2005

Preparation of acoustic emission data for neural network analysis using awk and C programs

Avinash Kaza
West Virginia University

Preparation of Acoustic Emission Data for Neural Network Analysis using AWK and C Programs

By

Avinash Kaza

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

Master of Science
In
Engineering

Roger H. L. Chen, Ph.D., Chair.
Alejandro Kiriakidis, Ph.D.
Bruce Kang, Ph.D.
Eric Johnson, Ph.D.

Department of Civil and Environmental Engineering

Morgantown, West Virginia
2005

Keywords: Acoustic Emissions, Fiber Reinforced Polymer Composites, Neural Networks, AWK

Abstract

Preparation of Acoustic Emission Data for Neural Network Analysis using awk and C programs

Avinash Kaza

Fiber Reinforced Polymer composites are relatively new to the bridge construction industry. One of the challenges to greater usage is the lack of a real-time nondestructive method to accurately evaluate them. A Neural Network program can be used to predict the status of a structure by analyzing summarized Acoustic Emission data from the structure in real-time.

Existing methods use tools such as MS Excel to extract the Neural Network input matrix from raw data having more than 30,000 rows of Acoustic Emission data for a simple three point bending test. This manual data extraction process is very time consuming. Therefore, application software was developed to efficiently summarize the Acoustic Emission data into a Neural Network input matrix. The awk and C scripts were used to develop the application. The awk programming language is designed to search for, match patterns, and perform actions on text files. The awk programs are generally quite small, easy to understand and are easily interpreted. This makes it a good pattern matching and data retrieval language. An awk program can be executed by using a gcc compiler, so a C module was developed to combine all the awk scripts into a single application. The application software was used to extract the Neural Network input matrix from previously conducted AE experiments; a comparison of manually prepared matrices and those prepared by application is presented.

Tension, Bending and Fatigue experiments of FRP specimens were conducted and the AE data obtained from the experiments were used to analyze the structural properties of the specimens with the help of the application software developed through this study. Loading quarter of the specimens was predicted using the Neural Network program. The predictions obtained from Neural Network program for the matrices prepared by the application software were found to be more accurate and consistent than the predictions from the manually prepared matrices.

Dedication

This thesis is dedicated to my beloved parents, Durga Prasad, Roja Rani, and my brother Bhavadeesh. Thank you for offering me the unconditional love and support.

Acknowledgements

I would like to thank my adviser, Dr. Roger H. L. Chen for providing me with this opportunity and for his guidance and knowledge throughout the project. I would like to extend my special thanks to my committee member, Dr. Alejandro Kiriakidis for his help, knowledge, support and supervision throughout this study. I would also like to express my appreciation to my Committee members, Dr. Bruce Kang and Dr. Eric Johnson, for reviewing this work.

Appreciation is extended to the USDOT/FHWA for giving me the opportunity to work on the project (project # DTFH61-01-C-00002). I would like to thank the staff of Constructed Facilities Center for their help throughout this work.

I would like to express my gratitude to Dr. Joseph B Morton of plant & soil science department and Mr. Alex Jalso of facilities and services for providing financial support for my graduate education.

A special note of appreciation goes to Mr. Jeong-Hoon Choi and Mr. Ryan Arnold for their help during this study. I would also like to thank my office mate Mr. Tee. A thank you also goes to Mr. Raghu Chandra Sankarayogi and Mr. Joseph Sweet for their kind help in writing this manuscript.

This work also constitutes the final report for Task 2.4 of USDOT/FHWA project# DTFH61-01-C-00002.

Table of Contents

List of Tables

List of Figures

Chapter1

Introduction

1.1 Introduction

A fiber-reinforced polymer (FRP) composite is a combination of a polymer or a plastic matrix such as polyester, vinyl ester or epoxy and a reinforcing agent such as glass, carbon or aramid. The primary functions of a matrix are to transfer stress between the reinforcing fibers, act as a glue to hold the fibers together, and protect the fibers from mechanical and environmental damage. The reinforcing agent carries load along the length of the fiber to provide strength and stiffness in one or more directions. Occasionally, fillers are also added to the composite to impart benefits like shrinkage control, surface smoothness, and crack resistance (ACMA, 2004). FRP materials are a good choice for bridge decks because of their strength and versatility. FRP composites have high strength and stiffness retention; they can be designed to provide a wide range of mechanical properties including tensile, flexural, impact and compressive strengths. Unlike traditional materials, FRP materials can have their strengths oriented to meet specific design requirements of an application. FRP composites have higher strength to weight ratios than most construction materials; this reduces the self weight and increases the live load capacity of FRP structures. Installation of FRP decks is rapid because of their modular construction approach and use of lighter erection equipment (Laosiriphong, 2004). The non-corrosive nature of FRP composites gives them the ability to successfully withstand the destructive effects of de-icing salts or saltwater spray from the ocean.

Apart from the advantages, failure analysis of FRP materials has been posing a variety of challenges to researchers. The reason is due to the complex nature of a composite structure and the fact that an FRP structure can fail in many different ways. A technique that can be used to non destructively evaluate complex materials is Acoustic Emission (AE) testing (McIntire, 1987). When the matrix cracks, the debonding between the fibers and matrix or fiber breakage can occur, and a resulting release of elastic energy occurs. This energy release is called Acoustic Emission. This energy propagates through the material in the form of stress waves; piezoelectric sensors can be used to detect these stress waves (Arnold, 2003). Structures such as FRP pressure vessels have demonstrated that AE is a viable method for determining the integrity of FRP composite structures (ASTM E 1067).

The AE parameters recorded during experiments can be correlated to the structural status such as the phases of failure and the damage locations. These correlations can be used along with some computer analysis to create a structural health monitoring system. One such computer analysis tool that can be used to help create the health monitoring system is Neural Network analysis (Arnold, 2003).

Neural Network (NN) analysis uses a complex set of equations and weighting functions to connect data to structural status. After the NN achieves the required mapping, the connection weights calculated can be used to predict status of other structures. This allows for a computer to analyze a data set and reach a conclusion without the aid of an operator, which can decrease the

chance of human error. Neural network programs have shown promise in many other applications such as pattern identification and the recognition of phases of failure (Mitchell, 1997).

It has been explained that AE method can be effective in predicting the phase of failure of FRP structures (Arnold, 2003). But, to develop a real-time structural health monitoring system for FRP structures, a data processing algorithm is necessary to group the data recorded by the sensors and convert the raw data into input matrix for the NN program. The NN program then analyses the data matrix and warns if the structure is approaching failure. It can also predict the ultimate failure load of the structure.

Existing methods use tools such as Microsoft Excel to extract the 10 row input matrix from raw AE data having more than 30,000 rows from a simple three point bending test. This process is not only tedious and time consuming but also is a big hurdle for the development of a real time structural health monitoring system. Efforts have been put forth through this study to solve the problems faced for the development of a real-time non-destructive evaluation technique for FRP materials. It is possible to predict the level and mode of failure of an FRP structure using Acoustic Emission and Neural Network techniques, but, there is a significant amount of work to be done on the AE data before it can be fed into the Neural Network program. A computer program has to be developed to extract the AE parameters such as count, energy, duration, and amplitude in a certain way to make connections to assess structural condition. The program must be easy to

use and simple to be modified in order to be called upon by the NN program or other programs in the future.

1.2 Objectives and Scope of the study

The objectives of this study are as follows:

- To develop a computer program which converts raw Acoustic Emission data into a Neural Network input matrix.
- To develop effective noise elimination algorithms.
- To confirm the proper functioning of the program by comparing the results with those obtained by using manual data conversion methods.

This report describes a detailed study of how to prepare the raw Acoustic Emission data for Neural Network analysis. Different approaches are discussed in order to extract a Neural Network input matrix from the raw data collected by LOCAN AT. Advantages of using awk programs and combining awk programs by using a C program are explained in detail.

This study includes some AE testing on FRP specimens. Two tension tests have been conducted to verify the noise elimination algorithms. Four bending tests were conducted on coupon specimens of a bridge deck structure to see how the specimen fails and how we can predict the failure of a complex structure. Two fatigue tests were also conducted so that the program developed through the study is modified to work with fatigue experiments as well.

This report is divided into 6 chapters. The first chapter is an introduction to the study. A literature review is presented in the second chapter. The third chapter is where all the algorithms written for the data conversion are explained

in detail and the comparison between manual and program based data conversion is shown graphically. All of the experiments conducted on FRP materials using Acoustic Emission testing are described in the fourth chapter along with results of the experiments. The fifth chapter describes the functioning of the back propagation Neural Network program and presents a comparison of NN predictions obtained between input matrices prepared manually and those prepared by using the program developed. The sixth chapter includes the conclusions reached from this study, applications of the program written during this study and recommendations for future research. The appendix includes the source codes developed in this project.

Chapter 2

Literature Review

2.1 FRP Composites and Acoustic Emission Techniques

2.1.1 FRP Composites

Fiber Reinforced Plastic (FRP) materials are being used for bridge construction and a lot of other applications. There are a lot of advantages in use of FRP structures; some of them are listed below:

1) Corrosion resistance
2) High strength to weight ratio
3) Ease of installation, because of their low self weight
4) Higher energy absorption

An FRP material consists of a fiber and a matrix made of resin. Most common types of fiber are glass fiber and carbon fiber. The fibers can be woven in many different ways to satisfy different loading situations, and the fibers are combined with resin to make a composite material.

The resin can be broadly classified into two groups, thermosets and thermo plastics. A thermoset is a kind of resin which, after curing, takes a final form and cannot be remolded into any other shape. The thermoplastics are manufactured at higher temperatures and they can be reheated and remolded. Most of the composite bridge decks used today are made of thermoset resins.

A lot of research has been done to study the failure of a composite material. The three modes of failure discovered in FRP materials are matrix cracking, matrix fiber debonding and fiber breakage.

Some of the disadvantages associated with FRP materials are as follows

1) High production cost
2) Complex failure mechanism
3) Lack of effective nondestructive testing methods
4) Non-decomposability

2.1.2 Acoustic Emission Techniques

Acoustic Emissions in materials can be described as transient elastic stress waves created by the rapid release of energy from localized sources. When a material is cracked or deformed, they emit this acoustic energy. The acoustic energy emitted from localized sources of the structure propagates through the structure and can be detected by acoustic sensors.

By analyzing the data collected by acoustic sensors, we can predict the failure of material. Acoustic Emission (AE) has been used as non-destructive testing method for a variety of materials. AE techniques have been used to detect debonding of reinforced bars in concrete (Hawkins et. al. 1988). AE has also been used to characterize hard wood from West Virginia (Chen et. al. 1992).

2.1.3 Usage of AE techniques for FRP composites

AE was first used to test FRP composite materials by testing fiber reinforced pressure vessels. The Society of Plastics Industry (SPI) developed an AE testing procedure to test glass-reinforced vessels (Arnold, 2003). The American Society for Testing Materials (ASTM) and the American Society of Mechanical Engineers (ASME) have also adopted similar procedures (Mc-Intire, 1987).

A common phenomenon shown by most materials is known as the Kaiser effect, named after a German AE researcher of 1950's. Because of the Kaiser effect, a material will only produce AE signals when loaded to a higher level than previous loads. However FRP materials do not display Kaiser Effect. In FRP materials, signals can be detected at all loading levels, even when the structure was previously loaded beyond the current stress level. This is known as Felicity effect (Arnold, 2003).

Beyond FRP pressure vessels, the use of AE in FRP structures is limited. AE evaluation of FRP composite specimens in tension and bending was conducted (Arnold, 2003). AE was also used to identify damage in stressed aramid FRP bars (Chen et. al. 1993). AE has also been used to identify fatigue failure modes in carbon fiber reinforced composite (Wevers et. al. 1991). AE along with acoustic waveguides has also been used to monitor FRP structures (Chen et. al. 1994) and the degree of curing and structural integrity of composite materials (Harrold & Sanjana 1986).

2.2 Neural Networks

A Neural Network (NN) program is based on an algorithm, which tries to mimic the working of a human brain. A NN program can process multiple pieces of information simultaneously to solve related problems all by itself.

The research on NN appears to have been started in 1800's, when an American Psychologist named James published a number of facts related to the human brain's structure and function. Among these were some of the basic

principles of correlational learning and associative memory (Eberhart et. al. 1996). The evolution of powerful personal computers sped up the development of Neural Networks.

Neural Networks can be referred to as information processing systems. The parametric input data along with the result to be predicted from the input is given to the Neural Network as a training set. The Neural Networks then make a bridge of weights between input parametric data and results. These weights are used to predict unknown results for other parametric inputs.

NN programs have been used previously in the prediction of damage levels and locations of damage zones for composite materials using Acoustic Emission data as parametric input (Arnold, 2003). NN programs have also been used along with acoustic wave guides to locate AE signals (Chen and Wassawapaisal, 2000) and for processing signals from hardwoods with various treatments (Chen and Chen, 1992).

2.3 Standards for AE testing techniques and FRP composites

ASTM standards were used to perform the tests in a reliable manner. Some of the ASTM standards for Acoustic Emission testing are as follows.

E 543: Standard Practice for Agencies Performing Nondestructive Testing

E 569:Standard Practice for Acoustic Emission Monitoring of Structures During Controlled Stimulation

E 650:Standard Guide for Mounting Piezoelectric Acoustic Emission Sensors

E 750:Standard Practice for Characterizing Acoustic Emission Instrumentation

E 976:Standard Guide for Determining the Reproducibility of Acoustic Emission Sensor Response

E 1067:Standard Practice for Acoustic Emission Examination of Fiberglass Reinforced Plastic Resin (FRP) Tanks/Vessels

E 1106:Standard Method for Primary Calibration of Acoustic Emission Sensors

E 1118:Standard Practice for Acoustic Emission Examination of Reinforced Thermosetting Resin Pipe (RTRP)

E 1211: Standard Practice for Leak Detection and Location Using Surface-Mounted Acoustic Emission Sensors

E 1316:Standard Terminology for Nondestructive Examinations

F 914:Standard Test Method for Acoustic Emission for Insulated Aerial Personnel Devices

The following are the ASTM standards relating to FRP materials and the testing methods used during this study:

D 638: Standard Test Method for Tensile Properties of Plastics

D 790: Standard Test Methods for Flexural Properties of Unreinforced and Reinforced Plastics and Electrical Insulating Materials.

D 883: Standard Terminology Relating to Plastics

D 3039: Standard Test Method for Tensile Properties of Polymer Matrix Composite Materials

E 4: Standard Practices for Force Verification of Testing Machines

2.4 Cygwin on Windows and awk interpreter

2.4.1 Overview of cygwin

Cygwin simulates Linux like environment for windows. A DLL file namely cygwin1.dll, acts as the emulation layer, which provides a substantial POSIX (Portable Operating System Interface) system call functionality. Cygwin also provides a collection of tools, which provide a Linux look and feel. The Cygwin DLL works with all versions of Windows except WindowsCE.

By installing Cygwin, we can have access to most of the UNIX utilities. The most important and useful tool we used for this research is the awk. The awk programming language is designed to search for, match patterns, and perform actions on text files. Apart from using awk we also used cygwin environment to use the gcc compiler to compile the C program.

Installation and usage of cygwin is fairly easy for conventional windows users. The installable DLL can be downloaded from www.cygwin.com. After installation, a directory (which is named after the system username) is created under cygwin home directory in which all the awk and C scripts are to be stored. After compilation of any C source codes, the executable file is generated in this folder. The user must grab that executable file and put it in *bin* folder under cygwin to make that executable as a shell command in cygwin window.

2.4.2 Introduction to awk

awk is best described as "*A Pattern Scanning and Processing Language*" (title of an article written by the three authors of the language). The name for the language comes from the initials of its designers namely Alfred V. Aho, Peter J. Weinberger, and Brian W. Kernighan (Dale Dougherty and Arnold Robbins, 1997).

awk's programming model gives the ability to transform structured data into a formatted report with minimum coding efforts. Apart from awk's capabilities to view a text file as a textual database made up of records and fields, it has numerous other advantages over other programming languages.

In this study, the awk utility has been efficiently used along with C program to extract necessary values from a large data file created by our data acquisition system (LOCAN-AT), and to format and prepare information for other programs.

The ability to call awk scripts at the command line without having to compile them before has given us the ability to combine more than one simple awk script using the C compiler.

One ability of awk's programming model which makes it so powerful is its ability to execute a set of instructions line by line throughout a text file without having the user specify the loop conditions.

Chapter3

Data Conversion Program

3.1 Need for the program

Previous researchers who worked with Acoustic Emission testing on FRP materials have proven that Neural Network (NN) analysis on the data collected from LOCAN AT can be used to predict the damage level of FRP material. To do that, they have developed a method for preparing the data from LOCAN AT and convert it to an input matrix for the NN program (Arnold, 2003).

The amount of data collected from LOCAN AT is huge, but the NN program needs an input matrix which is conveniently small, yet effectively represents all the AE parameters collected during the testing of specimen.

The raw data collected from LOCAN AT includes all the noise that is picked up by the AE sensors during experiments. These noises are also to be eliminated during the process of preparation of data for NN analysis. Noise that is picked up from the sensors can be from different sources such as the hydraulic pump (operational noise) and noise generated by the loading arm.

Most of the noises are eliminated during the careful inspection of the signals. One method that has been used to eliminate the noise is by eliminating signals which satisfy a certain criterion. For example, signals which have amplitude value less than a preset threshold values are considered as noise. A lower and upper limit for the frequency range of a signal is set, and every signal which falls outside this frequency range is considered noise.

In this study, in addition to the signal filtration techniques explained above, we developed another method to completely eliminate the noise caused by the loading arms. This method needs one sensor to be attached to each of the loading arms. The sensor on the loading arm picks the noise wave prior to the material sensor. We have proven by a series of pencil break tests that we can easily isolate the signals created by loading arms from the actual material AE signals.

After elimination of noise, the next step for creation of input matrix for NN program is called “Method of Summation” (Arnold, 2003). Method of summation is dividing the whole data file given by LOCAN AT into 40 rows, by loading. That is, dividing the load into 0 to 100%, in steps of 2.5% of the loading percentage. All the other AE parameters are averaged for each loading section.

After the process of elimination of noise and summation of data into 40 rows for each sensor, the data has to be normalized. Normalization of data is done to make data from each test equally influential. In other words, normalization of data ensures more accurate prediction of damage zone by NN program. For each experiment, the maximum and minimum values for each AE parameter were identified. The minimum parameter value is subtracted from each parameter and then divided by the difference of maximum and minimum value of parameter. The method is explained by equation below.

Normalized Count = (Count–minimum Count)/(maximum Count–minimum Count)

All the above-mentioned computations should be done for each sensor used in each experiment in order to prepare an input matrix for NN analysis. Tools like MS Excel were used previously to perform these computations. It is not only tedious and time consuming to use tools like Excel, but we also needed an automated way of doing this in order to develop a real-time structural health monitoring system. Therefore, we need one single program that can take a LOCAN AT data file, do all the things explained above and output the NN input matrix. In this study, we have developed a C program which calls a number of awk scripts and accomplishes all of these tasks.

3.2 Advantages of using awk

awk was first released with version 7 UNIX, in 1978. It was a language developed for system administrators to transform data into formatted report. It allowed users to view a text file as a textual database made up of records and fields.

awk considers each line in a input file as a record and each number separated by tab as a field. An awk program consists of a “main input loop” (Dougherty and Robbins, 1997). The main input loop consists of a set of instructions which will be executed for each input line from a file. This is the most important advantage of awk programming language over other languages. In

other languages, we have to open the input file and read one line at a time to process a set of instructions.

awk also has BEGIN and END procedures where we can write code. The code in the BEGIN procedure is executed before any input is read and the code in END procedure is executed after all input is read.

We use BEGIN typically to initialize some parameters. In the future, after sufficient research is done to standardize all the parameters needed to eliminate noise and calculate loads, these parameters should be defined in the BEGIN procedure of the awk scripts.

The END procedure is efficiently used in one of our scripts, namely Matrix.awk. We have used the procedure to write code to transform all the rows into columns and columns into rows. The code will be explained in detail in Sections 3.3.13, 3.3.14 and 3.3.21.

Apart from all above described advantages of using awk over other languages, the most important reason we chose awk is that we needed a very simple programming model to do the data transformation. We need a programming language which can be easily understood by future researchers. awk is much easier to learn than many other programming languages because it offers a well-defined and useful model to the programmer.

3.3 Explanation of awk scripts

3.3.1 Explanation of bending.awk

In bending experiments, we usually measure deflection of the material also for further analysis. But, for the Neural Network analysis we do not need the deflection values in the input matrix.

If the experiment is static bending, then we have to remove the deflection from the output file of the LOCAN AT. For tension or fatigue, we never used the deflection parameter while doing the experiment with LOCAN AT.

When the parameters are given to the program at command prompt, when the command prompt says

"Press b if Bending, press t if Tension and press f if fatigue:"

The user has to type b (or B) if they want the program to run bending.awk, else say t or f. Running the bending.awk will eliminate the deflection column from the input file.

The input file for this program should have the following fields separated by tab: Time, Load, Deflection, Channel number, Rise, Count, Energy, Duration, Amplitude and Average frequency

The output of the program will have the following fields separated by tab: Time, Load, Channel number, Rise, Count, Energy, Duration, Amplitude and Average frequency. The flow chart for this program is shown in Figure 3.1.

3.3.2 Step-by-step algorithm of bending.awk

Step1 Start

Step2 Check to see if the system has reached end of file. That is, check to see if the algorithm is run on all the rows of the input file. If yes, go to Step 4.

Step3 Output all columns in the input file into the output file, except the one column for deflection. Go to Step2.

Step4 Finish

3.3.3 Explanation of signalnumbering.awk

As explained in earlier chapters, one of the problems we have in signal analysis of LOCAN AT data is noise elimination. To eliminate the noise from the loading arms, we installed one sensor on each loading arm. The time taken by a signal to travel from the sensor on the loading arm to the nearest sensor on the testing specimen is measured using a simple pencil break test.

The logic behind the signal-numbering algorithm is as follows: if a signal reaches sensor-A first and then reaches sensor-B, then check the time taken by the signal to reach sensor-B after reaching sensor-A. If the time obtained from the pencil break test is greater than the time taken by the signal to travel from A to B, then both the records for sensor A and sensor B are numbered as single signal.

The input to this program should have the following fields separated by tab: Time, Load, Channel number, Rise, Count, Energy, Duration, Amplitude and Average frequency

The output from the program will have the following fields separated by tab: Time, Load, Channel number, Rise, Count, Energy, Duration, Amplitude and Average frequency, Signal number. The flow chart for this program is shown in Figures 3.2a and 3.2b

3.3.4 Step-by-Step Explanation of signalnumbering.awk

Step1 Start.

Step2 Initialize signal number to 1.

Step3 Check to see if the program reached end of file. If yes, go to Step9.

Step4 If the row is first row of input file, take the time parameter from the input file.

Step5 If the program is in the second row of input file, take the beginning time of the experiment from the row.

Step6 Set variable diff=(Time of the signal – Time from Step5).

Step7 Check to see if the diff calculated in Step6 is less than or equal to the time variable from Step4. If yes then signal number=signal number+1.

Step8 Output all data along with signal number, go to next row, then go back to Step3.

Step9 Finish.

3.3.5 Explanation of Eliminatenoise.awk

Once the signals are numbered, then it is easy to eliminate the noise. The logic is simple but very efficient in eliminating the noise from the loading arm. We take the channel numbers of the noise sensors which are installed on loading arms. In the input file, if the signal is from noise sensor, then the signal number is

stored into a variable. For every data signal, the signal number is checked with the variable that has the previous noise sensor's signal number. If the signal number of data signal matches with the variable, then the signal is considered as noise. If a signal reaches the noise sensor before it reaches the data sensor, then the signal is probably coming from the loading arm and this program along with the signal numbering program ensures that this kind of noise is not analyzed by the Neural Network program.

The input for this program will have the following fields separated by tab: Time, Load, Channel number, Rise, Count, Energy, Duration, Amplitude and Average frequency, Signal number.

The output from the program will have the following fields separated by tab: Time, Load, Channel number, Rise, Count, Energy, Duration, Amplitude and Average frequency. The flow chart for this program is shown in Figures 3.3a and 3.3b

3.3.6 Step-by-Step Explanation of Eliminatenoise.awk

Step1 Start.

Step2 Initialize variable progSignalnumber to 0.

Step3 Check to see if the program reached end of file. If yes go to Step8.

Step4 Check to see if the row is first row of the input file. If yes take the numbers of noise channels from the input file

Step5 Check to see if the channel number is one of the noise channels from Step4. If yes, set variable progSignalnumber to signal number of the current signal and go to next row in the input file and repeat this step.

Step6 Check to see if the signal number of the current signal in input file is greater than the number in the variable progSignalnumber. If yes, Output all the parameters for this signal except the signal number.

Step7 Propagate to the next row in the input file and go to Step3.

Step8 Finish.

3.3.7 Explanation of code1.awk

This program is the first step in grouping the data together for Neural Network analysis. The typical raw data file from a bending test that we receive from the LOCAN AT has about 30,000 rows with 10 columns. By the time the raw data is churned completely and made into the input matrix for Neural Network analysis, each experiment with two sensors has 40 rows with 6 columns each.

The task is to correctly represent all the 30,000 rows of data in 40 rows. To do that, we first eliminate all the noise from the loading arms by running through the above programs. We also get rid of some signals which do not represent the material failure. Finally, we group the data into averages and percentages of every 2.5% of loading percent from 0 to 100.

What the code1.awk program first does is. It converts the load values in input file, which might be numbers indicating voltage, into load values by using the multiplier and offset values given by the user. Then, it eliminates all the signals that are not considered to be material failure signals. This elimination is done by eliminating all the signals whose amplitude values are less than the

minimum threshold value given by the user. For the Bedford composites we usually use a minimum threshold value of 60dB. After that, the program extracts data for a given channel number and calculates the load percentage from the load in the input file and the average failure load given by the user. In addition to all the above things the program does, one important manipulation that code1 does to the data is that it eliminates the signals recorded after failure and during any experimental errors. That is, if the specimen has failed, the LOCAN AT is still collecting data even though load is dropping fast. Looking at the dropping load, these unwanted signals are eliminated. Since Code1.awk and all the following scripts are run separately for each channel, the output does not have channel numbers.

The input for this program should have the following fields separated by tab: Time, Load, Channel number, Rise, Count, Energy, Duration, Amplitude and Average frequency.

The output from the program will have the following fields separated by tab: Load%, Rise, Count, Energy, Duration, Amplitude and Average frequency. The flow chart for this program is shown in Figures 3.4a, 3.4b and 3.4c

3.3.8 Step-by-Step algorithm of Code1.awk

Step1 Start.

Step2 Initialize the variable Previous load to 0

Step3 Check to see if the program has reached end of file of input file. If yes, go to Step11.

Step4 Check to see if the row of the input file is first row. If yes, then take the program variables from first row of the input file. The input variables are Multiplier and Intercept for calculation of load values, lower threshold value for elimination of non-material failure signals from data, average failure load to calculate the load percentage, and channel number for which the program is run.

Step5 Determine whether the input row's amplitude value is greater than the given Lower threshold value. If not, go to next row in input file and go back to Step3.

Step6 Check and see if the channel number of the current row in the input file is equal to the channel number variable from Step4. If not, go to next row in input file and go back to Step3.

Step7 Convert the voltage in the input file to load by using the formula:

Load=(load from input file * Multiplier from Step4) + Intercept.

Step8 Convert load into load% by using the formula

Load%=(Load from Step7 * 100)/Average Fail load from Step4

Step9 Check to see if the Load% drops more than 2.5%. If yes, then either the material already failed or it can be an experimental error. So, if the condition satisfies, then go to next row in input file and go back to Step3.

Step10 Output the load% and all the other important AE parameters to the output file. Go to next row in the input file and go back to Step3.

Step11 Finish.

3.3.9 Explanation of Code2.awk

Code2 finishes the grouping of the data into multiples of 2.5% of loading from 0% to 100%, which makes the 40-row matrix of data. All of the important AE parameters for Neural Network analysis are averaged into sets of 2.5% of the loading parameter. Therefore, if there are 20 signals in the range of 2.5 to 5 of load%, then one record will be outputted into the output file having 2.5 as load% and average of all other AE parameters after this program is run.

This program also filters some signals based on the watch amplitude and the lower and upper frequency levels given by the user. So, if the amplitude of the signal from the input file is greater than the user defined watch amplitude and the average frequency calculated is between the lower and upper frequencies given by user, then the signal is called a hit. If a signal meets such a criterion, then the hit is incremented and other AE parameter values, namely count, energy, duration and amplitude, are added to the previous value.

Apart from all the above things, this program also performs one more important function; that is, if the load% jumps two or more steps from one row to next, then the program puts zeros in all columns in place of all missing multiples of 2.5.

After execution of this program, the data is already reduced to 40 rows and 6 columns. The two things left after this program are normalization and matrix arrangement.

The input for this program should have the following fields separated by tab: Load%, Rise, Count, Energy, Duration, Amplitude and Average frequency.

The output from the program will have the following fields separated by tab: Load%, Average count, Average energy, Average duration, Average amplitude and Average frequency. The flow chart for this program is shown in Figures 3.5a, 3.5b and 3.5c.

3.3.10 Step-by-Step algorithm of Code2.awk

Step1 Start.

Step2 Initialize variable m to 1 and variable n to 0.

Step3 Check to see if end of file is reached. If yes, go to Step23.

Step4 Identify whether the current row is first row of input file. If yes, take the variables from the input file for watch amplitude, lower frequency and upper frequency.

Step5 Check to see if the row is the second row of the input file. If not, go to Step8.

Step6 Is load in the input file greater than m*2.5? If not, go to Step8.

Step7 Output m*2.5 as load% in the output file and 0's for all other AE parameters. Increment m by 1 and go back to Step6.

Step8 Initialize variable bool to 1.

Step9 Check and see if load% in input file is less than or equal to m*2.5. If not, go to Step14.

Step10 Check to see if bool equal to 1. If not, go to Step13.

Step11 Verify if input amplitude is greater than watch amplitude, and average frequency is in between the lower and upper frequencies given by the user. If yes go to Step13.

Step12 Set Hit=Hit+1, Count=Count + Count from input, Energy=Energy + Energy from input, Duration=Duration + Duration from input and Amplitude=Amplitude + Amplitude from input.

Step13 set n=n+1 and bool=1. Go to the next row in the input file and go back to Step9.

Step14 Check to see if n equal to 0. If yes, then set n=1.

Step15 Calculate the averages. Count=Count/n, Energy=Energy/n, Duration=Duration/n, Amplitude=Amplitude/n.

Step16 Check to see if Load equal to printing load. If not, go to Step18.

Step17 Print m*2.5 and all 0's for AE parameters. Go to the next row in the input file and go back to Step3.

Step18 Check to see if the row is the second row of the input file. If yes, go to Step21.

Step19 Initiate printing load to the load from the input file.

Step20 Print load%, average count, average energy, average duration, average amplitude, average hit values into the output file.

Step21 Check to see if the input amplitude is greater than the watch amplitude and average frequency is in the range set by user. If yes, set Hit=1.

Step22 Increment m by 1 and set n and bool to 0. Go to next row in the input file and go back to Step9.

Step23 Finish.

3.3.11 Explanation of Mamin.awk

This program gets the maximum and minimum values of AE parameters recorded in the experiment. The next program uses the values that are outputted in this program as variables for normalization of data. The Neural Network analysis program requires all the matrix elements to be in the range 0 to 1. Therefore, normalization of the data has to be done before the matrix is given for NN analysis.

The program first initializes all the minimum variables, like minimum count, minimum energy, etc, to 100. The initialization is necessary in this case because awk interpreter by default initializes all variables to 0, and if we check with the default initialization of 0, we will not get the minimum of the data collected. Therefore, the initialization of minimum variables is done to 100 and then the program collects the minimum value from the experiment and sends it to an output file (mamin.txt). This file is used to supply parameters for the next program.

The input for this program should have the following fields separated by tab: Load%, Average count, Average energy, Average duration, Average amplitude and Average frequency.

The program does not change anything in the input file but, creates an output file called mamin.txt and stores the following values taken from the input file: maximum count, minimum count, maximum energy, minimum energy, maximum duration, minimum duration, maximum amplitude, minimum amplitude,

maximum hit and minimum hit. The flow chart for this program is shown in Figures 3.6a, 3.6b, 3.6c and 3.6d.

3.3.12 Step-by-Step algorithm of Mamin.awk

Step1 Start.

Step2 Check to see if the program reached the end of file. If yes go to Step25.

Step3 Is count from input file is greater than the variable maximum count. If not, go to Step5.

Step4 Assign count from input file to variable maximum count.

Step5 Is energy from input file is greater than the variable maximum energy? If not, go to Step7.

Step6 Assign energy value from input file to variable maximum energy.

Step7 Is duration from input file is greater than the variable maximum duration. If not, go to Step9.

Step8 Assign duration value from input file to variable maximum duration.

Step9 Is amplitude from input file is greater than the variable maximum amplitude? If not, go to Step11.

Step10 Assign amplitude value from input file to variable maximum amplitude.

Step11 Is hit from input file is greater than the variable maximum hit? If not, go to Step13.

Step12 Assign hit value from input file to variable maximum hit.

Step13 Check to see if count from input file is not equal to 0 and it is less than the variable minimum count. If not, go to Step15.

Step14 Assign count from input file to variable minimum count.

Step15 Check to see if energy from input file is not equal to 0 and it is less than the variable minimum energy. If not, go to Step15.

Step16 Assign energy from input file to variable minimum energy.

Step17 Check to see if duration from input file is not equal to 0 and it is less than the variable minimum duration. If not, go to Step15.

Step18 Assign duration from input file to variable minimum duration.

Step19 Check to see if amplitude from input file is not equal to 0 and it is less than the variable minimum amplitude. If no, go to Step15.

Step20 Assign amplitude from input file to variable minimum amplitude.

Step21 Check to see if hit from input file is not equal to 0 and it is less than the variable minimum hit. If not, go to Step15.

Step22 Assign hit from input file to variable minimum hit.

Step23 Go to next row in input file and go to Step2.

Step24 Output all the variables, namely maximum count, minimum count, maximum energy, minimum energy, maximum duration, minimum duration, maximum amplitude, minimum amplitude, maximum hit and minimum hit into the file mamin.txt.

Step25 Finish.

3.3.13 Explanation of Matrix.awk

Before giving the data as matrix to the Neural Network program, the data has to be normalized. By normalizing what we mean is, the data should be between 0 and 1 and still represent the AE parameters correctly. So, the formula

used is, AE matrix parameter = (AE parameter – Minimum parameter value in the file) / (Maximum parameter value in file – Minimum parameter value in file).

The formula mentioned above is used for all the five AE parameters being prepared for the Neural Network analysis. The maximum and minimum parameter values are already stored in a file by using the mamin.awk program, which was explained in previous sections.

After normalizing the data in the input file, the columns in the input file have to be changed to rows and rows to columns in order to be read by the NN program; therefore Matrix.awk also performs this operation in addition to the normalization of the data.

The input for the program will have the following fields separated by tab: Load%, Average count, Average energy, Average duration, Average amplitude and Average frequency. This program also takes the maximum and minimum vales of AE parameters extracted by program Mamin.awk as input.

The program has to be run once for each sensor. The program output is a file containing 4 matrices, each separated by a blank row. Each matrix has 5 rows and 10 columns. For each matrix the rows are the different AE parameters, the parameters from first row to fifth row are count, energy, duration, amplitude and hit, respectively. The flow chart for this program is shown in Figures 3.7a, 3.7b, 3.7c, 3.7d and 3.7e.

The columns in the matrix represent load percentage values. Each matrix has 10 load percentages. Combining all the matrices, we have 4x10=40 load percentage values from 2.5% to 100%. The first row, first column of first matrix

has the normalized value of count parameter collected from the experiment at 2.5% of loading. Table 3.1 and Table 3.2, show the columns and rows in the input matrices more clearly. The AE parameters Count, Energy, Duration, Amplitude and Hit are all normalized. There are four matrices like those shown in Tables 3.1 and 3.2, to 100% of loading percentage for each channel.

3.3.14 Step-by-Step algorithm of Matrix.awk

Step1 Start.

Step2 Check to see if the end of file of the input file has reached. If yes, go to Step21.

Step3 Is this first row of the input file? If not, go to Step5.

Step4 Get the variables for the program from this row. The variables are Maximum count, Minimum count, Maximum energy, Minimum energy, Maximum duration, Minimum duration, Maximum amplitude, Minimum amplitude, Maximum hit, Minimum hit and the file name given by user to store the matrix.

Step5 Set array variable a[row number, 1] = (count value from input file – Minimum count) / (Maximum count – Minimum count)

Step6 Check to see if a[row number, 1] is less than 0. If yes, set a[row number, 1] to 0.

Step7 If a[row number, 1] is greater than 1 then set a[row number, 1] to 1.

Step8 Set array variable a[row number, 2] = (energy value from input file – Minimum energy) / (Maximum energy – Minimum energy).

Step9 Check to see if a[row number, 2] is less than 0. If yes, set a[row number, 2] to 0.

Step10 If a[row number, 2] is greater than 1 then set a[row number, 2] to 1.

Step11 Set array variable a[row number, 3] = (duration value from input file – Minimum duration) / (Maximum duration – Minimum duration).

Step12 Check if a[row number, 3] is less than 0. If yes, set a[row number, 3] to 0.

Step13 If a[row number, 3] is greater than 1, then set a[row number, 3] to 1.

Step14 Set array variable a[row number, 4] = (amplitude value from input file – Minimum amplitude) / (Maximum amplitude – Minimum amplitude).

Step15 Check to see if a[row number, 4] is less than 0. If yes, set a[row number, 4] to 0.

Step16 If a[row number, 4] is greater than 1, then set a[row number, 4] to 1.

Step17 Set array variable a[row number, 5] = (hit value from input file – Minimum hit) / (Maximum hit – Minimum hit).

Step18 Check to see if a[row number, 5] is less than 0. If yes, set a[row number, 5] to 0.

Step19 If a[row number, 5] is greater than 1, then set a[row number, 5] to 1.

Step20 Go to next row in the input file and go to Step2.

Step21 Initiate *I* to 2, *j* to 1, *k* to 11 and variable *swap* to 2.

Step22 Is k less than 42? If not, then go to Step28.

Step23 Is j less than 6? If not, then go to Step27.

Step24 Is I less than or equal to k? if not, then go to Step26.

Step25 Print array variable a[I,j] into the file name given by user and increment I by 1.

Step26 Set I=swap and j=j+1.

Step27 Set j=1, I=k+1, k=k+10 and swap=I.

Step28 Finish.

3.4 Combining awk scripts using a C program

All the awk scripts can perform the data transformation from LOCAN AT's output to NN input matrix, but the user should call these awk scripts one after another at UNIX command line. Also, the experimental parameters needed to be supplied at command line for each script.

To develop a real-time structural health monitoring system, we need to have a compiler code which can call these awk scripts one after the other, supply the variable to the awk scripts and get the output matrix, without the user having to type the command line syntax to call each awk script.

The two different options we have in order to achieve this are: (1) to write a UNIX shell script, which would call the awk scripts and supply variables also to the scripts, or (2) to develop a C program which would do the same by using the *system()* function available in gcc compiler. We chose to develop a C program. A UNIX shell script though easy to develop, might present a problem in the future. C programming language is much more popular and more researchers feel comfortable with a C program than UNIX shell script.

Therefore, a C program was developed to take inputs from the user, call the appropriate awk scripts, give variables to scripts, and perform relevant operations on the data collected from the experiment based on the user's input.

Using this program, the raw data collected from experiments can be converted into Neural Network input instantaneously. The user is releaved from processing knowledge regarding the syntax for calling upon the awk scripts.

This executable file generated after compilation of the C program should be executed in UNIX shell environment (Cygwin for windows systems). A screen capture of runtime environment is shown in figure 3.9. The output files were already explained in detail in Section 3.3.13.

3.5 Explanation of C program

3.5.1 Explanation of dataconvert.c

This C program is used as a connector between all the independent awk scripts explained in the Section 3.3. The program opens the input file specified by the user, takes parameter inputs from users, calls appropriate awk scripts and stores the output given by the Matrix.awk in a file named by the user.

This program relieves the user from executing each of the 5 awk scripts, one after the other in the correct order. The program also takes care of variables that are to be supplied to the awk scripts at the command line.

In the future, this program can be modified to be the main module for real time structural health monitoring applications. If all the experimental parameters are standardized for a real-time situation, then the process of user input, of the

parameters can be eliminated. The flow chart for this program is shown in Figures 3.8a and 3.8b.

3.5.2 Step-by-step algorithm of dataconvert.c

Step1 Start.

Step2 Take input file name and ask what type of experiment this is.

Step3 Is the test bending? If no, go to Step5.

Step4 Execute Bending.awk on input file.

Step5 Ask if noise sensors were installed. If no, go to Step10.

Step6 Take time taken for acoustic wave from noise sensor to nearest material sensor.

Step7 Execute Signalnumbering.awk on input file.

Step8 Take the channel numbers of noise sensors.

Step9 Execute Noiseelimination.awk on input file.

Step10 Take caliberation parameters based on result from Step3.

Step11 Take number of channels used, then store in i.

Step12 Is i>=1? If no, go to Step20.

Step13 Take channel number into C.

Step14 Execute code1.awk for channel C.

Step15 Execute code2.awk for channel C.

Step16 Execute Mamin.awk for channel C.

Step17 Take output file name to store matrix for channel C.

Step18 Execute Matrix.awk for channel C and store output in filename given in Step17.

Step19 Set i=i-1 and go to Step12

Step20 Finish.

3.6 Compiling the C Program

The awk scripts will be interpreted when the application is executed every time. Therefore it is not necessary to compile the C program if changes are made to the awk scripts. However, if the C program is modified, the program has to be recompiled.

The C source file is stored in the directory shown below:

<cygwin directory>\home\<system username>

(For example): c:\cygwin\home\avinash

To compile the source file, user has to open the cygwin shell prompt and type *gcc dataconvert.c*.

After successfully compiling the modified source code, cygwin creates an executable file named a.exe in the same directory shown above. In order to make the exe a shell command, user has to copy the a.exe, rename it to dataconvert.exe and place it in the *bin* folder under cygwin directory. After all the above steps are complete, the command *dataconvert* in the cygwin shell prompt will start the application.

3.7 Results

3.7.1 Testing Procedure

To verify the performance of the code developed, the code is used to calculate the average AE parameters from raw data collected from ten bending experiments and six tension experiments. The average is computed and then plotted over manually calculated values. The plots shown in Figures 3.11 to 3.90

prove that the program developed through this study can do what used to be done manually.

3.7.2 Data Extraction for Plots

Since the final output of the program gives normalized AE values, we have to take data from the output of code2.awk In order to extract the average AE parameter values for every 2.5% change of loading from 2.5% to 100%. The output from code2.awk is stored in awk4.txt in the home directory under Cygwin installation. But, since code2.awk is executed once for each channel, the data from awk4.txt should be copied into a separate file before inputting the second channel number as the program is being executed.

The experimental parameters provided to the program during the extract process shown below in tables 3.3 and 3.4

3.7.3 Manual calculation vs. Program Calculation

The average values calculated by using the program are in good agreement with those calculated manually, except at the beginning of the loading and when the specimen is approaching failure. The comparison for test B1 and 1t is shown graphically in Figures 3.11 to 3.20. The comparison for all the other tests is shown in Appendix B. The reason for the mismatch at the initial stage of loading can be explained by the fact that the specimen emits a smaller number of AE signals at the beginning stages of loading. This can be further explained in detail by looking at data from first channel for the first bending experiment shown in Table 3.3, along with Figure 3.13 which plots average duration vs. loading.

As it can be seen in Figure 3.13, the manually calculated average duration shows 46.58 μs while the program calculates a value of 110.5 μs. The reason for the difference between the readings is while doing manually, the average duration was calculated by averaging all values from 19.48% to 24.25% of loading, where as the program calculated average duration by averaging just two values at 19.48% and 19.78%, since these are the only two signals between 17.5% and 20% load.

3.8 Summary

- awk programs were developed to automate the computations done on raw data collected from LOCAN AT.
- Noise elimination algorithms were developed in order to eliminate the signals generated by loading arms.
- Seven individual awk scripts should be run before the raw data transforms into 4 sets of 5 X 10 matrices, which will be the input to Neural Network program.
- A C program was developed to automate the process of calling awk scripts in sequence and obtaining the experimental parameters from user.
- The precision of the application was established by plotting the average values computed by the application over manually calculated values (Refer Figures 3.11 to 3.90).

