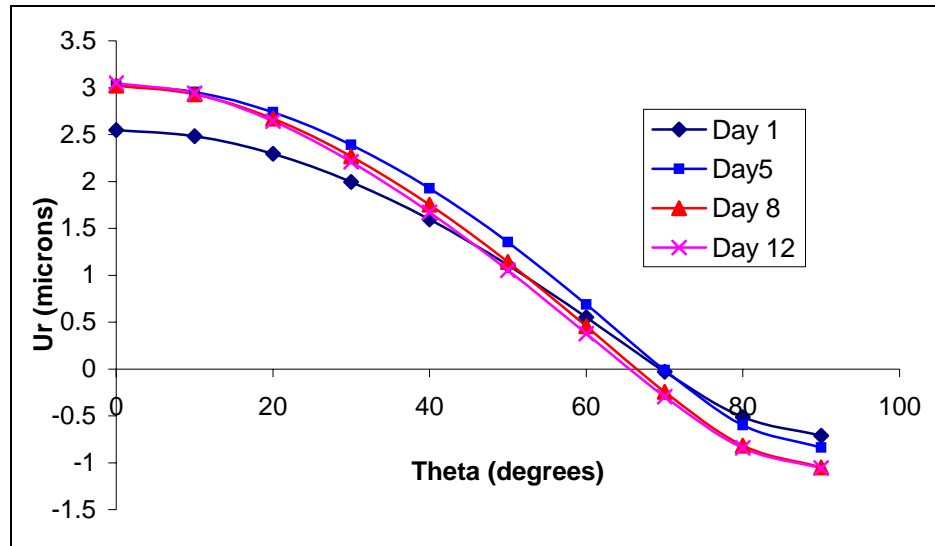
Figure 32 shows the variation of Ur – θ curves for different sets. The mean values of radial deformations (Ur) were plotted against the angle w.r.t loading axis (θ). It can be observed that the deformations on Day1 are less than those on Day 12.

The deformations on Day 1 were less than those on Day 12, as observed. A statistical test (ANOVA) was carried out on U_r values along the loading axes, for each set on each Day using JMPIN software. The results showed that the difference in the U_r values was not statistically significant. The p value obtained was 0.1176 ($p > 0.05$). Though the results were statistically insignificant it was observed that the cell became more compliant from Day 1 to Day 12.
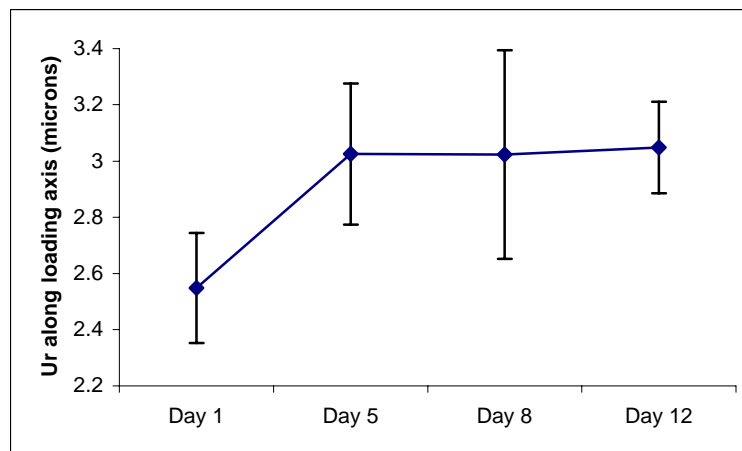
Figure 33 The graph shows variation of Ur values (along the loading axes – 90°) from Day 1 to Day 12. The standard deviations for each set can be observed.

61

6.2 RUTL:

As described earlier (section 4.7), RUTL is defined as the ratio of the difference in radial deformation in the direction perpendicular to loading, to that in the loading direction. For each set of analyses, RUTL was calculated. There was a difference in the values, but the difference was not statistically significant (p = 0.4577). The mean value of absolute RUTL for Day 1 is 0.27447, and that for Day 12 is 0.33705.
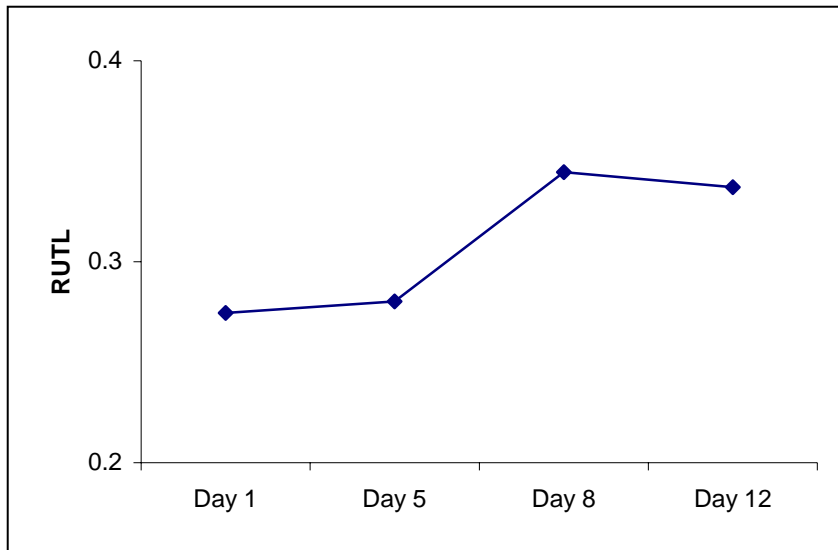
Figure 34 The variation of RUTL from Day 1 to Day 12.

A power analysis revealed that the power of these analyses (5 samples per group) was low (24.25%). The number of experiments that are to be carried out to provide results with a power of 70% was determined to be 13 samples per group. A higher accuracy of the results may be anticipated when more experiments and analyses are carried out.

6.3 Parametric Analysis:

6.3.1 Effect of Actin Properties on Ur and RUTL:

Parametric analysis was carried out to see the effect of variation of actin properties on the cell deformation. Elastic modulus was increased thousand fold (= 363 GPa) and the deformations were observed.
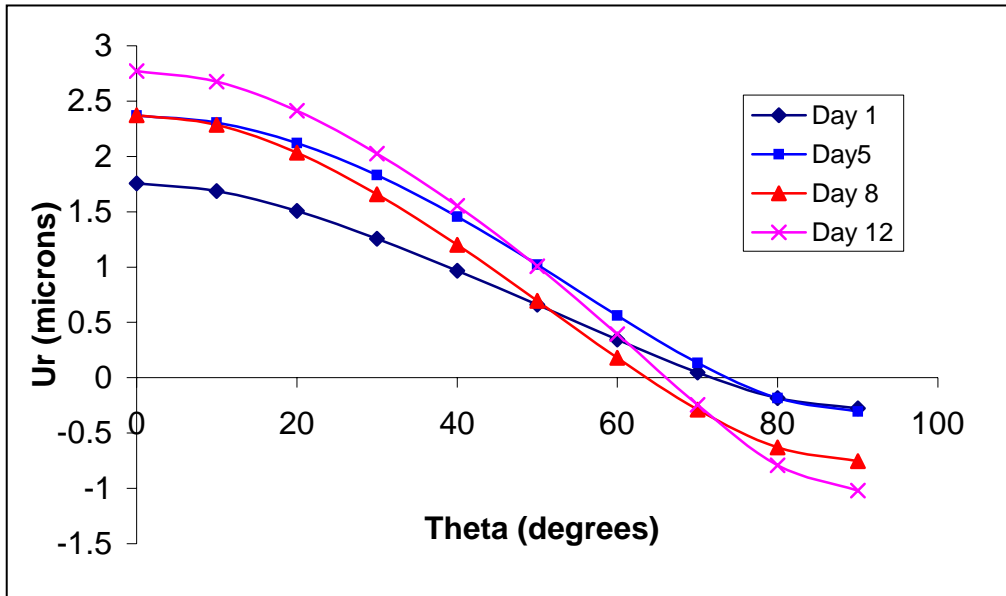
Figure 35 Variation of Ur – θ graphs from Day 1 to Day 12. Actin Elastic modulus increased 1000 fold (= 363 GPa)

The radial deformation behavior was similar in both the cases, magnitude of the difference in U_r increased with an increasing actin stiffness value. It was also observed that the cell tends to become more isotropic from Day 1 to Day 12. Analyses were carried out considering the cell to be a continuum element and aligning all the actin filaments in the horizontal directions. It was observed that the RUTL on Days 1 and 5 were lower than the Horizontal fibers case, and those on Days 8 and 12 were higher than the isotropic case. This makes us to believe that the cell tends to become more isotropic in nature, with increasing days in 3D culture.
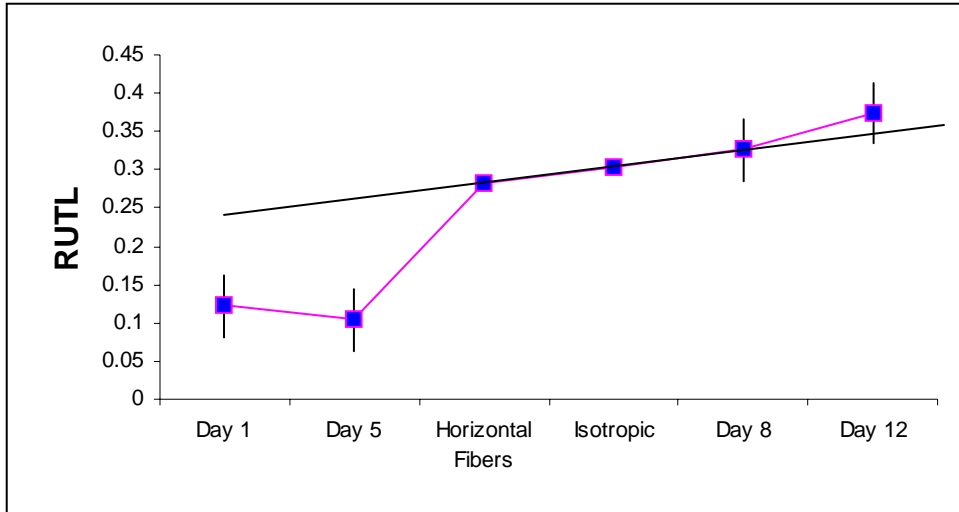
Figure 36 Variation of RUTL from Day 1 to Day 12

There was a significant difference (p<0.05) in the Ur values along the loading direction when the actin elastic modulus was increased 1000 times. Literature shows that Microtubules are another structural components of the cell that are 1000 times stiffer than actin in compression. Further studies need to be carried out incorporating all the structural parameters of the cell, for a better understanding of the exact scenario of the cytoskeletal reorganization and its implications on cellular response to mechanical loading.

