1999

High frequency ultrasonic characterization of carrot cell texture

Christopher Vick

SIMG-503

Senior Research

High Frequency Ultrasonic Characterization of Carrot Cell Texture

Final Report

Christopher W. Vick
Center for Imaging Science
Rochester Institute of Technology
May 1999

Table of Contents

High Frequency Ultrasonic Characterization of Carrot Cell Texture

Christopher W. Vick

Table of Contents

Title Page

High Frequency Ultrasonic Characterization of Carrot Cell Texture

Christopher W. Vick

Abstract

Ultrasound offers a versatile imaging modality. It has a long history of medical use, but system and signal processing limitations have limited its full range of capabilities. By examining other uses of ultrasound, such as materials testing, the effectiveness of medical ultrasound tissue identification could likely be improved.

To a limited degree, low frequency ultrasound has been used examine the texture of carrots tissue. These studies focused on the velocity and attenuation of low frequency ultrasound through a carrot sample; these results yielded a degree of ambiguity when attempting to identify a carrot texture. In our research, we attempted to characterize the texture of a carrot, based upon its frequency response.

Research showed that there are a number of sources of variation when imaging a carrot. First, the response of the transducer itself can vary. Replications of the same sample also yield varied results. This is a result of problems such as transducer misalignment, or coupling. Imaging different segments of a carrot can be a significant source of error. By far, the largest source of variation came from the imaging samples from different carrots. This variation is partly due to different transducer coupling, and possible non-uniform cooking. Also, to an unknown extent, slight differences in cell biology among different carrots also contributes to this high variation.

A response look up table was compiled for carrots cooked .5 to 16 minutes. When an attempt was made to identify the cooking times of 10 unknown samples, 4 of them were correctly identified. This inaccuracy in matching is due to high sources of variation noted above. A more customized matching program could yield better results. By combining frequency response studies with velocity and attenuation studies, a more accurate characterization model could likely be devised.

Table of Contents

This report is accepted in partial fulfillment of the
requirements of the course SIMG-503 Senior Research.

Title: High Frequency Ultrasonic Characterization of Carrot Cell Texture
Author: Christopher W. Vick
Project Advisor: Navalgund Rao
SIMG 503 Instructor: Joseph P. Hornak

Table of Contents

High Frequency Ultrasonic Characterization of Carrot Cell Texture

Christopher W. Vick

Acknowledgement: The following people provided invaluable assistance on this project:

Dr. Navalgund Rao	My research advisor; He inspired the research, supplying theory, groundwork, and support.
Maria Helguera	Thank you for your endless patience and insight into the ultrasound data acquisition process.
Brad Miller	Graciously and skillfully prepared the 'unknown' carrot samples.

Table of Contents

High Frequency Ultrasonic Characterization of Carrot Cell Texture

Christopher W. Vick

Introduction

Ultrasound is a fast, nondestructive, noninvasive, and relatively inexpensive imaging platform. Ultrasound has many uses in medical diagnosis. Such applications often consist of a skilled technician interpreting a ultrasound image, based on gray-level and texture. Unfortunately, such a process cannot be used to effectiv evaluate tissue structure. The human visual system is not capable of discriminating many higher-order textures. This has spurned the development of computer texture analysis methods; these methods cannot be effective until we learn more about the intricacies of ultrasound propagation through tissue.

It has long been known that the velocity and attenuation of ultrasound are characteristics of the medium t travel through. These parameters can be related to the physical properties of the material, such as density, elasticity, composition, and cell microstructure. In 1991, Self et al. (1), proposed the use of ultrasound for texture evaluation of plant tissues. Low frequency ultrasound has been used extensively to characterize the ripeness o fruits. Limited research has been conducted to measure the cell texture of cooked carrots using low ultrasound. And unfortunately, the experiments conducted yielded ambiguous results. It is presumed that frequency ultrasound carries complex microstructural information about a material; detailed examination of cellular structure using high frequency ultrasound (>1MHz) has not been attempted.

The hypothesis of this research is that high frequency ultrasound (5MHz) can be used to characterize a identify the cellular structure of carrots. When properly accounted for, high frequency ultrasound should yiel useful cellular texture information unavailable when low frequency ultrasound is used.

This research focuses on the high frequency evaluation of carrot cell texture. We focus on measuring frequency response curves of a variety of carrot samples, inducing texture changes through cooking. By statistically examining these functions, parameters for classifying the cell structure of the cooked carro derived. Using these parameters, a program can be devised that can identify how long a carrot has been cooked.

A number of long term goals will be addressed by this research. First, and foremost, this research will address whether plant texture characterization using high frequency ultrasound is realistic, or even possible. Furthermor this research will examine the biological devices for the attenuation and propagation of high frequency ultra through the carrot cell tissue. Eventually, this could allow for more comprehensive computer texture an models. We will discuss the possibility of applying comparable imaging techniques toward human characterization. This could in turn lend toward more accurate medical diagnosis, such as distinguishing health liver tissue from diseased liver tissue. Using noninvasive ultrasound for this process will spare a hospital patient the unnecessary pain of a biopsy, and the unnecessary fees associated with other imaging modalities, such as MR or X-Ray.

Background

The use of ultrasound, mechanical waves with a frequency above 20KHz, is becoming increasingl widespread. The ultrasound technique utilizes a transducer which is capable of generating and receiving h frequency ultrasonic vibrations. In application, these very short wavelength sound waves are reflected off very small surfaces inside a material before they are collected. By analyzing these complex reflections, the inside material can be imaged. Ultrasound offers a fast, nondestructive, noninvasive, and relatively inexpensive i platform. These properties have spawned the application of ultrasound to a variety of materials.

Ultrasound has long been used in medicine for diagnosis. A gray-value medical ultrasonic image can be made of a human liver section, for instance. A skilled ultrasound technician uses gray-level as well as texture information to interpret the image, such as identifying a lesion. The human visual system is very capal discriminating different textures from each other, but is hardly perfect (2). The use of computer texture analysis is being thoroughly researched. An artificial neural network has been used to develop an adaptive texture fe extraction method; currently it only exhibits a noticeable importance in a limited number of dataset situations (2). Without a doubt, numerous improvements need to be made to this system. This system needs to more effect account for imaging artifacts, frequency-dependent attenuation, backscatter, and other inherent imaging effects Such improvements can likely be derived from a close examination of other ultrasound techniques.

Ultrasound has been shown to be especially useful for the nondestructive testing of many other materials. The food industry has been using ultrasonic techniques for a number of years. It has been used for processe emulsification, cleaning and animal backfat thickness estimation (3). Ultrasound has been used to examine the structure of homogeneous materials, such as metals (4), and inhomogeneous materials , such as wood (5). Ultrasonics has been used to measure ripeness in bananas (1), melons (6) and avocados (7). The use of ultrasound for texture measurements of plant tissues has been proposed (8), but not thoroughly examined. This is the key aim of the proposed research.

Let us consider the idea of ultrasonic microstructure characterization. This can be done using ultra transmission measurements of wave speed and attenuation through a material. It has been shown that velc attenuation are characteristic of the physical properties of the material, such as density, composition, elasticity, and other cellular structure properties (8). Wave speed measurements are relatively insensitive to detai microstructure; attenuation yields a much better examination of microstructural detail(4). Unfortunately, separating attenuation effects due to diffraction, absorption, refraction, and scattering is difficult, if impossible. A lot of theoretical work is conducted in this respect.

A study was conducted to access the early possibility of using ultrasound to assess the quality of frui vegetables (9). It was concluded that the high attenuation in plant samples prevented an effective use of h frequency ultrasound. A subsequent study suggested this high attenuation was due to scattering from pl intercellular air spaces (10). Low frequency (<100kHz) ultrasound has been shown to alleviate high attenu problems (7), because scattering at long wavelengths is weakest. However, high wavelengths likely carry information about the microstructure of a material (4).

The texture of carrots has been under much study. A large number of biological studies have been cond toward understanding the tissue factors that affect the eating texture of fresh and cooked vegetables (11). Ahmed et al. (12), noted that when a carrot is subjected to thermal processing, its texture undergoes a number of physica and chemical changes. These changes have been observed and measured in a number of different research Recently, an attempt at the texture evaluation of carrots has been made using low frequency ultrasound. A examined the changes in low frequency (37kHz) velocity and attenuation among cooked carrots (13). In the aforementioned study, transmission changes were correlated with textural changes measured using an e microscope. The purpose of this study was to detect a difference between fresh and cooked carrots using l frequency ultrasound. Very little signal analysis was conducted in this study; the results are often vague, and overly speculative. In addition, there was a large range of error, and are many factors that this study didn't account for. Carrots are generally homogenous along each axis of symmetry. This allows a number of assumptions to b made about the carrot tissue; this, in effect, allows simplified theoretical considerations to be made abou propagation of sound waves through the cell structure. The above study didn't account for this, nor for fre dependent attenuation.

This research expands upon the experiment conducted by Nielsen and Martens (13), making a few careful changes to the methodology they used. Most importantly, higher frequency ultrasound was used (1MHz), as opposed to the 37kHz used by Nielsen and Martens. This substantially increases the attenuation effects, but also likely yields additional microstructure information previously unavailable. In our research, the focus is not the velocity and attenuation of the signal. The frequency analysis of the ultrasound signal is what is important here. With unique responses, carrots of varied cell texture can be differentiated. This research could likely also apply to other homogeneous vegetables (such as potatoes) also. Most importantly, the proposed research will contribute to a more comprehensive understanding of the ultrasonic imaging of cell structure. These signal analysis methods can likely be used to improve the performance of current texture analysis methods; this could allow for more accurate diagnosis and tissue characterization in humans.

Theory

When digitizing signals, one needs to be careful to avoid introducing aliasing, artifacts, or other undesi features into the recorded function (14). The Whittaker-Shannon sampling theorem states that a sinusoidal function must be sampled at a rate greater than the Nyquist frequency to avoid aliasing. The Nyquist Freque defined as:

$\varepsilon_{nyq} = 1/(2\Delta x)$, where Δx = sampling interval

In this project, as will be explained later, a pair of 5MHz transducers will be used. A chirp function w generated with the transmitting transducer, that scans from about 2 MHz to about 8 MHz. Thus we expect th highest output frequency to be about 8 MHz. So, based on the above equation:

Given: $\varepsilon_{nyq} = \varepsilon_{max}$ = 8 MHz
Sampling interval = $\Delta x = 1/(2\varepsilon_{nyq}) = 1/(2\ (8\text{ MHz}))$
Thus, Δx = 1/(16 MHz cycles/sec)
Δx = 62.5 nanoseconds

In order to avoid aliasing, the digitizer should be set above this interval. Thus, in this project, the digitizer was set to sample all f(x) at 50 ns.

The Fourier Transform is an important tool in analyzing the signals in this project. In essence, performi Fourier Transform on a function decomposes the function into all the separate frequencies that comprise it.