Load%	2.50%	5%	7.50%	10%	12.50%	15%	17.50%	20%	22.50%	25%
Count										
Energy										
Duration										
Amplitude										
Hit										

Table 3.1 First input matrix for Neural Network program

Load%	27.50%	30%	32.50%	35%	37.50%	40%	42.50%	45%	47.50%	50%
Count										
Energy										
Duration										
Amplitude										
Hit										

Table 3.2 Second input matrix for Neural Network program

Load %	Channel	Counts	Energy	Duration	Amplitude	Ave. Frequency
17.39	1	13	8	34	65	382
19.48	1	7	7	16	61	437
19.78	1	17	12	205	61	82
20.97	1	13	7	33	63	393
21.57	1	20	19	52	70	384
21.86	1	19	5	48	60	395
22.16	1	11	6	37	60	297
22.76	1	3	4	7	62	428
23.65	1	7	6	17	60	411
23.35	1	8	5	20	61	400
23.95	1	12	5	48	62	250
23.95	1	10	9	45	62	222
24.25	1	10	9	31	62	322

Table 3.3 Part of data from channel 1 for B1

Load Calibration Parameters	
Multiplier	-100
Offset	3.31

Noise Elimination Parameters	
Threshold	60dB
Watch Area Amplitude	70dB
Lower Limit Frequency	100kHz
Upper Limit Frequency	400kHz

Failure Loads	
B1	335.31 lbs
B2	347.31 lbs
B3	315.31 lbs
B4	345.31 lbs
B5	333.31 lbs
B6	270.31 lbs
B7	333.31 lbs
B8	314.31 lbs
B9	345.31 lbs
B10	298.31 lbs

Table 3.4 Bending Test Parameters

Load Calibration Parameters	
Multiplier	2.25
Offset	-0.03

Noise Elimination Parameters	
Threshold	60dB
Watch Area Amplitude	70dB
Lower Limit Frequency	100kHz
Upper Limit Frequency	400kHz

Failure Loads	
T1	8.0475 kips
T2	5.325 kips
T3	5.325 kips
T4	5.0775 kips
T5	5.2575 kips
T6	5.305 kips

