ESTIMATION OF NONLINEAR MECHANICAL PROPERTIES OF ATHEROSCLEROTIC PLAQUES

by

Ting F. Zhu

B.E. Engineering Mechanics
Tsinghua University, 2003

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN MECHANICAL ENGINEERING
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

SEPTEMBER 2005

Signature of Author ....
Department of Mechanical Engineering
August 20, 2005

Certified by ....
Dr. Roger D. Kamm, Professor
Department of Mechanical Engineering and Biological Engineering Division
Thesis Supervisor

Certified by ....
Dr. Mohammad R. Kaazempur Mofrad, Assistant Professor
Department of Bioengineering, University of California Berkeley
Thesis Supervisor

Accepted by ....
Dr. Lallit Anand, Professor
Department of Mechanical Engineering
Chairman, Department Committee on Graduate Students

Estimation of Nonlinear Mechanical Properties of Atherosclerotic Plaques

Ting F. Zhu

Submitted to the Department of Mechanical Engineering
on August 19, 2005 in partial fulfillment of the
Requirements for the Degree of Master of Science in
Mechanical Engineering

ABSTRACT

A numerical method has been developed to estimate the mechanical properties of atherosclerotic plaques by combining genetic algorithm with finite element methods. Plaque images derived from optical coherence tomography were employed to construct finite element models which were subsequently used in conjunction with a genetic algorithm to determine the parameters in a nonlinear constitutive model. A new multi-frame scheme is introduced to better perform the estimation on a nonlinear mechanical model and reduce the effects of noise. Results show while it is feasible to estimate the nonlinear mechanical properties of plaque, the accuracy can depend on various factors, especially the noise.

KEY WORDS: FEM, atherosclerotic plaques, parameter estimation, Mooney-Rivlin model, optical coherence tomography, image noise

Thesis Supervisor: Dr. Roger D. Kamm
Title: Professor of Mechanical Engineering and Biological Engineering

Thesis Supervisor: Dr. Mohammad R. Kaazempur-Mofrad
Title: Assistant Professor of Bioengineering

Acknowledgements

First I would like to acknowledge my thesis advisor Professor Roger D. Kamm and Professor Mohammad R. Kaazempur-Mofrad for sharing with me their expertise and their encouragement during my hardest time.

To my friend and colleague, Mo, who generously shared his research results with me and helped me with my work. Also many thanks to Alyx and Ray who shared her experiment data and presented me their interesting works.

Finally thank you Mom, Dad and Grandmothers. Your love and support provided me courage in the completion of my degree.

This research is funded by National Institutes of Health (grant 5-R01-HL70039).

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Atherosclerosis Pathology and Morphology

Atherosclerosis is the major cause of morbidity and mortality in industrialized countries. Major studies have been devoted to the understanding of the pathophysiological processes leading to this disease by attempting to relate it to mechanical, biochemical, and genetic factors.

The current state of understanding about atherosclerosis has been developed in four stages[1]. During the early days of study, atherosclerosis was considered a process of aging: when people get old, their artery hardens and therefore atherosclerosis takes place. A later theory, 'the lipid hypothesis', considers genetic factors and high cholesterol the main reasons to develop the atherosclerotic lesions. With the recognition of growth factors, 'the response-to-injury hypothesis' was introduced, which explains the vascular response to the initial lipid damages.

Representing the latest understanding of the disease is 'the inflammation hypothesis'. Inflammatory stimuli, e.g. oxidized low-density lipoprotein (LDL), can induce the production of adhesion molecules[2], which will further activate the circulating mononuclear cells via chemokine activation. These mononuclear cells will initiate a firm adhesion to the vascular walls via various adhesion molecules, such as ICAM-1 and VCAM-1 (Fig. 1)[1].

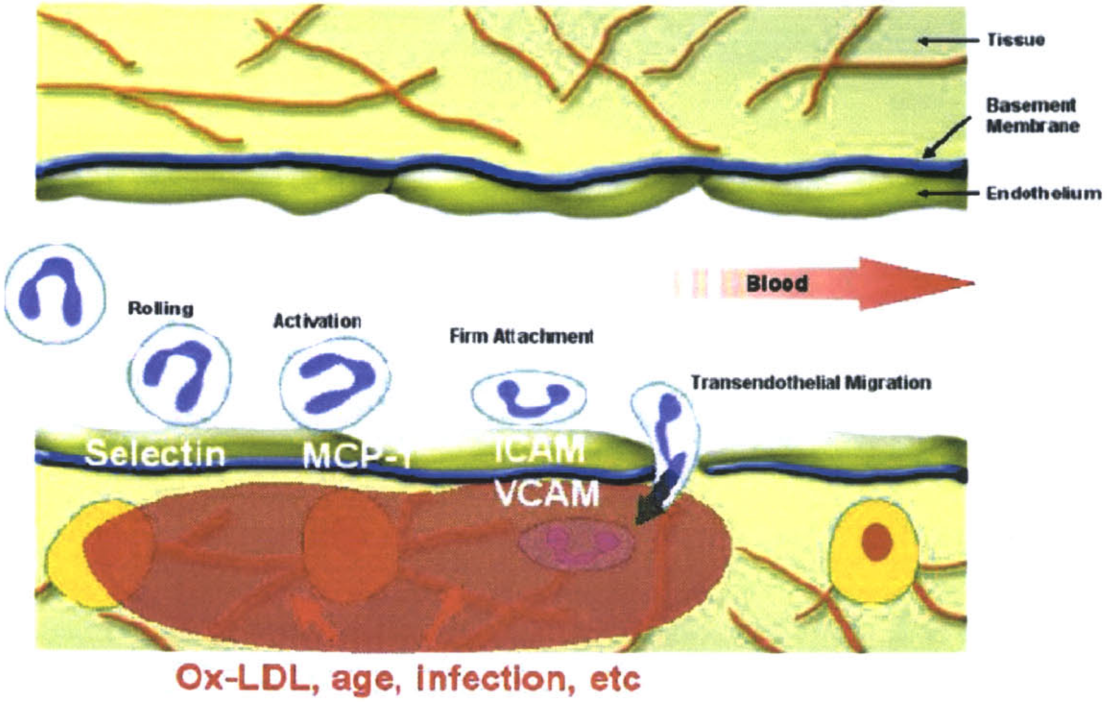


Fig. 1.1 Schematic illustration of the inflammation hypothesis [1].

These mononuclear cells then migrate through the junction of the endothelial cells and enter the vascular tissue. They will further absorb lipid substances and lead to the formation of foam cells, and therefore a lipid lesion. Smooth muscle cells, simultaneously migrate and localize to the intima as a step in the repair process. They eventually become a fibrous cap, coving the lipid region. These thin fibrous caps are subject to a risk of rupturing under certain conditions. Plaque rupture can cause advanced diseases like thrombosis and heart attack that may bring server consequences.

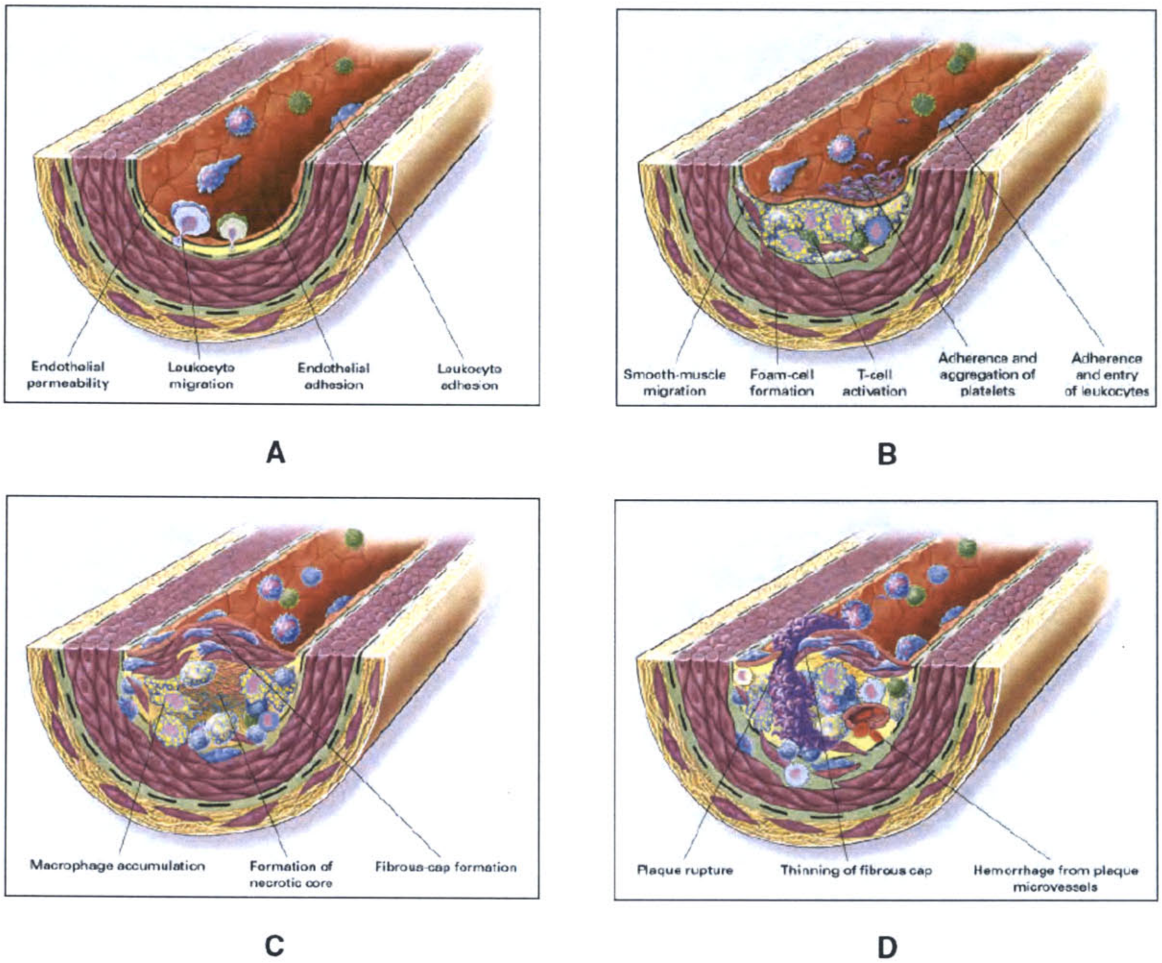


Fig. 1.2 Atherogenesis morphological progression. A. Mononuclear cells migrate. B. Fatty streak formation. C. Progression to intermediate and advanced disease. D. Fibrous cap formation [3].

1.2 Mechanical Factors in Atherosclerosis

Mechanical factors have long been suggested as contributors to the initiation and development of the desease [4]. Recent studies have uncovered the relationship between particular mechanical stress distributions and the risk of plaque rupture [5, 6]. To understand the underlying mechanism of this correlation and to help better analyze the nature of the disease, and eventually develop diagnostic methods for assessing the risk of a specific plaque to rupture, detailed information about the plaque geometry, load and boundary conditions and the mechanical properties of the vessel wall and plaque tissue is required.

While the plaque geometry can be obtained by advanced imaging techniques, e.g. intravascular ultrasound (IVUS) [7], optical coherence tomography (OCT) [8] and high resolution magnetic resonance imaging (MRI) [9], and similarly the corresponding boundary conditions can be reasonably well described, few data are available on the mechanical properties of plaque tissue, determination of which is crucial for detailed mechanical analysis of the plaque [10]. Furthermore, due to patient-to-patient variability in plaque composition and structure, acquiring patient-specific mechanical properties remains a key step in the analysis of plaque vulnerability. The physical characteristics of plaque tissue make it relatively difficult to directly measure the mechanical properties *ex vivo* [10]. Numerical methods have therefore been used to estimate plaque's mechanical properties non-invasively, by relating the strain field in a pressure-inflated vessel wall, derived through vascular elastography [7], to the finite element models constructed with prescribed mechanical properties, thereby optimizing the unknown distribution of mechanical properties that provide best agreement between the computational data and elastography.

Many numerical methods have been developed to estimate the mechanical property distribution using linear models. The calculus-based techniques [11, 12], commonly used to solve such problems, are typically complicated to implement and computationally expensive. Further, due to the need for direct inversion of the finite element matrix, such methods are not trivially applicable to nonlinear elastic models. Yet, the stress-strain constitutive laws of biomaterials are usually far more complex than isotropic-linear models. Vessel tissue constituents differ in the nature of their behavior and mechanical properties; for instance, collagen tissue usually behaves linearly, while elastin is nonlinear. Neglect of the nonlinearity of the tissue mechanical properties can hence result in substantial errors in the stress distribution. Although considerable research has been devoted to implementation of nonlinear mechanical properties [13], often the corresponding parameters can not be accurately determined [14]. A noninvasive method to estimate the nonlinear mechanical properties is therefore valuable for detailed mechanical analysis of arterial plaques. Compared to linear elastic models [15], the overall problem is complicated in nonlinear material models when the number of unknown parameters for each material exceeds one (Young's modulus or shear modulus for linear elastic model), for example to two (D1 and D2) in the Mooney-Rivlin model. One important issue that needs to be addressed in parameter estimation problems is the uniqueness of the solution and is discussed in detail in Chapter 3.

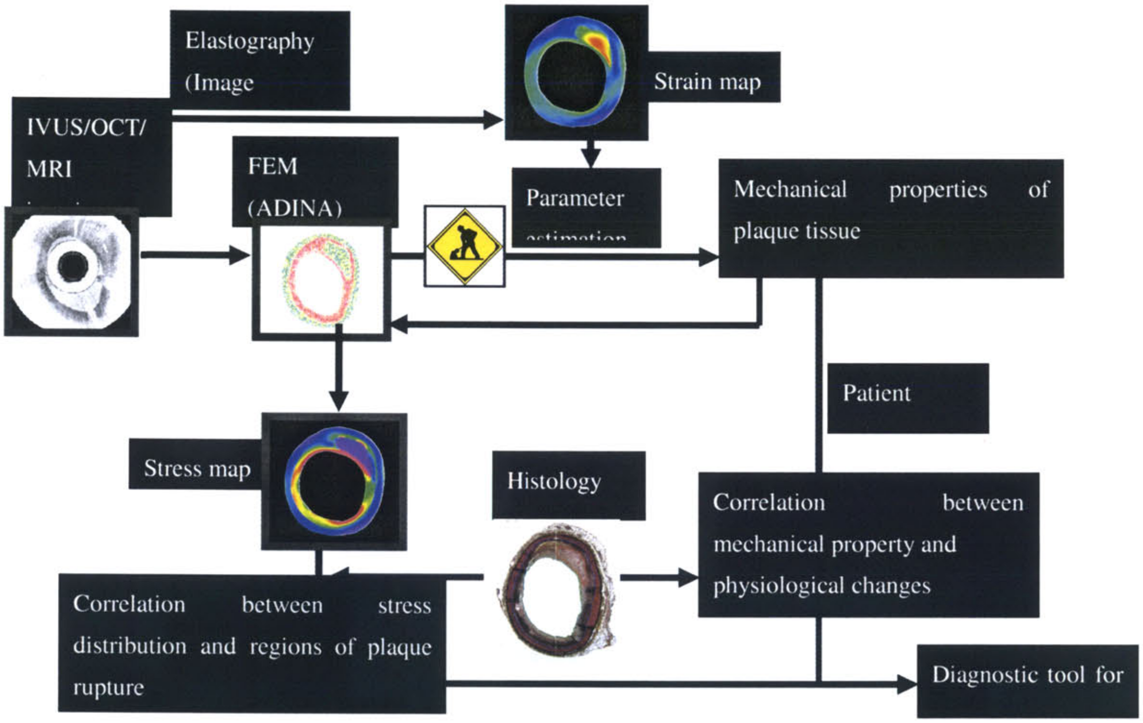

Elastography
(Image
IVUS/OCT/
MRI
Strain map
FEM
(ADINA)
Parameter
Mechanical properties of plaque tissue
Patient
Stress map
Histology
Correlation between mechanical property and physiological changes
Correlation between stress distribution and regions of plaque rupture
Diagnostic tool for

Fig. 1.3 Overall flow chart of the research.

1.2 Thesis Goals

Recent work [16] has been conducted to estimate the mechanical properties in 2D using a lumped parameter model and genetic algorithm that dramatically enhances the efficiency and flexibility of the estimation method, and without necessarily directly inverting the finite element matrix system. In this work, we extend our combined genetic/finite element algorithm to incorporate the nonlinear Mooney-Rivlin model for parameter estimation using patient-specific 2D plaque geometries. The uniqueness of solution, as well as the effect of noise, are discussed using a simple model, while introducing a multi-frame scheme (i.e. utilizing strain maps under at least two different pressure loads). Finally, an idealized 3D vessel geometry is employed to demonstrate the viability of the present nonlinear parameter estimation algorithm in 3D.

Chapter 2
Arterial Image Acquisition

The acquisition of arterial image is the first step in this research. Images of the atherosclerotic artery can provide the boundaries of the vascular components, i.e., the normal arterial wall, the fibrous cap and the lipid pool, which are used in the FEM modeling. Another important information that can be extracted is the deformation of the artery under the variation of lumenal blood pressure, that is the displacement or strain map of the artery under certain pressure change. Most of the contents presented in this chapter is adapted from the work of a previous graduate student in our lab, Alexandra Chau, on the OCT-based arterial elastography [17].

2.1 Optical Coherence Tomography (OCT) imaging

Optical coherence tomography (OCT) is the optical analog to time-of-flight B-mode ultrasound (which detects acoustic signal). OCT provides high-resolution cross-sectional images of human tissue [18, 19]. A beam of near infrared light is split into two, one sent into the sample and one used as the reference beam. Optical interferometry is used to measure back-reflections from tissue samples. Tissue structure can be detected in the depth or axial direction by varying the optical pathlength of the reference arm and in the lateral direction by rotating the sample beam circumferentially.

The major advantage of using OCT as an imaging modality is in its relatively higher spatial resolution (axial resolutions of 10μm and lateral resolutions of 25μm). This feature can substantially decrease the noise in the electrograph, which as characterized in the following chapters is a major factor limiting the estimation ability of the algorithm. The shortcomings of the OCT modality are: 1) the depth of imaging is limited in OCT, as a result the vessel used can not be too thick (usually within a diameter of 1mm) [17]; 2) it is an intravascular and therefore invasive imaging technique, limiting its clinical applications.

2.2 Intravascular Ultrasound (IVUS)

IVUS is currently the most widely used arterial imaging technique in clinical settings. It can acquire real-time cross-sectional images of coronary arteries *in vivo* [20]. Like OCT it is an invasive imaging technique, with relatively lower image resolutions, typically a high-frequency ultrasound (30-40 MHz) provides axial resolutions of 100μm and lateral of 200μm. Yet, it can be used to identify tissue components, namely lipid pool, fibrous cap, calcified region, etc.. IVUS has much larger penetration depth than OCT, usually of 4-10mm in diameter.