A flexible version of the Discrete Fourier Transform will be used, called the Fast Fourier Transform (FF There is a pre-existing FFT function in IDL that will prove useful for this project.

Given a function, f(x), the FFT of the function equals F(u). Where:

$$F(u) = FFT[f(x)] = \frac{1}{N}\sum_{x=0}^{N-1} f(x)e^{[-j\ 2\ (Pi)\ ux/N]}$$

As shown, N= the number of elements in the array.

The result is an array with N elements. Element 0 contains the zero frequency component, which is equal to 1/(NT), where T is the sampling interval. Element 1 contains the 2/(NT) frequency component, etc. All the way up to element N/2, which contains the components from the Nyquist frequency, the highest frequency that can sampled.

When imaging a carrot sample, if we characterize the system as linear & shift invariant, it can be modeled as a simple convolution process:

f_transducer(t) = the transducer function over time

f_carrot(t) = the carrot function over time
Sys(t) = the system function over time

By acknowledging a convolution process:

f_tr(t)*f_ca(t) = Sys(t)

By taking the FT of both sides, we arrive at:

FT[f_tr(t)*f_ca(t)] = FT[Sys(t)]

By using the Transform of a Convolution property of Fourier Transforms (Gaskill, 1978, p.196), we arrive at:

F_tr(u) F_ca(u) = Sys (u) (1)

where,
F_tr(u) = FT of f_tr(t)
F_ca(u)= FT of f_ca(t)
Sys(u) = FT of Sys(t)

Overall, it would be useful to remove the response of the transducers, so that only the carrot response remained. This can be done by taking Equation (1) into log space:

F_tr(u) F_ca(u) = Sys (u)
Log [F_tr(u) F_ca(u)] =Log [Sys (u)] using log mult. rule yields:
Log[F_tr(u)] – Log[F_ca(u)] = Log [Sys(u)] solve for F_ca(u)
Log[F_ca(u)] = Log[Sys(u)] + Log[F_tr(u)]
So, F_ca(u) = 10^(Log[Sys(u)] + Log[F_tr(u)])

This research will deal mainly with the response of the system as a whole, but it was the above technique that was used to allow the computer program to isolate the carrot frequency response.

Methods

Experimental Setup

The experimental setup consisted of a clear vertical jar. For measurements, this jar is filled with filtered water. Mounted at opposite ends of the jar were a pair of flat plane 5MHz ultrasound transducers. The transmitti transducer was a Panametrics: 5.0/.25, 113542; the receiving transducer, Panametrics V326, 5.0/.375, 9392 Diagram #2 shows the rest of the laboratory setup. For imaging, a sample is secured between the transducers. Fc this research, the signal generator (Polynomial Waveform Synthesizer, Model 2020) was programmed to produce a chirp function, using the following function: For 5u SIN(INT(1.5M+1.3M/1u*t)) FOR 5 m 0. This creates a chirp function scan a frequency range of about 1MHz to 9 MHz, centered at the transducers peak sensitivity, 5MHz. This signal is first sent through a (EIN Model 240L RF) power amplifier, then through the transducer. Next, the signal propagates through the longitudinal axis of the sample in the jar setup, and is capture by the receiving transducer. It is then amplified by a receiving amplifier (RITEC Broadband Reciever BR-64(where the system gain (in dB) can be manually adjusted. Before getting stored to disk, the output signal is digitized to 9 bits using a Data 6500 Digitizer. For all the measurements taken below, the output signals we sampled for 256 points, in 50 ns intervals (as explained above). In addition, the receiver gain for each measurement was recorded, for use in mathematical calculations as stated earlier.

Carrot Sample Preparation

Carrot samples were prepared in a method similar to that used by Nielsen and Martens . (13) The carrots used were ordinary supermarket Dole Carrots, bought in 32oz. bags. After purchase, they were stored i refrigerator. Before preparation, the carrots were cleaned, and sorted according to size. Carrots 19-38 mm diameter, without noticeable breaks, cuts, or bruises, were used. The tops and bottoms of the carrots were cut of and discarded. Using a Farberware Ultrapro Apple Corer (#81780), a 1.4cm diameter region of the carrot core was stamped out. This core was carefully cut crosswise, into individual small cylinder samples, 1.1cm in height. Thu the final sample dimensions were 1.1 cm in height, and 1.4cm in diameter (see Diagram #3.) Depending on segment of the carrot a sample is cut from, the xylem core diameter can vary greatly. As necessary, the size o xylem cores were measured and recorded. The samples were cut and placed in a plastic bag before cooking.

For the cooking experiments, the carrot cylinders were boiled at 100 degrees Celsius in filtered wate appropriate time of between 0 to 16 minutes, in 30 second intervals. This cooking serves to induce textural changes in the structure of the carrot. Immediately after the carrots were done cooking, they were cooled off filtered water at room temperature. Shortly thereafter, the samples were imaged in the aforementioned ultraso setup.

Trials / Data Acquisition

There were six main parts conducted for this research:

Part I: Measuring the response of the transducers

For the first part, it was important to measure the response curve of the transducers. This was done by filling the jar setup with water, and imaging in the absence of a carrot sample. The water offers no acoustic impediance, thu the ultrasound signal can propagate through the system, unaffected. When the output signal is collected, wha results is effectively the response of the transducers. Another factor to consider was the variability of this response curve. In order to test this, the transducer response was measured twenty times, on three different occasions. In each of these trials, the transducer was aligned, the imaging was done, then the transducer was misaligned, and the process was repeated. It is known when the transducers are aligned when a maximum amplitude signal is shown on the digitizer. This means that the transducer plates are parallel, and lined up.

Part II: Testing variation among repeated same sample readings

For the next part, it was desired to examine the amount of variation that has been shown to be inhere repeatedly imaging the same sample numerous times. For this trial, 18 random segment carrot cylinder sample were prepared, and cooked for a 1 minute interval. These samples were then imaged a total of five times each.

Part III: Testing variation among different segments of the same carrot

Along the length of a carrot, the xylem core diameter changes significantly. At the bottom of a carrot, it is only few millimeter in diameter, while near the top, the xylem core can be over a centimeter in diameter. The effect o the diameter of the xylem core on a carrot sample response curve was measured. To do this, 9 consecutive segment samples were cut from five different carrots from the same harvest. The xylem core diameter for th samples ranged from .1cm to .4 cm. For this part, all these samples were cooked for 1 minute.

Part IV: Testing variation among different carrots of equal cooking time

Next, the response curve variance among different carrots was measured. 16 same segments samples were cut from four different carrots of the same harvest. These samples were all cooked for a 1 minute interval, subsequently imaged.

Part V: Imaging samples of various cooking times

For this part, 64 random samples were cut from 8 different carrots of the same harvest. These samples wer separated into two batches of 32 samples. For each batch, one sample was cooked for each of the thirty-two : second intervals from 30 seconds to 16 minutes. Using the above setup, all these samples were imaged.

Part VI: Unknown sample trial

For the last section of the experiment, 10 random sample segments were cut from 5 random carrots of the sa harvest. These samples were cooked by an assistant, for 'secret' lengths of time. These 10 carrots were label a-j. Two were cooked for either 1,2,3,4, or 5 minutes. Another two were cooked for either 6,7,8,9, or : minutes. Two more were cooked for 11,12,13,14,15 or 16 minutes. The next two were cooked for a random : second interval under 16 minutes. The last two were cooked for any randomly picked time under 16 minutes. All of the a-j samples were mixed up, with their actual cooking times sealed in envelopes. These samples were imaged, in the same process as explained above. The data was then stored for later analysis.

Data Analysis

For the most part, the data obtained was evaluated using my original IDL program, entitled C.U.A.P. The sourc code for this Carrot Ultrasound Analysis Program can be found in Appendix A. Additional analysis was done using Microsoft Excel.

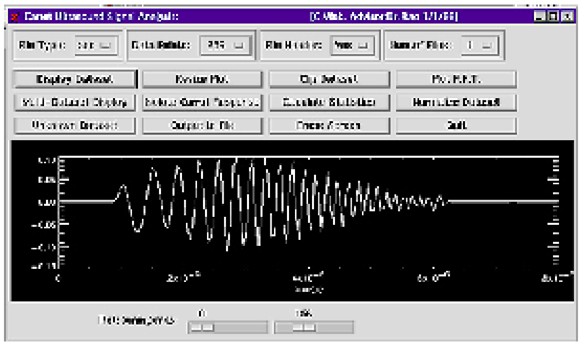

Diagram #1 : graphical user interface of the CUAP.

By pressing the ‘Display Dataset’ Button, a pickfile box emerges, in which the user can specify a file to display. By choosing ‘Multi-Dataset Display’, and setting "Num. Of Files" to the appropriate value, up to 16 files c displayed at once. A user can specify the number of points in his file, the file type, and if it has a header. B pressing the FFT button, the Fast Fourier Transform of the input function is computed by the equations shov Theory, and displayed. By graphing numerous sample readings at once, ‘Calculate Statistics’ can analyze them. As shown by the buttons above, data can be normalized, resized, clipped, isolated, and erased, as necessary addition, by inputing an unknown function, and pressing the ‘Unknown Dataset button,’ the sample cooking time is predicted by the program.

Results

Part I: Transducer Response

As an initial step of the research, the response of the transducers was characterized. This was done through the aforementioned process. Figure #2 shows the mean magnitude transducer response, averaged over 18 separate measurements. The standard deviation bars across the response curve convey the small variance across the measured response. The average variation was at about 6%.

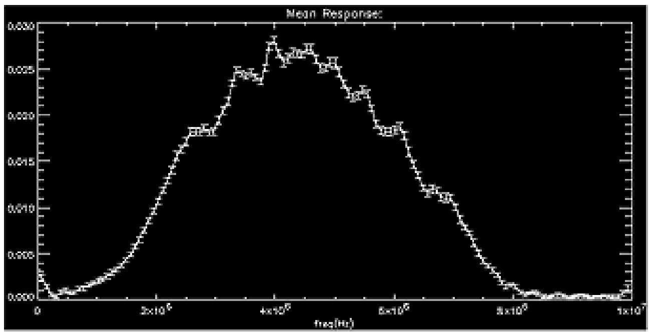

Figure #2: Mean Transducer Response. Bars show plus/minus one standard deviation.

Part II: Same Sample Variation

Next, variation from imaging the same sample repeatedly was measured. This was examined most thoroughly on a carrot cooked for one minute. Figure #3 shows the standard deviation of the mean magnitude system response. For this graph, sample readings were all normalized. This discards overall attenuation information, but still retains relative frequency dependent attenuation data. The average variation here is at 20%. This measurement readily agrees with the 21% replication variation reported by Cheng (1992) and Self et al. (1994). Notice that in the lower half of the frequency range, variation drops below 7%.

