

A TIME-COURSE STUDY OF THE ULTRASONIC PROPAGATION PROPERTIES OF  
ENZYMATICALLY DEGRADED ARTICULAR CARTILAGE USING THE  
SCANNING LASER ACOUSTIC MICROSCOPE AT 100 MHZ

BY

DAWN MARIA CHAMBERS

B.S., University of Illinois, 1992

THESIS

Submitted in partial fulfillment of the requirements  
for the degree of Master of Science in Electrical Engineering  
in the Graduate College of the  
University of Illinois at Urbana-Champaign, 1994

Urbana, Illinois

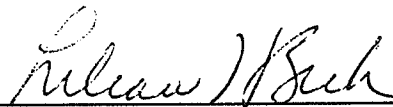
# UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

## GRADUATE COLLEGE DEPARTMENTAL FORMAT APPROVAL

THIS IS TO CERTIFY THAT THE FORMAT AND QUALITY OF PRESENTATION OF THE THESIS SUBMITTED BY DAWN MARIA CHAMBERS AS ONE OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE ARE ACCEPTABLE TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING.

JULY 14, 1994

*Date of Approval*

  
\_\_\_\_\_  
*Departmental Representative*

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

THE GRADUATE COLLEGE

JULY 1994

WE HEREBY RECOMMEND THAT THE THESIS BY

DAWN MARIA CHAMBERS

ENTITLED A TIME-COURSE STUDY OF THE ULTRASONIC PROPAGATION PROPERTIES  
OF ENZYMATICALLY DEGRADED ARTICULAR CARTILAGE USING THE  
SCANNING LASER ACOUSTIC MICROSCOPE AT 100 MHZ

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR  
THE DEGREE OF MASTER OF SCIENCE

*W. J. Beier*

Director of Thesis Research

*Anthony M. Frick*

Head of Department

Committee on Final Examination†

Chairperson

† Required for doctor's degree but not for master's.

## DEDICATION

This thesis is dedicated to my mother, Nancy Scanavino. You taught me that true strength comes from within and that perseverance guides the way to goal attainment. Your teachings have allowed me to accomplish my goals, thus far, and will lead the way to my future. Thank you for your guidance and your love.

## ACKNOWLEDGMENTS

I would like to thank my advisor, Professor William D. O'Brien, Jr., for his encouragement, support, and never-ending optimism.

In addition, I would like to thank my fellow graduate students and the staff of the Bioacoustics Research Laboratory for their support.

## TABLE OF CONTENTS

CHAPTER	PAGE
1 INTRODUCTION.....	1
2 EQUIPMENT.....	5
2.1. SLAM.....	5
2.2. DATA ACQUISITION.....	7
3 ULTRASONIC PROPERTIES.....	11
3.1. ULTRASONIC ATTENUATION COEFFICIENT.....	11
3.2. ULTRASONIC SPEED.....	14
3.3. HETEROGENEITY.....	18
4 METHODOLOGY.....	20
4.1. EXPERIMENTAL PROTOCOL AND PROCEDURES.....	20
4.2. STATISTICAL PROCEDURES.....	24
5 RESULTS AND DISCUSSION.....	25
5.1. RESULTS.....	25
5.1.1. ATTENUATION.....	25
5.1.2. SPEED.....	26
5.1.3. NORMALIZED HETEROGENEITY INDEX.....	38
5.2. DISCUSSION.....	45
APPENDIX A ATTENDMC PROGRAM LISTING.....	55
APPENDIX B GETIMG PROGRAM LISTING.....	60
APPENDIX C REGRESS PROGRAM LISTING.....	64
APPENDIX D SPEEDN PROGRAM LISTING.....	79
APPENDIX E SPEED DATA FROM THE SLAM.....	91

APPENDIX F <HI> DATA FROM THE SLAM.....	95
REFERENCES.....	99

## CHAPTER 1

### INTRODUCTION

Ultrasonic methods are available which allow noninvasive imaging and assessment of the acoustic properties of tissues. The methods have played a role in evaluation of wound healing [1-4], assessment of damaged heart tissue [5-7], and detection of osteoarthritis [8-14]. In order to make diagnoses, it is important that ultrasonic properties be correlated to material properties or the state of the tissue under study.

The use of ultrasound for the evaluation of wounds depends upon whether the measured acoustic properties correlate with tensile strength, morphology, and biochemical properties of wounds at various times during healing. Studies have been conducted on the ultrasonic properties of normal skin and wounds [1-4]. The ultrasonic speed and attenuation coefficient were directly correlated with tissue collagen concentration and inversely correlated with tissue water concentration [1, 4].

The ultrasonic backscatter coefficient has also been used to determine whether infarct, ischemic, and normal regions of myocardial tissue could be differentiated [6] and to investigate its quantitative relationship to collagen concentration in the myocardium [5]. In addition, a study by Sagar et al. [7] assessed the acoustic properties of normal and ischemic myocardium and correlated the changes with ultrasonic backscatter. The studies found that the damaged and normal tissue areas could be distinguished based upon the acoustic properties and backscatter coefficient.

Evaluation of osteoarthritic cartilage is another important application of ultrasound. Ultrasonographic methods exist which allow noninvasive imaging of



the knee to assess the thickness and integrity of the articular cartilage [10, 11]. Fibrillations, which are among the earliest changes in articular cartilage [8, 9], can also be imaged and quantitatively assessed using ultrasonic backscatter methods [12-14]. The ultrasonic propagation properties have also been investigated [15, 16]. Agemura and O'Brien [15] found that the ultrasonic speed was highly dependent on the zone in which it was assessed.

The matrix of articular cartilage exists as a framework consisting, primarily, of type II collagen. The morphology of the articular cartilage is anisotropic, as shown in Fig. 1. In the outer zone of the cartilage structure, known as the tangential zone, the collagen fiber bundles are oriented parallel to the articular surface and are tightly packed. The tangential zone comprises approximately 20% of the thickness while the next region, the transitional zone, represents about 30%. The transitional zone, where the collagen fiber bundle orientations are somewhat random, represents a gradual transition from one preferred orientation to the other. A note should be made that the boundaries between each of the regions are not as distinct as portrayed in Fig. 1. Beneath the transitional zone is the radial zone which represents the remainder of the articular cartilage. In the radial zone, the collagen fiber bundles are aligned perpendicularly to the articular surface and are not packed as tightly as in the tangential zone [17-19].

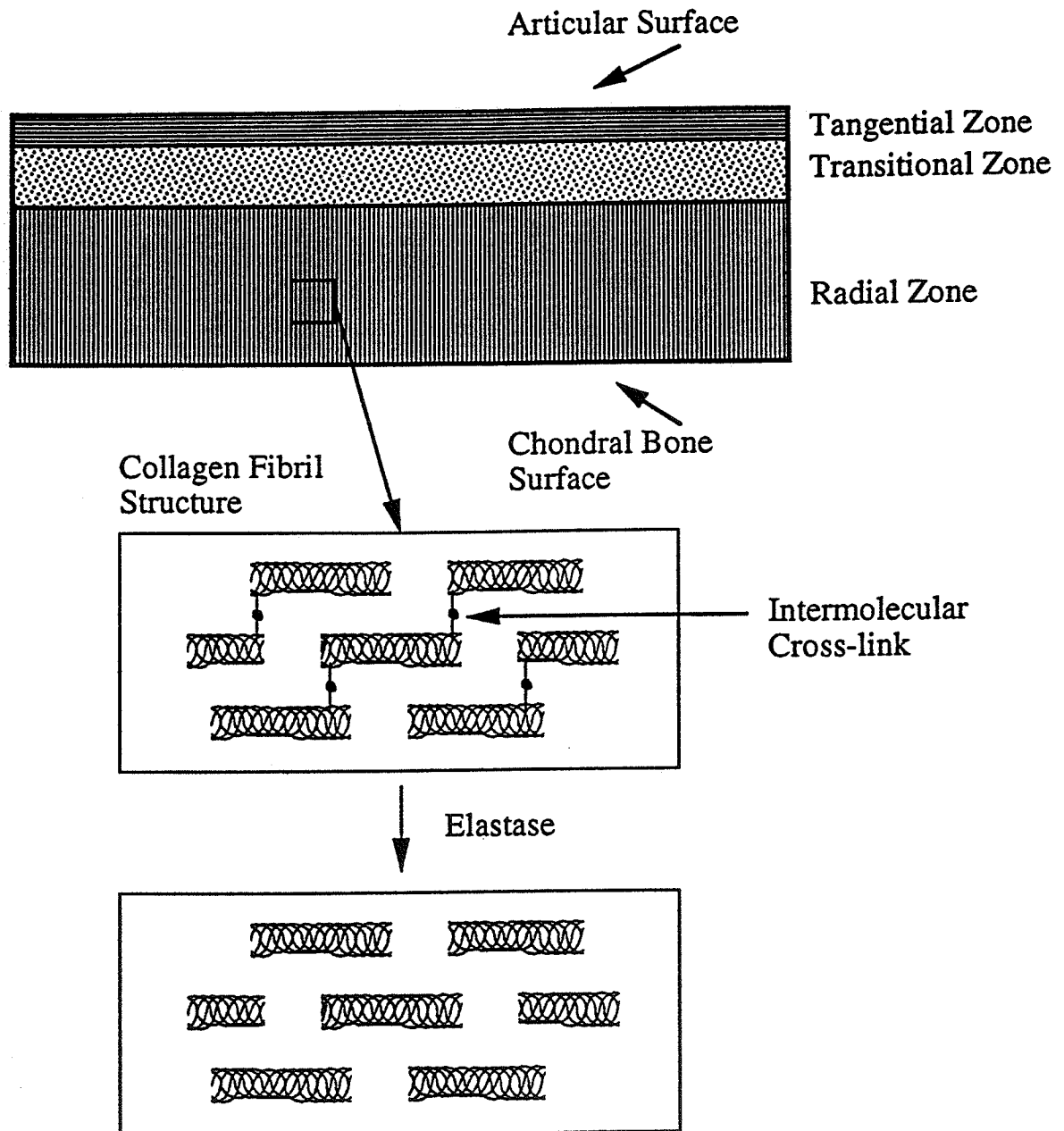


Figure 1. Diagram of the predominant orientations of collagen fibrils in different zones of articular cartilage. The lower panels illustrate the head-to-tail locations of cross-links between collagen molecules packed in fibrils and how the cross-links are disrupted by the telopeptidase activity of pancreatic elastase.

Enzymatic methods have been described which degrade the nonhelical terminal peptides of collagen molecules, thus destroying the intermolecular cross-links [20]. In addition, methods have been presented which selectively extract glycosaminoglycans, proteoglycans, and other noncollagen proteins from cartilage [21]. Agemura [15, 22] used a 100 MHz scanning laser acoustic microscope to study the effects of enzymatic degradation on the acoustic properties of articular cartilage. Cleaving the collagen cross-links resulted in significant changes in attenuation coefficient and the speed in the radial zone was statistically different from that in the transitional and tangential zones. The number of observations presented for the study was small; data were taken at only two times: before and after treatment. As a result, the study contained herein was designed to report the variations in the acoustic properties of articular cartilage during an entire 24-hour enzymatic degradation cycle.

## CHAPTER 2

### EQUIPMENT

#### 2.1. SLAM

The data taken for this experiment were gathered using a scanning laser acoustic microscope (Sonomicroscope 100<sup>®</sup>, Sonoscan, Inc., Bensenville, IL 60106) operating at 100 MHz. Ultrasonic properties, such as attenuation, speed, heterogeneity index, and normalized heterogeneity index, can be assessed from SLAM images. Details of the operating techniques for this instrument have been reported previously [23, 24], but an outline will be presented.

The SLAM is shown in block diagram form in Fig. 2. A layer of distilled water is placed on the fused silica stage. Then, a thin plastic sheet along with a ring spacer is laid on the stage. The sample and a few drops of buffer are placed in the ring spacer and covered with a semireflective coverslip. A piezoelectric transducer operating at 100 MHz is used to insonify the sample. The sound passes through the stage at 45° and refracts into the water at about 10°. As the sound propagates through the sample, a dynamic ripple, whose amplitude is proportional to the transmitted acoustic pressure, is created on the lower surface of the coverslip. A focused scanning laser beam is used to detect the ripple.

When the laser light strikes the gold-plated coverslip, a fraction of the light is transmitted through the sample since the coverslip is semitransparent and the rest is reflected. The portion of the laser light that passes through the sample is detected with a photodiode and is used to generate an optical image. The rest of the light is reflected at an angle that is dependent on the amplitude of the dynamic ripple at that position.

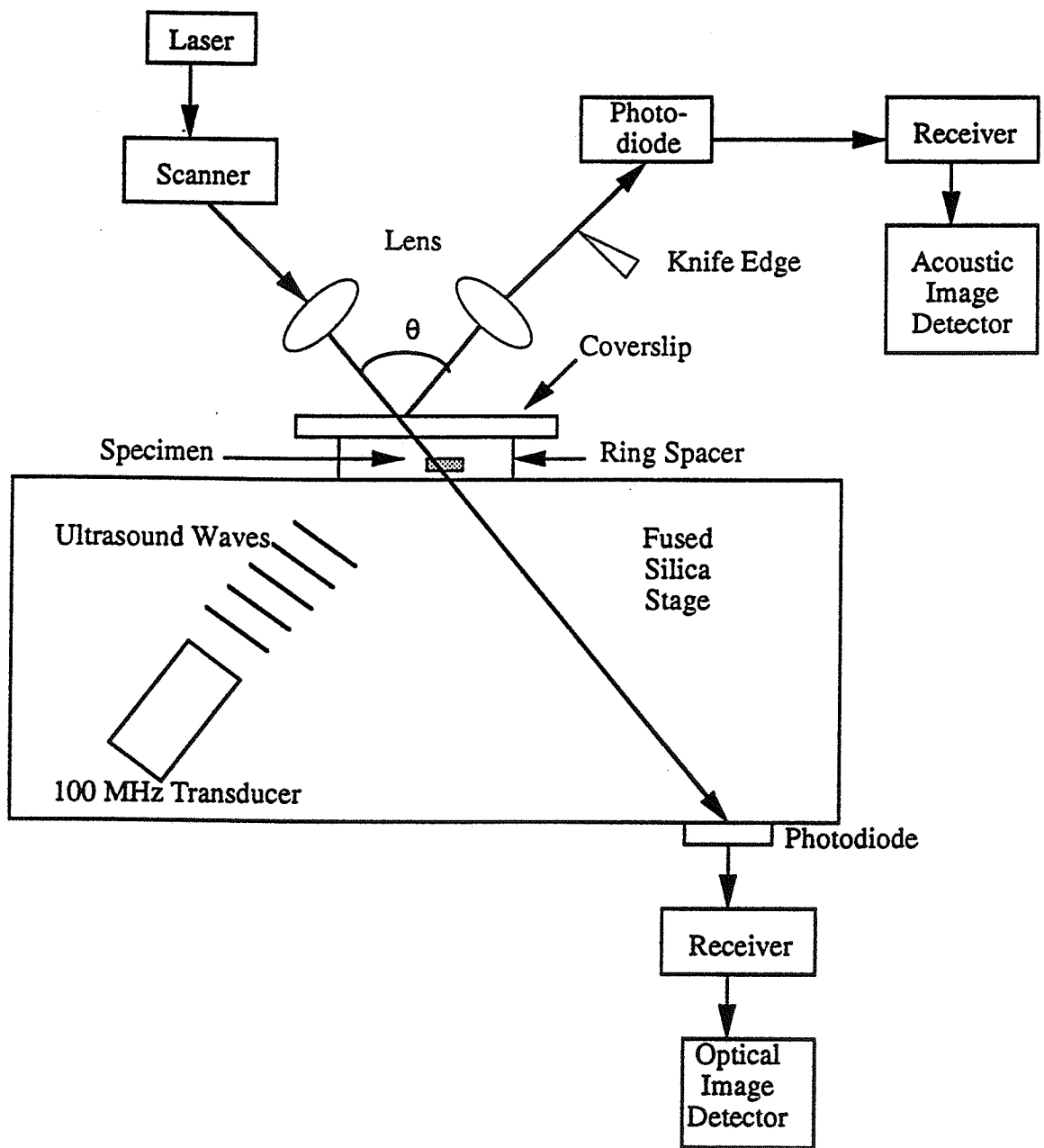


Figure 2. Block diagram of the SLAM.

As seen in the diagram, a knife edge is placed in the path of the reflected light to block part of the light reaching the photodiode. The amount of light blocked depends on the angle of the reflected laser light. This allows the angular modulation to be converted to light intensity modulation that can then be converted into an electrical signal proportional to the ripple amplitude. The SLAM processes the electrical signal to generate either an acoustic amplitude image or an interference line image. These images can be displayed on the SLAM monitor or on the Panasonic video monitor, model WV-5470.

The acoustic amplitude image shows the spatial amplitude of the acoustic field after it has passed through the specimen. This image is used to determine the attenuation coefficient of the specimen. When displayed on the monitor, the brighter areas correspond to areas of higher ultrasonic energy, while the darker areas indicate areas of lower ultrasonic energy.

The interference line image or interferogram is created by adding a reference signal, phase coherent with the transmitted acoustic wave, to the received signal. The image contains light and dark bands or fringes that indicate equiphase wavefronts of the ultrasonic field. The fringes shift to the right if the ultrasonic speed is increasing or to the left if the speed is decreasing.

## 2.2. Data Acquisition

Computer acquisition and analysis of the data from SLAM images provide quantitative determination of ultrasonic parameters such as attenuation, speed, heterogeneity index, and normalized heterogeneity index more quickly than could be done manually. The predecessor to the current data acquisition system is shown in block diagram form in Fig. 3 [25]. The Z-80 microprocessor controlled system was replaced for several reasons. It used both a minicomputer and a

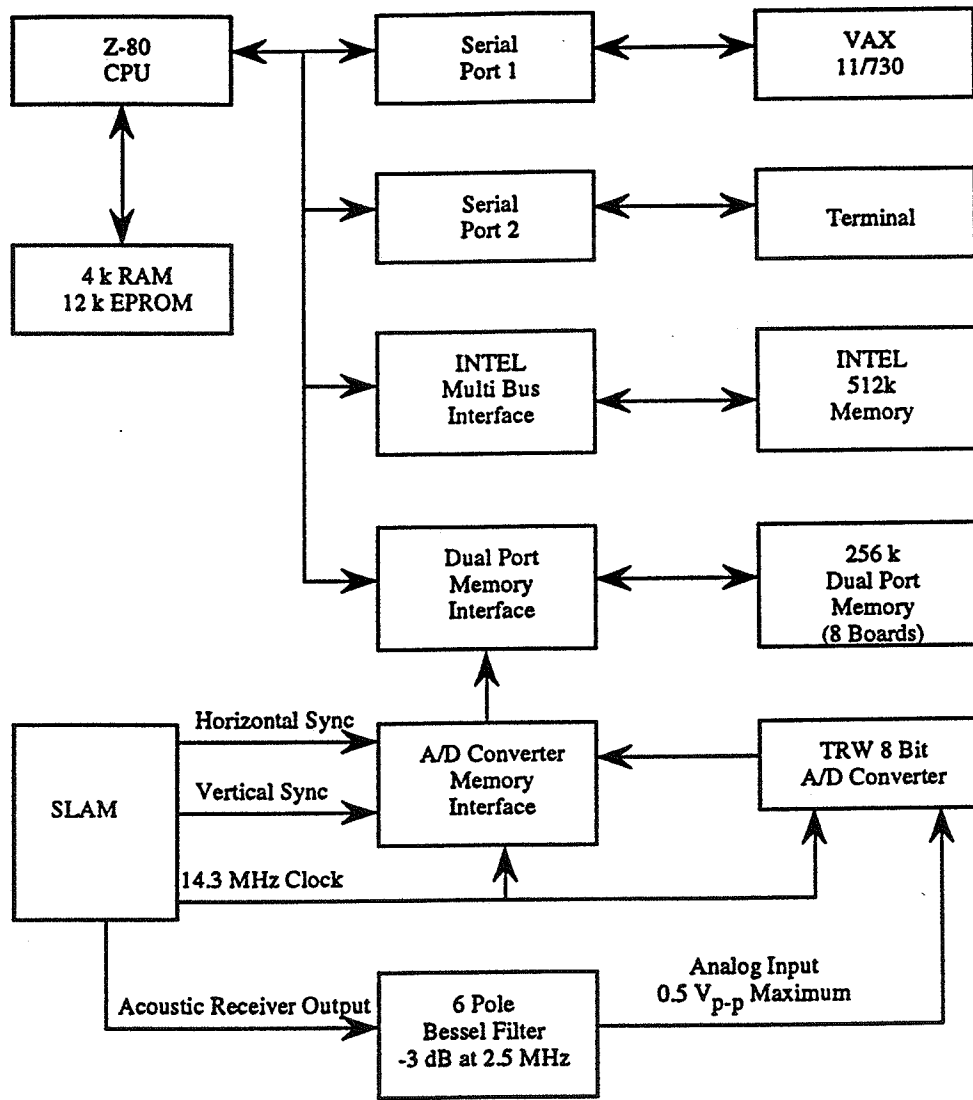


Figure 3. Block diagram of the SLAM data acquisition system used by Steiger.

mainframe which could provide reasonably fast computation times, but the transfer of data from the data acquisition system to the computer caused a bottleneck. Also, the mainframe was a time-shared system, and performance was considerably slower during peak usage hours. Finally, the analysis of the speed and attenuation coefficient data was performed after completing a SLAM session and was done off-line. This meant that the user was unaware of any problems that may have existed in the data until after the session was completed. It was then decided that a new system was needed.

Nicozisin designed the new system [26] which has undergone only a few modifications. One of the modifications was to upgrade the IBM Personal Computer AT to a 486, 50 MHz personal computer, Professional PC. The other modifications have been to the software to make it compatible with the new computer and to enhance certain functions. An overview of the current system will be presented here.

A frame grabber is used to digitize and store the video images. The system uses the Data Translation DT2851 [27] frame grabber which fits into an expansion slot of the computer. The frame grabber can then communicate with the computer through standard PC I/O registers. The DT2851 uses a standard composite video signal as an input and acquires one full frame which yields an image of 512 pixels by 480 lines.

A Data Translation DT2858 frame processor board [28] is also used which increases the speed of some imaging functions. The DT2858 fits into another expansion slot and plugs into the computer through the standard I/O registers. Instead of connecting through the computer, the frame processor connects directly to the DT2851 through a pair of 10 MHz asynchronous parallel I/O ports and allows high-speed transfer of images between the two devices. The main functions that the DT2858 performs for the system are frame averaging and



histograms. A histogram represents the distribution of pixel values in an image and is computed by incrementing a counter for each gray level value every time that a level is encountered in an image.

The software package from Data Translation, DT-IRIS [29], was used to simplify communications and data transfers between the frame grabber and frame processor. DT-IRIS is a library of subroutines written to access the Data Translation hardware. The subroutines are called as functions from programs written with the computer language C [30]. A listing of the programs used for data acquisition and analysis, which have been modified from the originals by Nicozisin [26], is contained in Appendices A - D.

## CHAPTER 3

### ULTRASONIC PROPERTIES

#### 3.1. Ultrasonic Attenuation Coefficient

The attenuation coefficient was determined from the acoustic amplitude image by measuring the average acoustic amplitude as a function of thickness [23]. The data were taken using a modified version (Appendix A) of the program "atten" [26].

Once invoked, the program displays a real time image from the SLAM on the video monitor with overlays. The overlays include an outlined box that represents the subarea to be used for calculating the average acoustic amplitude,  $V$ , and a line drawn from top to bottom on the left side of the screen that represents the minimum amount of blacked-out reference area required to calculate  $V$  correctly. Before taking data, the SLAM was adjusted to minimize the effects of ultrasonic field nonuniformity. This was done by placing an opaque rubber strip over the reference area, locating the subimage box over a uniform area of the image, and adjusting the receiver gain and video gain. The user generates a histogram, through computer interaction, which allows a wide dynamic range of the frame grabber to be used. The histogram provides the user with a graphic representation of pixel value distribution as well as the absolute minimum, absolute maximum, average, and most prevalent pixel values. From the histogram, the user can interactively adjust the gain to maximize the dynamic range without saturating the input.

After the appropriate gain adjustments were made, several values of  $V$  were recorded for each specimen thickness in the reference medium,  $V_r$ , and the

specimen region,  $V_s$ . The average acoustic amplitude value,  $V$ , is described mathematically by Eq. ( 1).

$$V = 10 \log \left\{ \frac{1}{n} \sum_{i=1}^n \left[ \frac{1}{3072} \sum_{i=y}^{y+31} \sum_{j=x}^{x+95} (v_{ij} - r_i) \right] \right\} \quad (1)$$

where

- $V$  = the average of the image area averages expressed in decibels,
- $n$  = the number of times the image is averaged,
- $i$  = image row index,
- $j$  = image column index,
- $y$  = starting row number of the imaging area,
- $x$  = starting column number of the imaging area,
- $v_{ij}$  = digitized value of the pixel at the  $i$ th row and  $j$ th column, and
- $r_i$  = average reference level of the first ten pixel values in the  $i$ th row.

$V$  values were taken in the reference and specimen areas (six in each area) and were recorded in a file, along with the time that the data were generated. The time that the data were generated was needed to analyze the results of the experiment. Then, the data were used to calculate insertion loss (IL), as defined in Eq. (2).

$$IL = \bar{V}_r - V_s \quad (2)$$

where

$$\begin{aligned}\bar{V}_r &= \text{the average of the } V \text{ values recorded for the reference medium, and} \\ V_s &= \text{the value for the specimen.}\end{aligned}$$

Six IL values were found for each thickness, and then the IL values were plotted versus thickness using a program called "regress" (Appendix C). The slope of the linear least squares fit line for the plot yielded the attenuation coefficient in decibels per unit thickness, i.e., dB/mm.

The accuracy of the IL values and attenuation coefficient was assessed elsewhere [25, 26]. The accuracy of the insertion loss was evaluated by taking  $V$  values for a thin layer of saline with varying amounts of inserted electrical attenuation. Twenty values were recorded for each of four levels of attenuation at a given video gain setting. The mean and standard deviations were computed for each level. The accuracy of  $V$ , calculated as two standard deviations of the mean, was  $\pm 0.35$  dB. IL is calculated by subtracting one  $V$  value from another, so the accuracy in IL is calculated using Eq. (3).

$$\Delta IL = \sqrt{[(\Delta V_s)^2 + (\Delta V_r)^2]} \quad (3)$$

where  $\Delta$  refers to the accuracy associated with a quantity. With an accuracy of  $\pm 0.35$  dB for both  $\Delta V_r$  and  $\Delta V_s$ , the accuracy for IL is  $\pm 0.5$  dB.

The accuracy and precision of the attenuation coefficient measurement were also evaluated. Using a 10% bovine serum albumin (BSA) solution, the average of several  $V$  values was taken for a very thin layer of BSA,  $< 10 \mu\text{m}$ , and used for  $V_r$ . Then, six to nine  $V$  values were calculated for each of four thicknesses of

BSA solution. The  $V$  values were input into the regression program to yield attenuation coefficient values. When compared to the known value, taking into account the overall uncertainty for the literature value, the accuracy uncertainty was  $\pm 5\%$  and the precision was  $\pm 16\%$ .

### 3.2. Ultrasonic Speed

The speed of sound in the specimen can be determined from the interferogram. The specimen is placed on the stage with a reference medium such that the interference image is separated into three separate horizontal regions. The top region contains the reference medium, followed by the specimen region, and the bottom reverts back to the reference medium. The interference lines shift upon entering the specimen region at the top and shift back upon reentering the lower reference region. The shifted interference lines create fringes, and based on the fringe shift, Eq. (4) [31] can be used to find the speed of sound,  $C_x$ , in the specimen

$$C_x = \left( \frac{C_0}{\sin \theta_0} \right) \sin \left\{ \tan^{-1} \left[ \left( \frac{1}{\tan \theta_0} - \frac{N\lambda_0}{T \sin \theta_0} \right)^{-1} \right] \right\} \quad (4)$$

where

- $C_x$  = speed of sound in the specimen,
- $C_0$  = speed of sound in the reference medium,
- $\theta_0$  = angle of sound from the normal in the reference medium,
- $N$  = normalized fringe shift,

- $\lambda_0$  = wavelength of sound in the reference medium, and  
 $T$  = thickness of specimen.

$C_0$  is a constant, typically 1520 m/s for saline at 30°C. The value for  $\theta_0$  can be calculated using Snell's Law

$$\theta_0 = \sin^{-1} \left[ \frac{C_0}{C_s} \sin \theta_s \right] \quad (5)$$

where

- $C_s$  = speed of sound in the fused-silica stage (5968 m/s), and  
 $\theta_s$  = angle at which the sound waves travel through the stage (45°).

If  $C_0$  is taken to be the typical value for saline, then  $\theta_0$  turns out to be 10.4°. Thus, all of the values are known for Eq. (4) except that of  $N$ .

The value of  $N$  can be found by measuring the shifts directly from the image monitor [31] or by using computerized methods [25, 26, 32-34]. The manual technique is not used, however, because it is time-consuming, depends on the user's experience, and allows only one speed value to be calculated for each fringe.