6.3.2 Effect of Gel size on Applied Load:

Earlier it was observed that the variation in the applied load could not be measured with our existing setup. Though the sensitivity of the equipment was increased, it was difficult to capture the variations in the load data. Hence it was important to estimate the sensitivity of the new equipment that is to be designed. In order to provide an estimate, 15% compressive strain was applied on the loading edge of the FE model of the gel. This procedure was carried out for gel sizes ranging from $20\mu^3$ to $500\mu^3$. The stresses on the loading edge were obtained for day 1 and day 12

64

for all the gel sizes. The loads on the edges were calculated by dividing the stresses by their corresponding areas. It was observed that for the $20\mu^3$ gel, the load on Day 1 was 48.8nN and that for Day 12 was 152.3nN. The difference in loads was 103nN, and thus the difference in edge stresses is in the order of $3\mu N$. This change can only be measured using atomic force techniques or nanotechnology. The graphs show the change in the edge loads in absolute (Delta F) and relative ((Delta F)/F) values.
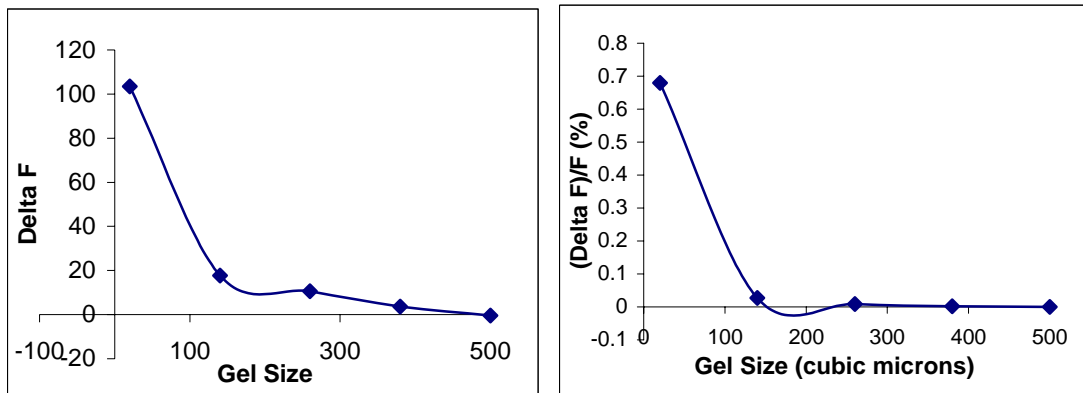
Figure 37 The left graph shows variation of applied loads with the gel sizes. The right graph shows percentage variation in applied loads with varying gel sizes.

6.3.3 Effect of Gel Size on Ur and RUTL:

Then we tried to observe the variation of U_r and RUTL with the gel size. A 15% compressive strain was applied on loading edge of the $20\mu^3$ gel-cell model. The deformations of the cell outline were observed. From the cell deformations, U_r and RUTL were obtained and analyzed for 4 sets cell-gel models.

It was observed that there was no significant difference (p=0.75 along Loading axis and p=0.5 along the direction perpendicular to loading) in U_r from Day 1 to Day 12. But, RUTL was a more sensitive parameter. There was a significant difference in the RUTL (p=0.0160) from Day 1 to Day 12. Hence we propose RUTL to be a more sensitive parameter to study the differences in cytoskeletal variations in

vivo. Figure 38 shows statistical analysis carried out on the RUTL values using JMPIN software.
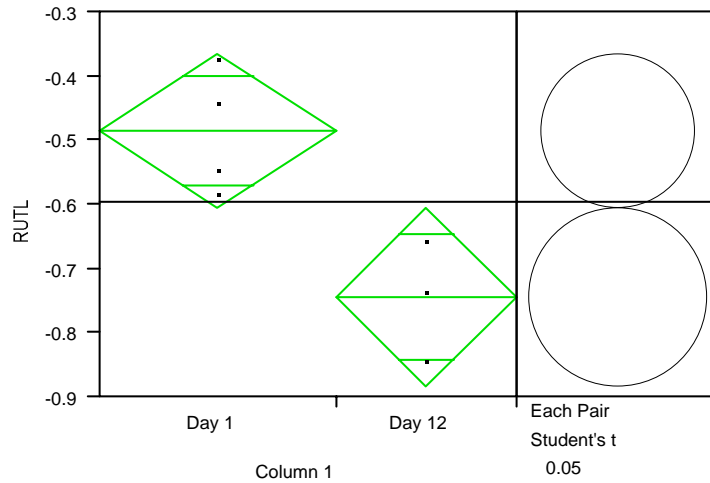
Figure 38 There exists a statistical difference in the values of RUTL from Day 1 to Day 12 (n=4)

6.4 Limitations:

This study provides an efficient way of observing the effect of variation of structural components of the cell on its material properties, but there are some limitations accompanying it. The cell model created highly depends on the accuracy of the images obtained from the microscope. Thus a more precise image provides a more accurate model. The reconstructed image also depends on threshold values chosen, which are user dependant.

The cells were assumed to be alive in the 3D culture period from Days 1 to 12, no experiments were carried out to check for their survival. OPTIMAS provided a number of lines representing the actin fibers, increasing the number of Truss elements in the analyses. Hence a trend line was obtained for each cluster of lines, and such trend lines were used for embedding in the cell element.

Recursive and coincident elements were deleted from the model, which may have an effect of the overall deformation. On the Day 12, elements were assumed to be rectangular in shape, though they appeared round in the cell cross-sectional images. Only actin filaments were considered to be present in the cell, effects of the other structural components of the cell were ignored. Also, viscoelastic properties of the gel were ignored.

6.5 Future Work:

A more realistic model can be created by incorporating all the structural components of the cell into the model. This may be achieved by staining different components using different fluorescent dyes. Confocal images of the cell along its diameter can be obtained to create a 3D model of the cell and to see the effect of lateral connectivity of the cell elements.

As observed from the project results, the difference in edge stress values is in the range of $3\propto N$, hence the future cell compression apparatus should be able to measure such differences. Nanotechnology techniques can be used to carry out cell compression experiments. The cell can be compressed at a micro level using Atomic force microcopy to provide more precise deformations. This will provide a basis for comparing experimental results with those obtained from the Finite Element Model.

References

1.   Buckwalter, J.A., T.A. Einhorn, and S.R. Simon, *Orthopaedic Basic Science*. 2nd Edition ed. 2000: American Academy of Orthopaedic Surgeons.

2.   Martin, J.A. and J.A. Buckwalter., *Aging, articular cartilage chondrocyte senescence and osteoarthritis.* Biogerontology, 2002. **3**: p. 257–264.

3.   Sandell, L.J. and T. Aigner, *Articular cartilage and changes in Arthritis: Cell biology of osteoarthritis.* Arthritis Res, 2001. **3**(2): p. 107-113.

4.   Trickey, T.R., M. Lee, and T. Guilak, *Viscoelastic properties of chondrocytes from normal and osteoarthritic human cartilage.* J Orthop Res, 2000. **18**(6): p. 891-8.

5.   M.Benkherourou, et al., *Standardization of a Method for charecterizing Low-Concentration Biogels:Elastic Properties of Low-Concentration Agarose Gels.* Journal of Biomechanical Engineering, 1999. **121**: p. 184-187.

6.   Survey, *Summary Health Statistics for U.S. Adults: National Health Interview Service 1999.* Vital and Health Statistics. **Series 10, Number 212**.

7.   Tim Hardingham, S.T. and, and A. Murdoch, *Tissue engineering: chondrocytes and cartilage.* Arthritis Res, 2002. **4(Suppl 3)**: p. S63-S68.

8.   Mukherjee, N., et al., *The enhancement of periosteal chondrogenesis in organ culture by dynamic fluid pressure.* J Orthop Res, 2001. **19**(4): p. 524-30.

9.   Bonassar, L.J., et al., *The effect of dynamic compression on the response of articular cartilage to insulin-like growth factor-I.* J Orthop Res, 2001. **19**(1): p. 11-7.

10.   Elder, S.H., et al., *Effect of compressive loading on chondrocyte differentiation in agarose cultures of chick limb-bud cells.* J Orthop Res, 2000. **18**(1): p. 78-86.

11.   Alberts, B., et al., *Molecular biology of the cell*. 3rd ed. 1994, New York and London: Garland Publishing, Inc.

12.   Yellowley, C.E., et al., *Effects of fluid flow on intracellular calcium in bovine articular chondrocytes.* Am J Physiol, 1997. **273**(1 Pt 1): p. C30-6.

13. Guilak, F., et al., *Mechanically induced calcium waves in articular chondrocytes are inhibited by gadolinium and amiloride.* J Orthop Res, 1999. **17**(3): p. 421-9.

14. Guilak, F., et al., *The deformation behavior and mechanical properties of chondrocytes in articular cartilage.* Osteoarthritis Cartilage, 1999. **7**(1): p. 59-70.