2.3 Magnetic Resonance Imaging (MRI)

Magnetic Resonance Imaging is an important non-invasive version of angiography, The improvements in high-resolution MRI will provide an opportunity to use MRI instead of OCT to acquire arterial images [9]. Researches have demonstrated that MRI is capable of determining atherosclerotic plaque components [21-23]. Although up to today the resolution of MRI is far from capable of elastography, the algorithm we are developing is generic and can be used when one day high-resolution MRI is available.

2.4 Post-processing of arterial images

For this research, the post-processing procedure includes the identification of arterial components and elastography. To identify the arterial components, i.e. segmentation, is an important step in this research and therefore was carried out in coordination with experienced physicians. In the following we briefly show the general rules of identifying different arterial components.

The fibrous cap region usually appears homogeneous, signal rich (see Fig. 2.1).

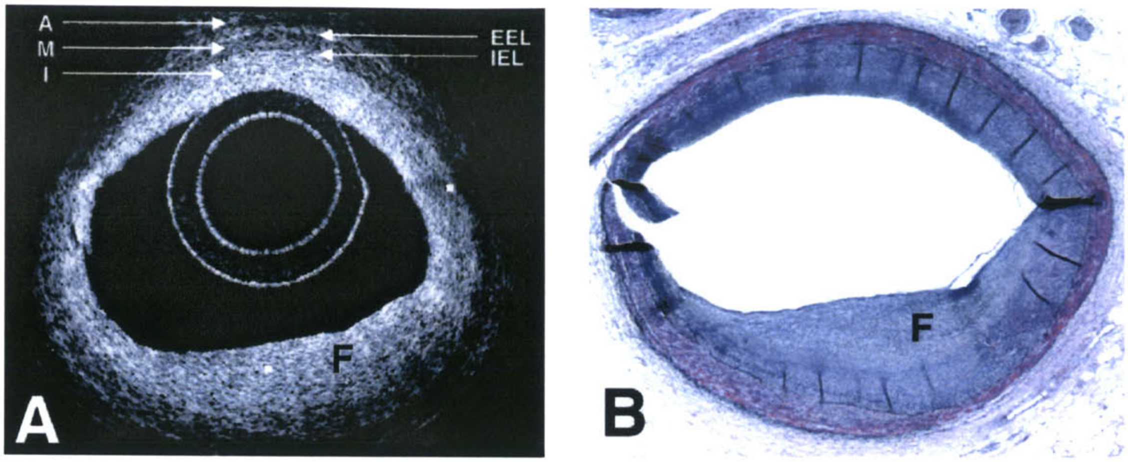


Fig. 2.1 OCT image (A) compared to histology image (B). F stands for fibrous cap region[8].

The calcified region usually show poor signal and with distinct borders (see Fig. 2.2).

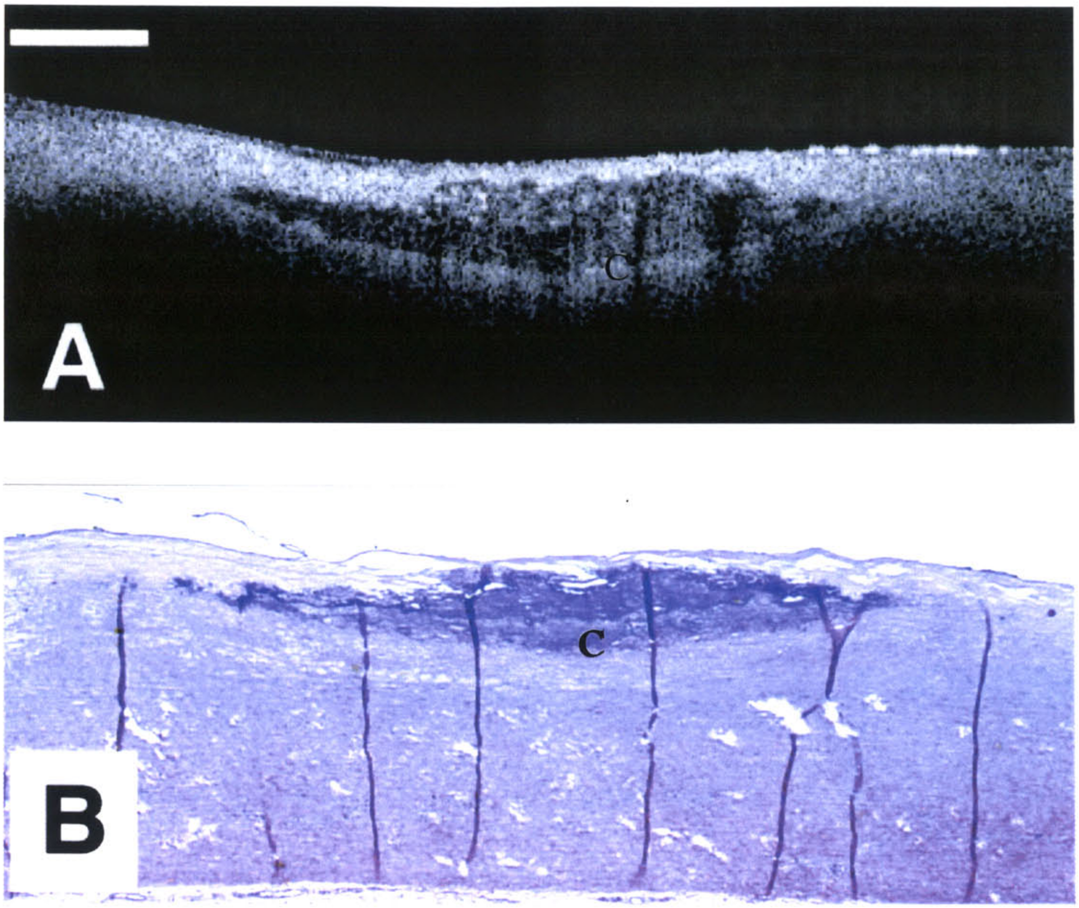


Fig. 2.2 OCT image (A) compared to histology image (B). C stands for calcified region[8].

The lipid pool usually appears signal poor regions with diffuse borders covered by a signal rich band, that is the fibrous cap (see Fig. 2.3).

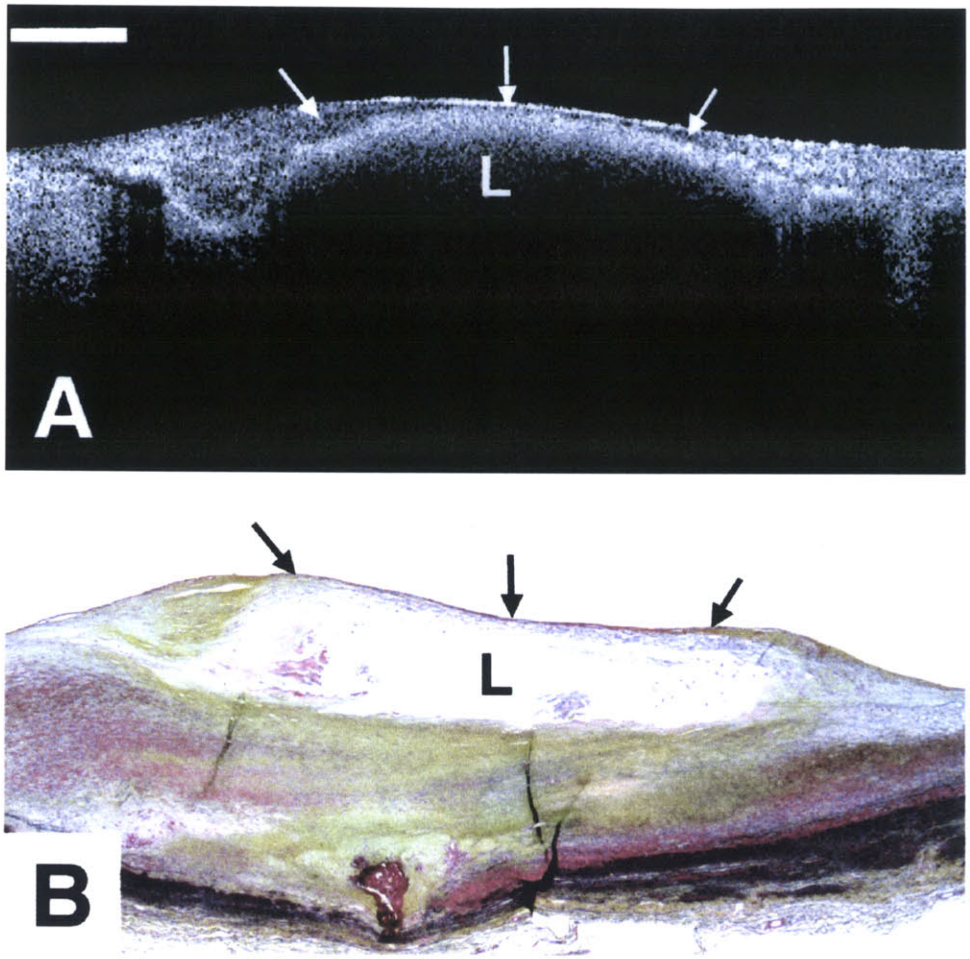


Fig. 2.3 OCT image (A) compared to histology image (B). L stands for lipid pool[8].

By these criteria, we can identify the region components in the atherosclerotic plaque. For instance, the segmentation of a lipid-rich plaque and a calcification-rich plaque were shown in the following (see Fig. 2.4).

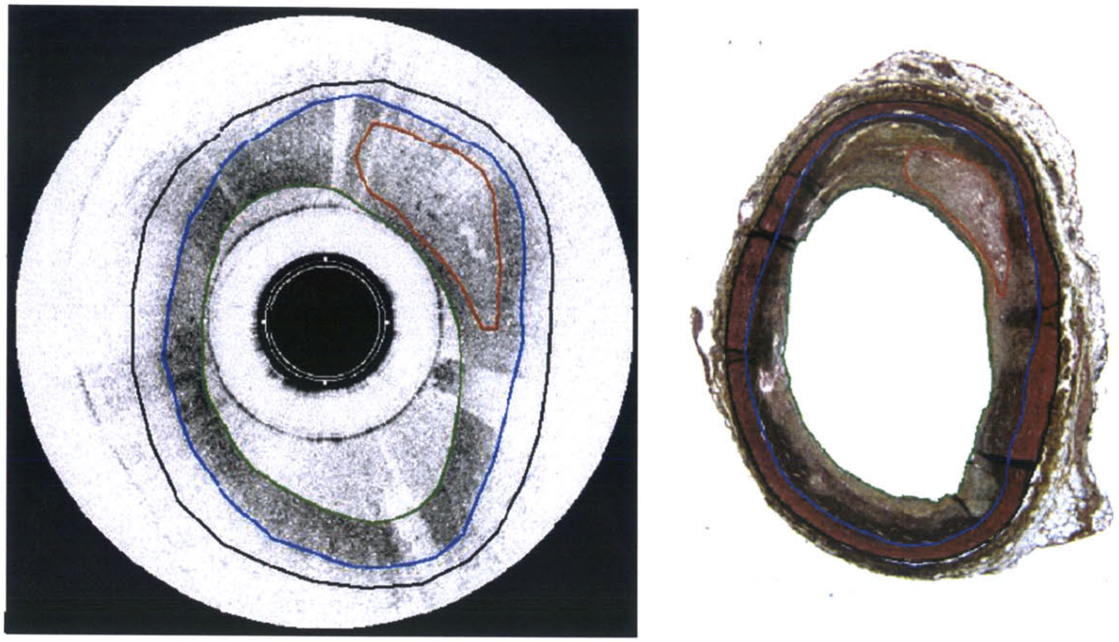


Fig. 2.4 Lipid-rich plaque segmentation (left) compared to histological images (right). The regions in red, blue and black contours are lipid pool, fibrous cap, and normal arterial wall, respectively[8].

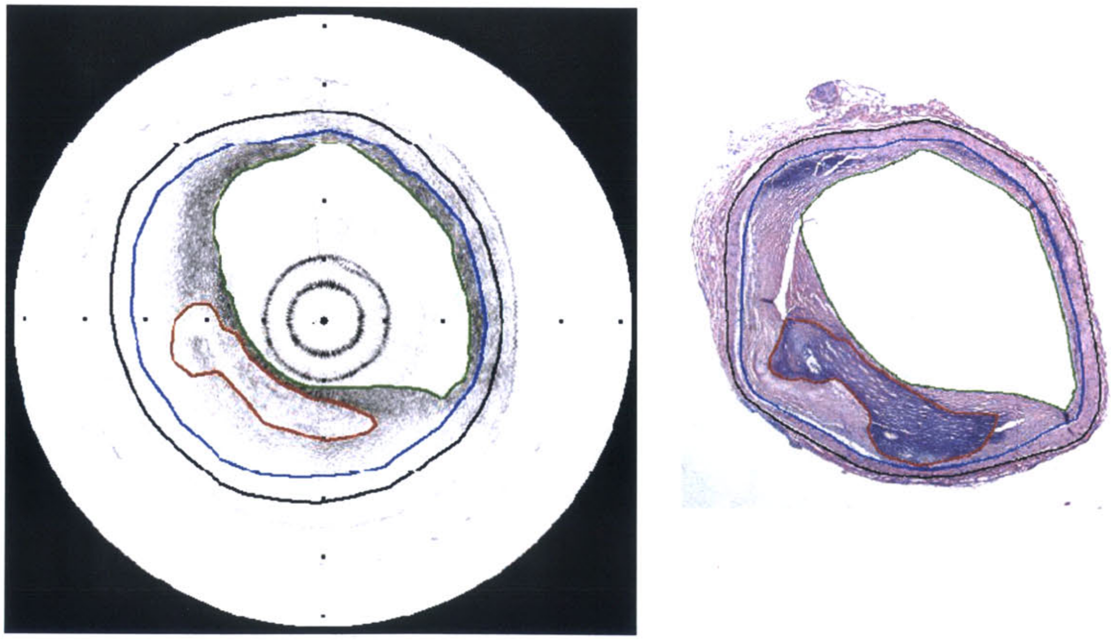


Fig. 2.5 Calcification-rich plaque segmentation (left) compared to histological images (right). The regions in red, blue and black contours are calcification, fibrous cap, and normal arterial wall respectively.

Palpation has been used by physicians to probe deep tissue for centuries. Elastography was proposed to provide a more quantitative and reliable means of assessing tissue elasticity [24-26] The whole process is an analog of palpation: first, the tissue is compressed/stretched, then imaging techniques, e.g. ultrasound, is used to capture the displacing specimen, then the images under different pressure/stretch are processed via cross-correlation techniques and give us the displacements. The displacement field can give us a strain map that can used to quantify properties of the tissue. De Korte et al. [20] applied this idea in estimating intravascular elasticity. The variation of blood pressure provides a natural mechanical excitation and IVUS was used to capture the arterial motion. Other mechanical excitation approaches, include dynamic loading, as opposed to static, can also be used[27].

Chapter 3

Parameter Estimation with Multi-frame Scheme

3.1 General Scheme of the Parameter Estimation

Generally speaking, an estimation method is comprised of the definition of fitness function and an iteration scheme. To look for the solution of a problem, one usually has to compare a certain number of possible solutions. A fitness function is used to evaluate the possible solutions, to determine how "fit" they are or how close the solution is to the real one. Usually the fitness function is a function with single input (the possible solution) and gives back a number that determines the fitness. In the current problem, the fitness function is derived from the difference between the measured and predicted effective strains

$$e = \sqrt{e_{xx}^2 + e_{yy}^2 + \tfrac{1}{2} e_{yz}^2}$$

Summed over all elements, the smaller the summed difference the more likely the corresponding parameters fit the true values. In practice, all strains are placed in a long vector, and the norm of the difference between the predicted and true strain vectors is the fitness value. An iteration scheme is designed to further bring the best fit parameter(s) to the next iteration.

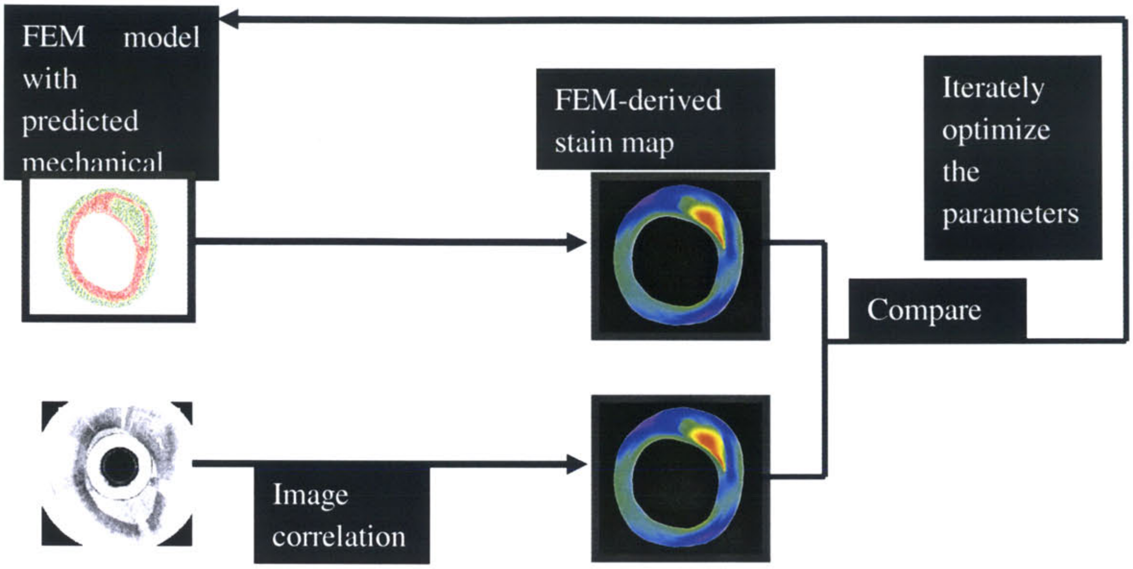

FEM model with predicted mechanical
FEM-derived stain map
Iterately optimize the parameters
Compare
Image correlation

Fig. 3.1 General scheme of parameter estimation and its applications.

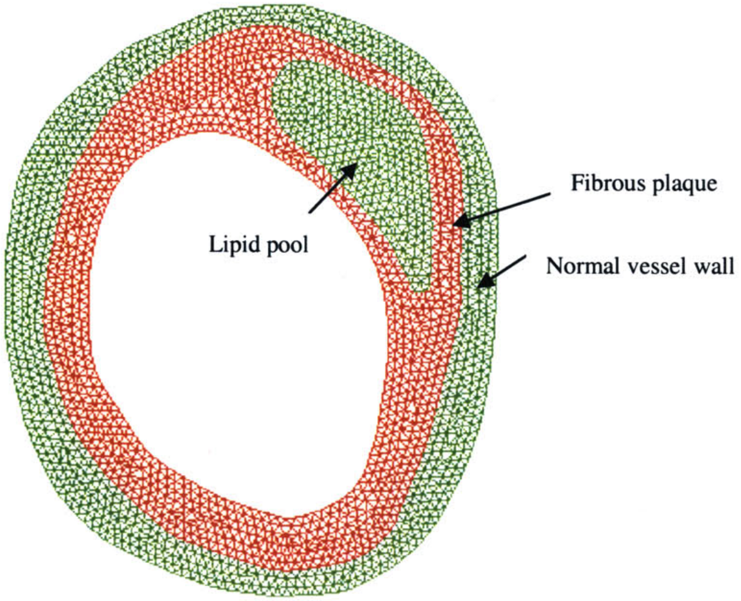
In this part of the study, for simplicity, we used the random exhaustive search, just for characterizing the multi-frame scheme. A genetic algorithm scheme is introduced in the next chapter, which was proven to have higher efficiency than the random exhaustive search method.