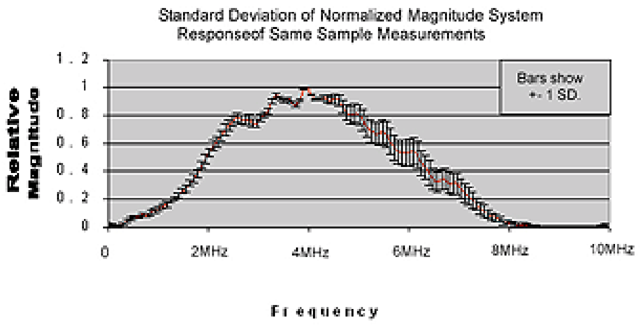

Figure #3:Standard Deviation of Normalized System Response with 1 minute cooked carrots.

Part III: Segment Variation

Unlike previous studies, the response curve along the length of a carrot was also examined. Figure #4 shows a three-dimensional plot of the system response of a carrot, as the xylem core diameter ranges from 0.2 to 0.4 cm. Particularly notice that the magnitude substantially drops.

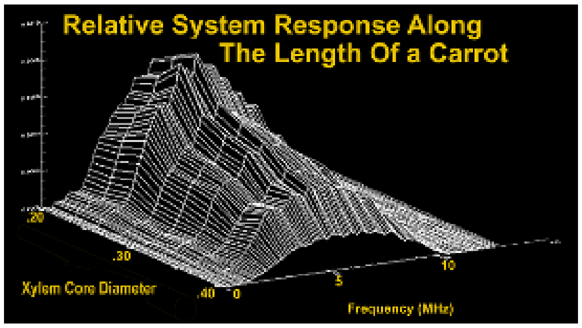

Figure #4: Relative System Response along length of a carrot

As can be seen above, even along the length of the carrot, a characteristically similar response of the carrot is retained. In the above case, the average variation among different carrot segments is 45%. On the other hand, once these functions are normalized, the average variation drops to below 15%.

Part IV: Carrot Variation

Next, a number of same segment samples were examined from different carrots. Figure #5 shows an example of the wide variation seen among carrots cooked for 2 minutes. Variation can be as high as 80%. Yet even after normalization, as shown, the , different carrots variation averages at 36%. The sources of this large amount of variation will be discussed later on.

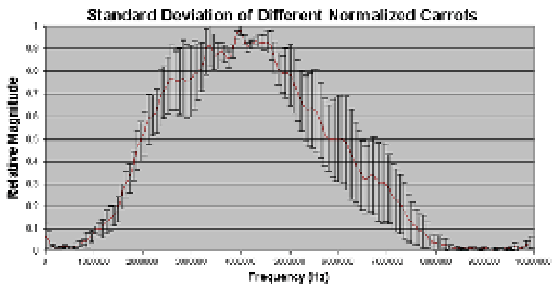

Figure #5: Standard Deviation among same segment.

Part V: Various Cooking Times

Figure #6 shows a compilation of the response curves of 16 samples, cooked from 1 to 16 minutes, in 1 minute intervals. Again, it is not the exact magnitude that is what is important, but rather the relative comparisons of the response of the different carrot textures. The frequency response changes seen in the plot can be explained by structural changes that were invoked through cooking. By normalizing this plot, we essentially arrive at a carrot texture look up table. This is shown in Figure #7. As can be seen, it is difficult to make sense of this plot by eye. But, as the hypothesis stated, through statistics, many of the curves can can be differentiated.

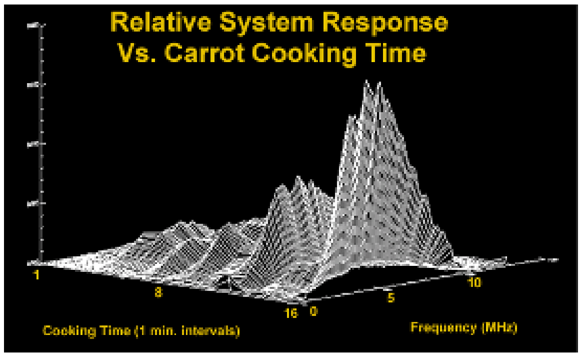

Figure #6: Relative System Response of samples cooked 1-16 min.

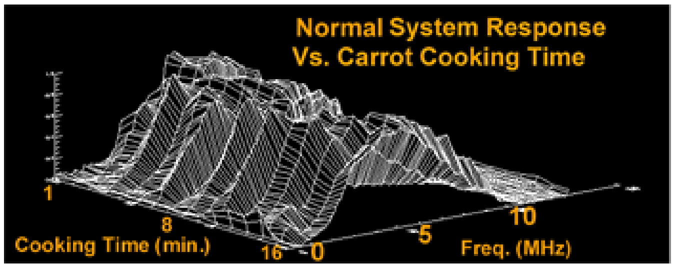

Figure #7: Normalized System Response Vs. Cooking Time

Figure #8 shows a side view of this plot, where the shape of individual curves are more readily noticable.

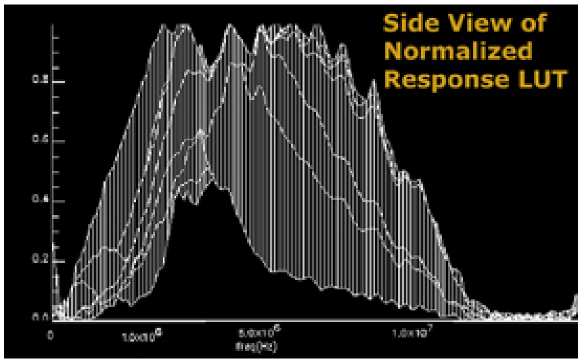

Figure #8: Side View of Normalized LUT

Looking at Figure #8, one can notice the unique curves of many of the carrot textures. Also evident, as exemplified by the upper right section, is the fact that a handful of the response curves have very similar corresponding sections. An important issue is whether the similar response curves are too similar in their entirety.

The purpose of the last experiment, the imaging of unknown samples, was to test the robustness of the above LUT. If indeed different texture response curves are too similar, then using this table to identify an unknown sample would yield error prone, or ambiguous results.

Part VI: Unknown Sample

Using the 'Unknown Dataset' option from the carrot ultrasound analysis program, the cooking times of the 10 unknown carrots were predicted. Below is an overview of the analysis of 5 of the samples:

Unknown Sample	Predicted Cooking Time	Unknown's variance from Predicted LUT	Actual Cooking time	Unknown's variance from Actual LUT	Is the program's prediction correct?
Carrot C	6 min.	22%	4 min.	32%	No
Carrot B	7 min.	11%	7 min., 18 sec.	16%	Close
Carrot D	9 min.	24%	9 min.	24%	Yes
Carrot A	12 min.	21%	12 min.	21%	Yes
Carrot E	16 min.	23%	13 min., 30 sec	32%	No

Figure #9: Unknown Sample Analysis

Overall, the program correctly matched 4 of the 10 unknown sample. Among the samples that weren't correctly matched, there was a standard deviation of 1.8 minutes. The average variance of unknown samples from their LUT response was 28%.

Discussion

As shown in figure #2, there was a 6% average variation of the transducer response. I am uncertain as standard amount of variation that is expected from ultrasound transducers of this quality. When properly accounted for, it seems that variation would be a limited source of error when characterizing carrots.

Ideally, when the same sample is imaged numerous times, the same response would be measured each time. As exemplified in figure #3, there is an average 20% variation between replications. All things being equal, looking at the transducer variation above, one would expect the variation here to be 6%. But in this case, we see an extra 14% variation. This may have been caused by different couplings of the samples between the transducers, transducer misalignment, or variation of pressure applied to the samples. During the measurement process, i assumed that that carrot was in a constant state. Realistically, it is reasonable to suggest that slight changes cou have occurred within the carrot. For instance, the sample imaged near the end of the trial was in the jar substantially longer than the sample imaged near the beginning. This gives the last sample more of an opportunity to absorb the water around it. What this means that the frequency response of a sample could vary to some degı with time. This research data was not analyzed in this respect, so this factor is not accounted for. Needless to say the majority of the variation was likely due to problems with transducer alignment and coupling.

As figure #4 shows, the frequency response of a sample varies along the length of the carrot. Magnitudes c vary by as much as 45% after only a few centimeters of distance. This is due the dense xylem that comprises th core of carrots. As this core increases in diameter, it attenuates more and more ultrasound. With large xyl diameters, a substantial portion of the input signal can be lost. This especially poses a problem when imaging sample near the top of the carrot. As noted, when these responses are normalized, the average variation drops t 15%. As earlier, different transducer coupling is also likely to be a strong factor in this. This 15% value suggests that there is slightly less variation in imaging different segments than in repeatedly imaging the same sample. T may seem somewhat surprising at first, but can be accounted for. Namely, all the samples in this trial remained ir the setup for the same length of time. This means that they all had the opportunity to absorb equal amounts of water, thus their response variation with respect to time is not a factor here. This realization implies that in previous trial, variation with respect to time is on the order of 5%. Overall, results from this trial suggest that xylem size increases, the response magnitude decreases equally across the total frequency response of the sample. It is this acessment that allows for the normalized response similarity among the different segments. Further testing needs to be conducted in order to evaluate whether this hypothesis holds true for xylem diameters larger than 0.4 cm.

As Figure #5 accentuates, there is a substantial amount of variation among the response of different carro Even normalized, this variation averages at 36%. As explained above, problems with transducer coupling alignment contribute to this number. Although there is a degree of biological commonality among carrots, no carrots can be exactly the same. When imaging them, variations in cell biology among different carrots contribute to frequency response variation. This varied biology can be due to a number of factors. Perhaps the carrots exposed to different pesticides. Maybe they were grown in different parts of the world, absorbed different minerals from the soil, or received different amounts of sunlight. Overall, many environmental factors could contribute carrot variation. In the course of this experiment, we attempted to limit these possible biological variations choosing samples from similar sized carrots from the same harvest. As seen, this was still not completely effective in eliminating variation. Another source of variation difficult to control is the cooking. Non-uniform cooking of the samples can also lead to some degree of variation in the texture of the carrot, and thus the variation i propagation characteristics of the sample.

Figure #6 displays the relative changes in frequency magnitude for carrot samples cooked up to 16 minutes For simplicity, only 1 minute cooking intervals are shown here; responses were actually measured in 30 se intervals. By examining how the carrot cell biology changes as a result of cooking, the observed response shifts can be explained. The compactness characterized with raw carrot tissue allows the passage of very little signal at all. In the first 3 minutes of cooking, there is denaturing of the cell membranes and changes in the turgor pressure of the sample. This causes the amount of attenuation to decrease substantially. After about 3 minutes, studies l noted that heat induces the cells into irregular shapes, and the development of intercellular cavities between Because the wavelength of the high frequency ultrasound is so small, these small spaces have a higher propensity to scatter it. This increases the observed attenuation. With additional cooking, these spaces grow in number and size. Eventually, there is a breakdown of the cell membranes. This allows the cytoplasm to flow freely into t cavities. In addition, outside water is increasingly being absorbed into the carrot sample. These contribute to making the carrot more uniformly homogenous, thus decreasing the overall attenuation. This point could correspond to the peak seen in the figure at about 12 minutes. With prolonged cooking, large cracks develop in the carrot cell structure. These can highly serve as scatterers; this could account for the degradation of signal seen near the 14 minute cooking mark. The further destruction of the cell structure, shown at 16 minutes cooking characterized by a substantial increase in the amount of signal passed. A more thorough correlation betwe frequency response and biology can be established by examining the cellular cross-sections of each sample at each cooking time. Due to time constraints, this was not attempted.