15. Burridge, K., et al., *Focal adhesions: transmembrane junctions between the extracellular matrix and the cytoskeleton.* Annu Rev Cell Biol, 1988. **4**: p. 487-525.

16. Bang, O.S., et al., *Association of focal adhesion kinase with fibronectin and paxillin is required for precartilage condensation of chick mesenchymal cells.* Biochem Biophys Res Commun, 2000. **278**(3): p. 522-9.

17. Guilak, F., *The deformation behavior and viscoelastic properties of chondrocytes in articular cartilage.* Biorheology, 2000. **37**(1-2): p. 27-44.

18. Kiviranta, I., et al., *Weight bearing controls glycosaminoglycan concentration and articular cartilage thickness in the knee joints of young beagle dogs.* Arthritis Rheum, 1987. **30**(7): p. 801-9.

19. Hochmuth, R.M. and N. Mohandas, *Uniaxial loading of the red-cell membrane.* J Biomech, 1972. **5**(5): p. 501-9.

20. Gittes, F., et al., *Flexural rigidity of microtubules and actin filaments measured from thermal fluctuations in shape.* J Cell Biol, 1993. **120**(4): p. 923-34.

21. Stamenovic, D. and M.F. Coughlin, *The role of prestress and architecture of the cytoskeleton and deformability of cytoskeletal filaments in mechanics of adherent cells: a quantitative analysis.* J Theor Biol, 1999. **201**(1): p. 63-74.

22. Ookawa, K., M. Sato, and N. Ohshima, *Morphological changes of endothelial cells after exposure to fluid- imposed shear stress: differential responses induced by extracellular matrices.* Biorheology, 1993. **30**(2): p. 131-40.

23. Smith, A.E., et al., *The mechanical properties of Saccharomyces cerevisiae.* Proc Natl Acad Sci U S A, 2000. **97**(18): p. 9871-4.

24.   Ingber, D.E., *Integrins, tensegrity, and mechanotransduction.* Gravit Space Biol Bull, 1997. **10**(2): p. 49-55.

25.   Guilak, F., G.R. Erickson, and H.P. Ting-Beall, *The effects of osmotic stress on the viscoelastic and physical properties of articular chondrocytes.* Biophys J, 2002. **82**(2): p. 720-7.

26.   Reiser, P.J., B.T. Stokes, and P.J. Walters, *Effects of immobilization on the isometric contractile properties of embryonic avian skeletal muscle.* Exp Neurol, 1988. **99**(1): p. 59-72.

27.   Zanetti, N.C. and M. Solursh, *Induction of chondrogenesis in limb mesenchymal cultures by disruption of the actin cytoskeleton.* J Cell Biol, 1984. **99**(1 Pt 1): p. 115-23.

28.   Guilak, F., et al., *Viscoelastic properties of intervertebral disc cells. Identification of two biomechanically distinct cell populations.* Spine, 1999. **24**(23): p. 2475-83.

29.   Guilak, F. and V.C. Mow, *The mechanical environment of the chondrocyte: a biphasic finite element model of cell-matrix interactions in articular cartilage.* J Biomech, 2000. **33**(12): p. 1663-73.

30.   Goodier, J.N., *Concentration of stress around spherical and cylindrical inclusions and flaws.* J Applied Mechanics, 1933. **55**: p. 39-44.

31.   Nussinovitch.A, Peleg.M, and Normand.M.D, *A modified and a Nonexponential Model for charecterization of the Stress relaxation of Agar and Alginate Gels.* Journal of Food Science, 1989. **54**(4): p. 1013-1016.

32.   MichelleA.LeRoux, Farshid Guilak, and Lori.A.Setton, *Compressive and shear properties of alginate gel:Effects of sodium ions and alginate concentration.* Journal of biomedical materials research, 1999. **47**(1): p. 46-53.

33.   Normand V, et al., *New insight into agarose gel mechanical properties.* Biomacromolecules, 2000. **1**(4): p. 730-8.

34.   Jin H and L. JL, *Determination of Poisson's ratio of articular cartilage by indentation using different-sized indenters.* J Biomech Eng, 2004. **126**(2): p. 138-45.

Appendix 1

Program To Calculate The Percentage Clustering Of Cells:

```c
/* This is a program to calculate distances between the given
points. It provides the nearest points and the average of distances
between 3 nearest points */
#include <stdio.h>
#include <math.h>
main()
{ float x[100],y[100],dist[100][100], x1,x2,y1,y2,min1[100],
            min2[100],min3[100],ave[100],totalave,sum,
            conv_fac,xc[100],yc[100];
  int n=0,i=0,k,j,l,flag1,flag2,i1=0;
  FILE *fp,*f1,*f2;
  conv_fac=594.00/28.00;
/* Factor to convert screen cm to microns (28 cm = 594 microns) */
  f1=fopen("points.txt","r");
  while(feof(f1)==0)
    {  fscanf(f1,"%f %f \n",&xc[i],&yc[i]);
       x[i]=conv_fac*xc[i];
       y[i]=conv_fac*yc[i];
       i=i+1;
       n=n+1;
    }
  fclose(f1);
/* Reading data from the file "points.txt" */
  for(i=0;i<n;i++)
    printf("Point is %f %f \n",x[i],y[i]);
/* Printing the points on the screen */
/* Calculating the first, second and third minimum distances between
the given points */
  for(i=0;i<n;i++)
    { min1[i]=10000;
      min2[i]=10000;
      min3[i]=10000;
      x1=x[i];
      y1=y[i];
      for(k=0;k<n;k++)
      { x2=x[k];
        y2=y[k];
        dist[i][k]=sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
        printf("distance bet point %d and point %d is
            %f\n",(i+1),(k+1), dist[i][k] );
        if((i!=k)&&(dist[i][k]<min1[i]))
          { min1[i]=dist[i][k];
            flag1=k;
          }
      }
      printf("min distance for point %d is %f \n",(i+1),min1[i]);
      if(n>2)
      { for(j=0;j<n;j++)
        { if((dist[i][j]<min2[i])&&(i!=j)&&(flag1!=j))
          { min2[i]=dist[i][j];
```

71

```c
                flag2=j;
              }
          }
       }
       printf("2nd min distance for point %d is %f
\n",(i+1),min2[i]);
       }
       if(n>3)
       { for(l=0;l<n;l++)
       { if((dist[i][l]<min3[i])&&(i!=l)&&(flag1!=l)&&(flag2!=l))
          min3[i]=dist[i][l];
       }
       printf("3rd min distance for point %d is %f
\n",(i+1),min3[i]);
       }
   }
/* The distance between a point and its nearest point is sent to the
file "minpoint.txt" */
  rewind(f2);
  f2=fopen("minpoint.txt","w");
  for(i1=0;i1<n;i1++)
    { fprintf(f2,"%f\n",min1[i1]);
      printf("Minimum distance of point %d is
%f\n",(i1+1),min1[i1]);
    }
  fclose(f2);
/* The average of distances between a point and its 3 nearest point
is sent to the file "avepoint.txt" */
  rewind(fp);
  fp=fopen("avepoint.txt","w");
  for(i=0;i<n;i++)
    { if(n>3)
      {  ave[i]=(min1[i]+min2[i]+min3[i])/3;
       sum=sum+ave[i];
       fprintf(fp,"%f\n",ave[i]);
      }
      if(n==3)
      {  ave[i]=(min1[i]+min2[i])/2;
       sum=sum+ave[i];
       fprintf(fp,"%f\n",ave[i]);
      }
      if(n==2)
      { ave[i]=min1[i];
      sum=sum+ave[i];
      fprintf(fp,"%f\n",ave[i]);
      }
      printf("Average of min distances of point %d is
%f\n",(i+1),ave[i]);
    }
  fclose(fp);
  totalave=sum/n;
/* totalave is the average of all the average distances */
  printf("Total average is %f",totalave);
}
```

H&E Staining Procedure:

Reagents	Time (Gel)
1. Xylene	2 min
2. Xylene	2 min
3. Xylene	2 min
4. 100% alcohol	1 min
5. 95% alcohol	1 min
6. 80% alcohol	1 min
7. 70% alcohol	1 min
8. Water Wash	1 min
9. Hematoxylin	1 min
10. Water Wash	Until Clear
11. Acid alcohol	5 dips
12. Water Rinse	2 min
13. Ammonia Water	5 dips
14. Water Rinse	5 min
15. Eosin	45 sec
16. 95% alcohol	5 dips
17. 95% alcohol	5 dips
18. 100% alcohol	1 min
19. 100% alcohol	1 min
20. Xylene	2 min
21. Xylene	2 min

Appendix 2

The elasticity solution for a spherical inclusion inside an infinite medium was found out by Goodier in 1933. Goodier, in his paper, stated that the stress intensity near the inclusion depends on the geometry but not the actual dimensions of the inclusion. Hence in order to analyze the behavior of a "spherical" inclusion inside an infinite medium an axisymmetric model was developed and subjected to a uniform compressive stress of 'T' N/mm^2. The medium was assumed to be elastic, isotropic and homogeneous.
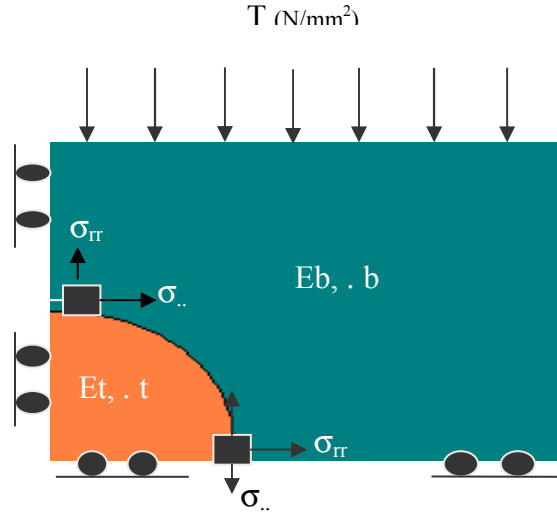
As can be observed from the figure, E_t, ν_t, μ_t are the Elastic modulus, Poisson's ratio, and shear modulus of the spherical inclusion and E_b, ν_b, μ_b are the respective properties of the medium. Radius of the spherical inclusion is 'a'. Goodier provided two sets of equations, one set to calculate the deformations and stresses at any point inside the medium and another set to calculate the stresses and deformations at any point inside the inclusion. But it can be observed that the deformations in both the cases will be equal at the boundary of the inclusion.