Table 3.5 Tension Test Parameters

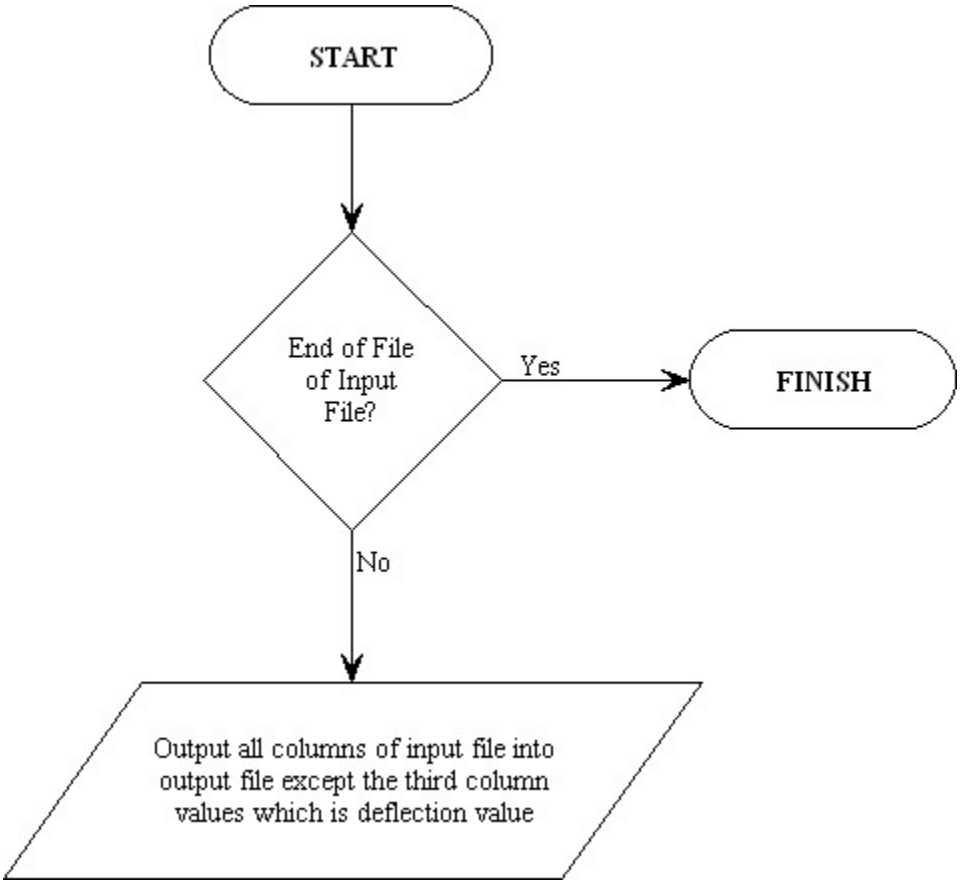


Figure 3.1 Flowchart of bending.awk

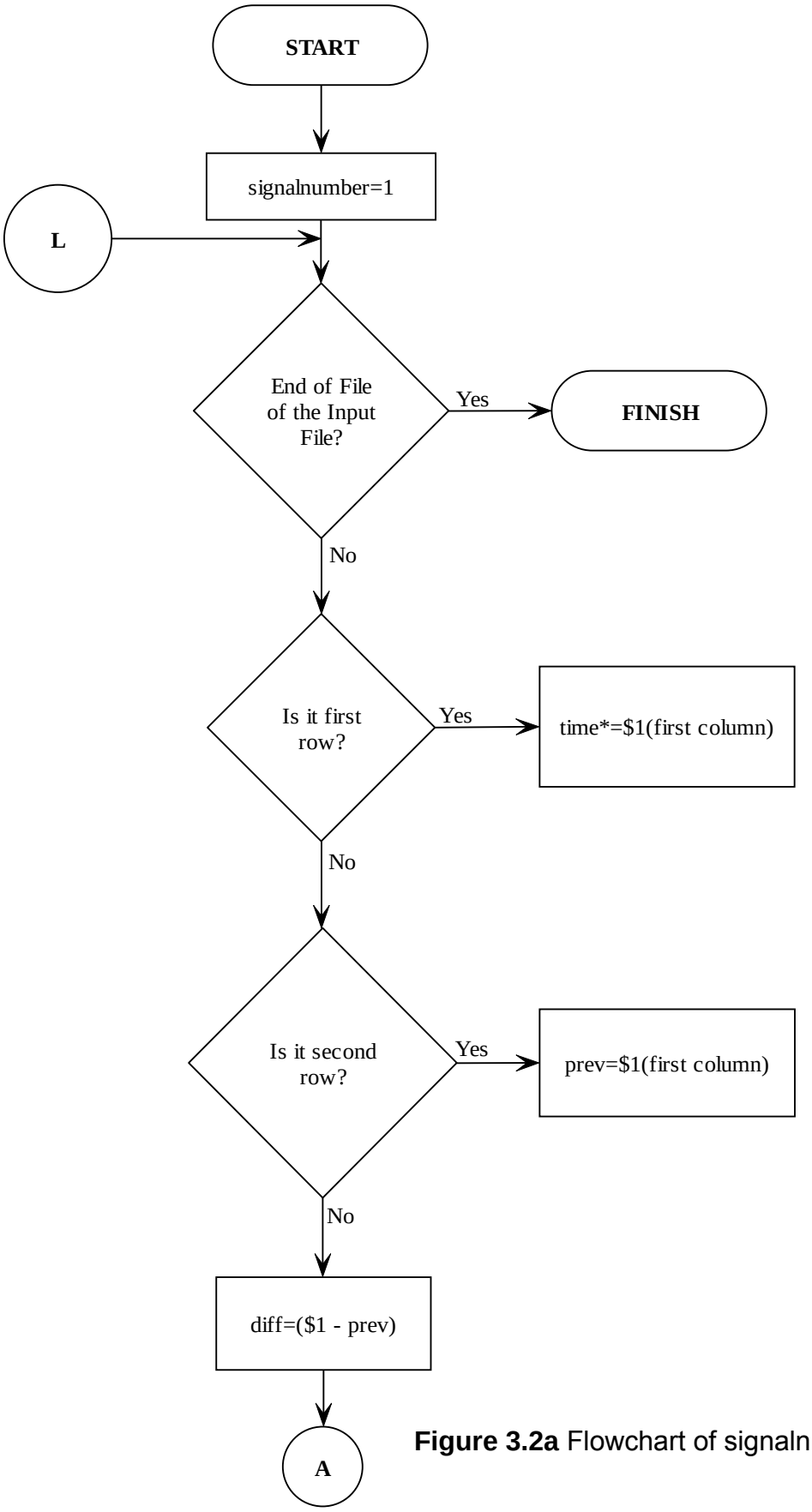


Figure 3.2a Flowchart of signalnumbering.awk

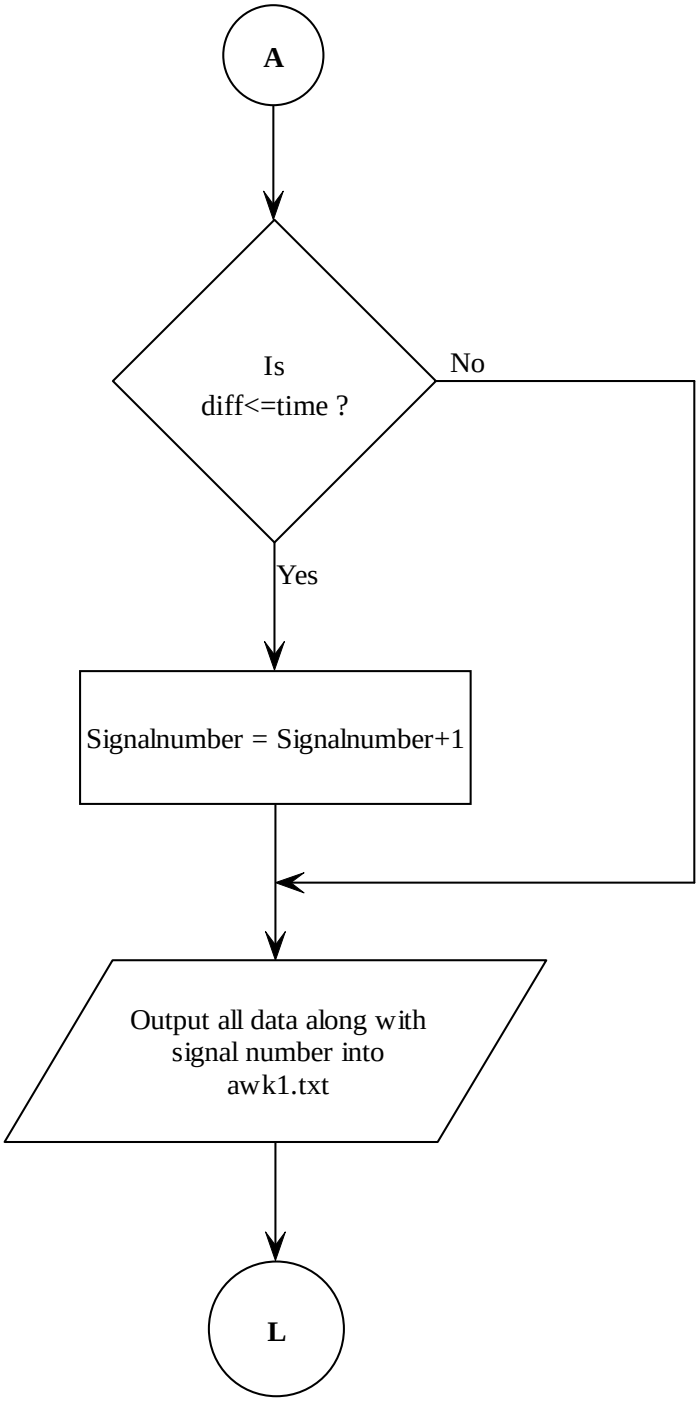


Figure 3.2b Flowchart of signalnumbering.awk

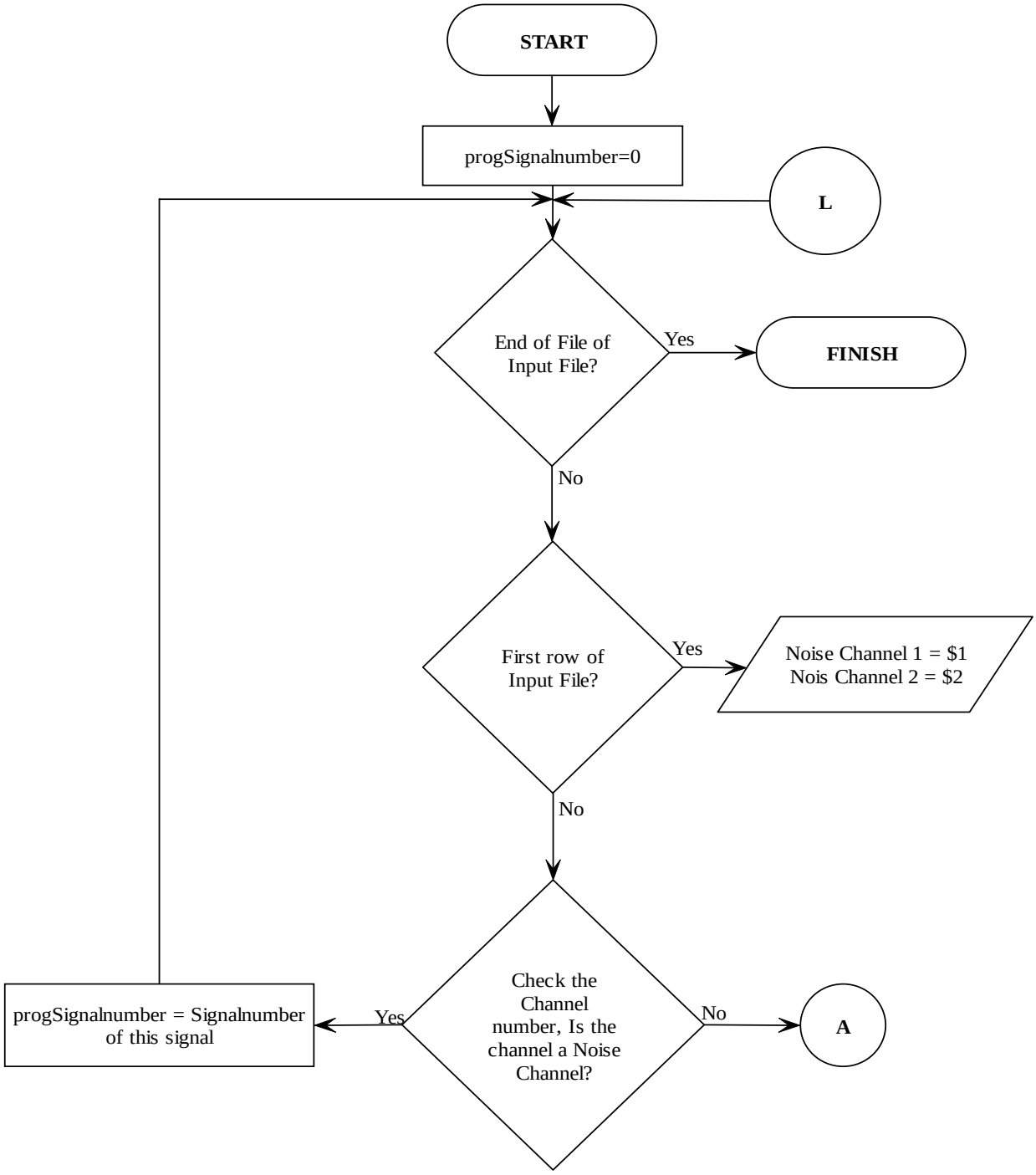


Figure 3.3a Flowchart of eliminatenoise.awk

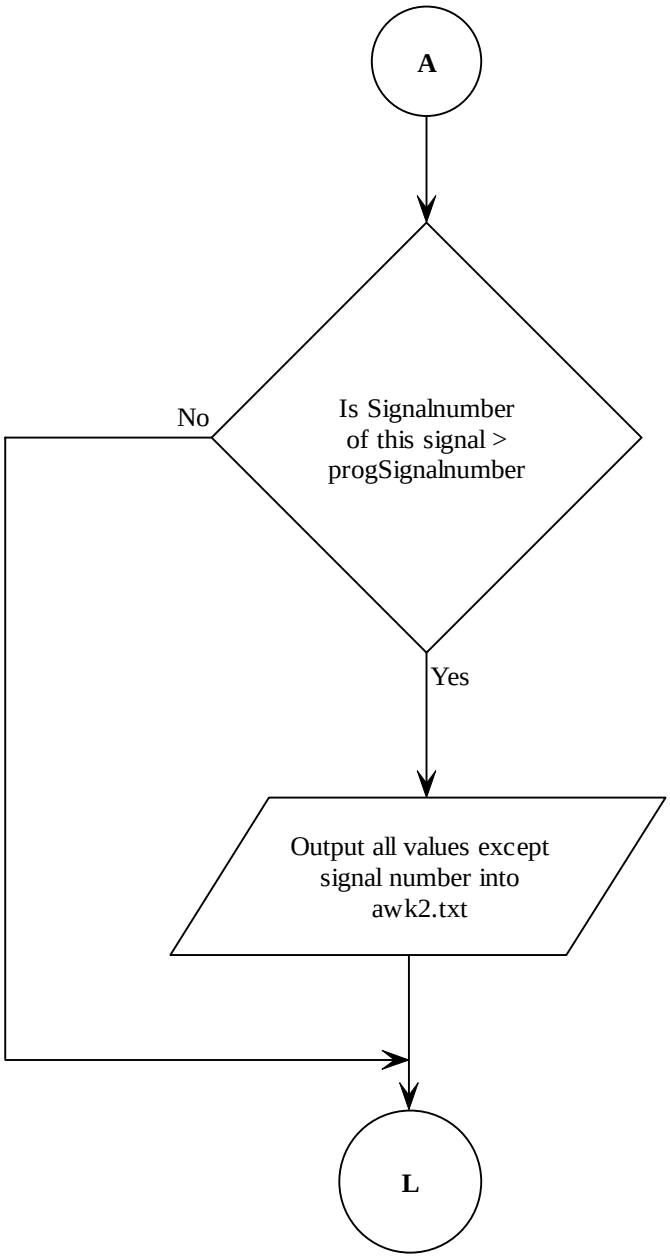


Figure 3.3b Flowchart of eliminatenoise.awk

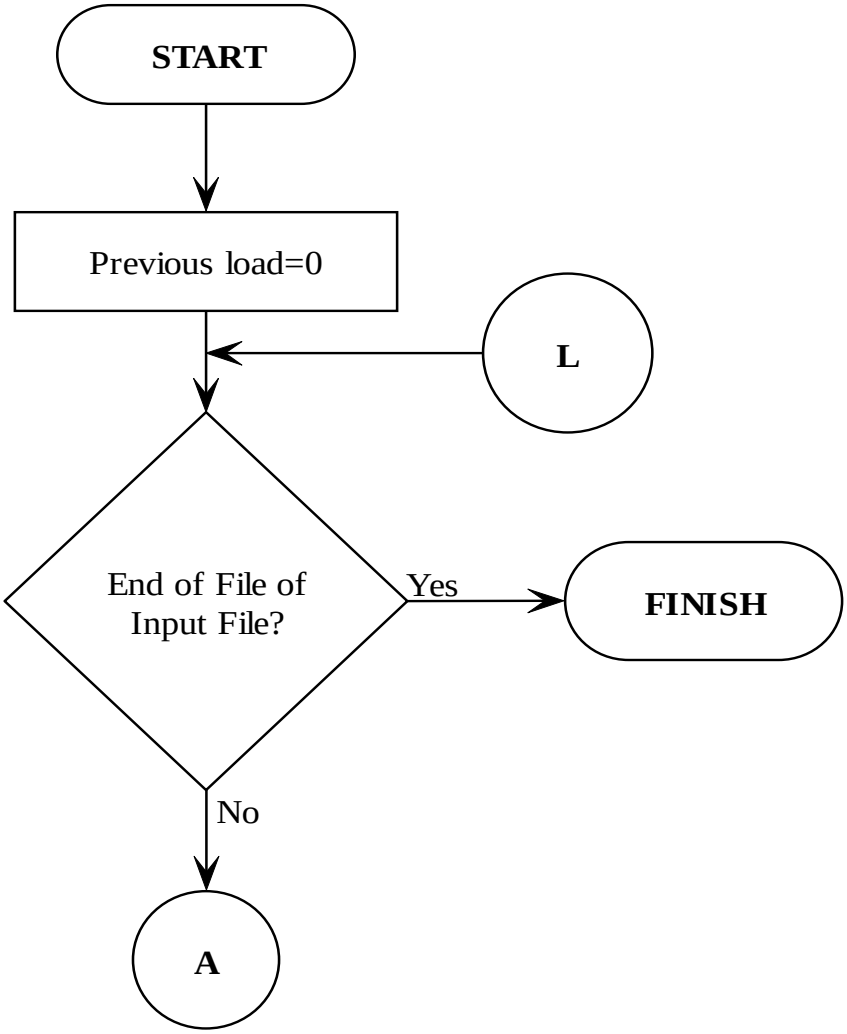


Figure 3.4a Flowchart of code1.awk

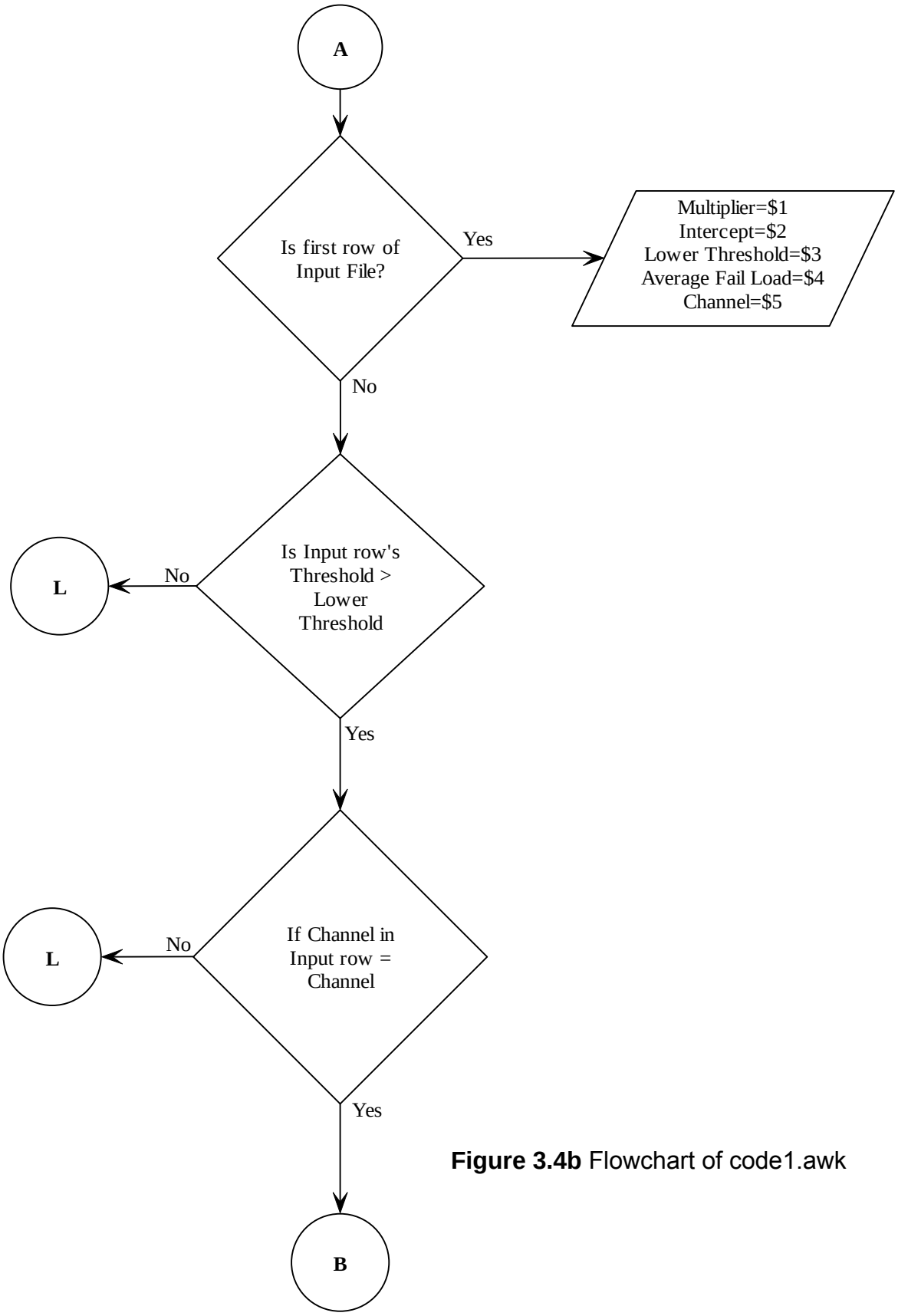


Figure 3.4b Flowchart of code1.awk

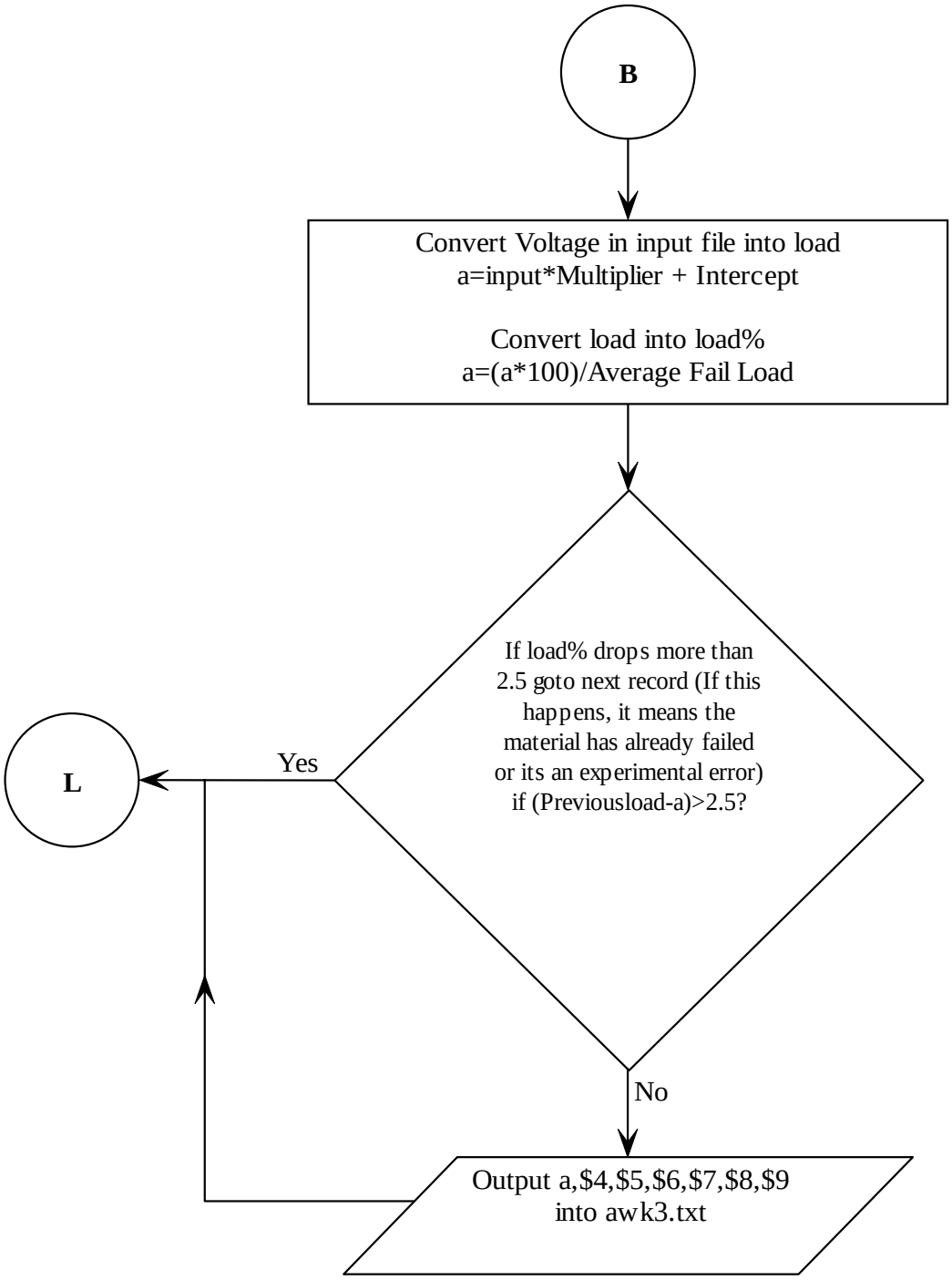


Figure 3.4c Flowchart of code1.awk

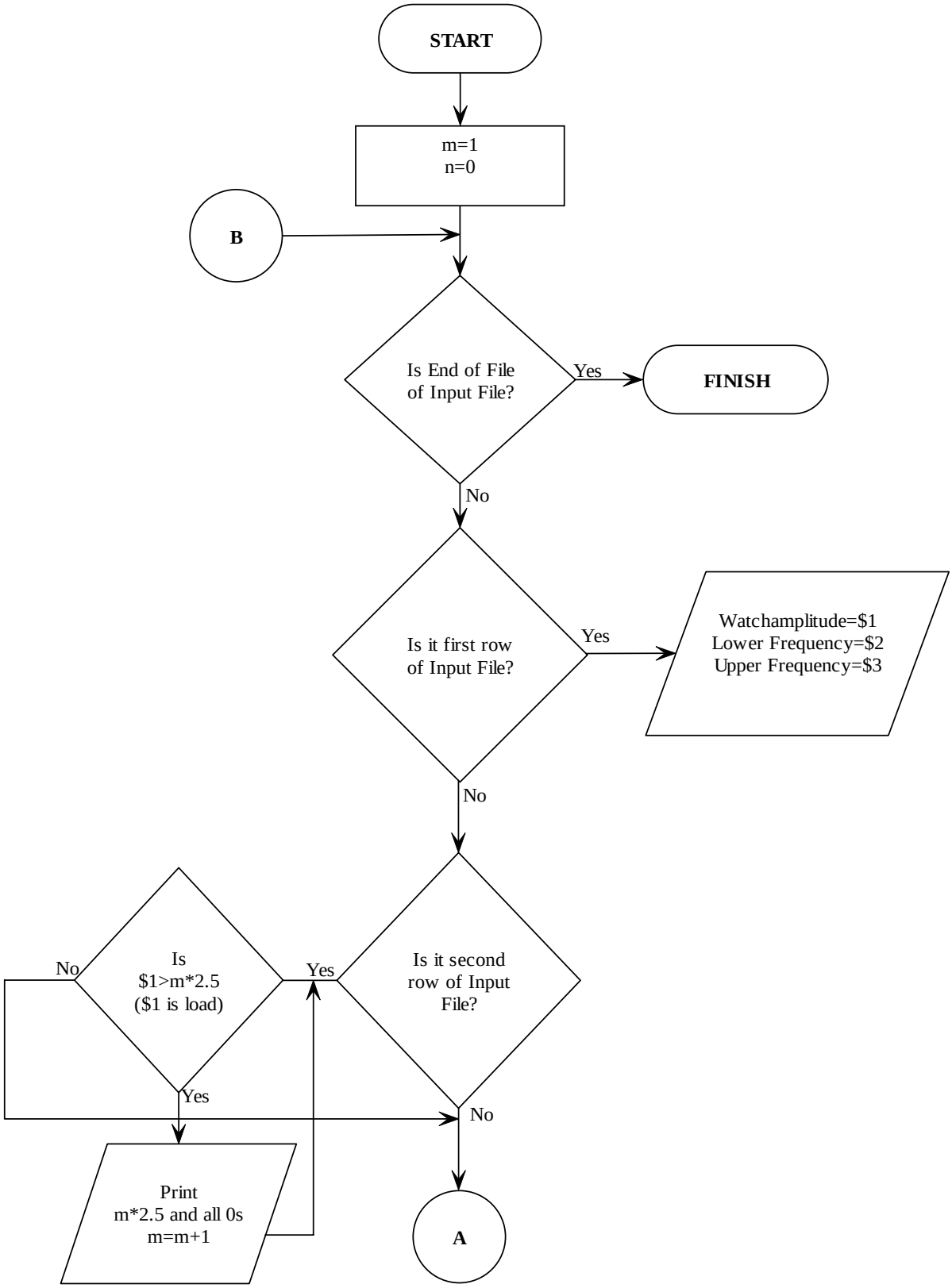


Figure 3.5a Flowchart of code2.awk

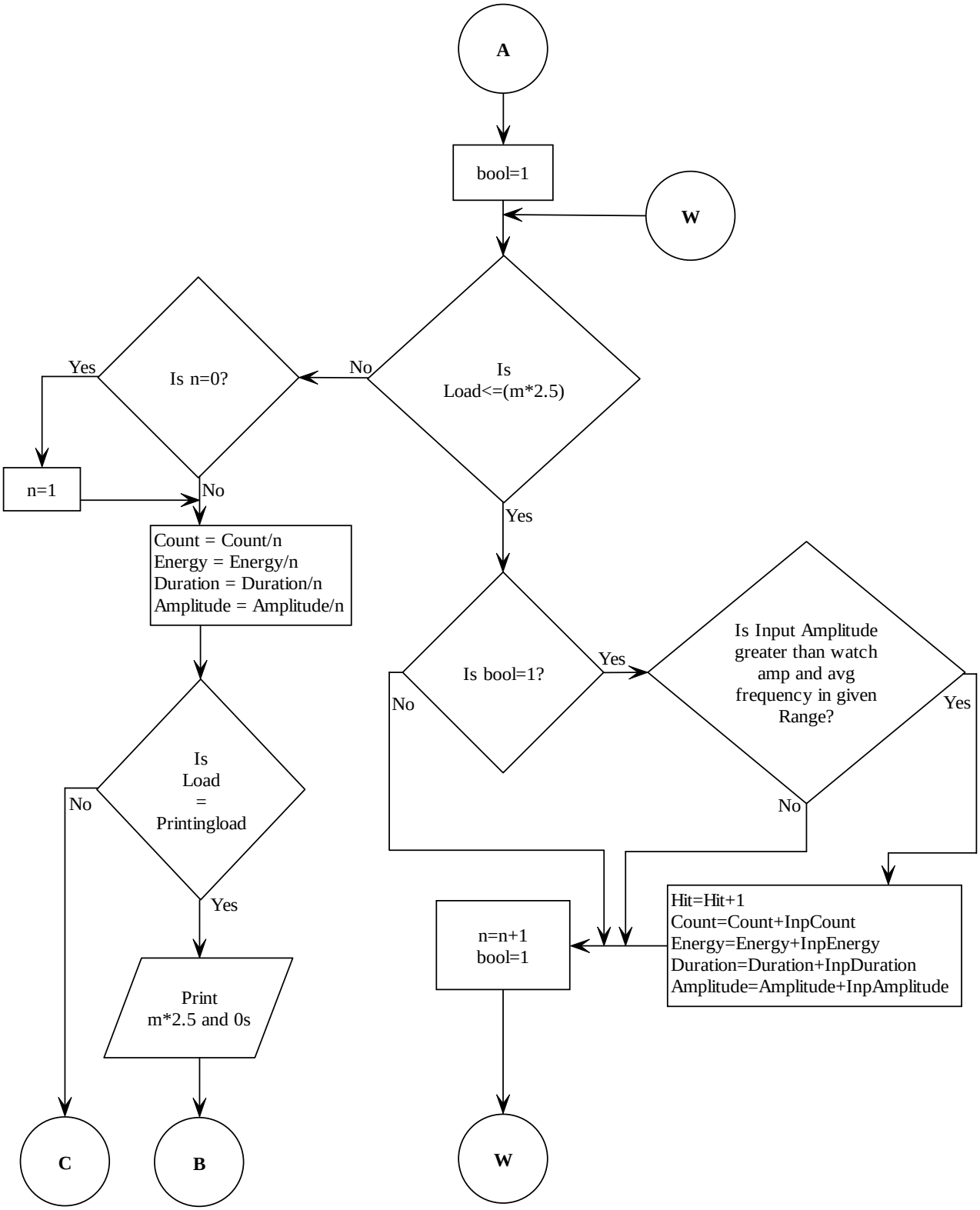


Figure 3.5b Flowchart of code2.awk

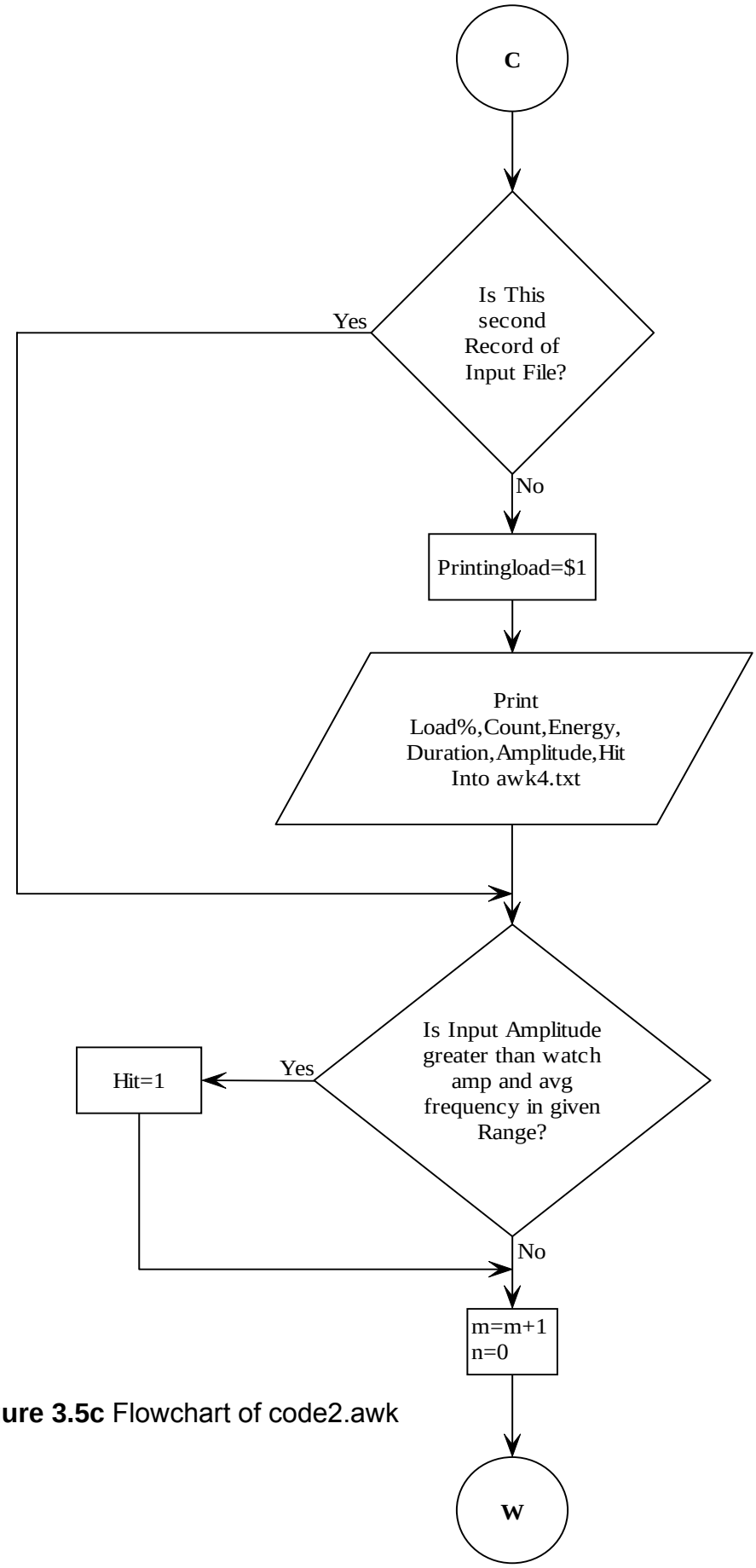


Figure 3.5c Flowchart of code2.awk

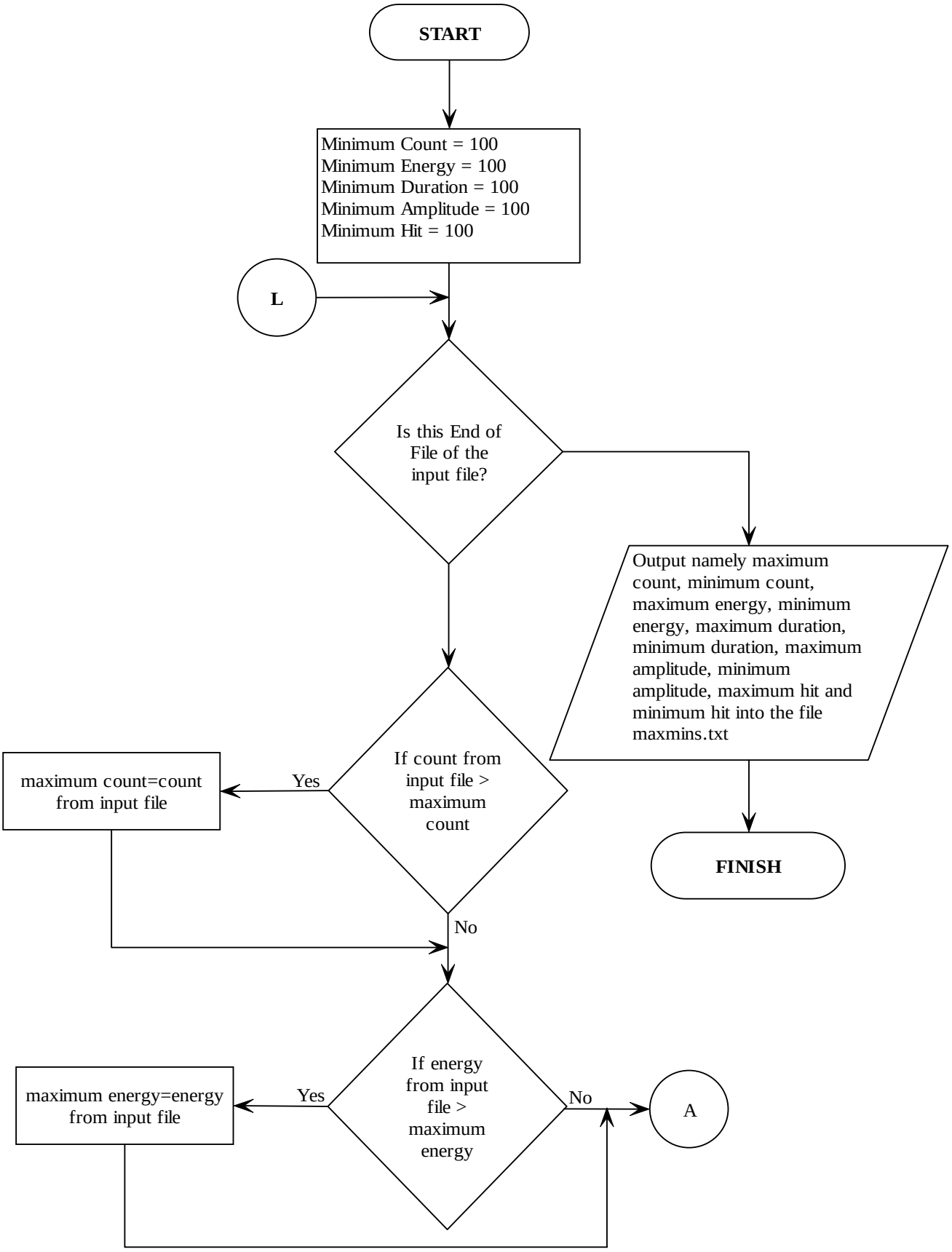


Figure 3.6a Flowchart of mamins.awk

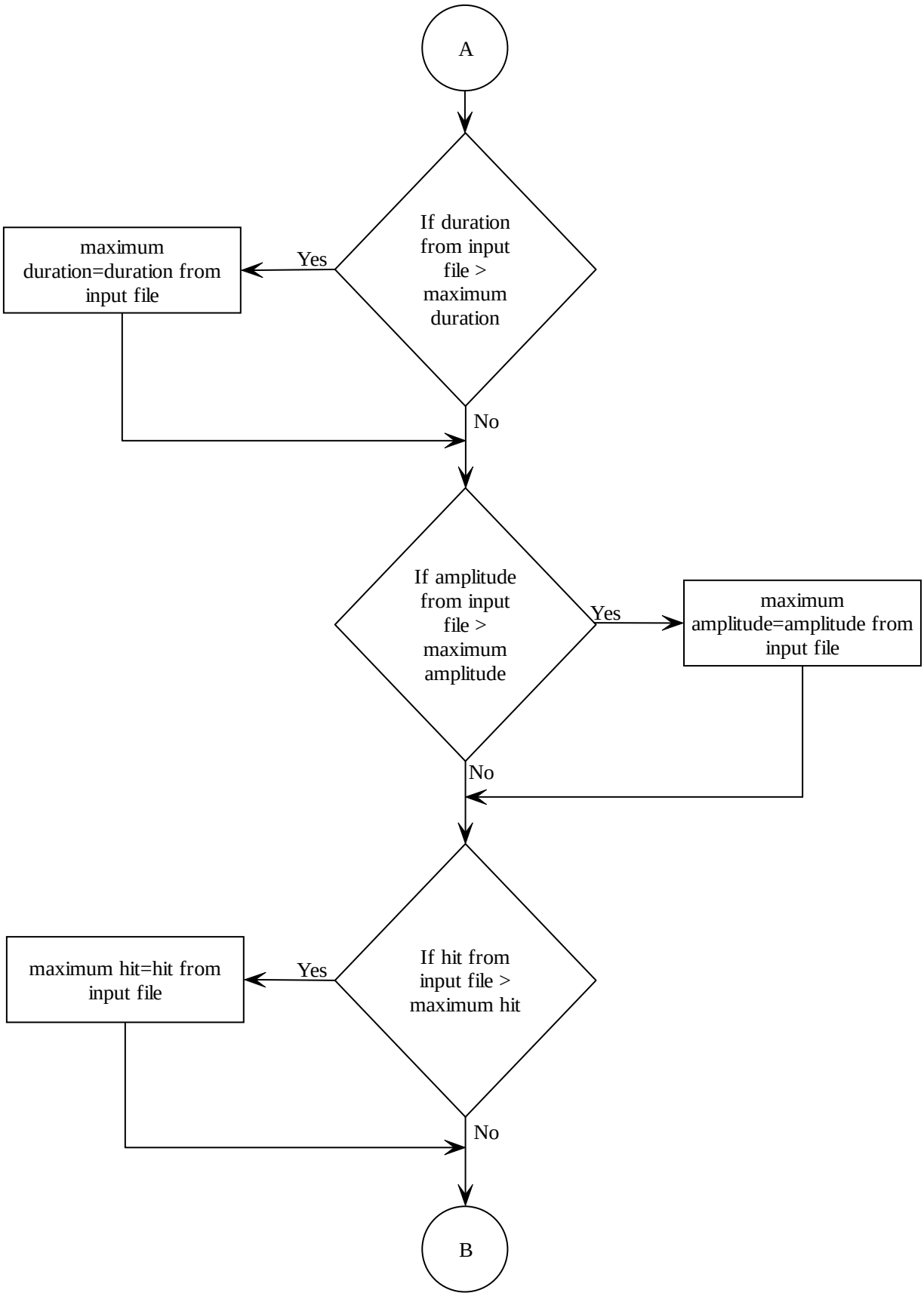


Figure 3.6b Flowchart of mamins.awk

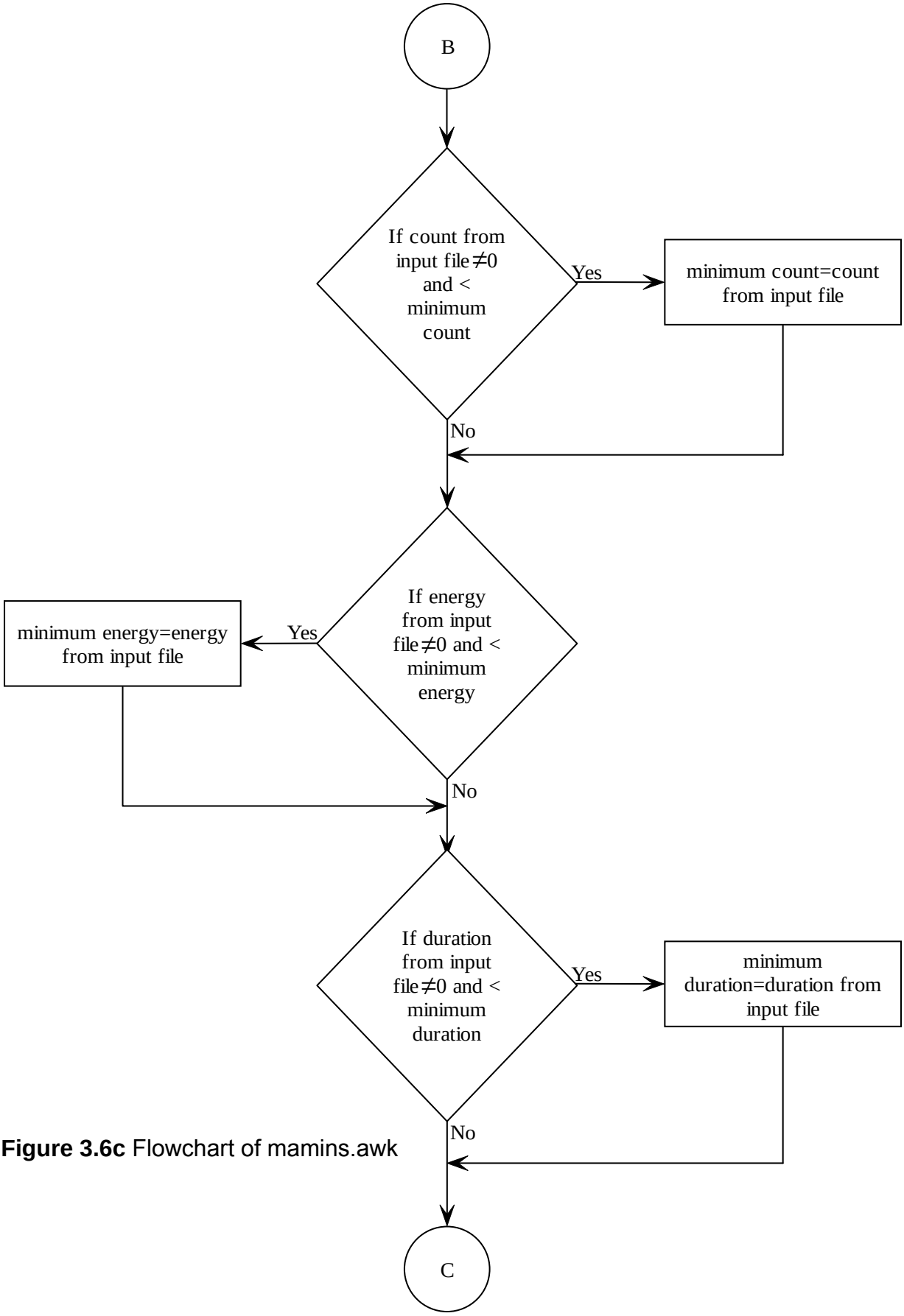


Figure 3.6c Flowchart of mamins.awk

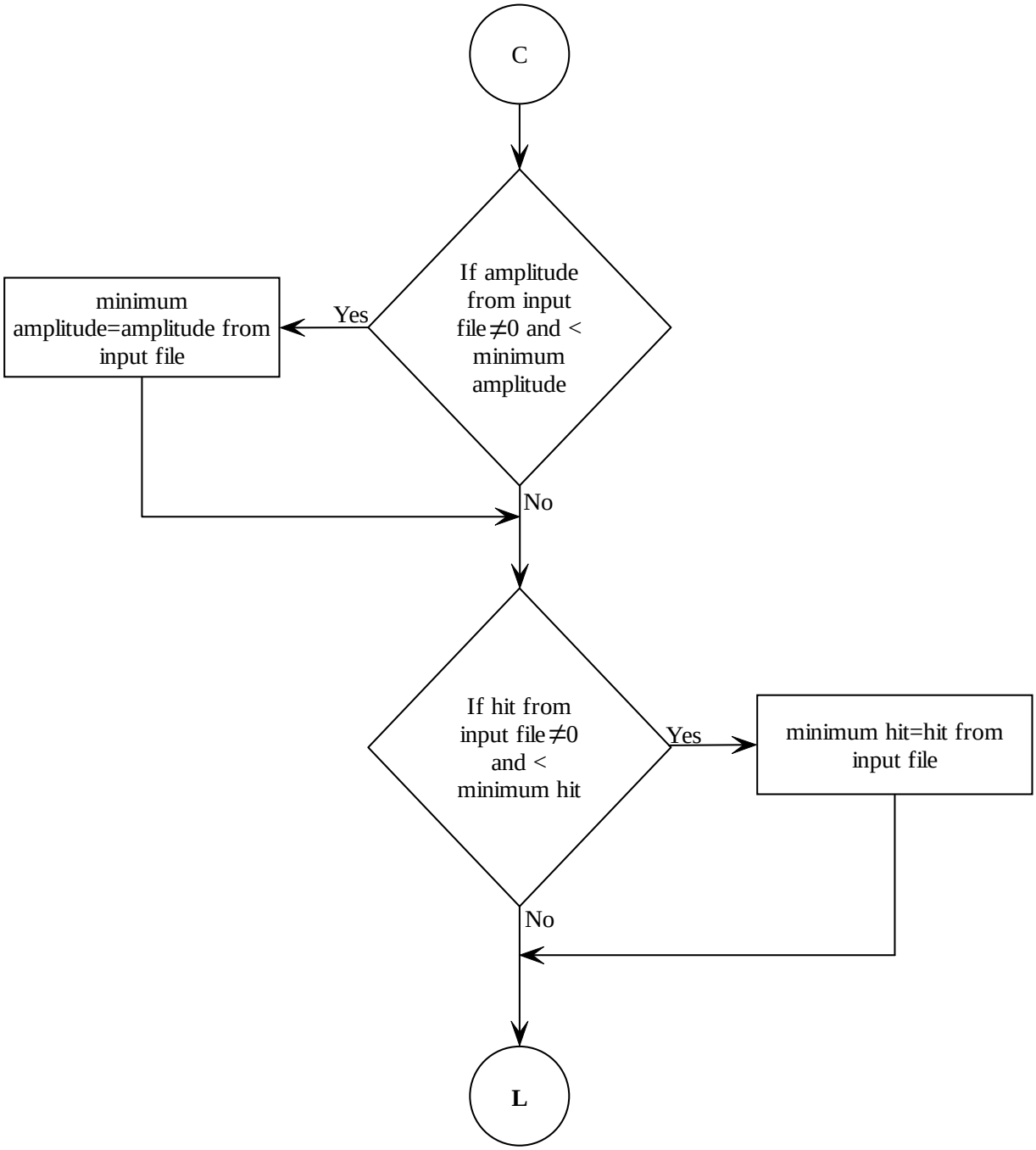


Figure 3.6d Flowchart of mamins.awk

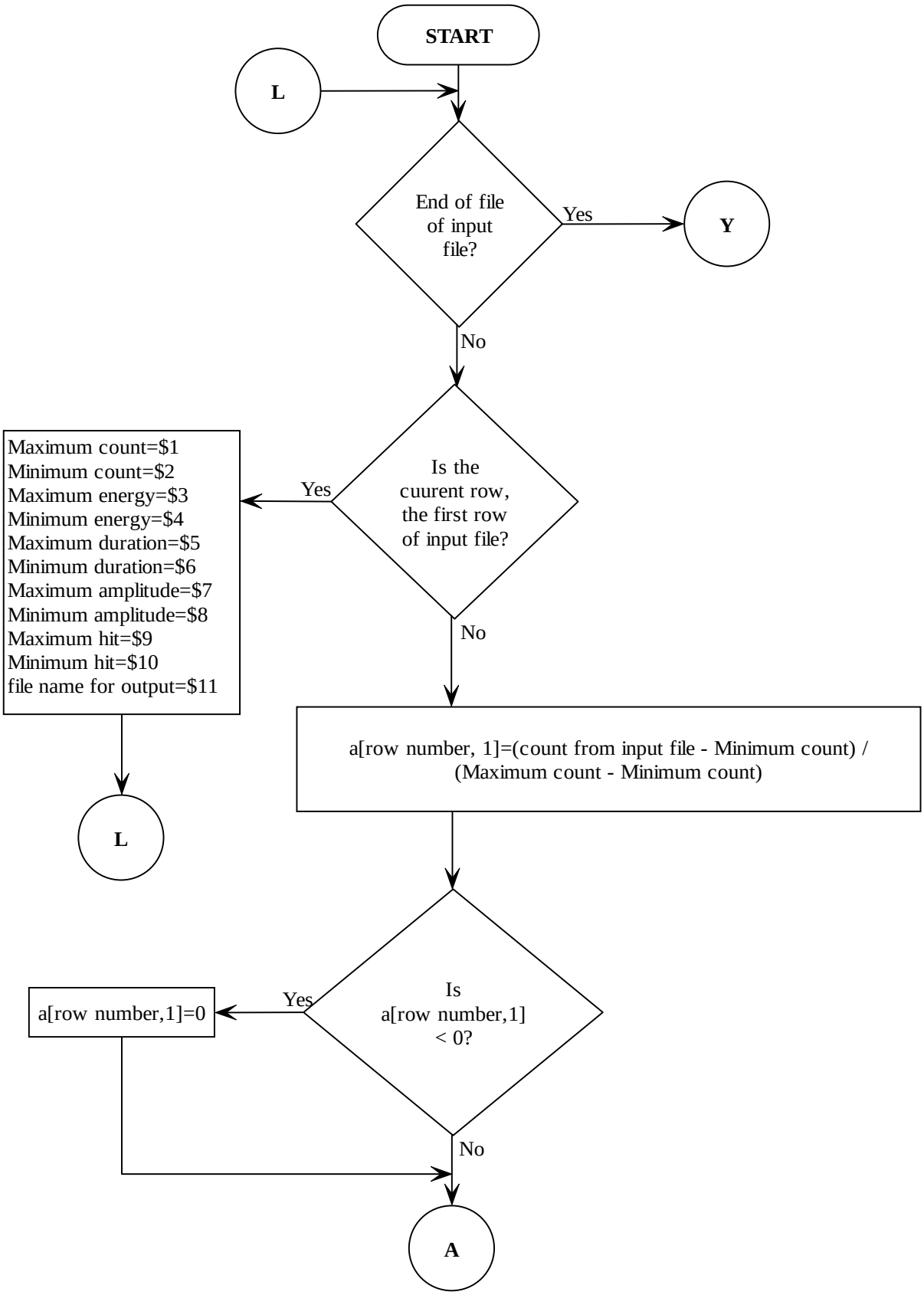


Figure 3.7a Flowchart of matrix.awk

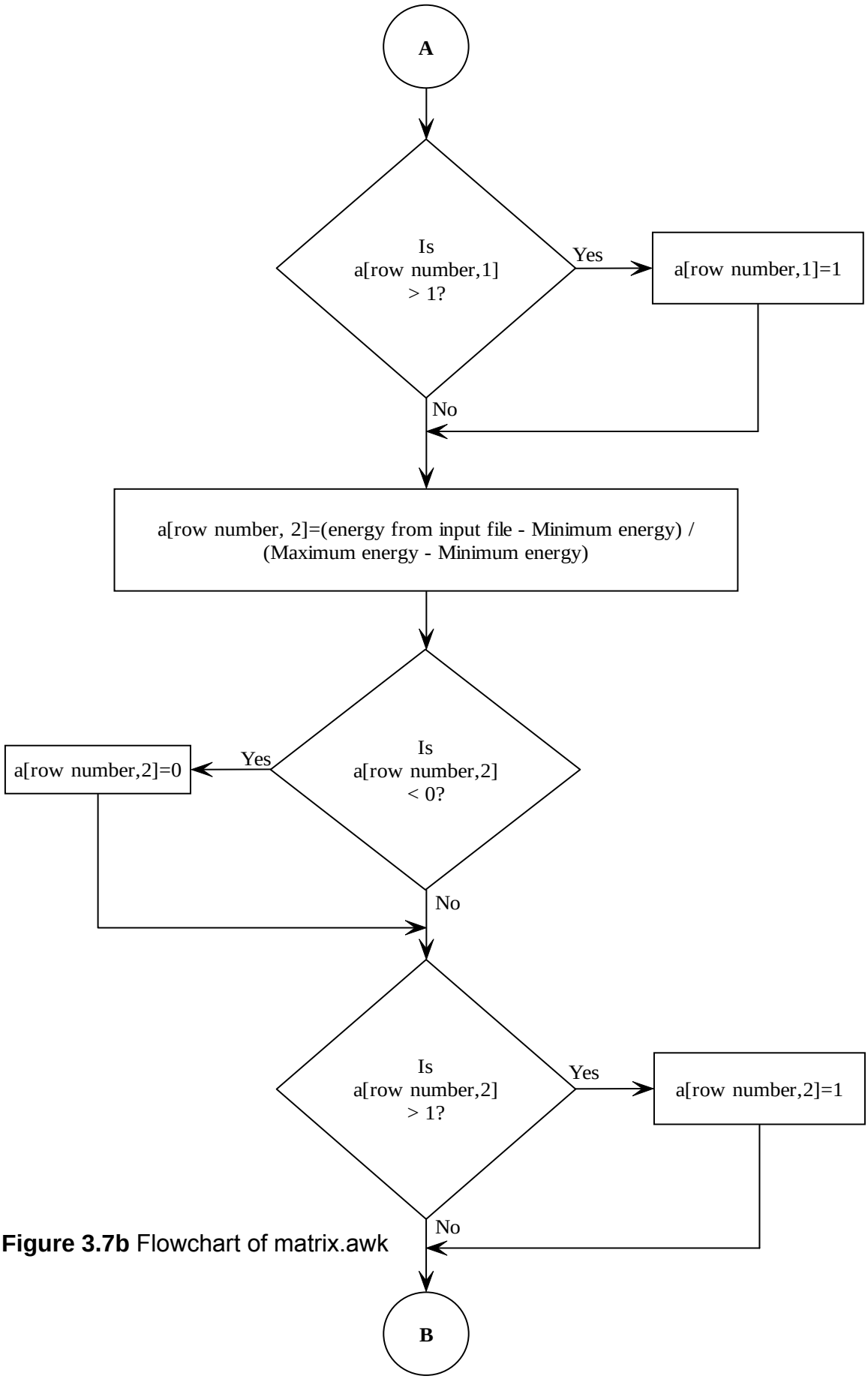


Figure 3.7b Flowchart of matrix.awk

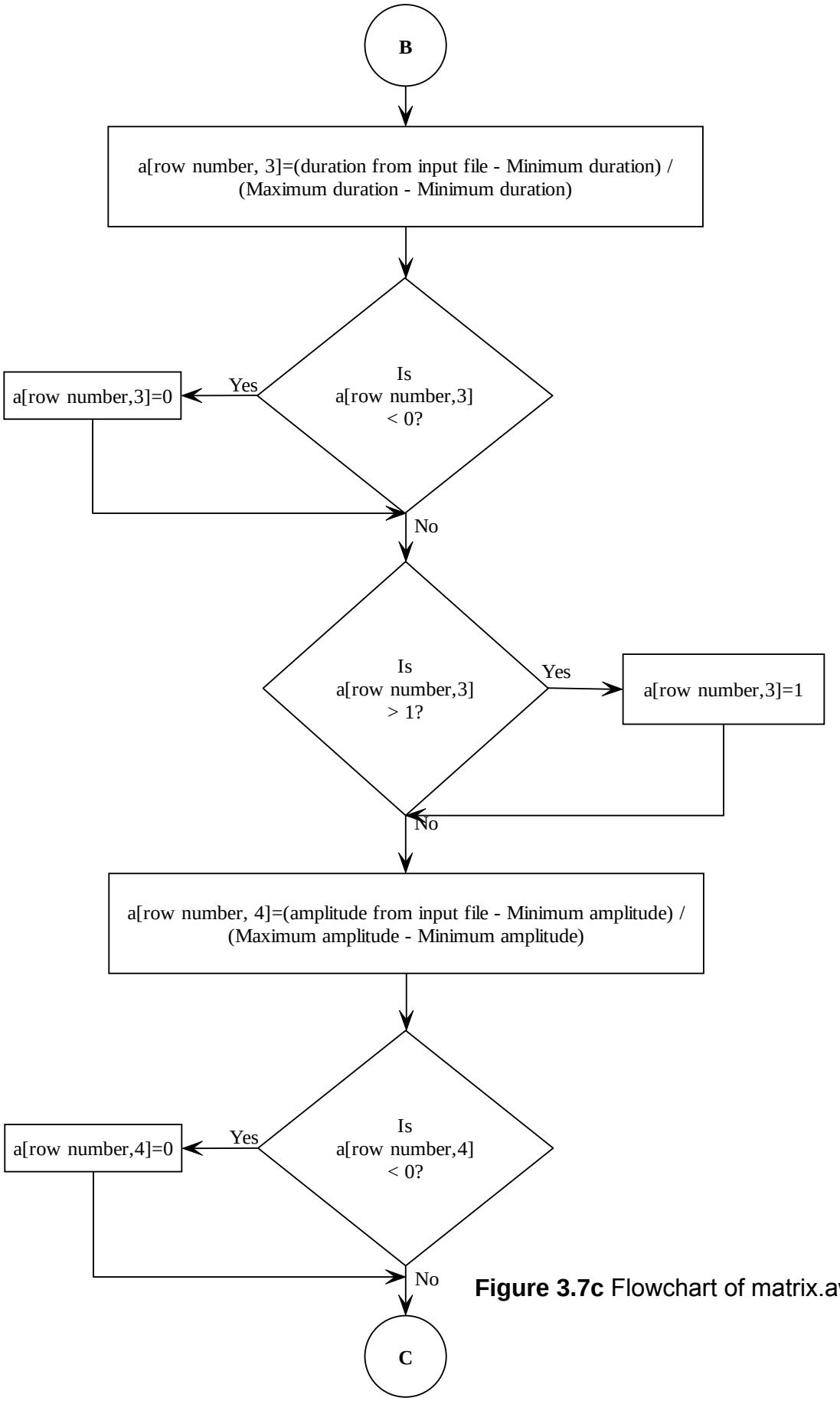


Figure 3.7c Flowchart of matrix.awk

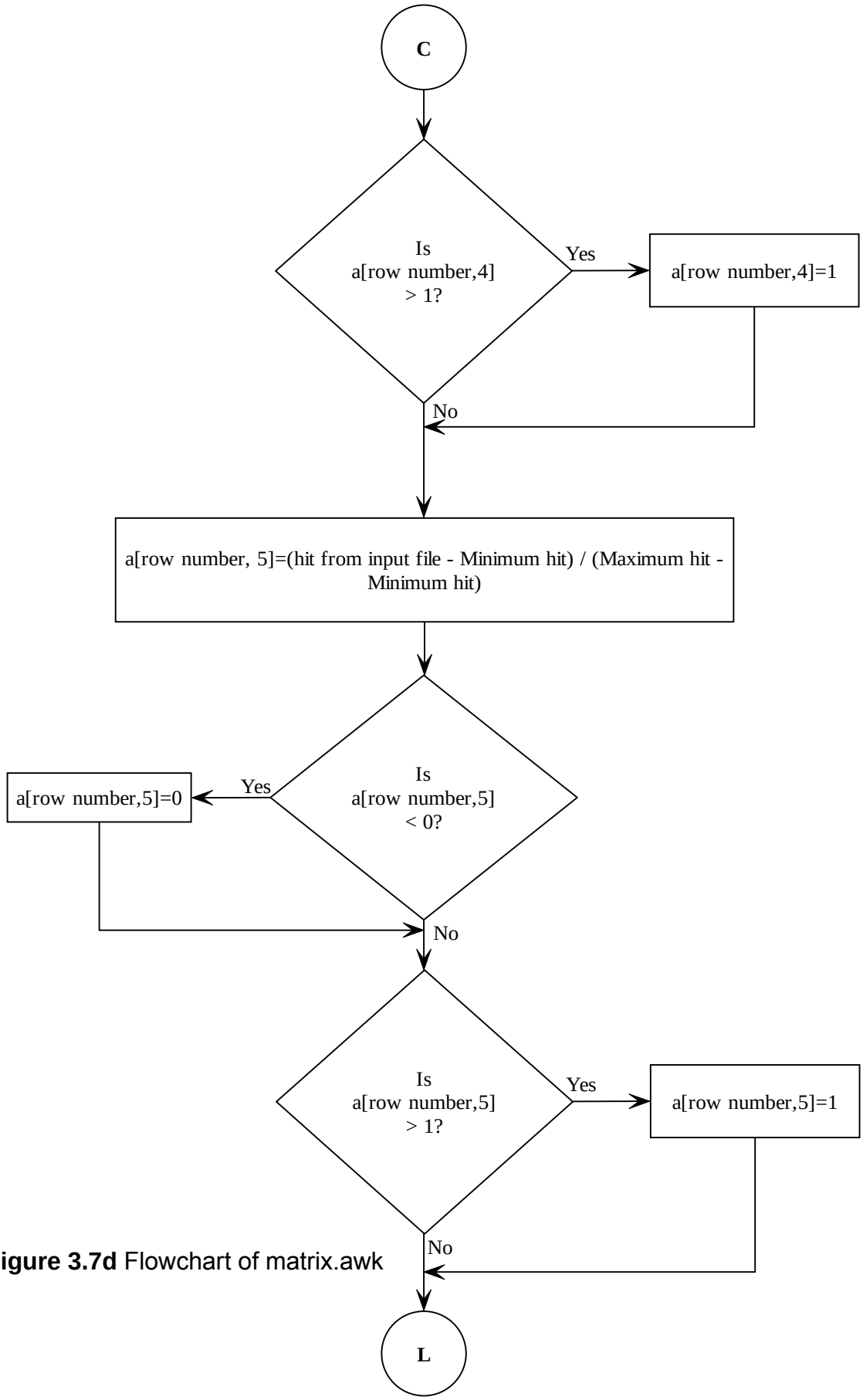


Figure 3.7d Flowchart of matrix.awk

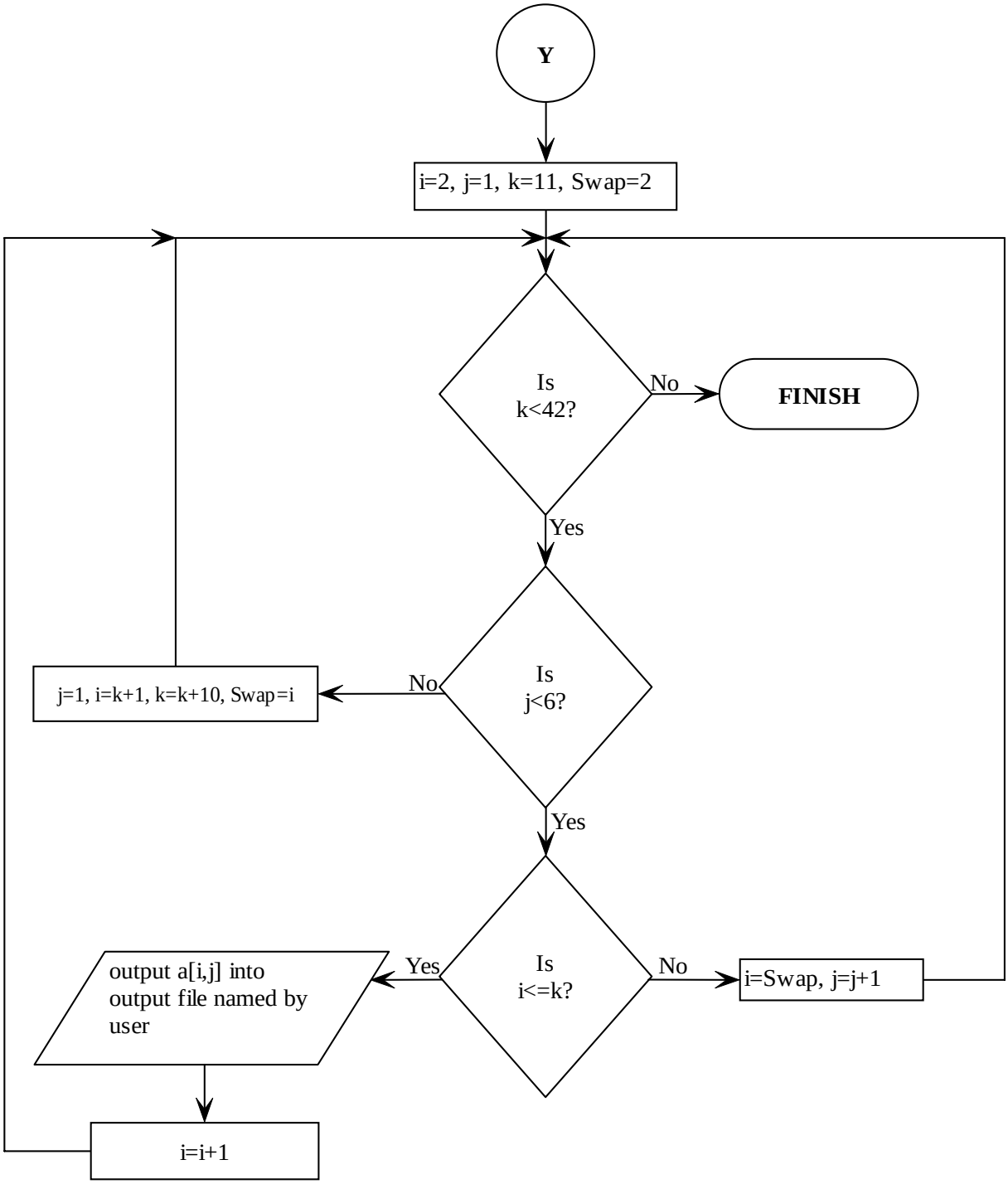


Figure 3.7e Flowchart of matrix.awk

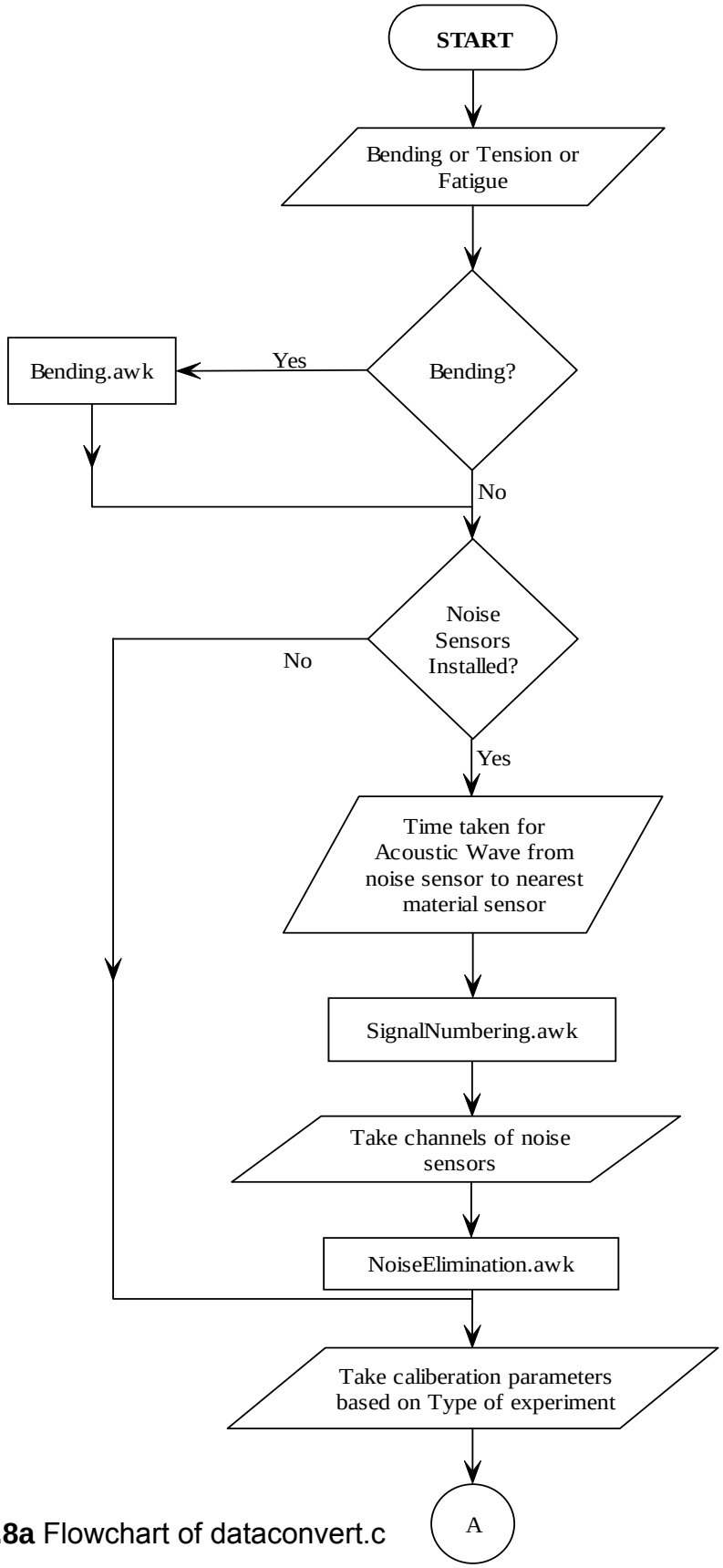


Figure 3.8a Flowchart of dataconvert.c

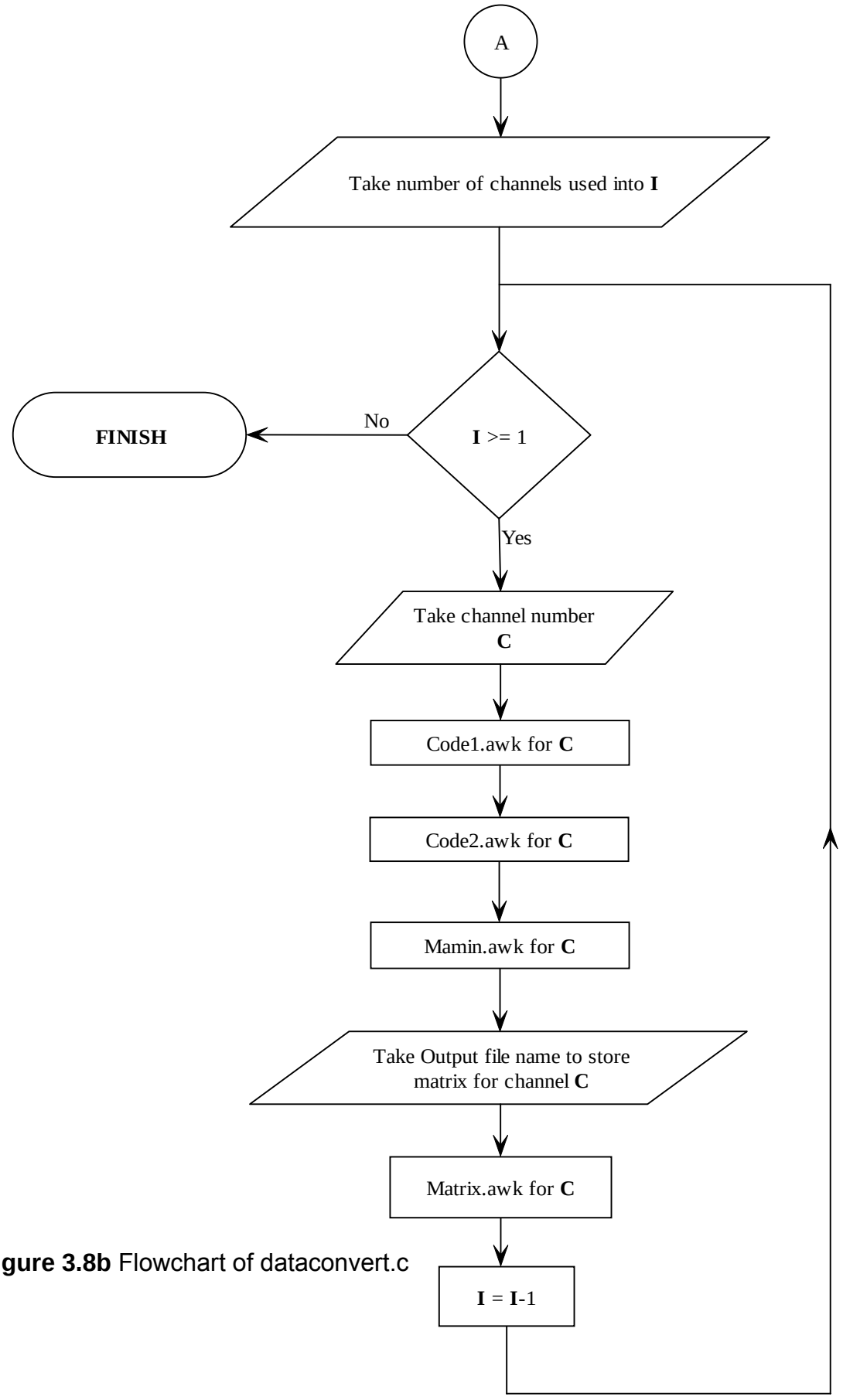


Figure 3.8b Flowchart of dataconvert.c

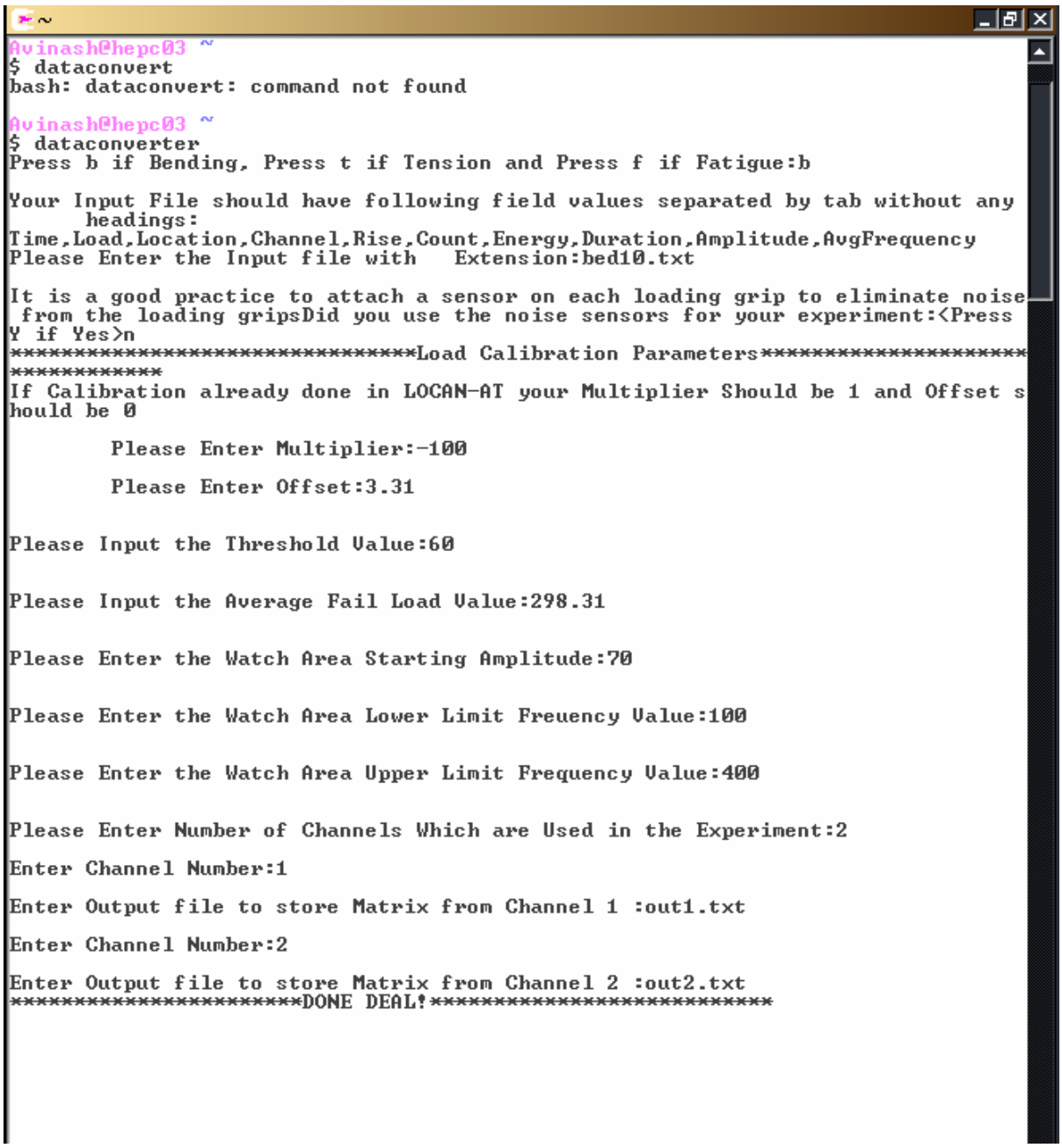
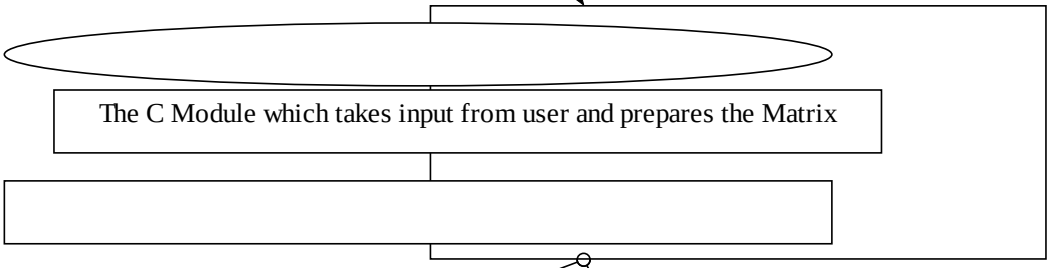


Figure 3.9 Screen Capture of dataconvert.exe

Raw Data with all information from LOCAN AT, 29,452 rows and 10 columns namely time, load, deflection, channel, rise, count, energy, duration, amplitude and average frequency (Headings from LOCAN AT removed)

9.3985482	-0.01	1.88	2	2	1	3	1	46	1000
10.2316043	-0.03	1.88	1	7	6	3	19	54	315
12.7367358	-0.04	1.88	2	1	1	2	1	46	1000
14.2790433	-0.05	1.88	1	13	5	2	16	52	312
14.8842503	-0.05	1.89	2	10	5	1	23	50	217
16.2464015	-0.09	1.91	2	14	2	1	14	49	142
16.8298543	-0.09	1.91	1	13	8	3	29	57	275
17.7238040	-0.10	1.91	2	14	6	3	21	54	285
17.9588743	-0.09	1.91	2	11	14	6	71	61	197
17.9588878	-0.09	1.91	6	11	9	3	46	57	195
17.9588925	-0.09	1.91	4	13	6	3	32	56	187
17.9589400	-0.09	1.91	1	8	4	2	35	51	114
23.1011013	-0.14	1.91	2	6	1	1	8	46	125
23.8129680	-0.16	1.91	1	8	8	4	35	56	228



Output from channel 1 in a file, named by user as out1.txt.(4 matrices, each of 10 columns and 5 rows)

0	0	0	0.323529	0	0	0	0.161765	0.372059	0.202206
0	0	0	0	0	0	0	0.120438	0.381387	0.17062
0	0	0	0.222385	0	0	0	0.14158	0.408856	0.202184
0	0	0	0	0	0	0	0.278985	0.438406	0.318841
0	0	0	0	0	0	0	0	0	0
0.129412	0.147899	0.1375	0.328922	0.237255	0	0.276471	0.2	0.134034	0.129412
0.160584	0.298227	0.291058	0.167275	0.053528	0.098135	0.350365	0.233577	0.18926	0.135879
0.304077	0.142468	0.090301	0.235852	0.118789	0	0.172659	0.137625	0.127373	0.06747
0.182195	0.284679	0.338768	0.305556	0.21256	0.177134	0.427536	0.289855	0.23913	0.23913
0	0	0	0	0	0	0.003021	0	0	0

Output from channel 2 in a file, named by user as out2.txt.(4 matrices, each of 10 columns and 5 rows)

0	0	0	0.323529	0	0	0	0.161765	0.372059	0.202206
0	0	0	0	0	0	0	0.120438	0.381387	0.17062
0	0	0	0.222385	0	0	0	0.14158	0.408856	0.202184
0	0	0	0	0	0	0	0.278985	0.438406	0.318841
0	0	0	0	0	0	0	0	0	0
0.129412	0.147899	0.1375	0.328922	0.237255	0	0.276471	0.2	0.134034	0.129412
0.160584	0.298227	0.291058	0.167275	0.053528	0.098135	0.350365	0.233577	0.18926	0.135879
0.304077	0.142468	0.090301	0.235852	0.118789	0	0.172659	0.137625	0.127373	0.06747
0.182195	0.284679	0.338768	0.305556	0.21256	0.177134	0.427536	0.289855	0.23913	0.23913
0	0	0	0	0	0	0.003021	0	0	0