One computer automated technique utilized was the spatial domain technique (SDT) with automated data acquisition [32-34]. This technique allowed detection of a fringe shift of 0.03 compared to 0.1 with the manual method. The SDT found 30 to 39 speed values from each horizontal line of the interferogram, which meant that mean, mode, and standard deviation could be calculated to obtain some measure of specimen heterogeneity. The method worked well on homogeneous

materials such as bovine liver, but often failed for heterogeneous materials such as canine skin. Another method, which handles heterogeneous samples, is the spatial frequency domain technique (SFDT) [24, 25].

The SFDT quantifies the normalized fringe shift,  $N$ , in the frequency domain rather than in the spatial domain. The Fourier transform of a horizontal raster line is its spatial frequency spectrum. If the phase components of the Fourier transforms of two raster signals, one shifted relative to the other, are evaluated at the frequency at which the spectrum is a maximum, the phase difference,  $\Delta\phi(-\zeta_0)$ , is given by

$$\Delta\phi(-\zeta_0) = \frac{2\pi y_0}{\lambda_y} = 2\pi N \quad (6)$$

where

- $\zeta_0$  = spatial frequency at which the spectrum is a maximum,
- $y_0$  = amount of horizontal shift in the signal,
- $\lambda_y$  = fringe line spacing, and
- $N$  = normalized fringe shift.

Equation (6) thus shows that the change in phase between two raster lines is related to  $N$ . To calculate  $\Delta\phi(-\zeta_0)$ , the phase for each raster line in the specimen is subtracted from the average phase determined for the reference raster lines. A phase unwrapping algorithm is used to solve the  $2\pi$  ambiguity problem inherent in the determination of phase using the Fourier transform.

The SFDT has several advantages over SDT. It works for heterogeneous specimens where the SDT often fails; it is more tolerant to noise and is twice as

fast. Because the SFDT analyzes the interferogram vertically rather than horizontally, it produces about 10 times more speed values per line analyzed and allows a contour, or speed profile, of speed versus data point to be generated. The speed profile, thus, allows a visualization of how the speed of sound changes from the reference to the specimen, and also within the specimen.

A few modifications to the SFDT were introduced by Steiger [25] to make the technique more adaptable to variations in the interferograms. The SFDT proposed by Tervola and O'Brien [24] assumed that the fringe line spacing remained constant over the entire image and determined the phase at a fixed frequency. An investigation of the digitized data from the Z-80 microprocessor controlled system for several interference images showed that this was not true. It was shown that a saline image had maximum frequency components between 700 and 800 kHz. A comparison of interferograms showed that the fringe spacing for heterogeneous specimens was much more variable than for homogeneous specimens. A study of the heterogeneous specimen revealed that the maximum frequency components fell between 600 and 800 kHz.

One modification by Steiger [25] compensated for the difference by computing the Fourier transform for a range of frequencies and then using the phase of the frequency with the maximum amplitude to compute  $N$ . Also, a discrete Fourier transform (DFT) replaced the fast Fourier transform (FFT) to reduce leakage in the Fourier transform by allowing the data window length to be an integer number of wavelengths.

Nicozisin [26] modified the SFDT slightly when the new data acquisition analysis system was introduced. Some features of the speed algorithm changed, which proved to be enhancements. Because the DT2851 samples a full video frame, 480 speed values are available in one pass as compared to 255 with the old



system [25]. Another feature provides the user with many more options that affect how and where the analysis is performed.

The main modification to the algorithm was in regard to the DFT calculation. A 512-point DFT was selected over the 768-point DFT used previously. Also, a 26-point window length is used to reduce the probability that the program would choose the incorrect frequency,  $\zeta_0$ . The reasoning behind the modifications can be found elsewhere [26].

Currently, N is computed using the modified SFDT to calculate the speed of sound from Eq. (4). The "speedn" program (Appendix D) takes care of these calculations and outputs the speed, its standard deviation, and the standard deviation of N.

### 3.3. Heterogeneity

The heterogeneity index (HI) is a measure of the acoustic heterogeneity of the specimen [25]. The SFDT allows many speed values to be generated which represent different areas of the specimen. The speed values provide an indication of the heterogeneity of the specimen. Equation (7) is a mathematical description for calculating the HI.

$$HI = (\sigma_S - \sigma_R) \times 100 \quad (7)$$

where  $\sigma_S$  is the standard deviation of the specimen speed values and  $\sigma_R$  is the standard deviation of the reference speed values.

As seen in the equation, finding the standard deviation of the specimen speed values alone is not enough. It was found that the SLAM system may contribute to the variation of the speed values through sources such as electrical noise, a

nonuniform sound field, and errors caused by finite word length in the data acquisition and computation system components. Thus, these contributions must be corrected when calculating the HI. The correction is made by subtracting the standard deviation of the reference medium, which is a homogeneous substance, from the standard deviation of the sample.

The value of  $N$  is another indicator that can be used to assess the heterogeneity of the material.  $N$  is used to calculate the speed, so it can also be used as a measure of heterogeneity. By taking the SD of the mean fringe value, the normalized heterogeneity index ( $\langle HI \rangle$ ) can be determined.

## CHAPTER 4

### METHODOLOGY

#### 4.1. Experimental Protocol and Procedures

Data were taken for 24 hr while the cartilage samples underwent enzymatic degradation. The enzymes used were type II-A porcine pancreatic elastase and type III bovine pancreatic trypsin. Both enzymes were ordered from Sigma<sup>®</sup> Chemical Company and stored in a Forma Bio-Freezer at -70°C.

The trypsin digestion was used to give a collagen-only structure, as described by Chun et al. [21]. The collagen-only structure was shown to be devoid of noncollagenous matrix, but retained the shape, integrity, and tensile strength of its collagen network.

The degradation of the tissue using elastase was performed to cleave the cross-links of the tissue (Fig. 1). According to Schmidt et al. [20], this enzyme specifically degrades the non-helical terminal peptides of collagen molecules, which are the principal cross-linking sites. Denaturant extraction showed that this treatment functionally destroys more than 70% of the intermolecular bonds.

The buffer, 0.05 M Tris-HCl, pH 8.5, was prepared at 23°C and stored in the refrigerator at 3°C. Both enzymes were prepared at a concentration of  $\frac{100\mu\text{g}}{\text{mL}}$  using the buffer and kept in an ice bath throughout the experiment.

The articular cartilage was excised immediately postmortem from the stifle (knee) joint of 1- to 2-year-old adult steers. The cartilage was removed from the patellar groove with a cylindrical 6-mm-diam punch and washed in saline plus protease inhibitors (2-mM phenylmethylsulfonyl fluoride, 10-mM N-ethyl maleimide, 2-mM ethylene diamine tetraacetic acid, and 5-mM benzamidinium-HCl).

The 6-mm disk of cartilage was cut in half, trimmed, and mounted so that sections perpendicular to the articular surface could be cut (Fig. 4). The plugs were mounted on cork with a standard mounting media for frozen sections, Ames Tissue-Tek<sup>®</sup> O.C.T. [(optimal cutting temperature), a polyvinyl alcohol, benzalkonium chloride, and polyethylene glycol gel], frozen in liquid nitrogen, and placed in Ziploc<sup>®</sup> bags for shipping. The plugs were shipped on dry ice from the University of Washington to the Bioacoustics Research Laboratory at the University of Illinois where they were stored in a Forma Bio-Freezer at -70°C pending analysis.

It was necessary that the samples be frozen both for transportation and sectioning, but the study was carried out so that only 1 cycle of freeze-thaw was needed. Freezing the specimens did not seem to be a problem since it has been shown that the mechanical properties of skin and wound tissue were not affected by rapid freezing in liquid nitrogen [35]. Similarly, the ultrasonic impedance among several tissues studied before and after freezing was found to be the same [36].

To prepare the samples for slicing, the corks were mounted with standard mounting media on an object disk of a Lipshaw Cryostat Microtome. The upper portion, about 100  $\mu\text{m}$ , of the sample was shaved off at -10 to -16°C to provide a uniformly flat surface for each of the cartilage slices. Then, the slices were made with the thinnest being first and the thickest last. After being sliced, the samples were rinsed with buffer to wash away any remaining embedding medium. Finally, the tissue slices were notched, as shown in Fig. 5, so that the orientation of the sample was constant throughout the experiment. Five thicknesses (30, 40, 50, 60, and 70  $\mu\text{m}$ ) were cut for each sample, though only the 40 - 70  $\mu\text{m}$  samples were used for analysis, as discussed in the results section (Ch. 5).

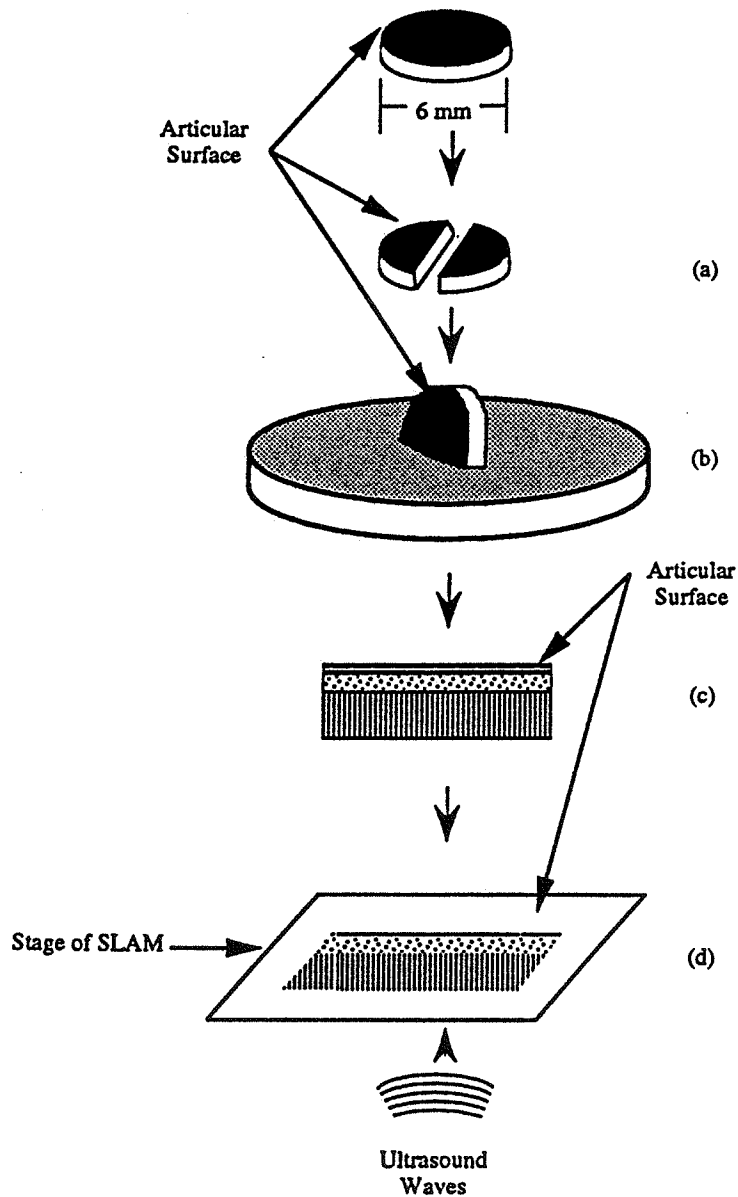


Figure 4. Diagram of cartilage orientation for sectioning. (a) The 6-mm-diam disks of cartilage were bisected with the articular surface facing upward. (b) Each half of the disk was mounted for sectioning perpendicular to the articular surface. (c) The cut section includes the tangential, transitional, and radial fiber orientations. (d) Also shown is the schematic relationship of the ultrasonic wave direction to the fibril orientation.

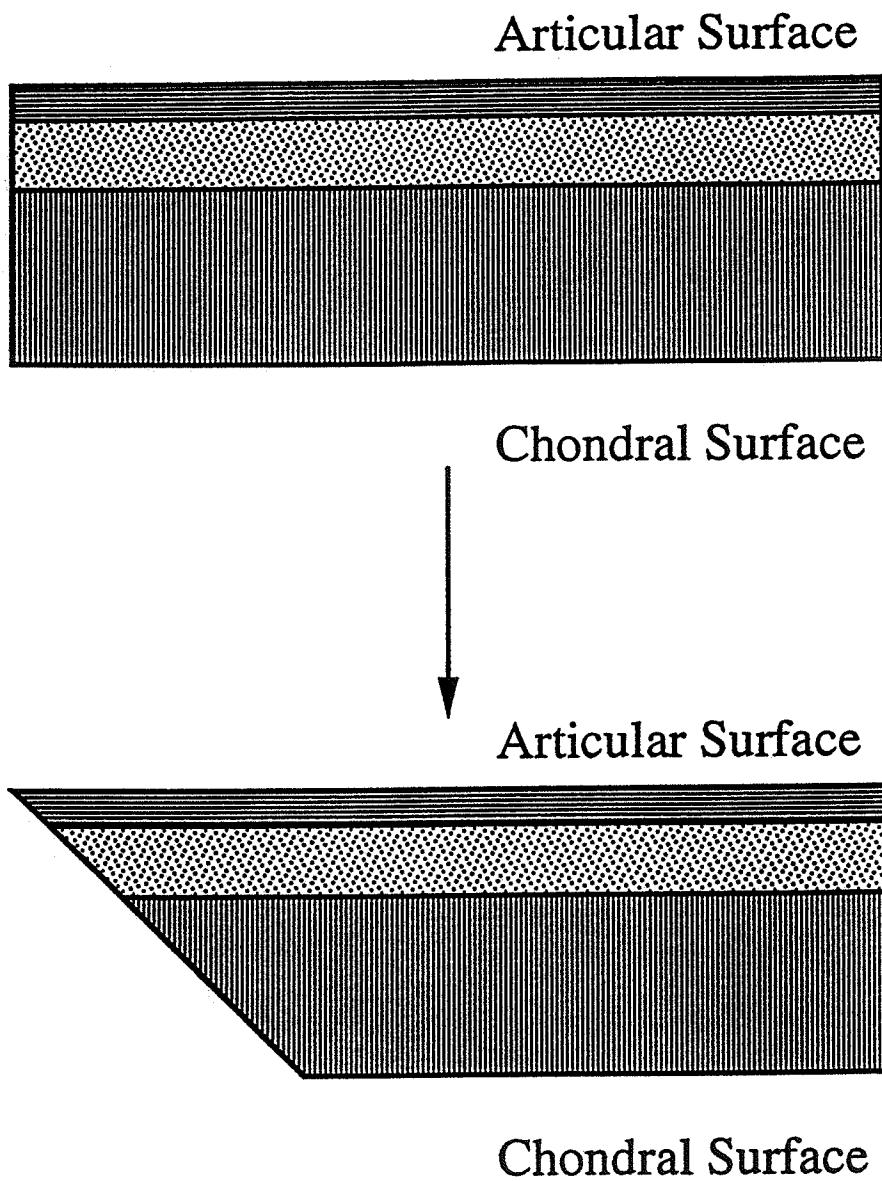


Figure 5. Diagram of the articular cartilage after notching, which helps retain orientation throughout the experiment.

Acoustic amplitude data (Appendix A) and an interferogram (Appendix B) were then taken for each of the thicknesses over a 24-hour period. The interference images were taken by selecting the option of the program to average 8 frames from the SLAM. In the first data acquired, the samples were suspended in the buffer. Then, the buffer was blotted away and an enzyme was applied. After each 24-hour experiment, the interferograms were analyzed (Appendix D) to extract speed, HI, and  $\langle HI \rangle$  values. Also, the acoustic amplitude data were used to calculate and plot IL values versus thickness (Appendix C) for quantification of the attenuation coefficient.

#### 4.2. Statistical Procedures

The significance of the acoustic properties versus treatment, fiber orientation, and time was evaluated using InStat<sup>®</sup> version 2.00, a statistical analysis program. The two-tailed Student's t test and multivariate analysis of variance (ANOVA) were used to determine p values. A probability of  $p < 0.05$  was considered significant for all tests.

CHAPTER 5  
RESULTS AND DISCUSSION

5.1. Results

5.1.1. Attenuation

Insertion loss data were recorded for each of the samples and plotted versus specimen thickness using a linear regression program to find the slope, or attenuation coefficient, of the sample. Table 1 outlines the attenuation coefficients obtained for each of the treatments.

Table 1. The SLAM attenuation coefficient.

Treatment	Sample	Atten. Coeff. (dB/mm)
Normal	A3A	-40
	B2A1	48
	B2A2	-2
	A3B	8
	B1B	41
	B2B	0
Trypsin	A3A	17
	B2A1	57
	B2A2	21
Elastase	A3B	-10
	B1B	41
	B2B	5

The data were ambiguous and no further analyses were performed.



### 5.1.2. Speed

The speed data varied between samples and thicknesses, but the same basic trend was observed for each sample. Mainly, the speed in the radial zone was largest while the speeds of the tangential and transitional zones were smaller. The tangential zone, usually, had a slightly higher speed than the transitional zone, though sometimes this was not the case.

The speed values were plotted to provide a time-course view of changes over 24 hr. Plots were generated for each of the sample thicknesses and fiber orientations for zero to 24 hr. (See Figs. 6 and 7.) As seen in the plots, the data for thicknesses 40-70  $\mu\text{m}$  were consistent, but the data for 30  $\mu\text{m}$  were outliers. For the trypsin study, the speed data for the 30  $\mu\text{m}$  samples were, on average, 2.8% different than the average speed data, whereas the data for the 40-70  $\mu\text{m}$  samples showed only a 0.8-1.7% difference. Similarly, the data for the 30  $\mu\text{m}$  samples from the elastase experiments were 2.3% different, on average, than the average speed data; the data for the 40-70  $\mu\text{m}$  samples were only 0.7-0.8% different than the average. Thus, it was decided to exclude the 30  $\mu\text{m}$  data.

To find if there was a general trend in the speed as time progressed, the data were averaged at time zero (buffer) and at 24 hours for each enzyme (trypsin and elastase) and then plotted (See Figs. 8 and 9.). The speed in all three regions decreased after 24 hr. In addition, plots were generated which showed the time-course of speed values. (See Figs. 10 and 11.)

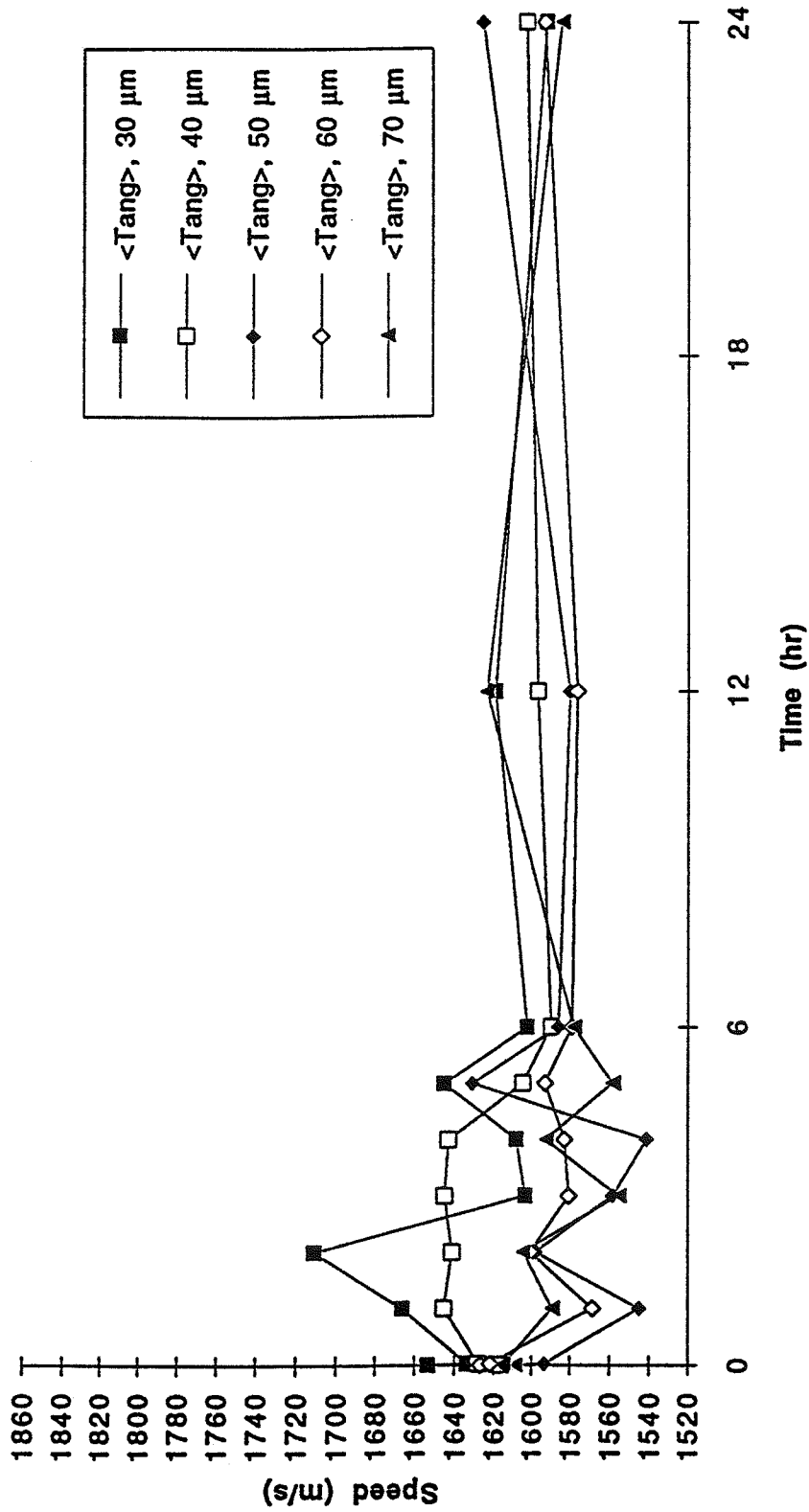


Figure 6(a). Time-course of speed in the tangential zone for thicknesses treated with trypsin.

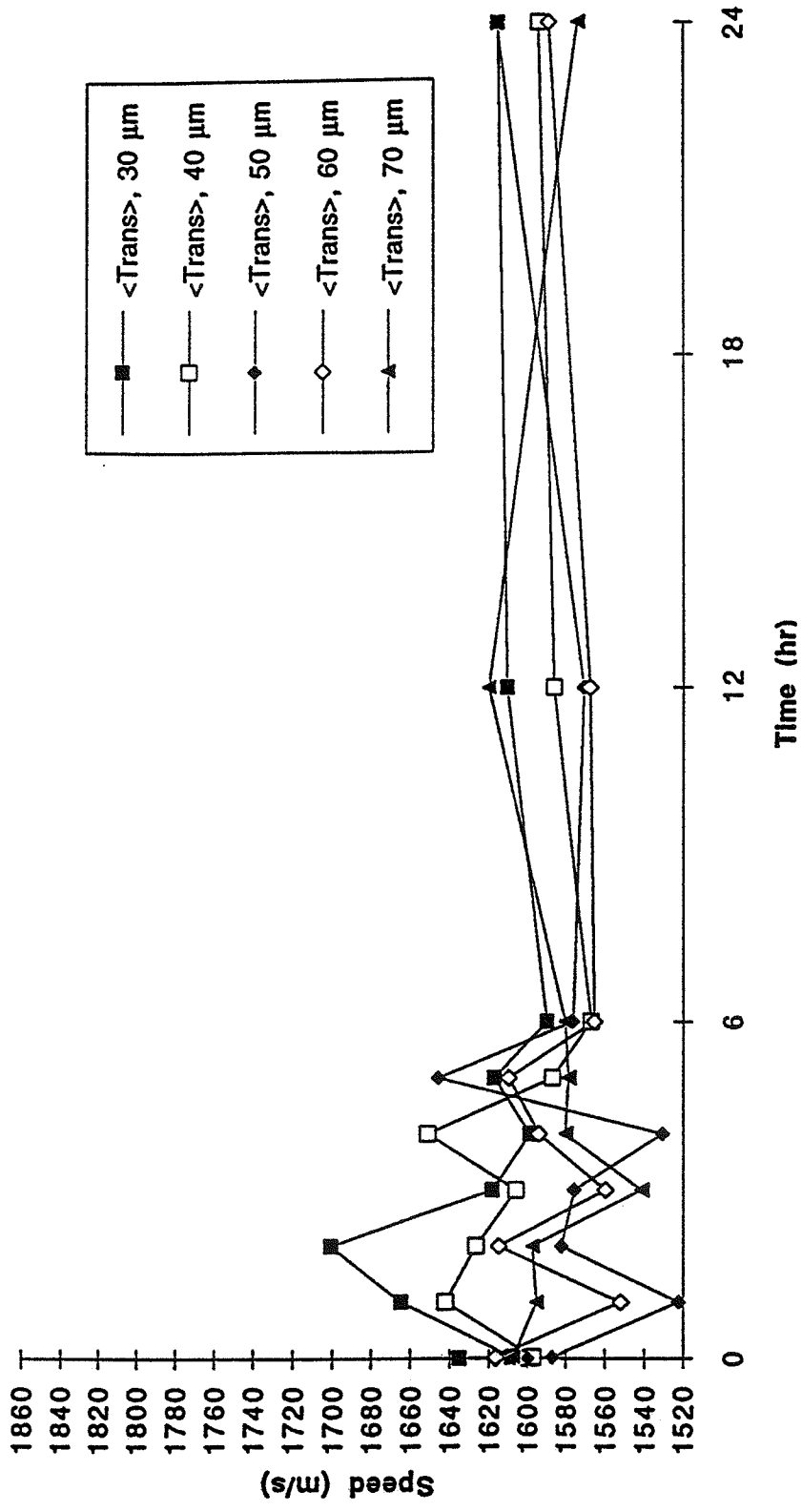


Figure 6(b). Time-course of speed in the transitional zone for thicknesses treated with trypsin.

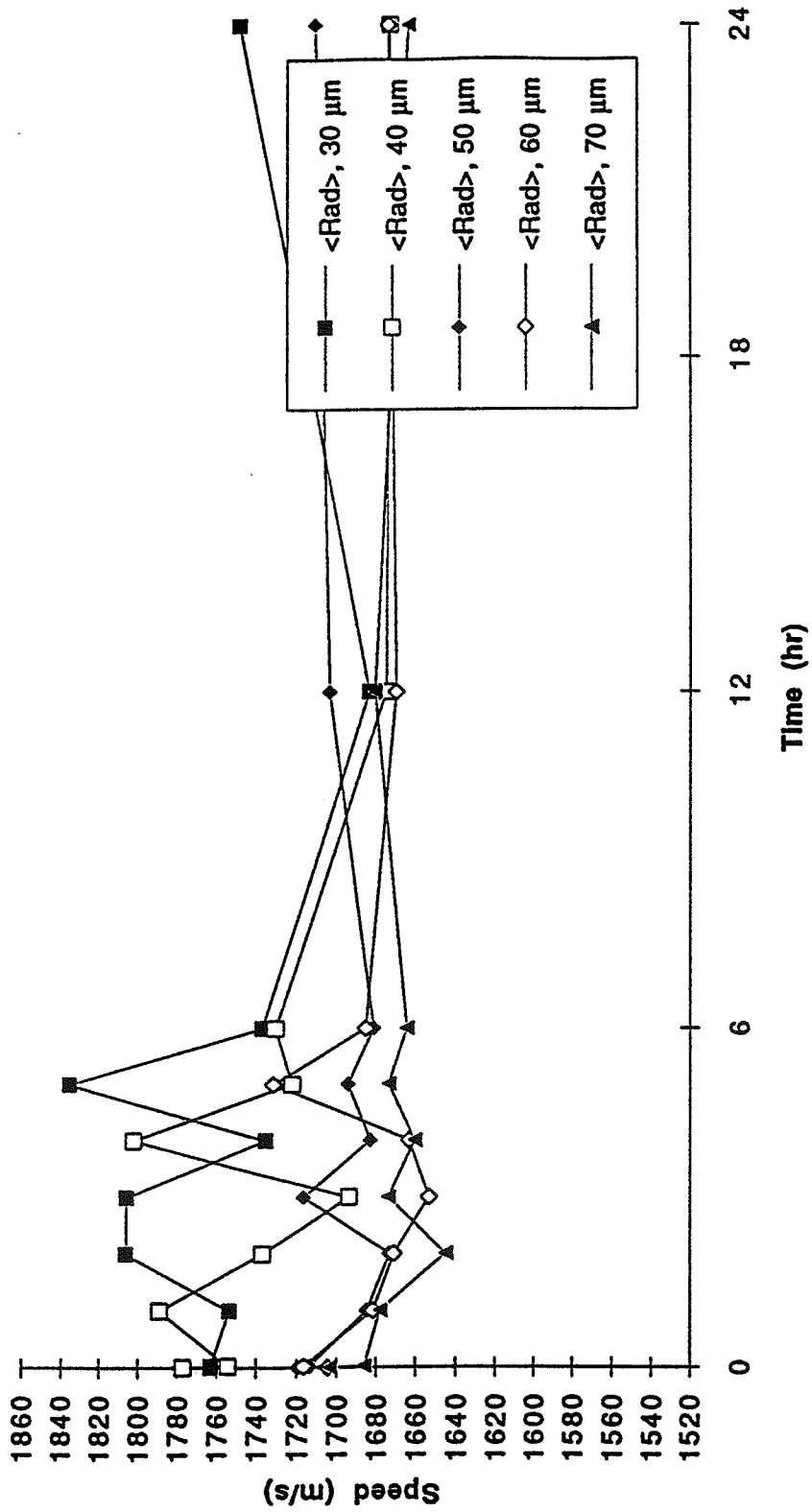


Figure 6(c). Time-course of speed in the radial zone for thicknesses treated with trypsin.