If U_r^b and $U_.^b$ are the radial and tangential deformations of medium

and U_r^t and $U_.^t$ are the radial and tangential deformations of inclusion

Then at r = a,

$$U_r^b = U_r^t \quad \text{and} \quad U_.^b = U_.^t$$

The following set of equations give the radial and tangential deformations at r = a,

$$U_r^b = -\frac{A}{r^2} - \frac{3B}{r^4} + \left(\frac{5-4._b}{1-2._b}\right)\frac{2C}{3r^3} + \left[\frac{-9B}{r^4} + \left(\frac{5-4._b}{1-2._b}\right)\frac{C}{r^2}\right]\cos 2. + \frac{Tr}{2E}\left[(1-._b) + (1+._b)\cos 2.\right]$$

$$U_.^b = -\left[\frac{6B}{r^4} + \frac{2C}{r^2}\right]\sin(2.) - \frac{Tr}{2E}(1+._b)\sin 2.$$

Where A, B and C are constants given by,

$$A = \frac{Ta^3[(-1-._b + 2._t)\propto_b + (1-2._b + ._t)\propto_t + 2._{b \cdot t}(\propto_b - \propto_t)]}{6\propto_b(1+._b)[(2-4._t)\propto_b + (1+._t)\propto_t]}$$

$$B = \frac{Ta^5(\propto_b - \propto_t)]}{8\propto_b[(7-5._b)\propto_b + (8-10._b)\propto_t]}$$

$$C = \frac{5Ta^3(1-2._b)(\propto_b - \propto_t)]}{8\propto_b[(7-5._b)\propto_b + (8-10._b)\propto_t]}$$

From the above equations, unknown properties of the inclusion can be calculated if the properties of medium and the deformations of inclusion are known.

Appendix 3

Experimentation Procedure for Determining Gel Stiffness (Macro Gels):

Agarose *(BRL/Gibco)* gels were prepared in different concentrations (1.2%, 2% and 3%). Testing was carried out on two types of agarose specimens – 3mm cubic gels and 4 mm cylindrical discs. Agarose powder was mixed with required amount of distilled water and heated in a microwave oven until the mixture just starts to boil. This mixture was then poured into custom-made molds (with two 60mm x 3mm x 3mm slots for the cubic gels, 3.62φ x 4mm length for the cylindrical discs) to produce the testing specimens of the required sizes. 3 mm cubic gels were measured for accuracy of dimensions using a Dial gage indicator (0.001 inches, *B.C.Ames Co, Waltham, Mass.*).

For precision, the dimension measurement for 3mm cubic gels was carried out under a microscope. The eyepiece of the microscope had two cross wires that are perpendicular to each other. One of the cross wires, was aligned with the edge of the gel and the dial gage reading was noted down. Then the base of the microscope was moved to align the cross wire with the opposite end of the gel. Again the reading of the dial gage indicator was noted down. The difference in two readings gives the actual gel dimension.

For example, in the case shown, the shaded area is the loading edge. First the dimensions of the loading area of the gel (h & w in the case shown) were calculated. Then the gel was flipped into the loading position and length 'l' of the gel was calculated. This procedure was repeated after the strain was applied on the gel, to obtain the deformations of the gel.
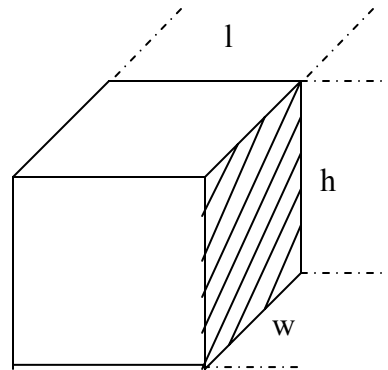
The gels were retrieved from the molds and each gel was then placed on a custom designed loading apparatus. Loading apparatus for the cubic gels (See Figure 39) consists of a microscopic slide at the bottom with a fixed and a movable aluminum platen. The gel was placed on the slide, in between the two platens, and was compressed to 25% strain by moving the aluminum platen using a micrometer. The compression was held for 10 min. Four successive compressions were applied on each cube. The loading apparatus for the cylindrical discs (See Figure 40) consists of a loading platen of 5mm diameter controlled by a stepper motor. The readings of both displacement (*DVRT Microstrain, Burlington VT*) and applied load (*load cell Entran, 0-20g*) were collected using Labtech software in a Gateway 2000 computer. Load/Time, Displacement/Time and Stress/Strain curves are obtained in Microsoft Excel.

Figure 39 The picture on the left shows the compression apparatus for 3mm cubic gels and that on the right shows the mold to generate the gels.
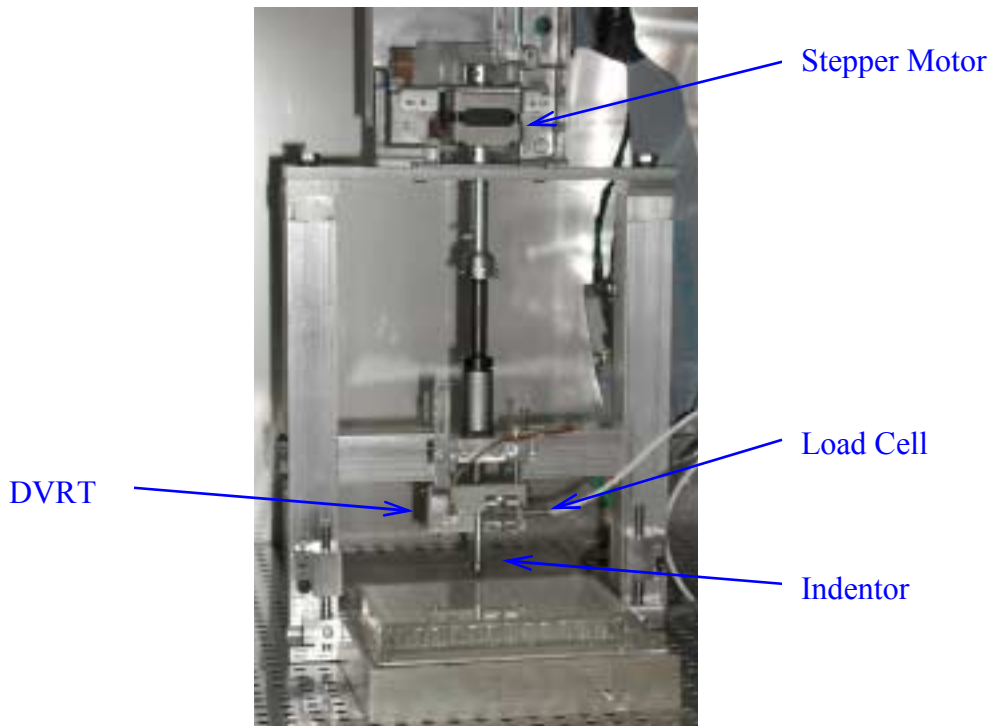
Figure 40 The picture shows the compression apparatus for the 4.5mm cylindrical discs

Appendix 4

Maple Program For Finding Ur Of Cell:

```
> restart;
> ha1:=(2-4*Pt)*Ub+(1+Pt)*Ut;
```
$$ha1 := (2 - 4\,Pt)\,Ub + (1 + Pt)\,Ut$$

```
> ha2:=(1-Pb)*(1-2*Pb);
```
$$ha2 := (1 - Pb)\,(1 - 2\,Pb)$$

```
> H:=T*ha2/(2*(1+Pb)*ha1);
```
$$H := \frac{1}{2}\,\frac{T(1 - Pb)(1 - 2\,Pb)}{(1 + Pb)((2 - 4\,Pt)\,Ub + (1 + Pt)\,Ut)}$$

```
> fa1:=4*((7-5*Pb)*Ub+(8-10*Pb)*Ut);
```
$$fa1 := 4\,(7 - 5\,Pb)\,Ub + 4\,(8 - 10\,Pb)\,Ut$$

```
> fa2:=5*(1-Pb);
```
$$fa2 := 5 - 5\,Pb$$

```
> F:=fa2*T/fa1;
```
$$F := \frac{(5 - 5\,Pb)\,T}{4\,(7 - 5\,Pb)\,Ub + 4\,(8 - 10\,Pb)\,Ut}$$

```
> Urt:=(H*r)+(F*r)+(3*F*r*cos(2*theta));
```
$$Urt := \frac{1}{2}\,\frac{T(1 - Pb)(1 - 2\,Pb)\,r}{(1 + Pb)((2 - 4\,Pt)\,Ub + (1 + Pt)\,Ut)} + \frac{(5 - 5\,Pb)\,T\,r}{4\,(7 - 5\,Pb)\,Ub + 4\,(8 - 10\,Pb)\,Ut}$$
$$+ \frac{3\,(5 - 5\,Pb)\,T\,r\cos(2 \,.\,)}{4\,(7 - 5\,Pb)\,Ub + 4\,(8 - 10\,Pb)\,Ut}$$

```
> Ut:=Et/(2*(1+Pt));
```
$$Ut := \frac{Et}{2 + 2\,Pt}$$

```
> Ub:=Eb/(2*(1+Pb));
```
$$Ub := \frac{Eb}{2 + 2\,Pb}$$

```
> Pb:=0.49; Pt:=0.49;
```
$$Pb := .49$$
$$Pt := .49$$

```
> Eb:=0.0297;
```
$$Eb := .0297$$

```
> T:=0.00484;
```
$$T := .00484$$

```
> r:=5.0436;
```
$$r := 5.0436$$

```
> evalf(H):
```

> **evalf(F):**
> **evalf(Urt);**

$$.00008355451170 \, \frac{1}{.0003986577181 + .5000000000 \, Et}$$

$$+ \frac{.06224811120}{.1813892618 + 4.161073824 \, Et} + \frac{.1867443336 \cos(2..)}{.1813892618 + 4.161073824 \, Et}$$

Appendix 5

Experimentation Procedure for Determining Gel Stiffness (Micro Gels):

Agarose *(BRL/Gibco)* gels were prepared in different concentrations (1.2%, 2% and 3%). Agarose powder was mixed with required amount of distilled water and heated in a microwave oven until the mixture just starts to boil. In order to carry out the compression experiments, a gel of 500 microns was created as shown in the picture. The mold consists of a micro slide mounted with a fixed brass piece and a movable indenter. The actuator is a dental wire of 500-micron square cross section. Load is applied with a stepper motor. A 1gm load cell measures the load. A cover slip is placed on top of the brass plates. Agarose mixture is then injected into the slot between the loading and fixed platens of the compression apparatus by using a hypodermic needle and allowed to harden.