Figure 3.10 Schematic showing the working of dataconvert.exe

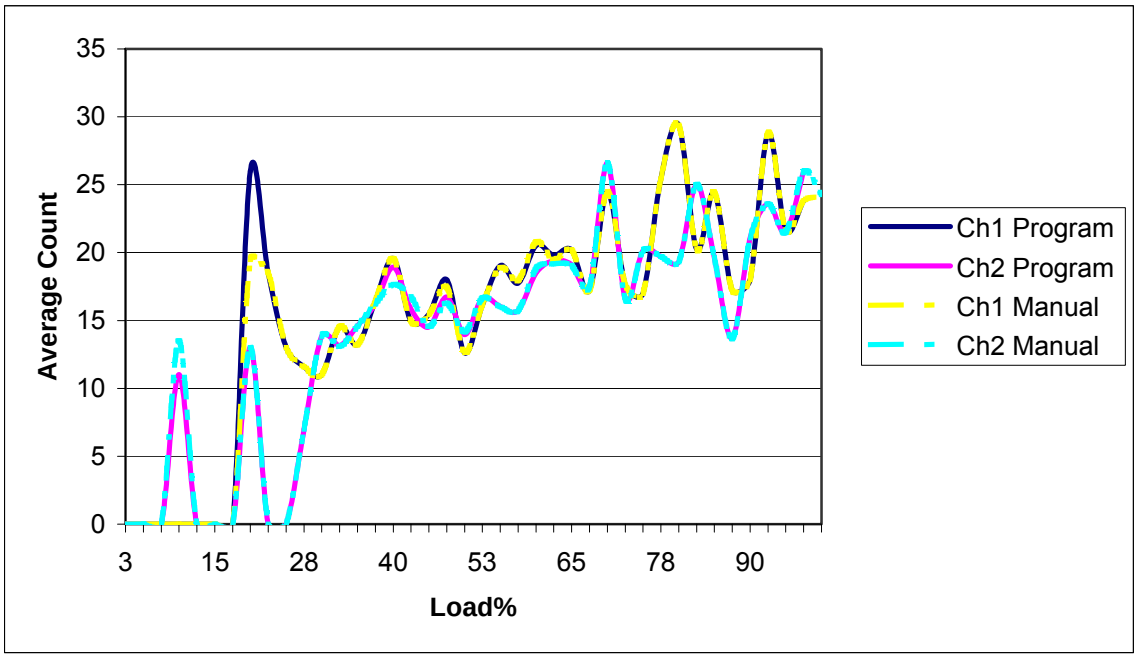


Figure 3.11 Test B2, Average Count vs. Load%

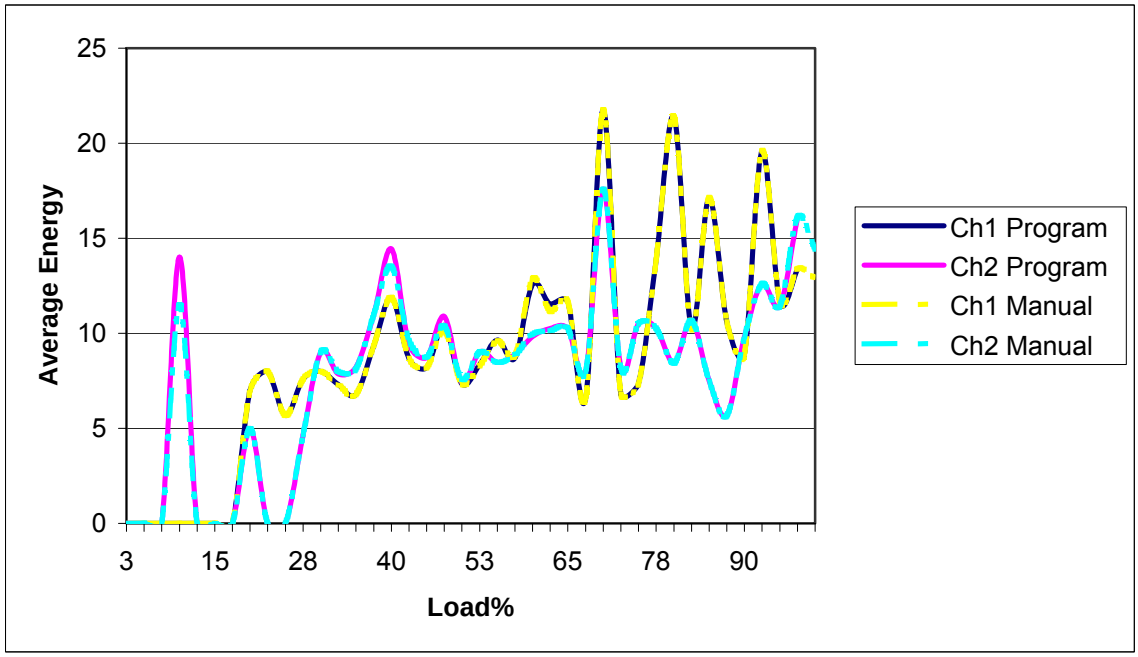


Figure 3.12 Test B2, Average Energy vs. Load%

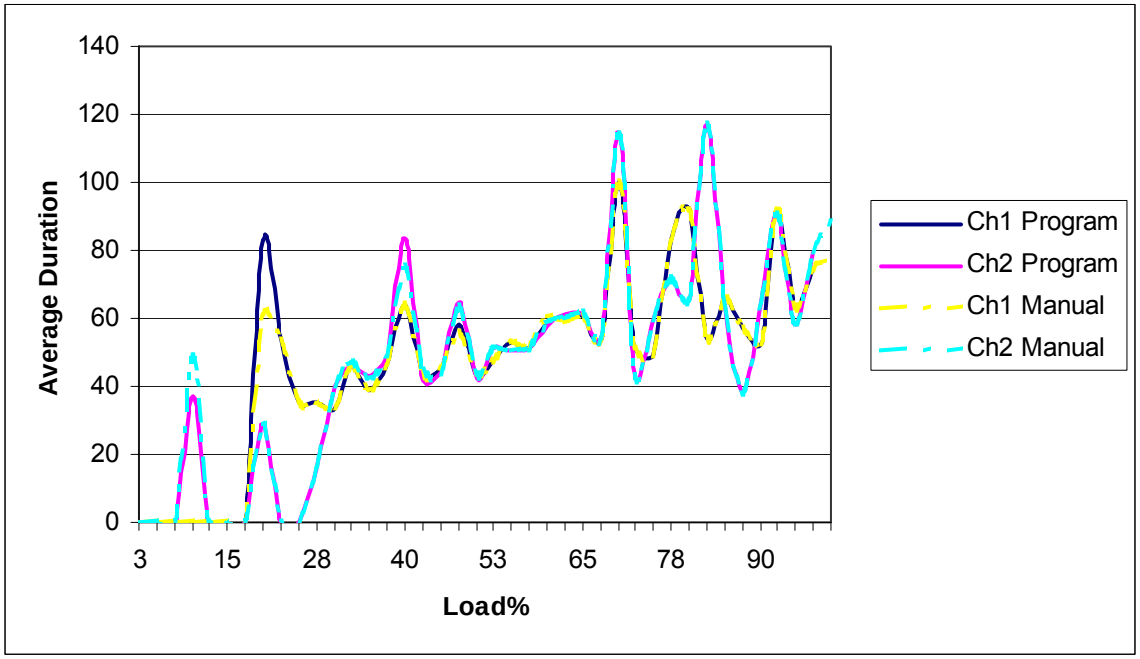


Figure 3.13 Test B2, Average Duration vs. Load%

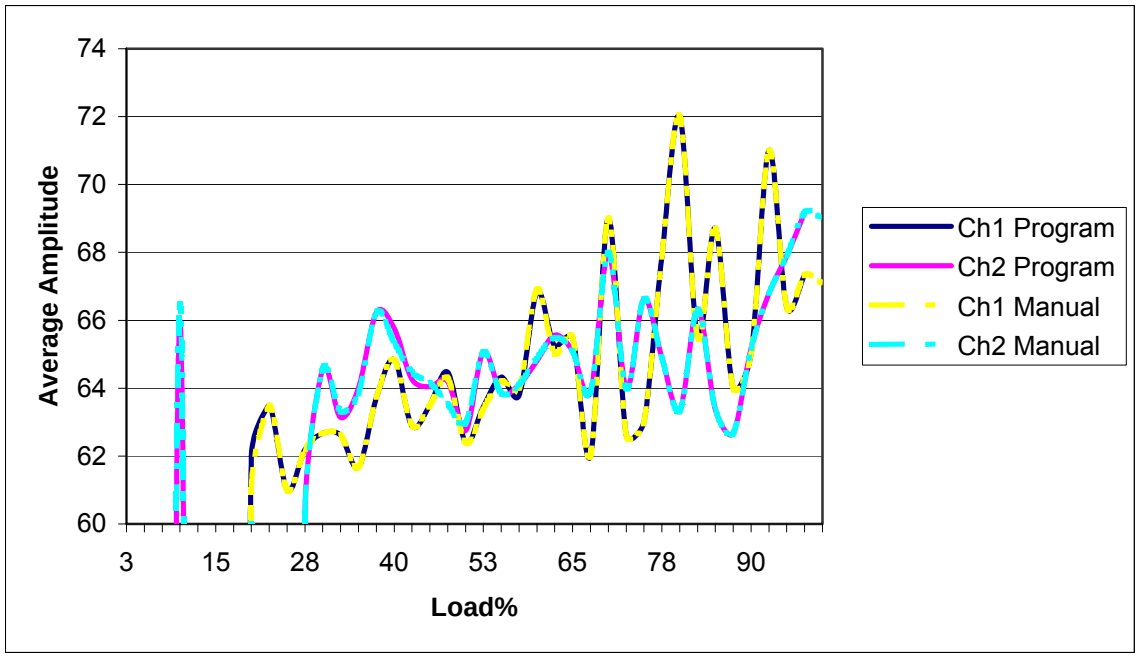


Figure 3.14 Test B2, Average Amplitude vs. Load%

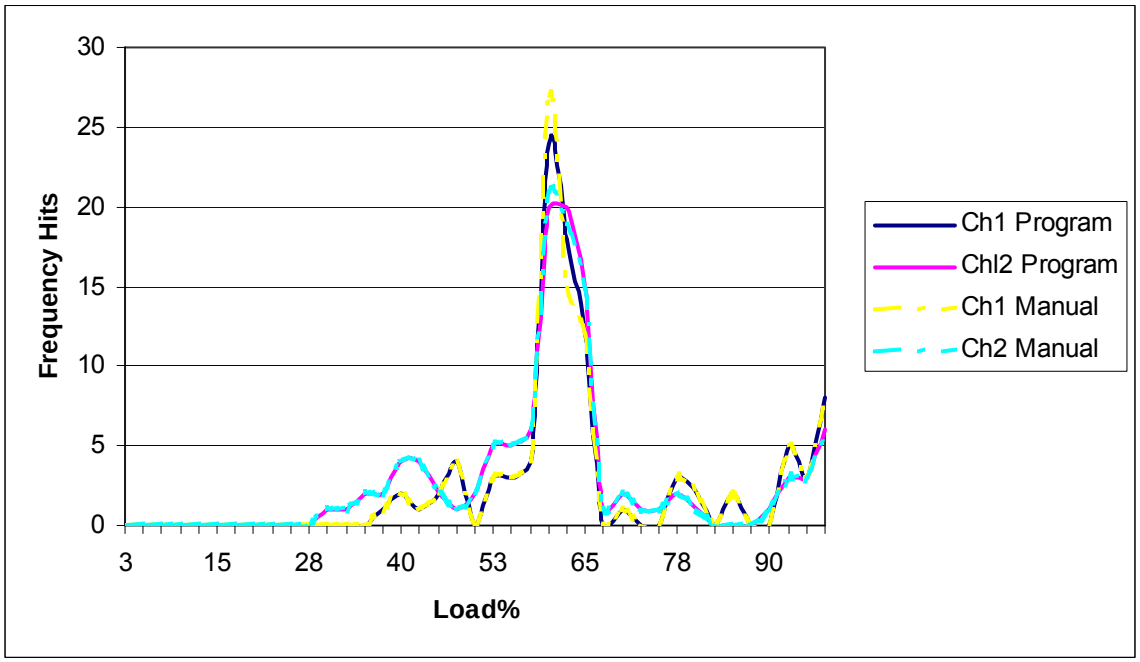


Figure 3.15 Test B2, Frequency Hits vs. Load%

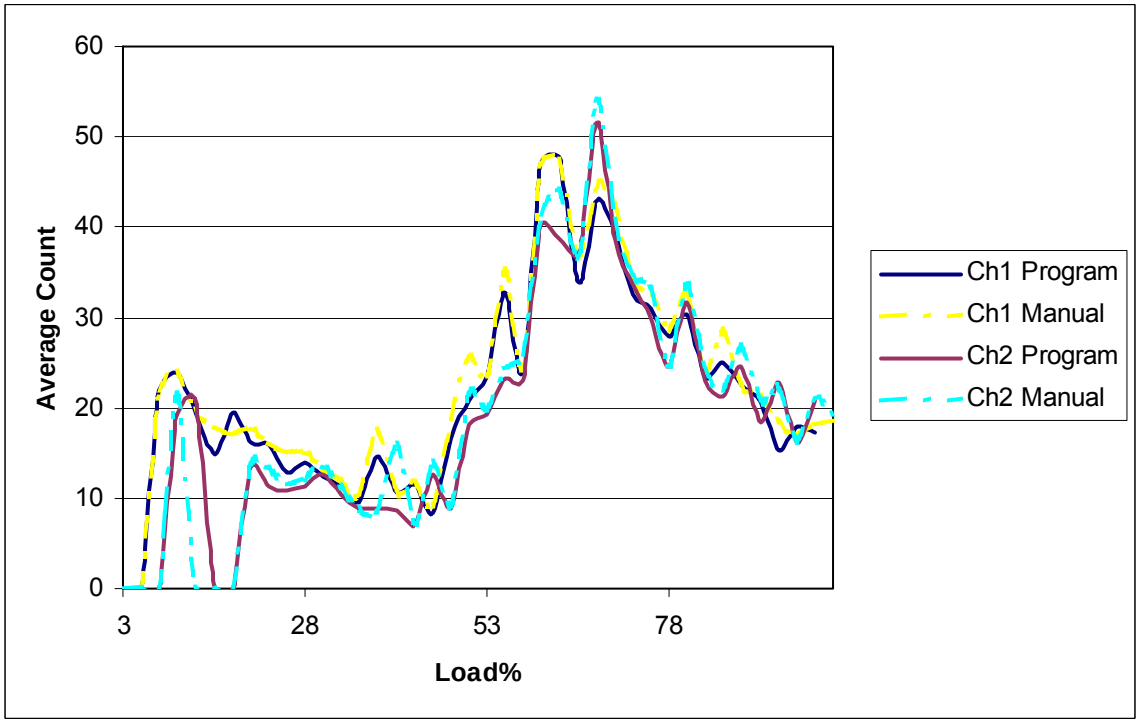


Figure 3.16 Test 1t, Average Count vs. Load%

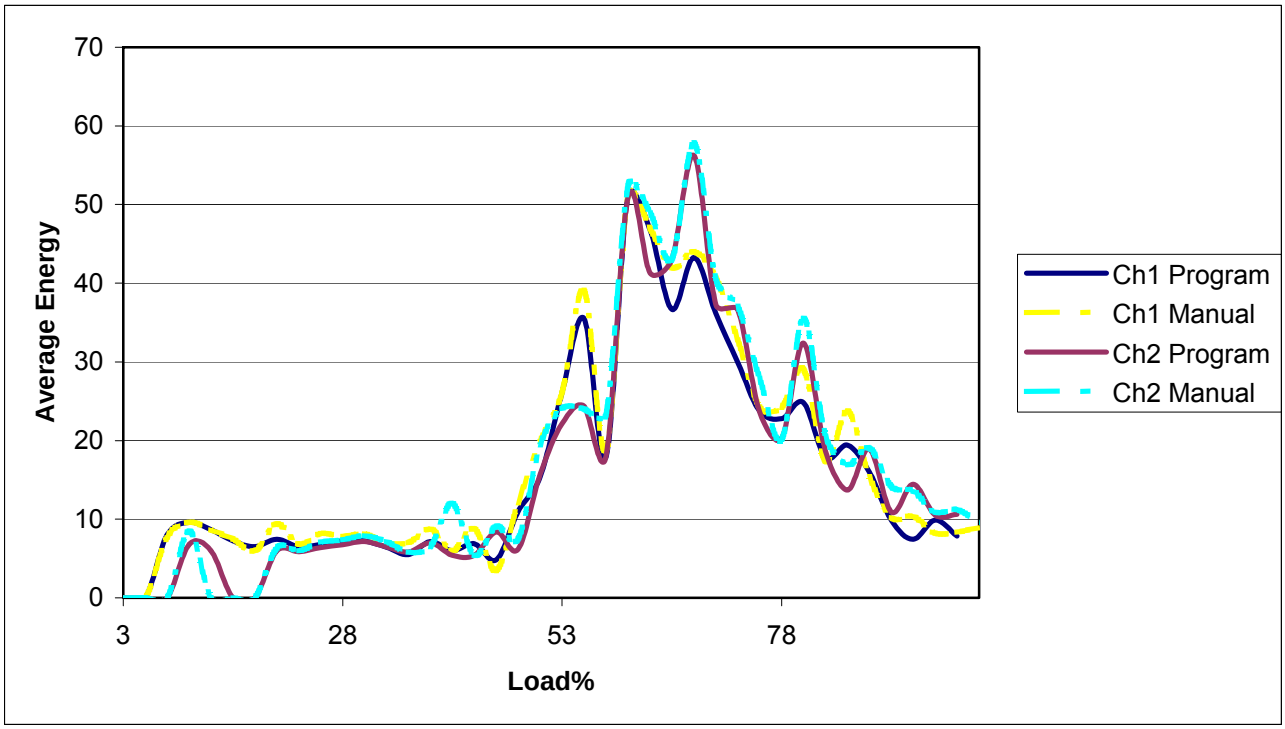


Figure 3.17 Test 1t, Average Energy vs. Load%

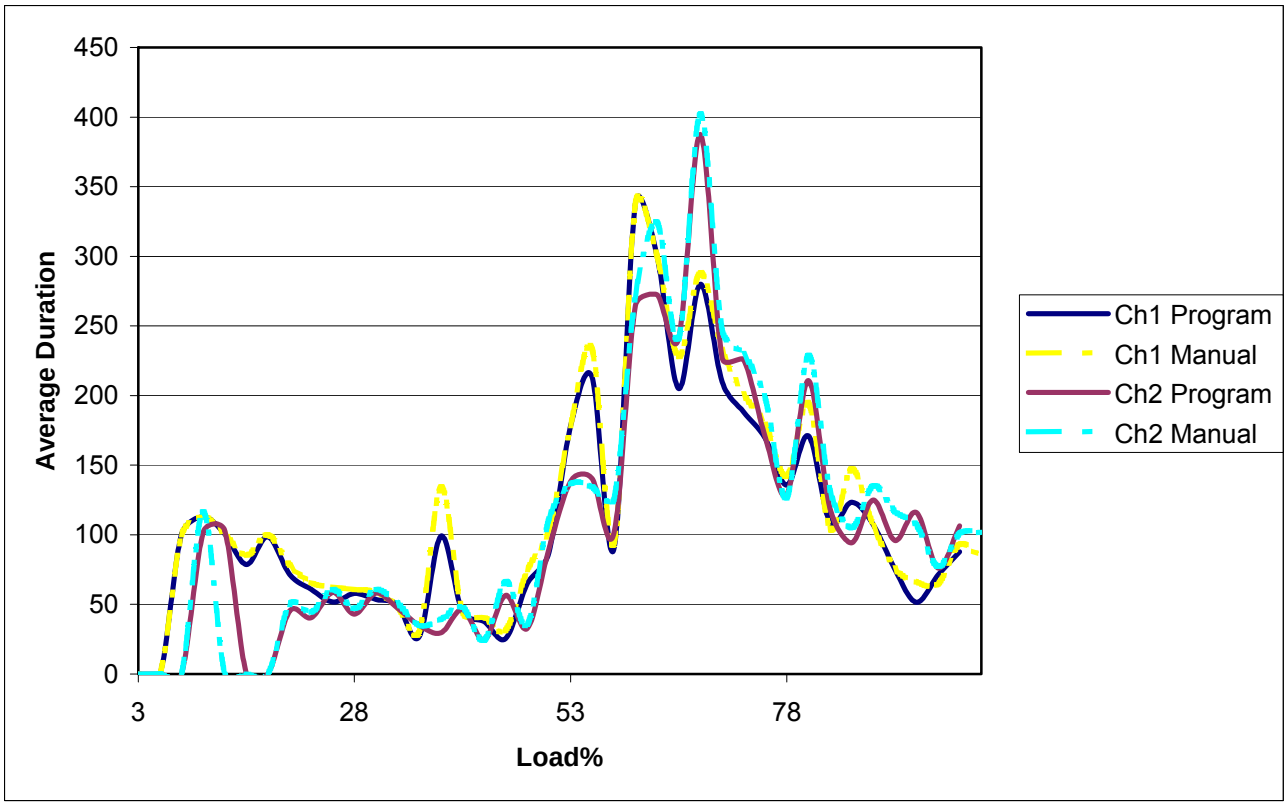


Figure 3.18 Test 1t, Average Duration vs. Load%

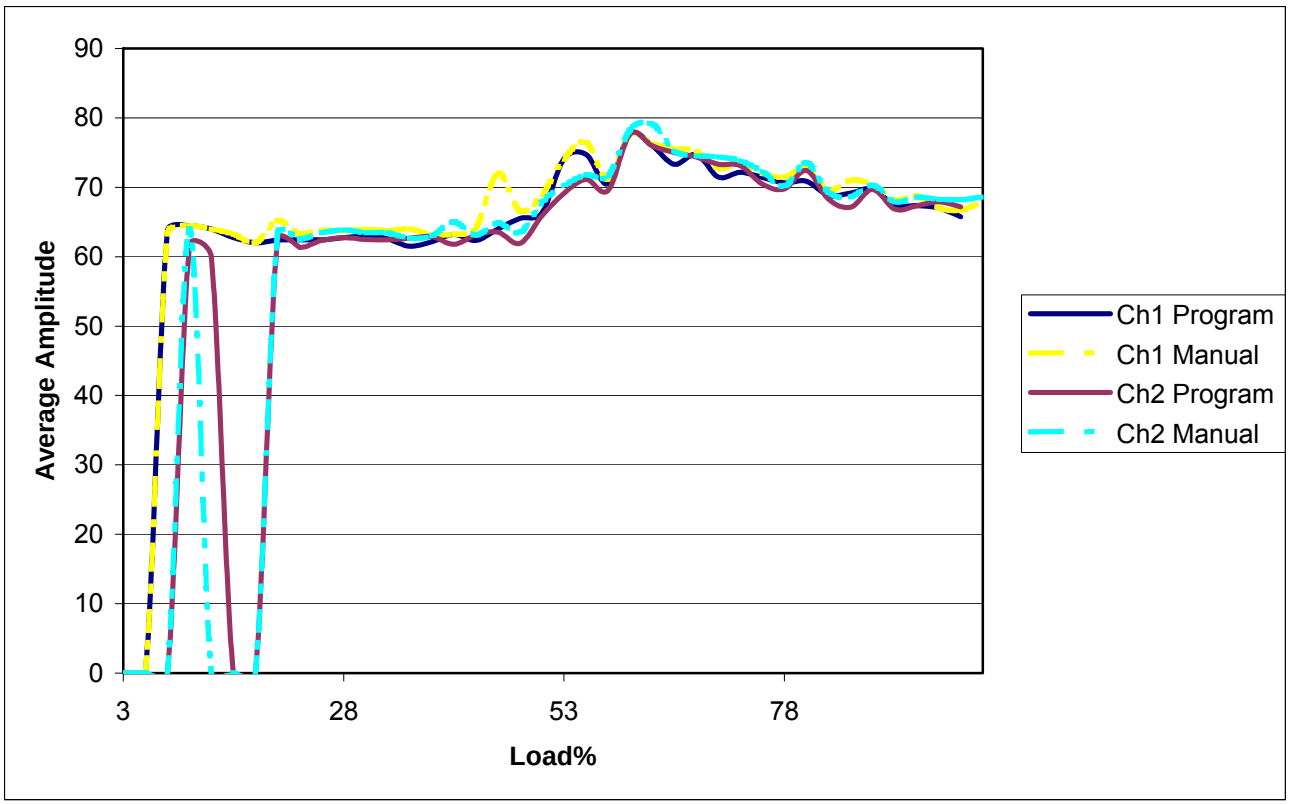


Figure 3.19 Test 1t, Average Amplitude vs. Load%

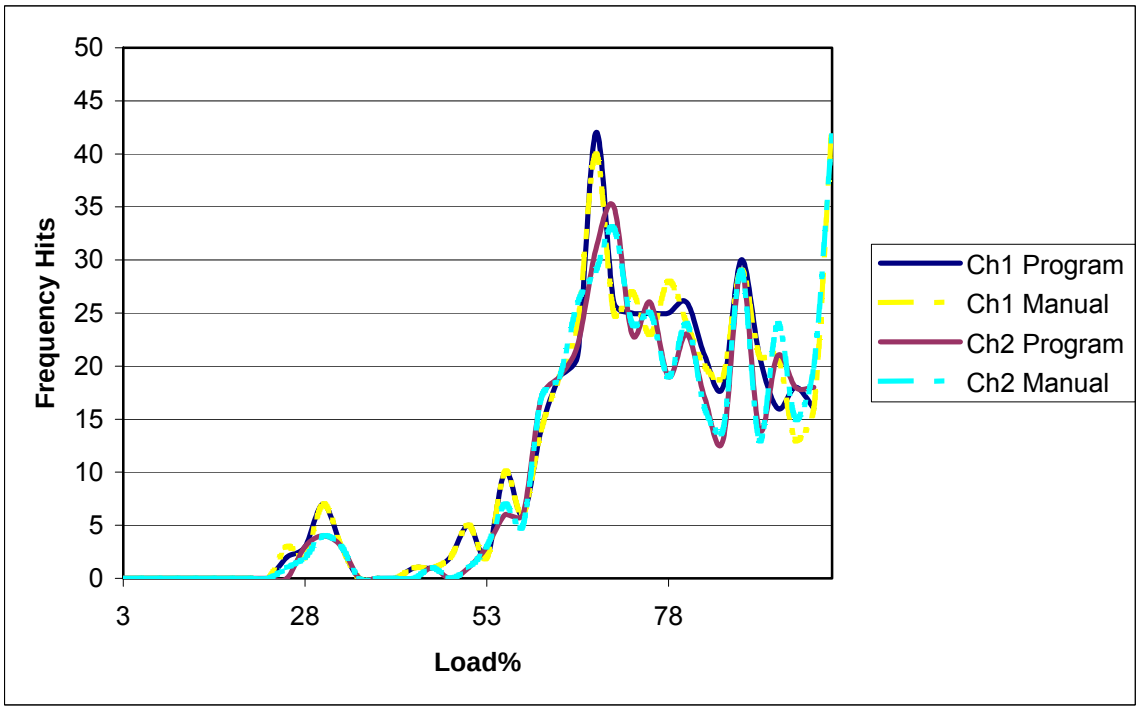


Figure 3.20 Test 1t, Frequency Hits vs. Load%

Chapter 4

Experiments and Results

4.1 Tension Test

4.1.1 Specimens

The two specimens used for the tension tests were prepared in the Chemical Engineering Laboratory using a hand lay-up technique (Yeh, 2003). These specimens, which have four layers, were made by Yeh (2003) using E glass as fiber and epoxy vinyl ester as resin. The resin was mixed with a powder of nano-engineered clay material. The specimens were then soaked in alkali solution for a period of 90 days. Research was conducted on the effect of alkali solution on the Fiber of the FRP material when nano-engineered clay was mixed with the resin (Yeh, 2003). Two specimens designated as Ten1 and Ten2 were tested to study the AE characteristics of FRP materials under tensile stress. The dimensions of specimen are: 7 inches in length, 1 inch in width and 0.04 inches in thickness. Each specimen had tabs attached on both ends for gripping. Two sensors were placed on each specimen, two inches apart and equidistant from the center. Two noise sensors were also installed for each experiment, one on each of the two loading grips. This allowed us to eliminate all the noise from the loading grips. The schematic of the tension specimen along with the sensors is shown in Figure 4.9.

A thin layer of hot melt glue was used as a couplant to glue the sensors to the specimen. A couplant is a substance that provides an acoustic link between the specimen and the sensor. The couplant also fills the microscopic gaps that

exist between the sensor and the specimen surfaces. It has been shown that a thin layer of hot melt glue attaches sensors efficiently and acts as a couplant without resulting in large material attenuation (Fultineer 1997).

4.1.2 Experimental Equipment

A hydraulic powered Instron 8501 loading frame was used to apply tension for the tests. An Instron 8500 plus controller was used to control the actuator of the loading frame. The Instron Loading frame, the hydraulic pump and the 8500 Plus controller can be seen in Figures 4.1, 4.2 and 4.3 respectively.

The AE data acquisition system used was the LOCAN AT system (Physical Acoustics Corporation). The LOCAN AT system can capture AE data with six channels. In addition, four parametric inputs like load and deflection can also be recorded. The LOCAN AT system can be operated through a DOS machine running SPARTAN 2000 software. The SPARTAN 2000 software saves AE data files to the DOS machine for future analysis. The LOCAN AT system can be seen along with the DOS machine in Figure 4.4. Piezoelectric sensors (Physical Acoustics Corporation) called PICO sensors, were used to record the AE from the specimens during the tests. The sensors measure 0.188 inches in diameter and 0.157 inches in height. The voltage signal produced by these piezoelectric sensors is amplified using a Model 1220A preamplifier in 40dB gain selection mode. After amplification, the signals are sent into the LOCAN AT system through the six available channels. The sensor and amplifier can be seen in Figures 4.5 and 4.6, respectively.

The LOCAN AT system was connected to the Instron computer controller in order to record the applied load. The loading was collected through the first parametric input and is measured in voltage; the voltage is then converted to load by using a multiplier and an offset. In order to scale this voltage output, the correct Multiplier and Offset numbers were calculated using the following equations (Arnold, 2003).

Multiplier = (L1-L2)/(V1-V2)

Offset = (V1*L2 - V2*L1)/(V1 – V2) = L1 – Multiplier * V1

Where,

L1, L2 = Loading (Load values shown in 8500 plus controller)

V1, V2 = Voltage (Parameter1 values shown in SPARTAN 2000)

These values were found to be 2.25 for multiplier and –0.03 for offset to convert voltage into load in kips.

Two tension tests were conducted in this study, Ten1 and Ten2. The first specimen, Ten1, failed at 4 kips, and the second specimen, Ten2, failed at 5.6 kips. Data from both the experiments was converted into ASCII text files from binary data files by using ATASC.exe. SPARTAN200 supplies the executable and after each experiment the data is given to this executable and the data in the text files is later analyzed and various graphs can be plotted using MS EXCEL application.

4.1.3 Software Setup

For each experiment to be conducted using SPARTAN 2000, the experimental settings should be entered in the application. Once the settings are created and saved as a .ini file, the initialization file can be loaded for similar experiments using the menu driven graphic user interface of SPARTAN 2000.

First, the channels used in the experiment are selected. For each channel selected, some settings should be made in order to filter out the experimental background noises. For our tension tests, a gain value of 20dB which is acceptable for medium and low sensitivity is selected. Medium and low sensitivity selection is best suited for composite materials.

A threshold value and type of threshold is selected for each channel. A fixed threshold value of 45dB is found to be best suited for our experiments. It should be noted that the gain plus threshold value must fall with in a range of 45-88dB; for the values we selected, the gain threshold is 65dB. A Pre-Amplitude gain value of 40dB is selected. After selecting the channels and setting the above described parameters, the parameters to be recorded during the experiment were chosen. Along with the six important AE parameters, namely Rise time, Count, Energy, Duration, Amplitude and Average Frequency, we also selected time of test, which is used for noise elimination as explained in Sections 3.3.3 to 3.3.6.

Parameteric-1 is selected to record load values from the 8500 plus controller. The multiplier, offset and units for the load parameter can also be

specified. The hardware settings explained above can be entered through the standard hardware setup screen shown in Figure 4.7.

After entering the standard hardware settings, some advanced hardware parameters should also be set. Peak Definition Time (PDT), Hit Definitions Time (HDT) and Hit Lockout Time (HLT) are three advanced channel settings, which are timing parameters for the signal measurement process (Arnold, 2003). PDT ensures that the signal peak for rise time measurements is correctly identified. The recommended value for PDT is 20-50 μsec for composites; a mid-range value of 35 μsec is selected for our experiments. The HDT parameter ensures that each AE signal is counted as one and only one hit. A value of 150 μsec is selected for HDT. It was selected from the recommended 100-200 μsec range for composites. For HLT input, the recommended value of 300 is used.

4.1.4 Noise Elimination

In the tension tests, the noise elimination algorithms developed in Sections 3.3.4 and 3.3.5 were used for the first time. A sensor was placed on each of the two loading grips while doing the tension experiments. Before conducting the tension experiments, pencil break tests were done on the specimens to establish the speed of AE waves in the material of the specimen. It was found that an AE wave takes 8 μsec to travel 1 inch, which is the distance between the loading grip and nearest material sensor. Therefore, if a signal is initiated at the loading grip, then the signal should take more than 8 μsec to reach the nearest noise sensor. The signal numbering algorithm is explained in detail in Chapter 3(see section 3.3.3).

Also it can be seen from Figures 4.23 and 4.24 that the noise signals have high duration and low amplitudes. The noise is also eliminated from the AE data by filtering out the long duration, low amplitude signals.

4.1.5 Frequency vs. Amplitude

The Frequency vs. Amplitude graph was plotted to study if previously reported patterns (Arnold, 2003) by the AE data are displayed as the specimen approaches failure. The AE data was divided into four quarters based on the load parameter. The data was split to study the AE data patterns in each quarter of loading. Very few signals were observed in the first two quarters. In the first quarter, the plot shows that all the signals recorded in this quarter had amplitude values less than 70dB. In the second quarter, there were comparatively more signals, but again, less than 5 signals were recorded which had an amplitude value higher than 80dB. In third and fourth quarters, more and more signals are recorded, and the number of signals with high amplitude value (70-100dB) was higher than the first two quarters. The patterns found in the frequency vs. Amplitude plots match patterns found previously by Arnold (2003). The increment in the number of high amplitude signals is a clear indication of fiber breakage in the specimen, which leads to the failure of specimen. The Frequency vs. Amplitude plots are shown in Figures 4.11 to 4.18.

4.1.6 Amplitude vs. Duration

The Amplitude vs. Duration plot is another very important information for analyzing AE data. These plots are shown for both Ten1 and Ten2 experiments

in Figures 4.19 to 4.28. As explained in Section 3.4.1, the data was first split into four quarters and then the graphs were plotted per each quarter of loading. Interesting trends were observed through these plots to separate the AE signals emitted by the composite material during different modes of failure. In the first two quarters of loading, the AE signals recorded had low amplitude values and comparatively very few signals were found to have high duration in these quarters. But, as the loading approaches the final quarter, a lot of signals can be seen with high amplitude and duration values. The trends shown in these plots match the similar trends shown by the experiments conducted by Arnold (2003). There were many short duration, high amplitude signals observed in Ten2. These signals can be attributed to the nano-clay material added to the resin of the composite material

4.1.7 Sum of Energy vs. Load

For the two tension specimens tested, summations of Energy vs. Load graphs were plotted. These graphs depict the level of signal activity at certain loads (Arnold, 2003).

The graphs show the increase in energy as the failure load approaches. It can be seen in both the graphs that the peak energy is shown before the failure load is reached. This high energy AE data before the failure load is reached can be interpreted as warning signals from the specimen before failure. The sum of energy vs. load graphs for Ten1 and Ten2 can be seen in Figures 4.29 and 4.30 respectively.

4.1.8 Data conversion to NN Input Matrix

The AE data obtained through tension experiments was run through the data conversion algorithms explained in Chapter 3. The load calibration and noise elimination parameters used for the program are shown in Table 3.5. In addition, a time parameter was also inputted for the noise elimination algorithm. A value of 8 μ sec was selected after a series of pencil break tests. The average values of AE parameters obtained were plotted against load % values. These plots can be seen in Figures 4.31 and 4.32 for experiment Ten1 and Ten2 respectively. For experiment Ten1, the plots go to zero often because of the fact that very few number of signals were recorded during the experiment. The failure of specimen was near sensor2, therefore sensor2 recorded a little higher duration and higher energy values as the load% approaches 100%. For experiment Ten2, the ultimate failure occurred near sensor1. Even though high energy signals were recorded by sensor1 around 45% of loading, the ultimate failure happened near sensor1, therefore as the load% approaches 100%, channel1 records high energy and longer duration signals.

The NN input matrix obtained from the program is used to predict the loading quarter of both the tension specimens. The predictions obtained from NN program are shown in Table 4.1. Prediction performance of the NN is determined by the Correct Rate (Explained in detail in Section 5.2.5). The NN program predicted at a Correct Rate of 25 for Ten1, but if the error value is observed for fourth quarter prediction, it is just 0.41. The NN prediction is valid if the error is $\pm$0.40. So, the prediction error in prediction of 4th quarter for Ten1 is negligible.

The second tension experiment Ten2, was predicted at a Correct Rate of 75, which is very successful prediction.

4.1.9 Discussion of Tension Test Results

It can be observed from the plots that the AE data recorded for specimen Ten1 was much less than that recorded for specimen Ten2. Also the specimen Ten1 failed at 4 kips while Ten2 was loaded up to 5.6 kips. Since both the specimens were manufactured using hand-lay up technique, Ten1 might have developed more voids while manufacturing. The presence of voids in the specimens might explain the lower number of AE signals since, the AE signals cannot travel through voids as effectively as it can travel through the composite matrix or fiber. The graphs showing Average AE parameter vs. Load % for specimen Ten1 drop to zero at a number of load % values. The reason for this is the lack of sufficient number of AE signals from the experiment.

For specimen Ten2, the Figure 4.32 shows that Channel 1 picked up a lot higher values of count and duration. The failure of specimen took place between sensor1 and the nearest loading grip, very close to sensor1. Count is the number of times a signal's amplitude crosses the preset threshold value. This means that count and duration are directly related to each other. This is the reason the average count and the average duration graphs reported higher values for channel1.

Because of the smaller number of AE signals recorded for Ten1, the Neural Network prediction achieved a low Correct Rate of 25%. But, for Ten2 the Neural Network prediction was good with a Correct Rate of 75%.

4.2 Bending Tests

4.2.1 Specimens

The three specimens used for bending tests were cut from a single module of prodeck4, which is a pultruded bridge deck structure manufactured by Bedford Composite which constitutes of Eglass fiber and epoxy vinyl ester. The specimen measures 29 inches in length, 2 inches in width and 4 inches in height. The thickness of the bridge deck material is 0.4 inches. The specimen has a box cross-section with webs 5.75 inches apart from each other, center to center. A picture of the prodeck4 component (single module) is shown in Figure 4.33.

The sensor placement plays a very important role in the success of AE detection. For a composite specimen having a complicated cross section, it is very difficult to predict the mode and location of major failure. It has been deduced that the specimen will fail either due to bending of top/bottom plates or the specimen might fail in shear at the webs. According to these deductions, four sensors were attached on each of the three bending test specimens. Three sensors were attached to the bottom surface of the specimen and one sensor was attached below the loading arm on the top plate of the specimen. A schematic of the specimen and sensor placement is shown in Figure 4.34. Three bending tests were conducted on the specimens described above. The first two specimens are referred as Bend1 and Bend2. These specimens were tested using three point bending methods and loaded until failure. With the third specimen, the effect of fatigue on the bridge deck coupon specimen was

investigated. One million cycles of loading was applied with an fmax value of 195 lbs (30% of ultimate strength) and a Pmin value of 62 lbs. The stress ratio, R, was 0.32 and the loading cycles were applied at a frequency of 1 Hz. Small visible cracks could be observed due to fatigue. This specimen referred as FatBend was then tested for AE characteristics using static three point bending test.

4.2.2 Experimental Equipment

INSTRON 1331 was used as a loading frame for these bending tests. An MTS hydraulic pump with a capacity of 3000 psi is used for hydraulic pressure generation for the actuator. A maximum load of 10 kips can be applied using this loading frame. The maximum displacement of the frame is 5 inches. An INSTRON load cell is used to measure the load that is being applied. Loading of the frame is controlled either by load or displacement. Figure 4.36 shows the INSTRON loading frame, actuator, controller and Hydraulic pump.

The loading controller used was an MTS 407 controller, which is a single channel, digitally-supervised servo controller. The 407 controller provides complete control of the test system hydraulics. The 407 can be used to control a hydraulic power supply and a hydraulic service manifold to apply low and high hydraulic pressure to the test system. High hydraulic pressure was used to load the bending specimens tested in this study. The controller is equipped with data output sockets which were used to take the load data into the LOCAN AT system.

To collect displacement values for bending tests, a 2000 HR-DC LVDT was used. The LVDT has a nominal linear range of ± 2 inches. The LVDT is powered by using a BK precision model 1660 triple output DC power supply. The LVDT is connected to a LOCAN AT system through a D15 input port named Parametric. The LVDT was calibrated with the LOCAN AT system. A curve is plotted with the values to find the multiplier (slope of curve) and offset (intercept) values. These values are entered into hardware setup screens of LOCAN AT. The multiplier was found to be 0.3044 and the intercept was 0.0728.

In order to conduct three point bending tests on the large specimens shown in Figure 4.34, existing loading supports could not be used. A previously fabricated loading support which has a fixed support arrangement on one end and a roller support arrangement on the other end was used. The support was modified a little in order to suit our needs. Two angle brackets were fitted to the bottom surface of the support. A circular shaft with keyway hole is welded at the bottom of the brackets. A small bolt was welded to the brackets to hold the LVDT. The fabricated support along with a loaded specimen can be seen in Figure 4.35.

4.2.3 Software Setup

All the software settings explained in Section 4.1.3 are repeated for bending experiments as well, except for some additions and changes were made since the INSTRON machine & controller being used for bending is different. The multiplier and offset value had to be recalculated. The same method as explained in Section 4.1.3 was used to determine these values. The multiplier was found to

be -100 and the offset was 3.31. The parameter 2 value was also selected in the hardware setup screen in order to record the deflection values from LVDT.

4.2.4 Frequency vs. Amplitude

The Frequency vs. Amplitude graphs for Bend1 and Bend2 display very clear patterns. Figures 4.38 to 4.45 show the Frequency vs. Amplitude plots for Bend1 and Bend2. It has been reported in previous studies (Arnold, 2003) that in Frequency vs Amplitude plots, many signals appear in the area bounded by amplitude greater than 70dB and frequency range between 100 kHz-400 kHz as the specimen approaches failure. It can be seen from the plots for tests Bend1 and Bend2 that, in third quarter especially, many signals appear in the above mentioned area.

Frequency vs. Amplitude graphs are also plotted for the four quarters of the specimen FatBend. This specimen as explained in Section 4.2.1 was first loaded with about a million cycles of fatigue. The Frequency vs. Amplitude plots for FatBend are shown in Figures 4.46 to 4.49. These plots show a number of signals in the 3rd and 4th quarters. High amplitude signals appear more in the 4th quarter than in the 3rd quarter. In the 1st and 2nd quarters, there are less than 10 signals per quarter. The first two quarters for a bending specimen, usually consists of signals emitted due to matrix cracking (Arnold, 2003). Since, the specimen was first loaded with a million cycles of fatigue loading before the bending tests, the matrix cracking in the specimen has already occurred,

explaining the low number of signals in first two quarters of loading from FatBend specimen.

4.2.5 Amplitude vs. Duration

The Amplitude vs. Duration plots also show clear patterns as the specimen approaches failure. These plots for tests Bend1 and Bend2 are shown in Figures 4.50 to 4.57. The plots for Bend1 and Bend2 are in good agreement with previously reported results (Arnold, 2003). The signals start at 0 μ sec and 40dB for first two quarters. As the loading progresses to the 3rd and 4th quarters, the signals explode outward on the graph. Both the tests show the maximum number of signals in 3rd quarter.

Amplitude vs. Duration plots for test FatBend are shown in Figures 4.58 to 4.61. As explained in the previous section, due to the fatigue loading applied on the specimen, the tests shows less than 10 signals in the first two quarters. In the third quarter, a considerable number of signals can be seen in the high duration region. The fourth quarter shows the maximum number of signals with numerous long duration and high amplitude signals.

4.2.6 Sum of Energy vs. Load

The Summation Energy vs. Load plots for specimen Bend1 and Bend2 are shown in Figure 4.62 and Figure 4.63 respectively. The slope of the graph decreases at the end of loading at about 410 lbs for both Bend1 and Bend2.

The Summation Energy vs. Load plot for specimen FatBend is shown in Figure 4.64. As can be observed from the graph, the energy emitted in the form of AE signals is zero until the load value reaches 400lbs. This is because the

specimen has already lost that energy in form of matrix cracking when it was under fatigue loading. There is a steady increase of energy as the load increased from 400lbs to 600lbs. From 600lbs to 650lbs, the slope of graph decreases because there are a smaller number of strong signals during that loading period.

4.2.7 Load vs. Deflection

Deflection for bending specimens was measured as explained in section 4.2.2. The deflection measured is the maximum deflection observed at mid-span of the specimen. The deflection values obtained are plotted against load values for all three bending tests conducted. The plots are shown in Figures 4.65 to 4.67.

As can be seen from figures, specimen Bend1 maintains a steady load-deflection curve, until the load value reaches the ultimate failure value (633.31lbs). The curve becomes a straight line after reaching the failure load, and the deflection increases from 1 inch to 1.3 inches. Load vs. Deflection curve for specimen Bend2 also exhibits the same trend as Bend1. Except for a drop in slope of the curve before 600lbs. This can be attributed to a big crack formed in the specimen before the ultimate failure. For specimen FatBend, it can be observed that slope of the curve is greater than the previous specimens. Also the maximum deflection is about 1.6 inches, which is greater than the deflection of Bend1 and Bend2. The reason being, a million cycles of fatigue were already applied on this specimen.

4.2.8 Data Conversion to NN Input Matrix

The AE data obtained through bending experiments Bend1 and Bend2 was given to the data conversion program explained in Chapter3. The program parameters used are shown in Table 3.3. The failure loads for Bend1 and Bend2, which are 633.31lbs and 643.31lbs are also given as inputs to the program. The average AE parameters values obtained were plotted against load%. These plots are shown in Figures 4.68 and 4.69. The graphs are comparable to similar graphs plotted in Chapter3, Figure 3.11 to 3.15.

The NN input matrix obtained from the program is used to predict the loading quarters of Bend1 and Bend2. The predictions obtained from NN program are shown in Table 4.2. The loading quarters for both the specimens were predicted at a Correct Rate of 50%. As can be seen from the Frequency vs. Amplitude plots and Amplitude vs. Duration plots, there was a maximum number of strong AE signals in the 3rd loading quarter. The NN program correctly predicted the 3rd loading quarter for both Bend1 and Bend2. This means that the specimen has given a lot of warning signals in the third quarter.

4.2.9 Discussion of Bending Tests Results

All the specimens failed due to the shear stress induced by the webs close to the supports. The failure was in form of a big crack appearing at the connection of the web and the bottom plate of the specimen. Even though Channels 3 and 4, which are installed at the middle of specimen, (see Figure 4.34) received some signals, Channels 1 and 2 were the main source of data. Since, Channels 1 and 2 are closer to the failure, they received stronger and

more numerous signals. Therefore, Channels 1 and 2 were used for data analysis and NN input matrix generation. Specimens Bend1 and Bend2 failed at 633.31lbs and 643.31lbs, respectively. In FatBend, fewer than 10 signals were recorded until the load reached 400lbs; this was because the specimen was previously loaded with a million cycles of fatigue.

The strong patterns shown by AE data in various plots are in good agreement with previously reported (Arnold, 2003) trends. The trending of average AE parameter vs. Load% graphs is comparable to those shown in Chapter3. The NN program could correctly predict the third loading quarter. This shows that AE testing along with the data conversion program and NN program can be used for non-destructive structural evaluation of FRP materials.

4.3 Fatigue Tests

4.3.1 Specimens and sensor placement

Two fatigue tests were conducted on specimens, which were single cell square cross sectional components of a 6' FRP bridge deck. The bridge deck was manufactured using the pultrusion technique at Bedford Composites Inc. The constituent materials used for the construction of the deck are E Glass fiber and Epoxy Vinyl Ester resin. The single cell component measures 6' in length, 6.22" width and 4" in height. The thickness of the FRP material is 0.4"

The sensors placement was decided by observing the failure locations in similar specimens previously tested under fatigue. The common failure observed in the specimen was delamination of the top plate of the square cross sectional

specimen. The failure location was observed at a distance of 1 foot from the loading point. Four sensors were used to record the AE signals from the material. The four sensors were placed on the side plates of the specimen, 18” from the center of the specimen. A schematic of a fatigue specimen and sensor placement is shown in Figure 4.70.

Two Fatigue tests were conducted on the specimen. The ultimate failure load for the single cell was determined to be 20kips. The first fatigue test was conducted with a Pmax value of 10kips (50% of ultimate strength) and a Pmin value of 1 kip. The stress ratio is 0.1. The cycles were applied at a frequency of 1 Hz. The second Fatigue test was conducted with a Pmax value of 14kips (70% of ultimate strength), and at a stress ratio of 0.1, Pmin was selected to be 1.4 kips. A static load of 14kips was applied on the second specimen before fatigue. After fatigue was done, the static load was applied on the second specimen until failure which happened at 13kips. The first two tests are referred as fat1 and fat2. The Bending test conducted before fatigue is referred as BendFat, and the bending test after fatigue is called FatBend1.

4.3.2 Experimental Equipment

A 55 kip MTS actuator was used to load the specimen. The actuator was supported from a steel frame bolted to the floor. Two concrete blocks with roller arrangement were used as supports for the specimen Figure 4.70.

The actuator was powered by a Series 505 Silent Flow Hydraulic Power Unit from MTS. The Hydraulic Power Units (HPU) produces the high-pressure hydraulic fluid for test system operation. The six pump assembly in the HPU

draws hydraulic fluid from the reservoir and pressurizes it to a maximum of 3000psi. The manifold of the HPU combines the output of separate pumps to deliver the full output of the HPU through a single port. The manifold provides solenoid control of the high/low pressure output from the individual pumps. The manifold also contains relief values for each pump circuit and a bypass circuit to maintain the hydraulic temperature during low flow conditions.