3.2 Finite element analysis

Finite element models, both in 2D and 3D, were employed to test the viability of the estimation algorithm. 2D images of excised lipid-laden arteries were obtained through optical coherence tomography (OCT) [28]. Excised coronary arteries were collected from autopsies and stored in PBS at 4ºC until imaging occurred, within 72 hours. The specimen was place on a scaffold [28] and 0 pressure was applied to the inner lumen of the vessel (relaxed). OCT provided cross-sectional images of the entire length of the vessel segments. Digital images were processed, imported into an FEM package, ADINA (Watertown, MA), and used to construct finite element models (see Fig. 3.2 A). Specifically, 9-node 2D plain strain elements were utilized to mesh the model geometry, at a sufficient mesh density based on grid convergence studies.

A 3D plaque model consisting of a cylindrical arterial segment fixed on both ends, with a crescent-shaped fibrous plaque and a sphere-like lipid pool was also constructed (Fig. 3.2 B).

A



B

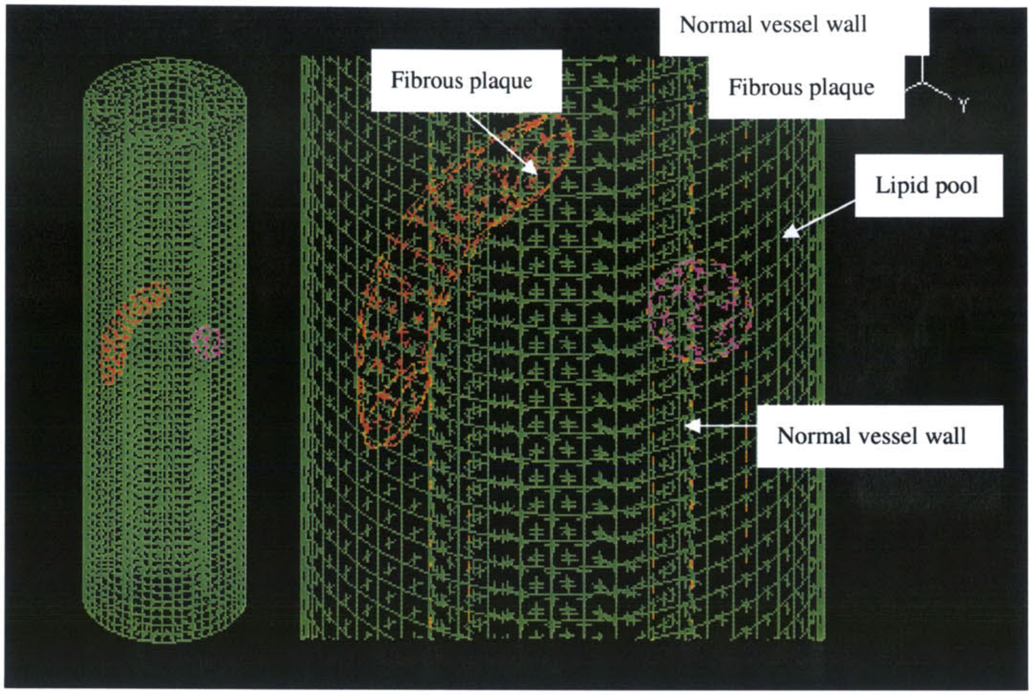


Fig. 3.2 A) 2D Geometry of OCT-derived atherosclerotic vessel segmentation, meshed in ADINA B) Finite element mesh of a 3D idealized artery segment with a fibrous plaque and a lipid pool intra-plaque features.

A pressure load was applied on the vessel lumen, increasing from 0 to 16 kPa (120 mmHg) in 24 timesteps for both 2D and 3D cases. Mixed interpolation formulation was applied.

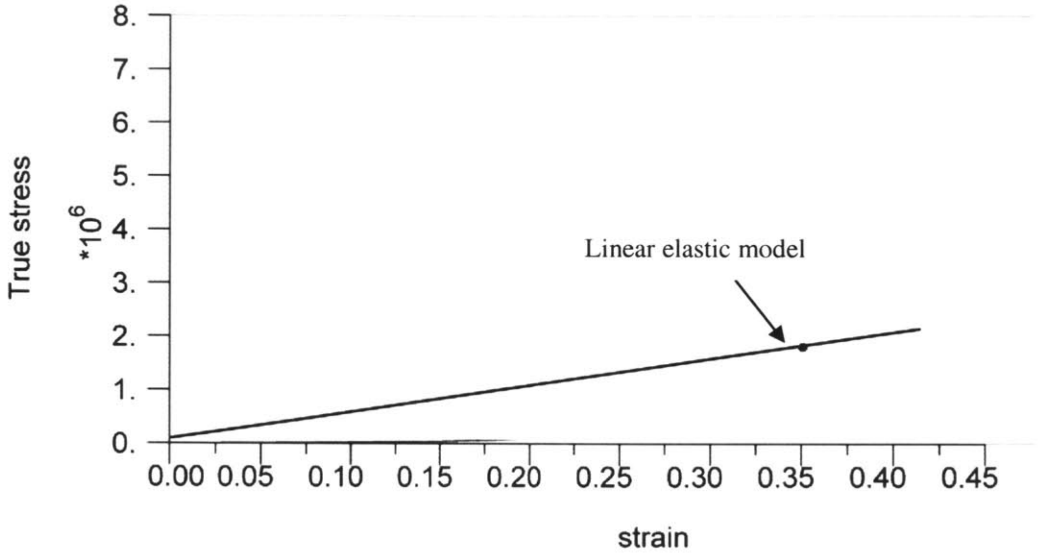
The Mooney-Rivlin model was used to estimate the mechanical properties [29, 30] of the corresponding regions in the FEM model, namely normal vessel wall, fibrous plaque, and lipid. The Mooney-Rivlin model is defined by the strain energy density function $W = D_1\left(e^{D_2(I_1-3)} - 1\right)$ where W is the strain energy density, D_1 and D_2 are material constants, and I_1 is the first invariant of the Cauchy-Green deformation tensor. The product D_1D_2 is proportional to the elastic modulus of the material, while D_2 is related to its strain-stiffening behavior. The values for D_1 and D_2 were taken from previous literature[31] (see Table 3.1). A typical Mooney-Rivlin stress-strain curve corresponding to the fibrous plaque tissue is shown in Fig. 3A.

Strain fields calculated at each time step were utilized as fictitious elastography data in our current characterization study, which in practice will be obtained experimentally.

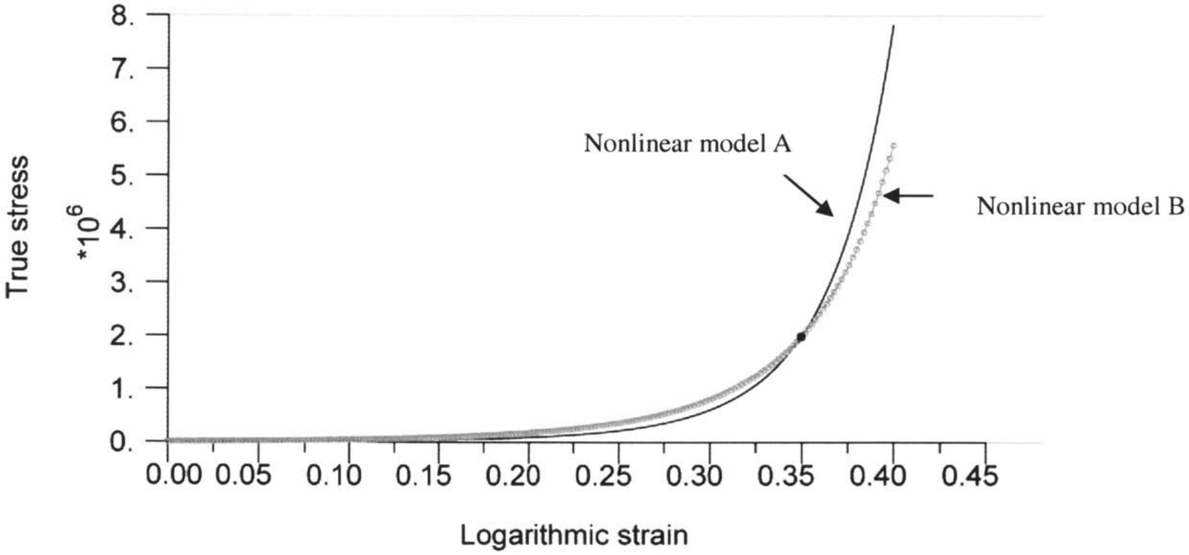
3.3 Parameter estimation: Multi-frame scheme

A multi-frame scheme is introduced here to facilitate the nonlinear parameter estimation. One important issue that needs to be addressed in parameter estimation problems is the uniqueness of the solution. Compared to linear elastic models [15], the overall problem is complicated in nonlinear material models where the number of unknown parameters for each material exceeds one. For instance, consider a 1D problem, e.g. a cantilever under stretch force load at one end, with a single homogeneous linear elastic material of unknown stiffness. Knowing the strain under a given force, one can easily determine the Young's modulus of elasticity for the material (Fig. 3.3 A). However, if the material's constitutive law is nonlinear, for instance Mooney-Rivlin model defined by D_1 and D_2 parameters, there would potentially exist numerous combinations of D_1 and D_2 that can fit the strain distribution under a given load. That is, the solution is not unique (see Fig. 3.3 B). For a Mooney-Rivlin model, a minimum of two strain/load configurations ('two frames') is required to uniquely capture the stress-strain curve (see Fig. 3.3 C). Moreover, the result of estimation is expected to be sensitive to the underlying noise and uncertainty in elastography procedure, both in the measured strain and/or pressure load (see Fig. 3.3 C). One remedy is to obtain multiple frames of elastography data at incremental pressure loads, and incorporate more available data to the parameter estimation algorithm (see for example Fig. 3.3 D, where 12 frames with noisy measurements are used). By fitting the curve to a number of linearly independent points, we expect to obtain an optimized solution. The comparison between single-frame and multi-frame schemes will be discussed in the following sections, using results from our algorithm. Although in real cases it can be far more complicated than we discussed above: when more than one element is used, different elements may have to bear different strain, even if a single load is applied. In a real problem, if a single-frame method is used for a non-linear problem, the algorithm tends to optimize the most influential parameter only. In general, from the authors' experience (see results below), the analysis above provides a general guideline how the algorithm can perform given the number of frame used.

A


True stress
*10^6
8.
7.
6.
5.
4.
3.
2.
1.
0.
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
strain
Linear elastic model

B


True stress
*10^6
8.
7.
6.
5.
4.
3.
2.
1.
0.
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
Logarithmic strain
Nonlinear model A
Nonlinear model B

C

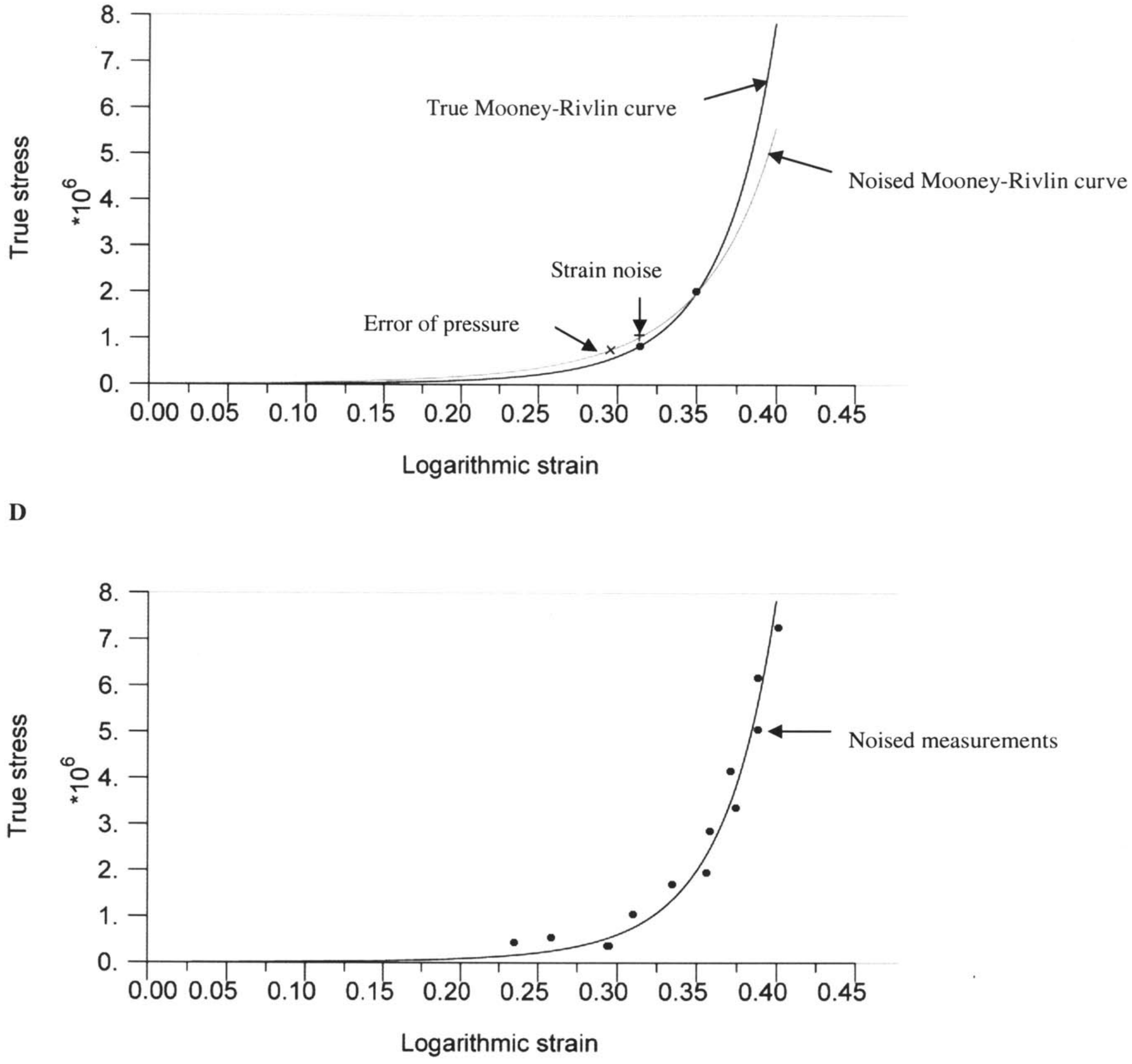


Fig. 3.3 Schematic stress-strain curve of a 1D problem

A) Having the strain at one given force (black dot in the figure) we can determine the linear-elastic parameter of the material. B) Having the strain at one given force (black dot in the figure) there is no unique nonlinear model to fit the strain, where there can be numerous solutions. C) Having two frames (black dots), it is possible to determine the Mooney-Rivlin model where D1 and D2 are unknown (solid black curve), provided there's no image noise nor pressure error. However, if the strain is noisy (black cross) or the pressure measurement has error (black block), the curve fitted can convey large error (dot grey curve). D) When given more frames than two, the curve tends to satisfy all the given frames, minimizing the distance to all given points.

3.4 Random Exhaustive Search

To assess the multi-frame scheme in our current 2D and 3D models described previously, the strain data are extracted at specific time steps from the finite element model, where the applied load is known corresponding to the imposed incremental pressures ramping from 0 to 16 kPa within 24 time steps. The corresponding strain maps were imported into the algorithm for comparison with the elastography data. An initial population (of size 400 in the current study) of totally 6 material parameters (D_1 and D_2 for arterial wall, fibrous plaque, and lipid) were generated randomly in the initial search field as listed in Table 3.1. This covers a reasonable but relatively small range of possible values for each of the parameters. The best fit that minimizes the difference between strain vector generated by the algorithm and that obtained from elastography is considered the solution.

Table 3.1: True values of Mooney-Rivlin parameters and initial search field

Mooney-Rivlin parameters	True values		Initial search field of estimation algorithm	
	D_1 [Pa]	D_2	D_1 [Pa]	D_1
Arterial wall	2644.7	8.365	2000~4000	7~10
Fibrous plaque	5105.3	13	4000~6000	10~14
Lipid	50	0.5	20~60	0.3~0.6

To test the sensitivity of the overall multi-frame algorithm, white Gaussian noise (namely, 1%, 5%, and 10%) was added to the elastography strain data, and the robustness of the parameter estimation algorithm was tested using both single-frame or multi-frame schemes. Furthermore, the effect of pressure inaccuracy on the parameter estimation was assessed by applying 1%, 5%, and 10% pseudo error in the input pressure.

The two-frame method shows a distinctively smaller error as well as smaller standard deviation as compared to the single-frame method (see Table 3.2).