Once the gel hardens, the needle is carefully removed without damaging the gel. The gel dimensions in x,y directions were measured using a micro pipette attached to standard micrometers. The gel was then compressed to 25% strain by a stepper motor. The compression was held for 10 min. Four successive compressions were applied on each cube. The readings of both displacement (*DVRT Microstrain, Burlington VT*) and applied load (*load cell Entran, 0-2g*) were collected using Labtech software in a Gateway 2000 computer. Load/Time, Displacement/Time and Stress/Strain curves are obtained in Microsoft Excel.
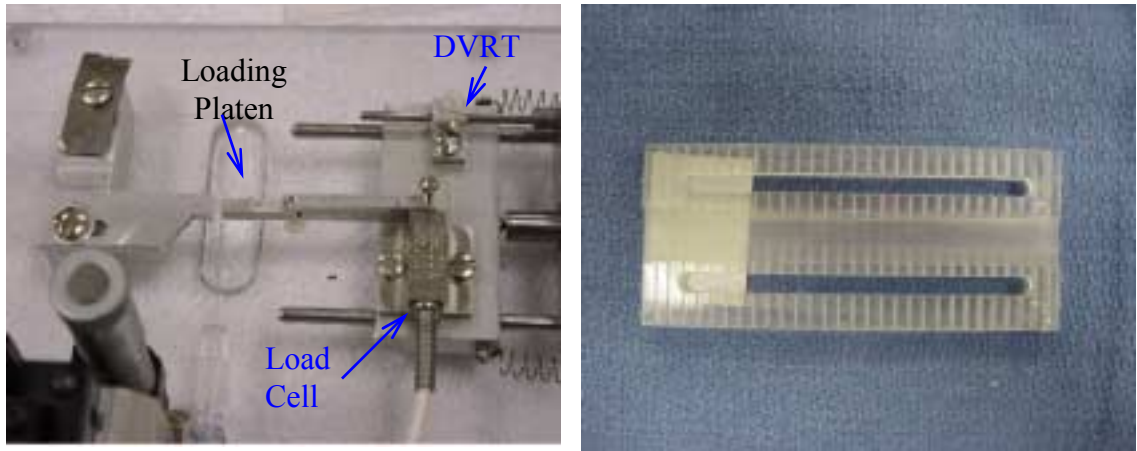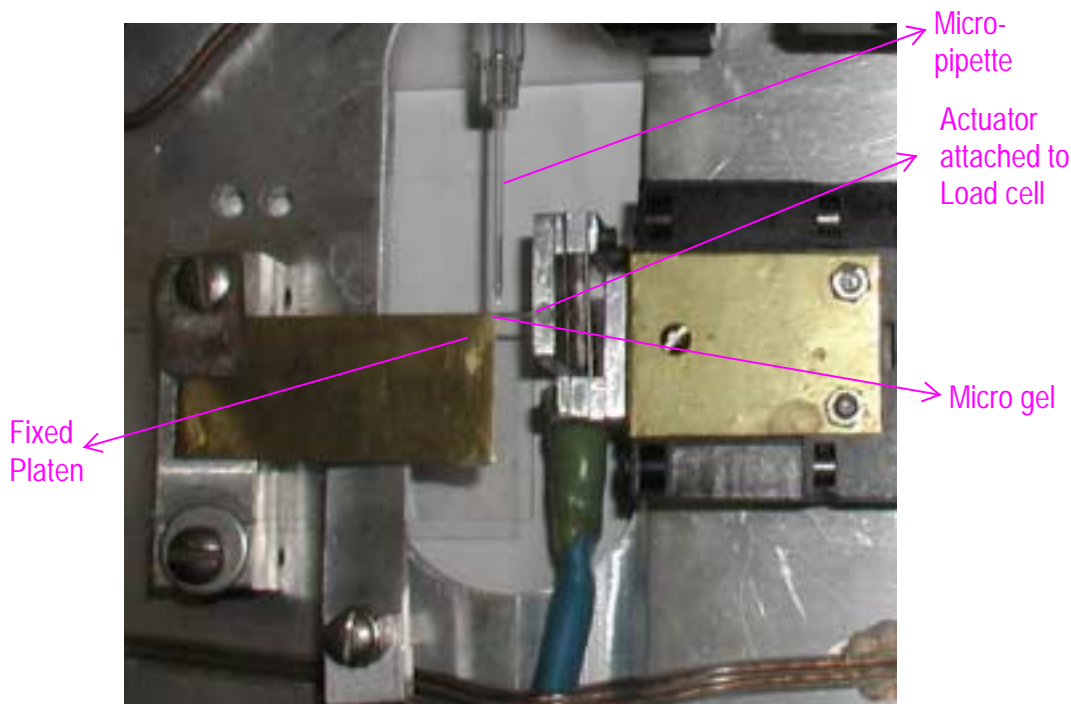

Figure 41 The picture shows the mold cum compression apparatus for the 0.5 micron cubic gels.

C Program For Loading/Unloading Using Stepper Motor:

A stepper motor operated the load cell, the following 'C' program was used for operating the stepper motor. An executable file was created after the program DSKDRV_3.C wascompiled and run in C. The input file for this code is a file "Command.dat" containing the input for the stepper motor.

```
/**
** Program DSKDRV_3.C
**
** Opens file "command.dat". Fetches each line in turn acting on the
** command.
**
** Courtesy: Towanda Malone, Morgan State University, August 10, '96
*/

#include <stdio.h>                                          /* 1 */
#include <dos.h>                                            /* 2 */
#include <sys\timeb.h>                                      /* 3 */
                                                            /* 4 */
#define CW 1                                                /* 6 */
#define CCW 0                                               /* 7 */
/* #define DATA 0x03bc */                                   /* 8 */
/* int data = 0x00; */                                      /* 9 */
unsigned char data = 0x00;                                  /* 10 */
short int DATA = 0x03bc;                                    /* 11 */
void Drive_dir (int dir);                                   /* 12 */
void Drive_step (int time);                                 /* 13 */
void turn_motor (int dir, int time, int duration);         /* 14 */
void pause_motor (int duration);                            /* 15 */
                                                            /* 16 */
void main (void)                                            /* 17 */
{                                                           /* 18 */
   FILE *fr;                                                /* 19 */
   int x, dir, duration, speed, time;                      /* 20 */
   char line[50], op;                                      /* 21 */
   fr = fopen( "a:\command.dat","rt");                     /* 22 */
   while(1)                                                 /* 23 */
   {                                                        /* 24 */
     fgets (line, 50, fr);                                  /* 25 */
     switch(toupper(line[0]))                               /* 26 */
     {                                                      /* 27 */
      case 'P':                                             /* 28 */
         sscanf (line, "%c %d", &op, &duration);            /* 29 */
         pause_motor(duration);                             /* 30 */
         break;                                             /* 31 */
                                                            /* 32 */
      case 'Q':                                             /* 33 */
         fcloseall();                                       /* 34 */
         exit(0);                                           /* 35 */
                                                            /* 36 */
      case 'T':                                             /* 37 */
         sscanf (line, "%c %d %d %d",                       /* 38 */
               &op, &dir, &speed, &duration);               /* 39 */
         turn_motor (dir, speed, duration);                 /* 40 */
         break;                                             /* 41 */
                                                            /* 42 */
      default:                                              /* 43 */
```

```
            printf(" Unknown Command\n");                    /* 44 */
            exit(0);                                         /* 45 */
      }                                                      /* 46 */
   }                                                         /* 47 */
}                                                            /* 48 */
                                                             /* 49 */
void turn_motor(int dir, int time, int duration)            /* 50 */
{                                                            /* 51 */
   struct timeb t_curr, t_start;                             /* 52 */
   int t_diff;                                               /* 53 */
                                                             /* 54 */
   ftime(&t_start);                                          /* 55 */
   Drive_dir( dir);                                          /* 56 */
   do                                                        /* 57 */
   {                                                         /* 58 */
      Drive_step(time);                                      /* 59 */
      ftime(&t_curr);                                        /* 60 */
      t_diff = (int) (1000.0 *(t_curr.time - t_start.time)   /* 61 */
            + (t_curr.millitm - t_start.millitm));           /* 62 */
   }                                                         /* 63 */
   while (t_diff <duration);                                 /* 64 */
}                                                            /* 65 */
                                                             /* 66 */
                                                             /* 67 */
void pause_motor (int duration)                              /* 68 */
{                                                            /* 69 */
   struct timeb t_curr, t_start;                             /* 70 */
   int t_diff;                                               /* 71 */
                                                             /* 72 */
   ftime(&t_start);                                          /* 73 */
   outportb (DATA, 0x00);                                    /* 74 */
   do                                                        /* 75 */
   {                                                         /* 76 */
      ftime(&t_curr);                                        /* 77 */
      t_diff = (int) (1000.0 * (t_curr.time - t_start.time)  /* 78 */
            + (t_curr.millitm - t_start.millitm));           /* 79 */
   }                                                         /* 80 */
   while (t_diff < duration);                                /* 81 */
}                                                            /* 82 */
                                                             /* 83 */
void Drive_dir(int dir)                                      /* 84 */
{                                                            /* 85 */
   if ( dir == CW)                                           /* 86 */
   {                                                         /* 87 */
      data = data|0x02;                                      /* 88 */
      outportb(DATA, data);                                  /* 89 */
   }                                                         /* 90 */
   else                                                      /* 91 */
   {                                                         /* 92 */
      data = data & (~0x02);                                 /* 93 */
      outportb(DATA, data);                                  /* 94 */
    }                                                        /* 95 */
}                                                            /* 96 */
                                                             /* 97 */
void Drive_step (int time)                                   /* 98 */
{                                                            /* 99 */
```

```
        data = data | 0x01;                                /* 100 */
outportb(DATA, data);                                      /* 101 */
        delay (time/2);                                    /* 102 */
        data = data ^ 0x01;                                /* 103 */
        outportb(DATA, data);                              /* 104 */
        delay (time/2);                                    /* 105 */
}                                                          /* 106 */
```