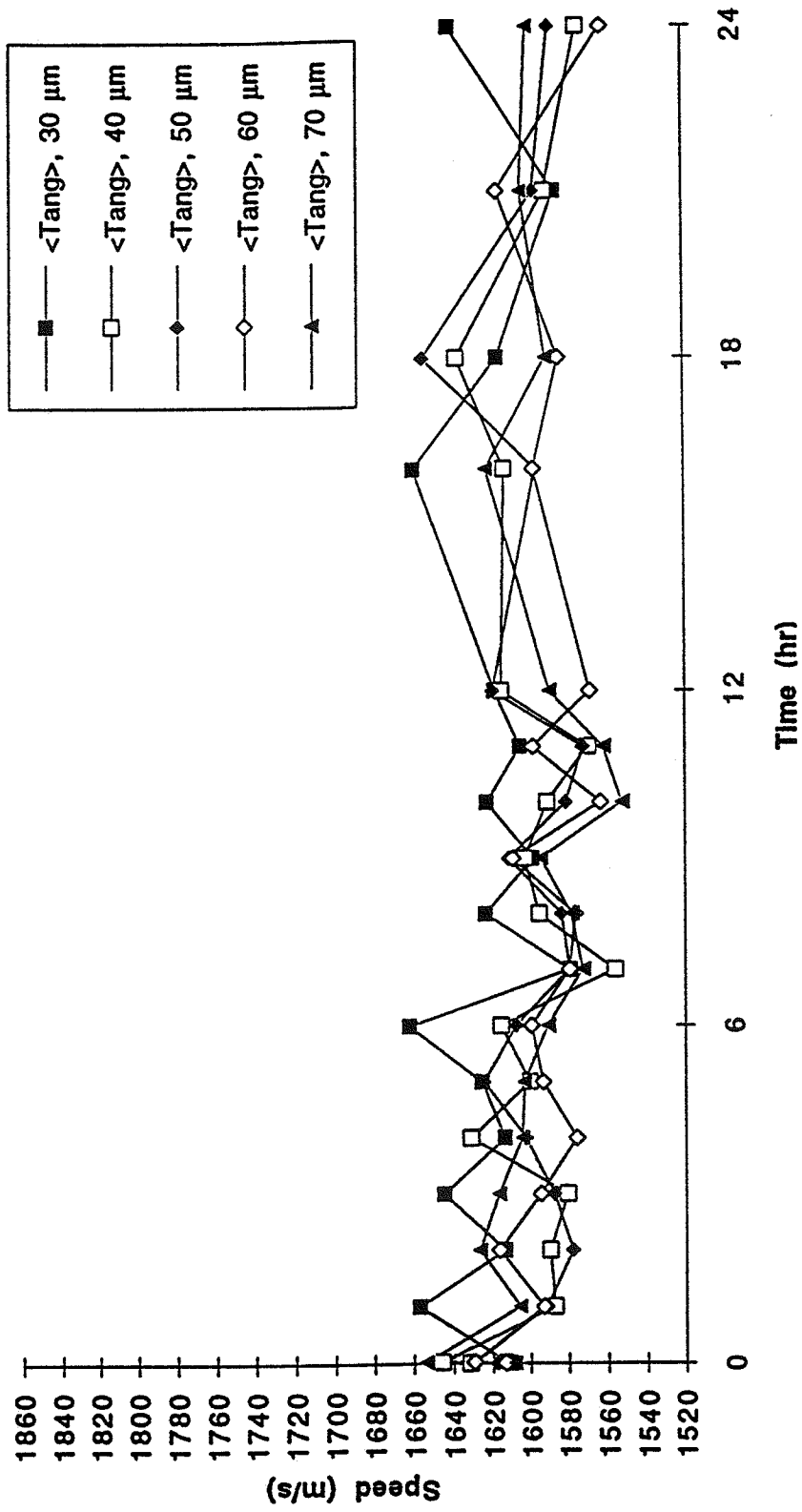


Figure 7(a). Time-course of speed in the tangential zone for thicknesses treated with elastase.

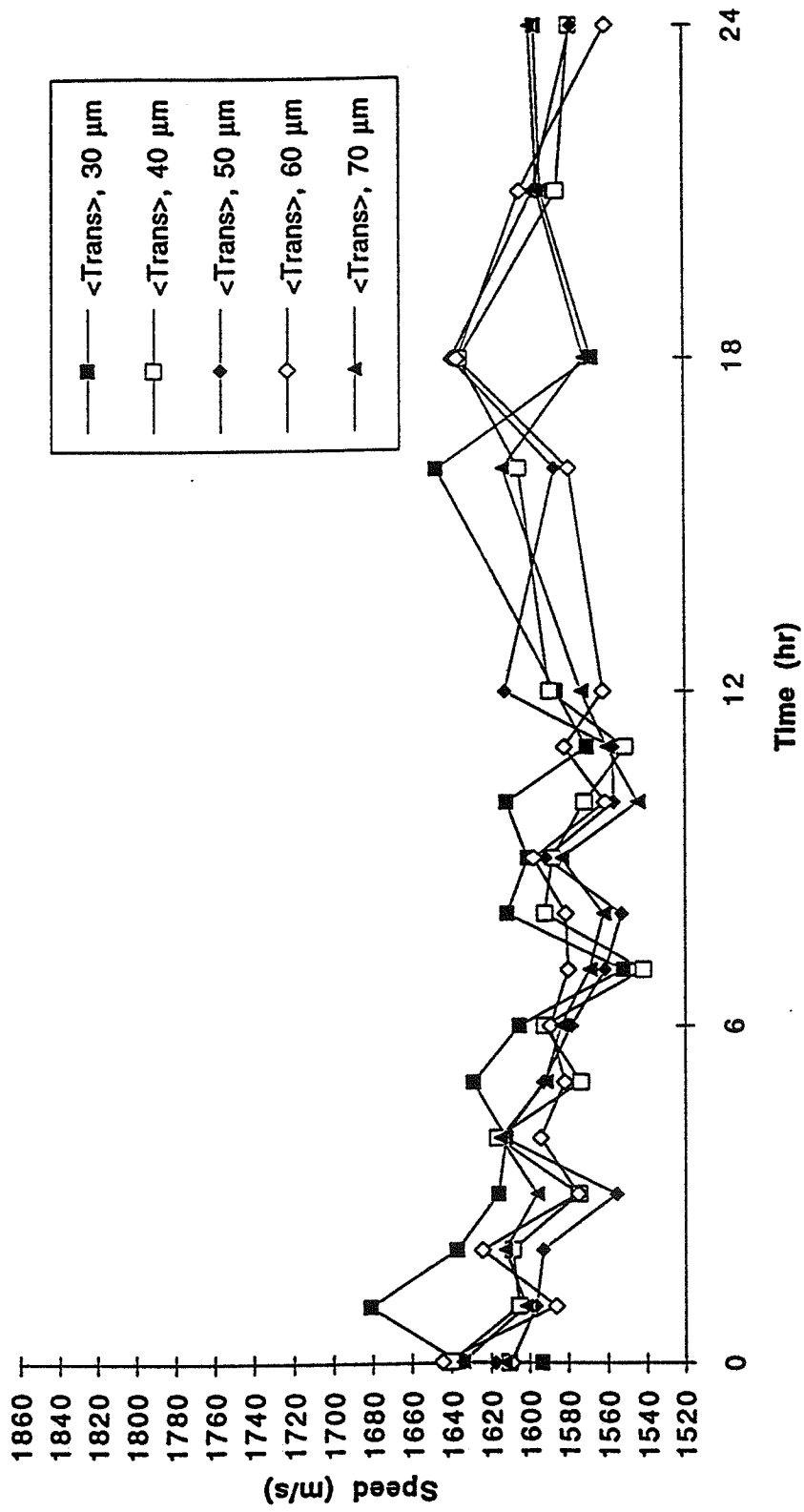


Figure 7(b). Time-course of speed in the transitional zone for thicknesses treated with elastase.

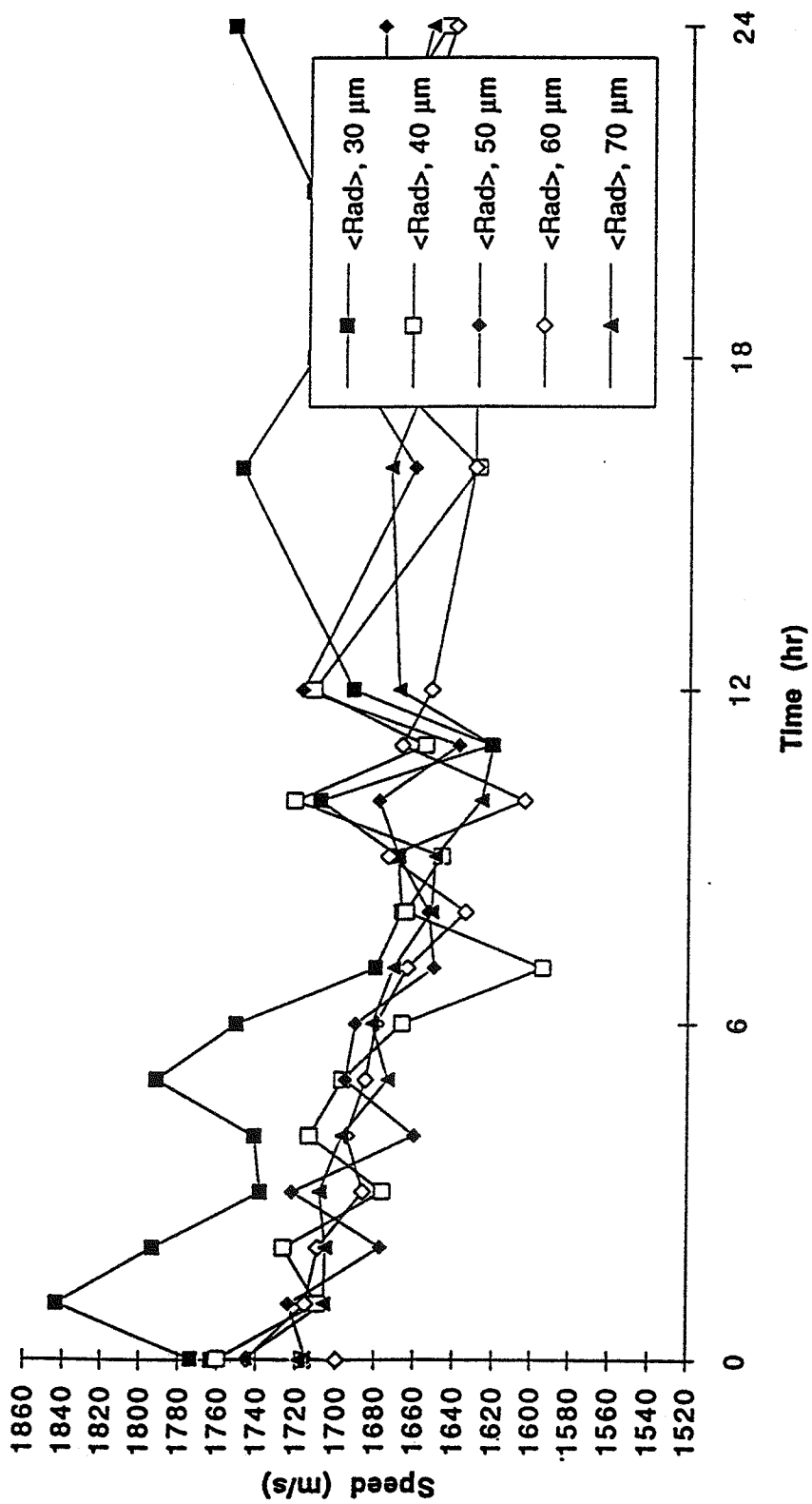


Figure 7(c). Time-course of speed in the radial zone for thicknesses treated with elastase.

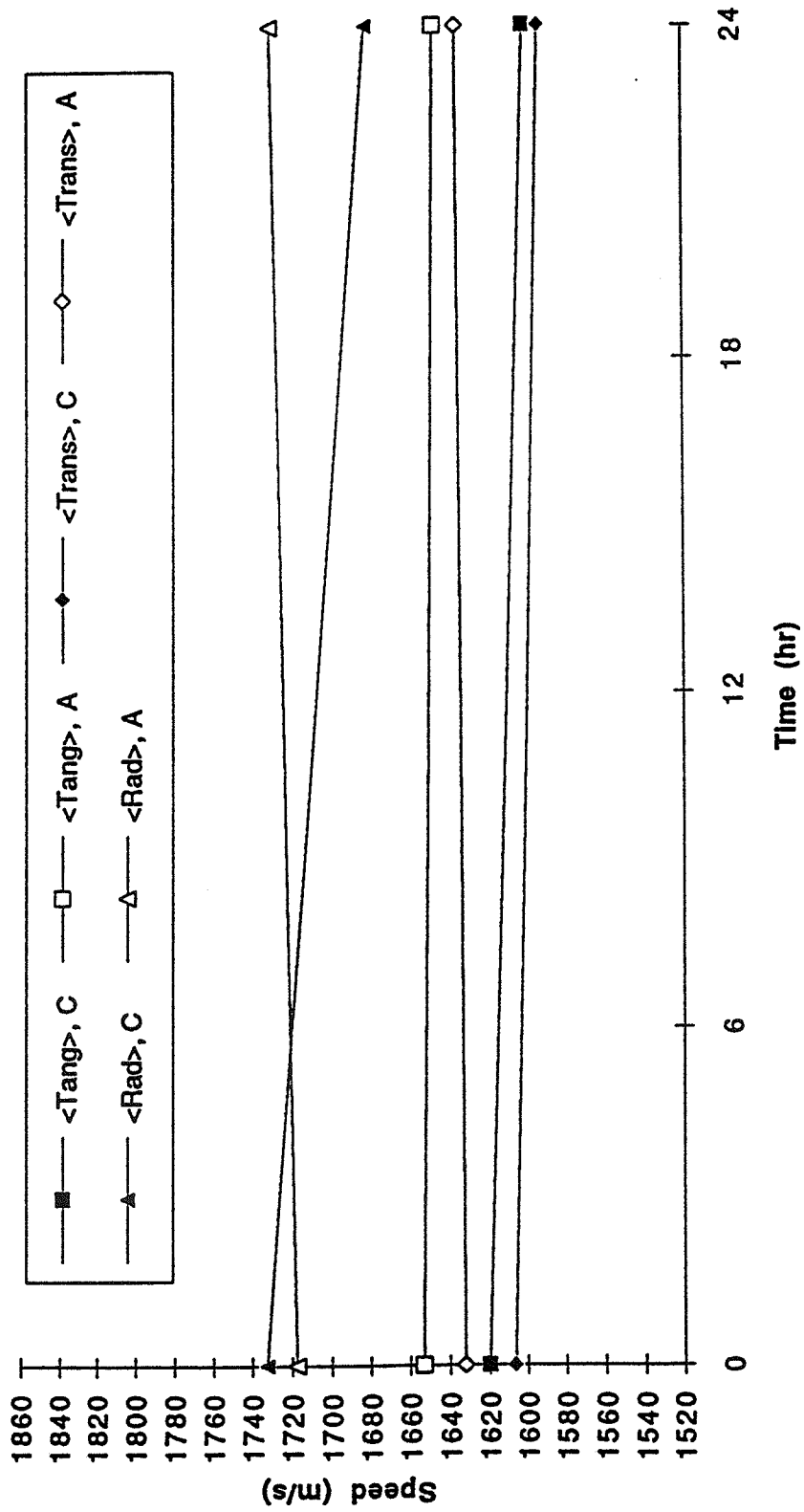


Figure 8. Comparison of the trypsin study speed data of this experiment (C) with that of the Agemura study (A).



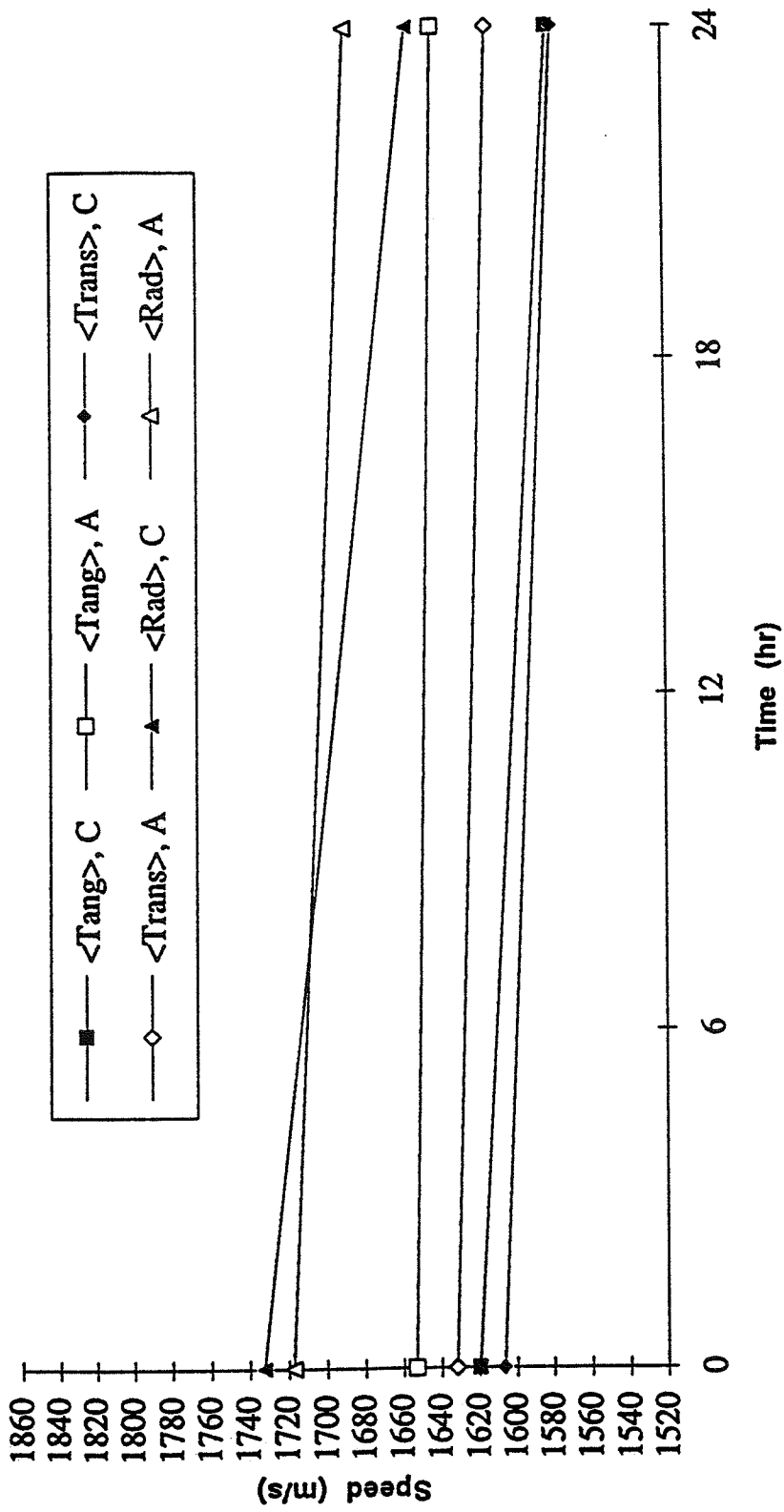


Figure 9. Comparison of the elastase study speed data of this experiment (C) with that of the Agemura study (A).

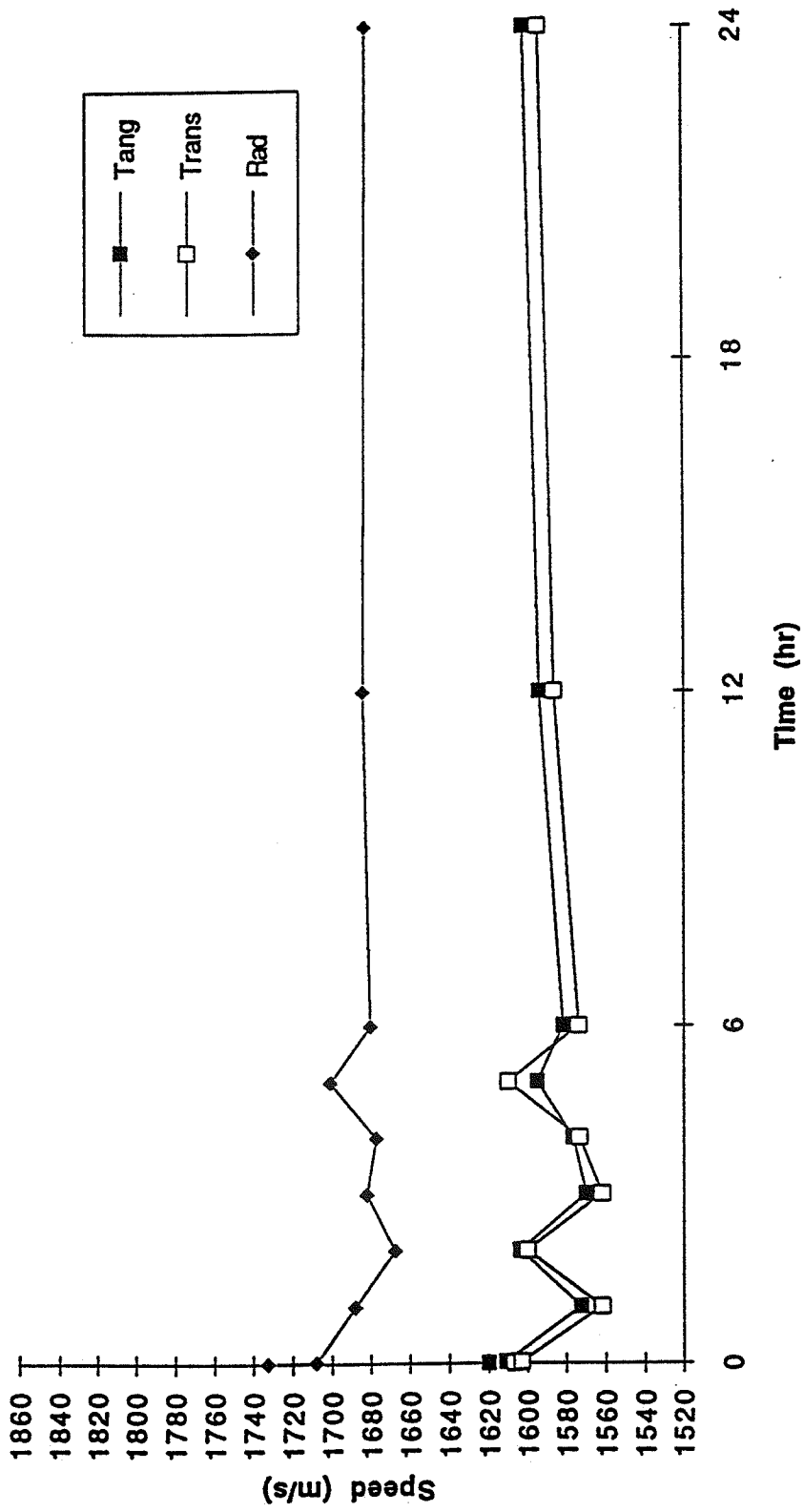


Figure 10. Time-course of speed data for the trypsin study.

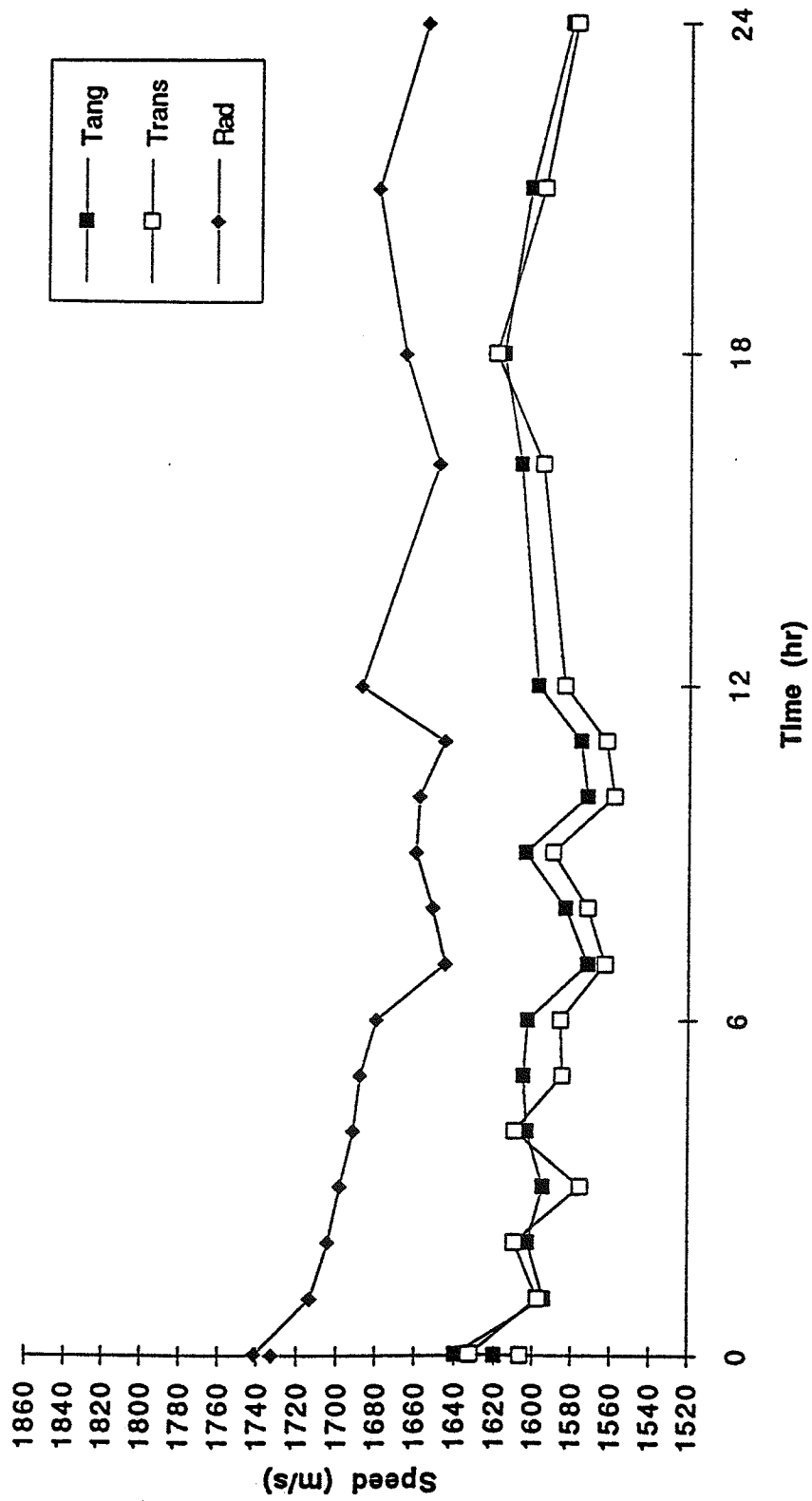


Figure 11. Time-course of the speed data for the elastase study.

Analysis of variance (ANOVA) evaluated the influence of treatment, fiber orientation, and time on the propagation speed. The first tests were performed to assess treatment, but the treatment did not have a significant effect on the speed for any of the times or fiber orientations (Table 2).

Table 2. The ANOVA results for the effect of treatment on speed.

Time (hours)	Fiber Orientation	P value
6	Tangential	0.090
	Transitional	0.23
	Radial	0.55
12	Tangential	0.83
	Transitional	0.77
	Radial	0.57
24	Tangential	0.13
	Transitional	0.37
	Radial	0.18

The fiber orientation influenced the speed for all three treatments at all times (Table 3).

Table 3. The ANOVA results for the effect of orientation on speed.

Treatment	Time (hours)	P value	Post-Test Results*
Normal		< 0.0001	Tang. vs Rad.; Trans. vs Rad.
Trypsin	6	< 0.0001	Tang. vs Rad.; Trans. vs Rad.
	12	< 0.0001	Tang. vs Rad.; Trans. vs Rad.
	24	< 0.0001	Tang. vs Rad.; Trans. vs Rad.
Elastase	6	< 0.0001	Tang. vs Rad.; Trans. vs Rad.
	12	< 0.0001	Tang. vs Rad.; Trans. vs Rad.
	24	< 0.0001	Tang. vs Rad.; Trans. vs Rad.

The post-test results indicate that no significant differences occurred between the tangential and transitional regions for any of the treatments.

---

\* The post-test results indicate the cases for which  $p < 0.05$ .

The last factor taken into account was time, which showed a significant effect on speed (Table 4).

Table 4. The ANOVA results for the effect of time on speed.