For the last part, ten unknown samples were used to do preliminary tests on the response curve look up t shown in Figure #7. Of these 10 samples, four were correctly identified. In these cases, the error was on the amount of variation was on the same order as the previous tests. The results shows that this process has the potential of making positive matches, but a more in depth investigation needs to be conducted.

A different response matching routine could likely yield better results. The program efficiency might bene from applying a low pass filter to the data and data LUT before analysis. Another analysis approach would to choose one unique valued common point (or series of points) from all the normalized LUT cooking times, associate that parameter with carrot's cooking time. This would be based on the hypothesis that there are particular frequencies that all the samples attenuate to a unique degree.

Overall, initial results prove promising, but not completely perfect. As shown above, there are many different sources of variation in imaging a carrot sample. When creating the LUT, there is likely on the order of 20-30% error introduced to the measurements. In addition, when an unknown sample is measured, another 20-30% error could be introduced here. The compilation of all these errors has the potential of causing similarities among different response curves, and thus a mismatch. It would be interesting to combine this frequency response analysis approach with previous studies about velocity and attenuation. This would yield three variable with which to identify a sample. This would likely yield a much more accurate and robust texture identification model.

Conclusions

As shown, there is the potential for a high degree of variation when imaging a carrot sample. Sources frequency response error include: variation in the transducer response, variation in same sample readings, variation along the length of a sample, and variation among the biology of different carrots. All these factors ass add to identification inaccuracy. By simply relying on a closest fit match, my program currently can only pr cooking times in intervals that were mentioned. To predict a time outside of these intervals would be hig speculative, and error prone. In addition, although many of the cooking response curves are unique, some of ther are very similar. This adds increased uncertainty in correctly identifying a carrot's texture. At present, only unknown samples have been examined. The program needs to be further tested to evaluate its full potentia efficiency.

In this research, it has been shown that changes in the carrot can be associated with corresponding fre response changes. To a limited degree, it has been shown that high frequency ultrasound can be used to identify the cell texture of some carrots. Combining frequency information with velocity and attenuation information would likely provide a firmer foundation for identification. With further study and analysis, a similar procedu might prove useful for studying other tissues.

Table of Contents

References:

1. Self, G.K, Povey, M.J.W., and Wainwright H. Non-destructive methods of evaluating maturity, Ripening, and quality in tropical fruits. Abstracts of Orally Contributed papers, XXIII International Horticultural Congress 650. (1990)

2. Huisman, H.J., and Thijssen, J.M. Adaptive texture feature extraction with application to ultrasonic image analysis. *Ultrasonic Imaging* 20:132-148, (1998)

3. Povey, M.J.W., and McClements, K.J. Ultrasonics in food engineering. Part I, Introduction and experimental methods. *Journal of Food Engineering* 8:217-245, (1988)

4. Weaver, Richard. Ultrasonics in aluminum foam. *Ultrasonics*. 36:435-442. (1998)

5. Feeney, F.E., Chivers, R.C. Evertsen, J.A., and Keating, J. The influence of inhomogeneity on the propagation of ultrasound in wood. *Ultrasonics* 36:449-453, (1998)

6. Galili, N., Mizrach, A., and Rosenhouse, G. Ultrasonic testing of whole fruit for nondestructive quality evaluation. An ASAE/CSAE meeting presentation paper No. 936026. (1993)

7. Self, G.K., Ordozgoiti, E., Povey, M.J.W., and Wainwright, H. Ultrasonic Evaluation Of ripening avocado flesh. *Posth. Bio. Tech.* 4:111-116, (1994)

8. Self, G.K, Povey, M.J.W., and Wainwright, H. What do ultrasound measurements in Fruit and vegetables tell you? *Developments in Acoustics and Ultrasonics*, M.J.M. Povey and D.J. McClements (Ed.), The Procter Dept. Of Food Science, The University of Leeds, Leeds LS2 9JT. Institute of Physics Publishing. Bristol. (1992)

9. Javanaud, C. Application of ultrasound to food systems. *Ultrasonics* 26:117-23, (1998)

10. Povey, M.J.W. Ultrasonics in food engineering. Part II: Applications. *Journal of Food Engineering*, 9:1-20 (1989)

11. Greve, H., McArdle, R.N., Gohlke, J.R., and Labavitch, J.M. Impact of heating on Carrot firmness: changes in cell wall components. *Journal of Agricultural and Food Chemistry* 42:2900-2906, (1994)

12. Ahmed, E.M., Mirza, S., and Arreola, A.G. Ultrastructural and textural changes in processed carrot tissue. *Journal of Food Quality* 14:321-330, (1991)

13. Nielsen,M.and Martens,H.J. Low frequency ultrasonics for texture measurement in Cooked carrots (daucus carota). *Journal of Food Science* 62:1167-1175, (1997)

14. Gaskill, J.D.,*Linear Systems, Fourier Transforms, and Optics*.John Wiley & Sons, New York, 1978. (ISBN 0-471-29288-5)

Table of Contents | Thesis

High Frequency Ultrasonic Characterization of Carrot Cell Texture

Christopher W. Vick

Appendix A: Carrot Ultrasound Analysis Program - IDL Source Code

The above program, "cuap.pro", was written especially for this research project. It is downloadable in Word 98 document format. It allows for the easy display and analysis of the ultrasound signal data. In order to use this program, data files must contain 128, 256, 512, or 1024 values. In addition, the user is limited to analyzing only 16 files at once. By adjusting the source code, these two small restraints can easily be overcome.

Here is the orignal version (without documentation)
For you to compile:cuap.pro

Table of Contents | Thesis

Appendex A:

cuap.pro

```
pro cuap_event, event                                                    /*************************************************/
Widget_Control, event.id, Get_Value=buttonValue                /*      This program was written to analyze ultrasound   */
CASE buttonValue OF                                                      /*      signals measured in the process of my research,   */
                                                                         /*      'High Frequency Ultrasonic Characterization of   */
 'Display Dataset': BEGIN                                                /*      Carrot Cell Texture."  It is submitted in partial   */
       erase                                                             /*      fulfillment of Senior Research, SIMG-503,        */
       widget_control, event.top, get_uvalue=data              /*      taught by Joseph P. Hornak, at the Chester  F.  */
                                                    /*      Carlson Center for Imaging Science, at the         */
   CASE data.type OF                                                     /*      Rochester Institute of Technology.              */
          'asc': BEGIN                                                   /*      5/10/99          By:   Christopher W. Vick      */
                                                                         /*                      Advisor:   Dr. Rao              */
        data_filename=pickfile(/READ, filter='*.*')            /***********************************************/
               dB = 0.0     ;For making gain calibration                 ;User can pick a single file to display.
               Ag = 1.0     ;amplifier gain

                              T = data.t                    ;Sampling interval
                              header=data.hd                             ;Will file have a header? Yes or No.

                              head=fltarr(6)
               openr,1,data_filename                                     ;Read first value out of the file.
               readf,1,head
               close,1
               dB= head(0)                                               ;First value in file is the gain in dB.
               IF(dB EQ 1) THEN dB=0.0
                              Ag = 10.0^(dB/20.0)       ;dB = 20 log (A) , A=Gain
                              T = head(3)
                              IF(head(3) EQ 0) THEN T=data.t
               file_length = data.pt                                     ;Determine file size from drop down menu.
               function_data=fltarr(file_length)                         ;Create array to hold file data.
               openr,1,data_filename                                     ;Open file,
               readf,1,function_data                                     ;        read in data,
               close,1                                                   ;                  close file.
               END
          'byt': BEGIN
               image_data=pickfile(/READ, filter='*.BYT')                ;Program is currently unable to handle files of the
               END                                                       ;   type 'byt'.
           ELSE: BEGIN
               print, 'Program Error.'
               END
       ENDCASE
               image_size1=1
               image_size2=data.pt
        header=data.hd
        headerless_fun=fltarr(data.pt)
        FOR i=6,data.pt-1 DO BEGIN
           headerless_fun(0)=0                                           ;If function has a header, which fills the first 6
           headerless_fun(1)=0                                           ;   slots, then set those first six values equal
           headerless_fun(2)=0                                           ;   to zero.
           headerless_fun(3)=0
           headerless_fun(4)=0
           headerless_fun(5)=0
           headerless_fun(i)=function_data(i)
         ENDFOR
        N=data.pt
        T=T*1

        xdata=fltarr(N)
        xdata=findgen(N)                                                 ;Make array with the correct spatial
        xdata=xdata*T                                                    ;   labels for the x-axis.
        max=N*T                                                          ;Maximum possible spatial value,
        the_max=N                                                        ;   given array size, and sampling
   for i=0, 100 DO BEGIN                                                 ;   interval.
        IF (function_data(N-1-i) EQ 0) THEN the_max=N-i
        IF (function_data(N-2-1) NE 0) THEN GOTO, HERE                   ;Determine the element number of the
  ENDFOR                                                                 ;   value  zero.  So plot can be moved,
                                                                         ;   and unneeded zeros will not be shown.
        HERE:
        the_min=0                                                        ;Determine the lowest element number
        for i=0, 100 DO BEGIN                                            ;   of  the value zero., so plot can be
                IF (function_data(i) EQ 0) THEN the_min=i                ;   adjusted accordingly.
        IF (function_data(i+1) NE 0) THEN GOTO, HERE2
        ENDFOR
```

```
      HERE2:

      slider1=data.sd1
      slider2=data.sd2
      widget_control, slider1, set_value=the_min                          ;Set the sliders to the minimum and
      widget_control, slider2, set_value=the_max                          ;maximum zero values.

          the_domain=1
                                                                          ;domain=1,time domain
                                                                          ;domain=2,freq domain
      the_min=the_min*T                                                   ;calculate spatial value of min.
      the_max=the_max*T                                                   ;calculate spatial value of max.

           IF(header EQ 'Yes') THEN function_data=headerless_fun
                                                                          ;Adjust for possible amplifier gain.
                   function_data = function_data/Ag                       ;Vin = Vout/A  , A=Gain

      image_info={image1:function_data,t:T,s_dev:data.s_dev,sizeY:image_size2,sd1:data.sd1,
               sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:0,rY:0,
               domain:the_domain, multi:data.multi,marray:data.marray}
                                                                          ;Set all the data into a global
                                                                          ;     variable.

      widget_control, event.top, set_uvalue=image_info

     plot, xdata, function_data, XTitle='time(s)',XRange=[the_min,the_max]
                                                                          ;Plot the function to the screen.
     END

'Resize Plot': BEGIN

   erase
   widget_control, event.top, get_uvalue=data                             ;collect all data from global variable.
   slider1=data.sd1
   slider2=data.sd2
   widget_control, slider1,get_value=s1_value                             ;determine first slider location
   widget_control, slider2,get_value=s2_value                             ;determine second slider location
   function_data=data.image1

   N=data.pt
   T=data.t
   xdata=fltarr(N)
   xdata=findgen(N)
   IF (data.domain EQ 1) THEN xdata=xdata*T                                ;domain=1,time domain
   IF (data.domain NE 1) THEN xdata=xdata*(1.0/(N*T))                      ;domain=2,freq domain

   IF (data.domain EQ 1) THEN max=s2_value*T                              ;Based on whether data is in the spatial or
   IF (data.domain EQ 1) THEN min=s1_value*T                              ;     frequency domain, when the plot is made,
   IF (data.domain NE 1) THEN max=s2_value*(1.0/(N*T))                    ;     see that axis is labeled accordingly.
   IF (data.domain NE 1) THEN min=s1_value*(1.0/(N*T))
   IF(data.domain NE 3) THEN IF(data.domain EQ 1) THEN plot, xdata, function_data, XTitle='time(s)',XRange=[min,max]
   IF(data.domain NE 3) THEN IF(data.domain EQ 2) THEN plot, xdata, function_data, Title='F.F.T.',XTitle='freq(Hz)',XRange=[min,max]
   IF(data.domain NE 3) THEN IF(data.domain EQ 6) THEN plot, xdata, function_data,Title='F.F.T.',XTitle='freq(Hz)',XRange=[min,max]

   IF(data.domain EQ 3) THEN SURFACE, data.marray, Az=s2_value, Ax=s1_value  ;For resizing multiple datasets.

   FOR j=0,data.pt-1 DO BEGIN                                             ;Cut the array in half, because 2nd half is
   IF(data.domain EQ 4) THEN half_fft_array=fltarr(data.pt/2,data.multi)  ;    merely a copy of the first.
   ENDFOR                                                                 ;For a multidimensional array, cut all
                                                                          ;     of the sets in half.
   IF(data.domain EQ 7) THEN half_fft_array=fltarr(data.pt/2,data.multi)

   FOR i=0, data.pt/2-1 DO BEGIN
   IF(data.domain EQ 4) THEN half_fft_array(i,*)=data.marray(i,*)
   IF(data.domain EQ 7) THEN half_fft_array(i,*)=data.marray(i,*)
   ENDFOR

   IF(data.domain EQ 4) THEN SURFACE, half_fft_array, Az=s2_value, Ax=s1_value
   IF(data.domain EQ 7) THEN SURFACE, half_fft_array, Az=S2_value, Ax=s1_value

          image_Info={image1:data.image1, t:data.t, s_dev:data.s_dev, sizeY:data.sizeY, sd1:data.sd1,
                    sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:data.rX,rY:data.rY,
                    domain:data.domain,multi:data.multi,marray:data.marray}
                                                                          ;Feed all the original data back into the
                                                                          ;     global variable.
   widget_control, event.top, set_uvalue=image_info
   END

'Isolate Carrot Response': BEGIN                                          ;Remove response of transducers.
```

```
  widget_control, event.top, get_uvalue=data                          ;Retrieve global data.

       slider1=data.sd1
       slider2=data.sd2
  widget_control, slider2, get_value=s2_value
  widget_control, slider1, get_value=s1_value

       trans_data=fltarr(data.pt)                                     ;Make array to hold transducer response.

  If(data.domain EQ 4) THEN dat=fltarr(data.pt,data.multi)
  IF(data.domain NE 4) THEN dat=fltarr(data.pt)
       data_filename = pickfile(/READ,filter='*.asc')                 ;User picks the name of the transducer
  openr,1,data_filename                                               ;     data file.  File is opened, data is
  readf,1,trans_data                                                  ;     emptied into an array, then closed.
  close,1
  dm=data.multi - 1
  If(data.domain EQ 4) THEN dat=data.marray
       IF(data.domain NE 4) THEN dat=data.image1

       FOR q=0,5 DO BEGIN
  FOR r=0,dm DO BEGIN
  IF(data.domain EQ 4) THEN dat(q,r)=0
  IF(data.domain NE 4) THEN dat(q)=0
       ENDFOR
  ENDFOR

  IF(data.domain NE 4) THEN trans_data(0)=0                           ;Zero out header of transducer data.
  IF(data.domain NE 4) THEN trans_data(1)=0
  IF(data.domain NE 4) THEN trans_data(2)=0
  IF(data.domain NE 4) THEN trans_data(3)=0
  IF(data.domain NE 4) THEN trans_data(4)=0
  IF(data.domain NE 4) THEN trans_data(5)=0

  complex_trans_data=COMPLEX(data.pt)
       complex_trans_data=FFT(trans_data)
  real_part=fltarr(data.pt)
       imag_part=fltarr(data.pt)                                      ;Compute the FFT magnitude of the
       real_part=FLOAT(complex_trans_data)                            ;     transducer data.  Yields the response
       imag_part=IMAGINARY(complex_trans_data)                        ;    of the transducers.