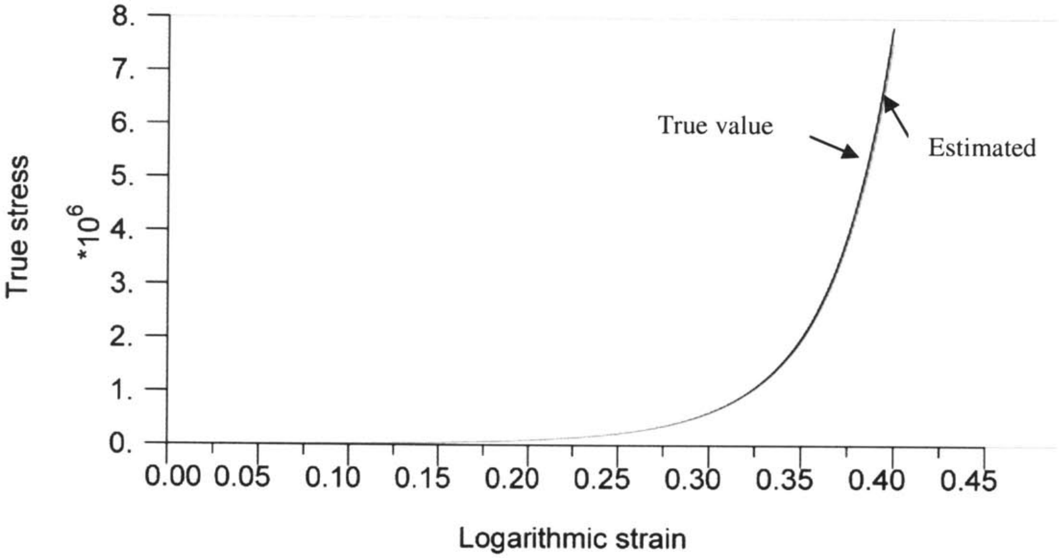
Table 3.2: Estimation results from 1-frame and 2-frame methods with noise-free data

Mooney-Rivlin	1-frame estimated results (based on 8 runs)	
parameters	D1 [Pa] (error)±SD	D2 (error)±SD
Arterial wall	3526.8 (33.4%)±33.5%	7.2 (14.2%)±22.4%
Fibrous plaque	5222.8 (2.3%)±6.7%	13.3 (1.9%)±5.0%
Lipid	56.2 (12.3%)±47.0%	0.6 (18.7%)±31.6%
Mooney-Rivlin	2-frame estimated results (based on 8 runs)	
parameters	D1 [Pa] (error)±SD	D2 (error)±SD
Arterial wall	2613.4 (1.2%)±7.1%	8.5 (1.2%)±5.2%
Fibrous plaque	4966.2 (2.7%)±2.3%	13.0 (0.3%)±1.0%
Lipid	43.4 (13.2%)±20.2%	0.5 (5.9%)±19.5%

The Mooney-Rivlin stress-strain curve for the arterial wall and lipid pool regions were used to evaluate the estimated vs. true parameters, based on the results given in Table 3.2 with a 2-frame method (see Fig. 3.4). For normal arterial wall (Fig. 3.4

A), the two curves agree well, suggesting little difference between true and estimated parameters. For lipid pool (Fig. 3.4 B), however, a considerable error was observed which is believed to be mainly due to lipid's relative softness compared to other wall regions that bear most of the pressure load. This lends itself to 'near-singular' behavior in lipid's estimated elastic modulus. That is, a small change in the magnitude (although large in percentage) of lipid's mechanical property yields negligible effect on the overall strain map. Nevertheless, since the contribution of lipid to the overall stress field is minor[13], the stress calculation in atherosclerotic vessel wall is not compromised.

A



B

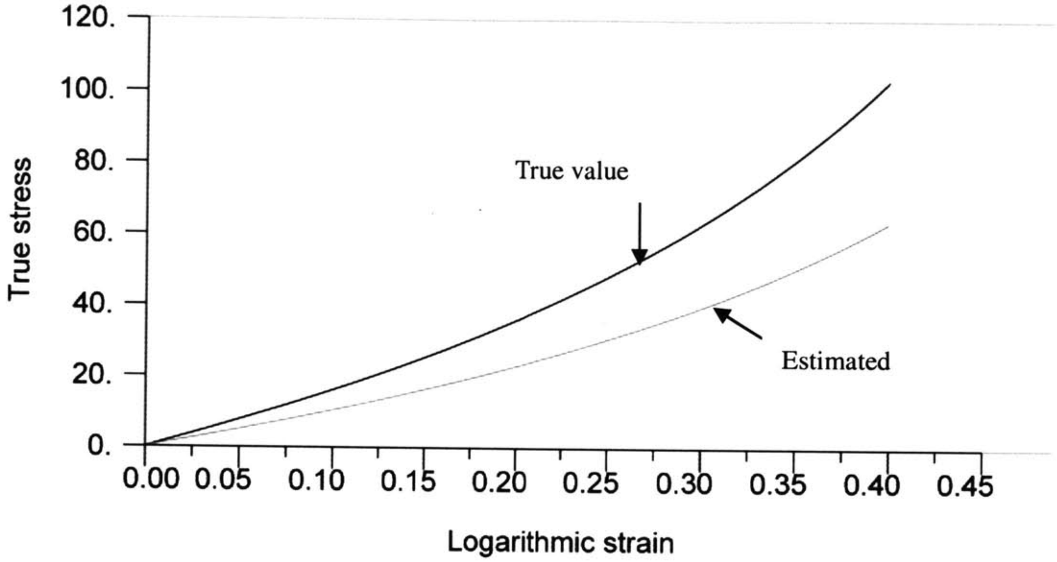


Figure 3.4 Stress-strain curve of vessel wall and lipid pool: comparison of true value and estimation result (table 3.2).

A) Comparison of stress-strain curve of vessel wall. Black curve is drawn from true values wall and gray curve is from the estimated results (very close). B) Comparison of stress-strain curve of lipid pool. Black curve is drawn from true values wall and gray curve is from the estimated results. Notice the Y axis scale is different in the two figures. Lipid pool is much softer than blood vessel wall and hence bears smaller stress under same strain conditions.

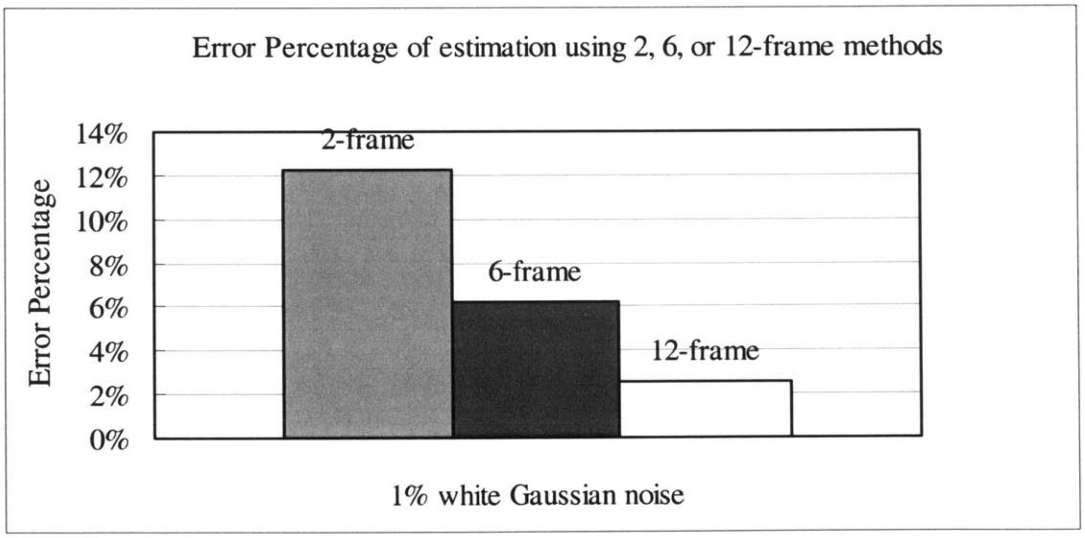
The sensitivity of the algorithm to the image (strain) noise was next assessed by using different levels of noise and frame numbers (Table 3.3 and Fig. 3.5)

Table 3.3: Estimation results from 1% noised strain data

Mooney-Rivlin	2-Frame Estimated results		6-Frame Estimated results		12-Frame Estimated results	
parameters	D_1 [Pa] (error)	D_2 (error)	D_1 [Pa] (error)	D_2 (error)	D_1 [Pa] (error)	D_2 (error)
Arterial wall	2117.7 (24.9%)	9.9 (15.5%)	2366.0 (11.8%)	9.1 (8.3%)	2585.6 (2.3%)	8.6 (2.4%)
Fibrous plaque	5370.3 (4.9%)	12.5 (3.7%)	4866.7 (4.9%)	13.0 (0.1%)	4859.5 (5.1%)	13.1 (0.5%)
Lipid	41.8 (19.6%)	0.5 (5.5%)	54.4 (8.1%)	0.5 (7.1%)	40.9 (22.2%)	0.6 (10.3%)

To evaluate the overall error in each case, we used the average error for the material parameters excluding the lipid pool, which has a relatively large standard deviation as discussed earlier. The overall error decreased as the number of frames used in algorithm was increased (Table 3.3 and Fig. 3.5 A). The parameter estimation error increased as the underlying (imposed) noise was elevated from 1% to 5% and 10%. At a 10% noise, the maximum error level was less than 7% which is reasonably small[13], suggesting that the algorithm is robust and shows a reasonably low sensitivity to the noise in the strain data. Though no comparable algorithm exists for nonlinear models, the present algorithm is generally less sensitive to elastography noise in contrast with the calculus-based algorithms with linear-elastic models[32].

A



B

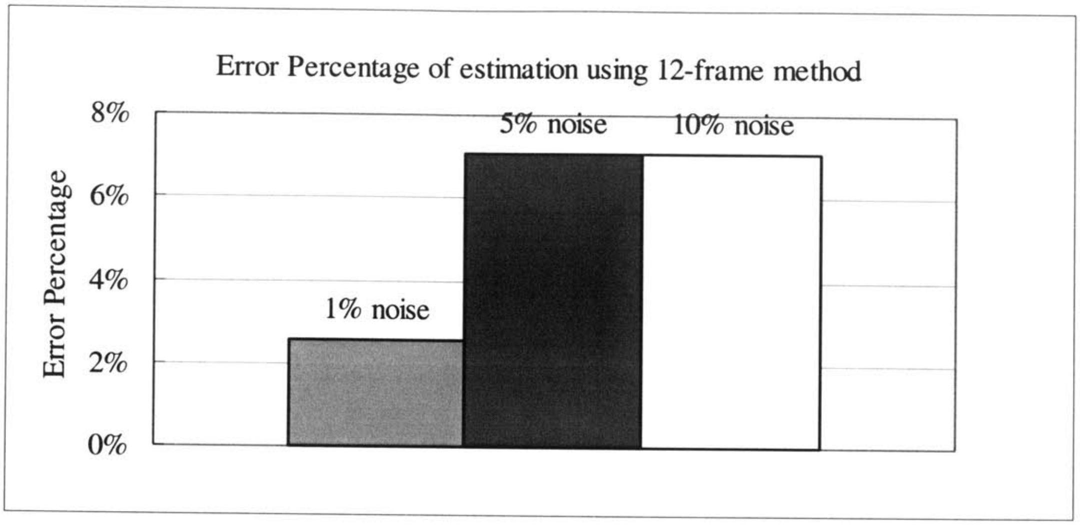


Figure 3.5. Sensitively analysis of the algorithm to the image noise. Error percentage is defined as the average error of all the parameters except that of the lipid's. A) Comparison of 2-frame, 6-frame and 12-frame methods under 1% strain noise (white Gaussian). B) Trend of error percentage increases up to 7% when strain noise increases from 1% to 5% and 10%, using the 12-frame method. No significant difference between 5% and 10% results was found.

To further characterize the overall genetic/FEM algorithm, we next tested the sensitivity of the algorithm (12-frame) to the error in pressure measurement (see Fig. 3.6). A 10% uncertainty in the pressure data yielded overall error levels up to 15%.

A

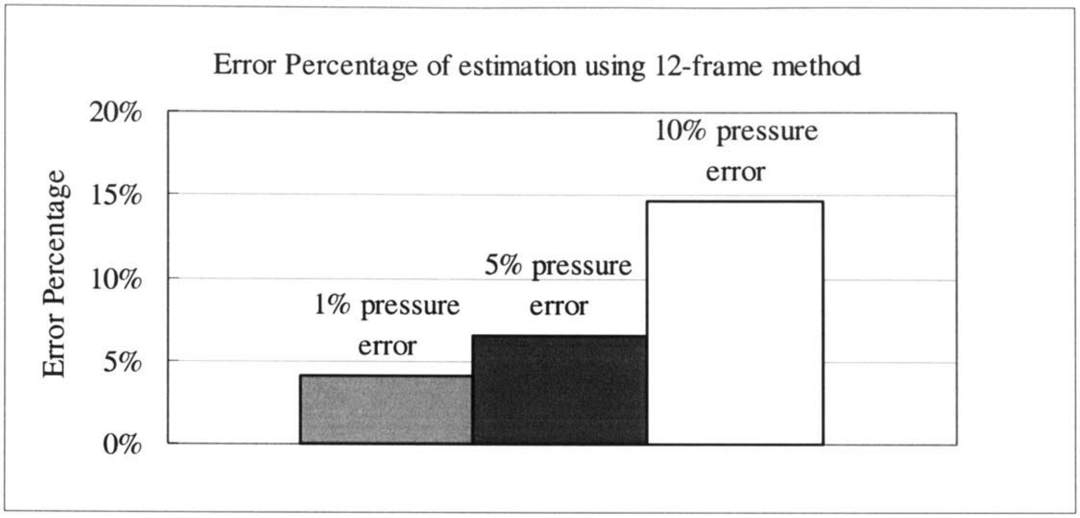


Figure 3.6 Sensitively analysis of the algorithm to the pressure measurement.

Trend of error percentage increases up to 15% when 1%, 5%, and 10% higher-than-normal pressure are used for the estimation.

3.3 Extension to 3D Model

To test the performance of the present nonlinear genetic/FEM algorithm in estimating the mechanical properties of plaques in 3D, a preliminary study was conducted using an idealized 3D geometry (Fig. 3.1B). The error between the real and estimated mechanical properties for intra-plaque regions was less than 15% (see Table 3.4). Though further investigation is needed to verify the feasibility of the algorithm on 3D model, the current result suggests the viability of our algorithm in 3D applications.

Table 3.4 Estimated mechanical properties on 3D model

Mooney-Rivlin	2-Frame Estimated results	
parameters	D_1 [Pa] (error)	D_2 (error)
Arterial wall	2801.0 (5.9%)	8.4 (1.2%)
Fibrous plaque	5105.3 (0.4%)	12.1 (6.9%)
Lipid	57.0 (14.0%)	0.46 (8.0%)

Chapter 4

Genetic Algorithm Approach

4.1 The Genetic Algorithm Search Scheme

A combined genetic/FEM algorithm was earlier developed to estimate the linear elastic mechanical properties of atherosclerotic tissues [16]. Briefly, a genetic algorithm is a search method that simulates biological evolution[33] by using the Darwinian principle of survival of the fittest to build search solutions. It was fist studied by David Goldberg, under the goal of optimizing parameters in a slightly different way than traditional method[34]. Genetic algorithms, developed by John Holland and colleagues, are search methods that simulate biological evolution through naturally occurring genetic operations on chromosomes [33]. Genetic algorithms begin with a predefined initial population of individuals, typically created randomly from a field of possible search solutions. Each "individual" in the population has a corresponding fitness value, which quantifies how fit the individual is in comparison to others. In the current problem, the fitness function is derived from the difference between the measured and predicted effective strains

$$e = \sqrt{e_{xx}^2 + e_{yy}^2 + \tfrac{1}{2} e_{yz}^2}$$

Summed over all elements, the smaller the summed difference is, the greater the probability that the individual will advance to the next population generation. Through pseudo genetic operations, such as crossover reproduction, the "fittest" individuals in the population are selected to survive to the next generation and are used as parents for the generation of new individuals in the population of next iteration.

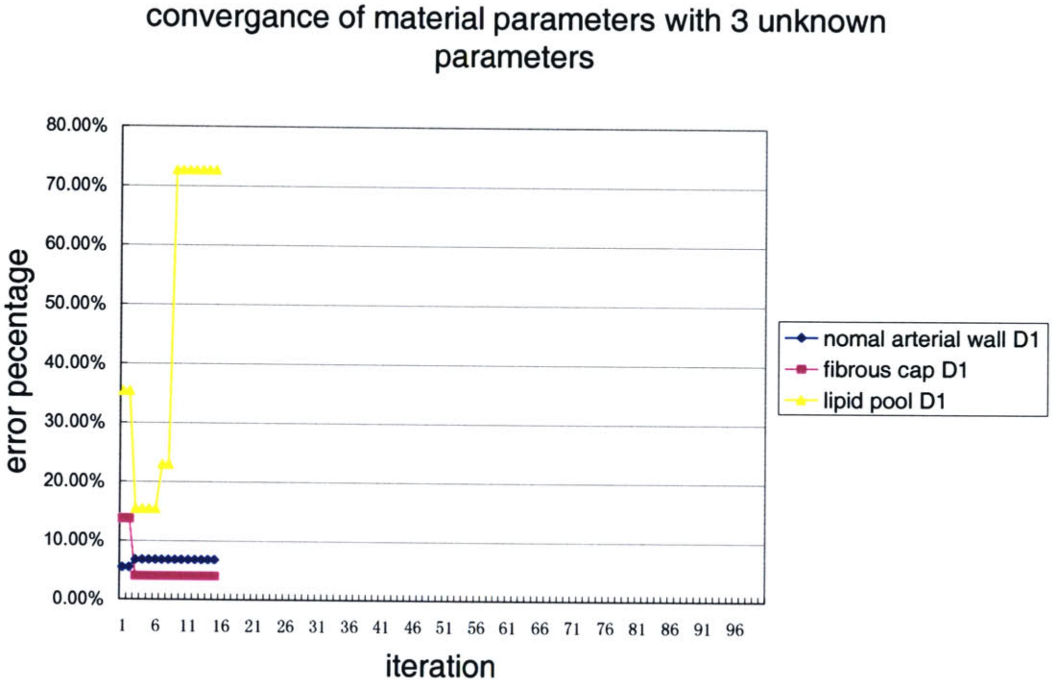
In the current study, we extend this combined algorithm to incorporate nonlinear mechanical properties (namely the Mooney-Rivlin model). As shown in Fig. 3.2 A, for each of the vascular regions: fibrous plaque, lipid pool and vessel wall, two parameters (D1 and D2, as defined in Mooney-Rivlin model) are needed to describe the mechanical property. Therefore, there are totally six unknown parameters for a typical problem.

The code we developed in this study is derived from part of Ahmad S. Khalil's work [35], which is on using genetic algorithm to estimate linear elastic vascular mechanical properties. In this study, to improve the robustness of the algorithm, we extended the algorithm by incorporating a "mutation" feature. Briefly, in each iteration, a stream of "new blood" (i.e., independently generated parameters) are added into the population in each iteration. Hence, ideally, if run for a long enough time and the number of parameters it tried out approaches infinity, it should closely find the true values. However, in our experiment, genetic algorithm without mutation does not work well for the 6 parameters problem. The reason for this might be, when the number of parameters increases from 3 to 6, the odds of recombining these parameters correctly is squared. That means one may see a very slow improvement by doing recombination with 6 parameters, which is consistent with our results (not shown). However, in the genetic algorithm with mutation, because newly generated parameters are brought in each iteration, and not with complete randomness, one does not have to walk through all the possible parameters in the search field. In other words, the speed of approaching the ultimate true value of these methods is different. And as in the result we show, the genetic algorithm method with mutation appears to be more efficient than random exhaustive search.

4.2 Estimation Results

First, we apply the non-linear estimation code in a simplified situation, where we D2 for each region is assumed by imposing the true values. Therefore each region has only one parameter to be estimated. As shown in Fig. 4.1 A, an initial population of 40 is used for each iteration. As we have also found in the linear elastic problem, as well as the nonlinear problem solved by random exhaustive method, an accurate estimation of lipid is always not achievable. The error percentage of parameters other than the lipid region reaches around 5% after around 200 calls to ADINA. This is comparable with the linear-elastic results[35].