Treatment	Fiber Orientation	P value	Post-Test Results
Trypsin	Tangential	0.005	0 vs 6 hr
	Transitional	0.0034	0 vs 6 hr
	Radial	0.013	0 vs 6 hr; 0 vs 12 hr; 0 vs 24 hr
Elastase	Tangential	0.0023	0 vs 24 hr
	Transitional	0.011	0 vs 6 hr; 0 vs 24 hr
	Radial	0.0019	0 vs 6 hr; 0 vs 24 hr

For the trypsin data, the only significant difference between the tangential and transitional regions was for normal vs. 6 hr. The data in the radial region, however, showed significant differences between normal and each of 6, 12, and 24 hr.

The data from the elastase experiments indicated a significant difference between normal and 24 hr for all three regions. In addition, the transitional and radial zones showed significant differences between normal and 6 hr.

### 5.1.3. Normalized Heterogeneity Index

Also evaluated was the heterogeneity of the articular cartilage during the enzymatic breakdown. Similar to the evaluation of the speed data, the <HI> values were averaged at time zero (buffer) and at 24 hr for each enzyme (trypsin and elastase) and then plotted. (See Figs. 12 and 13.) Each of the enzymes affected the heterogeneity of the sample differently. The <HI> for the trypsin treated samples decreased in the radial and transitional zones across the time-course of the experiment, while it increased in the tangential region (Fig. 12).

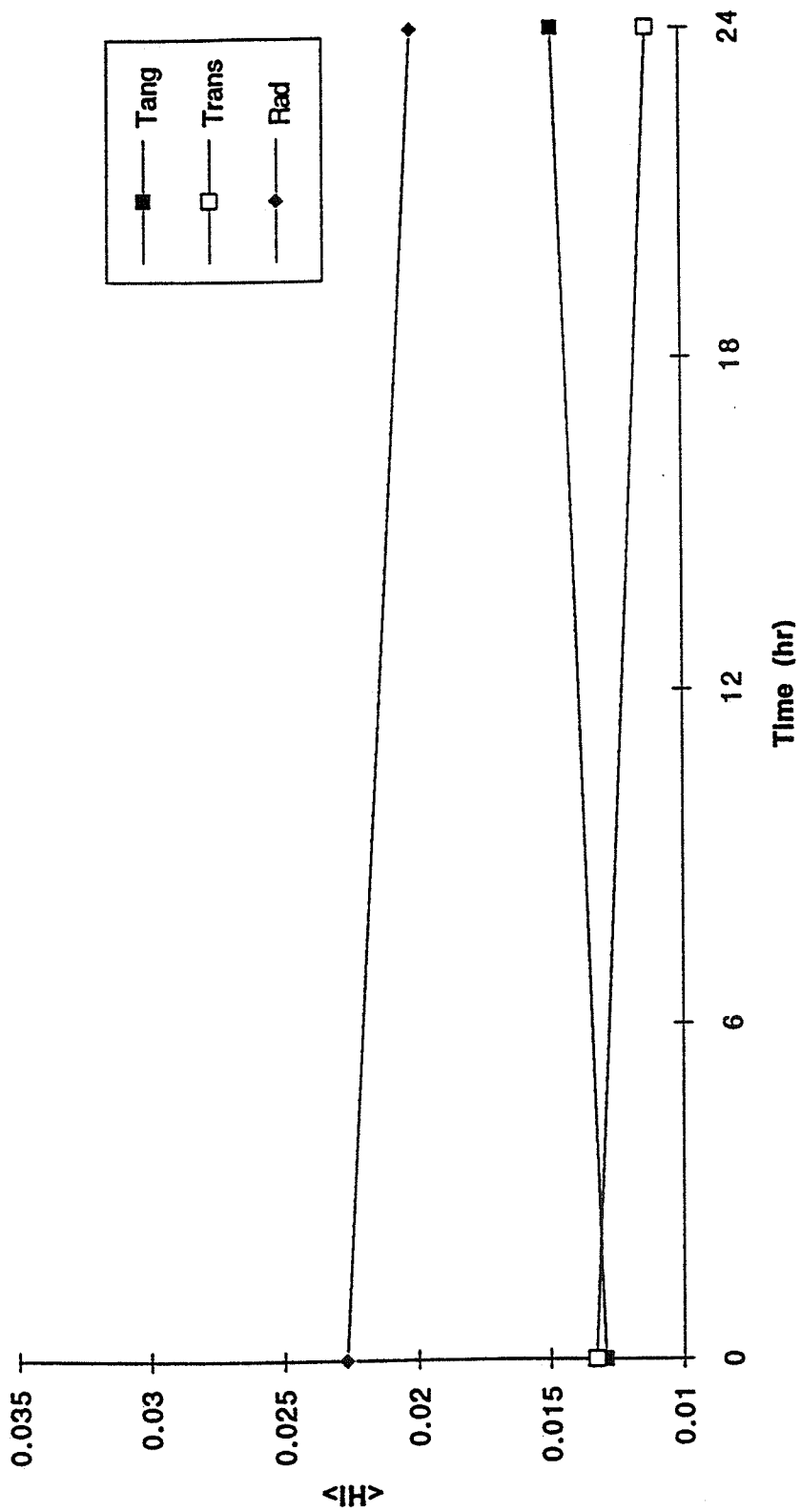


Figure 12.  $\langle HI \rangle$  for the trypsin study at 0 and 24 hr.

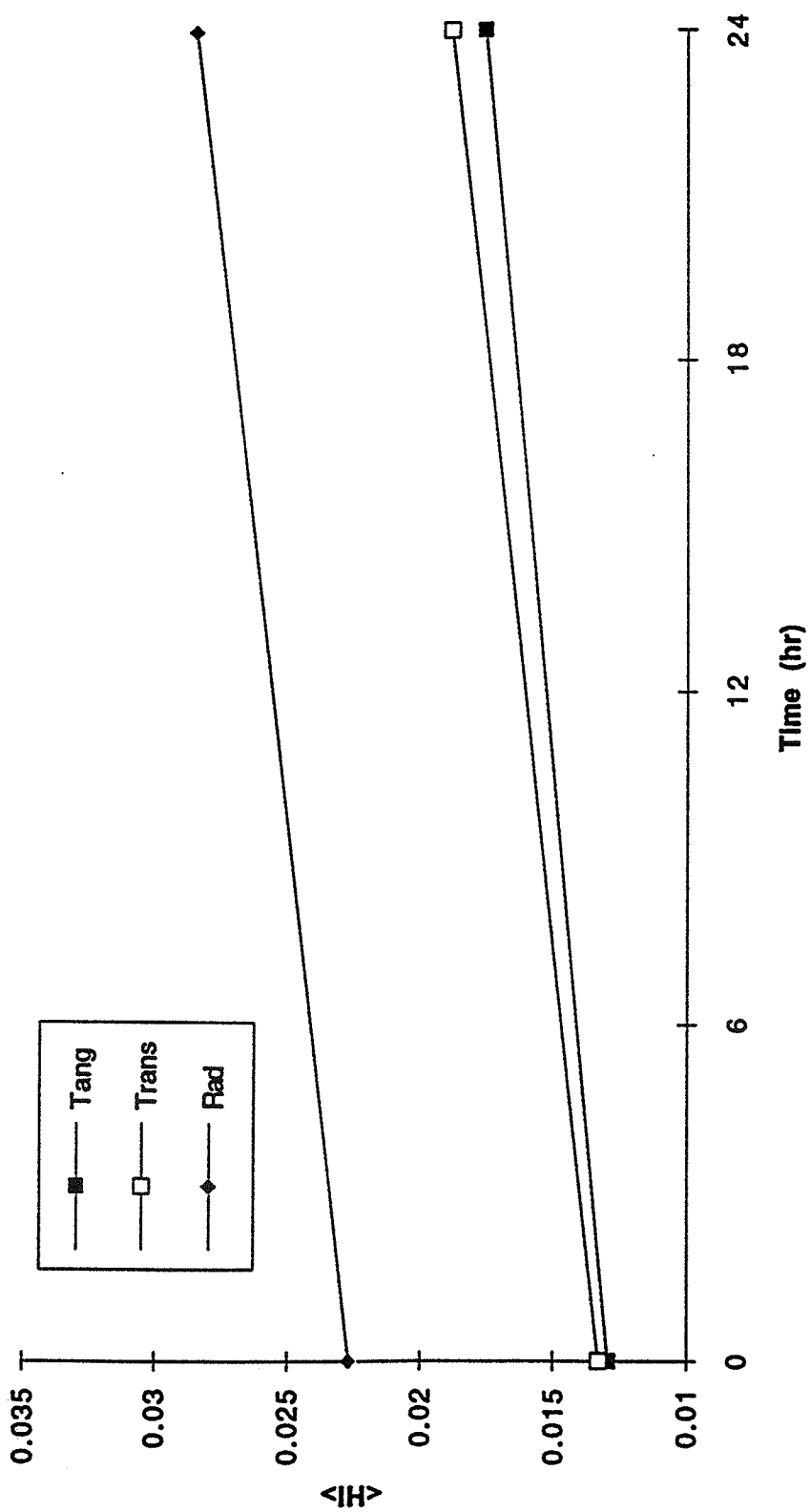


Figure 13.  $\langle HI \rangle$  for the elastase study at 0 and 24 hr.

The <HI> for the samples in the elastase study, however, increased over time in all three regions (Fig. 13).

Plots were also generated to provide a view of the time-course evolution of the data. (See Figs. 14 and 15.) The <HI> for the trypsin study peaks rapidly and then approaches its original value as the reaction reaches completion (Fig. 14). The curves for the study with the elastase rise with the application of the enzyme and then oscillate (Fig. 15).

ANOVA testing was also performed on the <HI> data to determine the effects of treatment, fiber orientation, and time. The effect of treatment on <HI> (Table 5) indicates that treatment does not influence the <HI> at 6 hr.

Table 5. The ANOVA results for the effect of treatment on <HI>.

Time (hours)	Fiber Orientation	P value	Post-Test Results
6	Tangential	0.52	-
	Transitional	0.16	-
	Radial	0.69	-
12	Tangential	0.018	Norm. vs Elas.; Elas. vs Tryp.
	Transitional	0.11	-
	Radial	0.34	-
24	Tangential	0.060	Norm. vs Elas
	Transitional	0.0019	Norm. vs Elas.; Elas. vs Tryp.
	Radial	0.026	Norm. vs Elas.; Elas. vs Tryp.

The <HI> varied at 12 hr in the tangential region and at 24 hr in all three regions due to treatment.



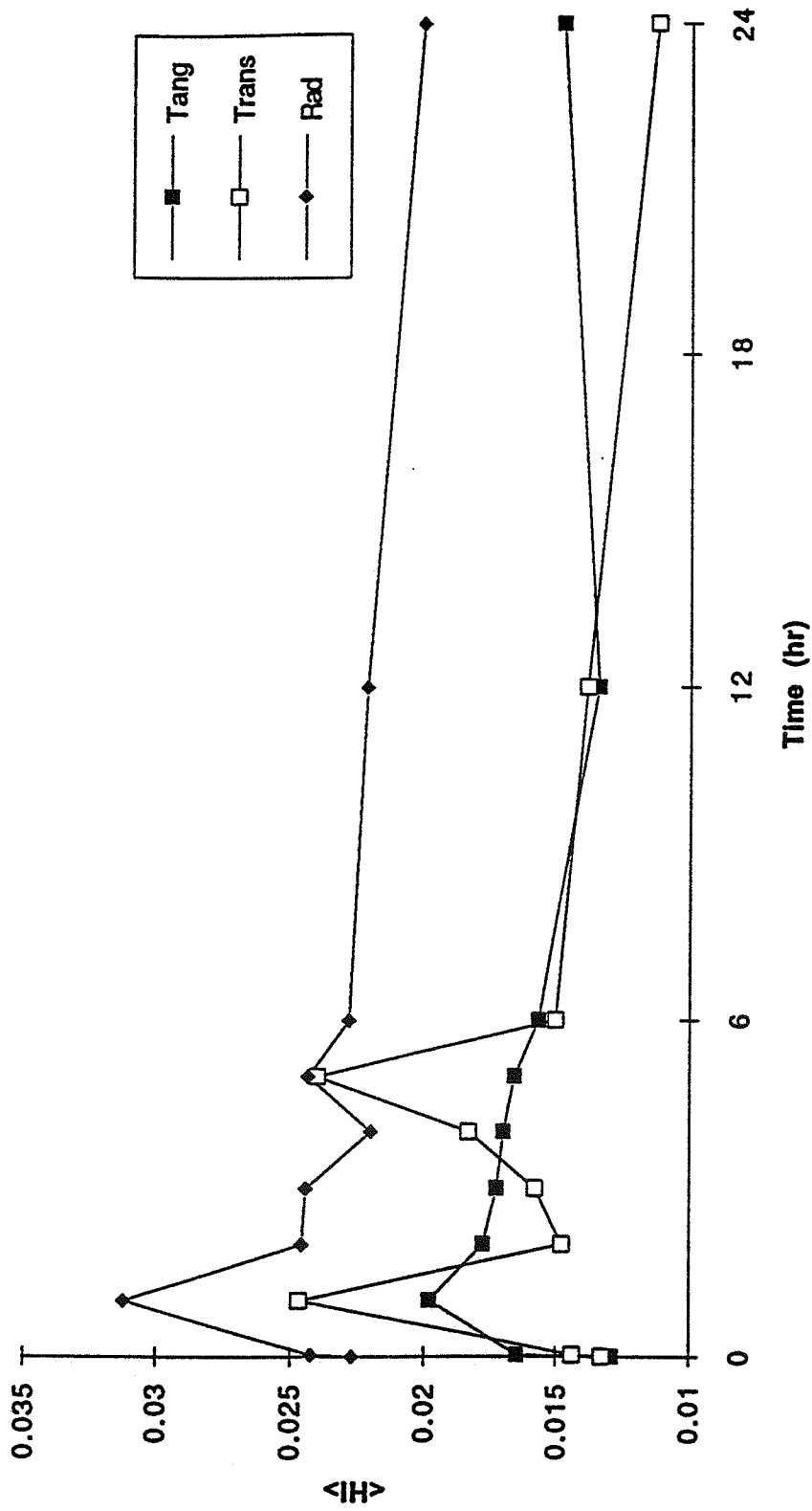


Figure 14. Time-course of the  $\langle HI \rangle$  data for the trypsin study.

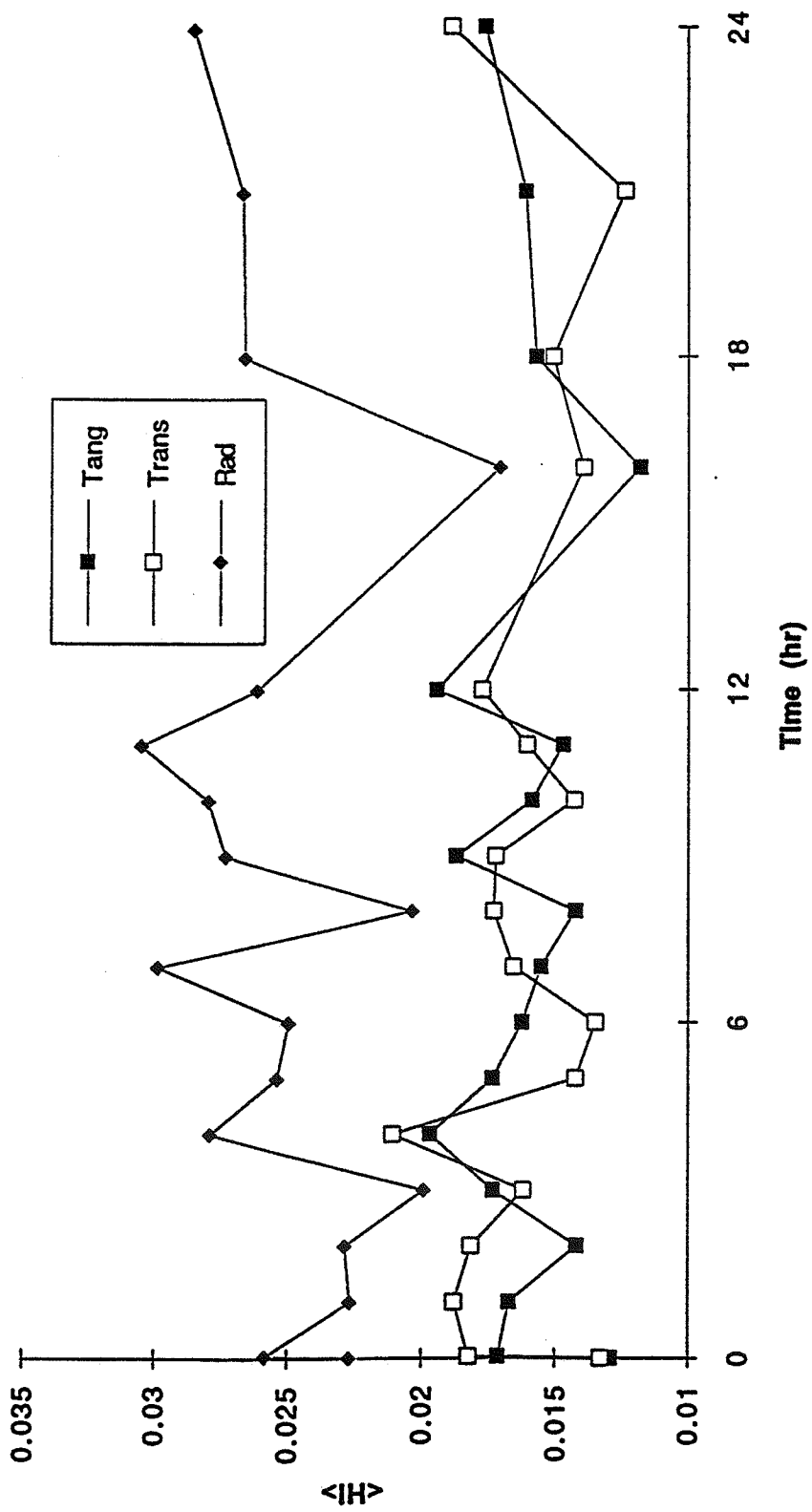


Figure 15. Time-course of the  $\langle HI \rangle$  data for the elastase study.

Next, statistical analysis was performed to assess orientation on <HI>. The results (Table 6) indicate that orientation was found to influence the <HI> for all cases except for elastase at 12 hr.

Table 6. The ANOVA results for the effect of fiber orientation on <HI>.

Treatment	Time (hours)	P value	Post-Test Results
Normal Trypsin		< 0.0001	Tang. vs Rad.; Trans. vs Rad.
	6	< 0.0001	Tang. vs Rad.; Trans. vs Rad.
	12	< 0.0001	Tang. vs Rad.; Trans. vs Rad.
	24	0.0025	Tang. vs Rad.; Trans. vs Rad.
Elastase	6	< 0.0001	Tang. vs Rad.; Trans. vs Rad.
	12	0.10	-
	24	0.007	Tang. vs Rad.; Trans. vs Rad.

The post-test revealed that the fiber orientation was significant only for the cases of tangential vs. radial and transitional vs. radial.

The last factor investigated was time (Table 7).

Table 7. The ANOVA results for the effect of time on <HI>.

Treatment	Fiber Orientation	P value	Post-Test Results
Trypsin	Tangential	0.71	-
	Transitional	0.092	-
	Radial	0.57	-
Elastase	Tangential	0.011	0 vs 12 hr
	Transitional	0.0063	6 vs 24 hr; 0 vs 12 hr; 0 vs 24 hr
	Radial	0.11	-

Degradation time affected the <HI> only in the tangential and transitional fiber orientations of the tissue treated elastase. The effect of time, based upon post-test results, was significant for normal vs. 12 hr in both the tangential and transitional regions. It was also found to be significant for 6 vs. 24 hr and normal vs. 24 hr in the transitional zone.

## 5.2. Discussion

The attenuation coefficient results were ambiguous. When it was discovered that the attenuation data were not consistent with the values reported by Agemura and O'Brien [15, 22], one of the mounted samples was reanalyzed. The thicknesses for the run that yielded poor results had been taken between the top and the middle of the mounted sample while the thicknesses used for reanalysis were taken from near the cork. The reanalyzed sample had an attenuation coefficient of 178 dB/mm (Fig. 16) which was consistent with previously published results ( $88\pm 9$  and  $105\pm 11$  dB/mm  $\pm$  s.e.) [15, 22].

Based upon the effect of fiber orientation on  $\langle HI \rangle$  (Table 6), the heterogeneity of the sample would seem to be the factor that most affected the attenuation coefficient. The methodology for calculating the attenuation coefficient assumed that the sample was homogeneous. Since the fiber orientation had a significant effect on the heterogeneity, it would also have had a significant effect on the attenuation. The IL data were taken in different regions of the tissue which would lead one to believe that the IL values at different locations in a single thickness would be significantly different. The IL values did not agree with this though. For example, in the plot (Fig. 17) for sample A3A, normal treatment, the IL values for the 30  $\mu$ m sample varied over a range of approximately 4 dB, while for sample A3B, normal treatment (Fig. 18), the IL values covered a range of about 1.6 dB. Similarly, the IL values for the 60  $\mu$ m thickness of sample A3A (Fig. 17) were spread across a range of 1.2 dB, while for sample A3B (Fig. 18), the range was approximately 3.6 dB.

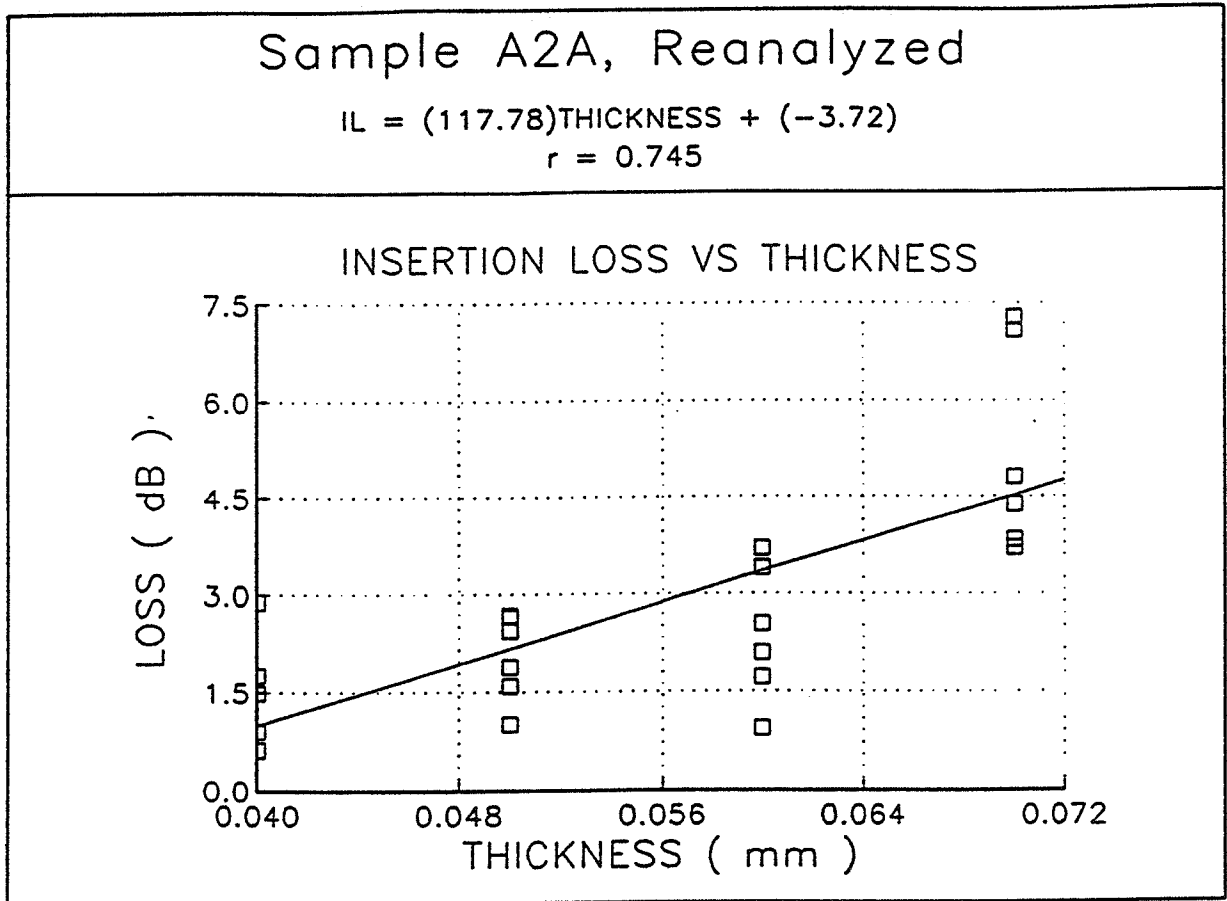


Figure 16. Plot of insertion loss (IL) vs thickness for the reanalyzed sample A2A suspended in buffer.

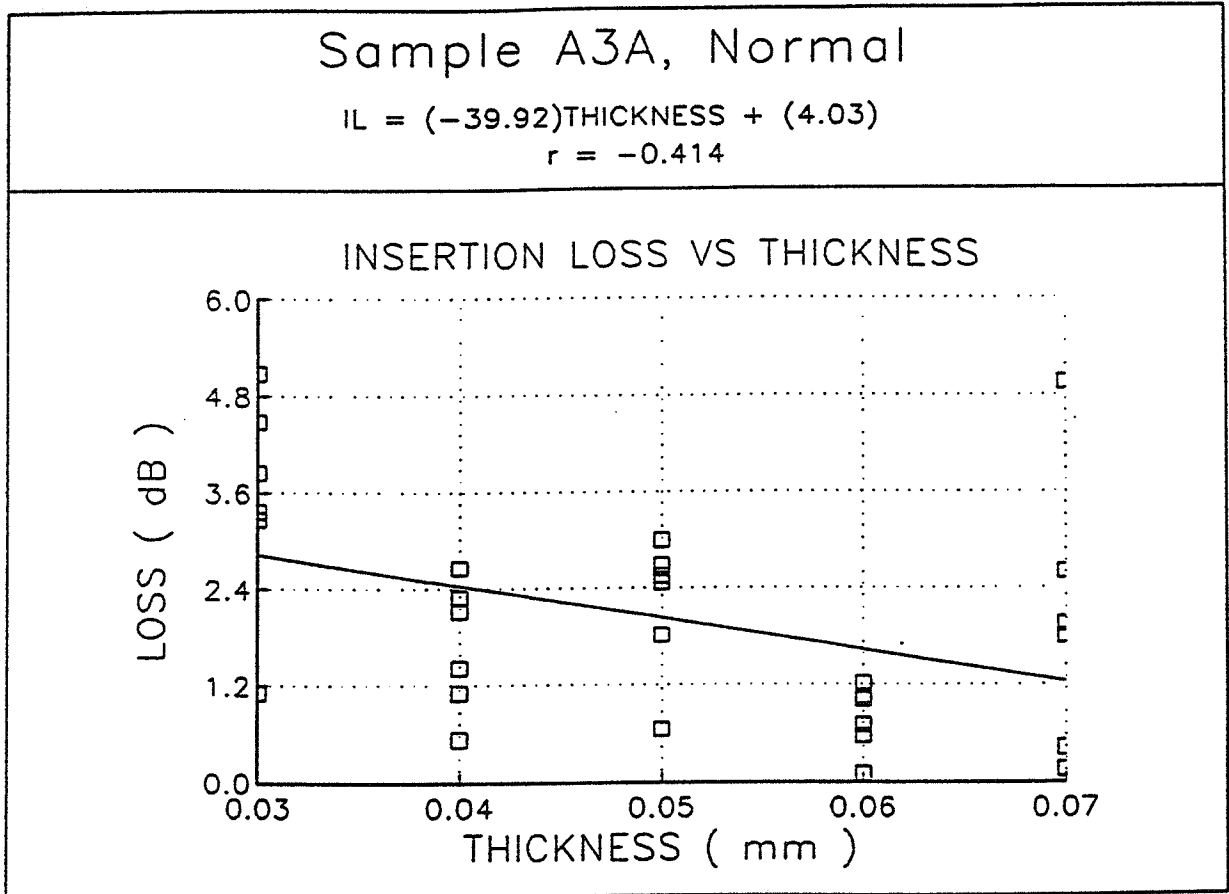


Figure 17. Plot of insertion loss (IL) vs thickness for sample A3A suspended in buffer.