Sample input file:

A sample command.dat file is as follows,

T 1 1500 30000
P 5000
T 0 1500 30000
Q

The above code, applies load on the gel for 30 secs, pauses for 5 seconds, and removes the load over a period of 30 secs and quits.

Appendix 6

Experimentation Procedure For Culturing And Embedding Cells In 3D And Obtaining Cell Images Using LSM:

2% Agarose *(BRL/Gibco)* gels were prepared by mixing 2g of agarose powder in 100mL of sterile distilled water. The mixture was heated in a microwave oven until it just starts to boil (about 30 sec). Once the mixture reaches about 40^oC, 5 $\propto$L of ATDC5 cells were added to this mixture. If the temperature of the mixture is high ($>40^oC$), then the cells may die. This gel-cell mixture is then poured into custom-made molds (to form gels of size 1.2mm width, 0.5mm thickness) to produce the testing specimens of the required sizes. About 8 gel-cell constructs were prepared in a single attempt. The gels were retrieved from the molds and then placed in four 2 well glass slides *(Lab-Tek® II Chamber Slide™ system)* to form 4 samples (each of 2 gel-cell constructs). MEMS culture media was added to culture the cells in 3D. All this procedure is carried out in sterile conditions.

The 4 testing samples were stained, one each, on Day 1, Day 5, Day 8 and Day 12 respectively. The stained samples were observed under the confocal microscope to obtain the cytoskeletal structural variation in the cells, due to growth in 3D environment.
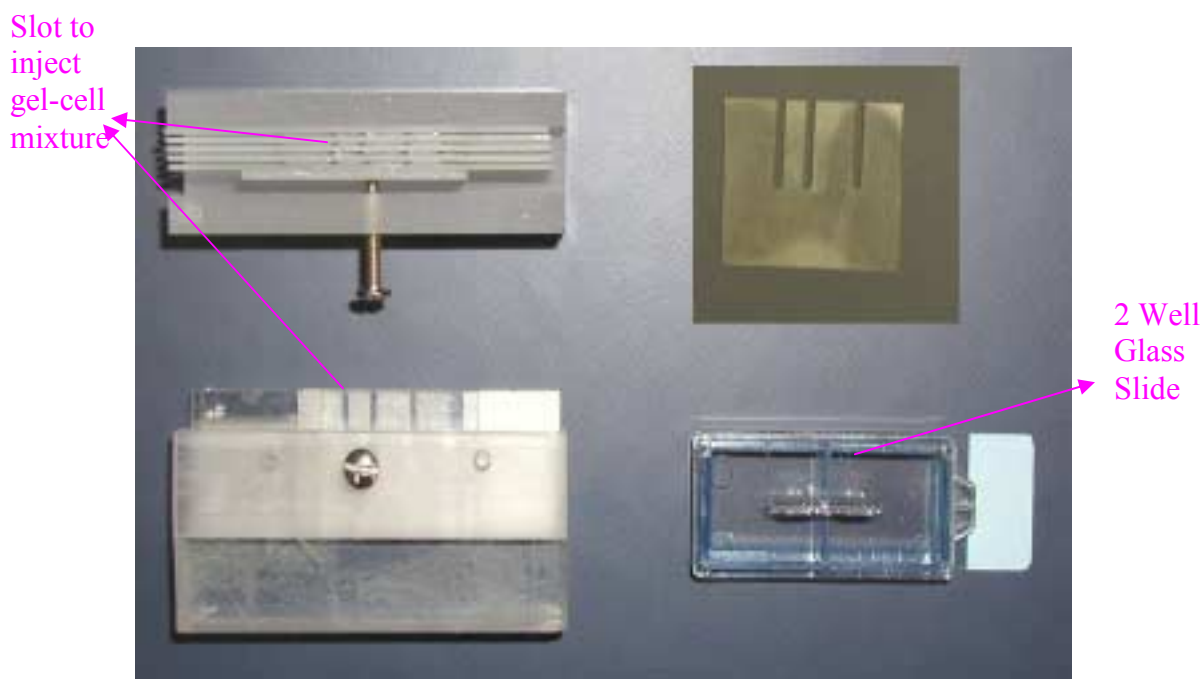
Figure 42 Top(Left top) and front(Left bottom and right top) views of the gel-making mold. Two well glass slides (right below)

Actin Staining of ATDC5 cells:

1. Wash Cell-Gel construct twice with prewarmed phosphate-buffered saline, pH 7.4 (PBS).
2. Fix the sample in 3.7% formaldehyde solution in PBS for 10 minutes at room temperature. (Methanol can disrupt actin during the fixation process. Therefore, it is best to avoid any methanol containing fixatives. The preferred fixative is methanol-free formaldehyde.)
3. Wash two or more times with PBS.
4. Place each Cell-Gel construct on a coverslip and extract it with a solution of 0.1% Triton® X-100 in PBS for 2 to 5 minutes.
5. Wash two or more times with PBS.
6. When staining with the Alexa Flour fluorescent phallotoxins, dilute 5 ∝L methanolic stock solution into 200 ∝L PBS for each coverslip to be stained. Pre-incubate fixed cells with PBS containing 1% BSA for 20 – 30 minutes prior to adding the phallotoxin staining solution. Do not rinse. When staining more than one coverslip, adjust volumes accordingly.
7. Place the staining solution on the coverslip for 20 minutes at room temperature (generally, any temperature between 4°C and 37°C is suitable).
8. Wash two or more times with PBS.

Add thin layer of mounting media to prevent photo bleaching.

Appendix 7

Program File for Displaying Cell Outline in I-DEAS:

   Enter outline points into a file "point.dat". Run the program file "format.f". Output will be an I-DEAS program file "format.prg" which when run in I-DEAS, plots the points of the cell outline.

   The following program "format.f" reads the outline point co-ordinates from the file "point.dat" and creates a macro "formats.prg" to display the points in I-DEAS

```
c  ****************************************************************
c                 format.f
c  ****************************************************************
       PROGRAM FORMAT
c  --------------------------------------------------------------------
c    program reads the point co-ordinates from a file "point.dat"
c    and creates a macro format.prg to plot the points
c  --------------------------------------------------------------------
       CHARACTER*80 FNAME
       FNAME='point'
       OPEN(UNIT=9,FILE='format.prg',STATUS='UNKNOWN')
       write(9,'(A)') 'K : $ return '
c  --------------------------------------------------------------------
c    I-DEAS > Master Modeler > Create 3D points
c  --------------------------------------------------------------------
       write(9,'(A)') 'K : $ $ $ /ta MM '
       write(9,'(A)') 'K : $ /de '
       write(9,'(A)') 'BP: 1 1 -9.866E-02 -4.545E-01 -11.6E8 0'
       write(9,'(A)') 'BR: 1 1 -9.866E-02 -4.545E-01 -11.6E8 256'
       write(9,'(A)') 'K : '
       write(9,'(A)') 'K : '
       write(9,'(A)') 'K : '
       write(9,'(A)') 'K : $ /cr m3 p '
       write(9,'(A)') 'K : KEY '
c  --------------------------------------------------------------------
c    reading co-ordinates from point.dat
c  --------------------------------------------------------------------
       open(unit=10, file='point.dat', status='old')
       do 100 i=1,10000
       read(10,*,iostat=nerr) r2,r3
       if (nerr .eq. 0) then
       write(9,1000) 'K :',r2,r3,0.0
       endif
 100   continue
       close(unit=10)
       write(9,'(A)') 'K : '
       write(9,'(A)') 'K : $ mpos :; /F PR E '
       write(9,'(A)') 'E : **** END OF SESSION **** '
 1000  FORMAT(A,E14.4,E14.4,E14.4)
       end
```

Appendix 8

FORTRAN Code For Creating Embedded Actin Filaments:

The program "truss.f" generates an ABAQUS input file "opmodel.inp" to create embedded 2D Truss elements in a cell model. Joining the points obtained from OPTIMAS images generates the cytoskeletal truss elements. Two consecutive points when joined represent the given line in the OPTIMAS image. The input file is "linecoords.dat" and the output file is "opmodel.inp".

```
c ************************************************************
c                          truss.f
c ************************************************************

        PROGRAM OPLINES

        INTEGER n,i,j,k
        REAL arxy(9990,4),dist(999,999)
        INTEGER arnn(9990)
        INTEGER node_num,n1,n2,n3,n4,elem_num
        REAL x1,y1
        INTEGER arrnn(999,999)

        EQUIVALENCE (arnn(1),arxy(1,1))

c ----------------------------------------------------------
c Reading points from the file "linecoords.dat" into an array
c ----------------------------------------------------------
        OPEN(UNIT=30,FILE="opmodel.inp",STATUS='unknown')
        WRITE(30,'(A)') '*NODE, SYSTEM=R'
        OPEN(UNIT=10,FILE="linecoords.dat",STATUS='old')
        DO 100 i=1,2000
          nn=i
          READ(10,*,END=1111) arnn(nn),arxy(i,2),arxy(i,3)
          IF (arnn(nn) .NE. 0) THEN
        WRITE(30,6000) arnn(nn)+100000,',',arxy(i,2),',',arxy(i,3),',',arxy(i,4)
          END IF
          n=n+1
100     CONTINUE
1111    GOTO 1112
1112    CLOSE(UNIT=10)
        WRITE(*,*) 'n= ',n

        WRITE(30,'(A)') '*ELEMENT,TYPE=T2D2, ELSET=TRUS'
c ----------------------------------------------------------
c Creating the output file "opmodel.inp" with embedded actin filaments
c ----------------------------------------------------------
        ielno=1
        DO 120 k=1,n-1
        IF(mod(k,2) .NE. 0) Then
          nnk=k
          n1=arnn(nnk)
```

```
              x1=arxy(k,2)
              y1=arxy(k,3)
              n2=arnn(nnk+1)
              x2=arxy(k+1,2)
              y2=arxy(k+1,3)
              WRITE(30,*) ielno+100000,',',n1+100000,',',n2+100000
              ielno=ielno+1
           ENDIF
120        CONTINUE
           WRITE(*,*) 'elem num is ',ielno
           elem_num=ielno
           WRITE(*,*) 'elem_num is ',elem_num
           WRITE(30,*) ' *EMBEDDED ELEMENT, HOST ELSET=CELL'
           WRITE(30,*) 'TRUS'
           WRITE(30,*) '*SOLID SECTION,ELSET=TRUS,MATERIAL=M0000001'
           WRITE(30,*) ' 5.0265E-5, 3'
           WRITE(30,*) '*MATERIAL,NAME=M0000001'
           WRITE(30,*) '*ELASTIC,TYPE=ISOTROPIC'
           WRITE(30,*) ' 0.363E-03,  4.900E-01'
           WRITE(30,*) '*DENSITY'
           WRITE(30,*) ' 7.820E-06,'
           WRITE(30,*) '*EXPANSION,TYPE=ISO,ZERO=21.85'
           WRITE(30,*) ' 1.170E-05,'
           WRITE(30,*) '*CONDUCTIVITY,TYPE=ISO'
           WRITE(30,*) ' 4.500E+04,'
           WRITE(30,*) '*NODE PRINT,FREQUENCY=        1,GLOBAL=YES'
           WRITE(30,*) 'U ,'
           WRITE(30,*) '*EL FILE,FREQUENCY= 1,POSITION=NODES,DIRECTIONS=YES'
           WRITE(30,*) 'S,E'
           WRITE(30,*) '*END STEP'
           CLOSE(UNIT=30)
6000       FORMAT(I,A,E14.4,A,E14.4,A,E14.4)
           END
```

The following program "solid.f" creates 4 noded CPS4 elements (Day 12) to embed
into the cell model. The input file is "solidcoords.dat" and the output file is
"somodel.inp".

```
c ****************************************************************
c                           solid.f
c ****************************************************************

      PROGRAM OPLINES

      INTEGER n,i,j,k
      REAL arxy(9990,4),dist(999,999)
      INTEGER arnn(9990)
      INTEGER node_num,n1,n2,n3,n4,elem_num
      REAL x1,y1
      INTEGER arrnn(999,999)

      EQUIVALENCE (arnn(1),arxy(1,1))

c ----------------------------------------------------------------
c Reading points from the file "solidcoords.dat" into an array
c ----------------------------------------------------------------
```

```fortran
        OPEN(UNIT=30,FILE="somodel.inp",STATUS='unknown')
        WRITE(30,'(A)') '*NODE, SYSTEM=R'
        OPEN(UNIT=10,FILE="solidcoords.dat",STATUS='old')
        DO 100 i=1,2000
          nn=i
          READ(10,*,END=1111) arnn(nn),arxy(i,2),arxy(i,3)
          n=n+1
100     CONTINUE
1111    GOTO 1112
1112    CLOSE(UNIT=10)
        WRITE(*,*) 'n= ',n
        DO 110 j=1,n
        IF(mod(j,2) .NE. 0) Then
          nnj=j
          WRITE(30,6000) arnn(nnj)+100000,',',arxy(j,2),',',arxy(j,3)
          WRITE(30,6000)
arnn(nnj)+200000,',',arxy(j+1,2),',',arxy(j,3)
          WRITE(30,6000)
arnn(nnj+1)+100000,',',arxy(j+1,2),',',arxy(j+1,3)
          WRITE(30,6000)
arnn(nnj+1)+200000,',',arxy(j,2),',',arxy(j+1,3)
        ENDIF
110     CONTINUE
        WRITE(30,'(A)') '*ELEMENT,TYPE=CPS4, ELSET=SOLID'
c ---------------------------------------------------------------------
c Creating the output file "opmodel.inp" with embedded actin
filaments
c ---------------------------------------------------------------------
        ielno=1
        DO 120 k=1,n
        IF(mod(k,2) .NE. 0) Then
          nnk=k
          n1=arnn(nnk)
          x1=arxy(k,2)
          y1=arxy(k,3)
          n2=arnn(nnk+1)
          x2=arxy(k+1,2)
          y2=arxy(k+1,3)
          n3=arnn(nnk)+200000
          x3=arxy(k+1,2)
          y3=arxy(k,3)
          n4=(arnn(nnk+1))+200000
          x4=arxy(k,2)
          y4=arxy(k+1,3)
          WRITE(30,*) ielno+100000,',',n1+100000,',',n4,',',
     $      n2+100000,',',n3
          ielno=ielno+1
        ENDIF
120     CONTINUE
        WRITE(*,*) 'elem num is ',ielno
        elem_num=ielno
        WRITE(*,*) 'elem_num is ',elem_num
        WRITE(30,*) '*EMBEDDED ELEMENT, HOST ELSET=CELL'
        WRITE(30,*) 'SOLID'
        WRITE(30,*) '*SHELL SECTION,ELSET=SOLID,MATERIAL=M0000001'
        WRITE(30,*) '1.000E-02  3'
        WRITE(30,*) '*MATERIAL,NAME=M0000001'
```

```
       WRITE(30,*) '*ELASTIC,TYPE=ISOTROPIC'
       WRITE(30,*) ' 0.363E-03,  4.900E-01'
       WRITE(30,*) '*DENSITY'
       WRITE(30,*) ' 7.820E-06,'
       WRITE(30,*) '*EXPANSION,TYPE=ISO,ZERO=21.85'
       WRITE(30,*) ' 1.170E-05,'
       WRITE(30,*) '*CONDUCTIVITY,TYPE=ISO'
       WRITE(30,*) ' 4.500E+04,'
       WRITE(30,*) '*NODE PRINT,FREQUENCY=        1,GLOBAL=YES'
       WRITE(30,*) 'U ,'
       WRITE(30,*) '*EL FILE,FREQUENCY=
1,POSITION=NODES,DIRECTIONS=YES'
       WRITE(30,*) 'S,E'
       WRITE(30,*) '*END STEP'
       CLOSE(UNIT=30)
6000   FORMAT(I,A,E14.4,A,E14.4,A,E14.4)
       END
```

Appendix 9

Fortran Code For Calculating Center Of The Cell:

This program calculates the centroid of a given irregular closed area (cell in our case). An approximate value of the centroid should be entered, from which the actual centroid is calculated.

```
c *****************************************************************
c                 Centcirc.f
c *****************************************************************

        PROGRAM CENTROID

        INTEGER n,i,j,k
        REAL arxy(9990,4),dist(999,999),area(9990)
        INTEGER arnn(9990)
        INTEGER node_num,n1,n2,n3,n4,elem_num
        REAL x1,y1,x2,y2,xi,yi,xcent(9990),ycent(9990)
        INTEGER arrnn(999,999)

        EQUIVALENCE (arnn(1),arxy(1,1))

c ----------------------------------------------------------------
c Reading points from the file "ornode.dat" into an array
c ----------------------------------------------------------------

        OPEN(UNIT=10,FILE="ornode.dat",STATUS='old')
        n=0
        DO 100 i=1,2000
          nn=i
          READ(10,*,END=1111) arnn(nn),arxy(i,2),arxy(i,3),arxy(i,4)
          WRITE(*,6000) arnn(nn),',',arxy(i,2),',',arxy(i,3),',',
                                      arxy(i,4)
          n=n+1
100     CONTINUE
1111    GOTO 1112
1112    CLOSE(UNIT=10)
        WRITE(*,*) 'n= ',n
        WRITE(*,*) 'Enter the approximate centroid(Xi,Yi): '
        READ(*,*) xi,yi
        WRITE(*,*) xi,yi
        cc=1
        aa=1
        sigma_xcent=0.0
        sigma_ycent=0.0
        sigma_area=0.0
        DO 120 k=1,n
          nnk=k
          n1=arnn(nnk)
          x1=arxy(k,2)
          y1=arxy(k,3)
          IF (k .NE. n) THEN
```

```fortran
          n2=arnn(nnk+1)
          x2=arxy(k+1,2)
          y2=arxy(k+1,3)
        ELSE IF (k .EQ. n) THEN
          n2=arnn(1)
          x2=arxy(1,2)
          y2=arxy(1,3)
        ENDIF
        xcent(cc)=(x1+x2+xi)/3
        sigma_xcent=sigma_xcent+xcent(cc)
        ycent(cc)=(y1+y2+yi)/3
        sigma_ycent=sigma_ycent+ycent(cc)
        WRITE(*,*) xcent(cc),ycent(cc)
        area(aa)=(((x2-x1)*(yi-y1))-((xi-x1)*(y2-y1)))/2
        sigma_area=sigma_area+area(aa)
        WRITE(*,*) area(aa)
        cc=cc+1
        aa=aa+1
120   CONTINUE
      centroidx=0.0
      centroidy=0.0
      DO 130 j=1,aa
       centroidx=centroidx+((xcent(j)*area(j))/sigma_area)
       centroidy=centroidy+((ycent(j)*area(j))/sigma_area)
130   CONTINUE
      WRITE(*,*) 'CENTROID IS ',centroidx,',',centroidy
6000  FORMAT(I,A,E14.4,A,E14.4,A,E14.4)
      END
```