A



B

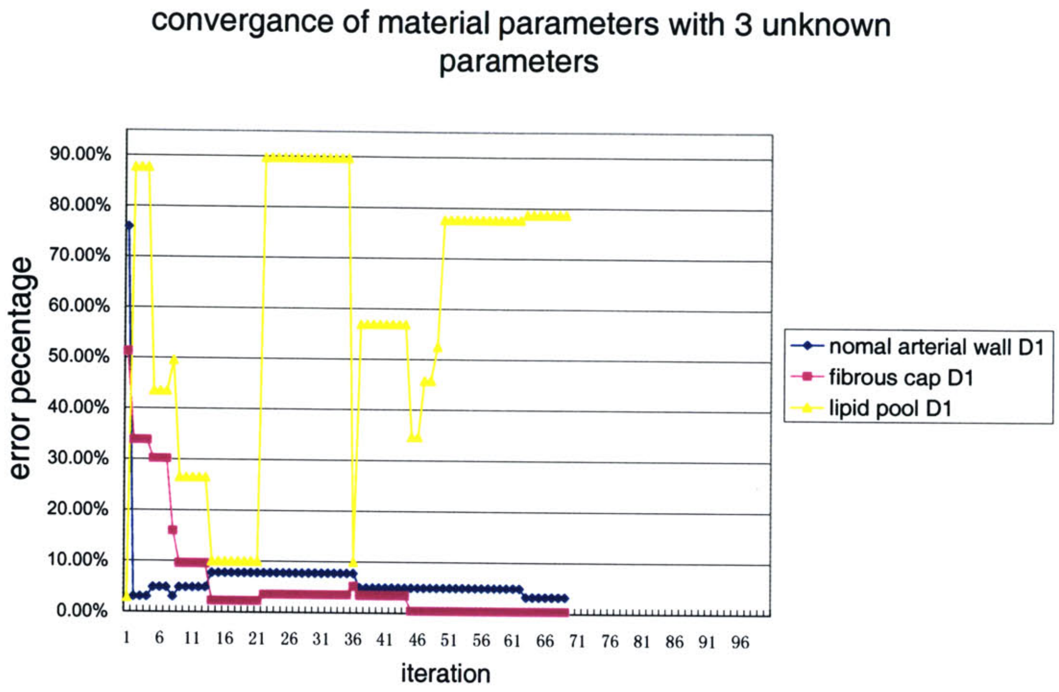
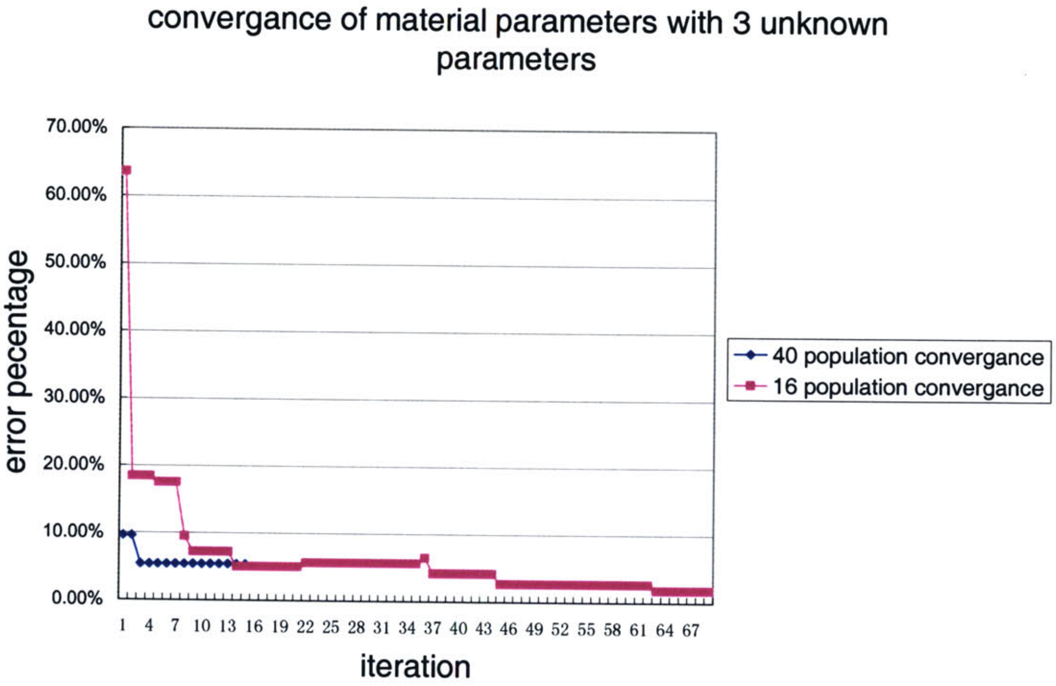


Fig. 4.1 Convergence of the 3 parameters with an A) initial population of 40 and B) initial population of 16.

As we did in for the random exhaustive estimation, we select an overall error percentage defined as the average error percentage of all the parameters (in this case all D1) except the lipid's. As shown in Fig. 4.2 A) the convergence of the overall error percentage is plotted against the iteration and in B) with respect to the total call to ADINA, which presents the computational expense. As we can see by using the initial population of 40, the result after initial iteration is closer to the true value than that using 16, but after about 200 calls to ADINA, both have achieved decent accuracy (about 5% error). It is acknowledged that to achieve certain accuracy with minimal total computational expense, an optimal population number exists, as too small or too big population are both practically inefficient. However, the difference between 40 and 16 as established by existing data in Fig. 4.2 B appears to be small.

A



B

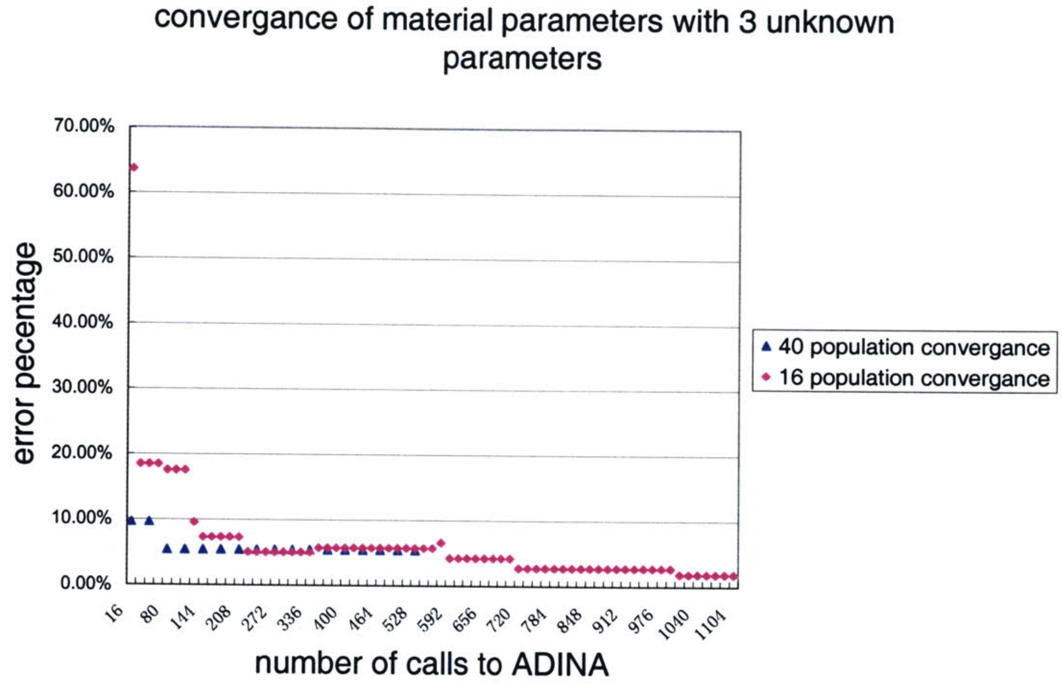


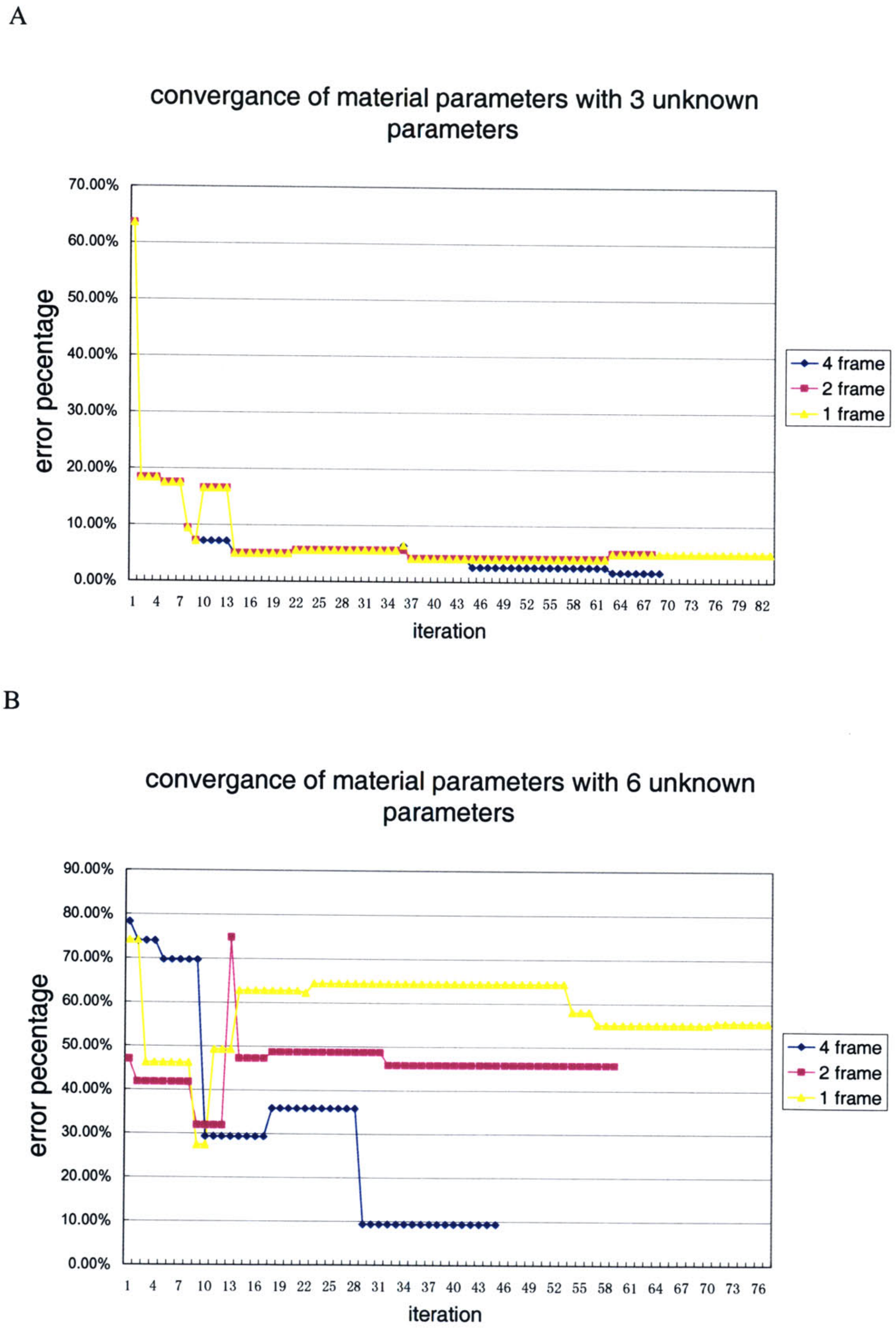
Fig. 4.2 A) the convergence of the overall error percentage is plotted against the iteration and in B) with respect to the total call to ADINA.

In the previous results, a single frame method was used, and in the following we test (an initial population of 40 is used in call cases) if multiframe method performs differently in the problem with 3 and 6 unknown parameters. In the 6-unknown-parameter problem the initial search field is listed in Table 4.1. Note that the range of initial search field is much larger (covering a range 10 folds) than that used for the random exhaustive estimation shown in Table 3.1, for it was impossible for random exhaustive estimation to get satisfying results with such a large search field.

Table 4.1: True values of Mooney-Rivlin parameters and initial search field

Mooney-Rivlin parameters	True values		Initial search field of estimation algorithm	
	D_1 [Pa]	D_2	D_1 [Pa]	D_1
Arterial wall	2644.7	8.365	1000~10000	1~10
Fibrous plaque	5105.3	13	1000~10000	10~100
Lipid	50	0.5	10~100	0.1~1

In Fig. 4.3 A, we show that increasing the number of frames used for the estimation from 1 to 2 and to 4 does not increase the accuracy/efficiency of the algorithm significantly. However, for the 6-unknown-parameter problem, as we discussed in Chapter 3, at least 2 frames is required to approach the true values. And further increasing the number of frames can reduce the effect the noise. In Fig. 4.3 B, we show that using 4 frames, the error percentage reaches less than 10% around 28 iterations. By using 1 frame it is virtually impossible for the algorithm to approach the true values. Theoretically by using 2 frames it is possible to converge to the true value. However, even in the current study where strain maps are generated numerically, having noise is inevitable. Hence we see an improved convergence by using the 4-frame method. Also, it is worthwhile to note here that the number of frames used in the algorithm does not necessarily increase the computational cost, i.e. the number of call to the FEM software or the time cost of each call, because for a nonlinear FEM procedure, a certain number of steps (in this case, 24) is required anyway.


A

convergance of material parameters with 3 unknown parameters

B

convergance of material parameters with 6 unknown parameters

Fig. 4.3 Increasing the number of frames used for the estimation for A) the 3-unknown-parameter problem and B) the 6-unknown-parameter problem.

As we mentioned above, genetic algorithms can perform the estimation within a much larger range of possible values than the random exhaustive search can afford. We are also interested to see how different these two methods perform given the same computational intensity available, for both of them could find the true value eventually but their speed of approaching the value is different. In Fig. 4.4, we show that at the same computational expensive (in terms of total number of calls to ADINA) genetic algorithm reaches a much better accuracy than the random exhaustive search. Because of the randomness in generating the initial population in given search field, the result of random exhaustive search may vary. Hence, standard deviations are calculated for the random exhaustive search results, each based on 8 independent runs. 4 frames are used in both and the initial search range is according to Table 4.1.

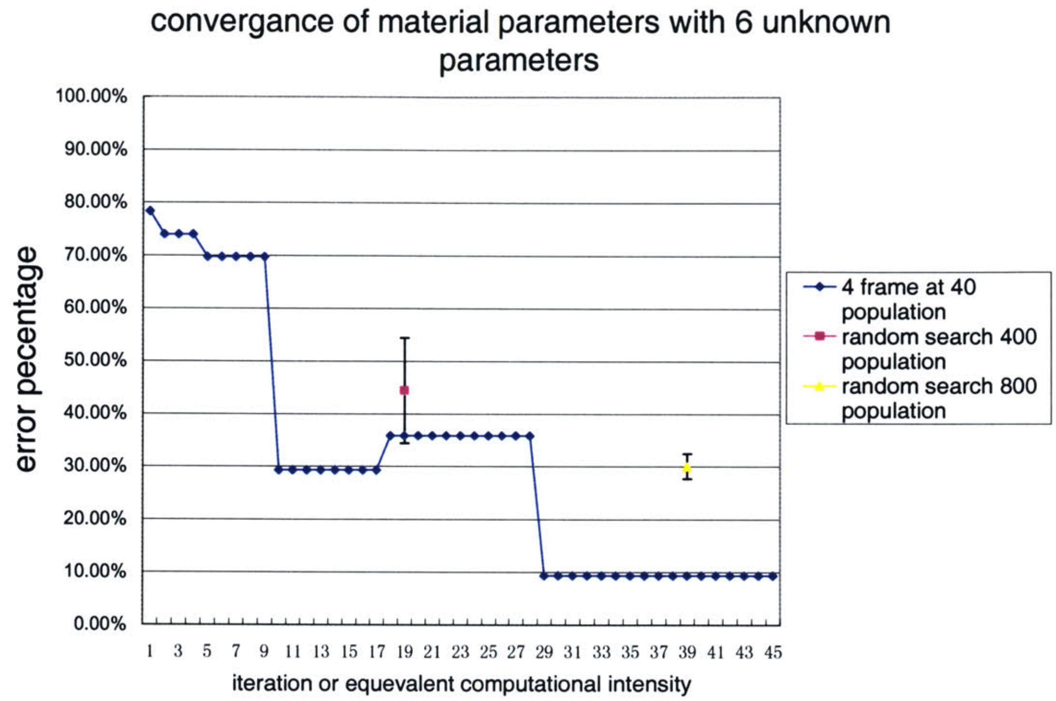


Fig. 4.4 Random exhaustive search VS. genetic algorithm in terms of computational efficiency.

The effect of noise and how the increase of frame used in the estimation can be used against the influence of noise are comprehensively studied in Chapter 3 for the random exhaustive search. We also show in Fig. 4.5 that for the genetic algorithm the same arguments we made in Chapter may be also applicable, i.e. although 2 is the minimal enough number of frames for the non-linear estimation without noise, a larger number used can compensate the effect of noise.

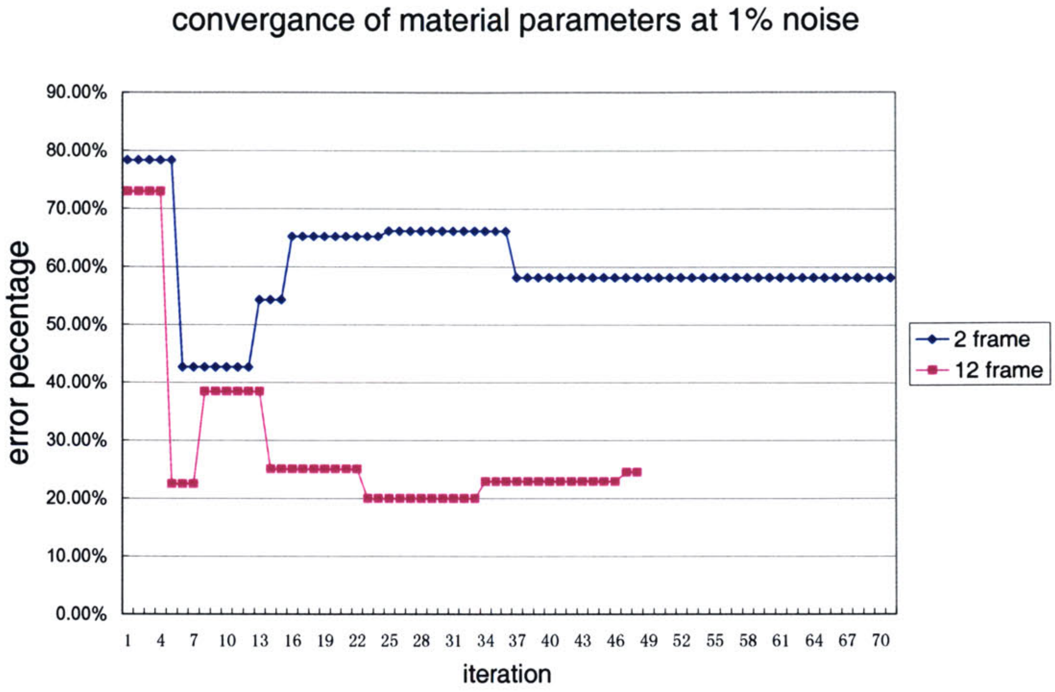


Fig. 4.5 Effect of noise on the estimation A) with 2 frames and B) using 12 frames.