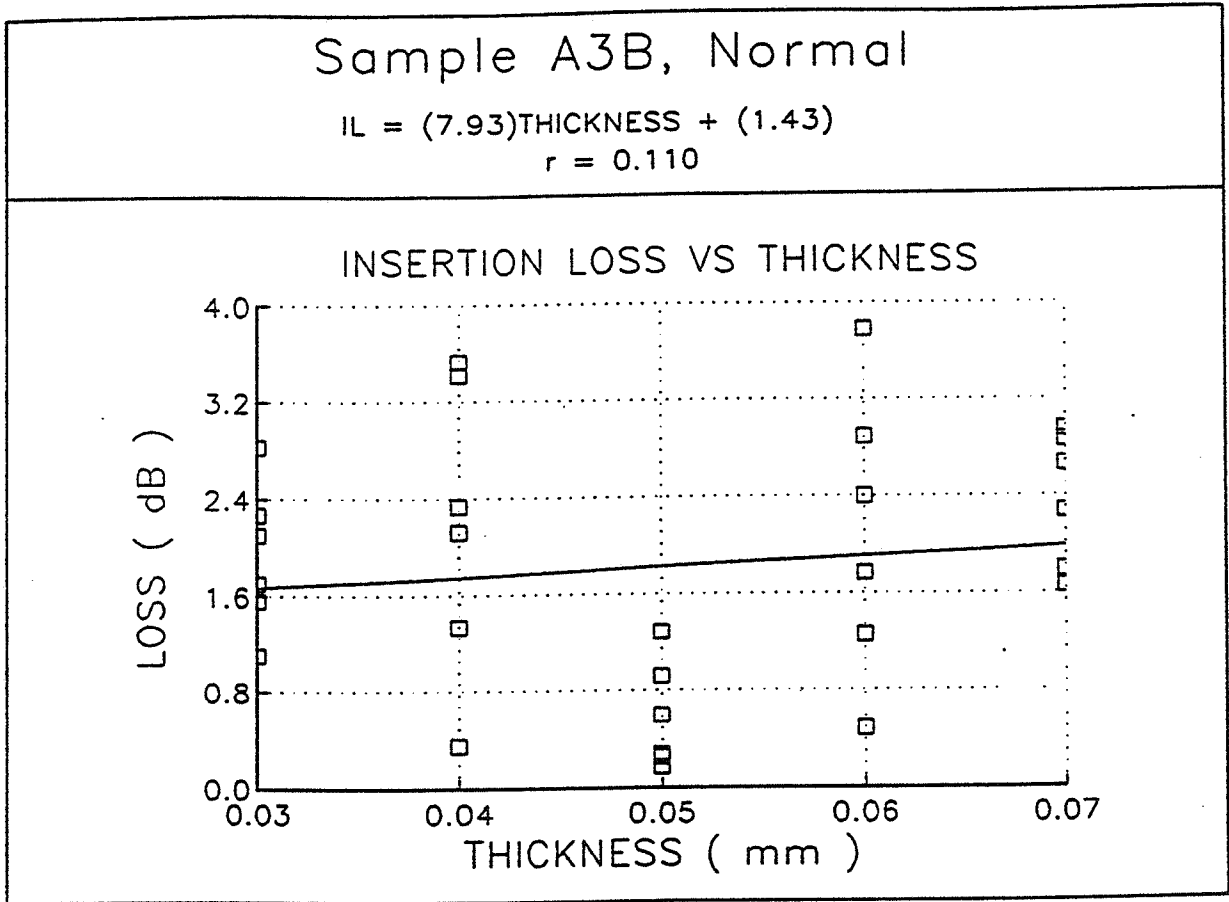


Figure 18. Plot of insertion loss (IL) vs thickness for the reanalyzed sample A3B suspended in buffer.

Dehydration could have also affected the attenuation coefficient. Since the attenuation coefficient calculated for the thicknesses taken near the top of the mounted specimen was ambiguous and the attenuation coefficient for the thicknesses near the bottom was in agreement with values published by Agemura, the mounted specimens may have been dehydrated near the surface. Though possible, the dehydration theory does not agree with previous results [4] which found that attenuation and speed are closely linked. If dehydration were the cause of the poor attenuation data, then it would seem that the speed data would have been ambiguous. However, the percent errors between the speed data from the Agemura study and this study were less than 4% (Table 8).

Table 8. The % error between the published speed values in [15, 22], averaged, and the speed values, averaged, found in the current study.

	Tangential	Transitional	Radial
Normal	2.0	1.6	.9
Trypsin	2.8	2.6	2.8
Elastase	3.7	2.2	2.0

The effect of treatment on speed, with respect to fiber orientation, was found not to be significant. This observation agrees with the study by Agemura and O'Brien [15].

Degradation time proved to have an effect on the speed. For the trypsin treatment, the only significant difference between speed values in the tangential and transitional regions was for zero vs 6 hr (Table 4). The speed decreased during the first 6 hr of the experiment (Fig. 10) and then increased slightly by 24 hr. The speed at 24 hr was found not to be significantly different from the time zero value (Table 4).



The speed values for the radial region changed significantly between zero and 6 hr (Table 4, Fig. 10). Then, the speed remained relatively constant for the rest of the experiment (Fig. 10). At 24 hr the speed had changed, shown in Figure 8, to decrease from time zero, by a significant amount (Table 4). Thus, the trypsin influenced the speed of sound in the material during the first 6 hr of the experiment as a steady-state value was approached.

The time-course of speed values for the elastase treatment followed a similar trend as the trypsin treatment (Fig. 9). The primary difference between the two was that the speed values in the tangential and transitional regions changed significantly after 24 hr for the elastase treatment, which was not the case for the trypsin treatment (Table 4). Thus, the speed was affected in all three regions after 24 hr when the elastase treatment was applied.

The speed was influenced most by the collagen fibril orientation (Table 3). The post-test indicated that the tangential vs. transitional regions were not significantly different ( $p > 0.05$ ) for all three treatments. This result was expected since it was often difficult to distinguish between the tangential and transitional regions when extracting the data from the speed profile. For example, in Fig. 19 the tangential and transitional regions are easily distinguishable, but in Fig. 20 the choice of the boundary becomes a judgmental call by the individual extracting data from the images.

During the first six hr of the trypsin treatment experiment (Fig. 14),  $\langle HI \rangle$  rose and then decreased to a relatively steady-state value for the rest of the degradation. Degradation time did not have a significant effect on the  $\langle HI \rangle$  (Table 7).

For the elastase treatment (Fig. 15), the  $\langle HI \rangle$  values rose with the application of the enzyme at approximately 2 min., and then oscillated for the rest of the experiment. The  $\langle HI \rangle$  oscillated about a mean value of about 0.025 in the radial

zone; in the tangential and transitional zones, the mean value was about 0.018. According to the post-test results (Table 7), time influenced the  $\langle HI \rangle$  for tangential and transitional fiber orientations. However, the trends, shown in Figs. 14 and 15, seem to be more important to understanding the effect of the elastase on the tissue. A study which monitors the  $\langle HI \rangle$  and the degree to which the crosslinks have been cleaved could provide some valuable insight as to why  $\langle HI \rangle$  oscillates throughout the experiment.

The  $\langle HI \rangle$  was also found to be affected by the fiber orientation for both treatments (Table 6). As was the case for the speed data, the fiber orientations were found to be significant for the radial vs. tangential and radial vs. transitional cases.

The elastase treatment influenced the  $\langle HI \rangle$  for the tangential orientation at 12 hr and the transitional and radial orientations at 24 hr. (See Table 6.) The result which indicated that the treatment had an influence on the  $\langle HI \rangle$  may be misleading, though, considering the path that the  $\langle HI \rangle$  follows across 24 hr for each of the treatments. (See Figs. 14 and 15.) Recall, for the trypsin study, the  $\langle HI \rangle$  increased and then asymptotically decayed (Fig. 14), while in the elastase study, it increased and then oscillated about a mean value (Fig. 15). The trends seem to be more important than absolute time comparisons.

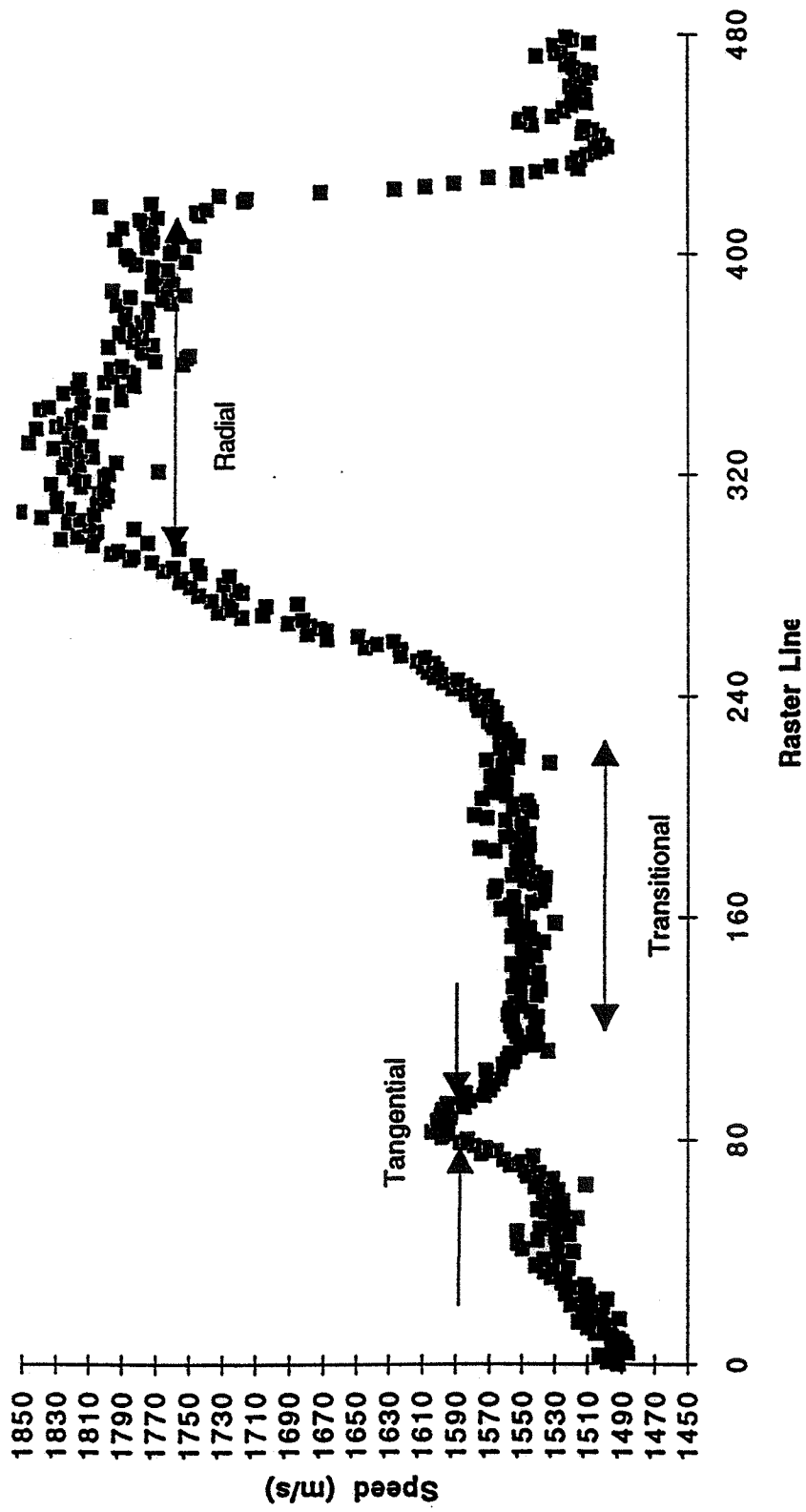


Figure 19. Speed profile for which the tangential and transitional regions are readily distinguishable.

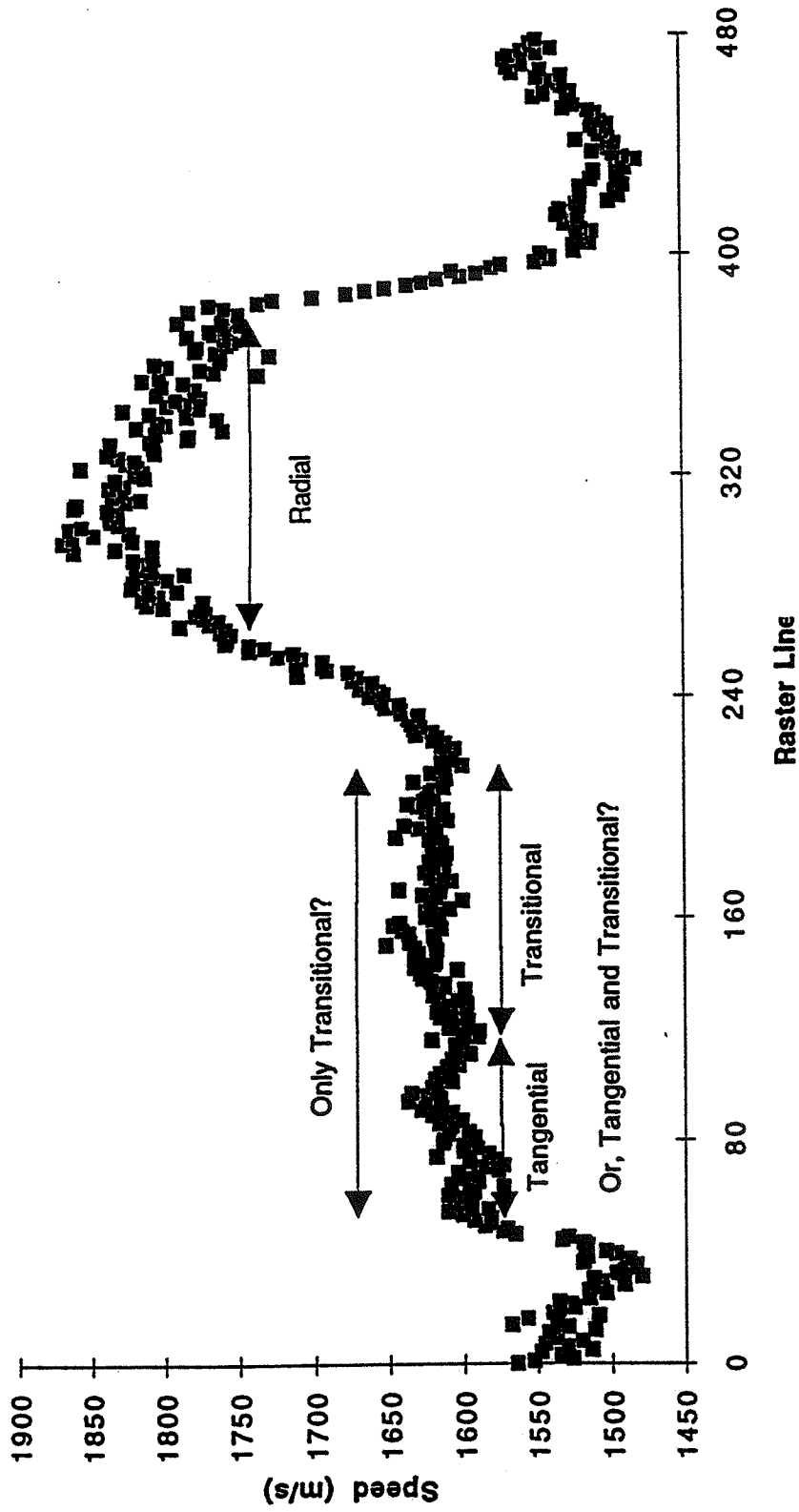


Figure 20. Speed profile for which the tangential and transitional regions are not readily distinguishable.

In summary, the experimental results showed:

- The attenuation coefficient could not be assessed, most likely due to the heterogeneity of the sample.
- The speed fluctuated in the first 6 to 12 hr as it approached a steady-state value. (See Figs. 10 and 11.)
- The  $\langle HI \rangle$  for the trypsin treatment (Fig. 14) rose and asymptotically decreased to a steady-state value during the first 6 hr of the experiment. The  $\langle HI \rangle$  for the elastase treatment (Fig. 15) rose with the application of the enzyme and then oscillated about a mean value.
- The data, for both speed and  $\langle HI \rangle$ , in the tangential and transitional regions were indistinguishable. (See Tables 3 and 6.)
- The speed data between the elastase and trypsin treatments did not vary significantly at 6, 12, and 24 hr (Table 2).

There is still much work to be done. Only three samples, four thicknesses each, were tested for each of the enzyme treatments. This provided a base from which to start, but many more samples need to be tested. Further studies need to be conducted that focus on the first 6 hr of the experiment since that is the time-frame during which the speed is most variable. Also, the collagen content and degree of cleaving of the cross-links should be assessed during the first critical hours of the experiment. The results from biochemical analysis of the tissue could provide insight as to why the acoustic properties change significantly at the start of the experiment.

APPENDIX A  
ATTENDMC PROGRAM LISTING

```

/*****
/*
/* "attendmc" calculates values for finding insertion loss and attenuation
/* coefficient. These values are stored in a file along with the
/* time that the data were collected and formatted such that they
/* can be used by the program called "regress."
/*
/*****

#include <c:\slam\isdefs.h>
#include <c:\slam\iserrs.h>
#include <conio.h>
#include <math.h>
#include <stdio.h>
#include <malloc.h>
#include <time.h>

main()
{
    int i, j=0, k, line[8], row, col, height, length, minpixel, maxpixel;
    int *red, *green, *blue, *ref, *values;
    long int bin1[256];
    float db, total, temp, curref, thick;
    char ch=0, filename[40];
    time_t ltime;
    FILE *fptr, *outptr;

/*****
/*
/* i,j,k - counters and array pointers
/* line[] - array used to store line data to frame grabber for drawing
/* row, col, height, length - specify the subimage area used for data
/* minpixel - the minimum pixel value
/* maxpixel - the maximum pixel value
/* red,green,blue - pointers to arrays that hold the values for ILUT
/* ref - pointer to array that holds the reference pixel values
/* values - pointer to array that holds subimage area pixel values
/* bin1[] - array that holds the histogram bin values
/* db - the value calculated for the subimage area
/* total - temporary value used for summing up pixel values in subimage area
/* temp - temporary variable
/* curref - average of the reference values of the current raster line
/* thick - specimen thickness
/* ch - temporary character variable for user input

```

```

/* filename[] - user-specified output file name          */
/* fptr - file pointer to subimage area box data        */
/* outptr - output file pointer                        */
/*
/*****
*/

/* allocate memory for the following arrays              */

red = (int *)calloc( 256, sizeof(int) );
if (red == NULL) { printf("red allocation failed"); exit(); }
green = (int *)calloc( 256, sizeof(int) );
if (green == NULL) { printf("green allocation failed"); exit(); }
blue = (int *)calloc( 256, sizeof(int) );
if (blue == NULL) { printf("blue allocation failed"); exit(); }
ref = (int *)calloc( 1000, sizeof(int) );
if (ref == NULL) { printf("ref allocation failed"); exit(); }
values = (int *)calloc( 10000, sizeof(int) );
if (values == NULL) { printf("values allocation failed"); exit(); }

/* open file that contains box size and location information */

fptr = fopen("c:\\slam\\attwin.pos", "r+");
fscanf(fptr, "%d %d %d %d", &row, &col, &height, &length);
fclose(fptr);

/* initialize and reset frame grabber                  */

IS_INITIALIZE();
IS_RESET();
IS_SET_SYNC_SOURCE(1);
IS_PASSTHRU();
IS_SET_GRAPHIC_POSITION(row,col);
IS_SET_FOREGROUND(1);

/* set up input look-up table 7 so that box can be overlaid on screen */

for (i=0; i<256; i+=2)
{
    red[i] = i;
    green[i] = i;
    blue[i] = i;
    red[i+1] = 255;
    green[i+1] = 0;
    blue[i+1] = 0;
}
IS_LOAD_OLUT(7,red,green,blue);

/* draw box and reference line on screen                */

line[0] = row;
line[1] = col + length - 1;
line[2] = row + height - 1;
line[3] = col + length - 1;
line[4] = row + height - 1;

```

```

line[5] = col;
line[6] = row;
line[7] = col;
IS_FRAME_CLEAR(0);
IS_DRAW_LINES(0,4,line);
IS_SET_GRAPHIC_POSITION(0,10);
line[0] = 479;
line[1] = 10;
IS_DRAW_LINES(0,1,line);
IS_LOAD_MASK(1);
IS_SELECT_ILUT(5);
IS_SELECT_OLUT(7);
IS_PASSTHRU();
IS_DISPLAY(1);

```

```

printf("\nAdjust SLAM so that image area in box is as uniform and bright");
printf("\nas possible. A black reference region must include the area");
printf("\non the left side of the screen. When complete, hit any key");
printf("\nto perform histogram. Adjust receiver gain to maximize ");
printf("\nbrightness without saturating.");
ch = getch();

```

```

/* set up display so that box and line will not be overwritten by image */

```

```

IS_ACQUIRE(0,1);
IS_SELECT_ILUT(0);
IS_LOAD_MASK(0);
IS_SELECT_OLUT(0);
IS_SELECT_INPUT_FRAME(1);
IS_SELECT_OUTPUT_FRAME(1);
IS_PASSTHRU();
IS_DISPLAY(1);

```

```

/* perform histogram on image for proper dynamic range setup */

```

```

hist(1,1);
IS_LOAD_MASK(1);
IS_SELECT_OUTPUT_FRAME(0);
IS_SELECT_INPUT_FRAME(0);
IS_SELECT_ILUT(5);
IS_SELECT_OLUT(7);
IS_PASSTHRU();
IS_DISPLAY(1);

```

```

/* obtain output file name and specimen thickness */

```

```

printf("\nOutput file name: ");
scanf("%s", filename);
outptr = fopen(filename,"a");
printf("\nSpecimen thickness (micrometers): ");
scanf("%f", &thick);
fprintf(outptr,"t %f\n",thick/1000.0);

```



```

time(&lt;time);          /* Get time and put it into the output file */
fprintf(outptr, "%s", ctime(&lt;time));

printf("\npress 'r' for reference and 's' for specimen. ");
printf("\npress ENTER to quit. \n");

/* begin taking data                                     */

while ( (ch = getch()) != 13)
{

/* acquire and average 8 frames to buffer 1 (will not overwrite box and line)*/

IS_ACQUIRE(0,1);
IS_SELECT_ILUT(0);
IS_LOAD_MASK(0);
acquire(8,1);
IS_LOAD_MASK(1);
IS_SELECT_INPUT_FRAME(0);
IS_SELECT_ILUT(5);
IS_SET_ACTIVE_REGION(0,0,512,512);

/* perform histogram to get pixel information           */

IS_HISTOGRAM(1, bin1);
IS_SET_ACTIVE_REGION(row,0,height,10);

/* get reference pixel values from left portion of screen */

IS_GET_REGION(1,ref);
IS_SET_ACTIVE_REGION(row,col,height,length);

/* get subimage area pixels from boxed region          */

IS_GET_REGION(1,values);
IS_PASSTHRU();
bin1[0] -= 16384;          /* histogram calculated for 512x512 */

/* find the minimum and maximum pixels                 */

for (i=0; i<256; i++)
{
if ( bin1[i] != 0 )
{
minpixel = i;
i = 256;
}
}
for (i=255; i>=0; i--)
{
if ( bin1[i] != 0 )
{

```

```

        maxpixel = i;
        i = -1;
    }
}

/* calculate the average reference value and subtract it from each pixel in */
/* the subimage area. Then add all of these normalized pixel values.    */

    total = 0.0;
    for (i=0; i<height; i++)
    {
        curref = 0.0;
        for (k=0; k<10; k++)
            curref += (float)ref[i*10 + k];
        curref /= 10.0;
        for (k=0; k<length; k++)
        {
            temp = (float)values[i*length + k] - curref;
            total += temp;
        }
    }

    if (total <= 0)
        total = 1.0;

/* convert the total value to decibels and display it on the screen, and */
/* write to the output file if necessary.                                */

    db = ( 10.0 * log10( total/(float)(length*height) ) );
    printf("<min: %3d max: %3d> atten: \x1B[7m%7.3f dB\x1B[0m",
           minpixel,maxpixel,db);
    if (ch=='r') printf(" R\n");
    else if (ch=='s') printf(" S\n");
    else printf(" (unrecorded)\n");

    if ( (ch == 'r') || (ch == 's') )
        fprintf(outptr, "%c %7.3f\n", ch, db);
}

/* close output file and reset frame grabber                            */

fclose(outptr);
IS_RESET();
IS_END();
}

```

APPENDIX B  
GETIMG PROGRAM LISTING

```

/*****
/* This program grabs an interference image of a specimen from the SLAM. */
/*****

#include <math.h>
#include <stdio.h>
#include <time.h>
#include <c:\slam\isdefs.h>
#include <c:\slam\iserrs.h>
#include <conio.h>                               /*****
#define DELAY 128000                             /* delay value between grabbing frames */
#define WINLEN 26                               /* DFT window length, no. sample points used */
#define LOW 28                                  /* lower bound of DFT frequency indicies checked */
#define HIGH 48                                 /* upper bound of DFT frequency indicies checked */
#define SLAMFRQ 100.0                           /* ultrasonic frequency of SLAM--100 MHz */
#define REFVEL 1520.0                           /* reference speed of sound in saline--1520 m/s */
#define DFTSIZE 512                             /* DFT size used in calculation */
#define SAMPLFREQ 9752.0                       /* sampling frequency in kHz */
#define LIMIT 3.14                             /* limit in radians used for unwrapping phase */
#define DEFAULTCOLUMN 232 /* default analysis starting column */
#define DEFAULTTOP1 0 /* default of top of reference region 1 */
#define DEFAULTBOT1 79 /* default of bottom of reference region 1 */
#define DEFAULTTOP2 400 /* default of top of reference region 2 */
#define DEFAULTBOT2 479 /* default of bottom of reference region 2 */
/*****

float phase[480];
float nvals[480];
main()
{
    FILE *outptr;
    int i, j, k, ref[2][2], xbeg, thick, skip=0, total=0, datbuff[WINLEN],
        reftotal=0, flag1=0, flag2=0, curbuf=1, avgnum, lines[2],
        tflag;
    float dcofst=0, delta, oldphase, corcoef=0, x, r, z, tempmag,
        pi=4.0*atan(1.0), mag[480], freq[480], phs[480],
        recoef[HIGH-LOW+1][WINLEN], imcoef[HIGH-LOW+1][WINLEN];
    char ch, filename[80], infile[80], repeat='y', specname[80], input[20];
    time_t ltime;

    int startcol[480],topref1[512],botref1[512],topref2[512],botref2[512],refspeed,
        stage = 0;

/*****
/*                                     */
/* phase[] - contains initially all phase values, then converted to speed */
/*****

```

```

/* ref[][] - array that contains the boundaries for the reference regions */
/* xbeg - beginning column of analysis */
/* thick - thickness of current specimen */
/* skip - skip value of analysis--raster line (0 + skip*i) */
/* total - total number of raster lines used in analysis */
/* datbuff[] - array containing the pixel values from frame grabber */
/* reftotal - total number of reference lines used in analysis */
/* flag1 - when set, indicates that reference region 1 is valid */
/* flag2 - when set, indicates that reference region 2 is valid */
/* curbuf - currently displayed buffer */
/* avgnum - number of frames averaged for analysis */
/* lines[] - array containing end point of a drawn line */
/* dcofst - the average pixel value of the current window (and raster) */
/* delta - difference between current phase value and old phase value */
/* oldphase - phase value immediately preceding current phase value */
/* corcoef - correlation coefficient of best fit phase reference line */
/* x - current pixel value minus the dc offset value (dcofst) */
/* r - sum of real parts of DFT calculation of current frequency */
/* z - sum of imaginary parts of DFT calculation of current frequency */
/* tempmag - sum of r squared plus z squared */
/* mag[] - array containing all the maximum magnitudes */
/* freq[] - array containing all the selected DFT frequencies */
/* phs[] - array containing the unwrapped phase values */
/* recoef[][] - array containing the real coefficients of DFT summation */
/* imcoef[][] - array containing the imaginary coefficients of DFT sum */
/* repeat - flag used to continue analysis--while 'y', continue */
/* input[] - array used to contain input number string */
/* startcol, topref1, botref1, topref2, botref2 - arrays temporarily
/* containing pixel values overwritten by display */
/* refspeed - speed of sound in reference medium */
/* stage - flag signifying which stage is being used */
/* tflag - when set, don't get time */
/* */
/*****

printf("\nThis program calculates the cross-sectional speed from a ");
printf("displayed image.\nDefault values for inputs are given by \"[]\".\n");

while (repeat != 'n' ) /* continue analysis until changed */
{

/***** set up frame grabber board for analysis calculation of speed *****/

IS_INITIALIZE(); /* initialize frame grabber */
IS_SELECT_OUTPUT_FRAME(0); /* select buffer 0 for image output */
IS_SELECT_ILUT(0);
IS_SELECT_OLUT(0); /* select output lookup table 0 */
IS_LOAD_MASK(0);
IS_SET_ACTIVE_REGION(0,0,512,512);
IS_DISPLAY(1); /* turn image display on */

/***** obtain desired image *****/

```

```

printf("\nIs desired image currently being displayed? [n] ");
if ( (ch=getche()) != 'y')    /* if image is display go to analysis */
{
printf("\nDo you want to use an image stored in a file? [n] ");
if ( (ch=getche()) == 'y')
{
tflag = 1;          /* Don't get time if image file */
printf("\nfile name: "); /* retrieve image from file */
scanf("%s",infile);
while ( (i=IS_RESTORE(0,0,0,infile)) != 0)
{
printf("\nreenter file name: ");
scanf("%s",infile);
}
}
}
else
{
/* if real time image desired, */
IS_RESET();          /* reset frame grabber, */
/* set sync source to external, */
if ( (i = IS_SET_SYNC_SOURCE(1)) != 0)
{
IS_END();          /* if there is no sync source, */
exit();          /* exit program. */
}
IS_PASSTHRU();          /* set display in pass-thru mode.*/
IS_DISPLAY(1);
printf("\nWould you like to perform a histogram first? [n] ");
if ( (ch=getche()) == 'y')
hist(0, 1);          /* call histogram function. */
fflush(stdin);
printf("\nHow many frames would you like to average? [8] ");
gets(input);
if (input[0] == 0)
avgnum = 8;
else
avgnum = atoi(input);

if (avgnum < 1)
avgnum = 1;
printf("\nAcquiring and averaging %d frame(s) from SLAM.",avgnum);
acquire(avgnum,0); /* call acquire funtion to acquire */
} /* and average j frames to buffer 0.*/
}

IS_SET_SYNC_SOURCE(0);          /* set sync source to internal. */

printf("\nEnter file name (8 characters + 3 for extension: ");