  mag_trans=fltarr(data.pt)
       mag_trans=sqrt(real_part^2.0+imag_part^2.0)

       IF(data.domain NE 4) THEN complex_dat=COMPLEX(data.pt)
       ;IF(data.domain NE 4) THEN complex_dat=FFT(dat)

 IF(data.domain EQ 4) THEN dat_fft_mag=fltarr(data.pt,data.multi)

 FOR i=0,data.multi-1 DO BEGIN

 IF(data.domain EQ 4) THEN complex_num=COMPLEX(data.pt)
 IF(data.domain EQ 4) THEN array=fltarr(data.pt)
 IF(data.domain EQ 4) THEN array=data.marray(*,i)
 IF(data.domain EQ 4) THEN complex_num=FFT(array)
 IF(data.domain EQ 4) THEN real_part=fltarr(data.pt)
 IF(data.domain EQ 4) THEN imag_part=fltarr(data.pt)
 IF(data.domain EQ 4) THEN real_part=FLOAT(complex_num)
 IF(data.domain EQ 4) THEN imag_part=IMAGINARY(complex_num)

 FOR j=0,data.pt-1 DO BEGIN
    IF(data.domain EQ 4) THEN dat_fft_mag(j,i)=sqrt((real_part(j))^2.0+(imag_part(j))^2.0)
 ENDFOR
 ENDFOR

       two_dim_tran=fltarr(data.pt, data.multi)

       FOR i=0,data.multi-1 DO BEGIN
          IF(data.domain EQ 4) THEN two_dim_tran(*,i)=mag_trans
      ENDFOR

  log_trans_array=fltarr(data.pt)                                          ;Go into log space, to
       FOR i=0,data.pt-1 DO BEGIN                                          ;     Isolate carrot response.
  IF(mag_trans(i) GT 0) THEN log_trans_array(i)=ALOG10(mag_trans(i))       ; Uses equations shown in
  IF(mag_trans(i) EQ 0) THEN log_trans_array(i)= 0                         ;     report theory section.
       IF(mag_trans(i) LT 0) THEN log_trans_array(i)= 0
       ENDFOR

  IF(data.domain NE 4) THEN log_output_array=fltarr(data.pt)
  IF(data.domain EQ 4) THEN log_output_array=fltarr(data.pt, data.multi)

  FOR i=0,data.pt-1 DO BEGIN
```

```
      FOR j=0,dm DO BEGIN
              IF(data.domain EQ 4) THEN IF(dat(i,j) GT 0) THEN log_output_array(i,j)=ALOG10(dat(i,j))
      IF(data.domain EQ 4) THEN IF(dat(i,j) EQ 0) THEN log_output_array(i,j)= 0  ;all dat changed from dat_fft_mag
      IF(data.domain EQ 4) THEN IF(dat(i,j) LT 0) THEN log_output_array(i,j) = 0
              ENDFOR
      ENDFOR

      FOR i=0,data.pt-1 DO BEGIN
      value=dat(i)
              IF(data.domain NE 4) THEN IF(dat(i) GT 0) THEN log_output_array(i)=ALOG10(value)
      IF(data.domain NE 4) THEN IF(dat(i) EQ 0) THEN log_output_array(i)= 0
      IF(data.domain NE 4) THEN IF(dat(i) LT 0) THEN log_output_array(i)= 0
      ENDFOR

      IF(data.domain NE 4) THEN carrot_response=fltarr(data.pt)
         IF(data.domain EQ 4) THEN carrot_response=fltarr(data.pt,data.multi)
              FOR i=0,data.pt-1 DO BEGIN
             difference=log_output_array(i)-log_trans_array(i)
              IF(data.domain NE 4) THEN carrot_response(i)=10.0^(difference)
        ENDFOR
             IF(data.domain EQ 4) THEN two_dim_trans_array=fltarr(data.pt, data.multi)
              FOR i=0,dm DO BEGIN
               IF(data.domain EQ 4) THEN two_dim_trans_array(*,i)=log_trans_array
             ENDFOR
        IF(data.domain EQ 4) THEN mag_td=fltarr(data.pt,data.multi)
        IF(data.domain EQ 4) THEN mag_td=data.marray

      log_out_array=fltarr(data.pt, data.multi)
      FOR i=0, data.pt-1 DO BEGIN
        FOR j=0, data.multi-1 DO BEGIN
          IF(data.domain EQ 4) THEN IF(mag_td(i,j) GT 0) THEN log_out_array(i,j)=ALOG10(mag_td(i,j))
          IF(data.domain EQ 4) THEN IF(mag_td(i,j) EQ 0) THEN log_out_array(i,j)=0
          IF(data.domain EQ 4) THEN IF(mag_td(i,j) LT 0) THEN log_out_array(i,j)=0
        ENDFOR
      ENDFOR

      td_carrot_response=fltarr(data.pt, data.multi)
      FOR i=0, data.pt-1 DO BEGIN
         FOR j=0, dm DO BEGIN
              IF(data.domain EQ 4) THEN td_carrot_response(i,j)=10.0^(log_out_array(i,j)-two_dim_trans_array(i,j))
         ENDFOR
      ENDFOR

    N=data.pt
    T=data.t
    xdata=fltarr(N)
    xdata=findgen(N)
    xdata=xdata*(1.0/(N*T))      ;domain=2,freq domain
    max=s2_value*(1.0/(N*T))
    min=s1_value*(1.0/(N*T))
                                                                        ;Finally, plot the isolated carrot
                                                                        ;     response curve.
    IF(data.domain NE 4) THEN plot, xdata, carrot_response, Title='Carrot Response:', XTitle='freq(Hz)', XRange=[min,max]
    IF(data.domain EQ 4) THEN half_fft_array=fltarr(data.pt/2,data.multi)

    FOR i=0, data.pt/2-1 DO BEGIN
        IF(data.domain EQ 4) THEN half_fft_array(i,*)=td_carrot_response(i,*)
    ENDFOR

    IF(data.domain EQ 4) THEN SURFACE, half_fft_array, Az=30, Ax=30
    IF(data.domain NE 4) THEN dmain=6
    IF(data.domain EQ 4) THEN dmain=7

        image_Info={image1:carrot_response, t:data.t, s_dev:data.s_dev, sizeY:data.sizeY,sd1:data.sd1,
                        sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:data.rX,ry:data.rY,domain:dmain,
                        multi:data.multi,marray:td_carrot_response}
     widget_control, event.top, set_uvalue=image_info
      END

'Calculate Statistics': BEGIN

    widget_control, event.top, get_uvalue=data

    slider1=data.sd1
    slider2=data.sd2
    widget_control, slider2, get_value=s2_value
    widget_control, slider1, get_value=s1_value

    N=data.pt
    T=data.t
```

```
   xdata=fltarr(N)
   xdata=findgen(N)
   xdata=xdata*(1.0/(N*T))
   max=10E6
   min=0

   multi_data_array=fltarr(data.pt/2,data.multi)
   multi_data_array=data.marray                          ;Declare arrays to hold the
   mean_array=fltarr(data.pt/2)                          ;     statistical data.
   statistics=fltarr(4)
   single_array=fltarr(data.multi)
   var_array=fltarr(data.pt/2)

   i=0
   j=0

   FOR i=0,data.pt/2-1 DO BEGIN
       FOR j=0,data.multi-1 DO BEGIN
           single_array(j)=multi_data_array(i,j)
       ENDFOR
           statistics=MOMENT(single_array)               ;Calculate the mean, variance
           mean_array(i)=statistics[0]                   ;     of  all the datasets that are
           var_array(i)=statistics[1]                    ;     graphed.
   ENDFOR

   sd_array=fltarr(data.pt/2)
   sd_array=sqrt(var_array)                              ;Compute standard deviation, from variance.

   erase
                                                         ;Display mean graph, with appropriate
                                                         ;     SD error bars.
   plot, xdata, mean_array, Title='Mean Response:', XTitle='freq(Hz)', XRange=[min,max]
   Errplot, xdata, mean_array-sd_array, mean_array+sd_array

   k=0
   CT=0.0

   FOR i=0,data.pt/2-1 DO BEGIN
       CT=CT+((sd_array(i)/mean_array(i))*100)           ;Calculate the % variation.
   ENDFOR

   print, CT/(data.pt/2)                                 ;Print out variation value.
   dmain=0
   dmain=2   ;Back to freq domain.

       image_Info={image1:mean_array, t:data.t, s_dev:sd_array, sizeY:data.sizeY,sd1:data.sd1,
                  sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:data.rX,ry:data.rY,domain:dmain,
                  multi:data.multi,marray:data.marray}

  widget_control, event.top, set_uvalue=image_info
    END

'Normalize Dataset': BEGIN
     erase
     widget_control, event.top, get_uvalue=data

            N=data.pt
             T=data.pt
             xdata=fltarr(N)
             xdata=findgen(N)
     xdata=xdata*(1.0/(N*T))

     data_array=fltarr(data.pt,data.multi)
             data_array=data.marray
             normal_array=fltarr(data.pt,data.multi)

             the_max=0.0
     FOR i=0,data.multi-1 DO BEGIN
            the_max=MAX(data_array(*,i))                 ;Normalize a function by determining its
             normal_array(*,i)=data_array(*,i)/the_max   ;   maximum value, and dividing all the
     ENDFOR                                              ;   values by it.  Thus scales a function
                                                         ;   between 0 and 1.
             half_array=fltarr(data.pt/2,data.multi)
     FOR j=0,data.pt/2-1 DO BEGIN
            half_array(j,*)=normal_array(j,*)
     ENDFOR

             surface, half_array
             image_info={image1:data.image1,t:data.t,s_dev:data.s_dev,sizeY:data.sizeY,sd1:data.sd1,
                        sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:data.rX,rY:data.rY,
                                   domain:data.domain,multi:data.multi,marray:normal_array}
       widget_control, event.top, set_uvalue=image_info
       END

'Unknown Dataset': BEGIN                                   ;Perform analysis on the unknown dataset.
     widget_control, event.top, get_uvalue=data
             N=data.pt
             T=data.t
             xdata=fltarr(N)
             xdata=findgen(N)
             xdata=xdata*(1.0/(N*T))

             unk_data=fltarr(N)
             unk_filename=pickfile(/READ,filter='*.*')        ;User picks the unknown filename.
             openr,1,unk_filename
             readf,1,unk_data
             close,1

             unk_data(0)=0                                    ;Header is set to zero.
             unk_data(1)=0
             unk_data(2)=0
             unk_data(3)=0
             unk_data(4)=0
             unk_data(5)=0

     complex_unk=COMPLEX(N)                                ;FFT of the unknown sample is
     complex_unk=FFT(unk_data)                             ;     computed.
             real_part=fltarr(N)
             imag_part=fltarr(N)
             real_part=FLOAT(complex_unk)
             imag_part=IMAGINARY(complex_unk)
             mag=fltarr(N)
             mag=sqrt(real_part^2.0+imag_part^2.0)
             norm_mag=fltarr(N)                               ;Determine FFT Magnitude.
             max_mag=0.0
             max_mag=MAX(mag)
             norm_mag=mag/max_mag

     stored_array=fltarr(data.pt,data.multi)
     stored_array=data.marray
     pos_result=fltarr(data.multi)                          ;Will stores variances, to decide match.

     temp_array=fltarr(2)
     stats=fltarr(4)
     var=0.0

     FOR i=0,data.multi-1 DO BEGIN
         FOR j=0,data.pt/2-1 DO BEGIN                      ;Calculate statistics of the possible
             temp_array(0)=norm_mag(j)                    ;   matches.
            temp_array(1)=stored_array(j,i)
            stats=MOMENT(temp_array)
            var=stats(1)
            pos_result(i)=pos_result(i)+var
         ENDFOR
     ENDFOR

     invert_result=fltarr(data.multi)
     for l=0, data.multi-1 DO BEGIN
     invert_result(l)=pos_result(data.multi-1-l)
     ENDFOR

     min_element=0
     FOR k=0,data.multi-1 DO BEGIN
         print, k, ' k: ', pos_result(k)                   ;Pick the function with the closest match.
         IF(invert_result(k) LT invert_result(min_element)) THEN min_element=k
         print, min_element, ' min: ', invert_result(min_element)
     ENDFOR

             print, "Your Unknown Most Closely Matches Function: ",1+min_element;adjust for 0 element #

     END

"Plot F.F.T.': BEGIN
      erase
      widget_control, event.top, get_uvalue=data

      N=data.pt
      T=data.t
      xdata=fltarr(N)
      xdata=findgen(N)
```

```
        xdata=xdata*(1.0/(N*T))

        slider2=data.sd2
        widget_control, slider2, set_value=data.pt/2

        IF(data.domain NE 3) THEN fft_data=fltarr(N)                    ;Simple.  Calculate the magnitude of
        IF(data.domain NE 3) THEN fft_data=FFT(data.image1)             ;   the FFT of the input dataset.  Uses
        IF(data.domain NE 3) THEN max=N/2*(1.0/(N*T))                   ;  the canned IDL fft algorithm.
        IF(data.domain EQ 3) THEN fft_data=fltarr(N,data.multi)
        IF(data.domain EQ 3) THEN fft_data=data.marray                  ;This here is the FFT for a single
        IF(data.domain EQ 3) THEN array=fltarr(N)                       ;     dataset.
        IF(data.domain EQ 3) THEN complex_array=COMPLEX(N)
        IF(data.domain EQ 3) THEN fft_mag=fltarr(data.pt,data.multi)