The controller used to control the loading of the actuator was MTS 407, which is explained in Section 4.2.2. A function generator module of the MTS 407 controller was used to apply the cyclic loading of the specimen. The function generator has a built in cycle counter, which is passed to the LOCAN AT machine along with the load parameter. Different types of loading functions can be selected through the function generator. A sine wave form with 1Hz frequency was selected for our experiment.

4.3.3 Software Setup

The software setup in SPARTAN 2000 is the same as explained in Section 4.2.3, except that to capture the cycle number from the 407 controller, the CycleCounter (CC) parameter was selected in the SPARTAN 2000 hardware settings screen, shown in Figure 4.7.

4.3.4 Frequency vs. Amplitude

The Frequency vs. Amplitude graphs were plotted for Fat1 and Fat2 after dividing the data in four quarters based on the number of cycles. The Frequency vs. Amplitude plots are shown in Figures 4.74 to 4.81. It can be observed from

the plots that for the 3rd and 4th quarters, a similar trend is shown as that of the bending tests.

For the test BendFat and FatBend1, the Frequency vs. Amplitude graphs can be seen in Figures 4.82 to 4.89. For the test BendFat, the first quarter is from 0 to 3.5 kips. This is not corresponding to 25% of ultimate loading. As the specimen was only loaded upto 70% of ultimate load, the first two quarters show the signal recorded when loaded from 0 to 7 kips, which is only about 35% of the ultimate load, so, the test BendFat does not show a large number of signals that can be seen in the first two quarters. But, the 3rd and 4th quarters for both BendFat and FatBend1 tests show a lot of high amplitude signals.

4.3.5 Amplitude vs. Duration

The Amplitude vs. Duration graphs for Fat1 and Fat2 are shown in Figures 4.90 to 4.97. The Amplitude vs. Duration plots for fatigue show a number of long duration signals in the fourth quarter, as the specimen approaches failure. The Amplitude vs. Duration plots for BendFat and FatBend1 tests are shown in Figures 4.98 to 4.105. Due to the reasons explained in a previous section, not many signals were recorded in the first two quarters. However, a good amount of warning signals can be observed in the 3rd and 4th quarters, especially on FatBend1. This is because the specimen Fat2 failed completely while applying static load after the fatigue, Therefore FatBend1 shows a lot of warning signals before the ultimate failure.

4.3.6 Sum of Energy vs. No of Cycles

The summation energy vs. number of cycles graphs are plotted for Fat1 and Fat2, they are shown in Figures 4.106 and 4.107 respectively.

An important observation that can be made from the graph plotted for Fat1 is, the change in slope of the graph clearly show three modes of failure of the specimen. From 0 to 400 cycles, the slope of the graph depicts the AE energy emitted from the specimen due to matrix cracking. The slope of the graph decreases from 400 to 1200 cycles. This means that the material emitted less amount of energy in this range. In the final phase, from 1200 to 1524 cycles, the ultimate failure of the specimen occurred due to delamination of top plate of the specimen and due to failure of the side plates of the specimen as shown in Figure 4.73 (a). The observations observed for Fat1 cannot be seen for Fat2. Since, the specimen failed after just 208 cycles. However, a sharp increase of AE energy level starting at around 198 cycles is a clear indication of warning signals before the final failure.

4.3.7 Data Conversion to NN Input Matrix

The data conversion program was executed for the data collected from Fat1 and Fat2 tests. Instead of failure load, failure cycle number is inputted into the program. The Fat1 failed at 1524 cycles and for Fat2, failure occurred after 208 cycles. The average AE parameters from the four channels are plotted against the % number of cycles. These plots can be seen in Figures 4.110 to 4.1113. The basic trend of the AE parameters observed for similar graphs from bending tests can also be seen in the fatigue specimens. As can be seen from

Figures 4.73 (a) and (b) for Fat1 and Fat2 respectively, the channels 1 and 3 received high more number of signals than 2 and 4, the failure of Fat1 is shown in Figure 4.73(a) . specimen Fat2 failed due to delamination of top plate above sensors 2 and 4 as shown in Figure 4.73(b), that is the reason for high number of signals recorded by those sensors.

4.3.8 Discussion of Fatigue Test Results

The Fatigue failure occurred in both specimens due to deterioration of top surface plate about 1 foot from the loading plate. Some cracks were observed in the side plates directly below the loading plate. By looking at trends shown by the Average Frequency vs. Amplitude, Amplitude vs. Duration and Energy vs, No of Cycles plots, the FRP bridge deck's final failure can be warned or even predicted using these AE parameters. More specimens should be tested for fatigue to confirm this observation.

The specimens failed after a much lower number of cycles of fatigue than expected. The ultimate failure load selected for the specimen was determined by static tests conducted on similar specimens by other researchers. The actual ultimate failure load of the specimens Fat1 and Fat2 could be lower than 20 kips, which means, the Pmax values selected for Fat1 and Fat2 are more than 50% and 70% of the actual ultimate load. This is probably why the specimens failed within very limited number of cycles.

The NN program can be trained to predict the loading stage of FRP bridge decks under fatigue, this will be explained in detail in next chapter.

4.4 Summary

- Two tension tests were conducted on FRP samples made by using a resin mixed with nano-engineered clay material.
- Observation of many short duration signals having high amplitude values in the *Amplitude Vs. Duration* plots can be thought of as increased brittleness in the FRP material due to the addition of nano-engineered clay material in the resin. More experiments should be conducted to confirm this.
- Three bending tests were conducted on 2 inch wide Prodeck4 samples. The samples failed in shear stress induced by the webs close to the supports.
- The Neural Network program correctly predicted the third loading quarter for all the samples. This proves that AE testing along with the application developed, and the NN program can be used for non-destructive structural evaluation of FRP materials.
- Two fatigue tests were conducted on 6 feet long, single celled, square cross sectioned FRP bridge decks.
- AE parameters could predict the final failure of the fatigue specimens.

Ten1		
QR	Predicted	Error
1	1.76	-0.76
2	1.67	0.33
3	1.88	1.12
4	3.59	0.41
CR	25	

Ten2		
QR	Predicted	Error
1	2.02	-1.02
2	2.38	-0.38
3	3.01	-0.01
4	3.63	0.31
CR	75	

Table 4.1 Neural Network predictions of tension tests

Bend1		
QR	Predicted	Error
1	1.58	-0.58
2	3.16	-1.16
3	3.13	-0.13
4	3.71	0.29
CR	50	

Bend2		
QR	Predicted	Error
1	0.98	0.02
2	2.78	-0.78
3	3.40	-0.40
4	3.43	0.57
CR	50	