```

```
scanf("%s",filename);
while ( (i=IS_SAVE(0,0,1,0,filename)) != 0)
{
    printf("\nreenter file name: ");
    scanf("%s",filename);
}
IS_END();

printf("\nWould you like to continue? [y] ");
repeat = getche();
skip = 0;          /* reset skip value */
}
}
```

APPENDIX C  
REGRESS PROGRAM LISTING

```

/*****
/* SLAM REGRESSION ANALYSIS: STATISTICAL REGRESSION ANALYSIS OF */
/*          INSERTION LOSS VS THICKNESS          */
/*          TO DETERMINE THE ULTRASONIC          */
/*          ATTENUATION COEFFICIENT.            */
/*
/* THE USER INPUTS THE REFERENCE AND SAMPLE INTENSITY VALUES, */
/* NUMBER OF THICKNESSES, AND THICKNESS VALUES. THE          */
/* PROGRAM CALCULATES AVERAGE OF THE REFERENCES (VR) AND      */
/* SUBTRACTS VR FROM EACH OF THE SAMPLE INTENSITY VALUES TO */
/* GET THE INSERTION LOSS VALUES. THE NUMBER OF INTENSITY    */
/* VALUES PER THICKNESS FOR THE REFERENCE AND SAMPLE NEED   */
/* NOT BE THE SAME. REFDEX AND SMPDEX ARRAYS KEEP TRACK OF    */
/* THE NUMBER OF INTENSITY VALUES PER THICKNESS. THE        */
/* MAXIMUM NUMBER OF ENTRIES FOR 3 THICKNESSES IS 28 ( 24 FOR */
/* 4 THICKNESSES ) TO PREVENT THE SCREEN FROM SCROLLING. THE */
/* PROGRAM THEN PERFORMS THE REGRESSION ANALYSIS ON THE       */
/* DATA PAIRS. THE USER TYPES THE DIRECTORY, FILE NAME AND   */
/* TITLE FOR THE OUTPUT. THE OUTPUT FILE CONSISTS OF THE DATA */
/* ARRAY OF THICKNESS AND INSERTION LOSS VALUES, SLOPE      */
/* (ATTENUATION COEFFICIENT ), INTERCEPT, STANDARD ERROR AND */
/* CONFIDENCE INTERVALS OF BOTH THE SLOPE AND THE INTERCEPT,*/
/* NUMBER OF DATA PAIRS, F TEST AND CORRELATION COEFFICIENT. */
/* USING GRAPHIC 4.1 SUBROUTINES THE DATA IS PLOTTED AND THE */
/* REGRESSION LINE IS GRAPHED. A .PIC FILE CAN THEN BE CREATED*/
/* FOR PRINTING THE GRAPH USING LOTUS 1-2-3 OR WORD PERFECT  */
/* 5.0.
/*
/* <<<< NOTE >>>>
/* IF GRAPHIC 5.0 IS PURCHASED, WHICH HAS THE PROPER DRIVER  */
/* FOR THE HP LASER JET, A .PIC FILE WILL NOT BE NECESSARY. A */
/* PRINTOUT CAN BE GENERATED DIRECTLY. A COLOR PLOT ON THE   */
/* BRUNNING CAN ALSO BE CREATED BY
/*     1) IF GRAPHIC 5.0 HAS THE DRIVER FOR THE HP7475A      */
/*         (BRUNNING )
/*     2) CREATING A .PIC FILE AND PRINTING USING LOTUS 1-2-3 A*/

/* A LIST OF THE VARIABLES USED IS SUMMARIZED BELOW.
/* -----
/* -- FUNCTIONS -- */
/* absol() = FUNCTION THAT RETURNS THE ABSOLUTE VALUE
/*
/* plot() = FUNCTION THAT GRAPHS THE DATA USING GRAPHIC 4.1
/*
/* -- ARRAYS -- */

```

```

/* refer[ 10 ][ 20 ] = DATA ARRAY WITH REFERENCE INTENSITIES      */
/*                                                                    */
/* sample[ 10 ][ 20 ] = DATA ARRAY WITH SAMPLE INTENSITIES        */
/*                                                                    */
/*           10 = MAX NUMBER OF THICKNESSES                        */
/*           20 = MAX NUMBER OF INTENSITY VALUES PER THICKNESS   */
/*                                                                    */
/* dataprs[ 0 ][ n ] = DATA ARRAY WITH SPECIMEN THICKNESS ( mm )  */
/*                                                                    */
/* dataprs[ 1 ][ n ] = DATA ARRAY WITH INSERTION LOSS ( dB )      */
/*                                                                    */
/* tdata[ 51 ] = DATA ARRAY OF POSSIBLE T VALUES                 */
/* refavg[ 10 ] = ARRAY OF REFERENCE AVERAGES FOR EACH THICKNESS   */
/*                                                                    */
/* refdex[ 10 ] = ARRAY OF THE NUMBER OF REFERENCE INTENSITY VALUES*/
/*                FOR EACH THICKNESS                               */
/* smpdex[ 10 ] = ARRAY OF THE NUMBER OF SAMPLE INTENSITY VALUES  */
/*                FOR EACH THICKNESS                               */
/*                                                                    */
/* R = ARRAY WITH FOLLOWING RESULTS:                                */
/* R [ 1 ] = SLOPE                                                  */
/* R [ 2 ] = STANDARD ERROR OF SLOPE ( SE )                        */
/* R [ 3 ] = 95% CONFIDENCE INTERVAL FOR SLOPE                    */
/*                                                                    */
/* R [ 4 ] = INTERCEPT                                           */
/* R [ 5 ] = STANDARD ERROR OF INTERCEPT ( SE )                 */
/* R [ 6 ] = 95% CONFIDENCE INTERVAL FOR INTERCEPT              */
/*                                                                    */
/* R [ 7 ] = CORRELATION COEFFICIENT ( r )                          */
/* R [ 8 ] = PERCENT VARIATION ABOUT MEAN EXPLAINED BY THE        */
/*                REGRESSION ( R ** 2 % )                          */
/*                                                                    */
/* R [ 9 ] = F TEST FOR REGRESSION ( F )                            */
/* R [ 10 ] = VARIANCE ABOUT THE REGRESSION ( S ** 2 )             */
/*                                                                    */
/* R [ 11 ] = T VALUE FOR N-2 DEGREES OF FREEDOM AND a= 0.025 ( T )*/
/*                                                                    */
/* R [ 12 ] = SUM OF SQUARES, TOTAL, CORRECTED FOR MEAN ( SS MEAN )*/
/*                                                                    */
/* R [ 13 ] = SUM OF SQUARES DUE TO REGRESSION ( SS REG )         */
/*                                                                    */
/* R [ 14 ] = SUM OF SQUARES ABOUT THE REGRESSION DUE TO          */
/*                RESIDUALS ( SS RES )                             */
/*                                                                    */
/* R [ 15 ] = SUM ( THICKNESS * IL )                                */
/* ** R [ 16 ] = SUM THICKNESS                                     */
/* ** R [ 17 ] = SUM IL                                           */
/* ** R [ 18 ] = SUM ( THICKNESS ** 2 )                            */
/* ** R [ 19 ] = SUM ( IL ** 2 )                                   */
/* ** R [ 20 ] = SUM ( ( THICKNESS - MEAN THICKNESS ) ** 2 )     */
/* **                                                                */
/* ** input[ 20 ] = THICKNESS NUMBER FROM USER                    */
/* ** title [ LIMIT ] = TITLE OF SPECIMEN                          */
/* ** file_name [ LIMIT ] = DIRECTORY AND FILE NAME FOR OUTPUT    */
/* **                                                                */
/* **                                                                */

```



```

/** -- VARIABLES -- */
/** n = # OF DATA PAIRS */
/* refsum = SUM OF REFERENCE INTENSITIES */
/* thickness = THICKNESS VALUES */
/* del = DENOMINATOR IN THE SLOPE AND INTERCEPT EQUATIONS */
/*
/* z = DENOMINATOR IN THE CORRELATION COEFFICIENT EQUATION */
/*
/* CM = CONFIDENCE LIMIT FOR THE SLOPE */
/* CI = CONFIDENCE LIMIT FOR THE INTERCEPT */
/*
/* cursor_x; cursor_y = X AND Y POSITIONS OF THE CURSOR ON THE */
/* MONITOR */
/*
/* flag; dummy = ERROR DETECTORS, CHECKS IF VARIABLE IN SQUARE */
/* ROOT IS NEGATIVE */
/* num_thick = NUMBER OF THICKNESSES */
/* num = THICKNESS NUMBER */
/* line = LINE NUMBER */
/* a = b = INDEX FOR INCREMENTING THE DATA PAIRS ARRAY; TOTAL */
/* NUMBER OF DATA PAIRS */
/* dex; index; outdex; count = INDEXES */
/* ch = DETERMINES IF USER NEEDS TO MAKE CORRECTIONS */
/*
/* type = DETERMINES WHICH SET OF DATA NEEDS TO BE CORRECTED, */
/* REFERENCE OR SAMPLE */
/*
/*
/*
/*****

```

```

#include < math.h >
#include < stdio.h >
#include < graph.h >
#include < graphs.h >
#define ROWS 2
#define COLUMNS 200
#define LIMIT 81
#define RETURN 13
#define BLANK " "
#define NORMAL "\x1B[0m" /* \ */
#define UNDER "\x1B[4m" /* ANSI.SYS ESCAPE CHARACTERS FOR */
#define CLRSCR _clearscreen( _GWINDOW ) /* NORMAL MODE, UNDERLINE */
#define STP _settextposition /* SCREEN AND CURSOR POSITION */
#define CURPOS1( x, y ) cursor_y = 3, cursor_x = 1; /* \ */
#define CURPOS2( x, y ) cursor_y = 3, cursor_x = 40; /* \ */
#define CURPOS3( x, y ) cursor_y = 4, cursor_x = 19; /* SET CURSOR POSITION */
#define CURPOS4( x, y ) cursor_y = 3, cursor_x = 43; /* FOR COLUMN DISPLAY. */
#define CURPOS5( x, y ) cursor_y = 4, cursor_x = 59; /* INTENSITY VALUES */

```

```
float datapr[ ROWS ][ COLUMNS ], refer[ 10 ][ 20 ], sample[ 10 ][ 20 ];
```

```
float tdata [ 51 ] = { 12.706, 4.303, 3.182, 2.776, 2.571, 2.447, 2.365,
2.306, 2.262, 2.228, 2.201, 2.179, 2.160, 2.145,
```

```

2.132, 2.120, 2.110, 2.101, 2.093, 2.086, 2.080,
2.074, 2.069, 2.064, 2.060, 2.056, 2.052, 2.048,
2.045, 2.042, 2.040, 2.037, 2.034, 2.032, 2.030,
2.028, 2.026, 2.024, 2.023, 2.021, 2.009, 2.000,
1.994, 1.990, 1.987, 1.984, 1.972, 1.968, 1.966,
1.965, 1.960
};

```

```

main()
{
FILE *fptr, *outfptr;
float absol();
void plot();
float refavg[ 10 ], R [ 20 ], thick[ 10 ];
float refsum = 0, thickness = 0, n, dummy, temp;
float del = 0, z = 0, CM = 0, CI = 0;
int refdex[ 10 ], smpdex[ 10 ];
int a = 0, b = 0, i, l, m, cursor_x, cursor_y, flag;
int dex, num_thick = 0, num = 1, line = 1, index, outdex, count;
int result, option;
int rcount, scount, r_index, s_index, num_ref, num_sam;
char input[ 20 ], title[ LIMIT ], file_name[ LIMIT ], letter[ 2 ];
char file, ch, type, check;

for ( l = 0; l < 10; l++ )
{
refdex[ l ] = 0;
smpdex[ l ] = 0;
}

CLRSCR;
printf( " *****\n" );
printf( " * * * * * \n" );
printf( " * STATISTICAL REGRESSION ANALYSIS * \n" );
printf( " * * * * * \n" );
printf( " *****\n" );
printf( "\n\n\n" );
printf( "This program uses the reference and sample intensities ( dB ) and the\n" );
printf( "sample thickness ( mm ) to determine the ultrasonic attenuation coefficient.\n" );

printf( "\nIs the data in a file? [ y ] " );
fflush( stdin );
file = getche();
if ( ( file == RETURN ) || ( file == 'y' ) || ( file == 'Y' ) )
{
printf( "\nEnter file name: " );
gets( file_name );
while ( ( fptr = fopen( file_name, "r" ) ) == NULL )
{
printf( "\n*** ERROR! *** Can't open --> %s.", file_name );
printf( "\nEnter new file name: \n--> " );
gets( file_name );
}
}

while ( ( result = fscanf( fptr, "%s %f", letter, &temp ) ) != EOF )

```

```

    {
        if ( letter[ 0 ] == 'r' )
        {
            refer[ num_thick - 1 ][ refdex[ num_thick - 1 ] ] = temp;
            refdex[ num_thick - 1 ]++;
        }
        if ( letter[ 0 ] == 's' )
        {
            sample[ num_thick - 1 ][ smpdex[ num_thick - 1 ] ] = temp;
            smpdex[ num_thick - 1 ]++;
        }
        if ( letter[ 0 ] == 't' )
        {
            thick[ num_thick ] = temp;
            num_thick++;
        }
    }
    fclose( fptr );
}

if ( ( file == 'n' ) || ( file == 'N' ) )
{
    printf( "\nEnter the intensity values below. " );
    printf( "\nEnter 0 to quit.\n\n" );
    printf( "How many thicknesses? [ 3 ] " ); /* QUERY FOR # OF THICKNESSES */
    gets( input );                          /* DEFAULT: 3. */
    if ( input[ 0 ] == 0 ) /* ELSE, ASCII TO INTEGER FUNCTION CONVERTS */
        num_thick = 3; /* INPUT TO NUMBER */
    else
        num_thick = atoi( input );

/* ----- */
/* --- ENTERS THE REFERENCE INTENSITY VALUES INTO A TWO
DIMENSIONAL ARRAY --- */
/* ----- */
    CLRSCR;
    printf( "%sREFERENCE INTENSITY VALUES ( dB ): %s\n", UNDER,
NORMAL );
    CURPOS1( x, y );
    for ( dex = 0; dex < num_thick; dex++ )
    {
        if ( cursor_y >= 25 ) /* CHECKS IF CURSOR HAS REACHED BOTTOM */
            CURPOS2( x, y ); /* OF SCREEN, IF SO MOVE CURSOR TO TOP OF */
                                /* NEXT COLUMN */

        index = 0;
        STP( cursor_y, cursor_x );
        printf( "Thickness %d: \n", num++ );
        cursor_y++;
        do
        {
            if ( cursor_y >= 25 )
                CURPOS2( x, y );
            STP( cursor_y, cursor_x );
            printf( "Enter value: " );

```

```

        scanf( "%f", &refer[ dex ][ index ] );
        cursor_y++;
    }
    while ( refer[ dex ][ index++ ] != 0 );
    refdex[ dex ] = index - 1;
    cursor_y++;
}

/* ----- */
/* --- ENTERS THE SAMPLE INTENSITY VALUES INTO A TWO DIMENSIONAL
ARRAY --- */
/* ----- */
    num = 1;
    CLRSCR;
    printf( "%sSAMPLE INTENSTIY VALUES ( dB ): %s\n", UNDER, NORMAL );
    CURPOS1( x, y );
    for ( dex = 0; dex < num_thick; dex++ )
    {
        if ( cursor_y >= 25 ) /* CHECKS IF CURSOR HAS REACHEDBOTTOM OF*/
            CURPOS2( x, y ); /* SCREEN. IF SO, MOVE CURSOR TO TOP OF */
            /* NEXT COLUMN */

        index = 0;
        STP( cursor_y, cursor_x );
        printf( "Thickness %d: \n", num++ );
        cursor_y++;
        do
        {
            if ( cursor_y >= 25 )
                CURPOS2( x, y );
            STP( cursor_y, cursor_x );
            printf( "Enter value: " );
            scanf( "%f", &sample[ dex ][ index ] );
            cursor_y++;
        }
        while ( sample[ dex ][ index++ ] != 0 );
        smpdex[ dex ] = index - 1;
        cursor_y++;
    }
}

/* ----- */
/* --- PRINTS THE REFERENCE AND SAMPLE INTENSITY VALUES ( 4
COLUMNS WIDE ) ----- */
/* ----- */
    do
    {
        flag = 0;
        do
        {
            num = 1;
            CLRSCR;
            printf( "%sREFERENCE INTENSITY VALUES ( dB ) %s", UNDER,
NORMAL );
            CURPOS1( x, y );

```

```

for ( outdex = 0; outdex < num_thick; outdex++ )
{
    if ( cursor_y >= 22 )
        CURPOS3( x, y );
    STP( cursor_y, cursor_x );
    printf( "Thickness %d: \n", num++ );
    cursor_y++;
    line = 1;
    for ( count = 0; count < refdex[ outdex ]; count++ )
    {
        if ( cursor_y >= 22 )
            CURPOS3( x, y );
        STP( cursor_y, cursor_x );
        printf( "%2d: %7.4f\n", line++, refer[ outdex ][ count ] );
        cursor_y++;
    }
    cursor_y++;
}
num = 1;
STP( 1, 43 );
printf( "%sSAMPLE INTENSTIY VALUES ( dB ) %s", UNDER, NORMAL );
CURPOS4( x, y );
for ( outdex = 0; outdex < num_thick; outdex++ )
{
    if ( cursor_y >= 22 )
        CURPOS5( x, y );
    STP( cursor_y, cursor_x );
    printf( "Thickness %d: \n", num++ );
    cursor_y++;
    line = 1;
    for ( count = 0; count < smpdex[ outdex ]; count++ )
    {
        if ( cursor_y >= 22 )
            CURPOS5( x, y );
        STP( cursor_y, cursor_x );
        printf( "%2d: %7.4f\n", line++, sample[ outdex ][ count ] );
        cursor_y++;
    }
    cursor_y++;
}

/* ----- */
/* --- USER TO MAKE CORRECTIONS IF NECESSARY. USER MUST ENTER S
OR R ( SAMPLE OR --- */
/* --- REFERENCE ), THICKNESS NUMBER AND LINE NUMBER
--- */
/* ----- */
STP( 22, 1 );
printf( "1. Edit a line.\n" );
printf( "2. Delete a line.\n" );
printf( "3. Exit." );
printf( "      Enter option --> " );
fflush( stdin );
scanf( "%d", &option );

```

```

if ( option == 1 )
{
    STP( 22, 1 );
    printf( BLANK ); printf( BLANK ); printf( BLANK ); printf( BLANK );
    STP( 22, 1 );
    printf( "Enter type, thickness # ");
    printf( "and line # ( r 2 5 ): " );
    fflush( stdin );
    scanf( "%c %d %d", &type, &num, &line );
    if ( type == 'r' || type == 'R' )
    {
        printf( "Enter intensity value: " );
        fflush( stdin );
        scanf( "%f", &refer[ num - 1 ][ line - 1 ] );
    }
    else
    {
        printf( "Enter intensity value: " );
        fflush( stdin );
        scanf( "%f", &sample[ num - 1 ][ line - 1 ] );
    }
}

if ( option == 2 )
{
    do
    {
        STP( 22, 1 );
        printf( BLANK ); printf( BLANK ); printf( BLANK ); printf( BLANK
);
        STP( 22, 1 );
        printf( "Enter the type, thickness # and line #" );
        printf( " to delete ( s 2 4 ): " );
        fflush( stdin );
        scanf( "%c %d %d", &type, &num, &line );
        if ( ( type == 'r' ) || ( type == 'R' ) )
        {
            printf( "%2d %2.4f", line, refer[ num - 1 ][ line - 1 ] );
            printf( " OK to delete? ( y / n ) " );
            fflush( stdin );
            scanf( "%c", &check );
            if ( ( check == 'y' ) || ( check == 'Y' ) )
            {
                if ( line == refdex[ num - 1 ] )
                    refer[ num - 1 ][ line - 1 ] = 0.0;
                else
                    refer[ num - 1 ][ line - 1 ] = refer[ num - 1 ][ refdex[ num - 1 ] -
1 ];
                refdex[ num - 1 ]--;
            }
        }
        if ( ( type == 's' ) || ( type == 'S' ) )
        {

```

```

        printf( "%2d %2.4f", line, sample[ num - 1 ][ line - 1 ] );
        printf( "    OK to delete? ( y / n ) " );
        fflush( stdin );
        scanf( "%c", &check );
        if ( ( check == 'y' ) || ( check == 'Y' ) )
        {
            if ( line == smpdex[ num - 1 ] )
                sample[ num - 1 ][ line - 1 ] = 0.0;
            else
                sample[ num - 1 ][ line - 1 ] = sample[ num - 1 ][ smpdex[ num
- 1 ] - 1 ];
                smpdex[ num - 1 ]--;
        }
    }
    while ( ( check == 'n' ) || ( check == 'N' ) );
}
while ( option != 3 );

/* ----- */
/* --- PROMPTS USER FOR THICKNESS VALUES, ENTERS THE VALUES INTO
THE DATA ARRAY --- */
/* ----- */
do
{
    num = 1;
    CLRSCR;
    for ( outdex = 0; outdex < num_thick; outdex++ )
    {
        if ( ( file == 'n' ) || ( file == 'N' ) )
        {
            printf( "Enter thickness %d value ( mm ): ", num++ );
            scanf( "%f", &thick[ outdex ] );
        }
        for ( count = 0; count < smpdex[ outdex ]; count++ )
        {
            datapr[ 0 ][ a ] = thick[ outdex ];
            a++;
        }
    }
    if ( ( file == 'n' ) || ( file == 'N' ) )
    {
        printf( "\nDo you want to make corrections? [ n ] " );
        fflush( stdin );
        ch = getche();
    }
}
while ( ( ch == 'y' ) || ( ch == 'Y' ) );

/* ----- */
/* --- CALCULATES THE AVERAGE REFERENCE INTENSITY FOR EACH OF
THE THICKNESSES, --- */

```

```

/* --- SUBTRACTS IT FROM EACH OF THE SAMPLE VALUES TO GET THE
INSERTION LOSS AND --- */
/* --- ENTERS THE VALUES INTO THE DATA ARRAY. ---
*/
/* ----- */
for ( outdex = 0; outdex < num_thick; outdex++ )
{
    for ( count = 0; count < refdex[ outdex ]; count++ )
        refsum += refer[ outdex ][ count ];
    refavg[ outdex ] = refsum / ( float )refdex[ outdex ];
    refsum = 0;
    for ( count = 0; count < smpdex[ outdex ]; count++ )
    {
        datapr[ 1 ][ b ] = absol( sample[ outdex ][ count ] -
                                refavg[ outdex ] );
        b++;
    }
}

/* ----- */
/* --- GETS AND OPENS FILE FOR OUTPUT, SENDS TITLE, DATA PAIRS AND
REGRESSION RESULTS */
/* ----- */
CLRSCR;
printf( "\n\nEnter the directory and file name for the output: \n-->" );
fflush( stdin );
gets( file_name );
while ( ( outfptr = fopen( file_name, "w" ) ) == NULL )
{
    printf( "\n**** ERROR! **** Can't open --> %s.", file_name );
    printf( "\n\nEnter new directory and file name: \n-->" );
    gets( file_name );
}
printf( "\n\nEnter the title for the output ( 40 characters max ): \n-->" );
fflush( stdin );
gets( title );
fprintf( outfptr, "LINEAR REGRESSION ANALYSIS\n\n" );
fputs( title, outfptr );
fputs( "\n\n", outfptr );

num_ref = 0;
num_sam = 0;
for ( dex = 0; dex < num_thick; dex++ )
{
    num_ref += refdex[ dex ];
    num_sam += smpdex[ dex ];
}

index = maxnum( maxnum( num_ref, num_sam ), a );
line = 1;
r_index = rcount = s_index = scout = 0;
for ( outdex = 0; outdex < index; outdex++ )
{
    fprintf( outfptr, "%2d ", line++ );
}

```



```

if ( outdex < a )
{
    fprintf( outfptr, "Thickness = %1.3f",
            datapr[ 0 ][ outdex ] );
    fprintf( outfptr, " IL = %7.4f ", datapr[ 1 ][ outdex ] );
}
else
    fprintf( outfptr, "                " );
if ( outdex < num_ref || r_index != 0 )
{
    fprintf( outfptr, "Reference = %7.4f ",
            refer[ rcount ][ r_index++ ] );
    if ( r_index == refdex[ rcount ] )
    {
        r_index = 0;
        rcount++;
    }
}
else
    fprintf( outfptr, "                " );
if ( outdex < num_sam || s_index != 0 )
{
    fprintf( outfptr, "Sample = %7.4f\n",
            sample[ scount ][ s_index++ ] );
    if ( s_index == smpdex[ scount ] )
    {
        s_index = 0;
        scount++;
    }
}
else
    fprintf( outfptr, "                \n" );
}

/*-----*/
/*--- STATISTICAL REGRESSION ANALYSIS ---*/
/*-----*/
for ( i = 0; i < 20; i++ )
    R [ i ] = 0.0;

n = ( float )a;
for ( i = 0; i < a; i++ )
{
    R [ 15 ] = R [ 15 ] + datapr[ 0 ][ i ] * datapr[ 1 ][ i ];
    R [ 16 ] += datapr[ 0 ][ i ];
    R [ 17 ] += datapr[ 1 ][ i ];
    R [ 18 ] = R [ 18 ] + datapr[ 0 ][ i ] * datapr[ 0 ][ i ];
    R [ 19 ] = R [ 19 ] + datapr[ 1 ][ i ] * datapr[ 1 ][ i ];
}

del = R [ 18 ] * n - R [ 16 ] * R [ 16 ];
R [ 1 ] = ( R [ 15 ] * n - R [ 16 ] * R [ 17 ] ) / del;      /* SLOPE */
R [ 4 ] = ( R [ 18 ] * R [ 17 ] - R [ 16 ] * R [ 15 ] ) / del; /* INTERCEPT */
dummy = del * ( R [ 19 ] * n - R [ 17 ] * R [ 17 ] );

```

```

if ( dummy < 0 )
    flag = 1;
z = ( float )sqrt ( ( double )dummy );
R [ 7 ] = ( R [ 15 ] * n - R [ 16 ] * R [ 17 ] ) / z; /*CORRELATION COEFFICIENT*/
R [ 12 ] = R [ 19 ] - ( R [ 17 ] * R [ 17 ] / n );
R [ 13 ] = R [ 1 ] * ( R [ 15 ] - R [ 16 ] * R [ 17 ] / n );
R [ 14 ] = R [ 12 ] - R [ 13 ];

for ( i = 0 ; i < a; i++ )
    R [ 20 ] = R [ 20 ] + ( datapr[ 0 ][ i ] - R [ 16 ] / n ) *
        ( datapr[ 0 ][ i ] - R [ 16 ] / n );

if ( n - 2 == 0 )
    R [ 10 ] = 0.0;
else
    R [ 10 ] = R [ 14 ] / ( n - 2 );

dummy = R [ 10 ] / R [ 20 ];
if ( dummy < 0 )
    flag = 1;
R [ 2 ] = ( float )sqrt( ( double )dummy ); /* STANDARD ERROR OF SLOPE */
dummy = R [ 10 ] * R [ 18 ] / ( n * R [ 20 ] );
R [ 5 ] = ( float )sqrt( ( double )dummy ); /*STANDARD ERROR OF INTERCEPT*/
if ( dummy < 0 )
    flag = 1;
if ( flag == 1 )
    {
        printf( "\n\n** ERROR! ** VARIABLE IN SQUARE ROOT LESS THAN
ZERO" );
        printf( "\nPress any key to continue or 'q' to exit" );
        while ( !kbhit() )
            {
                ch = getche();
                if ( ch == 'q' || ch == 'Q' )
                    exit( 1 );
            }
    }
while ( flag == 1 );

if ( R [ 10 ] == 0.0 )
    R [ 9 ] = 0.0;
else
    R [ 9 ] = R [ 13 ] / R [ 10 ]; /* F TEST */

R [ 8 ] = 100.0 * ( R [ 13 ] / R [ 12 ] );
n -= 2;

if ( n <= 40 )
    i = n - 1;
else if ( n <= 50 )
    i = 40;
else if ( n <= 60 )
    i = 41;