        IF(data.domain NE 3) THEN fft_mag=fltarr(data.pt)

      FOR i=0, data.multi-1 DO BEGIN
        array=fft_data(*,i)
        complex_array=FFT(array)                                        ;This is the FFT for a multidimensional
            IF(data.domain NE 3) THEN real_part=fltarr(data.pt)         ;     dataset.
            IF(data.domain NE 3) THEN imag_part=fltarr(data.pt)
            IF(data.domain NE 3) THEN real_part=FLOAT(fft_data)
            IF(data.domain NE 3) THEN imag_part=IMAGINARY(fft_data)
            IF(data.domain EQ 3) THEN real_part=fltarr(N)
            IF(data.domain EQ 3) THEN imag_part=fltarr(N)
            IF(data.domain EQ 3) THEN real_part=FLOAT(complex_array)
            IF(data.domain EQ 3) THEN imag_part=IMAGINARY(complex_array)
                  FOR j=0,data.pt-1 DO BEGIN
                      IF(data.domain EQ 3) THEN fft_mag(j,i)=sqrt((real_part(j))^2.0+(imag_part(j))^2.0)
                ENDFOR
                      IF(data.domain NE 3) THEN fft_mag=sqrt(real_part^2.0+imag_part^2.0)
      ENDFOR

        IF(data.domain EQ 3) THEN half_fft_data=fltarr(data.pt/2,data.multi)

       FOR i=0,N/2-1 DO BEGIN
            IF(data.domain EQ 3) THEN half_fft_data(i,*)=fft_mag(i,*)   ;Cut off second half of FFT.
       ENDFOR
                                                                        ;Plot FFT, single file case.
        IF(data.domain NE 3) THEN plot,xdata,fft_mag, Title='F.F.T.',XTitle='freq(Hz)',XRange=[0,max]
        IF(data.domain EQ 3) THEN SURFACE, half_fft_data                ;Plot the FFT, multi file case.
        IF(data.domain NE 3) THEN half_fft_data=0
        IF(data.domain EQ 3) THEN what_domain=4
        IF(data.domain NE 3) THEN what_domain=2

        image_info={image1:fft_mag,t:data.t,s_dev:data.s_dev,sizeY:data.sizeY,sd1:data.sd1,
                        sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:data.rX,
                        rY:data.rY,domain:what_domain,multi:data.multi,marray:fft_mag}
       widget_control, event.top, set_uvalue=image_info
    END

 'Output to file': BEGIN
        widget_control, event.top, get_uvalue=data                      ;Ask user for a filename
        output_filename=pickfile(/READ)                                 ;     in which to output the
        function_data=data.image1                                       ;     statistical data, for further
        function_sd=data.s_dev                                          ;     examination in Microsoft
        openw,1,output_filename                                         ;     Excel.

        FOR i=0, data.pt/2-1 DO BEGIN
                printf,1,function_data(i)                               ;Prints function data into the file.
        ENDFOR
        printf,1,'sd'

         FOR j=0, data.pt/2-1 DO BEGIN                                  ;Sends SD into the specified file.
                printf,1,function_sd(j)                                 ;Change this to send other information
        ENDFOR                                                          ;     into a file.
                close,1
        print, 'Data Written to: ', output_filename                     ;Verify that data was indeed written.
    END

'Multi-Dataset Display': BEGIN
        widget_control, event.top, get_uvalue=data

        first_filename=pickfile(/READ,filter='*.*')                                    ;Ask for up to 16 files
        second_filename=pickfile(/READ,filter='*.*')                                   ;   to display at once.
        IF(data.multi GT 2) THEN third_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 3) THEN fourth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 4) THEN fifth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 5) THEN sixth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 6) THEN seventh_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 7) THEN eigth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 8) THEN ninth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 9) THEN tenth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 10) THEN eleventh_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 11) THEN twelvth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 12) THEN thirteenth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 13) THEN fourteenth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 14) THEN fifteenth_filename=pickfile(/READ,filter='*.*')
        IF(data.multi GT 15) THEN sixteenth_filename=pickfile(/READ,filter='*.*')

        array_one=fltarr(data.pt)                                   ;Declare an array for
        array_two=fltarr(data.pt)                                   ;    each file, of the
        array_three=fltarr(data.pt)                                 ;    appropriate size.
        array_four=fltarr(data.pt)
        array_five=fltarr(data.pt)
        array_six=fltarr(data.pt)
        array_seven=fltarr(data.pt)
        array_eight=fltarr(data.pt)
        array_nine=fltarr(data.pt)
        array_ten=fltarr(data.pt)
        array_eleven=fltarr(data.pt)
        array_twelve=fltarr(data.pt)
        array_thirteen=fltarr(data.pt)
        array_fourteen=fltarr(data.pt)
        array_fifteen=fltarr(data.pt)
        array_sixteen=fltarr(data.pt)

             head=fltarr(6)
             openr,1,first_filename                                 ;Open first  file,  read
             readf,1,head                                           ;     the first (gain) value.
             close,1
             dB=head(0)
             Ag= 10.0^(dB/20.0)
             file_length = data.pt                                  ;Declare file to hold data, of
             array_one=fltarr(file_length)                          ;    proper size.
             openr,1,first_filename                                 ;Open file,
             readf,1,array_one                                      ;            read data into array,
             close,1                                                ;            close file.
             array_one=array_one/Ag                                 ; Adjust for gain.


             head=fltarr(6)                                         ;Open second file, read
             openr,1,second_filename                                ;    the first(gain) value.
             readf,1,head
             close,1
             dB=head(0)
             Ag= 10.0^(dB/20.0)
             array_two=fltarr(file_length)                          ;Declare file to hold data, of
             openr,1,second_filename                                ;     proper size.
             readf,1,array_two                                      ;Open file,
             close,1                                                ;            read data into array.
             array_two=array_two/Ag                                 ;Adjust for gain.

             head=fltarr(6)                                         ;Repeat above steps,  for all the
             IF(data.multi GT 2) THEN openr,1,third_filename        ;     files that the user specified,
             IF(data.multi GT 2) THEN readf,1,head                  ;     up to 16.
             IF(data.multi GT 2) THEN close,1
             dB=head(0)
             Ag= 10.0^(dB/20.0)
             IF(data.multi GT 2) THEN array_three=fltarr(file_length)
             IF(data.multi GT 2) THEN openr,1,third_filename
             IF(data.multi GT 2) THEN readf,1,array_three
             IF(data.multi GT 2) THEN close,1
             array_three=array_three/Ag

             head=fltarr(6)
             IF(data.multi GT 3) THEN openr,1,fourth_filename
             IF(data.multi GT 3) THEN readf,1,head
             IF(data.multi GT 3) THEN close,1
             dB=head(0)
             Ag= 10.0^(dB/20.0)
             IF(data.multi GT 3) THEN array_four=fltarr(file_length)
             IF(data.multi GT 3) THEN openr,1,fourth_filename
             IF(data.multi GT 3) THEN readf,1,array_four
             IF(data.multi GT 3) THEN close,1
             array_four=array_four/Ag

             head=fltarr(6)
             IF(data.multi GT 4) THEN openr,1,fifth_filename
             IF(data.multi GT 4) THEN readf,1,head
             IF(data.multi GT 4) THEN close,1
```

```
dB=head(0)
Ag= 10.0^(dB/20.0)
IF(data.multi GT 4) THEN array_five=fltarr(file_length)
IF(data.multi GT 4) THEN openr,1,fifth_filename
IF(data.multi GT 4) THEN readf,1,array_five
IF(data.multi GT 4) THEN close,1
array_five=array_five/Ag

head=fltarr(6)
IF(data.multi GT 5) THEN openr,1,sixth_filename
IF(data.multi GT 5) THEN readf,1,head
IF(data.multi GT 5) THEN close,1
dB=head(0)
Ag= 10.0^(dB/20.0)
IF(data.multi GT 5) THEN array_six=fltarr(file_length)
IF(data.multi GT 5) THEN openr,1,sixth_filename
IF(data.multi GT 5) THEN readf,1,array_six
IF(data.multi GT 5) THEN close,1
array_six=array_six/Ag

head=fltarr(6)
IF(data.multi GT 6) THEN openr,1,seventh_filename
IF(data.multi GT 6) THEN readf,1,head
IF(data.multi GT 6) THEN close,1
dB=head(0)
Ag= 10.0^(dB/20.0)
IF(data.multi GT 6) THEN array_seven=fltarr(file_length)
IF(data.multi GT 6) THEN openr,1,seventh_filename
IF(data.multi GT 6) THEN readf,1,array_seven
IF(data.multi GT 6) THEN close,1
array_seven=array_seven/Ag

head=fltarr(6)
IF(data.multi GT 7) THEN openr,1,eigth_filename
IF(data.multi GT 7) THEN readf,1,head
IF(data.multi GT 7) THEN close,1
dB=head(0)
Ag= 10.0^(dB/20.0)
IF(data.multi GT 7) THEN array_eight=fltarr(file_length)
IF(data.multi GT 7) THEN openr,1,eigth_filename
IF(data.multi GT 7) THEN readf,1,array_eight
IF(data.multi GT 7) THEN close,1
array_eight=array_eight/Ag

head=fltarr(6)
IF(data.multi GT 8) THEN openr,1,ninth_filename
IF(data.multi GT 8) THEN readf,1,head
IF(data.multi GT 8) THEN close,1
dB=head(0)
Ag= 10.0^(dB/20.0)
IF(data.multi GT 8) THEN array_nine=fltarr(file_length)
IF(data.multi GT 8) THEN openr,1,ninth_filename
IF(data.multi GT 8) THEN readf,1,array_nine
IF(data.multi GT 8) THEN close,1
array_nine=array_nine/Ag

head=fltarr(6)
IF(data.multi GT 9) THEN openr,1,tenth_filename
IF(data.multi GT 9) THEN readf,1,head
IF(data.multi GT 9) THEN close,1
dB=head(0)
Ag= 10.0^(dB/20.0)
IF(data.multi GT 9) THEN array_ten=fltarr(file_length)
IF(data.multi GT 9) THEN openr,1,tenth_filename
IF(data.multi GT 9) THEN readf,1,array_ten
IF(data.multi GT 9) THEN close,1
array_ten=array_ten/Ag

head=fltarr(6)
IF(data.multi GT 10) THEN openr,1,eleventh_filename
IF(data.multi GT 10) THEN readf,1,head
IF(data.multi GT 10) THEN close,1
dB=head(0)
Ag= 10.0^(dB/20.0)
IF(data.multi GT 10) THEN array_eleven=fltarr(file_length)
IF(data.multi GT 10) THEN openr,1,eleventh_filename
IF(data.multi GT 10) THEN readf,1,array_eleven
IF(data.multi GT 10) THEN close,1
array_eleven=array_eleven/Ag
```

```
          head=fltarr(6)
          IF(data.multi GT 11) THEN openr,1,twelvth_filename
          IF(data.multi GT 11) THEN readf,1,head
          IF(data.multi GT 11) THEN close,1
          dB=head(0)
          Ag= 10.0^(dB/20.0)
          IF(data.multi GT 11) THEN array_twelve=fltarr(file_length)
          IF(data.multi GT 11) THEN openr,1,twelvth_filename
          IF(data.multi GT 11) THEN readf,1,array_twelve
          IF(data.multi GT 11) THEN close,1
          array_twelve=array_twelve/Ag

          head=fltarr(6)
          IF(data.multi GT 12) THEN openr,1,thirteenth_filename
          IF(data.multi GT 12) THEN readf,1,head
          IF(data.multi GT 12) THEN close,1
          dB=head(0)
          Ag= 10.0^(dB/20.0)
          IF(data.multi GT 12) THEN array_thirteen=fltarr(file_length)
          IF(data.multi GT 12) THEN openr,1,thirteenth_filename
          IF(data.multi GT 12) THEN readf,1,array_thirteen
          IF(data.multi GT 12) THEN close,1
          array_thirteen=array_thirteen/Ag

          head=fltarr(6)
          IF(data.multi GT 13) THEN openr,1,fourteenth_filename