Table 4.2 Neural Network predictions of bending tests

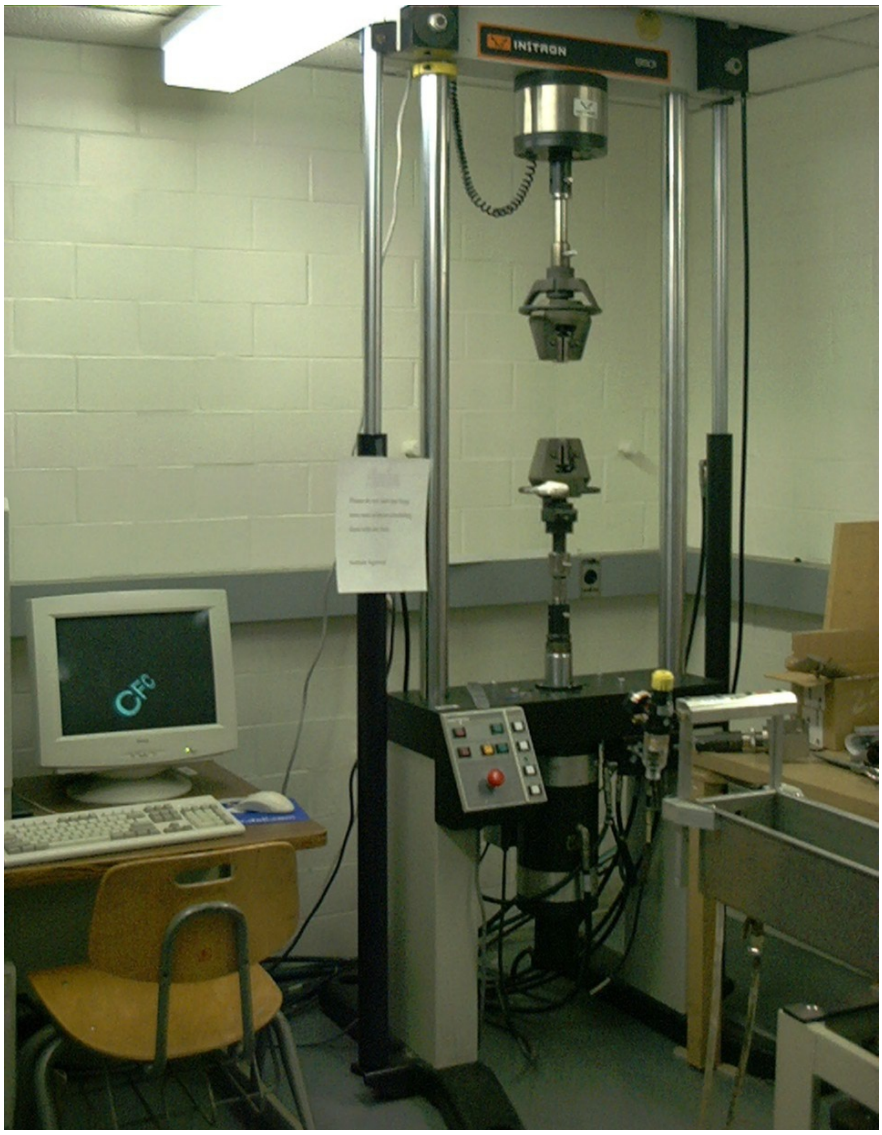


Figure 4.1 INSTRON 8501 Loading Frame, Load Cell, and Actuator

Figure 4.2 Hydraulic Power Supply

Figure 4.3 INSTRON 8500 Control Panel and PC

Figure 4.4 LOCAN AT and PC

Figure 4.5 PICO Sensors

Figure 4.6 Model 1220A Preamplifier

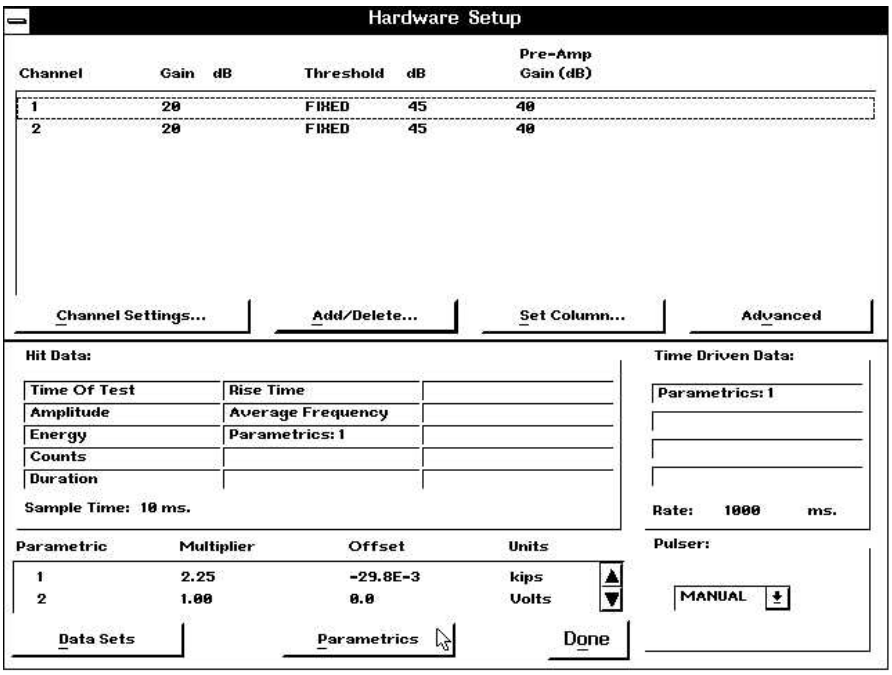


Figure 4.7 Hardware Setup Screen

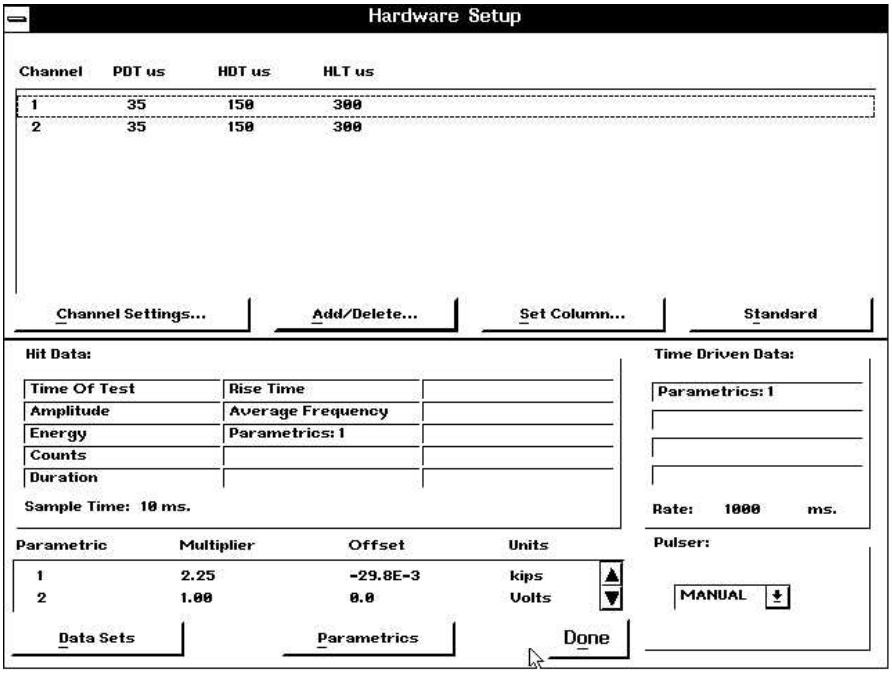


Figure 4.8 Advanced Hardware Setup Screen

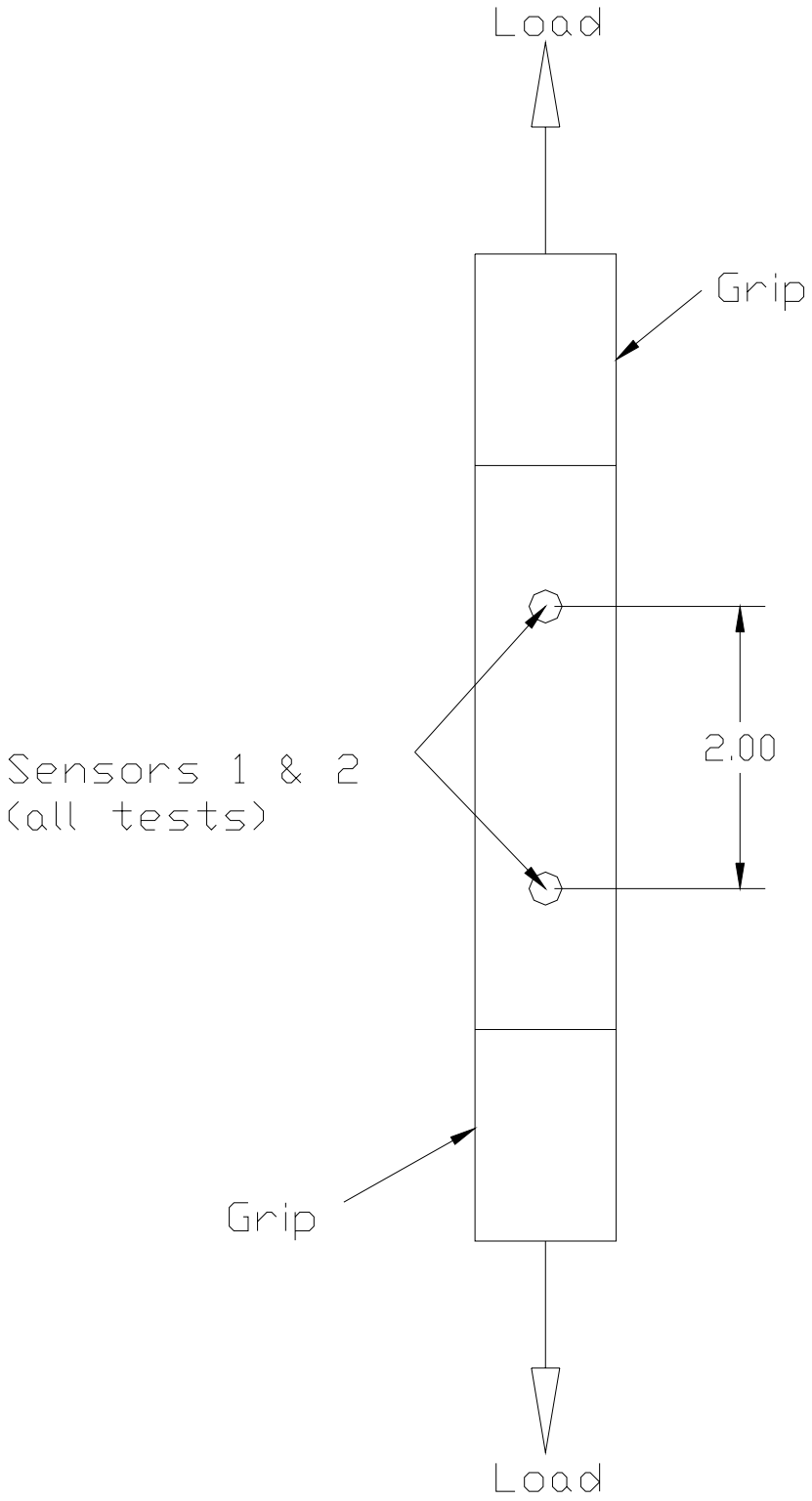


Figure 4.9 Sensor Placement

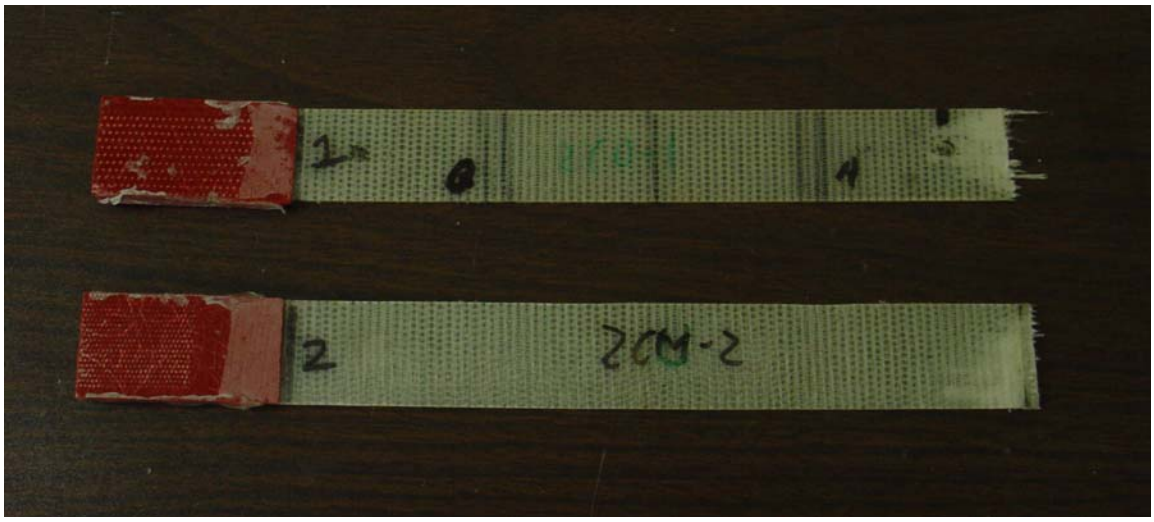


Figure 4.10 Tension specimens after failure

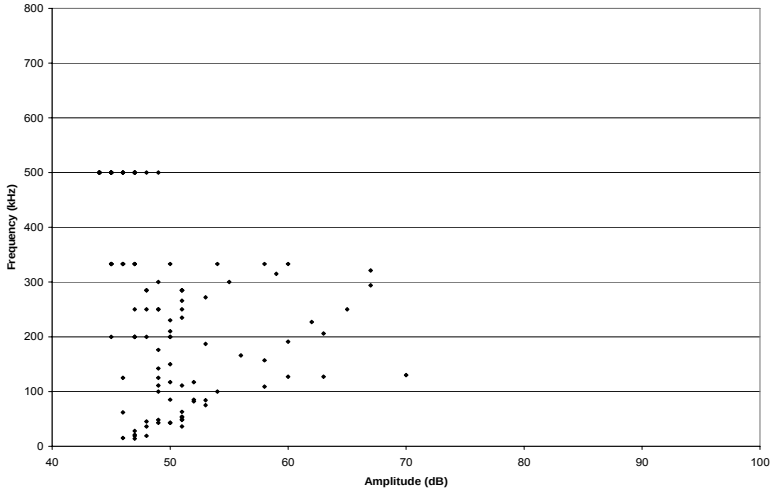


Figure 4.11 Frequency vs. Amplitude, Ten1 (0 – 1 kip),1st Quarter

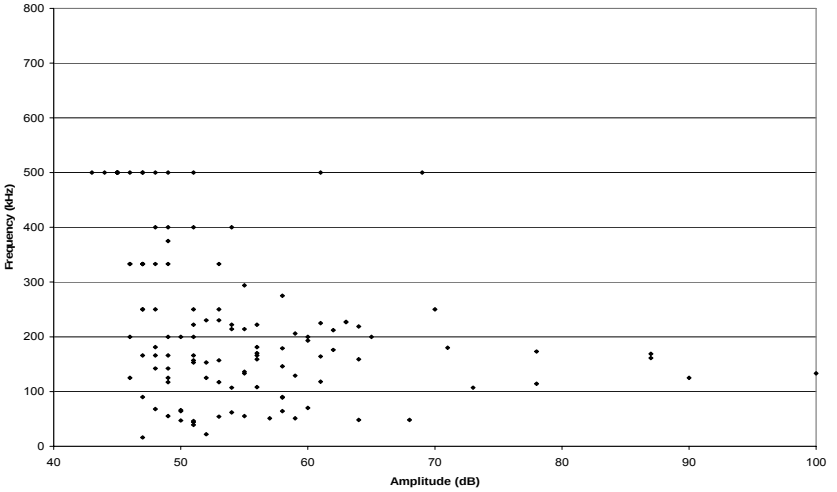


Figure 4.12 Frequency vs. Amplitude, Ten1 (1 – 2 kip), 2nd Quarter

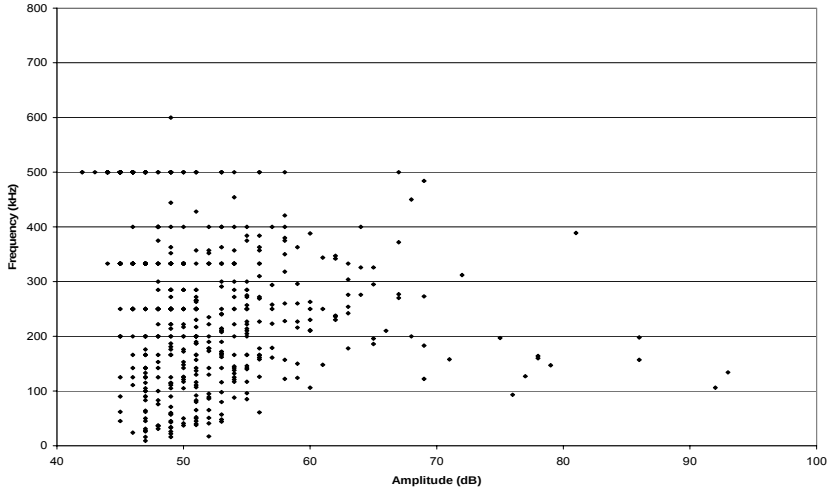


Figure 4.13 Frequency vs. Amplitude, Ten1 (2 – 3 kip), 3rd Quarter

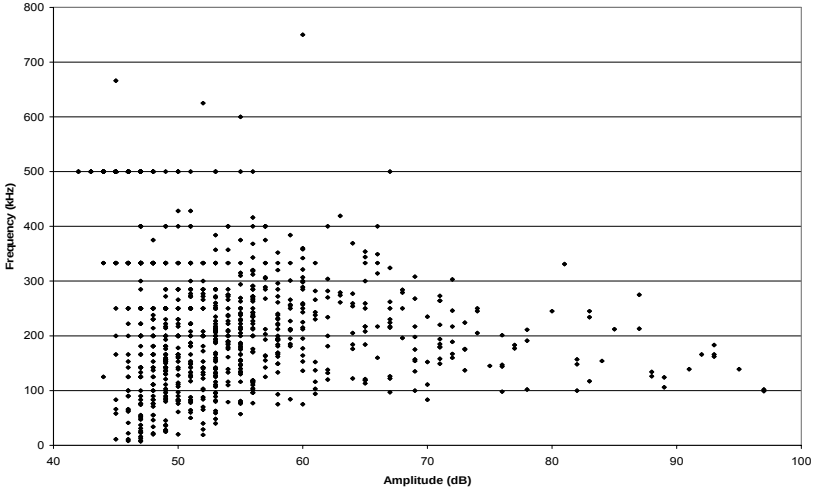


Figure 4.14 Frequency vs. Amplitude, Ten1 (3 – 4 kip), 4th Quarter

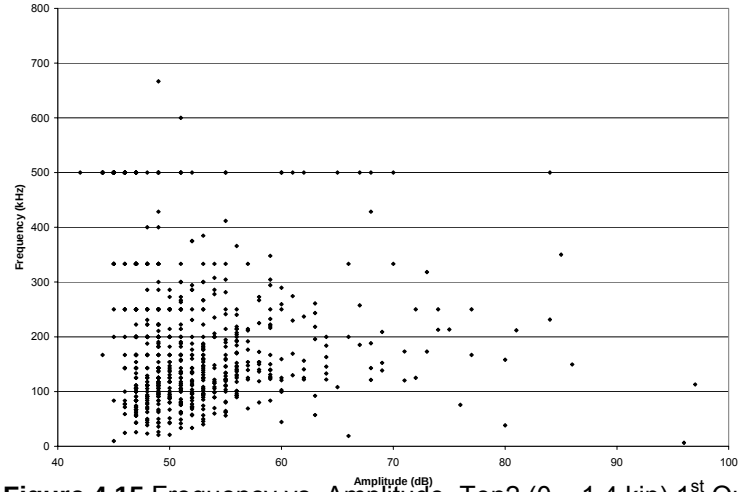


Figure 4.15 Frequency vs. Amplitude, Ten2 (0 – 1.4 kip),1st Quarter

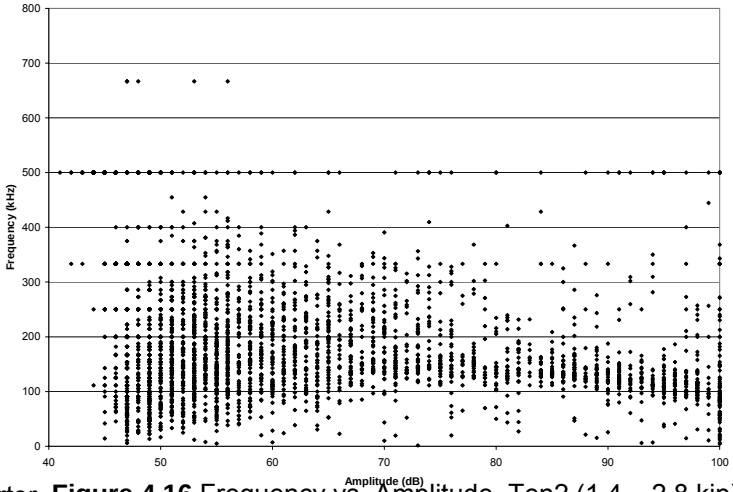


Figure 4.16 Frequency vs. Amplitude, Ten2 (1.4 – 2.8 kip),2nd Quarter

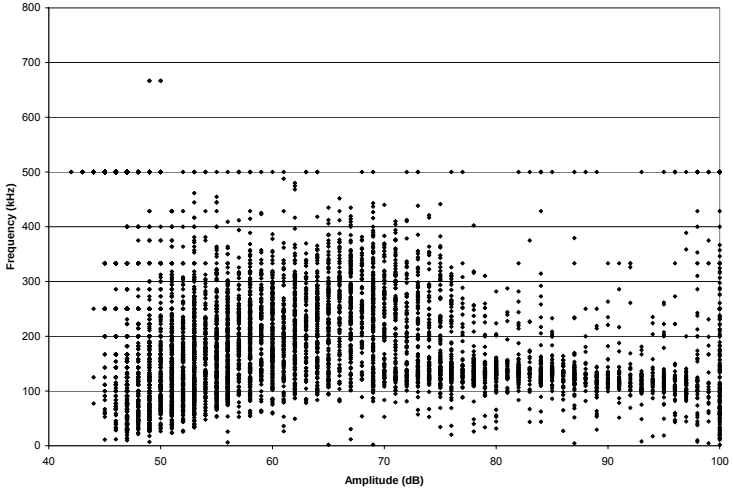


Figure 4.17 Frequency vs. Amplitude, Ten2 (2.8 – 3.2 kip),3rd Quarter

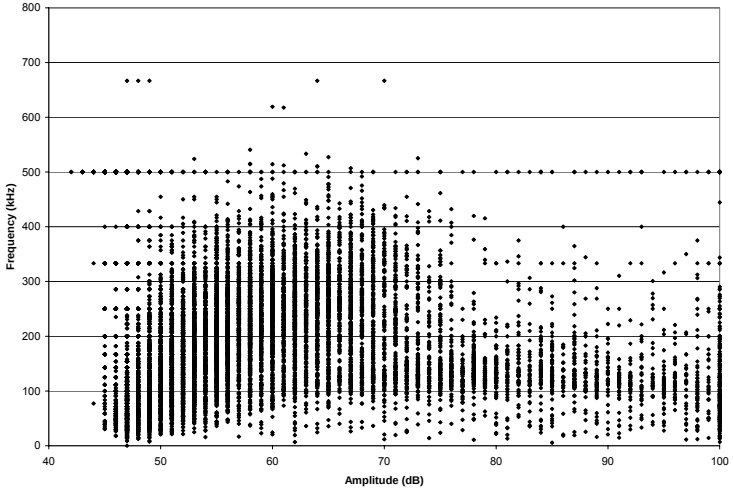


Figure 4.18 Frequency vs. Amplitude, Ten2 (3.2 – 4.6 kip),4th Quarter

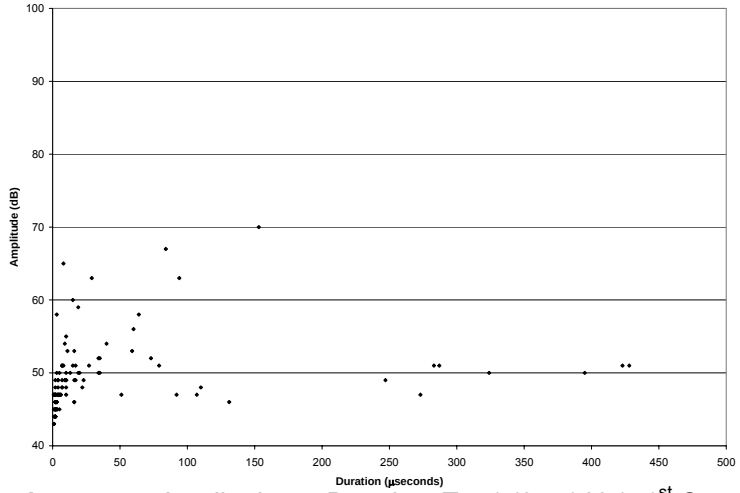


Figure 4.19 Amplitude vs. Duration, Ten1 (0 – 1 kip), 1st Quarter

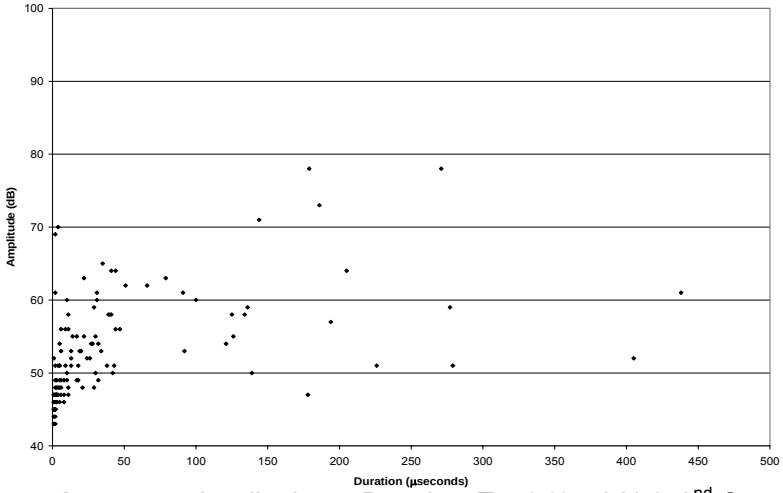


Figure 4.20 Amplitude vs. Duration, Ten1 (1 – 2 kip), 2nd Quarter

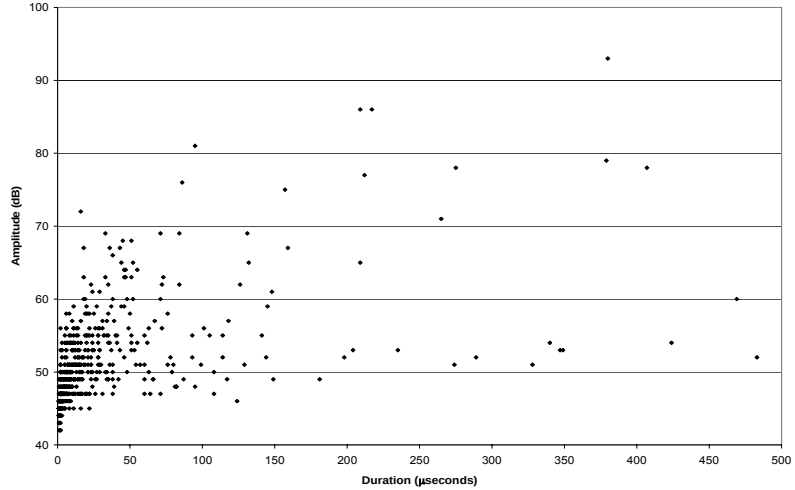


Figure 4.21 Amplitude vs. Duration, Ten1 (2 – 3 kip), 3rd Quarter

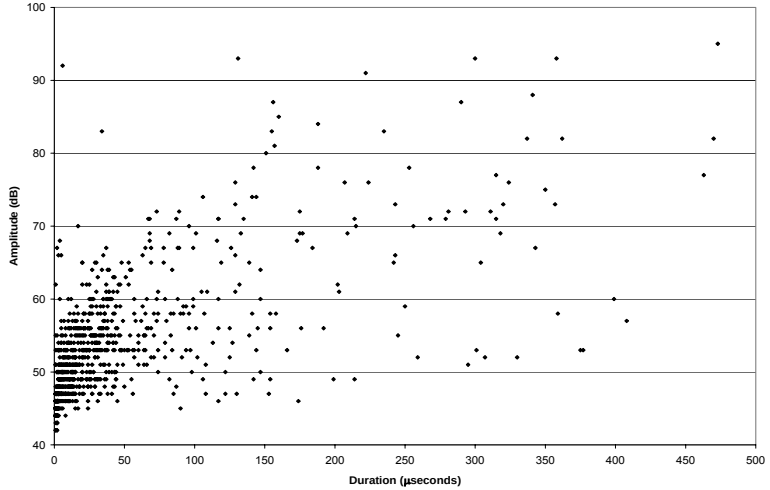


Figure 4.22 Amplitude vs. Duration, Ten1 (3 – 4 kip), 4th Quarter

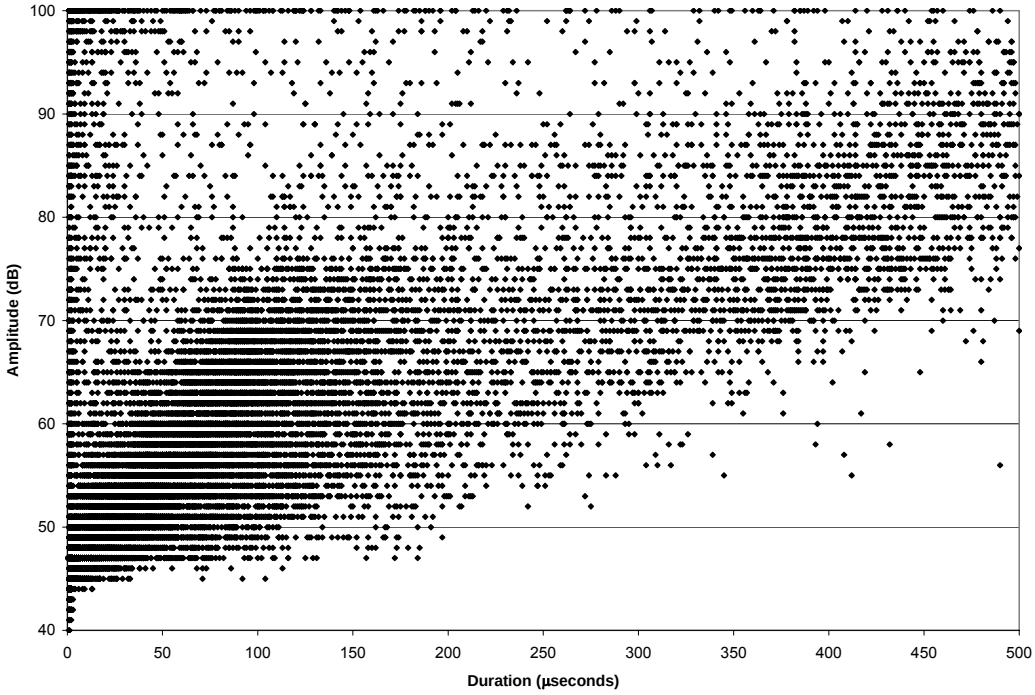


Figure 4.23 Amplitude vs. Duration of Material sensors,Ten2

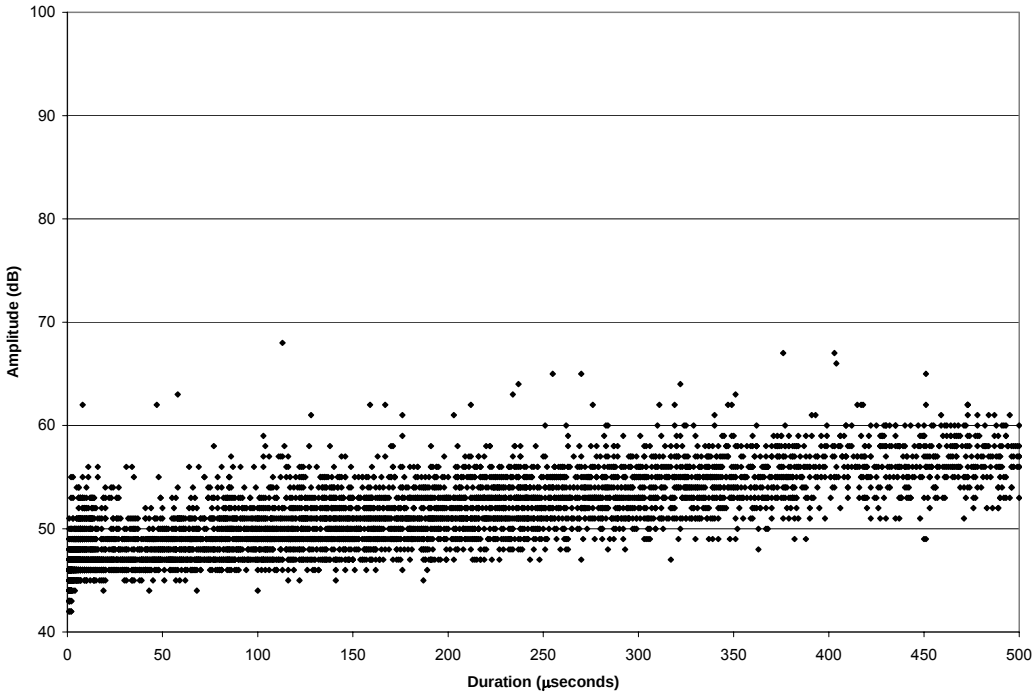


Figure 4.24 Amplitude vs. Duration of Noise sensors

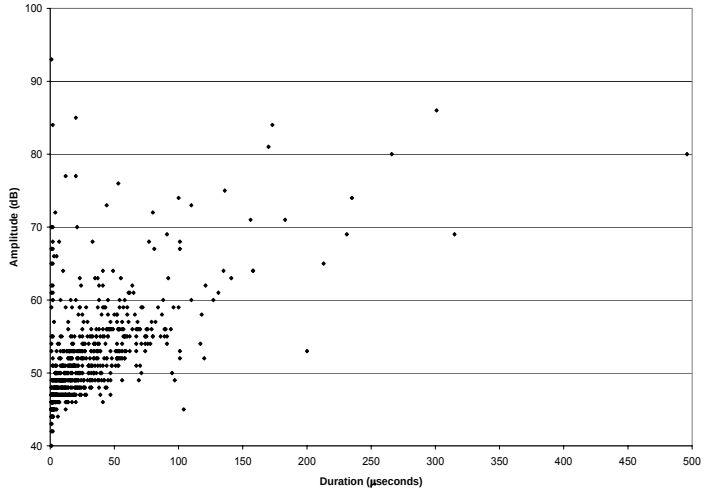


Figure 4.25 Amplitude vs. Duration, Ten2 (0 – 1.4 kip), 1st Quarter

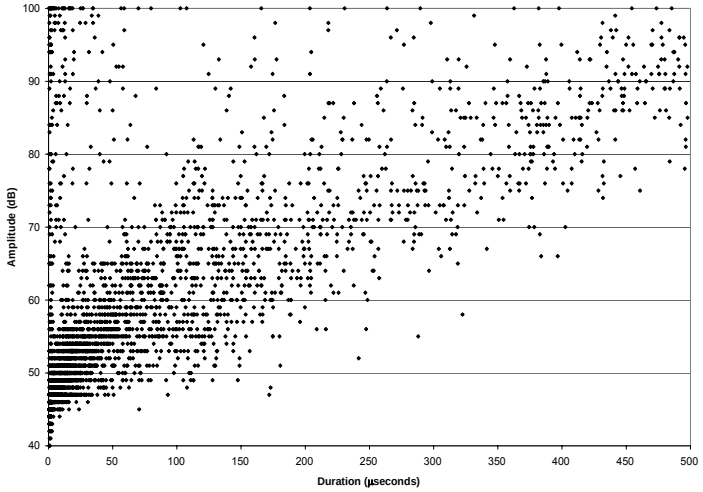


Figure 4.26 Amplitude vs. Duration, Ten2 (1.4 – 2.8Kip), 2nd Quarter

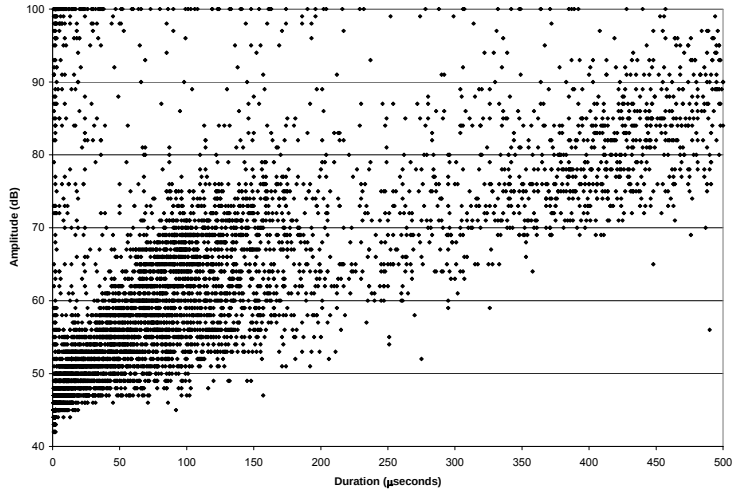


Figure 4.27 Amplitude vs. Duration, Ten2 (2.8 – 3.2kip), 3rd Quarter

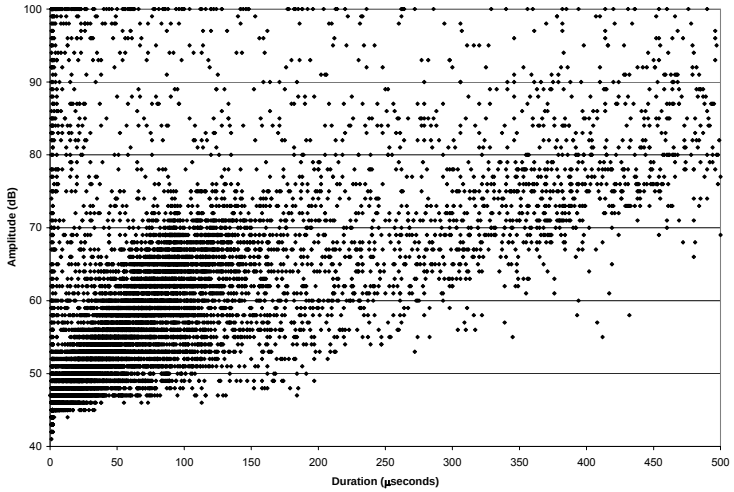


Figure 4.28 Amplitude vs. Duration, Ten2 (3.2 – 5.6kip), 4th Quarter

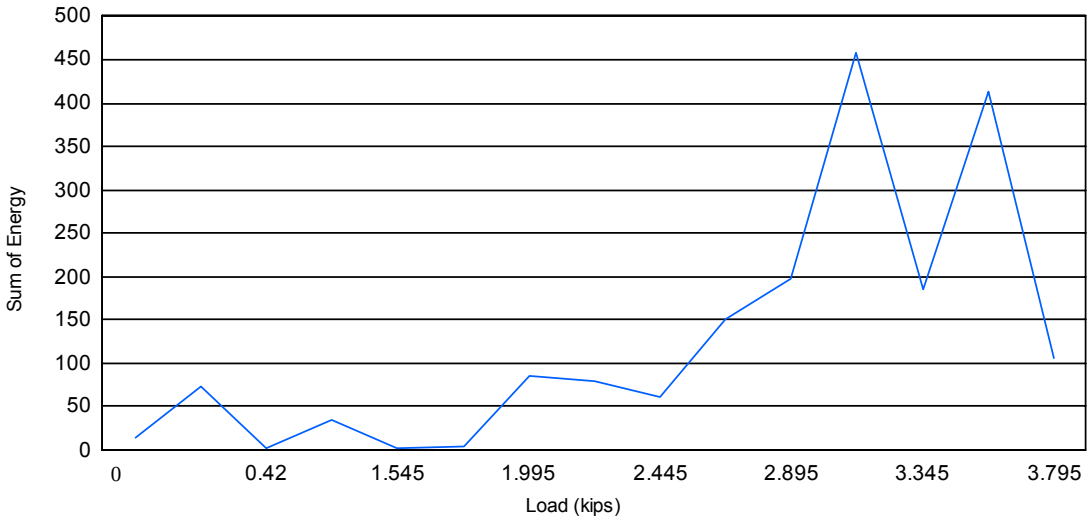


Figure 4.29 Sum of Energy vs. Load, Ten1

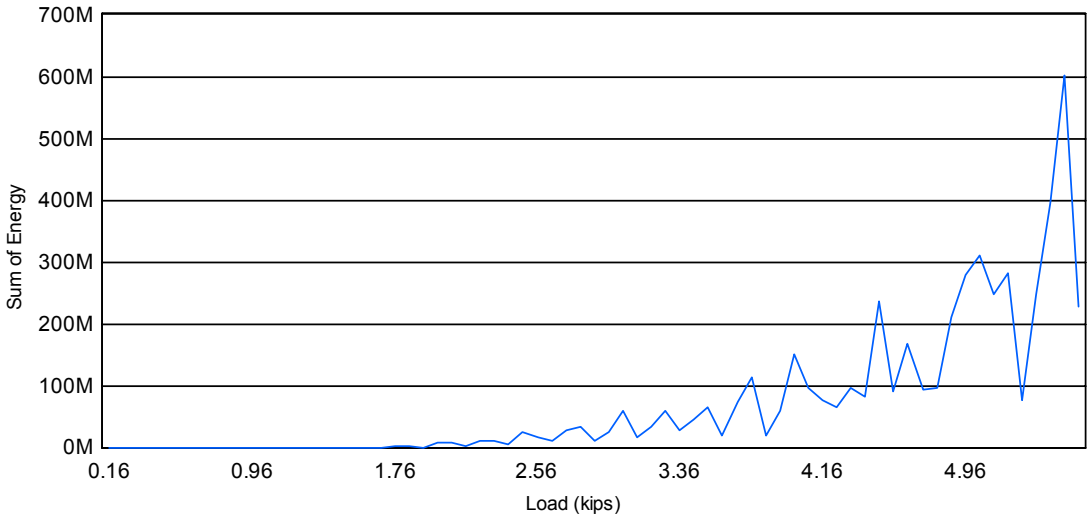


Figure 4.30 Sum of Energy vs. Load, Ten2

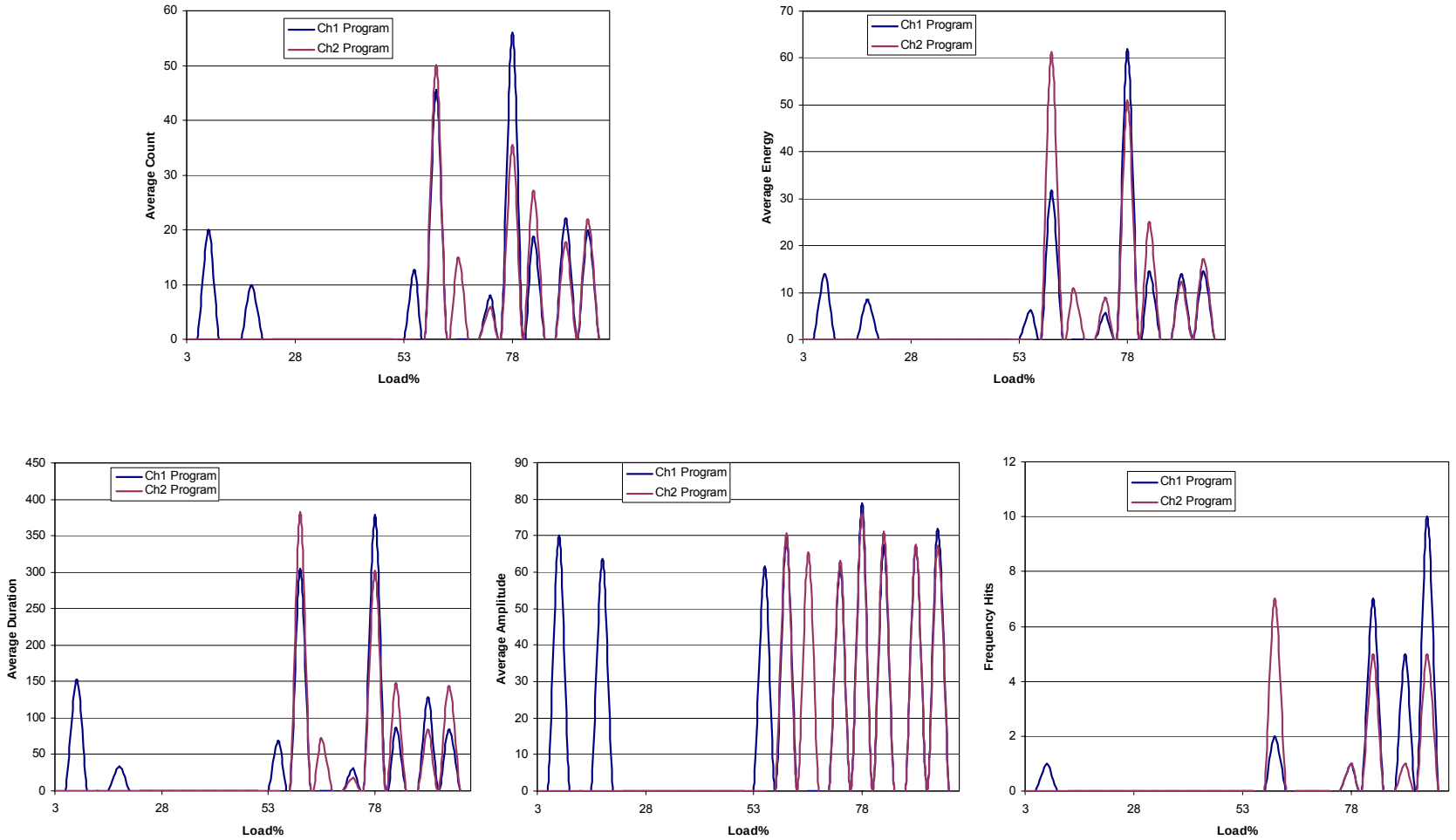


Figure 4.31 Test Ten1, Average AE parameters vs. Load%

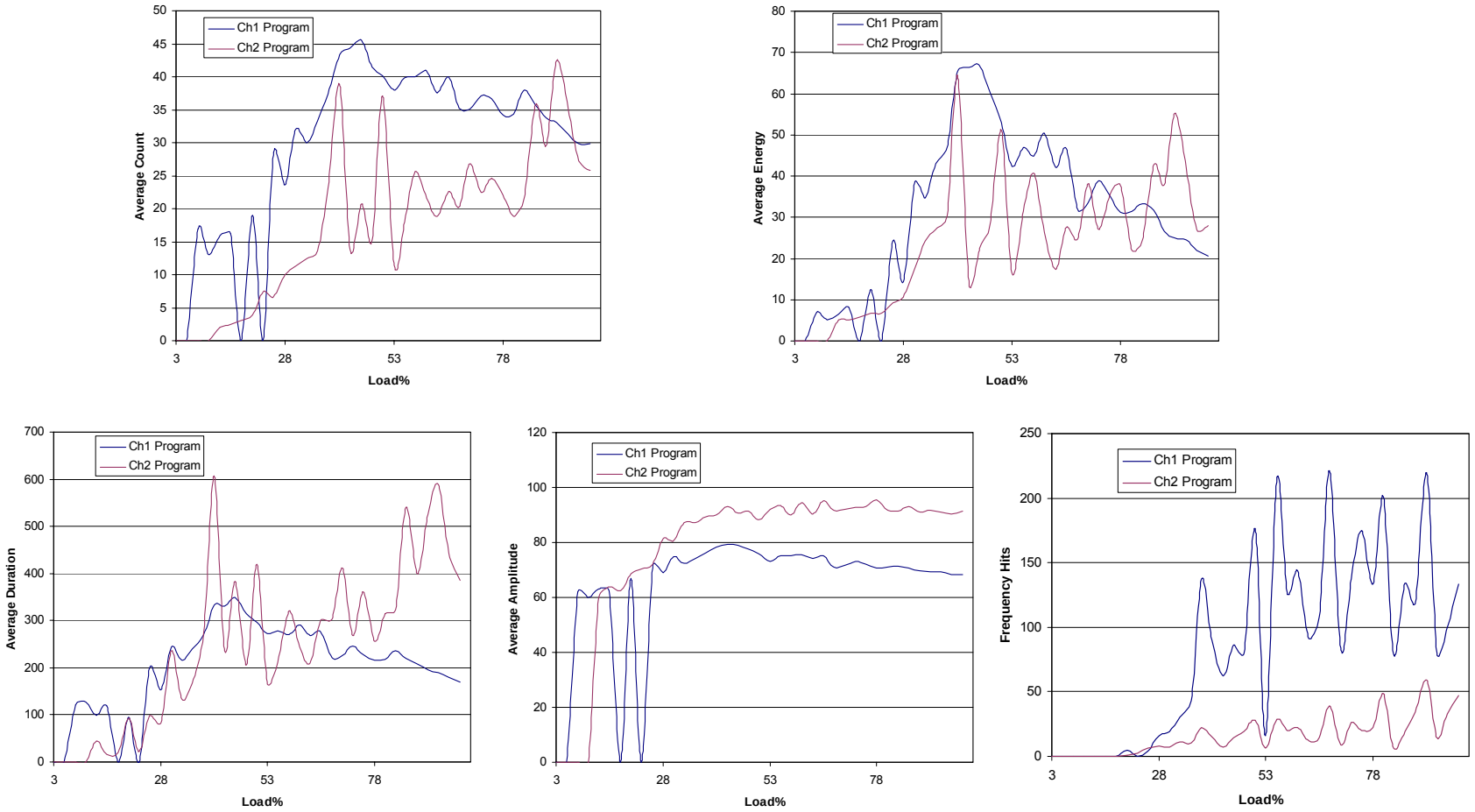


Figure 4.32 Test Ten2, Average AE parameters vs. Load%

Figure 4.33 Picture of Prodeck4 Single Unit.

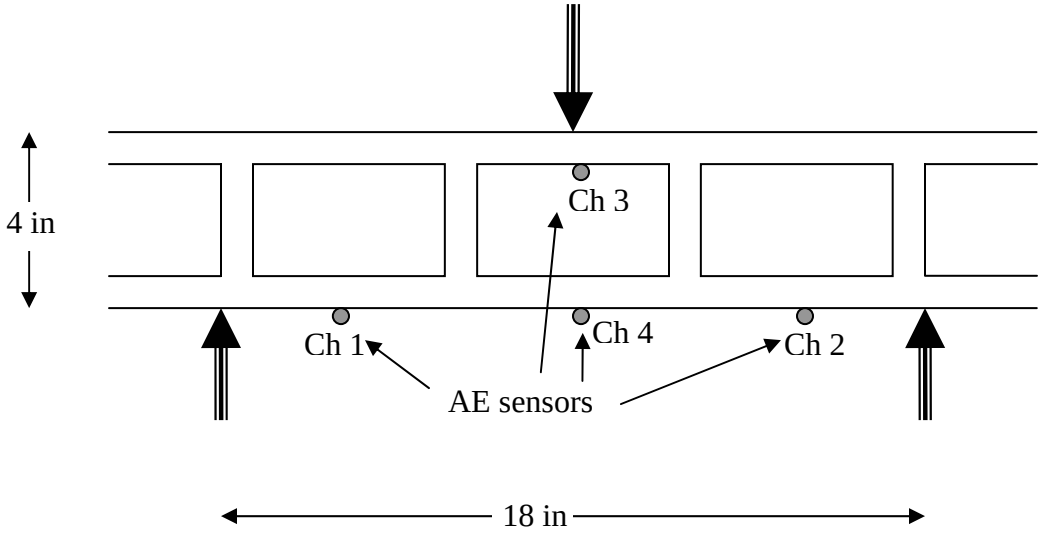


Figure 4.34 Bending specimen and sensor placement

Figure 4.35 Loaded bending specimen

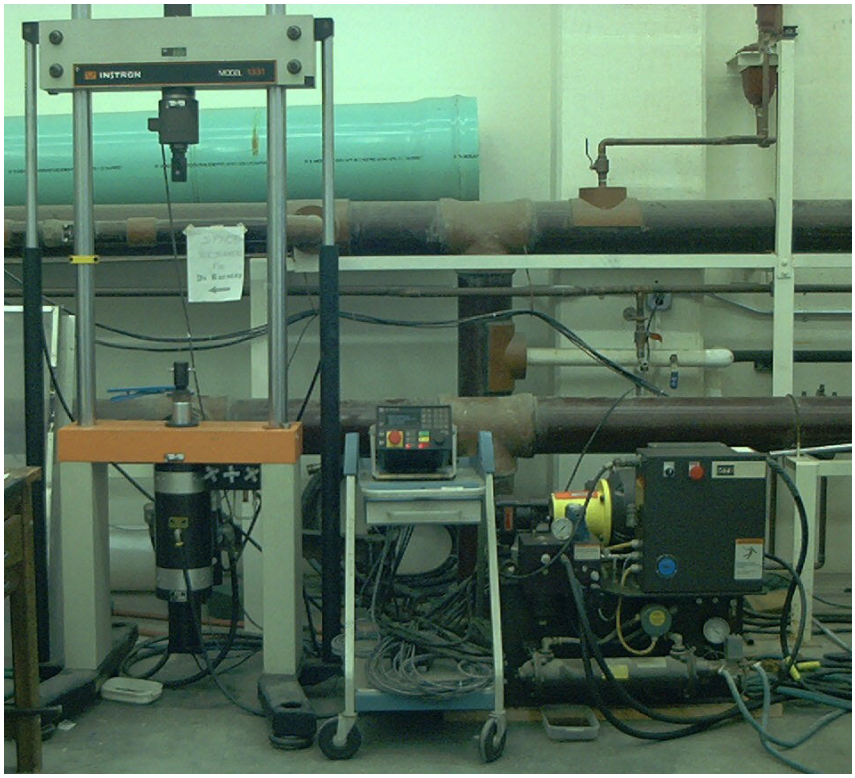
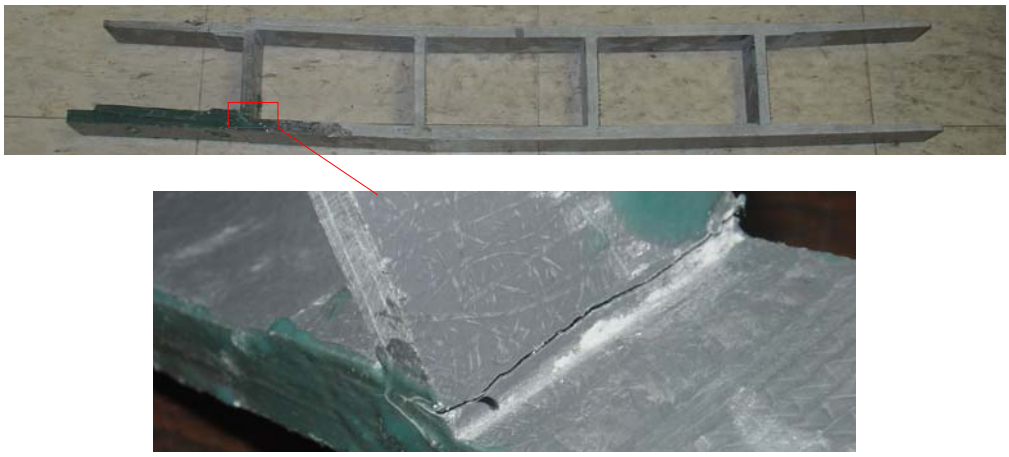


Figure 4.36 Bending test setup



(a)

(b)