We also tested the effect of higher noise, and with 5% noise it is already impossible to get accurate estimation (data not shown), even though 12 frames are used, indicating the sensitivity of the problem to the image noises.

Chapter 5
Summary & Conclusions

To mimic the strain-stiffening behavior of vascular tissues nonlinear constitutive models must be used. A lumped parameter genetic algorithm method, described earlier [16] as a robust, efficient means for parameter estimation, was extended here to incorporate a nonlinear elastic model. Nonlinear material models [30] do not easily lend themselves to calculus-based techniques for parameter estimation. Genetic algorithm, in contrast, is straight-forward and efficient when the model system can be lumped into a small number of parameters (e.g. less than 10). The algorithm was further characterized by quantifying its accuracy and low sensitivity to noise of the estimation on the current model.

Our 2D models, incorporating OCT-based subject-specific 2D images [13], involved FEM analysis with plain strain element, which is only valid if the vessel is either constrained longitudinally or if the longitudinal dimension is sufficiently large and the longitudinal strains are negligible. As the elastography data was generated with the same 2D FEM analysis, this does not influence the parameter estimation results. This may not be the case *in vivo*, as some segments of coronary vessels can undergo curvature change during the cardiac cycle. Longitudinal variations in plaque geometry might also significantly alter stress and strain fields, possibly affecting the accuracy of FEM· analysis and the overall parameter estimation algorithm. Due to this consideration, 3D FEM analysis was preliminarily investigated and the robustness of the algorithm in applicability to more complex and realistic FEM models was demonstrated. However, the out-of-plane strain is extremely hard to get from 3D elastography, because it is difficult to correlate the pixels between adjacent slices. This could become a major obstacle that limits the accuracy of 3D estimation.

Realistically, tissue mechanical properties are continuous and inhomogeneous in space. Lumping parameter is a strong assumption and can lead to artificial stress concentrations that undermine the viability of this method in assessing the plaque vulnerability. Nevertheless, when provided with the *in vivo* elastography data via OCT or high resolution MRI, this algorithm can estimate the patient-specific mechanical properties non-invasively. Mechanical properties are intrinsic to the specific tissue. For instance, *ex vivo* studies have observed that lipid's mechanical properties are influenced by its components [14]. Monitoring the change of such parameters *in vivo* allow for longitudinal studies that can potentially increase our understanding of the physiological change of the tissue during the progression of atherosclerosis. By differentiating the mechanical characteristics of vascular tissues with high or low risk of plaque rupture, it is possible to set up a diagnostic tool for assessment of plaque vulnerability based on the distribution of stress/strain and plaque geometry and composition.

In the current study, we performed quasi-static FEM analysis where the applied pressure load was incrementally raised. This is a valid assumption for *ex vivo* elastography, where the pressure load is applied slowly allowing sufficient time for the tissue to equilibrate. However, for the *in vivo* case, due to the blood pressure oscillation the dynamic effects of the vessel wall and/or blood and tissue surrounding it might not be negligible. Viscoelastic models[14] may also be needed for dynamic analysis, especially for the lipid pool component of the plaques. Such dynamic analyses lend themselves to genetic algorithms with lumped parameter model, which are much easier to implement as compared to calculus-based methods.

Another limitation of present study is that residual strain was not considered in the finite element analysis of the plaques. Unlike with linear elastic material modes, the issue of residual strain tends to become more important with nonlinear mechanical models. In the current study, elastography data obtained from the FEM model were used and hence the neglect of residual strains does not affect the parameter estimation algorithm. However, due to the lack of an accurate model to quantify the residual stress in an artery, it is difficult to assess the residual strain non-invasively. A recent study found that the cyclic strain distribution remains relatively unchanged by the inclusion of residual stress [36]. But still, the influence of residual strain on the nonlinear mechanical property estimation remains to be addressed.

The multi-frame scheme was introduced here to determine the nonlinear material model, and was used as an effective means for decreasing the sensitivity of the algorithm to the noise from both strain image and pressure measurement. This feature is not only useful for nonlinear material model but also helpful to the estimation based on linear-elastic model.

Genetic algorithm was proven to be a viable and relatively efficient method. But the image noise and pressure uncertainty strongly affects the accuracy of the estimation. We also realize although the multi-frame scheme may be helpful to solve the problem of noise, the real case could be far more complicated than the simple models we tested here. Hence, noise is still the biggest obstacle in developing such an estimation method, although ultimately with the development of imaging techniques and elastography this method might be applied for clinical purposes.

Bibliography

1. Yeh THE. Crp as a mediator of disease. *Circulation.* 2004:II11-14
2. Yeh ET, Anderson HV, Pasceri V, Willerson JT. C-reactive protein: Linking inflammation to cardiovascular complications. *Circulation.* 2001;104:974-975
3. Ross R. Atherosclerosis - an inflammatory disease. *N Engl J Med.* 1999;340:115-126
4. Davies MJ, Thomas T. The pathological basis and microanatomy of occlusive thrombus formation in human coronary arteries. *Philos Trans R Soc London, Ser B.* 1981;294:225-229
5. Loree HM, Kamm RD, Stringfellow RG, Lee RT. Effects of fibrous cap thickness on peak circumferential stress in model atherosclerotic vessels. *Circ Res.* 1992;71:850-858
6. Lee RT GA, Frank EH, Kamm RD and Schoen FJ. Structure-dependent dynamic mechanical-behavior of fibrous caps from human atherosclerotic plaques. 1991;83:1764-1770
7. de Korte CL, van der Steen AF, Cpedes EI, Pasterkamp G, Carlier SG, Mastik F, Schoneveld AH, Serruys PW, Bom N. Characterization of plaque components and vulnerability with intravascular ultrasound elastography. *Phys Med Biol.* 2000;45:1465-1475
8. Yabushita H, Bouma BE, Houser SL, Aretz HT, Jang IK, Schlendorf KH, Kauffman CR, Shishkov M, Kang DH, Halpern EF, Tearney GJ. Characterization of human atherosclerosis by optical coherence tomography. *Circulation.* 2002;106:1640-1645
9. Yuan C, Tsuruda JS, Beach KN, Hayes CE, Ferguson MS, Alpers CE, Foo TK, Strandness DE. Techniques for high-resolution mr imaging of atherosclerotic plaque. *J Magn Reson Imaging.* 1994;4:43-49
10. Richardson PD. Biomechanics of plaque rupture: Progress, problems, and new frontiers. *Ann Biomed Eng.* 2002;30:524-536
11. Skovoroda AR, Emelianov SY, O'Donnell M. Tissue elasticity reconstruction based on ultrasonic displacement and strain images. *IEEE Trans Ultrason Ferroelectr Freq Control.* 1995;42:747-765
12. Kallel F, Bertrand M. Tissue elasticity reconstruction using linear perturbation method. *IEEE Trans Med Imaging.* 1996;15:299-313
13. Williamson SD, Lam Y, Younis HF, Huang H, Patel S, Kaazempur-Mofrad MR, Kamm RD. On the sensitivity of wall stresses in diseased arteries to variable material properties. *J Biomech Eng.* 2003;125:147-155
14. Loree HM, Tobias BJ, Gibson LJ, Kamm RD, Small DM, Lee RT. Mechanical properties of model atherosclerotic lesion lipid pools. *Arterioscler Thromb.* 1994;14:230-234

15. Barbone PE, Bamber JC. Quantitative elasticity imaging: What can and cannot be inferred from strain images. *Phys Med Biol.* 2002;47:2147-2164

16. Khalil AS, Kamm RD, Bouma BE, Kaazempur-Mofrad MR. An fem/genetic algorithm approach for parameter estimation: Application in characterization of atherosclerotic plaques. *Journal of Computational Physics.* submitted

17. Chau AH, Chan RC, Shishkov M, MacNeill B, Iftimia N, Tearney GJ, Kamm RD, Bouma BE, Kaazempur-Mofrad MR. Finite element analysis of atherosclerotic plaques based on optical coherence tomography. *Ann Biomed Eng.* 2004

18. Schmitt JM. Oct elastography: Imaging microscopic deformation and strain of tissue. *Opt Express.* 1998;3:199-211

19. Tearney GJ, Brezinski ME, Bouma BE, Boppart SA, Pitris C, Southern JF, Fujimoto JG. In vivo endoscopic optical biopsy with optical coherence tomography. *Science.* 1997;276:2037-2039

20. de Korte CL, van der Steen AF, Céspedes EI, Pasterkamp G, Carlier SG, Mastik F, Schoneveld AH, Serruys PW, Bom N. Characterization of plaque components and vulnerability with intravascular ultrasound elastography. *Phys Med Biol.* 2000;45:1465-1475

21. Toussaint JF, Southern JF, Fuster V, Kantor HL. T2-weighted contrast for nmr characterization of human atherosclerosis. *Arterioscler Thromb Vasc Biol.* 1995;15:1533-1542

22. Toussaint JF, LaMuraglia GM, Southern JF, Fuster V, Kantor HL. Magnetic resonance images lipid, fibrous, calcified, hemorrhagic, and thrombotic components of human atherosclerosis in vivo. *Circulation.* 1996;94:932-938

23. Shinnar M, Fallon JT, Wehrli S, Levin M, Dalmacy D, Fayad ZA, Badimon JJ, Harrington M, Harrington E, Fuster V. The diagnostic accuracy of ex vivo mri for human atherosclerotic plaque characterization. *Arterioscler Thromb Vasc Biol.* 1999;19:2756-2761

24. Ophir J, Céspedes I, Ponnekanti H, Yazdi Y, Li X. Elastography: A quantitative method for imaging the elasticity of biological tissues. *Ultrason Imaging.* 1991;13:111-134

25. Ophir J, Céspedes I, Garra B, Ponnekanti H, Huang Y, Maklad N. Elastography: Ultrasonic imaging of tissue strain and elastic modulus in vivo. *Eur J Ultrasound.* 1996;3:49-70

26. Cespedes I, Ophir J, Ponnekanti H, Maklad N. Elastography: Elasticity imaging using ultrasound with application to muscle and breast in vivo. *Ultrason Imaging.* 1993;15:73-88

27. Greenleaf JF, Fatemi M, Insana M. Selected methods for imaging elastic properties of biological tissues. *Annu Rev Biomed Eng.* 2003;5:57-78

28. Chau AH, Chan RC, Shishkov M, MacNeill B, Iftimia N, Tearney GJ, Kamm RD, Bouma BE, Kaazempur-Mofrad MR. Finite element analysis of atherosclerotic plaques based on optical coherence tomography. *Ann Biomed Eng.* in press

29. Rivlin RS. "large elastic deformations of isotropic materials iv. Further developments of the general theory". *Philosophical Transactions of the Royal Society of London.* 1948;A 241:379-397

30. Bathe K-J. Finite element procedures. Upper Saddle River, New Jersey: Prentice Hall; 1996:592-594.

31. Huang H, Virmani R, Younis H, Burke AP, Kamm RD, Lee RT. The impact of calcification on the biomechanical stability of atherosclerotic plaques. *Circulation*. 2001;103:1051-1056

32. Oberai AA, Gokhale NH, Feij GR. Solution of inverse problems in elasticity using the adjoint method. *Inverse Problems*. 2003;19:297-313

33. Holland JH. *Adaptation in natural and artificial systems*. Ann Arbor, MI: The University of Michigan Press; 1975.

34. Goldberg DE. *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley; 1989.

35. Khalil AS. Model parameter estimation of atherosclerotic plaque mechanical properties: Calculus-based and heuristic algorithms. 2004:176

36. Kaazempur-Mofrad MR, Younis HF, Patel S, Isasi A, Chung C, Chan RC, Hinton DP, Lee RT, Kamm RD. Cyclic strain in human carotid bifurcation and its potential correlation to atherogenesis: Idealized and anatomically-realistic models. *Journal of Engineering Mathematics*. 2003;47:299-314