```

```

else if ( n <= 70 )
    i = 42;
else if ( n <= 80 )
    i = 43;
else if ( n <=90 )
    i = 44;
else if ( n <= 100 )
    i = 45;
else if ( n <= 200 )
    i = 46;
else if ( n <= 300 )
    i = 47;
else if ( n <= 400 )
    i = 48;
else if ( n <= 500 )
    i = 49;
else
    i = 50;

R [ 11 ] = tdata [ i ];
R [ 3 ] = R [ 11 ] * R [ 2 ];
R [ 6 ] = R [ 11 ] * R [ 5 ];
n += 2;
CM = R [ 11 ] * R [ 2 ];
CI = R [ 11 ] * R [ 5 ];
                                /* CONFIDENCE LIMIT OF SLOPE */
                                /* CONFIDENCE LIMIT OF INTERCEPT */

fprintf( outfptr,
        "\n\n\n           Standard Error      95%% Confidence Interval\n" );
fprintf( outfptr,
        "Slope %14.4f %17.4f %23.4f -- %3.4f\n",
        R [ 1 ], R [ 2 ], R [ 1 ] + CM, R [ 1 ] - CM );
fprintf( outfptr,
        "Intercept %10.4f %17.4f %23.4f -- %3.4f\n",
        R [ 4 ], R [ 5 ], R [ 4 ] + CI, R [ 4 ] - CI );
fprintf( outfptr, "\nn = %2.0f", n );
fprintf( outfptr, "\nF = %3.4f", R [ 9 ] );
fprintf( outfptr, "\nCorrelation Coefficient = %3.4f", R [ 7 ] );
fprintf( outfptr, "\n\n" );
fclose( outfptr );

/* ----- */
/* --- FUNCTION TO PLOT DATA AND REGRESSION LINE --- */
/* ----- */
plot( datapr[ 0 ], datapr[ 1 ], a, R [ 1 ], R [ 4 ], datapr[ 0 ][ a - 1 ],
      R [ 7 ], title );

}

/* ----- */
/* --- FUNCTION RETURNS ABSOLUTE VALUE OF NUMBER --- */
/* ----- */
float absol( x )

```

```

float x;
{
  if ( x < 0 )
    return ( -x );
  else
    return ( x );
}

int maxnum( x, y )
int x, y;
{
  if ( x > y )
    return( x );
  else
    return( y );
}

/* ----- */
/* --- GRAPHIC 4.1 SUBROUTINES --- */
/* ----- */
void plot( x, y, npts, slope, intercept, thick, r, title )
float x[ 200 ], y[ 200 ], slope, intercept, thick, r; /* y[ 200 ] = ARRAY WITH
                                                    /* INSERTION LOSS. */
int npts; /* x[ 200 ] = ARRAY WITH THICKNESS. */
char title[ 81 ]; /* thick = MAXIMUM THICKNESS VALUE. */
/* npts = # OF DATA PAIRS. */
{
  /* nydiv = # OF Y AXIS DIVISIONS. */
  int nxdiv, nydiv; /* nxdiv = # OF X AXIS DIVISIONS. */
  float xline[ 2 ], yline[ 2 ]; /* xline[ 2 ]; yline[ 2 ] = X AND Y PAIRS OF THE*/
  /* REGRESSION LINE. */
  char buffer1[ 100 ], buffer2[ 100 ]; /* buffer1[ 100 ] = STRING: REGRESSION LINE*/
  /* EQUATION. */
  bgnplot( 1, 'g', "linear.tkf" ); /* buffer2[ 100 ]=STRING: CORRELATION*/
  /* COEFFICIENT. */
  startplot( 0 );
  font( 4, "simplex.fnt", '\310', "triplex.fnt", '\311', "complex.fnt", '\312', "duplex.fnt",
  '\313' );
  page( 9.0, 0.0005 ); /* << 1ST BOX >> */
  pgshift( 0.0, 6.8545 ); /* SHIFTS PAGE FROM LOWER LEFT HAND CORNER */
  color( 14 ); /* COLOR */
  charspc( 1 ); /* CHARACTER SPACING */
  ctline( title, .27 ); /* CENTERS STRING PLACED BELOW PAGE SPECIFICATION*/
  /* ctline( STRING, HEIGHT ) */
  page( 9.0, 1.427 ); /* << 2ND BOX >> */
  pgshift( 0.0, 5.4275 );
  area2d( 8.5, 1.4 ); /* PLOT AREA */
  color( 2 );
  box(); /* DRAWS BOX AROUND PAGE AREA */

  sprintf( buffer1, "IL = (%3.2f)THICKNESS + (%3.2f)", slope, intercept );
  sprintf( buffer2, "r = %1.3f", r ); /* CONVERTS " " INTO STRING */
}

```

```

color( 14 );
charspc( 0 );                               /* DEFAULT CHARACTER SPACING */
prtfnt( 2.5, 0.5, buffer1, .15, 0 );        /* PRINTS STRING */
prtfnt( 4.0, 0.2, buffer2, .15, 0 );        /* prtfnt( X, Y, STRING, HEIGHT, ANGLE ) */
                                           /* X FROM LEFT EDGE, Y FROM LOWER LEFT OF PAGE */
page( 9.0, 5.4275 );                         /* << 3RD BOX, PLOT >> */
area2d( 8.0, 5.0 );
pgshift( 0.0, 0.0 );

xline[ 0 ] = 0.0;    /* REGRESSION LINE IS GRAPHED BY USING 2 POINTS: */
yline[ 0 ] = intercept; /* Y INTERCEPT AND SOME ARBITRARY VALUE */
xline[ 1 ] = thick + 0.25; /* GREATER THAN MAXIMUM THICKNESS VALUE */
yline[ 1 ] = slope * xline[ 1 ] + intercept;

color( 3 );
box();
grid( 1 );    /* DRAWS GRID */
upright( 1 ); /* UPRIGHT Y AXIS LABELS */
color( 10 );
nxdiv = 4;    /* # X AXIS DIVISIONS */
nydiv = 5;    /* # Y AXIS DIVISIONS */
scales( nxdiv, nydiv, x, y, npts ); /* SELF- SCALING AXES */
color( 13 );
titlht( 0.20 ); /* HEIGHT OF TITLE CHARACTERS */
heading( "\310INSERTION LOSS VS THICKNESS" );
xname( "\310THICKNESS ( mm )" );
yname( "\310LOSS ( dB )" );
color( 12 );
curve( xline, yline, 2, 0 ); /* DRAWS REGRESSION LINE, 0 = DRAW LINE ONLY */
curve( x, y, npts, -1 );    /* PLOTS DATA POINTS, -1 = SYMBOLS ONLY */
endplot();
stopplot();
}

```

APPENDIX D  
SPEEDN PROGRAM LISTING

```

/*****
/* This program uses the selected interference image of a specimen to find the speed of
/* sound in that specimen. The program uses a spatial frequency method to calculate the
/* speed after selecting a desired reference region and then stores and plots the resulting
/* data.
*****/

#include <math.h>
#include <stdio.h>
#include <c:\slam\isdefs.h>
#include <c:\slam\iserrs.h>
#include <conio.h>          /*****
#define DELAY 128000      /* delay value between grabbing frames          */
#define WINLEN 26        /* DFT window length, no. sample points used          */
#define LOW 28           /* lower bound of DFT frequency indicies checked      */
#define HIGH 48          /* upper bound of DFT frequency indicies checked     */
#define SLAMFRQ 100.0    /* ultrasonic frequency of SLAM--100 MHz              */
#define REFVEL 1520.0    /* reference speed of sound in saline--1520 m/s       */
#define DFTSIZE 512     /* DFT size used in calculation                       */
#define SAMPLFREQ 9752.0 /* sampling frequency in kHz                          */
#define LIMIT 3.14      /* limit in radians used for unwrapping phase         */
#define DEFAULTCOLUMN 232 /* default analysis starting column                   */
#define DEFAULTTOP1 0   /* default of top of reference region 1                */
#define DEFAULTBOT1 79 /* default of bottom of reference region 1             */
#define DEFAULTTOP2 400 /* default of top of reference region 2                */
#define DEFAULTBOT2 479 /* default of bottom of reference region 2            */
*****/

float phase[480];
float nvals[480];
main()
{
    FILE *outptr;
    int i, j, k, ref[2][2], xbeg, thick, skip=0, total=0, datbuff[WINLEN],
        reftotal=0, flag1=0, flag2=0, curbuf=1, avgnum, lines[2];
    float dcoffst=0, delta, oldphase, corcoef=0, x, r, z, tempmag,
        pi=4.0*atan(1.0), mag[480], freq[480], phs[480],
        recoef[HIGH-LOW+1][WINLEN], imcoef[HIGH-LOW+1][WINLEN];
    char ch, filename[80], infile[80], repeat='y', specname[80], input[20];

    int startcol[480],topref1[512],botref1[512],topref2[512],botref2[512],refspeed,
        stage = 0;

```

```

/*****
/*
/* phase[] - contains initially all phase values, then converted to speed */
/* ref[][] - array that contains the boundaries for the reference regions */
/* xbeg - beginning column of analysis */
/* thick - thickness of current specimen */
/* skip - skip value of analysis--raster line (0 + skip*i) */
/* total - total number of raster lines used in analysis */
/* datbuff[] - array containing the pixel values from frame grabber */
/* reftotal - total number of reference lines used in analysis */
/* flag1 - when set, indicates that reference region 1 is valid */
/* flag2 - when set, indicates that reference region 2 is valid */
/* curbuf - currently displayed buffer */
/* avgnum - number of frames averaged for analysis */
/* lines[] - array containing end point of a drawn line */
/* dcofst - the average pixel value of the current window (and raster) */
/* delta - difference between current phase value and old phase value */
/* oldphase - phase value immediately preceding current phase value */
/* corcoef - correlation coefficient of best fit phase reference line */
/* x - current pixel value minus the dc offset value (dcofst) */
/* r - sum of real parts of DFT calculation of current frequency */
/* z - sum of imaginary parts of DFT calculation of current frequency */
/* tempmag - sum of r squared plus z squared */
/* mag[] - array containing all the maximum magnitudes */
/* freq[] - array containing all the selected DFT frequencies */
/* phs[] - array containing the unwrapped phase values */
/* recoef[][] - array containing the real coefficients of DFT summation */
/* imcoef[][] - array containing the imaginary coefficients of DFT sum */
/* repeat - flag used to continue analysis--while 'y', continue */
/* input[] - array used to contain input number string */
/* startcol, topref1, botref1, topref2, botref2 - arrays temporarily
/* containing pixel values overwritten by display */
/* refspeed - speed of sound in reference medium */
/* stage - flag signifying which stage is being used */
/*
/*****

```

```

printf("\nThis program calculates the cross-sectional speed from a ");
printf("displayed image.\nDefault values for inputs are given by \"[]\".\n");

```

```

while (repeat != 'n') /* continue analysis until changed */
{

```

```

/* set up frame grabber board for analysis calculation of speed
*/

```

```

IS_INITIALIZE(); /* initialize frame grabber */
IS_SELECT_OUTPUT_FRAME(0); /* select buffer 0 for image output */
IS_SELECT_ILUT(0);
IS_SELECT_OLUT(0); /* select output lookup table 0 */
IS_LOAD_MASK(0);
IS_SET_ACTIVE_REGION(0,0,512,512);
IS_DISPLAY(1); /* turn image display on */

```

```

/* obtain desired image on which to perform analysis
*/

printf("\nIs desired image currently being displayed? [n] ");

if ( (ch=getche()) != 'y')    /* if image is display go to analysis */
{
printf("\nDo you want to use an image stored in a file? [n] ");
if ( (ch=getche()) == 'y')
{
printf("\nfile name: ");    /* retrieve image from file */
scanf("%s",infile);
while ( (i=IS_RESTORE(0,0,0,infile)) != 0)
{
printf("\nreenter file name: ");
scanf("%s",infile);
}
}
}
else
{
/* if real time image desired, */
IS_RESET();          /* reset frame grabber, */
/* set sync source to external, */
if ( (i = IS_SET_SYNC_SOURCE(1)) != 0)
{
IS_END();           /* if there is no sync source, */
exit();             /* exit program. */
}
IS_PASSTHRU();      /* set display in pass-thru mode.*/
IS_DISPLAY(1);
printf("\nWould you like to perform a histogram first? [n] ");
if ( (ch=getche()) == 'y')
hist(0, 1);        /* call histogram function. */
fflush(stdin);
printf("\nHow many frames would you like to average? [8] ");
gets(input);
if (input[0] == 0)
avgnum = 8;
else
avgnum = atoi(input);

if (avgnum < 1)
avgnum = 1;
printf("\nAcquiring and averaging %d frame(s) from SLAM.",avgnum);
acquire(avgnum,0); /* call acquire funtion to acquire */
} /* and average j frames to buffer 0.*/

}
IS_SET_SYNC_SOURCE(0); /* set sync source to internal. */

/* gather analysis information */

```



```

printf("\nEnter specimen thickness in micrometers: ");
scanf("%d",&thick);

fflush(stdin);
printf("\nWhat is the reference medium speed (m/s) ? [1520]");
gets(input);
if (input[0] == 0)
    refspeed = 1520;
else
    refspeed = atoi(input);
if (refspeed <= 0)
    refspeed = 1520;

/* determine which stage is being used (effects speed calculation) */

stage = 0;
printf("\nWhich stage is being used? ('g' for glass, 'w' for water) [g] ");
if ( (ch = getche()) == 'w')
    stage = 1;

/* skip value determines whether a full analysis (480 lines) or a reduced */
/* (and faster) analysis is performed. A skip of k uses every kth line. */

while ((skip < 1) || (skip > 20))
{
    fflush(stdin);
    printf("\nEnter skip value ([1] uses all 480 raster lines): ");
    gets(input);
    if (input[0] == 0)
        skip = 1;
    else
        skip = atoi(input);
    if ((skip < 1) || (skip > 20))
        printf("\n** skip value must be between 1 and 20. **");
}

IS_GET_PIXEL(0,DEFAULTTOP1,0,512,topref1);
IS_GET_PIXEL(0,DEFAULTBOT1,0,512,botref1);
IS_GET_PIXEL(0,DEFAULTTOP2,0,512,topref2);
IS_GET_PIXEL(0,DEFAULTBOT2,0,512,botref2);
IS_SET_ACTIVE_REGION(0,DEFAULTCOLUMN,480,1);
IS_GET_REGION(0,startcol);
IS_SET_FOREGROUND(255);
IS_SET_GRAPHIC_POSITION(DEFAULTTOP1,0);
lines[0] = DEFAULTTOP1; lines[1] = 511;
IS_DRAW_LINES(0,1,lines);
IS_SET_GRAPHIC_POSITION(DEFAULTBOT1,0);
lines[0] = DEFAULTBOT1;
IS_DRAW_LINES(0,1,lines);
IS_SET_GRAPHIC_POSITION(DEFAULTTOP2,0);
lines[0] = DEFAULTTOP2;
IS_DRAW_LINES(0,1,lines);

```

```

IS_SET_GRAPHIC_POSITION(DEFAULTBOT2,0);
lines[0] = DEFAULTBOT2;
IS_DRAW_LINES(0,1,lines);
IS_SET_GRAPHIC_POSITION(0,DEFAULTCOLUMN);
lines[0] = 479; lines[1] = DEFAULTCOLUMN;
IS_DRAW_LINES(0,1,lines);

printf("\nDo you want to use the default region specifications? [n] ");
ch = getch();

IS_PUT_REGION(0,startcol);
IS_PUT_PIXEL(0,DEFAULTTOP1,0,512,topref1);
IS_PUT_PIXEL(0,DEFAULTBOT1,0,512,botref1);
IS_PUT_PIXEL(0,DEFAULTTOP2,0,512,topref2);
IS_PUT_PIXEL(0,DEFAULTBOT2,0,512,botref2);

if ( ch != 'y')
    /* set reference regions manually */
    {
    printf("\nUse arrow keys to move cursor to desired postion.");
    printf("\nArrow keys on number pad with NUM LOCK on move by 10's");
    printf("\nHit \"enter\" to select postion.");
    printf("\nSelect column where velocity will be calculated.");
    cursor(&j,&xbeg);

do
    /* continue region setting until */
    /* both selected regions are valid. */
    {
    flag1 = 0;
    flag2 = 0;
    printf("\nSelect start and end rows for two reference regions; ");
    printf("\nsetting bottom row at value lower than top row disables");
    printf(" region.");
    printf("\nReference region 1: select top row. ");
    /* cursor function returns current position */
    cursor(&ref[0][0],&j);
    printf("\nReference region 1: select bottom row.");
    cursor(&ref[0][1],&j);
    /* if bottom row < top row, disable region 1 */
    /* by setting flag1. */
    if (ref[0][0] > ref[0][1])
        flag1 = 1;
    printf("\nReference region 2: select top row. ");
    cursor(&ref[1][0],&j);
    printf("\nReference region 2: select bottom row. ");
    cursor(&ref[1][1],&j);
    if (ref[1][0] > ref[1][1])
        flag2 = 1; /* if bottom row < top row, disable region 2 */
    /* by setting flag2. */

    /* if both regions disabled, report error. */
    if ((flag2 == 1) && (flag1 == 1))
        printf("\n*** Can't disable both regions. ***");
    }
}

```

```

    }

    while ((flag1 == 1) && (flag2 == 1)); /* condition of do while loop */

    }

else
    /* default region specifications: */
    {
    xbeg = DEFAULTCOLUMN; /* starting column. */
    ref[0][0] = DEFAULTTOP1; /* region 1 start. */
    ref[0][1] = DEFAULTBOT1; /* region 1 end. */
    ref[1][0] = DEFAULTTOP2; /* region 2 start. */
    ref[1][1] = DEFAULTBOT2; /* region 2 end. */
    }

/* obtain information for output data file. */

printf("\nEnter output data file name: ");
scanf("%s",filename);
fflush(stdin);
printf("\nEnter the name of the specimen: \n--> ");
gets(specname);
printf("\n\n");

/* calculate DFT summation exponential coefficients
*/

for (j=LOW; j<=HIGH; j++) /* only calculate for necessary */
    { /* frequencies. */
    for (k=0; k<WINLEN; k++) /* also only for WINLEN values */
        { /* (the other DFTSIZE-WINLEN values */
        x = 2.0*pi*(float)(j*k)/(float)DFTSIZE;
        /* are effectively zero-padded). */
        recoeff[j-LOW][k] = cos(x); /* calculate real part of coef. */
        imcoeff[j-LOW][k] = -sin(x); /* calculate imag. part of coef. */
        }
    }

/*****
/* begin analysis: find frequency with peak DFT magnitude and record phase */
*****/

total = 0; /* reset raster line total. */
for (i=0; i<480; i+=skip) /* use only every (skip value) line */
    {
    dcoffst = 0.0; /* reset dc offset. */
    mag[i] = 0.0; /* reset current maximum magnitude. */
    if (i%50==0) /* print every 50th index as time. */
        printf("%d ",i); /* until completion indicator. */
    }

/* obtain data from frame grabber length WINLEN starting with value at xbeg */
IS_GET_PIXEL(0,i,xbeg,WINLEN,datbuff);
for (j=0; j<WINLEN; j++)

```

```

        dcoffst += (float)datbuff[j];
        dcoffst /= (float)WINLEN; /* calculate average (dc) value. */

/* calculate DFT of current data sequence using precalculated exponentials. */
for (j=0; j<=HIGH-LOW; j++)
{
    r = 0.0; z = 0.0; /* reset DFT summations. */
    for (k=0; k<WINLEN; k++)
    {
        /* subtract dc value before mult. */
        x = ( (float)datbuff[k] - dcoffst );
        r += x * recoef[j][k];
        z += x * imcoef[j][k];
    }
    /* find if magnitude of DFT of current frequency is greater than max. */
    if ( (tempmag=sqrt(r*r + z*z) ) > mag[i] )
    {
        /* if new maximum, */
        mag[i] = tempmag; /* change maximum, */
        phase[i] = -atan2(z,r) + pi; /* calculate phase, */
        freq[i] = SAMPLFREQ/DFTSIZE*(j+LOW); /* find frequency */
    }
}
total = total + 1; /* increment total and continue on next */
} /* raster line. */
IS_END(); /* turn off frame grabber driver. */

/* unwrap phase data */

oldphase = phase[0]; /* set old phase to first phase value */
phs[0] = phase[0];
for (i=skip; i<480; i+=skip)
{
    delta = phase[i] - oldphase; /* calculate delta. */
    oldphase = phase[i]; /* update old phase. */

/* unwrap the data: if the difference between current phase and the most */
/* recent phase value (oldphase) is greater the limit then offset phase */
/* by the require amount. */
    if (fabs(delta) < LIMIT)
        phase[i] = phase[i-skip] + delta;
    else
        phase[i] = phase[i-skip] + delta - 2.0*pi*(int)(delta/pi);
    phs[i] = phase[i];
}
/* call speed funtion to convert phase to speed */

speed(ref, thick, &corcoef, skip, refspeed, stage, &reftotal);
/* open file to record data */
while ( (outptr=fopen(filename,"w")) == NULL)
{
    printf("\nCan't open file %s, enter new name: ",filename);
    scanf("%s",filename);
    fflush(stdin);
}

```

```

fprintf(outptr, "\nTotal number of raster lines used: %d", total);
fprintf(outptr, "\nFor specimen: ");
fputs(specname,outptr);
fprintf(outptr, "\nSpeed analysis calculated at column %d", xbeg);
if (flag1 != 1)
fprintf(outptr, "\nReference region 1: starting row - %d ending row - %d",
        ref[0][0],ref[0][1]);
else
    fprintf(outptr, "\nReference region 1: DISABLED");
if (flag2 != 1)
fprintf(outptr, "\nReference region 2: starting row - %d ending row - %d",
        ref[1][0],ref[1][1]);
else
    fprintf(outptr, "\nReference region 2: DISABLED");
fprintf(outptr, "\nTotal number of reference lines used: %d", reftotal);
fprintf(outptr, "\nCorrelation of best fit reference phase: %f",corcoef);
fprintf(outptr, "\nSpecimen thickness: %d micrometers", thick);
fprintf(outptr, " Reference speed: %d m/s", refspeed);
fprintf(outptr, "\n\nLine   Frequency   Phase   Speed   N\n");
fprintf(outptr, "      (kHz)      (rad)      (m/s)\n\n");
for (i=0; i<480; i+=skip)
    fprintf(outptr, "%-4d   %5.1f   %-8.5f   %10.5f   %10.5f\n",
            i,freq[i],phs[i], phase[i],nvals[i]);
fclose(outptr);
/* call velplot funtion to plot the speed data */
velplot(phase,total,skip);

IS_INITIALIZE();
IS_SET_ACTIVE_REGION(0,xbeg,480,1);
IS_GET_REGION(0,startcol);
IS_SET_GRAPHIC_POSITION(0,xbeg);
lines[0] = 479; lines[1] = xbeg;
IS_DRAW_LINES(0,1,lines);

printf("\nHit SPACE BAR to toggle between image and plot or ENTER ");
printf("to continue");
curbuf = 1;
while ( (ch=getch()) != 13)
    {
    if (curbuf == 0)
        {
        IS_SELECT_OLUT(7);
        curbuf = 1;
        }
    else
        {
        IS_SELECT_OLUT(0);
        curbuf = 0;
        }
    IS_SELECT_OUTPUT_FRAME(curbuf);
    }
IS_PUT_REGION(0,startcol);
IS_END();
printf("\nDo you want to save this image in a file? [y] ");

```

```

if ( (ch=getche()) != 'n')
{
printf("\nEnter file name (8 characters + 3 for extension: ");
scanf("%s",filename);
IS_INITIALIZE();
while ( (i=IS_SAVE(0,0,1,0,filename)) != 0)
{
printf("\nreenter file name: ");
scanf("%s",filename);
}
IS_END();
}

printf("\nWould you like to perform statistical analysis on the current ");
printf("speed plot?\n");
printf("(analysis can only be performed if skip = 1 ) [n] ");
if ( ( (ch = getche()) == 'y') && (skip == 1) )
speed_stats(phase, nvals, thick, xbeg, specname, filename);

printf("\nWould you like to continue? [y] ");
repeat = getche();
skip = 0;          /* reset skip value */
}
}

/*****
/* Speed Function
/* This function takes an array of phases and converts it to an array of
/* speeds for each raster line used. The speeds are calculated using the
/* spatial frequency domain algorithm.
*****/

speed(int ref[2][2], int thick, float *r, int skip, int refspeed, int stage,
int *total)
{
int i, j, k, l;
float wavelng, theta, sum_x=0., sum_y=0., sum_xx=0., sum_yy=0., temp,n=0.0,
sum_xy=0., delta, b, m, fn, v1, v2, v3, v4, pi=4.0*atan(1.0);

/*****
/* ref[][] - the array of the start and end points of the reference regions
/* thick - thickness of the specimen
/* *r - pointer to the correlation coefficient
/* skip - the skip value
/* refspeed - the reference speed in meters per second
/* stage - flag to tell which stage is being used
/* *total - pointer to the total number of reference lines used
/* wavelng - wavelength of sound in the reference medium
/* theta - angle from normal of the acoustic beam in the reference medium
/* sum_x - sum of the raster line indices
*****/

```

```

/* sum_y - sum of the phases */
/* sum_xx - sum of the squares of the raster line indices */
/* sum_yy - sum of of the squares of the phases */
/* sum_xy - sum of the product of the raster line indices and phases */
/* delta, fn, v1, v2, v3, v4, temp - temporary storage variables */
/* m - slope of the best-fit reference-phase line */
/* b - intercept of the best-fit reference-phase line */
/* n - total number of reference lines used for least squares algorithm */
/*****/

/* calculate some values required for algorithm */

wavelng = (float)refspeed/SLAMFRQ;

if (stage == 0) /* glass stage */
    theta = asin((float)refspeed*0.707107/5968.0);
else /* water stage */
    theta = asin((float)refspeed*sin(10.0*pi/180.0)/1509.0);

/* This section calculates a best-fit reference-phase using a least squares algorithm */
/* using the ref[][] array and the skip value find the correct values. */

if ((ref[0][0] % skip) != 0)
    j = (ref[0][0] - (ref[0][0] % skip) + skip);
else
    j = ref[0][0];
k = ref[0][1];
for (l=1; l<=2; l++)
    {
    for (i=j; i<=k; i+=skip)
        {
        n += 1.0;
        sum_x += (float)i;
        sum_y += phase[i];
        sum_xx += (float)i * (float)i;
        sum_yy += phase[i] * phase[i];
        sum_xy += (float)i * phase[i];
        }
    if ((ref[1][0] % skip) != 0)
        j = (ref[1][0] - (ref[1][0] % skip) + skip);
    else
        j = ref[1][0];
    k = ref[1][1];
    }

/* calculate the slope and intercept of the best-fit reference phase line */

delta = sum_xx*n - sum_x*sum_x;
m = (sum_xy*n - sum_x*sum_y)/delta;
b = (sum_xx*sum_y - sum_x*sum_xy)/delta;
temp = delta * fabs(sum_yy*n-sum_y*sum_y);

/* calculate the correlation coefficient
*/

```

```

if (temp > 0.0)
    *r = (sum_xy*n - sum_x*sum_y)/sqrt(temp);
v1 = thick * sin(theta);
v2 = v1 * tan(theta);
v3 = (float)refspeed/sin(theta);

/* once all values and the best-fit line have been calculated, perform */
/* conversion of the phase values to speed values for each raster line. */

for (i=0; i<480; i+=skip)
    {
    fn = (phase[i] - (m * (float)i + b))/(2.0*pi);
    v4 = fn * wavelng * tan(theta);
    phase[i] = v3 * sin(atan2(v2, (v1 - v4)));
    nvals[i] = fn;
    }
*total = (int)n;
}

speed_stats(float speed[480], float nvals[480],int thick, int col, char title[], char filename[])
{
FILE *outfptr;
float mean,stdev,sum,n_mean,n_stdev,n_sum;
int i,j,total=480,offset,rline,region=0,leftb,rightb;
char line[80];
printf("\n\noutput file name (specifying an existing file will ");
printf(" cause current data \nto be appended to previous data: ");
scanf("%s",line);
while ( (outfptr = fopen(line, "a")) == NULL)
    {
    printf("\nCan't open file %s. Enter new file name: ",line);
    scanf("%s",line);
    }
fprintf(outfptr,"Data from the file: %s\n", filename);
fputs(title, outfptr);
fprintf(outfptr, "\nSpeed analysis calculated at column %d\n", col);
fprintf(outfptr, "Specimen thickness: %d micrometers\n", thick);
printf("\nHow many regions would you like to specify? ");
scanf("%d", &region);
for (i=1; i<=region; i++)
    {
    printf("\nSelect left boundary of region %d. ", i);
    index_pointer(&leftb);
    printf("%d",leftb);
    printf("\nSelect right boundary of region %d. ", i);
    index_pointer(&rightb);
    while (rightb < leftb)
        {
        printf("\nRIGHT BOUNDARY MUST BE GREATER THAN LEFT
BOUNDARY.");
        printf("\nSelect right boundary of region %d. ", i);

```



```

        index_pointer(&rightb);
    }
    printf("%d",rightb);
    total = rightb - leftb + 1;
    sum = 0.0;
    n_sum = 0.0;
    for (j=leftb; j<=rightb; j++)
    {
        sum += speed[j];
        n_sum += nvals[j];
    }
    mean = sum/(float)total;
    n_mean = n_sum / (float)total;
    sum = 0.0;
    n_sum = 0.0;
    for (j=leftb; j<=rightb; j++)
    {
        sum += (speed[j] - mean)*(speed[j] - mean);
        n_sum += (nvals[j] - n_mean)*(nvals[j] - n_mean);
    }
    stdev = sqrt((double)(sum/(float)total));
    n_stdev = sqrt((double)(n_sum/(float)total));
    printf("\nREGION %d: lines %3d to %3d mean = %10.5f std = %8.5f\n
N_std = %8.5f\n",
        i, leftb, rightb, mean, stdev, n_stdev);
    fprintf(outfptr,
        "\nREGION %d: lines %3d to %3d mean = %10.5f std = %8.5f\n
N_std = %8.5f",
        i, leftb, rightb, mean, stdev,n_stdev);
    }
    fprintf(outfptr, "\n\n\n");
    printf("\n\n");
    fclose(outfptr);
}

```

## APPENDIX E

### SPEED DATA FROM THE SLAM

The following tables contain the time-course speed values for each sample.