Figure 4.37 Specimen after failure (a)bend 1 (b)bend 2

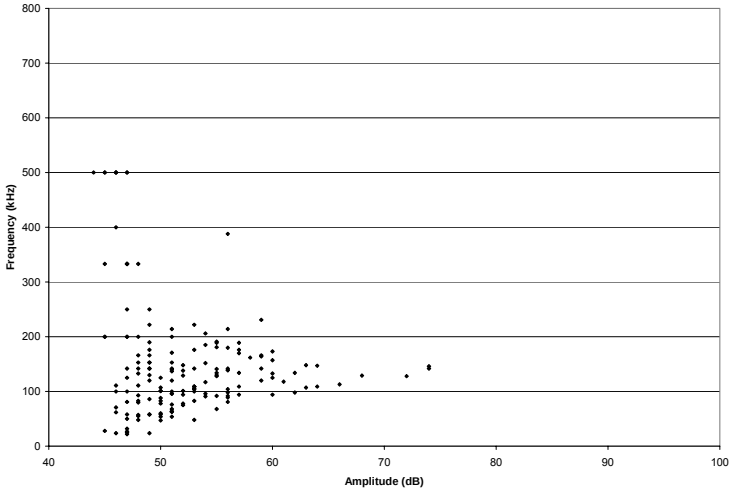


Figure 4.38 Frequency vs. Amplitude, Bend1 (0 – 158lbs) 1st Quarter

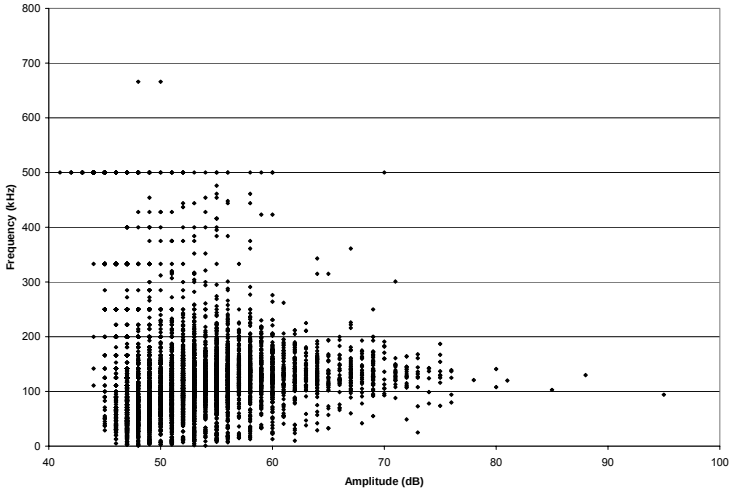


Figure 4.39 Frequency vs. Amplitude, Bend1 (158 – 316lbs) 2nd Quarter

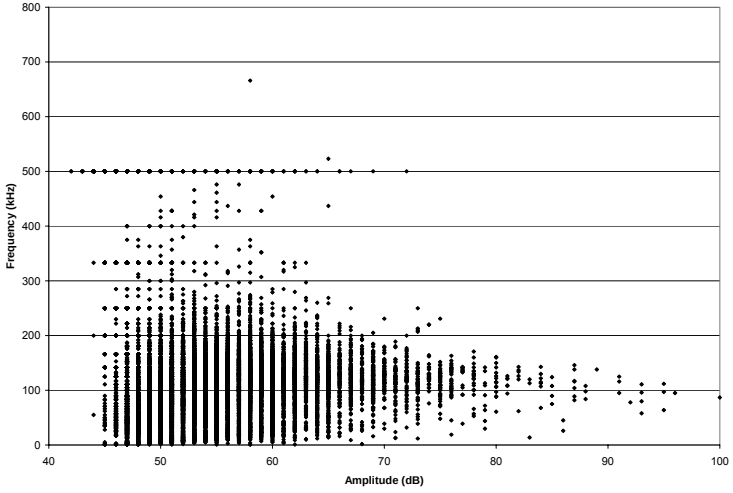


Figure 4.40 Frequency vs. Amplitude, Bend1 (316 – 474lbs), 3rd Quarter

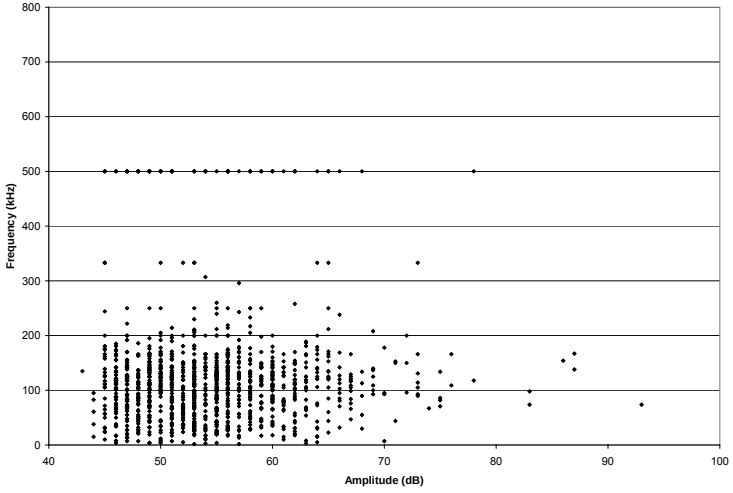


Figure 4.41 Frequency vs. Amplitude, Bend1 (474 – 633lbs), 4th Quarter

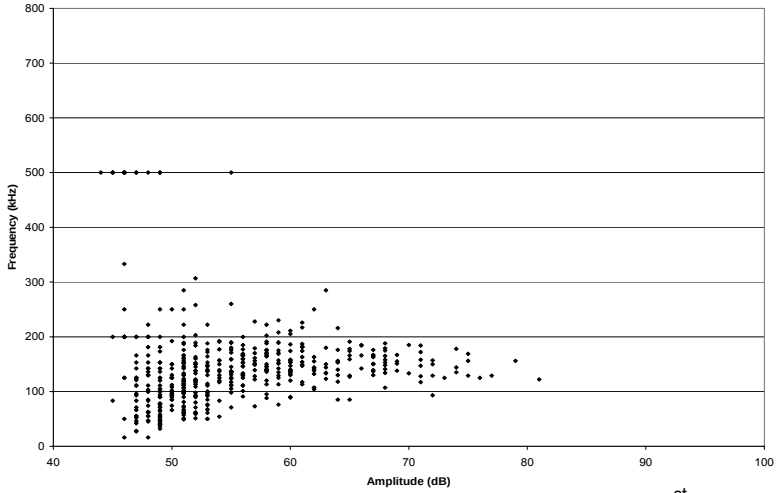


Figure 4.42 Frequency vs. Amplitude, Bend2 (0 – 161lbs), 1st Quarter

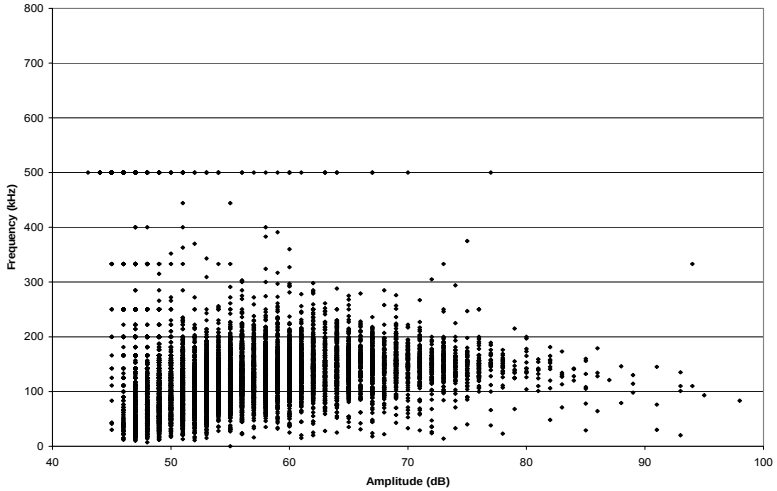


Figure 4.43 Frequency vs. Amplitude, Bend2 (161 – 322 lbs), 2nd Quarter

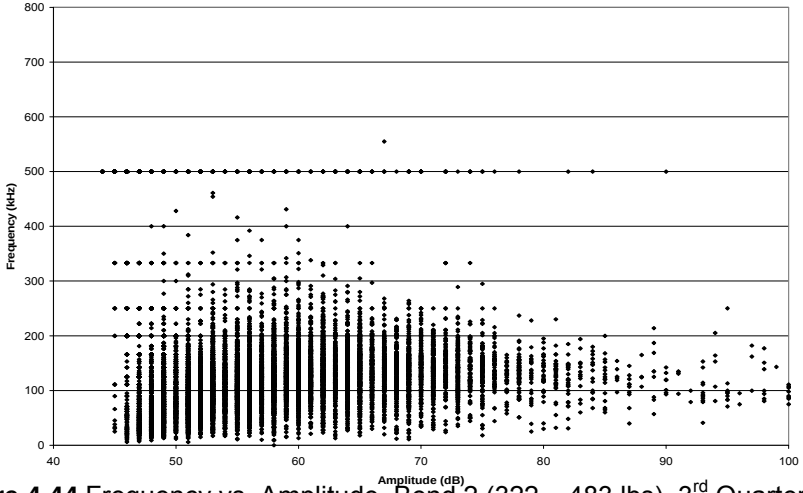


Figure 4.44 Frequency vs. Amplitude, Bend 2 (322 – 483 lbs), 3rd Quarter

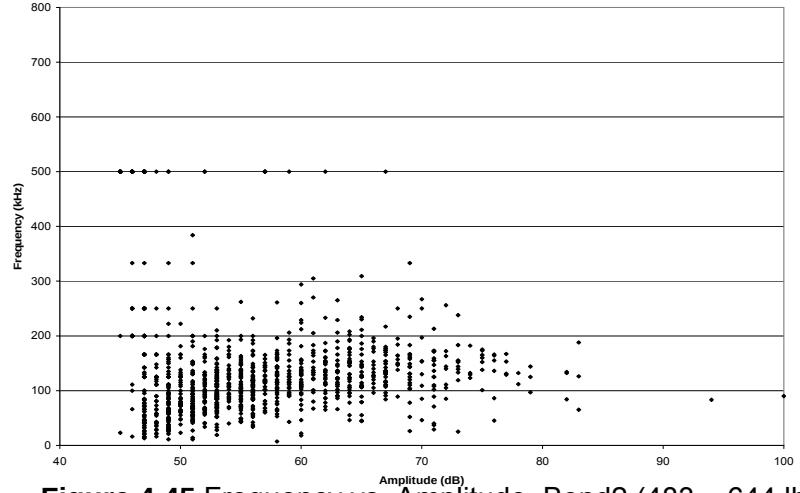


Figure 4.45 Frequency vs. Amplitude, Bend2 (483 – 644 lbs), 4th Quarter

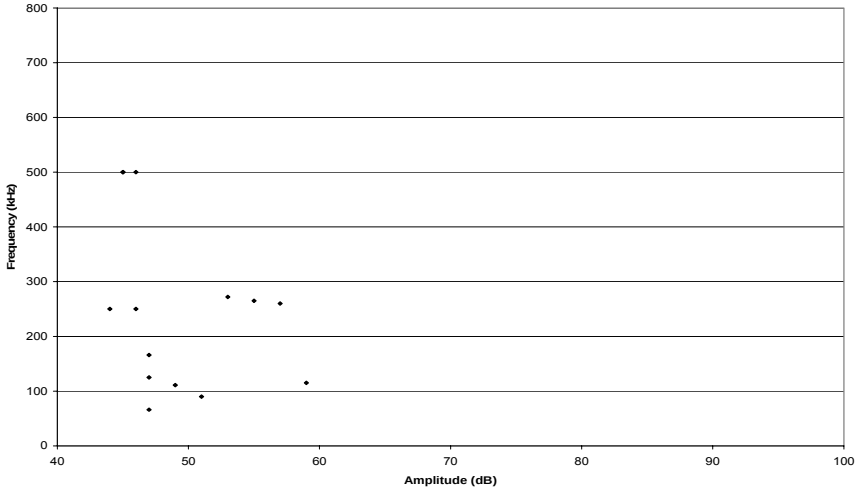


Figure 4.46 Frequency vs. Amplitude, FatBend (0 – 161 lbs), 1st Quarter

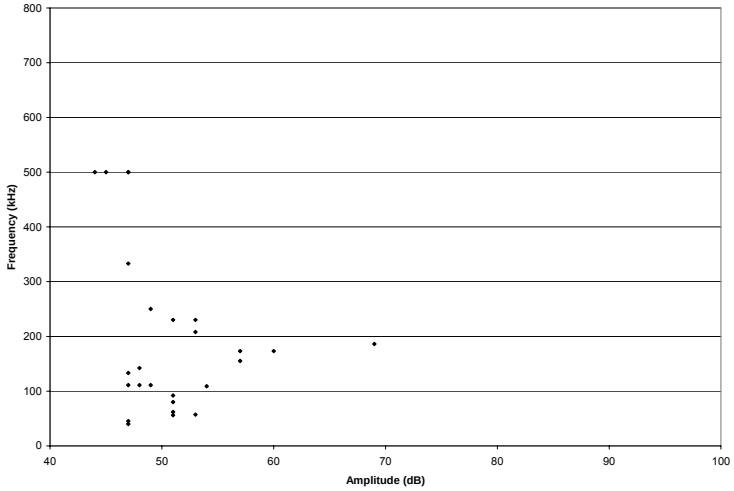


Figure 4.47 Frequency vs. Amplitude, FatBend (161 – 322 lbs), 2nd Quarter

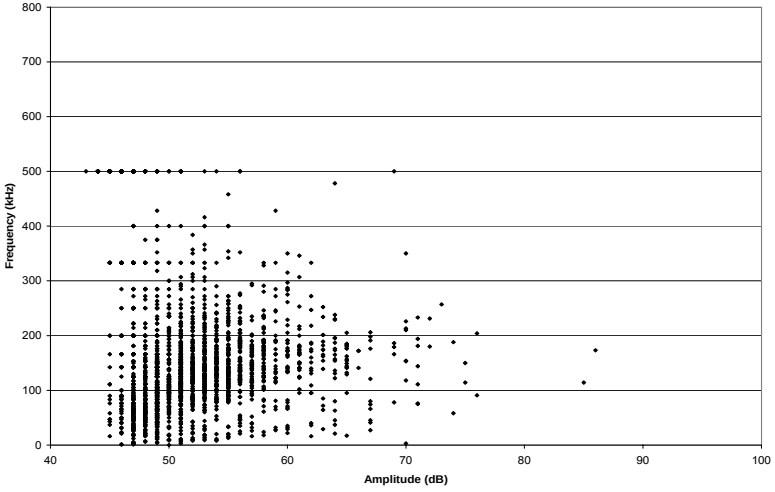


Figure 4.48 Frequency vs. Amplitude, FatBend (322 – 483 lbs), 3rd Quarter

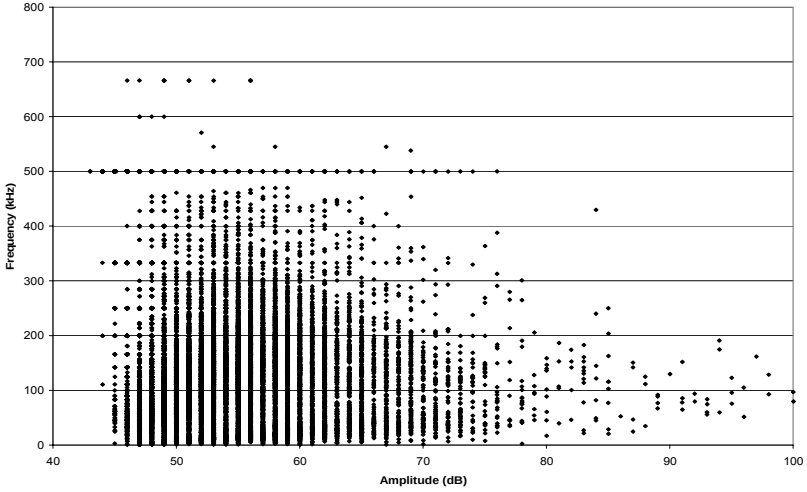


Figure 4.49 Frequency vs. Amplitude, FatBend (483 – 644 lbs), 4th Quarter

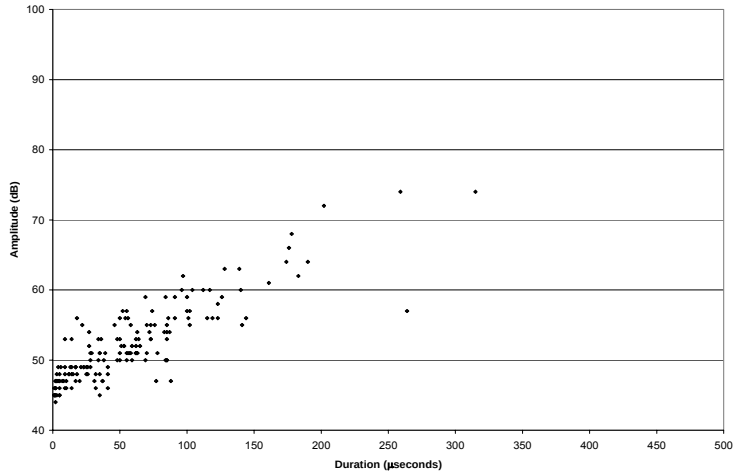


Figure 4.50 Amplitude vs. Duration, Bend1 (0 – 158lbs), 1st Quarter

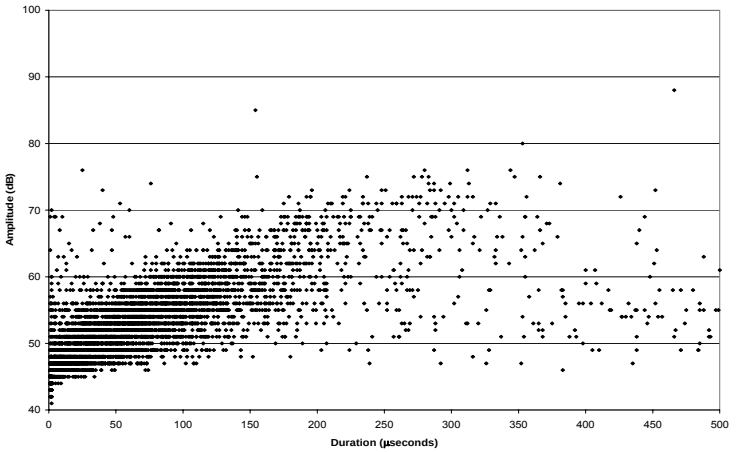


Figure 4.51 Amplitude vs. Duration, Bend1 (158 – 316lbs), 2nd Quarter

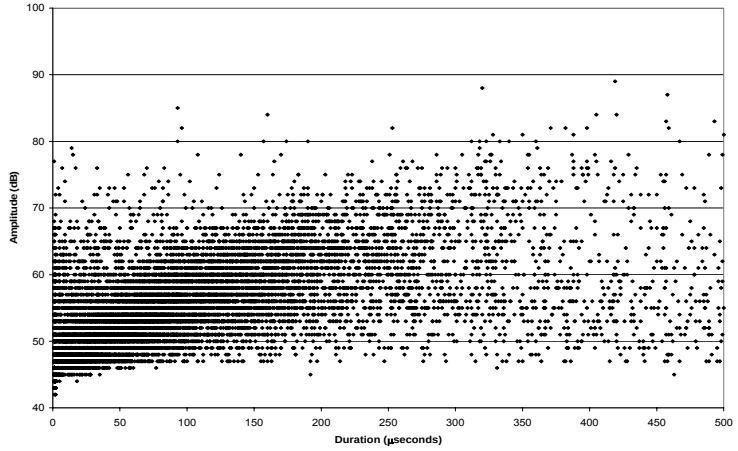


Figure 4.52 Amplitude vs. Duration, Bend1 (316 – 474lbs) 3rd Quarter

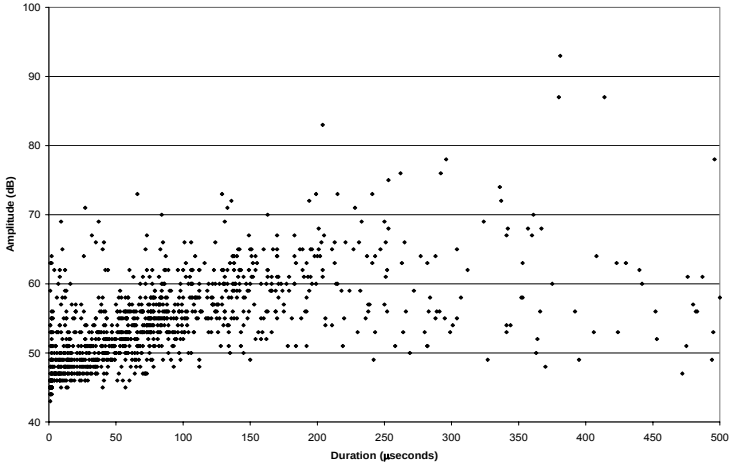


Figure 4.53 Amplitude vs. Duration, Bend1 (474 – 633lbs) 4th Quarter

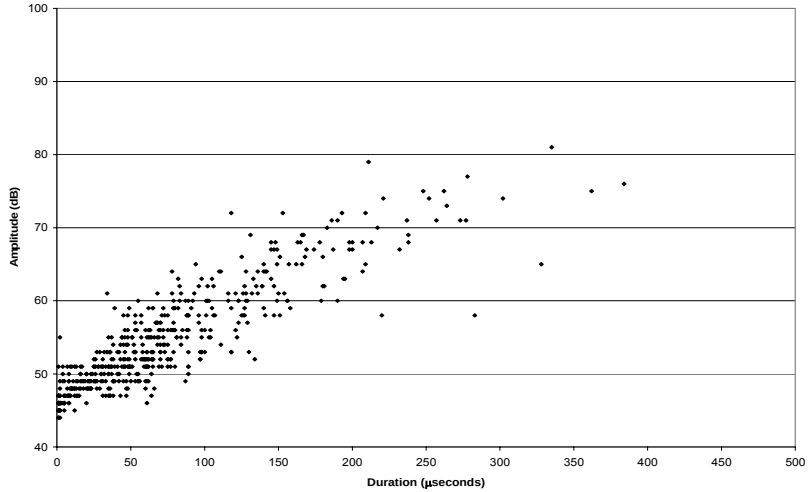


Figure 4.54 Amplitude vs. Duration, Bend2 (0 – 161 lbs) 1st Quarter

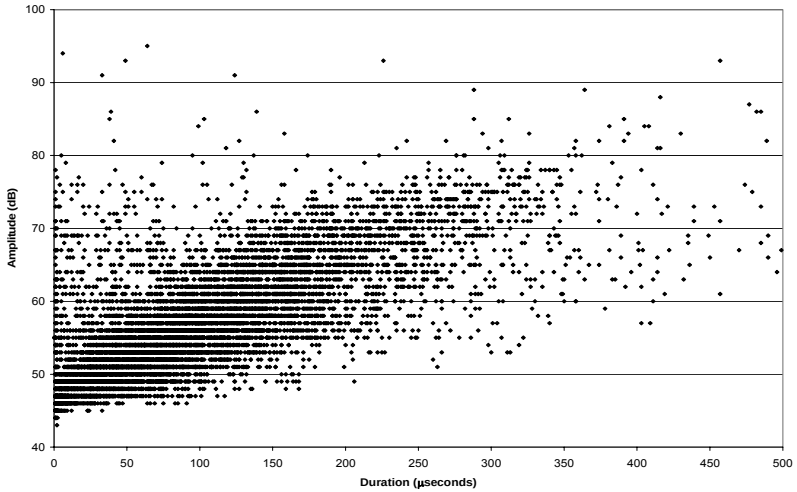


Figure 4.55 Amplitude vs. Duration, Bend2 (161 – 322 lbs) 2nd Quarter

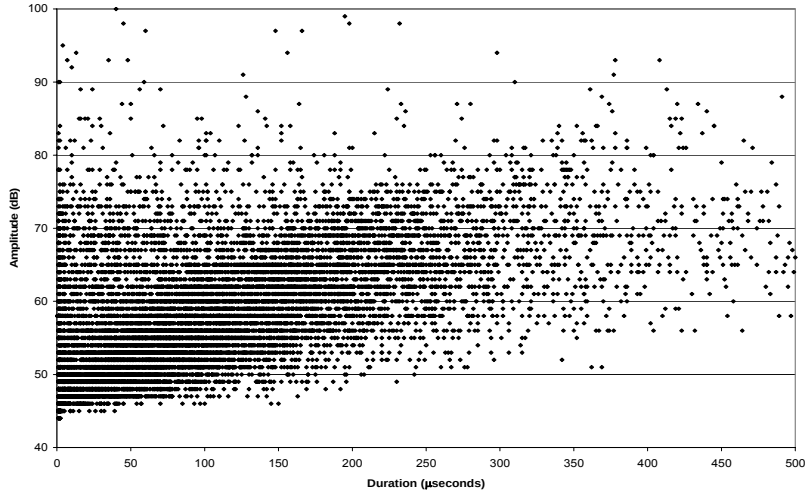


Figure 4.56 Amplitude vs. Duration, Bend2 (322 – 483 lbs) 3rd Quarter

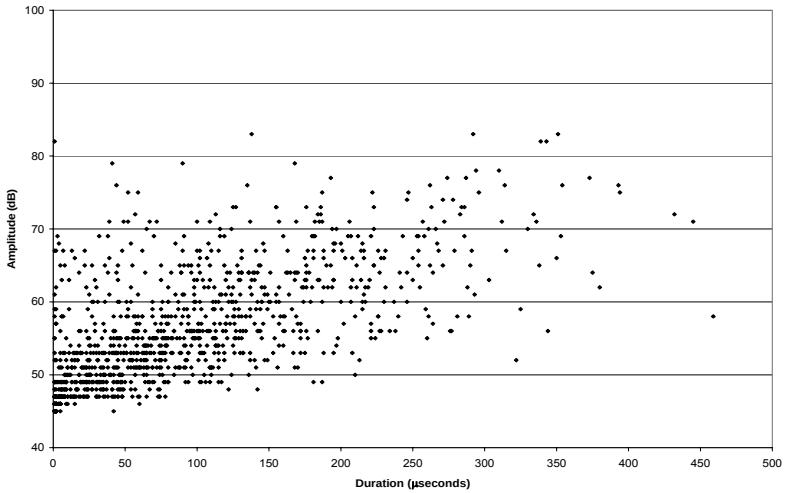


Figure 4.57 Amplitude vs. Duration, Bend2 (483 – 644 lbs) 4th Quarter

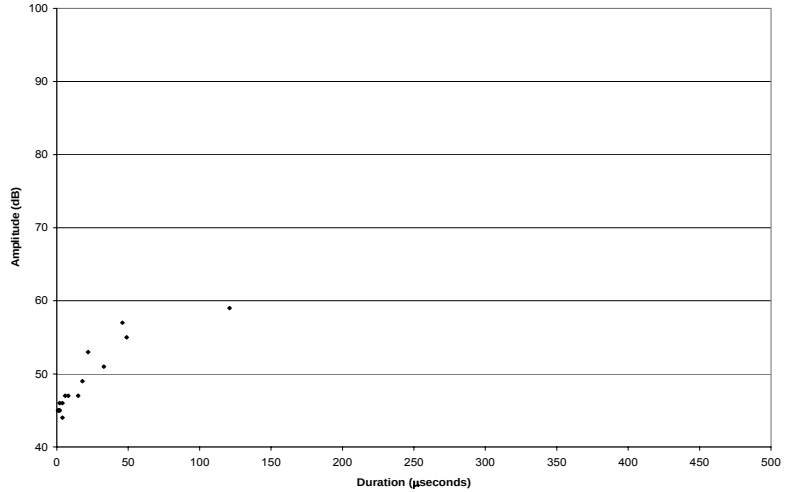


Figure 4.58 Amplitude vs. Duration, FatBend (0 – 161 lbs) 1st Quarter

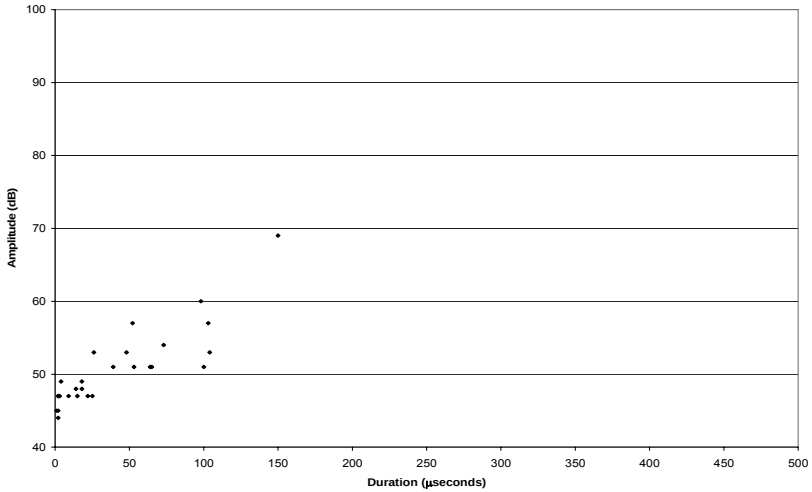


Figure 4.59 Amplitude vs. Duration, FatBend (161 – 322 lbs) 2nd Quarter

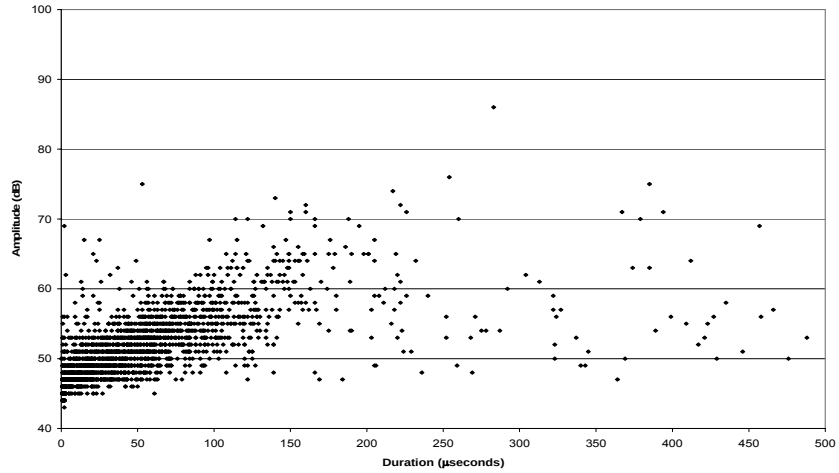


Figure 4.60 Amplitude vs. Duration, FatBend (322 – 483 lbs) 3rd Quarter

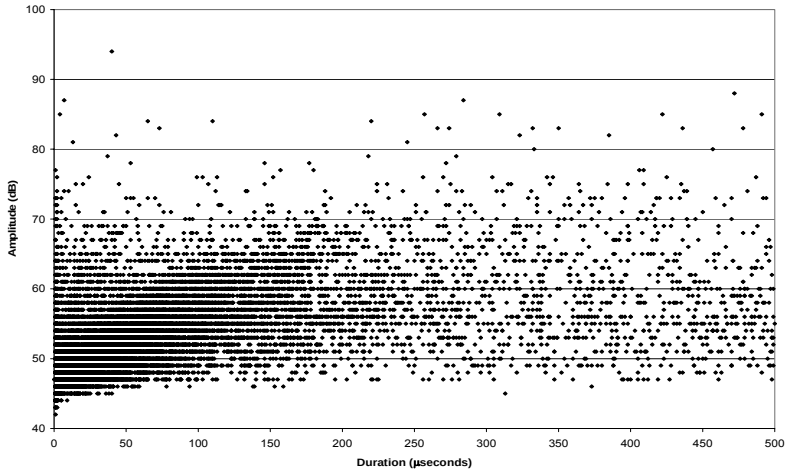


Figure 4.61 Amplitude vs. Duration, FatBend (483 – 644 lbs) 4th Quarter

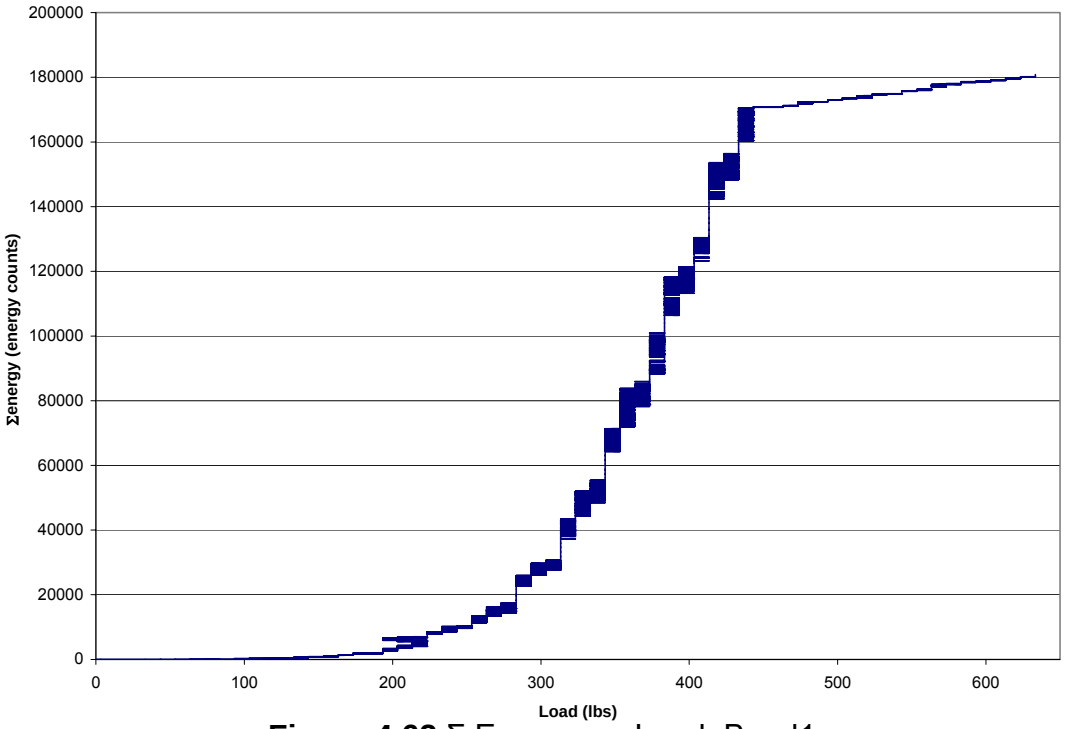


Figure 4.62 Σ Energy vs. Load, Bend1

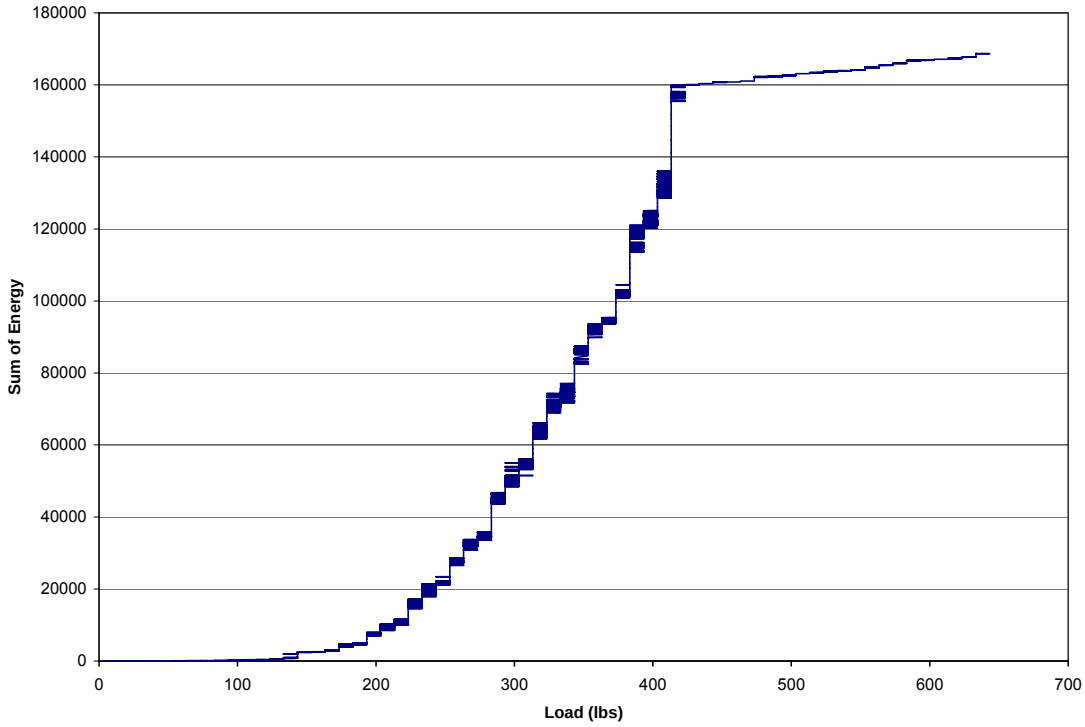


Figure 4.63 Σ Energy vs. Load, Bend2

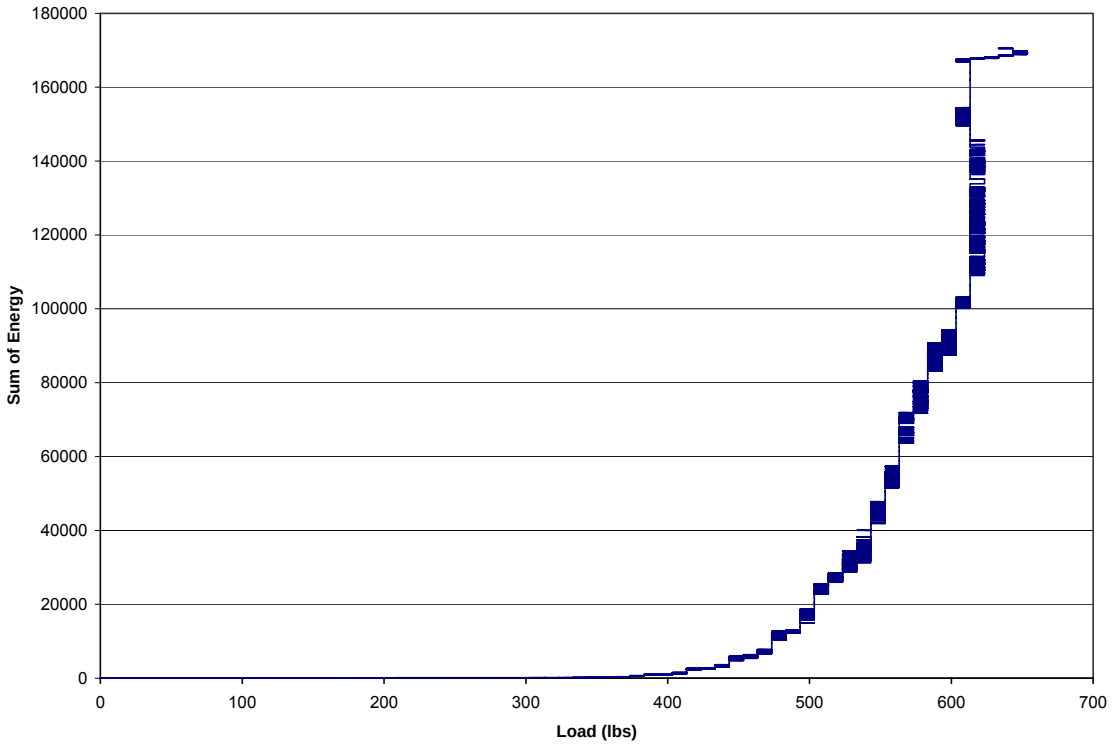


Figure 4.64 Σ Energy vs. Load, FatBend

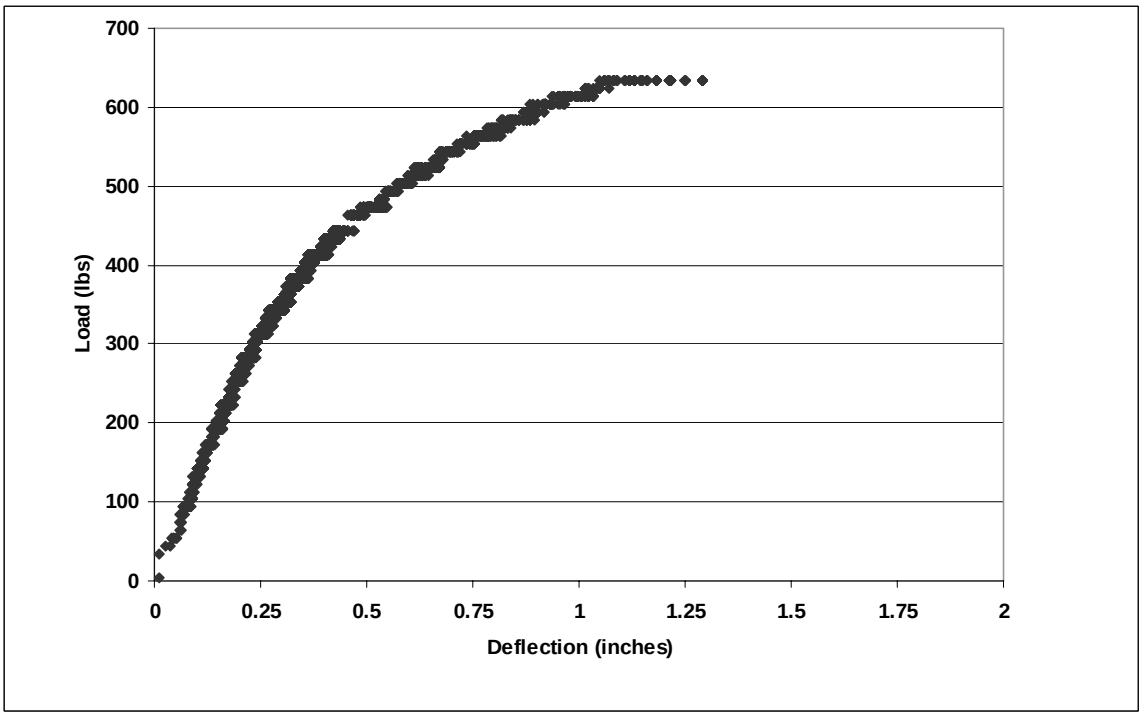


Figure 4.65 Load vs. Deflection, Bend1

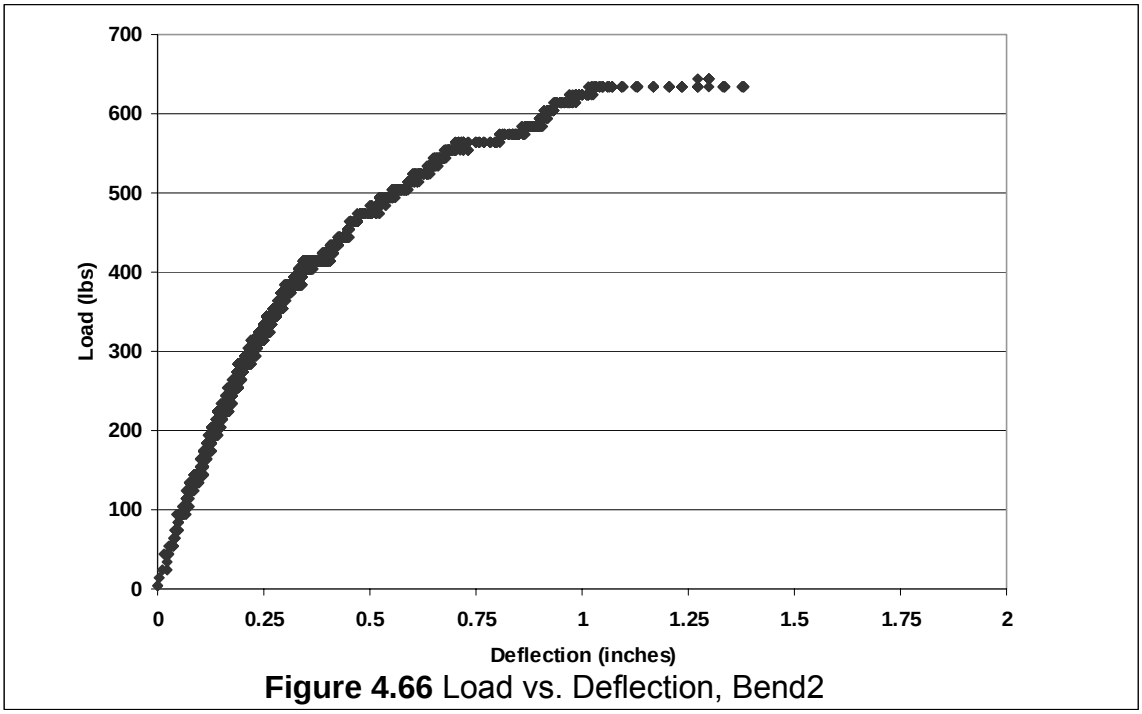


Figure 4.66 Load vs. Deflection, Bend2

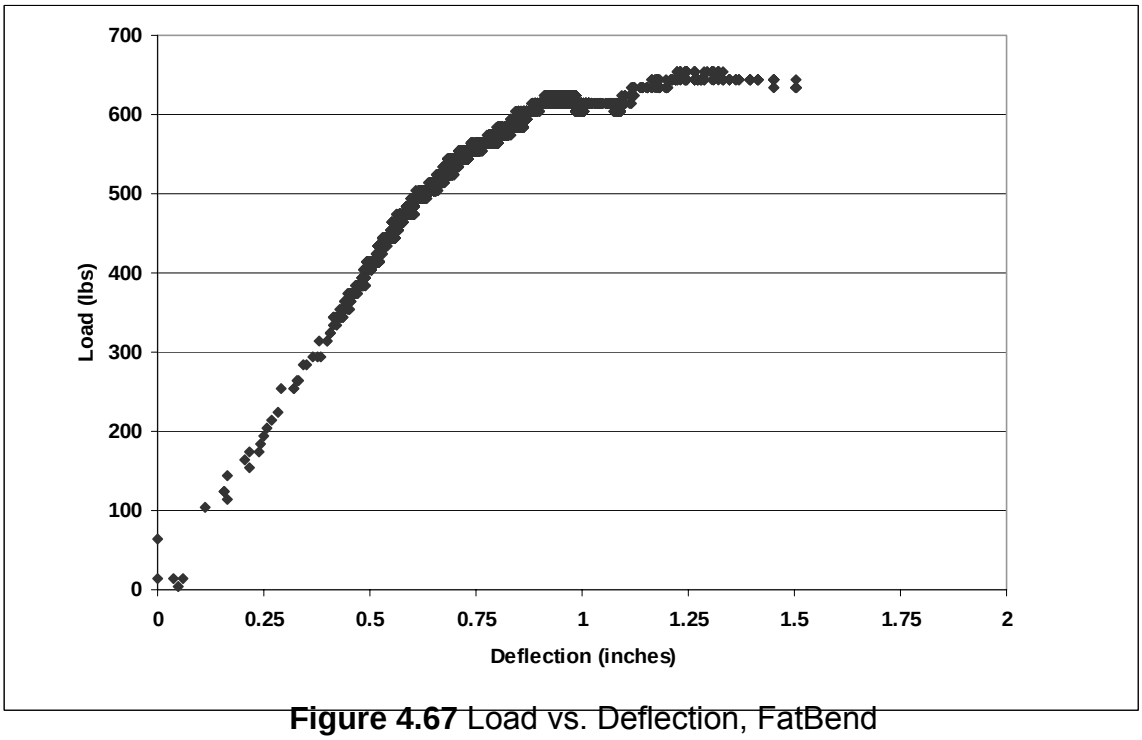


Figure 4.67 Load vs. Deflection, FatBend

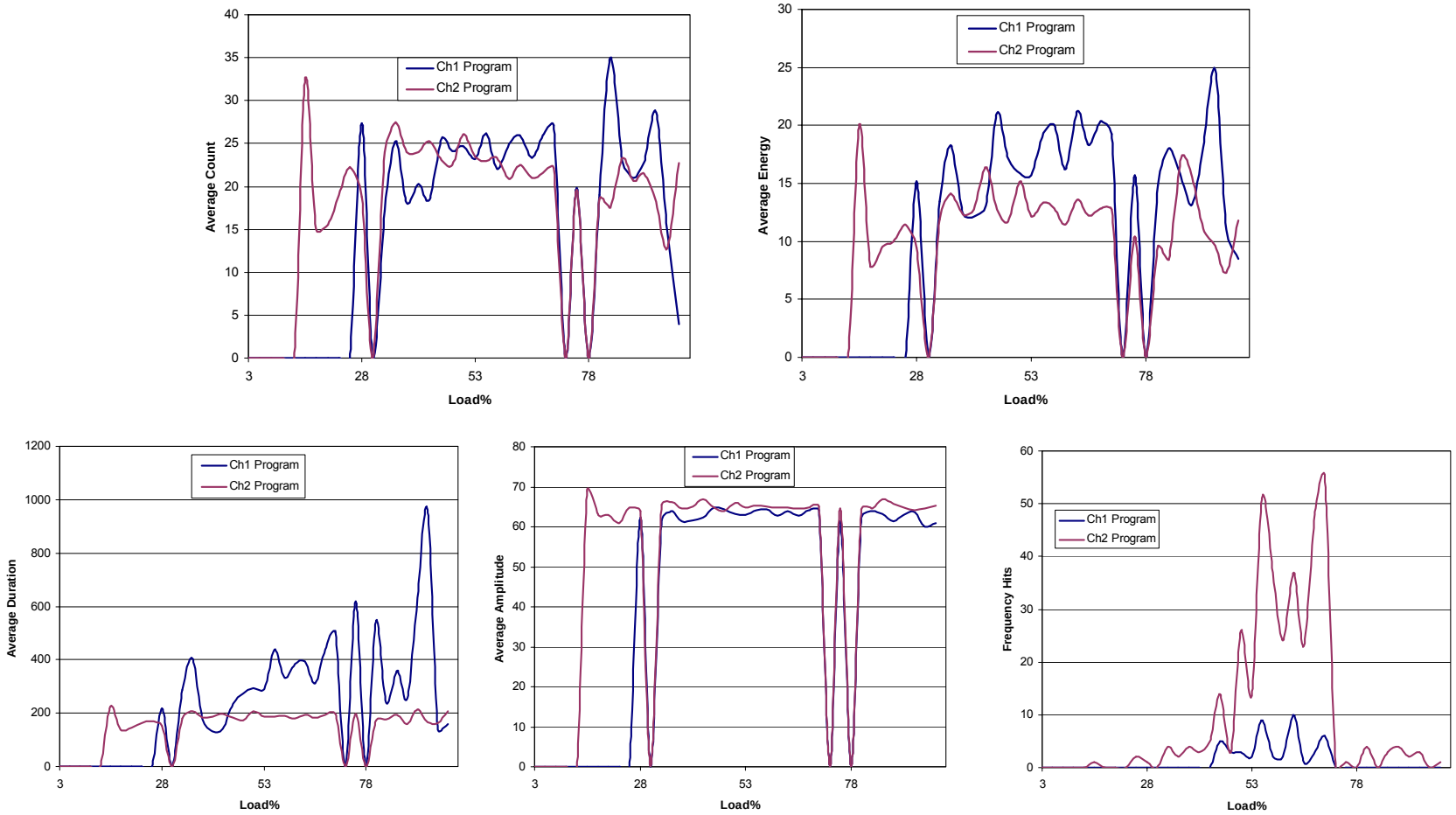


Figure 4.68 Test Bend1, Average AE parameters vs. Load%

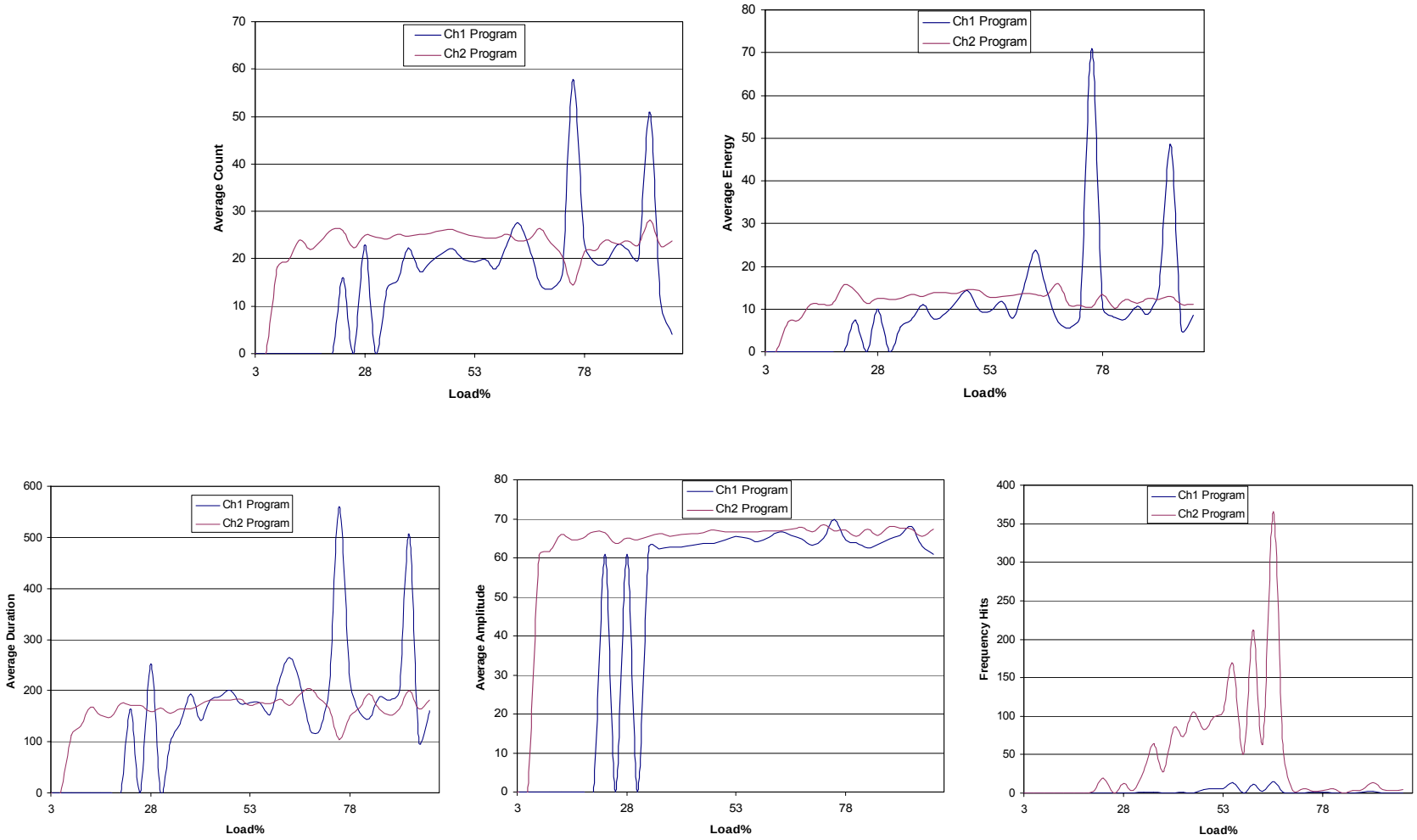


Figure 4.69 Test Bend2, Average AE parameters vs. Load%

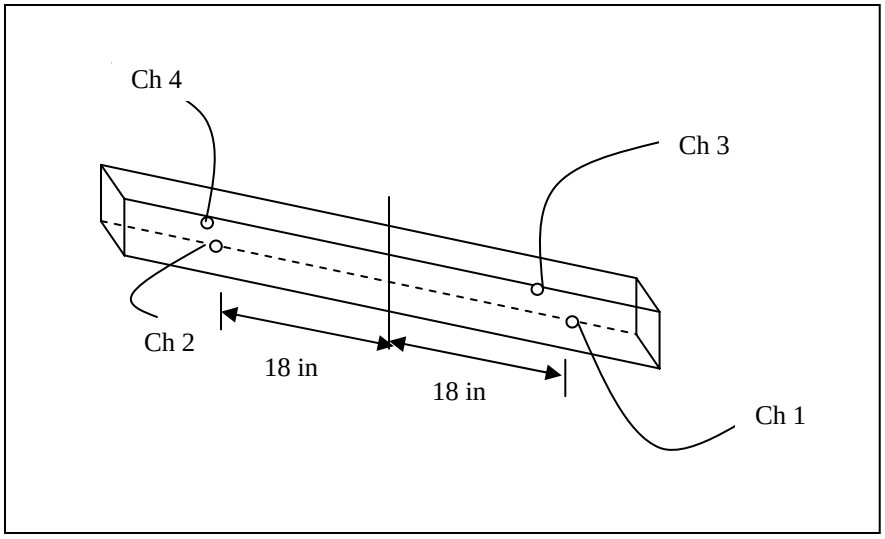
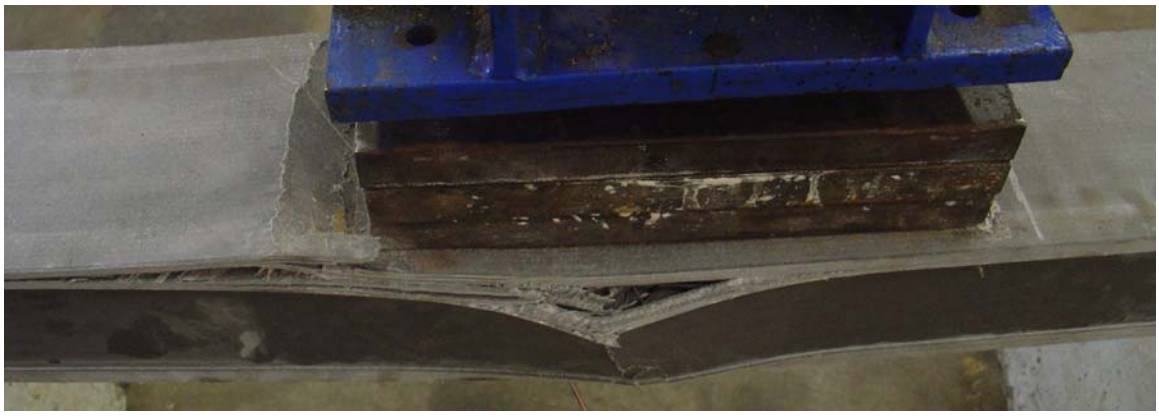


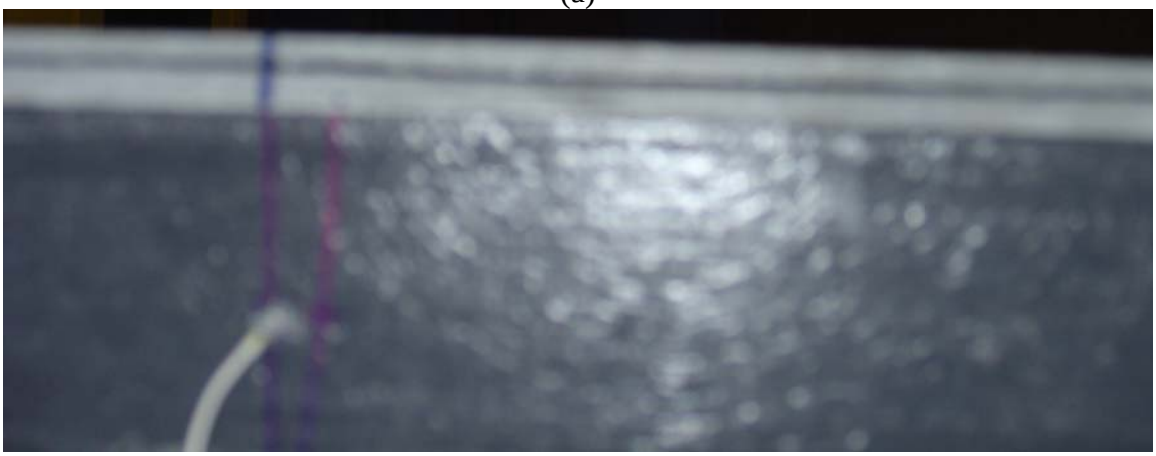
Figure 4.70 Sensor placement

Figure 4.71 Fatigue test setup

Figure 4.72 Fatigue specimen with sensors



(a)



(b)