Appendix A

Sample ADINA .in file of 2D arterial geometry with lipid pool

```
*
* Command file created from session file information stored within AUI database
*
*--- Database created 8 May 2004, 00:00:00 ---*
*--- by ADINA: AUI version 8.1.0 ---*
*
DATABASE NEW SAVE=NO PROMPT=NO
FEPROGRAM ADINA
CONTROL FILEVERSION=V81
*
FEPROGRAM PROGRAM=ADINA
*
CONTROL PLOTUNIT=PERCENT VERBOSE=YES ERRORLIM=0 LOGLIMIT=0
UNDO=5,
      PROMPTDE=UNKNOWN AUTOREPA=YES DRAWMATT=YES
DRAWTEXT=EXACT,
      DRAWLINE=EXACT DRAWFILL=EXACT AUTOMREB=YES ZONECOPY=NO,
      SWEEPCOI=YES SESSIONS=YES DYNAMICT=YES UPDATETH=YES
AUTOREGE=NO,
      ERRORACT=CONTINUE FILEVERS=V81 INITFCHE=NO SIGDIGIT=6,
      AUTOZONE=YES
*
FEPROGRAM PROGRAM=ADINA
*
COORDINATES POINT SYSTEM=0
1   0   0.00075193654500   0.00014205222000   0
2   0   0.00075193654500   0.00014835222000   0
3   0   0.00075193654500   0.00015465222000   0
4   0   0.00074563654500   0.00015465222000   0
5   0   0.00074563654500   0.00016095222000   0
6   0   0.00074563654500   0.00016725222000   0
7   0   0.00073933654500   0.00016725222000   0
8   0   0.00073933654500   0.00017355222000   0
9   0   0.00073933654500   0.00017985222000   0
10  0   0.00073303654500   0.00017985222000   0
```

```
11   0   0.00073303654500    0.00018615222000    0
12   0   0.00073303654500    0.00019245222000    0
13   0   0.00072673654500    0.00019245222000    0
14   0   0.00072673654500    0.00019875222000    0
15   0   0.00072673654500    0.00020505222000    0
16   0   0.00072043654500    0.00020505222000    0
17   0   0.00072043654500    0.00021135222000    0
18   0   0.00072043654500    0.00021765222000    0
19   0   0.00071413654500    0.00021765222000    0
20   0   0.00071413654500    0.00022395222000    0
21   0   0.00071413654500    0.00023025222000    0
22   0   0.00070783654500    0.00023025222000    0
23   0   0.00070783654500    0.00023655222000    0
24   0   0.00070783654500    0.00024285222000    0
25   0   0.00070153654500    0.00024285222000    0
26   0   0.00070153654500    0.00024915222000    0
27   0   0.00070153654500    0.00025545222000    0
28   0   0.00069523654500    0.00025545222000    0
29   0   0.00069523654500    0.00026175222000    0
30   0   0.00069523654500    0.00026805222000    0
31   0   0.00068893654500    0.00026805222000    0
32   0   0.00068893654500    0.00027435222000    0
33   0   0.00068893654500    0.00028065222000    0
34   0   0.00068263654500    0.00028065222000    0
35   0   0.00068263654500    0.00028695222000    0
36   0   0.00068263654500    0.00029325222000    0
Snip...
4257      0    0.00021643654500     -0.00116834778000    0
4258      0    0.00021013654500     -0.00116834778000    0
4259      0    0.00020383654500     -0.00116834778000    0
4260      0    0.00019753654500     -0.00116834778000    0
4261      0    0.00019123654500     -0.00116834778000    0
4262      0    0.00018493654500     -0.00116834778000    0
4263      0    0.00017863654500     -0.00116834778000    0
4264      0    0.00017233654500     -0.00116834778000    0
4265      0    0.00016603654500     -0.00116834778000    0
4266      0    0.00015973654500     -0.00116834778000    0
4267      0    0.00015343654500     -0.00116834778000    0
4268      0    0.00014713654500     -0.00116834778000    0
@
```

```
*
LINE POLYLINE NAME=1 TYPE=BEZIER
1 0.0000000000000 0.0000000000000 0.0000000000000
2 0.0000000000000 0.0000000000000 0.0000000000000
3 0.0000000000000 0.0000000000000 0.0000000000000
4 0.0000000000000 0.0000000000000 0.0000000000000
5 0.0000000000000 0.0000000000000 0.0000000000000
6 0.0000000000000 0.0000000000000 0.0000000000000
7 0.0000000000000 0.0000000000000 0.0000000000000
8 0.0000000000000 0.0000000000000 0.0000000000000
9 0.0000000000000 0.0000000000000 0.0000000000000
10 0.0000000000000 0.0000000000000 0.0000000000000
11 0.0000000000000 0.0000000000000 0.0000000000000
12 0.0000000000000 0.0000000000000 0.0000000000000
13 0.0000000000000 0.0000000000000 0.0000000000000
14 0.0000000000000 0.0000000000000 0.0000000000000
15 0.0000000000000 0.0000000000000 0.0000000000000
16 0.0000000000000 0.0000000000000 0.0000000000000
17 0.0000000000000 0.0000000000000 0.0000000000000
18 0.0000000000000 0.0000000000000 0.0000000000000
19 0.0000000000000 0.0000000000000 0.0000000000000
20 0.0000000000000 0.0000000000000 0.0000000000000
21 0.0000000000000 0.0000000000000 0.0000000000000
22 0.0000000000000 0.0000000000000 0.0000000000000
23 0.0000000000000 0.0000000000000 0.0000000000000
24 0.0000000000000 0.0000000000000 0.0000000000000
25 0.0000000000000 0.0000000000000 0.0000000000000
26 0.0000000000000 0.0000000000000 0.0000000000000
27 0.0000000000000 0.0000000000000 0.0000000000000
28 0.0000000000000 0.0000000000000 0.0000000000000
29 0.0000000000000 0.0000000000000 0.0000000000000
30 0.0000000000000 0.0000000000000 0.0000000000000
31 0.0000000000000 0.0000000000000 0.0000000000000
32 0.0000000000000 0.0000000000000 0.0000000000000
33 0.0000000000000 0.0000000000000 0.0000000000000
34 0.0000000000000 0.0000000000000 0.0000000000000
35 0.0000000000000 0.0000000000000 0.0000000000000
36 0.0000000000000 0.0000000000000 0.0000000000000
37 0.0000000000000 0.0000000000000 0.0000000000000
38 0.0000000000000 0.0000000000000 0.0000000000000
```

```
39 0.00000000000000 0.00000000000000 0.00000000000000
40 0.00000000000000 0.00000000000000 0.00000000000000
41 0.00000000000000 0.00000000000000 0.00000000000000
42 0.00000000000000 0.00000000000000 0.00000000000000
Snip...
4268 0.00000000000000 0.00000000000000 0.00000000000000
2796 0.00000000000000 0.00000000000000 0.00000000000000
@
*
LINE COMBINED NAME=8 COUPLED=YES RESTRICT=YES
@CLEAR
3
4
5
@
*
LINE COMBINED NAME=9 COUPLED=YES RESTRICT=YES
@CLEAR
6
7
@
*
BODY SHEET NAME=1 LINE=9 DELETE-L=NO
@CLEAR
8
@
*
BODY SHEET NAME=2 LINE=8 DELETE-L=NO
@CLEAR
2
1
@
*
BODY SHEET NAME=3 LINE=1 DELETE-L=NO
@CLEAR
@
*
MATERIAL MOONEY-RIVLIN NAME=1 C1=0.00000000000000 C2=0.00000000000000,
     C3=0.00000000000000 C4=0.00000000000000 C5=0.00000000000000,
     C6=0.00000000000000 C7=0.00000000000000 C8=0.00000000000000,
     C9=0.00000000000000 D1=2644.70000000000 D2=8.36500000000000,
     KAPPA=6.63689000000000E+07 DENSITY=0.00000000000000 FITTING-=0,
     VISCOELA=0 MDESCRIP='NONE'
*
MATERIAL MOONEY-RIVLIN NAME=2 C1=0.00000000000000 C2=0.00000000000000,
     C3=0.00000000000000 C4=0.00000000000000 C5=0.00000000000000,
     C6=0.00000000000000 C7=0.00000000000000 C8=0.00000000000000,
     C9=0.00000000000000 D1=5105.30000000000 D2=13.0000000000000,
     KAPPA=6.63689000000000E+07 DENSITY=0.00000000000000 FITTING-=0,
     VISCOELA=0 MDESCRIP='NONE'
*
MATERIAL MOONEY-RIVLIN NAME=3 C1=0.00000000000000 C2=0.00000000000000,
     C3=0.00000000000000 C4=0.00000000000000 C5=0.00000000000000,
     C6=0.00000000000000 C7=0.00000000000000 C8=0.00000000000000,
     C9=0.00000000000000 D1=50.00000000000 D2=0.50000000000000,
     KAPPA=6.63689000000000E+07 DENSITY=0.00000000000000 FITTING-=0,
     VISCOELA=0 MDESCRIP='NONE'
*
EGROUP TWODSOLID NAME=1 SUBTYPE=STRAIN DISPLACE=DEFAULT,
     STRAINS=DEFAULT    MATERIAL=1    INT=DEFAULT    RESULTS=STRESSES
DEGEN=NO,
     FORMULAT=2 STRESSRE=GLOBAL INITIALS=NONE FRACTUR=NO,
     CMASS=DEFAULT STRAIN-F=0 UL-FORMU=DEFAULT PNTGPS=0 NODGPS=0,
     LVUS1=0 LVUS2=0 SED=NO RUPTURE=ADINA INCOMPAT=DEFAULT,
     TIME-OFF=0.00000000000000 POROUS=NO WTMC=1.00000000000000,
     OPTION=NONE DESCRIPT='NONE'
*
EGROUP TWODSOLID NAME=2 SUBTYPE=STRAIN DISPLACE=DEFAULT,
     STRAINS=DEFAULT    MATERIAL=2    INT=DEFAULT    RESULTS=STRESSES
DEGEN=NO,
     FORMULAT=2 STRESSRE=GLOBAL INITIALS=NONE FRACTUR=NO,
     CMASS=DEFAULT STRAIN-F=0 UL-FORMU=DEFAULT PNTGPS=0 NODGPS=0,
     LVUS1=0 LVUS2=0 SED=NO RUPTURE=ADINA INCOMPAT=DEFAULT,
     TIME-OFF=0.00000000000000 POROUS=NO WTMC=1.00000000000000,
     OPTION=NONE DESCRIPT='NONE'
*
EGROUP TWODSOLID NAME=3 SUBTYPE=STRAIN DISPLACE=DEFAULT,
     STRAINS=DEFAULT    MATERIAL=3    INT=DEFAULT    RESULTS=STRESSES
DEGEN=NO,
     FORMULAT=2 STRESSRE=GLOBAL INITIALS=NONE FRACTUR=NO,
     CMASS=DEFAULT STRAIN-F=0 UL-FORMU=DEFAULT PNTGPS=0 NODGPS=0,
     LVUS1=0 LVUS2=0 SED=NO RUPTURE=ADINA INCOMPAT=DEFAULT,
     TIME-OFF=0.00000000000000 POROUS=NO WTMC=1.00000000000000,
     OPTION=NONE DESCRIPT='NONE'
*
GFACE NODES=9 NCOINCID=BOUNDARIES NCTOLERA=1.00000000000000E-05,
     SUBSTRUC=0 GROUP=1 PREFSHAP=QUAD-DIRECT BODY=1 COLLAPSE=NO,
     SIZE-FUN=0 MIDNODES=CURVED METHOD=DELAUNAY NLAYER=1
NLTABL=0
@CLEAR
@
*
GFACE NODES=9 NCOINCID=BOUNDARIES NCTOLERA=1.00000000000000E-05,
     SUBSTRUC=0 GROUP=2 PREFSHAP=QUAD-DIRECT BODY=2 COLLAPSE=NO,
     SIZE-FUN=0 MIDNODES=CURVED METHOD=DELAUNAY NLAYER=1
NLTABL=0
@CLEAR
@
*
GFACE NODES=9 NCOINCID=BOUNDARIES NCTOLERA=1.00000000000000E-05,
     SUBSTRUC=0 GROUP=3 PREFSHAP=QUAD-DIRECT BODY=3 COLLAPSE=NO,
     SIZE-FUN=0 MIDNODES=CURVED METHOD=DELAUNAY NLAYER=1
NLTABL=0
@CLEAR
@
*
BOUNDARIES SUBSTRUC=0
@CLEAR
@
*
MASTER ANALYSIS=STATIC MODEX=EXECUTE TSTART=0.00000000000000,
     IDOF=100111 OVALIZAT=NONE FLUIDPOT=AUTOMATIC CYCLICPA=1,
     IPOSIT=STOP REACTION=YES INITIALS=NO FSINTERA=NO IRINT=DEFAULT,
     CMASS=NO SHELLNDO=AUTOMATIC AUTOMATI=ATS SOLVER=SPARSE,
     CONTACT-=CONSTRAINT-FUNCTION TRELEASE=0.00000000000000,
     RESTART-=NO FRACTURE=NO LOAD-CAS=NO LOAD-PEN=NO
MAXSOLME=0,
     MTOTM=2 RECL=3000 SINGULAR=YES STIFFNES=1000.00000000000,
     MAP-OUTP=NONE MAP-FORM=NO NODAL-DE='' POROUS-C=NO
ADAPTIVE=0,
     ZOOM-LAB=1     AXIS-CYC=0     PERIODIC=NO     VECTOR-S=GEOMETRY
EPSI-FIR=NO
*
TIMEFUNCTION NAME=1 IFLIB=1 FPAR1=1.00000000000000,
     FPAR2=1.00000000000000 FPAR3=1.00000000000000,
     FPAR4=0.00000000000000 FPAR5=0.00000000000000,
     FPAR6=0.00000000000000
@CLEAR
0.00000000000000 0.00000000000000
24.0000000000000 16.0000000000000
@
*
DELETE FRAME SURFACE=CURREN
*
TIMESTEP NAME=DEFAULT
@CLEAR
24 1.00000000000000
@
*
ITERATION METHOD=FULL-NEWTON LINE-SEA=DEFAULT MAX-ITER=50,
     PRINTOUT=ALL
*
PPROCESS NPROC=1 MINEL=0 MAXEL=999999
*
EGCONTROL MAXELG=999999
*
LOAD PRESSURE NAME=1 MAGNITUD=1000 BETA=0.00000000000000,
     LINE=0
*
APPLY-LOAD BODY=3
@CLEAR
@
*
APPLY-LOAD BODY=3
@CLEAR
1   'PRESSURE' 1   'EDGE' 6 0 1 0.00000000000000 0 -1 0 2 0   'NO',
     0.00000000000000 0.00000000000000 1 0
2   'PRESSURE' 1   'EDGE' 7 0 1 0.00000000000000 0 -1 0 2 0   'NO',
     0.00000000000000 0.00000000000000 1 0
@
```

```
*
FIXITY NAME=NOZ
@CLEAR
 'X-TRANSLATION'
 'Z-TRANSLATION'
 'X-ROTATION'
 'Y-ROTATION'
 'Z-ROTATION'
 'OVALIZATION'
@
*
FIXBOUNDARY POINTS FIXITY=ALL
@CLEAR
3175   'ALL'
3907   'NOZ'
@
*
SUBDIVIDE BODY NAME=1 MODE=LENGTH SIZE=0.00007000000000000
@CLEAR
2
3
@
*
GFACE NODES=9 NCOINCID=BOUNDARIES NCTOLERA=1.00000000000000E-05,
      SUBSTRUC=0 GROUP=1 PREFSHAP=TRIANGULAR BODY=1 COLLAPSE=NO,
      SIZE-FUN=0    MIDNODES=CURVED    METHOD=ADVFRONT    NLAYER=1
NLTABL=0
@CLEAR
1
@
*
GFACE NODES=9 NCOINCID=BOUNDARIES NCTOLERA=1.00000000000000E-05,
      SUBSTRUC=0 GROUP=2 PREFSHAP=TRIANGULAR BODY=2 COLLAPSE=NO,
      SIZE-FUN=0    MIDNODES=CURVED    METHOD=ADVFRONT    NLAYER=1
NLTABL=0
@CLEAR
1
@
*
GFACE NODES=9 NCOINCID=BOUNDARIES NCTOLERA=1.00000000000000E-05,
      SUBSTRUC=0 GROUP=3 PREFSHAP=TRIANGULAR BODY=3 COLLAPSE=NO,
      SIZE-FUN=0 MIDNODES=CURVED METHOD=ADVFRONT NLAYER=1
NLTABL=0
@CLEAR
1
@
*
PRINTOUT ECHO=NO PRINTDEF=STRAINS INPUT-DA=1 OUTPUT=SELECTED,
      DISPLACE=YES VELOCITI=YES ACCELERA=YES IDISP=NO ITEMP=NO,
      ISTRAIN=NO IPIPE=NO STORAGE=NO LARGE-ST=NONE
*
PRINT-STEPS SUBSTRUC=0 REUSE=1
@CLEAR
1 21 24 1
@
*
PPROCESS NPROC=2 MINEL=0 MAXEL=999999
*
EGCONTROL MAXELG=999999
*
ADINA OPTIMIZE=SOLVER FILE=,
'C:\genetic_algorithm_2D\2DOCT.dat',
      FIXBOUND=YES MIDNODE=NO OVERWRIT=YES
*
```