**Data for each of the thicknesses from the typsin studies.**

**40  
μm**

Time (hr)	A3A			B2A1			B2A2		
	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)
0	1666	1668	1763	1615	1575	1852	1567	1554	1716
0.03	1626	1590	1699	1645	1607	1731	1612	1593	1832
1	1645	1642	1789	-	-	-	-	-	-
2	1641	1626	1737	-	-	-	-	-	-
3	1645	1606	1694	-	-	-	-	-	-
4	1643	1651	1802	-	-	-	-	-	-
5	1605	1587	1722	-	-	-	-	-	-
6	1576	1548	1699	1581	1568	1722	1611	1584	1771
12	1601	1585	1696	1626	1611	1670	1564	1562	1661
24	1622	1612	1661	1583	1576	1680	1604	1595	1681

**50  
μm**

Time (hr)	A3A			B2A1			B2A2		
	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)
0	1619	1612	1687	1587	1585	1676	1635	1601	1795
0.03	1564	1569	1645	1650	1632	1777	1565	1560	1718
1	1545	1522	1684	-	-	-	-	-	-
2	1598	1582	1673	-	-	-	-	-	-
3	1558	1575	1717	-	-	-	-	-	-
4	1541	1530	1683	-	-	-	-	-	-
5	1631	1646	1694	-	-	-	-	-	-
6	1591	1584	1677	1603	1600	1724	1563	1544	1643
12	-	-	-	1577	1569	1716	1583	1572	1693
24	-	-	-	1668	1654	1711	1584	1578	1713

**60  
μm**

Time (hr)	A3A			B2A1			B2A2		
	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)
0	1648	1639	1705	1622	1611	1699	1610	1599	1709
0.03	1624	1616	1760	1650	1645	1702	1591	1588	1687
1	1568	1552	1681	-	-	-	-	-	-
2	1600	1615	1671	-	-	-	-	-	-
3	1580	1559	1653	-	-	-	-	-	-
4	1582	1594	1663	-	-	-	-	-	-
5	1593	1610	1731	-	-	-	-	-	-
6	1623	1609	1763	1551	1538	1629	1562	1549	1665
12	1589	1572	1664	1574	1569	1675	1565	1561	1672
24	1642	1640	1688	1571	1564	1625	1567	1562	1711

**70  
μm**

Time (hr)	A3A			B2A1			B2A2		
	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)
0	1601	1608	1709	1643	1643	1707	1581	1578	1694
0.03	1634	1640	1725	1618	1609	1669	1591	1573	1664
1	1589	1595	1678	-	-	-	-	-	-
2	1604	1597	1645	-	-	-	-	-	-
3	1555	1541	1674	-	-	-	-	-	-
4	1592	1580	1660	-	-	-	-	-	-
5	1558	1579	1674	-	-	-	-	-	-
6	1538	1550	1657	1618	1611	1685	1578	1579	1651
12	1611	1631	1686	1647	1623	1687	1614	1607	1670
24	1598	1598	1668	1573	1558	1642	1583	1567	1683

**Data for each of the thicknesses from the elastase studies.**

**40**

**μm**

Time (hr)	A3B			B1B			B2B		
	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)
0	1622	1600	1666	1658	1656	1816	1614	1575	1666
0.03	1627	1642	1692	1669	1647	1836	1642	1632	1751
1	1580	1627	1665	1594	1584	1754	-	-	-
2	1573	1609	1690	1607	1609	1763	-	-	-
3	1595	1599	1617	1566	1550	1735	-	-	-
4	1608	1594	1660	1654	1639	1769	-	-	-
5	1630	1600	1686	1570	1546	1709	-	-	-
6	1676	1640	1702	1601	1595	1702	1569	1541	1596
7	-	-	-	-	-	-	1556	1542	1594
8	-	-	-	-	-	-	1595	1592	1665
9	-	-	-	-	-	-	1603	1588	1647
10	-	-	-	-	-	-	1591	1572	1723
11	-	-	-	-	-	-	1570	1551	1656
12	-	-	-	1625	1592	1739	1605	1586	1688
16	1613	1605	1629	-	-	-	-	-	-
18	1637	1635	1686	-	-	-	-	-	-
21	1574	1572	1659	1615	1601	1763	1584	1582	1674
24	1605	1645	1689	1564	1554	1672	1553	1536	1580

**50**

**μm**

Time (hr)	A3B			B1B			B2B		
	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)
0	1620	1622	1686	1602	1609	1759	1630	1621	1706
0.03	1637	1628	1677	1660	1648	1790	1591	1551	1677
1	1604	1617	1740	1578	1576	1707	-	-	-
2	1607	1629	1678	1549	1556	1676	-	-	-
3	1585	1559	1739	1590	1551	1706	-	-	-
4	1593	1614	1638	1611	1611	1682	-	-	-
5	1612	1578	1623	1636	1607	1769	-	-	-
6	1630	1618	1715	1592	1547	1703	1599	1569	1654
7	-	-	-	-	-	-	1579	1561	1650
8	-	-	-	-	-	-	1583	1553	1653
9	-	-	-	-	-	-	1611	1591	1669
10	-	-	-	-	-	-	1581	1556	1680
11	-	-	-	-	-	-	1572	1557	1638
12	-	-	-	1631	1622	1743	1608	1603	1695
16	1597	1586	1662	-	-	-	-	-	-
18	1655	1640	1698	-	-	-	-	-	-
21	1567	1565	1639	1628	1638	1727	1595	1587	1673
24	1604	1600	1649	1601	1579	1735	1561	1552	1655

**60  
μm**

Time (hr)	A3B			B1B			B2B		
	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)
0	1590	1585	1652	1629	1620	1748	1620	1621	1698
0.03	1583	1632	1723	1648	1659	1791	1657	1642	1720
1	1602	1590	1699	1584	1582	1731	-	-	-
2	1613	1627	1651	1619	1621	1768	-	-	-
3	1615	1603	1668	1573	1546	1705	-	-	-
4	1565	1571	1663	1586	1617	1727	-	-	-
5	1598	1594	1631	1588	1569	1740	-	-	-
6	1605	1599	1702	1612	1592	1679	1580	1576	1659
7	-	-	-	-	-	-	1579	1580	1664
8	-	-	-	-	-	-	1576	1581	1635
9	-	-	-	-	-	-	1609	1597	1674
10	-	-	-	-	-	-	1563	1561	1604
11	-	-	-	-	-	-	1598	1582	1668
12	-	-	-	1572	1568	1670	1565	1556	1636
16	1597	1579	1631	-	-	-	-	-	-
18	1584	1637	1632	-	-	-	-	-	-
21	1609	1592	1608	1618	1608	1731	1621	1613	1669
24	1564	1559	1602	1575	1587	1710	1546	1533	1616

**70  
μm**

Time (hr)	A3B			B1B			B2B		
	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)	Tang. (m/s)	Trans. (m/s)	Radial (m/s)
0	1634	1650	1709	1599	1599	1756	1592	1590	1690
0.03	1618	1610	1732	1698	1666	1812	1644	1627	1693
1	1610	1607	1694	1601	1596	1718	-	-	-
2	1634	1621	1678	1619	1603	1733	-	-	-
3	1615	1609	1684	1618	1583	1733	-	-	-
4	1604	1602	1672	1605	1628	1723	-	-	-
5	1601	1583	1646	1605	1599	1702	-	-	-
6	1603	1597	1666	1595	1581	1717	1574	1573	1664
7	-	-	-	-	-	-	1572	1569	1671
8	-	-	-	-	-	-	1578	1562	1652
9	-	-	-	-	-	-	1594	1583	1650
10	-	-	-	-	-	-	1552	1544	1627
11	-	-	-	-	-	-	1562	1560	1621
12	-	-	-	1582	1557	1705	1597	1589	1634
16	1623	1613	1675	-	-	-	-	-	-
18	1591	1571	1652	-	-	-	-	-	-
21	1594	1582	1658	1613	1605	1706	1605	1598	1665
24	1596	1608	1646	1619	1610	1712	1587	1579	1606

APPENDIX F

<HI> DATA FROM THE SLAM

The following tables contain the time-course <HI> values for each sample.

**Data for each of the thicknesses from the typsin studies.**

**40  $\mu\text{m}$**

Time (hr)	A3A			B2A1			B2A2		
	Tang.	Trans.	Radial	Tang.	Trans.	Radial	Tang.	Trans.	Radial
0	0.0117	0.0223	0.0123	0.0148	0.0129	0.0134	0.0179	0.0121	0.0354
0.03	0.0284	0.0148	0.0220	0.0131	0.0168	0.0255	0.0150	0.0078	0.0223
1	0.0192	0.0160	0.0476	-	-	-	-	-	-
2	0.0174	0.0151	0.0228	-	-	-	-	-	-
3	0.0257	0.0208	0.0160	-	-	-	-	-	-
4	0.0129	0.0198	0.0268	-	-	-	-	-	-
5	0.0147	0.0242	0.0221	-	-	-	-	-	-
6	0.0100	0.0153	0.0264	0.0166	0.0149	0.0319	0.0144	0.0136	0.0243
12	0.0147	0.0093	0.0257	0.0116	0.0174	0.0242	0.0149	0.0100	0.0180
24	0.0196	0.0127	0.0093	0.0074	0.0138	0.0137	0.0174	0.0132	0.0180

**50  $\mu\text{m}$**

Time (hr)	A3A			B2A1			B2A2		
	Tang.	Trans.	Radial	Tang.	Trans.	Radial	Tang.	Trans.	Radial
0	0.0130	0.0108	0.0186	0.0128	0.0155	0.0191	0.0120	0.0136	0.0229
0.03	0.0156	0.0155	0.0439	0.0143	0.0103	0.0183	0.0131	0.0102	0.0316
1	0.0209	0.0232	0.0272	-	-	-	-	-	-
2	0.0070	0.0088	0.0193	-	-	-	-	-	-
3	0.0136	0.0154	0.0439	-	-	-	-	-	-
4	0.0199	0.0197	0.0149	-	-	-	-	-	-
5	0.0198	0.0204	0.0188	-	-	-	-	-	-
6	0.0145	0.0122	0.0162	0.0197	0.0160	0.0189	0.0157	0.0110	0.0241
12	-	-	-	0.0097	0.0171	0.0230	0.0118	0.0070	0.0305
24	-	-	-	0.0109	0.0125	0.0100	0.0164	0.0174	0.0313

60  $\mu\text{m}$

Time (hr)	A3A			B2A1			B2A2		
	Tang.	Trans.	Radial	Tang.	Trans.	Radial	Tang.	Trans.	Radial
0	0.0185	0.0119	0.0232	0.0125	0.0180	0.0157	0.0131	0.0092	0.0236
0.03	0.0212	0.0153	0.0342	0.0156	0.0142	0.0234	0.0183	0.0287	0.0167
1	0.0142	0.0220	0.0251	-	-	-	-	-	-
2	0.0263	0.0220	0.0322	-	-	-	-	-	-
3	0.0119	0.0152	0.0152	-	-	-	-	-	-
4	0.0119	0.0168	0.0171	-	-	-	-	-	-
5	0.0176	0.0332	0.0368	-	-	-	-	-	-
6	0.0174	0.0190	0.0212	0.0117	0.0146	0.0188	0.0087	0.0127	0.0259
12	0.0104	0.0185	0.0143	0.0148	0.0132	0.0216	0.0161	0.0157	0.0217
24	0.0171	0.0104	0.0251	0.0109	0.0060	0.0259	0.0106	0.0058	0.0307

70  $\mu\text{m}$

Time (hr)	A3A			B2A1			B2A2		
	Tang.	Trans.	Radial	Tang.	Trans.	Radial	Tang.	Trans.	Radial
0	0.0131	0.0114	0.0353	0.0094	0.0124	0.0286	0.0150	0.0132	0.0099
0.03	0.0293	0.0142	0.0201	0.0098	0.0137	0.0221	0.0265	0.0195	0.0274
1	0.0183	0.0304	0.0419	-	-	-	-	-	-
2	0.0176	0.0136	0.0119	-	-	-	-	-	-
3	0.0176	0.0134	0.0183	-	-	-	-	-	-
4	0.0223	0.0219	0.0374	-	-	-	-	-	-
5	0.0166	0.0247	0.0175	-	-	-	-	-	-
6	0.0239	0.0225	0.0344	0.0157	0.0213	0.0243	0.0174	0.0124	0.0281
12	0.0155	0.0132	0.0194	0.0219	0.0189	0.0278	0.0125	0.0187	0.0325
24	0.0181	0.0091	0.0179	0.0128	0.0082	0.0150	0.0198	0.0152	0.0290

Data for each of the thicknesses from the elastase studies.

40  $\mu\text{m}$

Time (hr)	A3B			B1B			B2B		
	Tang.	Trans.	Radial	Tang.	Trans.	Radial	Tang.	Trans.	Radial
0	0.0144	0.0171	0.0174	0.0103	0.0167	0.0276	0.0129	0.0078	0.0292
0.03	0.0081	0.0090	0.0161	0.0180	0.0166	0.0199	0.0113	0.0144	0.0210
1	0.0082	0.0159	0.0154	0.0104	0.0113	0.0373	-	-	-
2	0.0105	0.0341	0.0146	0.0126	0.0103	0.0217	-	-	-
3	0.0277	0.0098	0.0147	0.0189	0.0143	0.0186	-	-	-
4	0.0099	0.0136	0.0142	0.0170	0.0278	0.0288	-	-	-
5	0.0169	0.0100	0.0195	0.0190	0.0142	0.0184	-	-	-
6	0.0145	0.0153	0.0080	0.0147	0.0131	0.0330	0.0160	0.0140	0.0202
7	-	-	-	-	-	-	0.0118	0.0158	0.0138
8	-	-	-	-	-	-	0.0158	0.0131	0.0229
9	-	-	-	-	-	-	0.0204	0.0175	0.0194
10	-	-	-	-	-	-	0.0141	0.0130	0.0238
11	-	-	-	-	-	-	0.0207	0.0235	0.0204
12	-	-	-	0.0179	0.0218	0.0263	0.0096	0.0102	0.0189
16	0.0078	0.0202	0.0159	-	-	-	-	-	-
18	0.0121	0.0114	0.0134	-	-	-	-	-	-
21	0.0140	0.0169	0.0157	0.0203	0.0125	0.0253	0.0124	0.0072	0.0160
24	0.0196	0.0152	0.0176	0.0164	0.0183	0.0377	0.0120	0.0127	0.0177

50  $\mu\text{m}$

Time (hr)	A3B			B1B			B2B		
	Tang.	Trans.	Radial	Tang.	Trans.	Radial	Tang.	Trans.	Radial
0	0.0098	0.0151	0.0291	0.0137	0.0136	0.0193	0.0077	0.0103	0.0351
0.03	0.0434	0.0127	0.0130	0.0151	0.0221	0.0243	0.0109	0.0211	0.0250
1	0.0124	0.0128	0.0161	0.0304	0.0254	0.0255	-	-	-
2	0.0132	0.0139	0.0275	0.0141	0.0294	0.0303	-	-	-
3	0.0091	0.0144	0.0139	0.0139	0.0202	0.0194	-	-	-
4	0.0166	0.0197	0.0343	0.0226	0.0165	0.0235	-	-	-
5	0.0131	0.0114	0.0110	0.0132	0.0113	0.0159	-	-	-
6	0.0208	0.0127	0.0373	0.0159	0.0132	0.0413	0.0112	0.0079	0.0273
7	-	-	-	-	-	-	0.0233	0.0151	0.0468
8	-	-	-	-	-	-	0.0214	0.0248	0.0208
9	-	-	-	-	-	-	0.0193	0.0217	0.0382
10	-	-	-	-	-	-	0.0157	0.0130	0.0285
11	-	-	-	-	-	-	0.0123	0.0137	0.0360
12	-	-	-	0.0332	0.0340	0.0270	0.0168	0.0146	0.0225
16	0.0097	0.0122	0.0145	-	-	-	-	-	-
18	0.0167	0.0216	0.0335	-	-	-	-	-	-
21	0.0129	0.0089	0.0196	0.0279	0.0142	0.0437	0.0094	0.0119	0.0267
24	0.0135	0.0261	0.0387	0.0174	0.0175	0.0395	0.0141	0.0164	0.0301



60  $\mu\text{m}$

Time (hr)	A3B			B1B			B2B		
	Tang.	Trans.	Radial	Tang.	Trans.	Radial	Tang.	Trans.	Radial
0	0.0116	0.0075	0.0137	0.0116	0.0146	0.0224	0.0198	0.0180	0.0248
0.03	0.0130	0.0191	0.0146	0.0198	0.0217	0.0280	0.0116	0.0172	0.0339
1	0.0161	0.0122	0.0172	0.0123	0.0218	0.0260	-	-	-
2	0.0129	0.0126	0.0267	0.0215	0.0239	0.0238	-	-	-
3	0.0120	0.0116	0.0195	0.0231	0.0176	0.0221	-	-	-
4	0.0136	0.0278	0.0213	0.0195	0.0288	0.0355	-	-	-
5	0.0179	0.0184	0.0189	0.0207	0.0223	0.0243	-	-	-
6	0.0320	0.0153	0.0256	0.0186	0.0176	0.0273	0.0127	0.0114	0.0252
7	-	-	-	-	-	-	0.0117	0.0180	0.0339
8	-	-	-	-	-	-	0.0112	0.0112	0.0238
9	-	-	-	-	-	-	0.0226	0.0197	0.0259
10	-	-	-	-	-	-	0.0157	0.0135	0.0332
11	-	-	-	-	-	-	0.0119	0.0119	0.0590
12	-	-	-	0.0198	0.0224	0.0314	0.0195	0.0157	0.0177
16	0.0116	0.0106	0.0209	-	-	-	-	-	-
18	0.0262	0.0150	0.0295	-	-	-	-	-	-
21	0.0187	0.0142	0.0240	0.0171	0.0131	0.0422	0.0165	0.0126	0.0248
24	0.0217	0.0131	0.0209	0.0199	0.0356	0.0438	0.0183	0.0190	0.0161

70  $\mu\text{m}$

Time (hr)	A3B			B1B			B2B		
	Tang.	Trans.	Radial	Tang.	Trans.	Radial	Tang.	Trans.	Radial
0	0.0116	0.0210	0.0173	0.0319	0.0236	0.0339	0.0112	0.0120	0.0170
0.03	0.0122	0.0146	0.0140	0.0258	0.0177	0.0810	0.0225	0.0247	0.0289
1	0.0244	0.0149	0.0183	0.0338	0.0402	0.0244	-	-	-
2	0.0146	0.0150	0.0232	0.0201	0.0195	0.0322	-	-	-
3	0.0235	0.0150	0.0248	0.0196	0.0192	0.0256	-	-	-
4	0.0222	0.0116	0.0382	0.0288	0.0343	0.0233	-	-	-
5	0.0157	0.0193	0.0285	0.0201	0.0109	0.0329	-	-	-
6	0.0148	0.0158	0.0239	0.0153	0.0110	0.0238	0.0144	0.0153	0.0161
7	-	-	-	-	-	-	0.0157	0.0088	0.0275
8	-	-	-	-	-	-	0.0129	0.0282	0.0167
9	-	-	-	-	-	-	0.0134	0.0150	0.0356
10	-	-	-	-	-	-	0.0191	0.0161	0.0224
11	-	-	-	-	-	-	0.0133	0.0168	0.0240
12	-	-	-	0.0273	0.0153	0.0335	0.0186	0.0171	0.0317
16	0.0119	0.0107	0.0178	-	-	-	-	-	-
18	0.0150	0.0173	0.0385	-	-	-	-	-	-
21	0.0181	0.0107	0.0235	0.0162	0.0193	0.0442	0.0153	0.0071	0.0251
24	0.0214	0.0165	0.0485	0.0231	0.0274	0.0271	0.0170	0.0160	0.0266

## REFERENCES

- [1] J. E. Olerud, W. D. O'Brien Jr., M. A. Riederer-Henderson, and A. W. Holmes, "Ultrasonic assessment of skin and surgical wound with the scanning laser acoustic microscope," *J. Invest. Derm.*, vol. 88, pp. 615-623, 1987.
- [2] M. A. Riederer-Henderson, J. E. Olerud, W. D. O'Brien Jr., F. K. Forster, D. L. Steiger, D. J. Ketterer, and G. F. Odland, "Biochemical and acoustical parameters of normal canine skin," *IEEE Trans. Biomed. Engr.*, vol. 35, no. 11, pp. 967-972, 1988.
- [3] D. L. Steiger, W. D. O'Brien Jr., J. E. Olerud, M. A. Riederer-Henderson, and G. F. Odland, "Measurement uncertainty assessment of the scanning laser acoustic microscope and application to canine skin and wound," *IEEE Trans. Ultrason., Ferroelec., Freq. Control*, vol. 35, no. 6, pp. 741-748, 1988.
- [4] J. E. Olerud, W. D. O'Brien Jr., M. A. Riederer-Henderson, D. L. Steiger, J. R. Debel, and G. F. Odland, "Correlation of tissue constituents with the acoustic properties of skin and wound," *Ultrasound Med. Biol.*, vol. 16, no. 1, pp. 55-64, 1990.
- [5] M. O'Donnell, J. W. Mimbs, and J. G. Miller, "Relationship between collagen and ultrasonic backscatter in myocardial tissue," *J. Acoust. Soc. Amer.*, vol. 69, no. 2, pp. 580-588, 1981.
- [6] K. W. Wear, M. R. Milunski, S. A. Wickline, J. E. Perez, B. E. Sobel, and J. G. Miller, "Differentiation between acutely ischemic myocardium and zones of completed infarction in dogs on the basis of frequency-dependent backscatter," *J. Acoust. Soc. Amer.*, vol. 85, no. 6, pp. 2634-2641, 1989.
- [7] K. B. Sagar, D. L. Steiger, W. D. O'Brien Jr., L. R. Pelc, T. L. Rhyne, L. S. Wann, R. A. Komorowski, and D. C. Warltier, "Quantitative ultrasonic assessment of normal and ischaemic myocardium with an acoustic microscope: relationship to integrated backscatter," *Card. Rsrch.*, vol. 24, no. 6, pp. 447-455, 1990.
- [8] R. J. Minns, F. S. Steven, and K. Hardinge, "Osteoarthritic articular cartilage lesions of the femoral head observed in scanning electron microscopy," *J. Path.*, vol. 122, pp. 63-70, 1977.
- [9] J. H. Bland and S. M. Cooper, "Osteoarthritis: a review of the cell biology involved and evidence for reversibility," *Sem. Arthritis Rheum.*, vol. 14, pp. 106-133, 1984.

- [10] M. L. Richardson, B. Selby, M. A. Montana, and L. A. Mack, "Ultrasonography of the knee," *Ultrason. Musc. Sys.*, vol. 26, pp. 63-75, 1988.
- [11] W. J. McCune, D. K. Dedrick, A. M. Aisen, and A. MacGuire, "Sonographic evaluation of osteoarthritic femoral condylar cartilage," *Clin. Orth. Rel. Rsrch.*, vol. 254, pp. 230-235, 1990.
- [12] N. T. Sanghvi, A. M. Snoddy, S. L. Myers, K. D. Brandt, C. R. Reilly, and T. D. Franklin Jr., "Characterization of normal and osteoarthritic cartilage using 25 MHz ultrasound" in *IEEE Ultrason. Sym.*, 1990.
- [13] R. S. Adler, D. K. Dedrick, T. J. Laing, E. H. Chiang, C. R. Meyer, P. H. Bland, and J. M. Rubine, "Quantitative assessment of cartilage surface roughness in osteoarthritis using high frequency ultrasound," *Ultrasound Med. Biol.*, vol. 18, no. 1, pp. 51-58, 1992.
- [14] K. A. Harasiewicz, H. K. W. Kim, P. S. Babyn, K. P. H. Pritzker, and F. S. Foster, "Ultrasound backscatter microscopy of articular cartilage in vitro" in *IEEE Ultrason. Sym.*, 1993.
- [15] D. H. Agemura and W. D. O'Brien Jr., "Ultrasonic propagation properties of articular cartilage at 100 MHz," *J. Acoust. Soc. Amer.*, vol. 87, no. 4, pp. 1786-1791, 1990.
- [16] D. A. Senzig, F. K. Forster, and J. E. Olerud, "Ultrasonic attenuation in articular cartilage," *J. Acoust. Soc. Amer.*, vol. 92, no. 2, pp. 676-681, 1992.
- [17] D. W. K. Hukins, R. M. Aspen, and Y. E. Yarker, "Fiber reinforcement and mechanical stability in articular cartilage," *Eng. Med.*, vol. 13, pp. 153-156, 1984.
- [18] D. W. L. Hukins and R. M. Aspden, "Composition and properties of connective Tissues," *Trends Biol. Sci.*, vol. 10, pp. 260-264, 1985.
- [19] A. K. Jeffery, G. W. Blunn, C. W. Archer, and G. Bentley, "Three-dimensional collagen architecture in bovine articular cartilage," *J. Bone Joint Surg.*, vol. 73-B, pp. 795-801, 1991.
- [20] M. B. Schmidt, J. M. Schoonbeck, V. C. Mow, D. R. Eyre, and L. E. Chun, "Effects of enzymatic extraction of proteoglycans on the tensile properties of articular cartilage," *Trans. Orthop. Res. Soc.*, vol. 12, pp. 134, 1987.
- [21] L. E. Chun, T. J. Koob, and D. R. Eyre, "Sequential enzymatic dissection of the proteoglycan complex from articular cartilage," *Trans. Orthop. Res. Soc.*, vol. 11, pp. 96, 1986.

- [22] D. H. Agemura, "Ultrasonic propagation properties of canine myocardium and bovine articular cartilage at 100 MHz," M.S. thesis, University of Illinois, Urbana, IL, 1988.
- [23] K. M. U. Tervola, S. G. Foster, and W. D. O'Brien Jr., "Attenuation coefficient measurement technique at 100 MHz with the scanning laser acoustic microscope," *IEEE Trans. Son. Ultrason.*, vol. SU-32, no. 2, pp. 259-265, 1985.
- [24] K. M. U. Tervola and W. D. O'Brien Jr., "Spatial frequency domain technique: an approach for analyzing the scanning laser acoustic microscope interferogram images," *IEEE Trans. Son. Ultrason.*, vol. SU-32, no. 4, pp. 544-554, 1985.
- [25] D. L. Steiger, "Ultrasonic assessment of skin wounds with the scanning laser acoustic microscope," M.S. thesis, University of Illinois, Urbana, IL, 1986.
- [26] D. D. Nicozisin, "An automated imaging data acquisition and analysis system for the scanning laser acoustic microscope," M.S. thesis, University of Illinois, Urbana, IL, 1989.
- [27] *User Manual for the DT2851 High Resolution Frame Grabber*, Marlborough, MA: Data Translation, Inc., 1988.
- [28] *User Manual for the DT2858 Auxiliary Frame Processor*, Marlborough, MA: Data Translation, Inc., 1988.
- [29] *User Manual for the DT-IRIS Subroutine Library*, Marlborough, MA: Data Translation, Inc., 1988.
- [30] B. W. Kernighan and D. M. Ritchie, *The C Programming Language*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1988.
- [31] S. A. Goss and W. D. O'Brien Jr., "Direct ultrasonic velocity measurements of mammalian collagen threads," *J. Acoust. Soc. Amer.*, vol. 65, no. 2, pp. 507-511, 1979.
- [32] S. Foster, "An image digitizing system for a scanning laser acoustic microscope," M.S. thesis, University of Illinois, Urbana, IL, 1981.
- [33] P. M. Embree, S. G. Forster, G. Bright, and W. D. O'Brien Jr., "Ultrasonic velocity spatial distribution analysis of biological materials with the scanning laser acoustic microscope" in *Acoustical Imaging*, M. Kaveh, R.K. Mueller, and J.F. Greenleaf, Eds. New York: Plenum Press, 1984, pp. 203-216.
- [34] P. M. Embree, K. M. U. Tervola, S. G. Foster, and W. D. O'Brien Jr., "Spatial distribution of the speed of sound in biological materials with the scanning laser acoustic microscope," *IEEE Trans. Son. Ultrason.*, vol. SU-32, no. 2, pp. 341-350, 1985.

- [35] R. D. Marangoni, A. A. Glaser, J. S. Must, G. S. Brody, T. G. Beckwith, G. R. Walker, and W. L. White, "Effect of storage and handling techniques on skin tissue properties," *Ann. NY Acad. Sci.*, vol. 136, pp. 429-454, 1966.
- [36] J. V. Geleskie and K. K. Shung, "Further studies on acoustic impedance of major bovineblood vessel walls," *J. Acoust. Soc. Amer.*, vol. 71, pp. 467-470, 1982.