          IF(data.multi GT 13) THEN readf,1,head
          IF(data.multi GT 13) THEN close,1
          dB=head(0)
          Ag= 10.0^(dB/20.0)
          IF(data.multi GT 13) THEN array_fourteen=fltarr(file_length)
          IF(data.multi GT 13) THEN openr,1,fourteenth_filename
          IF(data.multi GT 13) THEN readf,1,array_fourteen
          IF(data.multi GT 13) THEN close,1
          array_fourteen=array_fourteen/Ag

          head=fltarr(6)
          IF(data.multi GT 14) THEN openr,1,fifteenth_filename
          IF(data.multi GT 14) THEN readf,1,head
          IF(data.multi GT 14) THEN close,1
          dB=head(0)
          Ag= 10.0^(dB/20.0)
          IF(data.multi GT 14) THEN array_fifteen=fltarr(file_length)
          IF(data.multi GT 14) THEN openr,1,fifteenth_filename
          IF(data.multi GT 14) THEN readf,1,array_fifteen
          IF(data.multi GT 14) THEN close,1
          array_fifteen=array_fifteen/Ag

          head=fltarr(6)
          IF(data.multi GT 15) THEN openr,1,sixteenth_filename
          IF(data.multi GT 15) THEN readf,1,head
          IF(data.multi GT 15) THEN close,1
          dB=head(0)
          Ag= 10.0^(dB/20.0)
          IF(data.multi GT 15) THEN array_sixteen=fltarr(file_length)
          IF(data.multi GT 15) THEN openr,1,sixteenth_filename
          IF(data.multi GT 15) THEN readf,1,array_sixteen
          IF(data.multi GT 15) THEN close,1
          array_sixteen=array_sixteen/Ag


FOR i=0,5 DO BEGIN
          IF(data.hd EQ 'Yes') THEN    array_one(i)=0                         ;Zero out the header of each array.
          IF(data.hd EQ 'Yes') THEN    array_two(i)=0
          IF(data.hd EQ 'Yes') THEN    array_three(i)=0
          IF(data.hd EQ 'Yes') THEN    array_four(i)=0
          IF(data.hd EQ 'Yes') THEN    array_five(i)=0
          IF(data.hd EQ 'Yes') THEN    array_six(i)=0
          IF(data.hd EQ 'Yes') THEN    array_seven(i)=0
          IF(data.hd EQ 'Yes') THEN    array_eight(i)=0
          IF(data.hd EQ 'Yes') THEN    array_nine(i)=0
          IF(data.hd EQ 'Yes') THEN    array_ten(i)=0
          IF(data.hd EQ 'Yes') THEN    array_eleven(i)=0
          IF(data.hd EQ 'Yes') THEN    array_twelve(i)=0
          IF(data.hd EQ 'Yes') THEN    array_thirteen(i)=0
          IF(data.hd EQ 'Yes') THEN    array_fourteen(i)=0
          IF(data.hd EQ 'Yes') THEN    array_fifteen(i)=0
          IF(data.hd EQ 'Yes') THEN    array_sixteen(i)=0
 ENDFOR
```

```
               two_dim_array=fltarr(data.pt,data.multi)
               two_dim_array(*,data.multi-1)=array_one
               two_dim_array(*,data.multi-2)=array_two
               IF(data.multi GT 2) THEN two_dim_array(*,data.multi-3)=array_three      ;Compile all the data into a large
               IF(data.multi GT 3) THEN two_dim_array(*,data.multi-4)=array_four       ;     multidimensional array.
               IF(data.multi GT 4) THEN two_dim_array(*,data.multi-5)=array_five
               IF(data.multi GT 5) THEN two_dim_array(*,data.multi-6)=array_six
               IF(data.multi GT 6) THEN two_dim_array(*,data.multi-7)=array_seven
               IF(data.multi GT 7) THEN two_dim_array(*,data.multi-8)=array_eight
               IF(data.multi GT 8) THEN two_dim_array(*,data.multi-9)=array_nine
               IF(data.multi GT 9) THEN two_dim_array(*,data.multi-10)=array_ten
               IF(data.multi GT 10) THEN two_dim_array(*,data.multi-11)=array_eleven
               IF(data.multi GT 11) THEN two_dim_array(*,data.multi-12)=array_twelve
               IF(data.multi GT 12) THEN two_dim_array(*,data.multi-13)=array_thirteen
               IF(data.multi GT 13) THEN two_dim_array(*,data.multi-14)=array_fourteen
               IF(data.multi GT 14) THEN two_dim_array(*,data.multi-15)=array_fifteen
               IF(data.multi GT 15) THEN two_dim_array(*,data.multi-16)=array_sixteen

               AX=30        ;Default x angle of rotation for 3-D plot.
               AZ=50        ;Default z angle of rotation for 3-D plot.

               slider1=data.sd1
               slider2=data.sd2

               widget_control,slider1, get_value=s1_value                             ;Determine sliders value, to rotate plot.
               widget_control,slider2, get_value=s2_value


               SURFACE, two_dim_array ;, Az=s2_value, Ax=s1_value                     ;Plot the 3-D data function.

        image_info={image1:data.image1,t:data.t,s_dev:data.s_dev,sizeY:data.sizeY,sd1:data.sd1,
                    sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:data.rx,rY:data.rY,
                     domain:3,multi:data.multi,marray:two_dim_array}
        widget_control, event.top, set_uvalue=image_info
     END

'Clip Dataset': BEGIN                                                                 ;As necessary, unnecessary parts of the
                                                                                      ;     dataset can be clipped, such as
      widget_control, event.top, get_uvalue=data                                      ;     extra end zeros.
      slider1=data.sd1
      slider2=data.sd2
      widget_control, slider1,get_value=s1_value
      widget_control, slider2,get_value=s2_value
      function_data=data.image1
      clipped_function=fltarr(data.pt)
      clipped_function=function_data

      FOR i=0,s1_value-1 DO BEGIN
          IF(s1_value NE 0) THEN clipped_function(i)=0
      ENDFOR

      FOR i=s2_value+1,data.pt-1 DO BEGIN
          IF(s2_value LT data.pt) THEN clipped_function(i)=0
      ENDFOR

          image_Info={image1:clipped_function, t:data.t, s_dev:data.s_dev, sizeY:data.sizeY,sd1:data.sd1,
                      sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:data.rX,rY:data.rY,
                      domain:data.domain,multi:data.multi,marray:data.marray}
          widget_control, event.top, set_uvalue=image_info
      END

 'Quit': BEGIN
        Widget_Control, event.top, /Destroy
     END

'Erase Screen': BEGIN
           erase
        END

ENDCASE
END


  PRO HandleList, event                                                   ;For pull down menu.
    Widget_control, event.id, Get_Uvalue=options                          ;Keeps track of  file type.
    Widget_control, event.top, get_uvalue=data
      image_info={image1:data.image1,t:data.t,s_dev:data.s_dev,sizeY:data.sizeY, sd1:data.sd1,
                  sd2:data.sd2, type:options(event.index),pt:data.pt,hd:data.hd,widy:data.widy,
                  rX:data.rX,rY:data.rY,domain:data.domain,multi:data.multi,marray:data.marray}
      widget_control, event.top, set_uvalue=image_info
```

```
END

PRO HandleList2, event                                              ;For Pull down menu.
 Widget_control, event.id, Get_Uvalue=options2                      ;Keeps track of # of data points.
 Widget_control, event.top, get_uvalue=data
   image_info={image1:data.image1,t:data.t,s_dev:data.s_dev,sizeY:data.sizeY, sd1:data.sd1,
                     sd2:data.sd2,type:data.type,pt:options2(event.index),hd:data.hd,widy:data.widy,
                     rx:data.rX,rY:data.rY,domain:data.domain,multi:data.multi,marray:data.marray}
   widget_control, event.top, set_uvalue=image_info
END

PRO HandleList3, event                                              ;For pull down menu.
  Widget_control, event.id, Get_Uvalue=options3                     ;Keeps track if there is a
  Widget_control, event.top, get_uvalue=data                        ;    file header or not.
      image_info={image1:data.image1,t:data.t,s_dev:data.s_dev,sizeY:data.sizeY,sd1:data.sd1,
                        sd2:data.sd2,type:data.type,pt:data.pt,hd:options3(event.index),widy:data.widy,
                         rX:data.rX,rY:data.rY,domain:data.domain,multi:data.multi,marray:data.marray}
      widget_control, event.top, set_uvalue=image_info
  END


PRO HandleList4, event                                              ;For pull down menu.
       Widget_control, event.id, Get_Uvalue=options4                ;Keeps track of the  number
       Widget_control, event.top, get_uvalue=data                   ;     of files you specified.
             image_info={image1:data.image1,t:data.t,s_dev:data.s_dev,sizeY:data.sizeY,sd1:data.sd1,
             sd2:data.sd2,type:data.type,pt:data.pt,hd:data.hd,widy:data.widy,rX:data.rX,rY:data.rY,
             domain:data.domain,multi:options4(event.index),marray:data.marray}
        widget_control, event.top, set_uvalue=image_info
END


PRO fslider_size_event, event                                       ;Keeps track of the
   widget_control, event.top, get_uvalue=data                       ;     position of the sliders.
   slider1=data.sd1
   slider2=data.sd2
   Widget_control, slider1 ,get_value=s1_value
   Widget_control, slider2 ,get_value=s2_value
 END

PRO cuap
  t1b = Widget_Base(Column=1, Title='Carrot Ultrasound Signal Analysis:
                                          [C.Vick, Advisor:Dr.Rao 5/10/99]')
  t2b = Widget_Base(t1b, row=1)
  t4b = Widget_Base(t1b, row=3,grid_layout=1)
                                                                    ;Declare pull down menu widgets.
  options = ['asc', 'byt']
  listMenu = Widget_Droplist(t2b, Value=options, Title='File Type: ',
                                   Frame=1, Event_Pro='HandleList', UValue=options)
  options2 = ['256','128','512','1024', 'other']
  listMenu2 = Widget_Droplist(t2b, Value=options2, Title='Data Points: ',
                                    Frame=1, Event_Pro='HandleList2', UValue=options2)
  options3 = ['Yes', 'No']
  listMenu3 = Widget_Droplist(t2b, Value=options3, Title='File Header: ',
                                    Frame=1, Event_Pro='HandleList3', UValue=options3)
  options4 = ['1','2','3','4','5','6','7','8','9','10','11','12','13','14','15','16']
  listMenu4 = Widget_Droplist(t2b, Value=options4, Title='Num.of Files: ',
                                   Frame=1, Event_Pro='HandleList4', UValue=options4)

                                                                    ;Declare widget buttons.
  loadButton = Widget_Button(t4b, Value='Display Dataset')
  resiButton = Widget_Button(t4b, Value='Resize Plot')
  clipButton = Widget_Button(t4b, Value='Clip Dataset')
  fft_Button = Widget_Button(t4b, Value='Plot F.F.T.')

  horiButton = Widget_Button(t4b, Value='Multi-Dataset Display')
  carrButton = Widget_Button(t4b, Value='Isolate Carrot Response')
  variButton = Widget_Button(t4b, Value='Calculate Statistics')
  normButton = Widget_Button(t4b, Value='Normalize Dataset')

  enhaButton = Widget_Button(t4b, Value='Unknown Dataset')
  writButton = Widget_Button(t4b, Value='Output to file')
  erasButton = Widget_Button(t4b, Value='Erase Screen')
  quitButton = Widget_Button(t4b, Value='Quit')
                                                                    ;User can specify the size of the
   read, "Window Heigth [350]:", h                                  ;     display window.

  window1 = Widget_Draw(t1b, XSize=700, YSize=h)

  t3b=Widget_Base(t1b,row=1)
  text=Widget_label(t3b, Value=' Plot: Xmin,Xmax  ',dynamic_resize=1)
```

```
   f1=Widget_Slider(t3b, maximum=512)
   f2=Widget_Slider(t3b, maximum=512)
   widget_control,f1,set_value=0
   widget_control,f2,set_value=512

    widget_control, window1, get_value=w_id
                                                                     ;Declare global variable to hold
                                                                     ;     all the important data.
    image_info ={image1:0,t:50.0*(10.0^(-9.0)),s_dev:0,sizeY:0,sd1:f1,sd2:f2,type:'asc',
                  pt:256,hd:'Yes',widy:w_id,rX:0,rY:0,domain:1,multi:1,marray:0}

       Widget_control, t1b, set_uvalue= image_info
       XManager, 'fslider_size', t3b, event_handler='fslider_size_event'
   Widget_Control, t1b, /realize
   XManager, 'cuap', t1b, Event='cuap_event'

END
```