Appendix B

Sample Genetic Algorithm code

```
for mmm=1:1
save    'mmm.mat' mmm;

clear all;

format long e;

load 'mmm.mat';
```

```
% dos ( ['del ','Solutionrecord.mat']);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%
% USER SETTINGS

% Number of parameters
n = 6;
timestep = 4;
noiseratio = 0;
groupnumber = 3;
elements( 1 ) =   535; %7%         %4%1585   ;
elements( 2 ) =   674; %7%         %4%1854;
elements( 3 ) =   166; %7%         %4%526;
totalelements = sum( elements );

pop( 1 ) = 40 ;
pop( 2:100 ) = 40;
numMutation = 20;   %pop > 4*numMutation
% Eactual( 1 ) = 2644;
% Eactual( 2 ) = 5105;
% Eactual( 3 ) = 50;1.8e4
% Eactual( 4 ) = 8.36;
% Eactual( 5 ) = 13;
% Eactual( 6 ) = 0.5;20


maxValue( 1 ) = 10e3;
minValue( 1 ) = 1e3;

maxValue( 2 ) = 10e3;
minValue( 2 ) = 1e3;

maxValue( 3 ) = 100;
minValue( 3 ) = 10;

% maxValue( 3 ) = 1e4;
% minValue( 3 ) = 3e4;
% maxValue( 4 ) = 10;
% minValue( 4 ) = 7;
% maxValue( 5 ) = 14;
% minValue( 5 ) = 10;
% maxValue( 6 ) = 0.6;
% minValue( 6 ) = 0.3;


maxValue( 4 ) = 10;
minValue( 4 ) = 1;

maxValue( 5 ) = 100;
minValue( 5 ) = 10;

maxValue( 6 ) = 1;
minValue( 6 ) = 0.1;


if mmm==2
load 'Solutionrecord.mat';



maxValue( 1 ) = 4e3;
minValue( 1 ) = 2e3;
maxValue( 2 ) = Solutionrecord(2);
minValue( 2 ) = Solutionrecord(2);
% maxValue( 3 ) = 1e4;
% minValue( 3 ) = 3e4;
maxValue( 3 ) = 60;
minValue( 3 ) = 20;
maxValue( 4 ) = 10;
minValue( 4 ) = 7;
maxValue( 5 ) = Solutionrecord(5);
minValue( 5 ) = Solutionrecord(5);
maxValue( 6 ) = 0.6;
minValue( 6 ) = 0.3;
end

if mmm==3
load 'Solutionrecord.mat';

maxValue( 1 ) = Solutionrecord(1);
```

```
minValue( 1 ) = Solutionrecord(1);
maxValue( 2 ) = Solutionrecord(2);
minValue( 2 ) = Solutionrecord(2);
% maxValue( 3 ) = 1e4;
% minValue( 3 ) = 3e4;
maxValue( 3 ) = 60;
minValue( 3 ) = 20;
maxValue( 4 ) = Solutionrecord(4);
minValue( 4 ) = Solutionrecord(4);
maxValue( 5 ) = Solutionrecord(5);
minValue( 5 ) = Solutionrecord(5);
maxValue( 6 ) = 0.6;
minValue( 6 ) = 0.3;
end

%global GAdir;
GAdir = 'C:\genetic_algorithm_2D\';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%
ADINAdir = '"C:\Program Files\ADINA\ADINA System 8.1\bin\';
ADINAaui = strcat( ADINAdir, 'aui.exe" -b -m 100mb ' );
ADINA = strcat( ADINAdir, 'adina.exe" -b -s -m 100mb ' );


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%
initialfilename = '2DOCT';%      '2Dhistology'
iterfilename = strcat( initialfilename,'_');
skeletonFile = strcat( initialfilename,'.in');
skeletonFilePrefix = initialfilename;

          dos( ['"C:\Program Files\ADINA\ADINA System 8.1\bin\aui.exe" -b -m 200mb ',
skeletonFile ] );
           dos( ['"C:\Program Files\ADINA\ADINA System 8.1\bin\adina.exe" -b -s -m
200mb ', skeletonFilePrefix ] );
% % %
%   porfilename = strcat( skeletonFilePrefix,'.por');
% datfilename = strcat( skeletonFilePrefix,'.dat');
%   resfilename = strcat( skeletonFilePrefix,'.res');
%   modfilename = strcat( skeletonFilePrefix,'.mod');
%   dos ( ['del ',porfilename]);
%   dos ( ['del ',datfilename]);
%   dos ( ['del ',resfilename]);
% dos ( ['del ',modfilename]);

outfilename = strcat( skeletonFilePrefix,'.out');
%
[ actStrains ] = readOutFile('2DOCT.out', timestep,elements);    % ENSURE!
% if false == 1
% % break
% end

%%%%%%%%%%%%%%%%%Artifical Noise%%%%%%%%%%

% noise = 0;%-0.01;
% actStrains =
actStrains*(1+noise);%+actStrains.*(((rand(totalElements))-(rand(totalElements)))*noiseratio);

%
%dos( ['del ',outfilename]);
%
clear actualdisplacement;

actualdisplacement = reshape (actStrains,size(actStrains,1)*3,1);
actualdisplacementcomp = actualdisplacement;
for i= 1: size(actStrains,1)*3
    actualdisplacement(i)=actualdisplacement(i)*(1+randn*sqrt(noiseratio));
end

noisedifference = norm (actualdisplacement-actualdisplacementcomp)/norm
(actualdisplacement)

% for i = 1:totalelements
%       effActualStrains( i,1 ) = sqrt( actStrains( i,1 )^2 + actStrains( i,2 )^2 + 0.5 *
actStrains( i,3 )^2 );
% end
%
% clear actStrains;
```

```
clear Eoffspring;

[ Eoffspring ] = initialize( pop( 1 ),n,minValue,maxValue );
%plot (Eoffspring','.');
save Eoffspring.mat Eoffspring;
%
for iters = 1:size( pop,2 );

    if iters == 1
        numParents = pop( iters );
    else

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        % CREATE OFFSPRING FROM PARENTS

        clear Eoffspring; clear Eparents; clear survival;

        % get back raw (unnormalized) numbers
        for j = 1:groupnumber
            Efitness( :,n+1+j ) = Efitness( :,n+1+j );%* normalizer( j );
        end

        numParents = pop( iters ) / 2;
        numCross = numParents;
%           for i = 1:pop( iters )
%
%               survivalCurve = 1;
%
%               % size of survival: survival( 1 ) -> survival( pop( iters ) + 1 )
%               survival( 1 ) = 0;
%                     survival( pop( iters ) - ( i - 2 ) ) = i^survivalCurve /
sum( ( 1:1:pop( iters ) ).^survivalCurve );
%
%           end

        for i = 1:numParents
%               for j = 2:pop( iters ) + 1
%                   r = rand;
%                           if r < 0.5 + ( sum( survival( 1:j ) ) )/2 & r > 0.5 -
( sum( survival( 1:j ) ) )/2;
                       Eparents( i,1:groupnumber+n+1 ) = Efitness( i,1:groupnumber+n+1 );
%                          conv = 1;
%                      end
%                  end
          end
          % Best in Population must move on no matter what!
%                                   Eparents(    numParents,1:groupnumber    +    n+1    )    =
Efitness( 1,1:groupnumber+n+1 );

          % Eparents( 1:numParents,: ) = Efitness( 1:numParents,: );

          [ Eoffspring ] = randCrossover( n,Eparents,numCross );

          %%%%%%%%%%%%%%%%%Mutation    with    value    colse    to    the
best%%%%%%%%%%%%%%%
%                      for jj = 1:numMutation
%                 for j=1:n
%                                               Eoffspring    (round(rand*(numCross-3))+3,j)    =
( rand*( mean(Efitness(:,j))*1.4 - mean(Efitness(:,j))*0.7 ) + mean(Efitness(:,j))*0.7 );%values
in 0.7-1.4 Efitness(1,:) bound
%                 end
%                      end

%              for jj = 1:numMutation
%                 for j=1:n
%                 Eoffspring  (round(rand*(numCross-3))+3,j)  =  (  rand*(  Efitness(1,j)*1.3  -
Efitness(1,j)*0.7 ) + Efitness(1,j)*0.7 );%values in 0.7-1.4 Efitness(1,:) bound
%                 end
%            end
%                    for jj = 1:numMutation
%                 for j=1:n
%                 Eoffspring  (round(rand*(numCross-3))+3,j)  =  (  rand*(  Efitness(2,j)*1.3  -
Efitness(2,j)*0.7 ) + Efitness(2,j)*0.7 );%values in 0.7-1.4 Efitness(1,:) bound
%                 end
%            end

%
%                           for jj = 1:numMutation
%                for j=1:n
%               Eoffspring (round(rand*(numCross-3))+3,j) = ( rand*( mean(Efitness(:,j))*1.2 -
mean(Efitness(:,j))*0.8) + mean(Efitness(:,j))*0.8);%values in 0.7-1.4 Efitness(1,:) bound
%                end
%          end
%
%
% %            for jj = 1:numMutation
% %                 for j=1:n
% %                Eoffspring (round(rand*(numCross-3))+3,j) = ( rand*( Efitness(1,j)*1.2 -
Efitness(1,j)*0.8 ) + Efitness(1,j)*0.8 );%values in 0.7-1.4 Efitness(1,:) bound
% %                end
% %            end
% %
%                    for jj = 1:numMutation
%                 for j=1:n
%                Eoffspring (round(rand*(numCross-3))+3,j) = ( rand*( Efitness(1,j)*1.1 -
Efitness(1,j)*0.9 ) + Efitness(1,j)*0.9 );%values in 0.7-1.4 Efitness(1,:) bound
%                end
%          end
%                  %%%%%%%%%%%%%%%%%Mutation again with values off the
best%%%%%%%%%%%%%%
                  for jj = 1:numMutation
               for j=1:n
              Eoffspring (round(rand*(numCross-1))+1,j) = ( rand*( maxValue(j) -
minValue(j) ) + minValue(j) );
               end
          end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
    end

    for i = 1:numParents


        filenumber = num2str( i );
        filedot = '.in';
        fileprefix = strcat( iterfilename, filenumber );
        infilename = strcat( iterfilename, filenumber, filedot );

        % Writes .in file
        createInFileAutomated( n, Eoffspring( i,: ), infilename, fileprefix, skeletonFile,
GAdir );

        %% [,] = concatenation
        %% Create .dat file
        dos( ['"C:\Program Files\ADINA\ADINA System 8.1\bin\aui.exe" -b -m 200mb ',
infilename] );
        %% Run simulation
        dos( ['"C:\Program Files\ADINA\ADINA System 8.1\bin\adina.exe" -b -s -m 200mb ',
fileprefix] );

        porfilename = strcat( fileprefix,'.por');
        datfilename = strcat( fileprefix,'.dat');
        resfilename = strcat( fileprefix,'.res');
        modfilename = strcat( fileprefix,'.mod');
        infilename    = strcat( fileprefix,'.in');
        dos ( ['del ',porfilename]);
        dos ( ['del ',datfilename]);
        dos ( ['del ',resfilename]);
        dos ( ['del ',modfilename]);
        dos ( ['del ',infilename]);

    end

    for i = 1:numParents
        filenumber = num2str( i );
        filedot = '.out';
        outfilename = strcat( iterfilename, filenumber, filedot );

        [ YZstrains ] = readOutFile( outfilename, timestep,elements);




        dos( ['del ',outfilename]);
        % Make 1D vector of YY and ZZ strains

        clear effPredStrains;


        if size(actStrains,1)*3 ~= size(YZstrains,1)*3
            false =1;
        else false = 0;
        end

        if false ==1
            clear prddisplacement;
            prddisplacement = actualdisplacement *1e4;
%              prddisplacement = prddisplacement';
        else

 prddisplacement = reshape (YZstrains,size(actStrains,1)*3,1);
end

        diff = prddisplacement - actualdisplacement;

        diff( 1:elements( 1 ) ) = diff( 1:elements( 1 ) ) / elements( 1 );


        for j = 2:groupnumber

            diff( sum( elements( 1:j-1 ) ) + 1 : sum( elements( 1:j ) ) ) =...
                diff( sum( elements( 1:j-1 ) ) + 1 : sum( elements( 1:j ) ) ) / elements( j );

        end

        % rawFitness( i ) = norm( diff );
        Eoffspring( i,n+1 ) = 0;
        % Eoffspring( i,n+2 ) = norm( diff );

        Eoffspring( i,n+2 ) = norm( diff( 1:elements( 1 ) ) );
        for j = 2:groupnumber

            Eoffspring( i,n+1+j ) = norm( diff( sum( elements( 1:j-1 ) )+1 :
sum( elements( 1:j ) ) ) );
```

```
        end
    end

    clear YZstrains;

    clear Efitness;

    if iters == 1
        Efitness = Eoffspring;
    else
        Efitness( 1:numParents,: ) = Eoffspring( 1:numParents,: );
        Efitness( numParents+1:pop( iters ),: ) = Eparents( 1:numParents,: ); %
    end

    % Normalize

    % Normalize!
%      clear normalizer;
%      for j = 1:groupnumber
%          normalizer( j ) = sum( Efitness( :,n+1+j ) );
%      end
    % normalizer( n+1 ) = sum( Efitness( :,2*n+2 ) );

    % Efitness( 1:numParents,2*n+2 ) = Efitness( 1:numParents,2*n+2 ) / normalizer( n+1 );

%      for j = 1:groupnumber
%          % Efitness( :,n+2 ) = Efitness( :,n+2 ) / normalizer( n+1 );
%          Efitness( :,n+1+j ) = Efitness( :,n+1+j ) / normalizer( j );
%      end

    %        for j = 1:n
    %            alpha( j ) = 1;
    %        end
    %
%      if iters == 1
        alpha( 1 ) = 1/3;
        alpha( 2 ) = 1/3;
        alpha( 3 ) = 1/3;
%          alpha( 1 ) =alpha( 1 )/ sum (alpha);
%            alpha( 2) =alpha( 2 )/ sum (alpha);
%            alpha( 3 ) =alpha( 3 )/ sum (alpha);



%            alpha( 4 ) = 0.10;
%            alpha( 5 ) = 0.10;
%            alpha( 6 ) = 0.10;
%       else
          for j = 1:groupnumber
                alpha( j ) = 1;
           end
%       end
    %       else
    %            alpha( 1 ) = 0.6;
    %            alpha( 2 ) = 0.3;
    %            alpha( 3 ) = 0.1;
    %       end
    %
    % Weighted sum of each parameter's normalized fitness value
    Efitness( :,groupnumber+n+2 ) = zeros;
     for j = 1:groupnumber
          Efitness( :,groupnumber+n+2 ) = Efitness( :,groupnumber+n+2 ) +
alpha( j )*Efitness( :,n+1+j );
     end

%       % Normalize
%       clear sumNormalizer;
%       sumNormalizer = sum( Efitness( :,groupnumber+n+2 ) );
%
%       Efitness( :,groupnumber+n+2 ) = Efitness( :,groupnumber+n+2 ) / sumNormalizer;
%


    [ sortedRawFitness, Eindex ] = sort( Efitness( :,groupnumber+n+2 ) );
    for i = 1:pop( iters )
        Etemp( i,: ) = Efitness( Eindex( i ),: );
    end

    Efitness = Etemp;
    clear Etemp;
```

```
        iters
        Efitness(1:2,:)'
        mean (Efitness(1:2,:))';
        Efitnessrecord(iters,:)=Efitness(1,:);
        Efitnessrecord'
        save;
%
%        % Efitness( 1,: )'
%         plot (Efitness','+');
%
%                 iternumber = num2str( iters );
%
%
%               Efitnessname = strcat( 'Efitness', iternumber,'.mat' );
%
%               save Efitnessname Efitness;
%
%         save;



%         if iters == 1
%               E( 1:pop( iters ),: ) = Efitness( 1:pop( iters ),: );
%         else
            E( sum( pop( 1:iters-1 ) ) + 1 : sum( pop( 1:iters ) ),: ) = Efitness( 1:pop( iters ),: );
%         end

      % Hardwire the independent parameter!
%         if iters == 1
% %          for j = 1:n
% %                 if alpha( j ) >= 0.7
%                       % Efitness( :,j ) = Efitness( 1,j )
%                       clear Eoffspring;
%                       Eoffspring = Efitness;
%                       % end
%         end
%

end
```

```
    Solutionrecord(mmm,:)=Efitness(1,:);

    save 'Solutionrecord.mat' Solutionrecord
    mmm=mmm
    Solutionrecord'


end

% Econverged = Efitness( 1,: )
 save '0noise.mat';
%
%
%
% [mu,sigma,muci,sigmaci]=normfit(Efitness(:,1))

function [ strains, false] = readOutFile (filename, timestep,elements);%( n,filename)%,
elements ); %,sample,sampleSize );



totalelements = sum( elements );
false = 0;

fid = fopen( filename );
if fid == -1


    error( 'File not found or permission denied' );
end

readfalse = 0;


strains = [];
```

```
iterate = 0;

while( iterate == 0 )
    buffer = fgetl( fid );
    iterate = strncmp( buffer, ' S T R E S S     C A L C U L A T I O N S', 38 );

    if buffer == -1,
        false = 1;   break;
    end

end

strainHeader = 12;
for ii = 1:strainHeader, buffer = fgetl( fid ); end

for ii = 1:totalelements*timestep

    totalStrainYY = 0;
    totalStrainZZ = 0;
    totalStrainYZ = 0;
    buffer = fgetl( fid );
    buffer = fgetl( fid );
    for iii = 1:7
```

```
if buffer == -1,
    false = 1;    break;
end
[ strainxx_string, buffer ] = strtok( buffer );
[ strainxx_string, buffer ] = strtok( buffer );
[ strainxx_string, buffer ] = strtok( buffer );

if buffer == -1,
    false = 1;    break;
end
readfalsex = isempty (strainxx_string);

[ strainyy_string, buffer ] = strtok( buffer );
readfalsey = isempty (strainyy_string);

[ strainzz_string, buffer ] = strtok( buffer );
readfalsez = isempty (strainzz_string);

[ strainyz_string, buffer ] = strtok( buffer );




readfalse = 1-(readfalsex)*(readfalsey)*(readfalsez);
if readfalse ~=0
    strainyy = str2num( strainyy_string );
    strainzz = str2num( strainzz_string );
    strainyz = str2num( strainyz_string );
    totalStrainYY = totalStrainYY + strainyy;
    totalStrainZZ = totalStrainZZ + strainzz;
    totalStrainYZ = totalStrainYZ + strainyz;

    buffer = fgetl( fid );
    buffer = fgetl( fid );
    buffer = fgetl( fid );
    buffer = fgetl( fid );
```

```
            readfalsexx = isempty (str2num( strainxx_string ));
            if readfalsexx ==0



        strains( ii,1 ) = str2num( strainxx_string );
        strains( ii,2 ) = str2num( strainyy_string );
        strains( ii,3 ) = str2num( strainzz_string );



    end


    else

        iterate = 0;



        while( iterate == 0 )
            buffer = fgetl( fid );
            iterate = strncmp( buffer, ' S T R E S S     C A L C U L A T I O N S', 38 );


            if buffer == -1,
                false = 1;    break;
            end



        end


        strainHeader = 12;
        for ii = 1:strainHeader, buffer = fgetl( fid );
        end
```

```
totalStrainYY = 0;
totalStrainZZ = 0;
totalStrainYZ = 0;
buffer = fgetl( fid );
buffer = fgetl( fid );


if buffer == -1,
    false = 1;    break;
end
[ strainxx_string, buffer ] = strtok( buffer );
[ strainxx_string, buffer ] = strtok( buffer );
[ strainxx_string, buffer ] = strtok( buffer );

if buffer == -1,
    false = 1;    break;
end


[ strainyy_string, buffer ] = strtok( buffer );


[ strainzz_string, buffer ] = strtok( buffer );


[ strainyz_string, buffer ] = strtok( buffer );





strainyy = str2num( strainyy_string );
strainzz = str2num( strainzz_string );
strainyz = str2num( strainyz_string );
totalStrainYY = totalStrainYY + strainyy;
totalStrainZZ = totalStrainZZ + strainzz;
totalStrainYZ = totalStrainYZ + strainyz;

buffer = fgetl( fid );
```

```
            buffer = fgetl( fid );
            buffer = fgetl( fid );
            buffer = fgetl( fid );









            strains( ii,1 ) = str2num( strainxx_string );
            strains( ii,2 ) = str2num( strainyy_string );
            strains( ii,3 ) = str2num( strainzz_string );


        end
    end
end



fclose( fid );


function [ Eoffspring ] = randCrossover( n,Eparents,numCross )

numOffspring = size( Eparents,1 );

for i = 1:numCross

    crossPartner( i ) = round( rand*( numOffspring - 0.01 ) + 0.5 );


    while crossPartner( i ) == i
        crossPartner( i ) = round( rand*( numOffspring - 0.01 ) + 0.5 );
    end


    crossNo = round( rand*( n-1-0.01 ) + 0.5 );
    for j = 1:crossNo

        crossLoc( j ) = round( rand*( n-0.01 ) + 0.5 );
        if j > 1
            notYet = 0;
            while notYet == 0;
                for c = 1:j-1
                    if crossLoc( j ) == crossLoc( j-c )
                        crossLoc( j ) = round( rand*( n-0.01 ) + 0.5 );
                        break;
                    end
                    notYet = 1;
                end

            end

        end

        if crossLoc( j ) == 1;
            Eoffspring( i,crossLoc( j ) ) = Eparents( crossPartner( i ),crossLoc( j ) );
            Eoffspring( i,crossLoc( j )+1:n ) = Eparents( i,crossLoc( j )+1:n );
        elseif crossLoc( j ) == n
            Eoffspring( i,crossLoc( j ) ) = Eparents( crossPartner( i ),crossLoc( j ) );
            Eoffspring( i,1:crossLoc( j )-1 ) = Eparents( i,1:crossLoc( j )-1 );
        else
            Eoffspring( i,crossLoc( j ) ) = Eparents( crossPartner( i ),crossLoc( j ) );
            Eoffspring( i,1:crossLoc( j )-1 ) = Eparents( i,1:crossLoc( j )-1 );
            Eoffspring( i,crossLoc( j )+1:n ) = Eparents( i,crossLoc( j )+1:n );
        end
    end

end

function [ Eoffspring_init ] = initialize( initPop,n, minValue,maxValue)

% for i = 1 : initPop*n/n
%        Eoffspring_init( i,1 ) = ( rand*( 117600 ) + 39200 ); % * 10^5;
% end
% for i = 1 : initPop*n/n
%        Eoffspring_init( i,2 ) = ( rand*4800000 + 1600000 ); % * 10^7;
% end
% for i = 1 : initPop*n/n
%        Eoffspring_init( i,3 ) = ( rand*480 + 160 ); % * 10^3;
% end

% e1 = 4e3;
% e2 = 4e3;
% e3 = 1e2;
% fac=0.4;
% facmax=4;
%
% e1min = e1 - e1*fac;
% e1max = e1 + e1*facmax;
% e2min = e2 - e2*fac;
% e2max = e2 + e2*facmax;
% e3min = e3 - e3*fac;
% e3max = e3 + e3*facmax;

for j=1:n   %6 subject to change when the number of parameter increases
for i = 1 : initPop
     Eoffspring_init( i,j ) = ( rand*( maxValue(j) - minValue(j) ) + minValue(j) ); % * 10^5;
end
end
% for i = 1 : initPop*n/n
%        Eoffspring_init( i,2 ) = ( rand*( maxValue(2) - minValue(2) ) + minValue(2) ); % *
10^7;
% end
% for i = 1 : initPop*n/n
%         Eoffspring_init( i,3 ) = ( rand*( maxValue(3) - minValue(3) ) + minValue(3) ); % *
10^3;
%    end




%
% i = 2;
% sample( 1 ) = round( rand*(n*initPop-1) + 1 );
% while( i <= n*initPop )
%       sample( i ) = round( rand*(n*initPop-1) + 1 );
%           for ii = 1:i-1
%               if sample( i ) == sample( ii )
%                   i = i-1;
%                   break
%               end
%           end
%           i = i+1;
%       end
% └ └ └
% j = 1;
% for i = 1:n:n*initPop
%       Eoffspring_init( j,1 ) = random( sample( i ) );
%       Eoffspring_init( j,2 ) = random( sample( i+1 ) );
%       Eoffspring_init( j,3 ) = random( sample( i+2 ) );
%       j = j+1;
% end


function createInFileAutomated ( n, E, filename, fileprefix, skeleton, GAdir )
groupnumber = 3;
% D2(1)=8.36500000000000;
% D2(2)=13.0000000000000;
% D2(3)=5.00000000000000;
Kappa(1)=2.21229155000000E+07;
Kappa(2)=2.21229155000000E+07;
Kappa(3)=25000.000000000;
% Open the files.   If this returns a -1, we did not open the files
% successfully.
%KKK=8
fidR = fopen( skeleton, 'r' );
if fidR == -1
   error( 'File not found or permission denied' );
end

fidW = fopen( filename, 'w' );
if fidW == -1
   error( 'File not found or permission denied' );
end
```

```
iterate = 0;
lineNum1 = 0;
while ( iterate == 0 )
     buffer = fgetl( fidR );
     iterate = strncmp( buffer, 'MATERIAL MOONEY', 15);
     lineNum1 = lineNum1 + 1;
end

frewind( fidR );

for i = 1 : ( lineNum1 - 1 )
     buffer = fgetl( fidR );
     fprintf( fidW, '%s\n', buffer );
end

for i = 1 : groupnumber

     % Get the 'Material Elastic Name...' line from the skeleton file
     buffer = fgetl( fidR );
     % Get the 'Density...Alpha' line from the skeleton file
     buffer = fgetl( fidR );
     buffer = fgetl( fidR );
     buffer = fgetl( fidR );
     buffer = fgetl( fidR );
     buffer = fgetl( fidR );
     EndCommandBuffer = fgetl( fidR );
     %EndCommandBuffer1 = fgetl( fidR );

end

for i = 1 : groupnumber     %3 subject to change when # parameter increases
     fprintf( fidW, 'MATERIAL MOONEY-RIVLIN NAME=%g C1=0.00000000000000
C2=0.00000000000000,\n C3=0.00000000000000 C4=0.00000000000000
C5=0.00000000000000,\n C6=0.00000000000000 C7=0.00000000000000
C8=0.00000000000000,\n C9=0.00000000000000 D1=%14.14e D2=%14.14e,\n
KAPPA=%14.14e DENSITY=0.00000000000000 FITTING-=0,\n', i,
E(i),E(groupnumber+i),Kappa(i) );
     %3+i subject to change when number of parameter increases
     fprintf( fidW, '%s\n', buffer );
     fprintf( fidW, '%s\n', EndCommandBuffer );
```

```
    %fprintf( fidW, '%s\n', EndCommandBuffer1 );
end

iterate = 0;

while ( iterate == 0 )
    buffer = fgetl( fidR );
    fprintf( fidW, '%s\n', buffer );
    iterate = strncmp( buffer, 'ADINA OPTIMIZE=SOLVER FILE=,', 28);
end

% Get the next line
buffer = fgetl( fidR );
%fprintf( fidW, '%s\n', buffer );
%buffer = fgetl( fidR );

fprintf( fidW, '"C:\\genetic_algorithm_2d\\%s.dat",\n', fileprefix );

buffer = fgetl( fidR );
while ( buffer ~= ( -1 ) )
    fprintf( fidW, '%s\n', buffer );
    buffer = fgetl( fidR );
end
% KKK=9
fclose( fidR );
fclose( fidW );
```