Appendix 10

ABAQUS Subroutine To Extract Nodes Of A Given Node Set From An Input File:

Program to extract the number of undeformed nodal coordinates of a specified node set OUTLINE from the input file specified, and send them to an output file "nsetno.dat". If the name of the input file and the number of nodes in the node set are specified, then this program extracts the node numbers of the node set OUTLINE.

```
c ************************************************************
c                 exdenset.f
c ************************************************************

        SUBROUTINE HKSMAIN

        INCLUDE 'aba_param.inc'
        CHARACTER*80 FNAME
        DIMENSION ARRAY(513),JRRAY(NPRECD,513),RRAY(5,5),LRUNIT(2,1)
        DIMENSION NSRAY(2000)
        integer c,i

        EQUIVALENCE  (ARRAY(1),JRRAY(1,1))

        WRITE(6,*) 'Enter the name of the file (w/o extension) :'
        READ(5,'(A)') FNAME
        WRITE (*,*) 'Read file name to be: ',FNAME
        WRITE(*,*) 'Enter the number of nodes in nset OUTLINE:'
        READ(*,*) nsetn
        WRITE (*,*) 'Read no. of nodes to be: ',nsetn
        DO 200 j=1,2000
        NSRAY(j)=0
 200    CONTINUE
c----------------------------------------------------------------
c       Creating the file "nsetno.dat"
c----------------------------------------------------------------
        OPEN(UNIT=9,FILE='nsetno.dat',STATUS='UNKNOWN')
c
        NRU=1
        LRUNIT(1,1)=8
        LRUNIT(2,1)=2
        LOUTF=0
        CALL INITPF(FNAME,NRU,LRUNIT,LOUTF)
        JUNIT=8
        CALL DBRNU(JUNIT)
c----------------------------------------------------------------
c       to extract number of nodes in the node set OUTLINE
c----------------------------------------------------------------
        c=2
        i=5
        DO 100 K1=1,9999999
           CALL DBFILE(0,ARRAY,JRCD)
```

```fortran
        IF (JRCD .NE. 0) GO TO 110
        KEY=JRRAY(1,2)
c

        IF (KEY .EQ. 1931) THEN
           IF (ARRAY(3) .EQ. 'OUTLINE') THEN
              NSRAY(1)=JRRAY(1,4)
              WRITE(9,*) NSRAY(1)
              DO 120 i=5,nsetn+3
                NSRAY(c)=JRRAY(1,i)
                WRITE(9,*) NSRAY(c)
                c=c+1
 120  CONTINUE
           END IF
        END IF

 100    CONTINUE
 110    CONTINUE
      CLOSE(UNIT=9)
      END
```

Appendix 11

ABAQUS Subroutine To Extract Nodal Deformations Of A Given Node Set:

Program to extract the undeformed and deformed nodal coordinates of the node set OUTLINE from the specified input file and send them to output files "ornode.dat" and "denode.dat" respectively. The program "exdenset.f" should be run before running this code.

```
c *************************************************************
c                    exdeno.f
c *************************************************************

        SUBROUTINE HKSMAIN

        INCLUDE 'aba_param.inc'
        CHARACTER*80 FNAME
        CHARACTER*8 ELSET
        DIMENSION ARRAY(513),JRRAY(NPRECD,513),RRAY(5,5),LRUNIT(2,1)
        DIMENSION INRAY(2000,4),IDEF1(2000,4)
        DIMENSION NSRAY(2000)
        DOUBLE PRECISION RNRAY(2000,4),RDEF1(2000,4)

        EQUIVALENCE  (RNRAY(1,1),INRAY(1,1))
        EQUIVALENCE  (RDEF1(1,1),IDEF1(1,1))
        EQUIVALENCE  (ARRAY(1),JRRAY(1,1))

        WRITE(6,*) 'Enter the name of the file (w/o extension) :'
        READ(5,'(A)') FNAME
        WRITE (*,*) 'Read file name to be: ',FNAME
c
        OPEN(UNIT=9,FILE='ornode.dat',STATUS='UNKNOWN')
        OPEN(UNIT=10,FILE='denode.dat',STATUS='UNKNOWN')
        OPEN(UNIT=11,FILE='nsetno.dat',STATUS='OLD')
c
        NRU=1
        LRUNIT(1,1)=8
        LRUNIT(2,1)=2
        LOUTF=0
        CALL INITPF(FNAME,NRU,LRUNIT,LOUTF)
        JUNIT=8
        CALL DBRNU(JUNIT)
c ------------------------------------------------------------------
c to extract number of nodes in the node set OUTLINE, reading from
c the file "nsetno.dat"
c ------------------------------------------------------------------
        c=2
        i=5
        DO 200 i=1,9999
          READ(11,*,END=1111) NSRAY(i)
          WRITE(*,*) NSRAY(i)
        nn=nn+1
 200   CONTINUE
 1111  GOTO 2222
```

```fortran
 2222  CLOSE(UNIT=11)
       DO 100 K1=1,99999
99
          CALL DBFILE(0,ARRAY,JRCD)
          IF (JRCD .NE. 0) GO TO 210
          KEY=JRRAY(1,2)
        INO=1
c -------------------------------------------------------------------
c to extract nodal undeformed coordinates and store them in an array
c and the file "ornode.dat"
c -------------------------------------------------------------------
              IF (KEY .EQ. 1901) THEN
              flag=0
                  DO 220 j=1,nn
                    IF (JRRAY(1,3) .EQ. NSRAY(j)) THEN
                    flag=1
                    cou=cou+1
                    END IF
 220  CONTINUE
                    IF (flag .EQ. 1) THEN
                  INRAY(INO,1)=JRRAY(1,3)
                  RNRAY(INO,2)=ARRAY(4)
                  IF ((RNRAY(INO,2) .GT. -1E-05) .AND.
     $            (RNRAY(INO,2) .LT. 1E-05) ) THEN
                    RNRAY(INO,2)=0.0
                  END IF
c
                  RNRAY(INO,3)=ARRAY(5)
                  IF ((RNRAY(INO,3) .GT. -1E-05) .AND.
     $            (RNRAY(INO,3) .LT. 1E-05) ) THEN
                    RNRAY(INO,3)=0.0
                  END IF
c
                  RNRAY(INO,4)=ARRAY(6)
                  IF ((RNRAY(INO,4) .GT. -1E-05) .AND.
     $            (RNRAY(INO,4) .LT. 1E-05) ) THEN
                    RNRAY(INO,4)=0.0
                  END IF
                  WRITE(9,1000) JRRAY(1,3), RNRAY(INO,2),
     $            RNRAY(INO,3), RNRAY(INO,4)
                  INO=INO+1
                  END IF
                END IF

c -------------------------------------------------------------------
c to extract nodal deformed co-ordinates and store them in an array
c and the file "denode.dat"
c -------------------------------------------------------------------
           IDNO=1
             IF (KEY .EQ. 101) THEN
             flag1=0
                 DO 320 k=1,nn
                   IF (JRRAY(1,3) .EQ. NSRAY(k)) THEN
                   flag1=1
                   END IF
 320  CONTINUE
           IF (flag1 .EQ. 1) THEN
```

```fortran
                  IDEF1(IDNO,1)=JRRAY(1,3)
                  RDEF1(IDNO,2)=ARRAY(4)
                  IF ((RDEF1(IDNO,2) .GT. -1E-05) .AND.
     $                (RDEF1(IDNO,2) .LT. 1E-05) ) THEN
                     RDEF1(IDNO,2)=0.0
                  END IF
C
                  RDEF1(IDNO,3)=ARRAY(5)
                  IF ((RDEF1(IDNO,3) .GT. -1E-05) .AND.
     $                (RDEF1(IDNO,3) .LT. 1E-05) ) THEN
                     RDEF1(IDNO,3)=0.0
                  END IF
C
                  RDEF1(IDNO,4)=ARRAY(6)
                  IF ((RDEF1(IDNO,4) .GT. -1E-05) .AND.
     $                (RDEF1(IDNO,4) .LT. 1E-05) ) THEN
                     RDEF1(IDNO,4)=0.0
                  END IF
                  WRITE(10,1000) IDEF1(IDNO,1), RDEF1(IDNO,2),
     $                RDEF1(IDNO,3), RDEF1(IDNO,4)
                  IDNO=IDNO+1
               END IF
            END IF
C
C ----------------------------------------------------------------------
C        to print the number of nodes and elements on screen
C ----------------------------------------------------------------------
         IF (KEY .EQ. 1921) THEN
            WRITE(*,*) 'NO OF ELEMENTS IS ',JRRAY(1,7)
            IELNO=JRRAY(1,7)
            WRITE(*,*) 'NO OF NODES IS ', JRRAY(1,8)
            INNO=JRRAY(1,8)
         END IF
 100     CONTINUE
 210   CONTINUE
C
 1000  FORMAT(I,E14.4,E14.4,E14.4)
       CLOSE(UNIT=9)

       END
```