Figure 4.73 Fatigue specimens after failure (a) Fat1 (b) Fat2

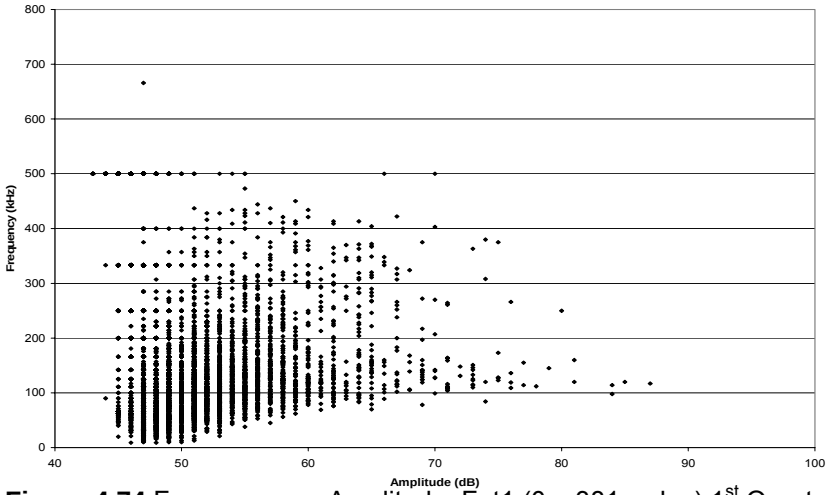


Figure 4.74 Frequency vs. Amplitude, Fat1 (0 – 381cycles) 1st Quarter

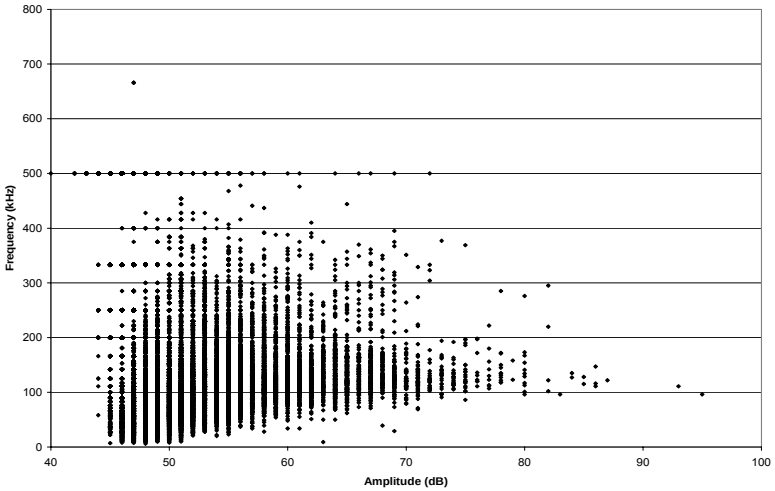


Figure 4.75 Frequency vs. Amplitude, Fat1 (381 – 762cycles) 2nd Quarter

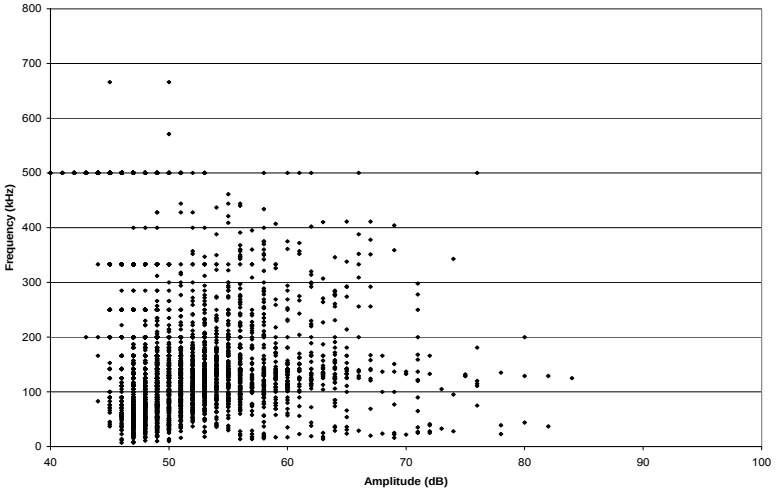


Figure 4.76 Frequency vs. Amplitude, Fat1 (762 – 1143cycles) 3rd Quarter

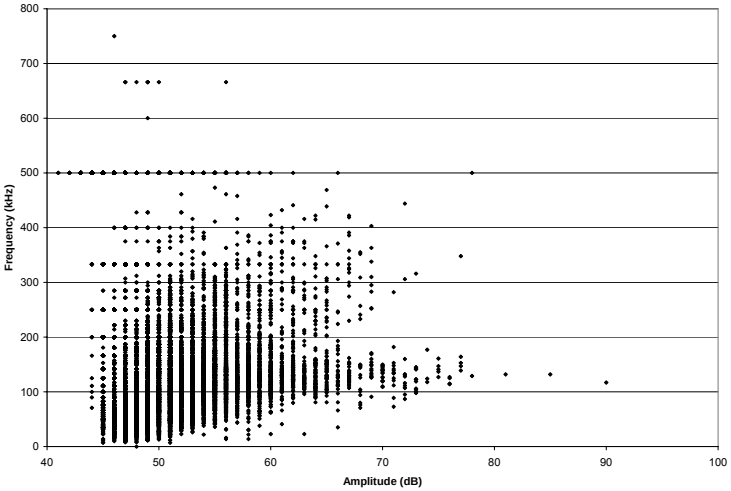


Figure 4.77 Frequency vs. Amplitude, Fat1 (1143 – 1524cycles) 4th Quarter

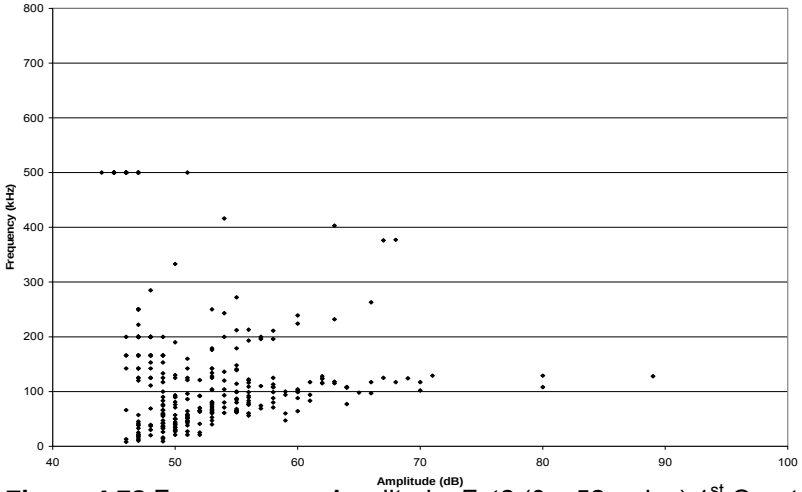


Figure 4.78 Frequency vs. Amplitude, Fat2 (0 – 52cycles) 1st Quarter

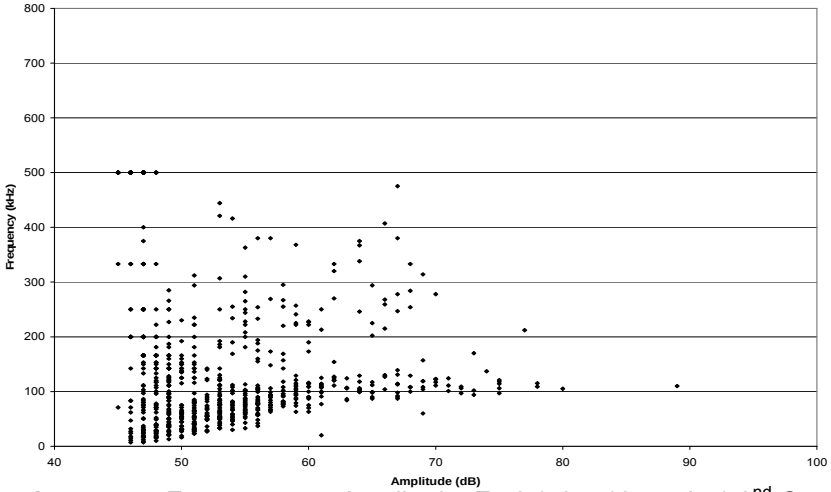


Figure 4.79 Frequency vs. Amplitude, Fat2 (52 – 104cycles) 2nd Quarter

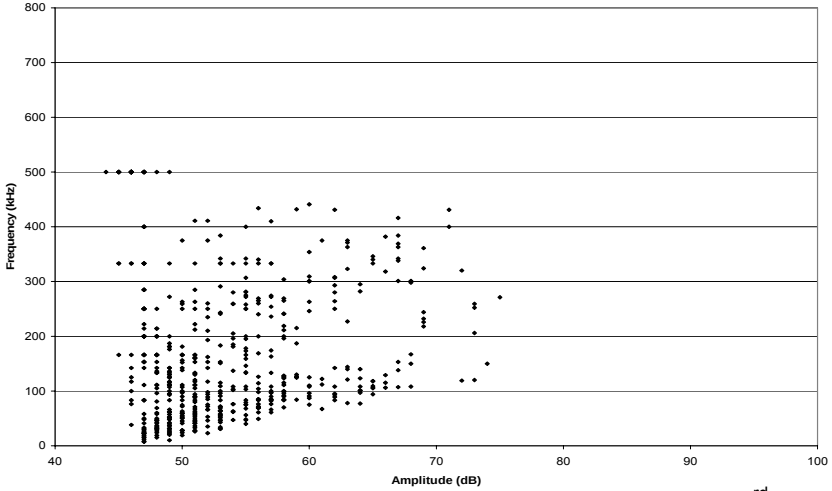


Figure 4.80 Frequency vs. Amplitude, Fat2 (104 – 156cycles) 3rd Quarter

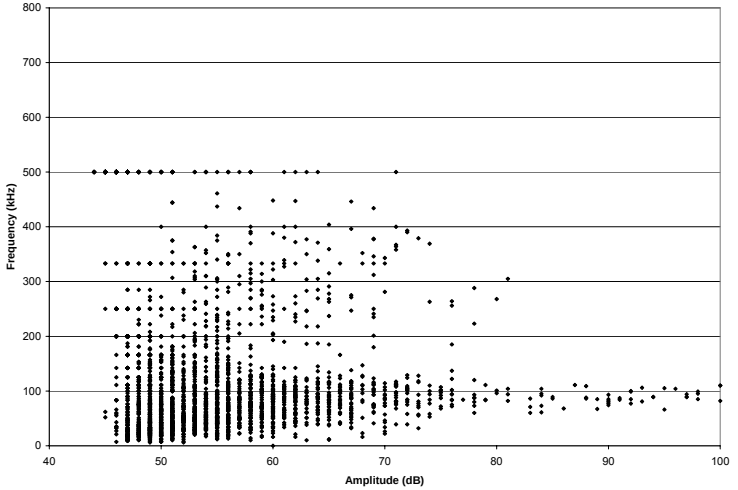


Figure 4.81 Frequency Vs Amplitude, Fat2 (156 – 208 cycles) 4th Quarter

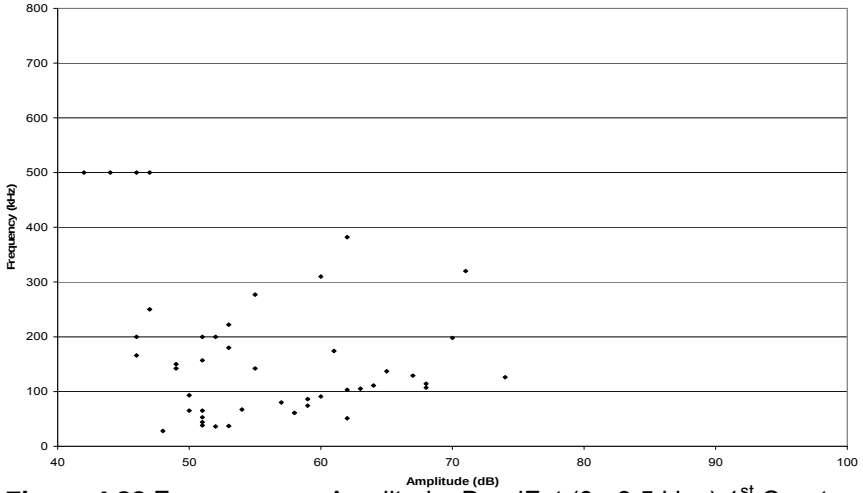


Figure 4.82 Frequency vs. Amplitude, BendFat (0 - 3.5 kips) 1st Quarter

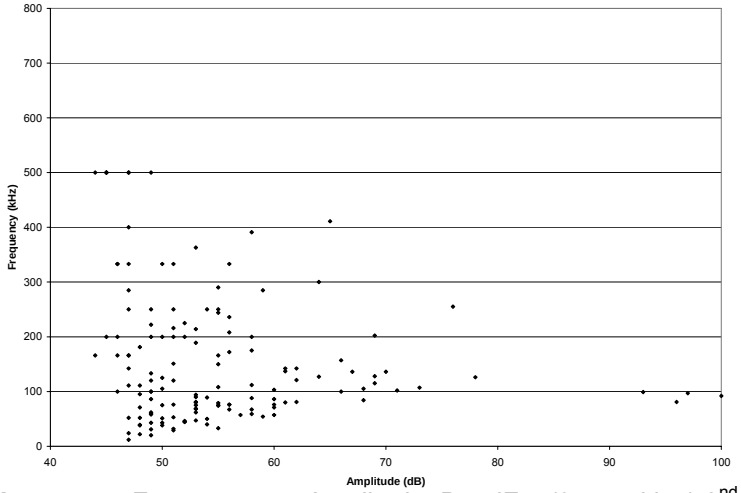


Figure 4.83 Frequency vs. Amplitude, BendFat (3.5 - 7 kips) 2nd Quarter

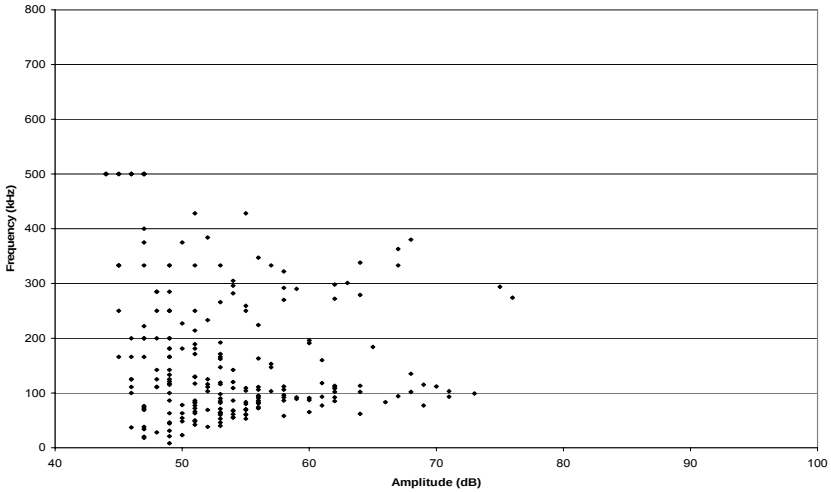


Figure 4.84 Frequency vs. Amplitude, BendFat (7 – 10.5 kips) 3rd Quarter

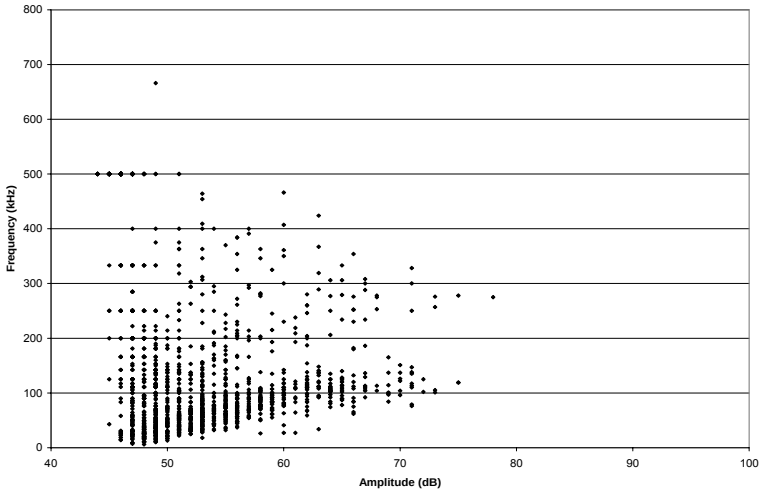


Figure 4.85 Frequency vs. Amplitude, BendFat (10.5 – 14 kips) 4th Quarter

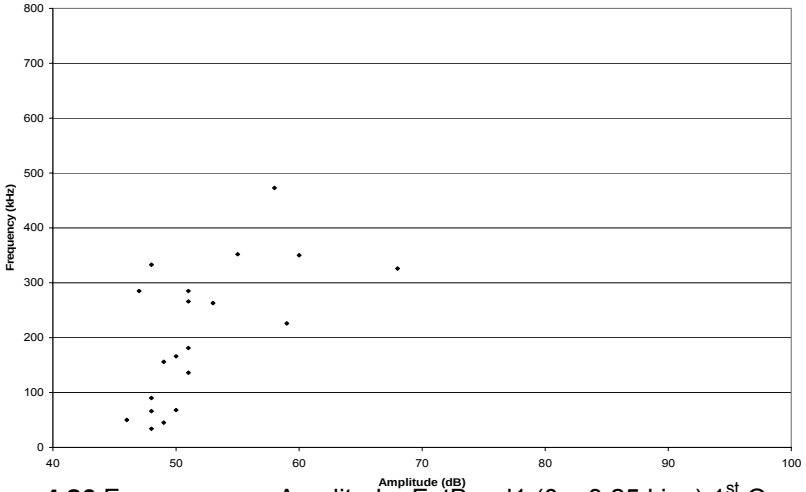


Figure 4.86 Frequency vs. Amplitude, FatBend1 (0 – 3.25 kips) 1st Quarter

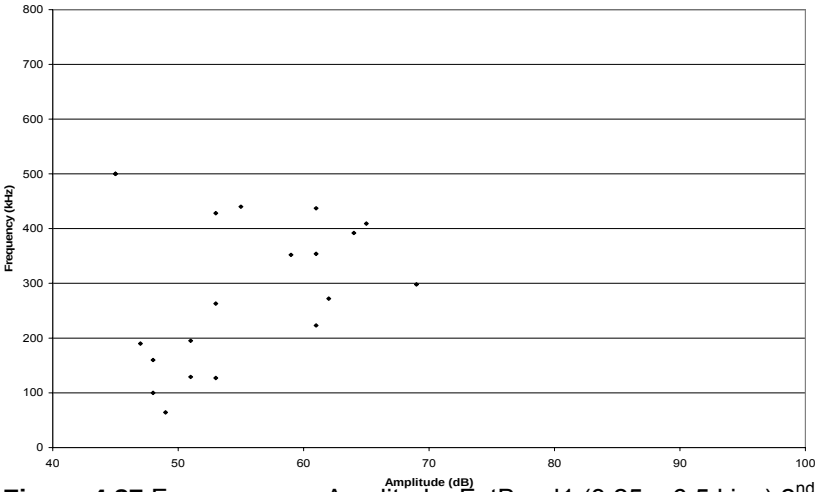


Figure 4.87 Frequency vs. Amplitude, FatBend1 (3.25 – 6.5 kips) 2nd Quarter

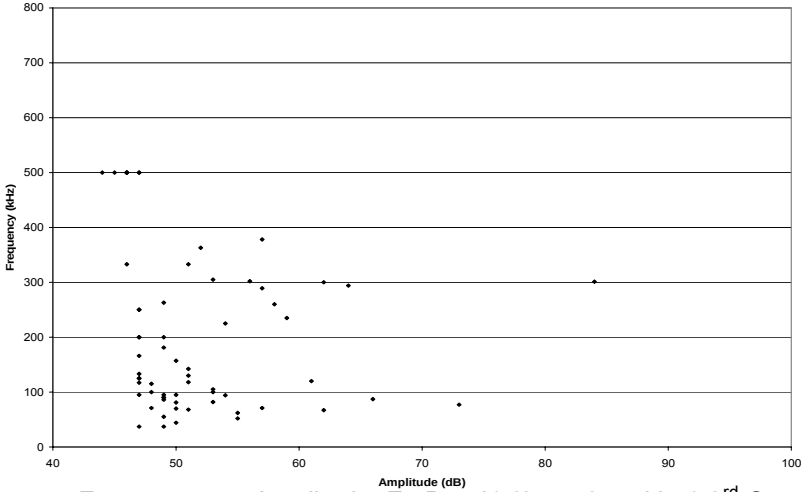


Figure 4.88 Frequency vs. Amplitude, FatBend1 (6.5 – 9.75 kips) 3rd Quarter

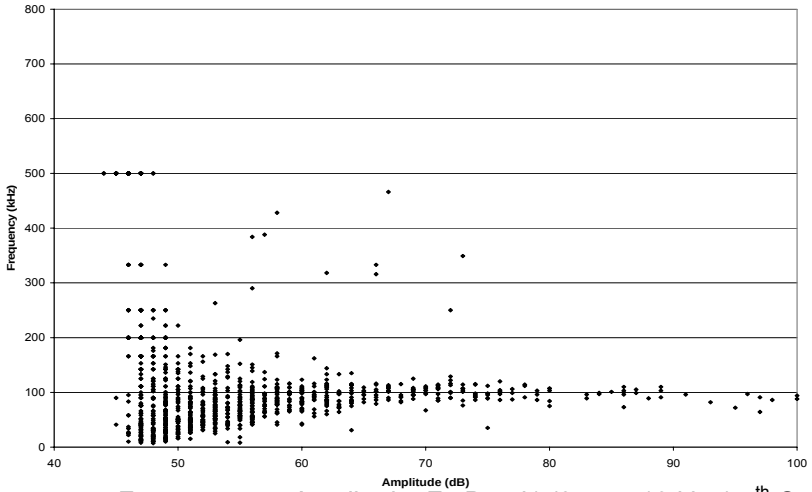


Figure 4.89 Frequency vs. Amplitude, FatBend1 (9.75 – 13 kips) 4th Quarter

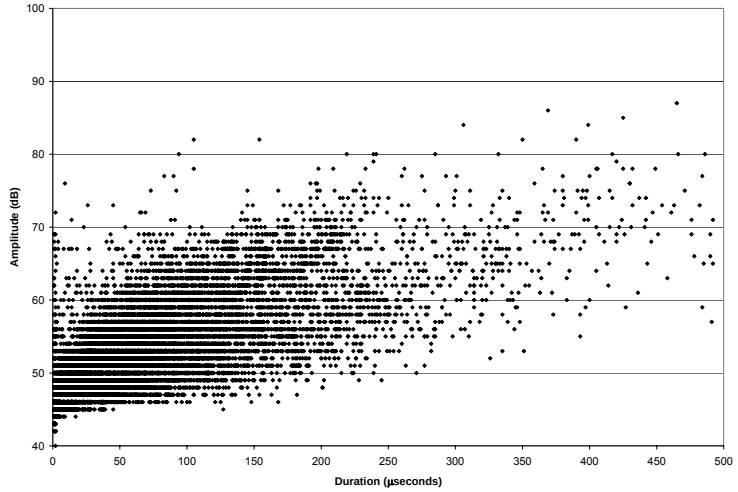


Figure 4.90 Amplitude vs. Duration, Fat1 (0 – 381cycles) 1st Quarter

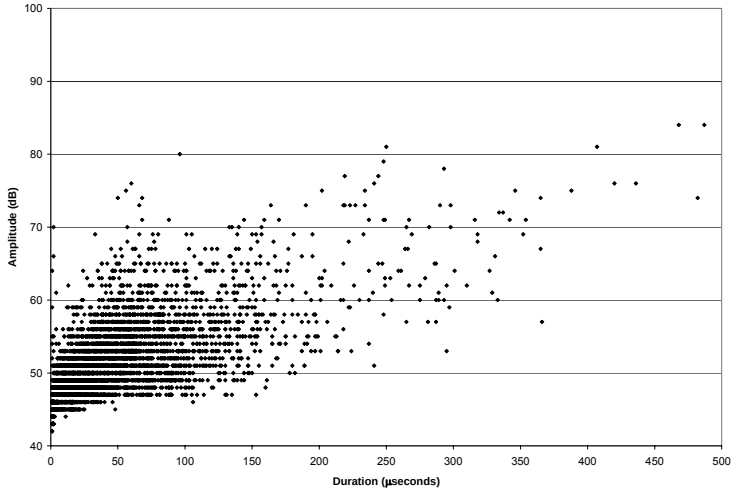


Figure 4.91 Amplitude vs. Duration, Fat1 (381 – 762cycles) 2nd Quarter

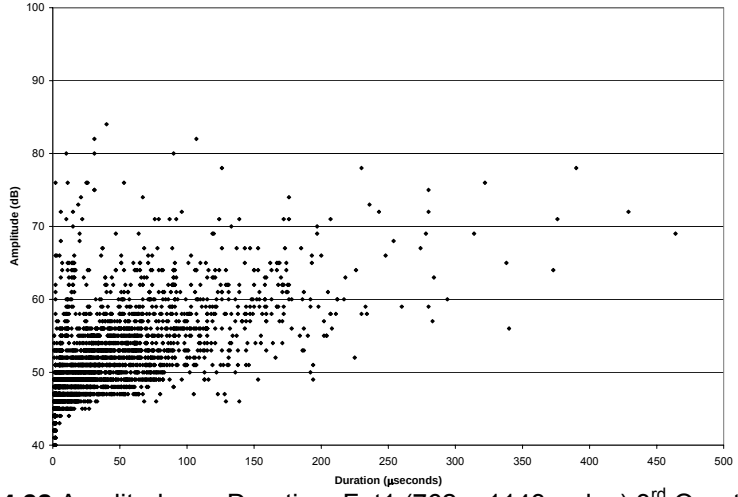


Figure 4.92 Amplitude vs. Duration, Fat1 (762 – 1143cycles) 3rd Quarter

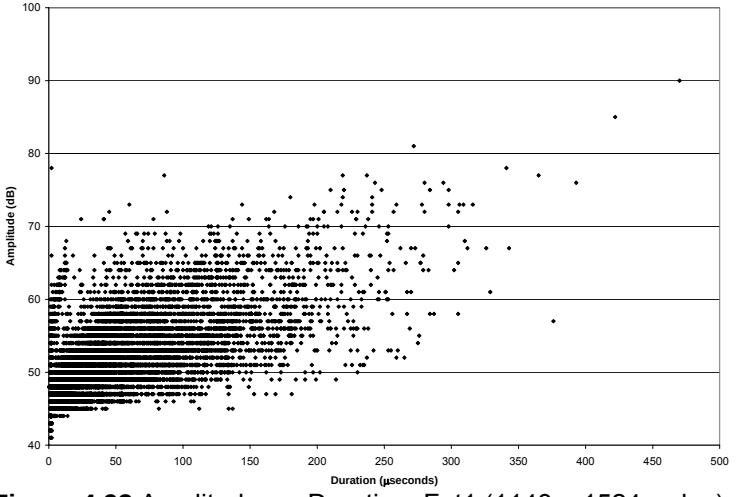


Figure 4.93 Amplitude vs. Duration, Fat1 (1143 – 1524cycles) 4th Quarter

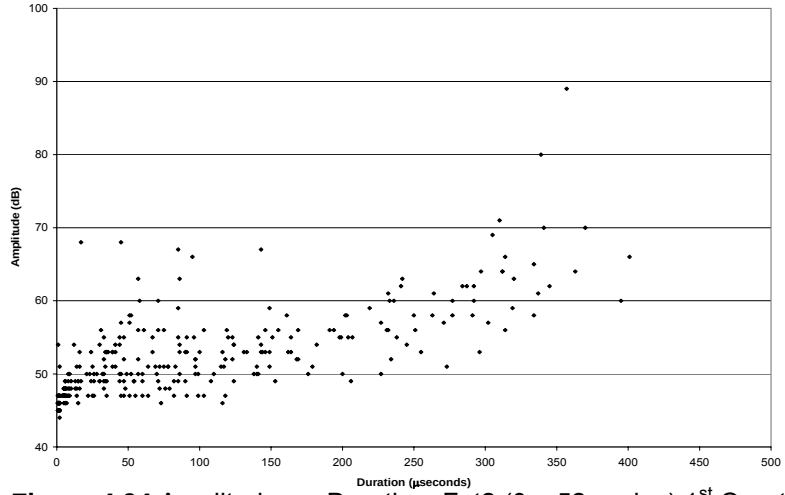


Figure 4.94 Amplitude vs. Duration, Fat2 (0 – 52 cycles) 1st Quarter

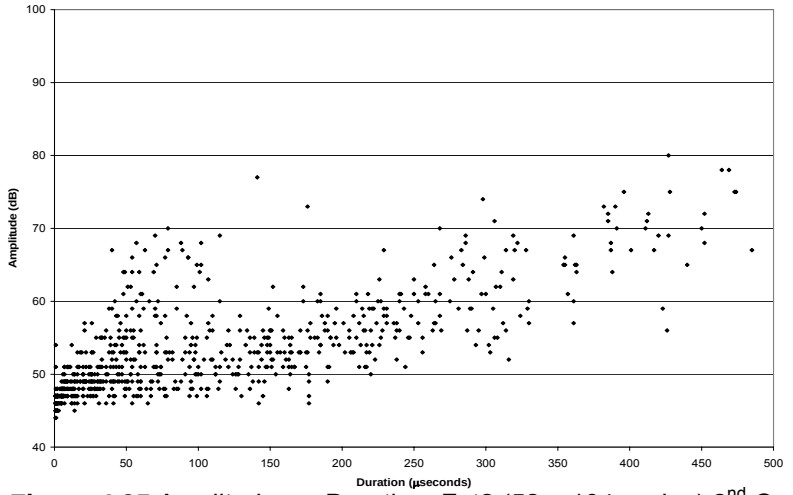


Figure 4.95 Amplitude vs. Duration, Fat2 (52 – 104 cycles) 2nd Quarter

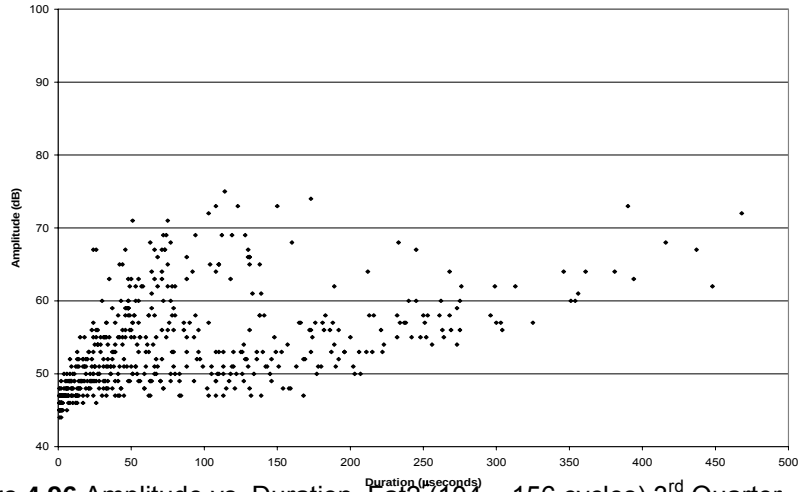


Figure 4.96 Amplitude vs. Duration, Fat2 (104 – 156 cycles) 3rd Quarter

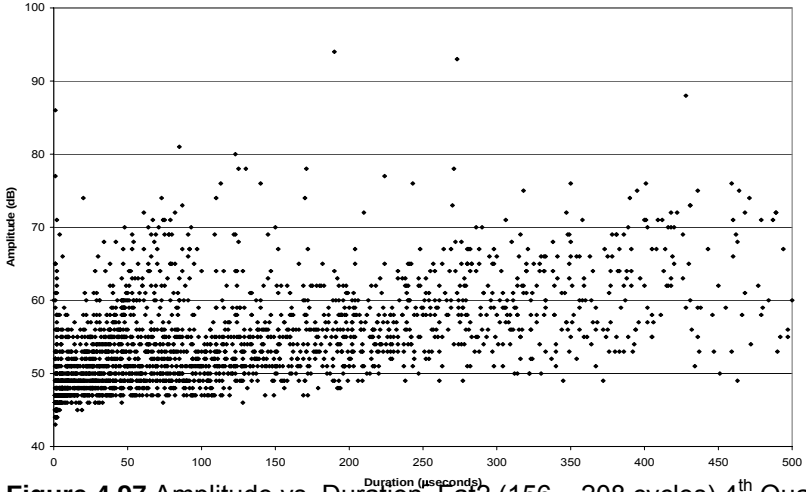


Figure 4.97 Amplitude vs. Duration, Fat2 (156 – 208 cycles) 4th Quarter

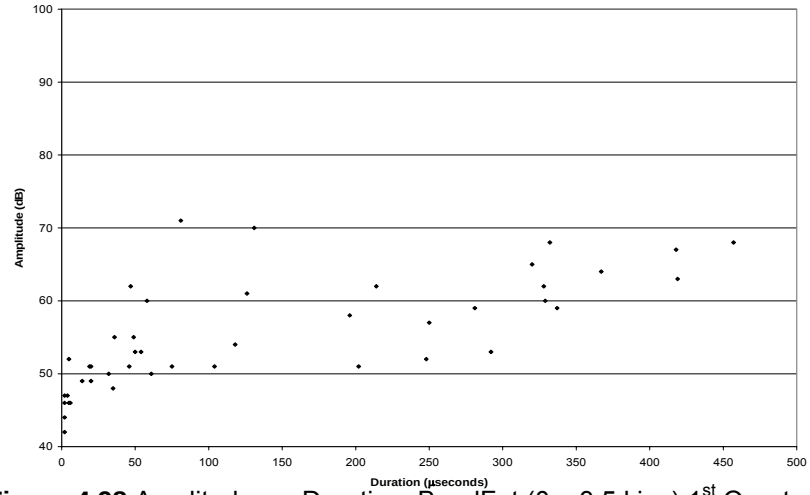


Figure 4.98 Amplitude vs. Duration, BendFat (0 – 3.5 kips) 1st Quarter

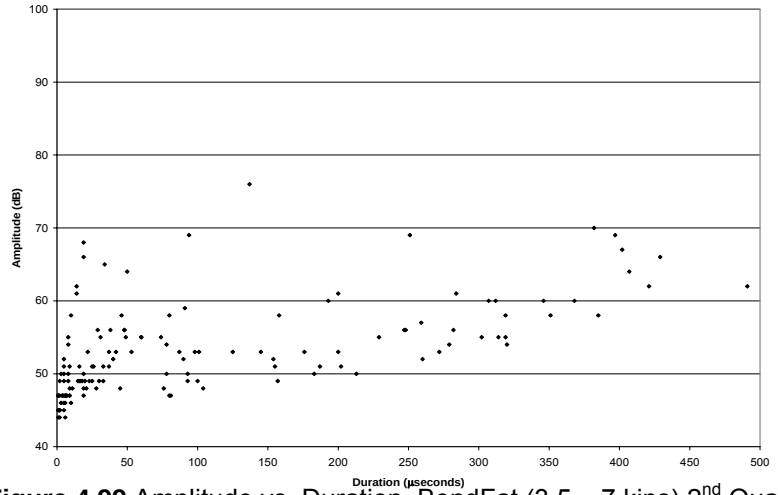


Figure 4.99 Amplitude vs. Duration, BendFat (3.5 – 7 kips) 2nd Quarter

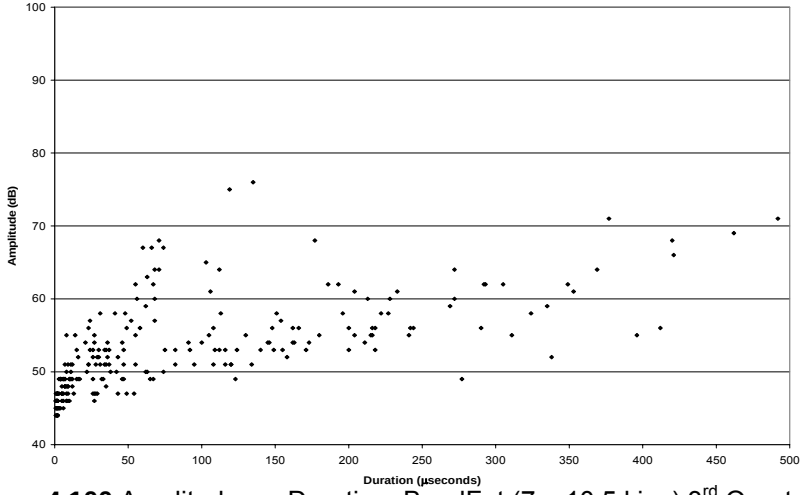


Figure 4.100 Amplitude vs. Duration, BendFat (7 – 10.5 kips) 3rd Quarter

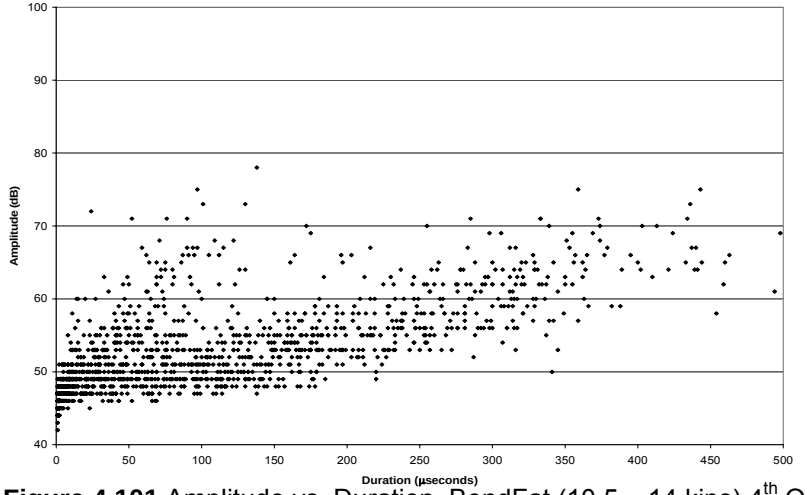


Figure 4.101 Amplitude vs. Duration, BendFat (10.5 – 14 kips) 4th Quarter

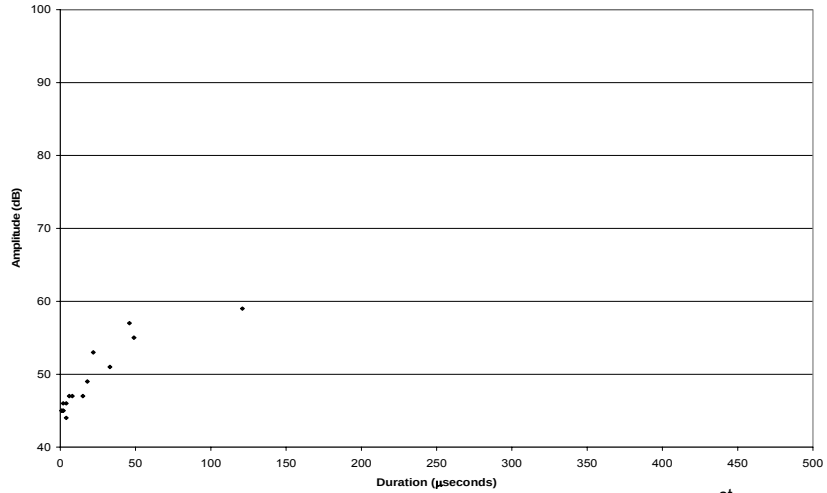


Figure 4.102 Amplitude vs. Duration, FatBend1 (0 – 3.25 kips) 1st Quarter

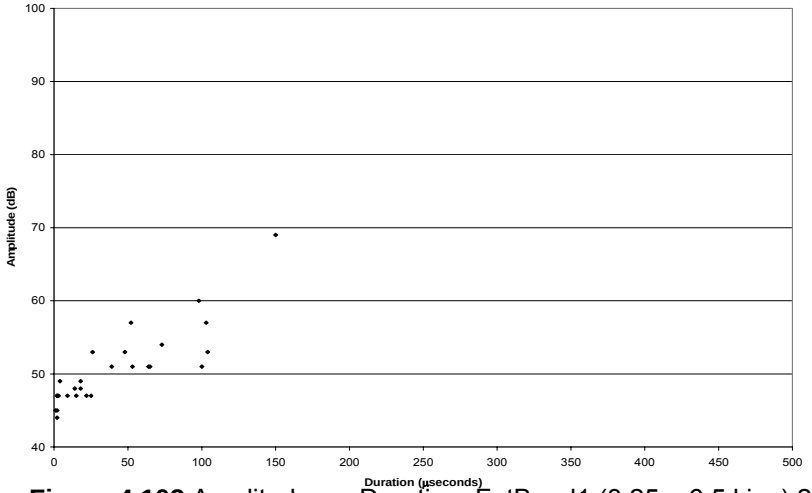


Figure 4.103 Amplitude vs. Duration, FatBend1 (3.25 – 6.5 kips) 2nd

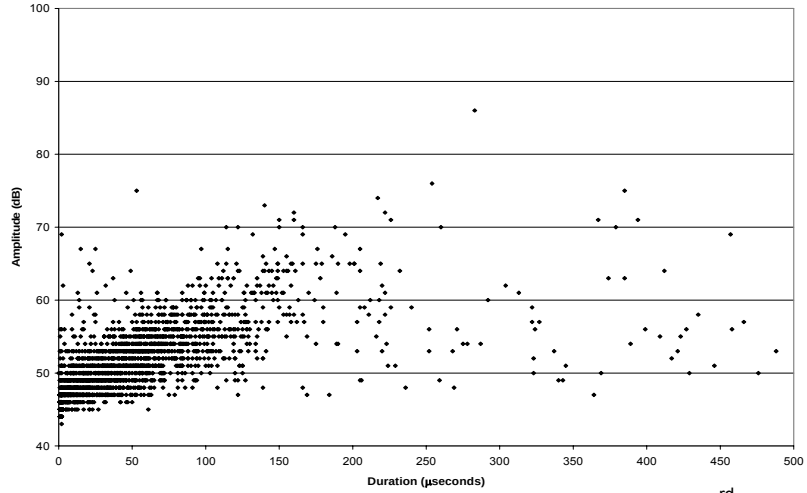


Figure 4.104 Amplitude vs. Duration, FatBend1 (6.5 – 9.75 kips) 3rd Quarter

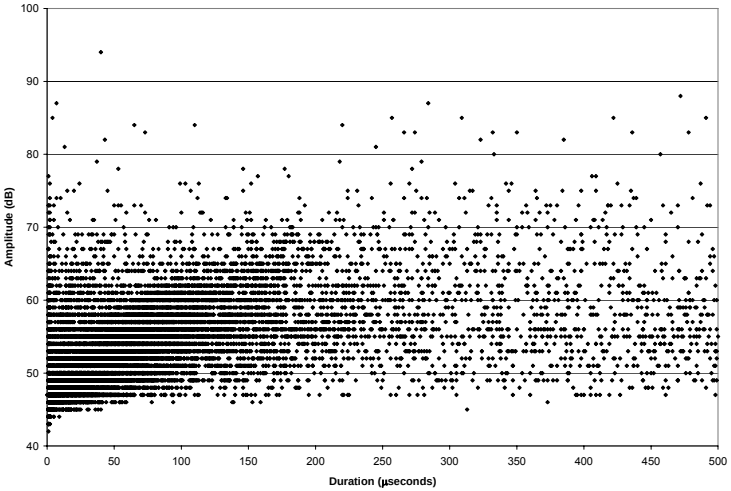


Figure 4.105 Amplitude vs. Duration, FatBend1 (9.75 – 14 kips) 4th Quarter

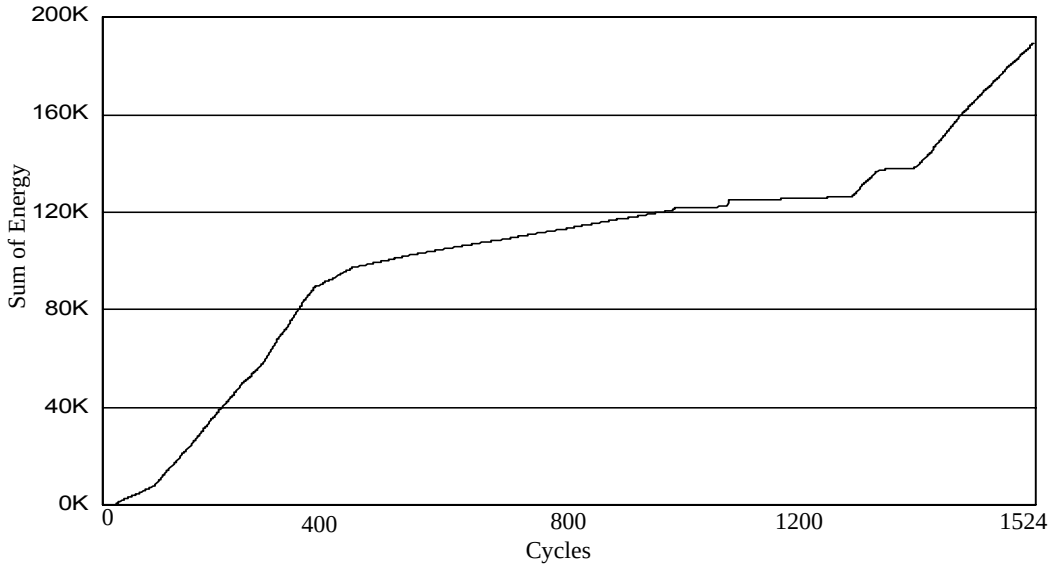


Figure 4.106 Sum of Energy vs. No of Cycles, Fat1

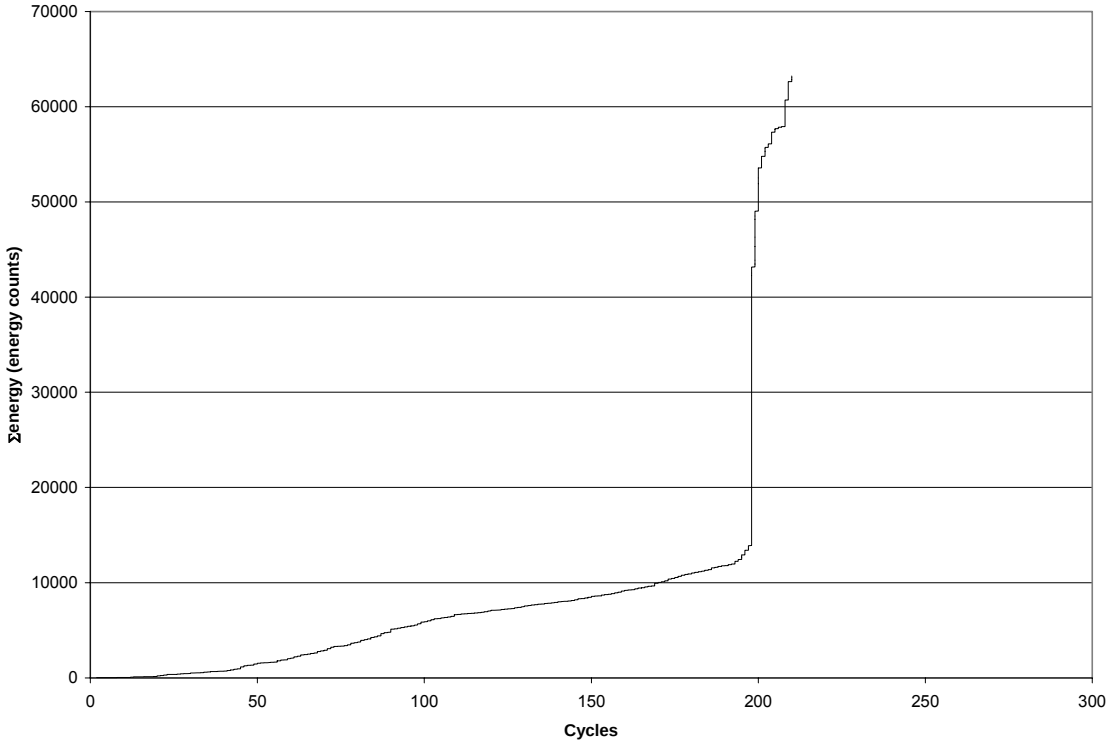


Figure 4.107 Sum of Energy vs. No of Cycles, Fat2

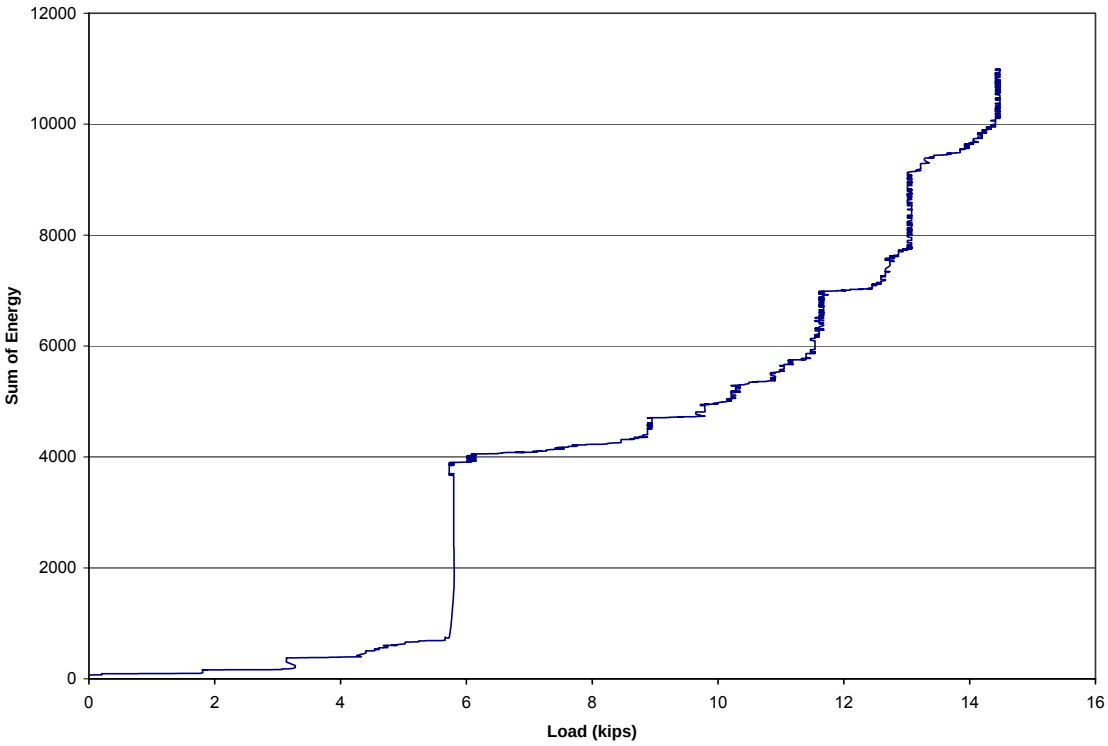


Figure 4.108 Sum of Energy vs. Load, BendFat

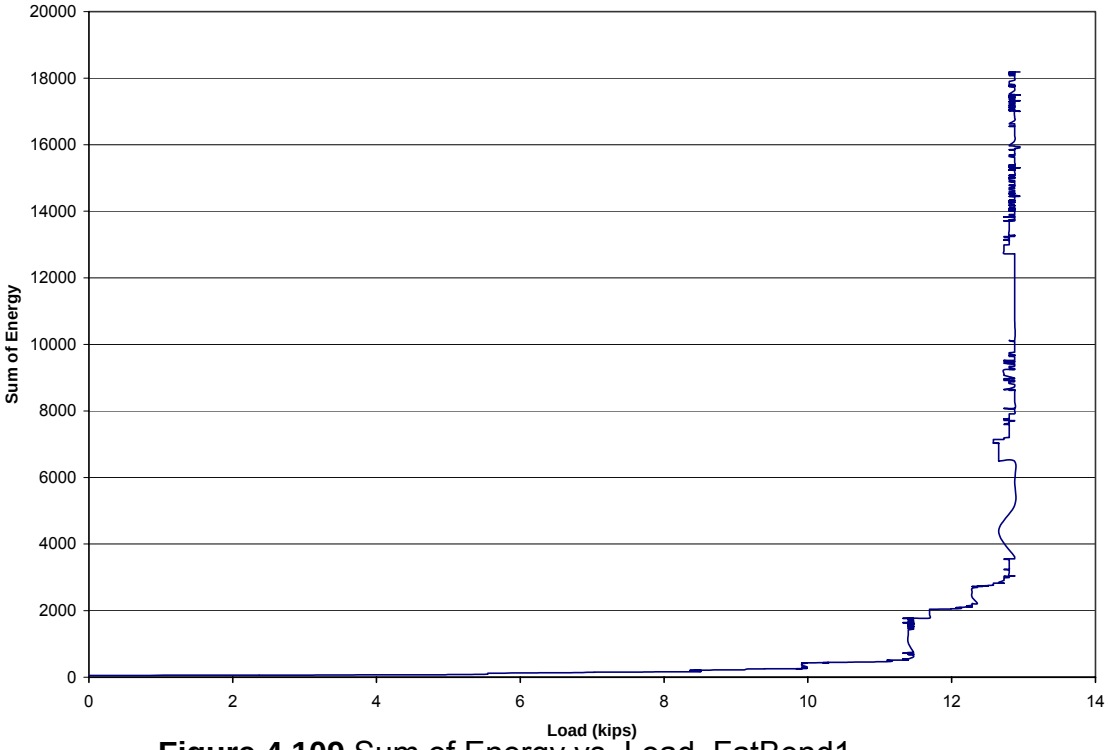


Figure 4.109 Sum of Energy vs. Load, FatBend1

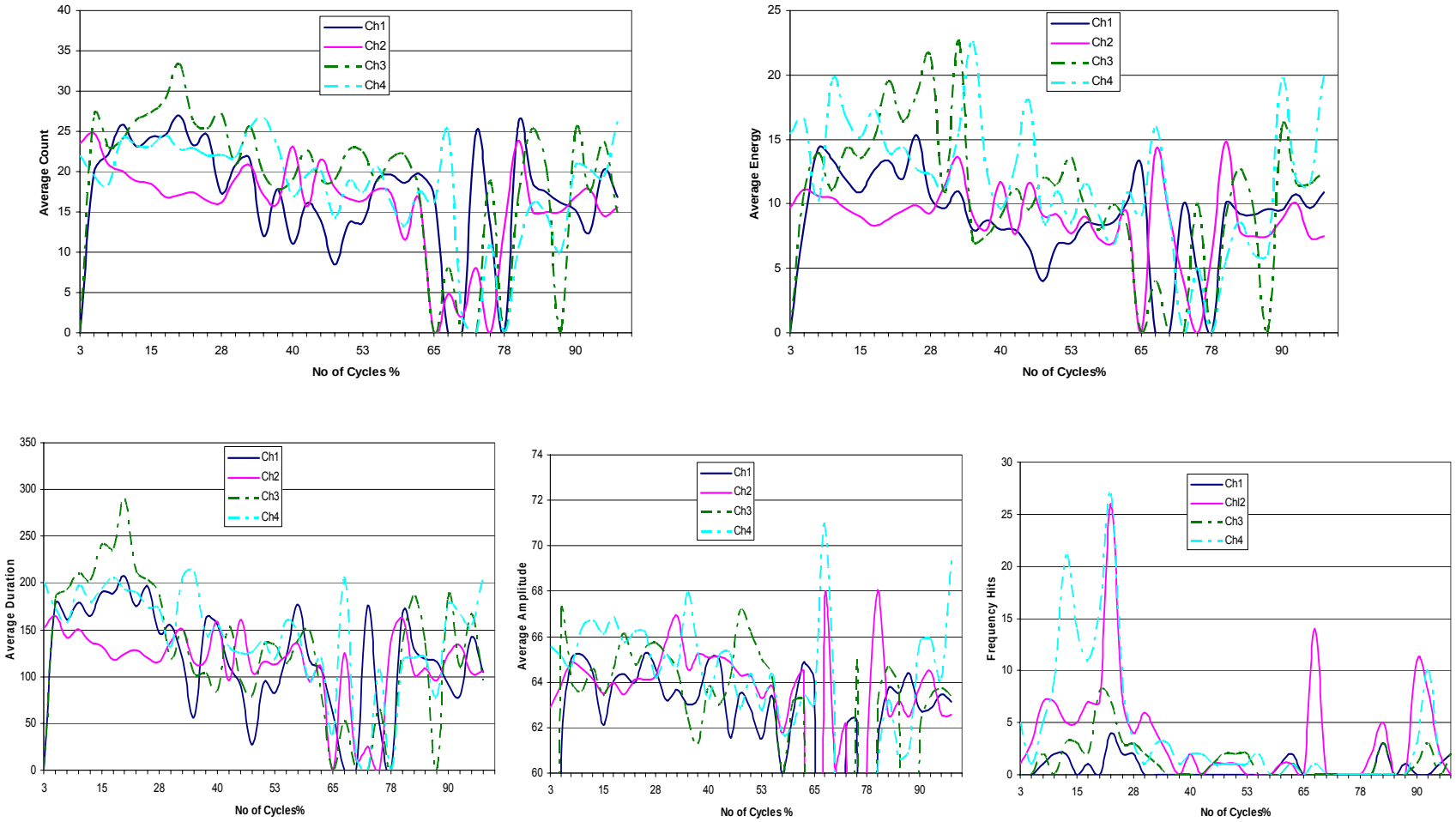


Figure 4.110 Test Fat1, Average AE parameters vs. No of Cycles %

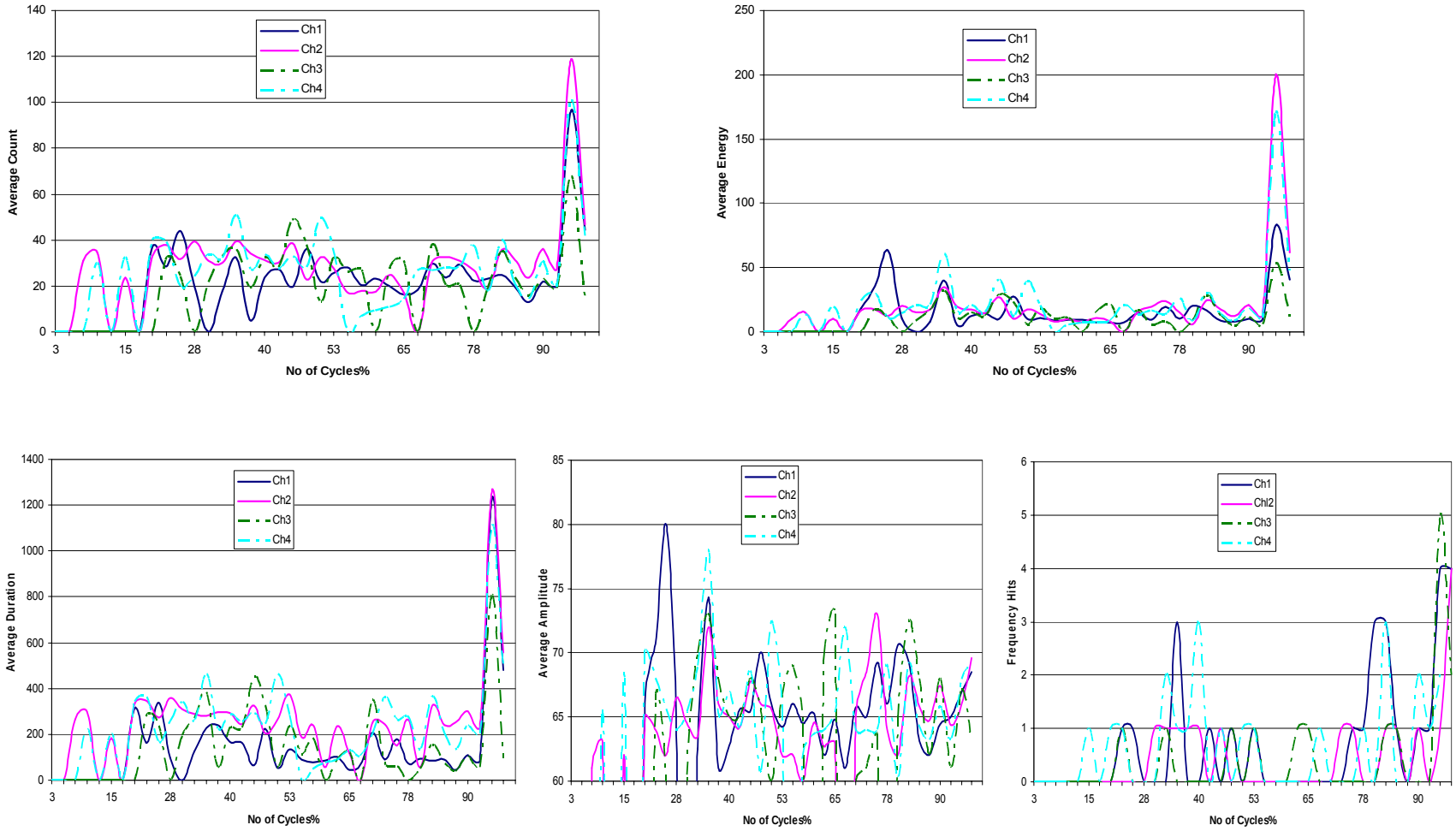


Figure 4.111 Test Fat2, Average AE parameters vs. No of Cycles %

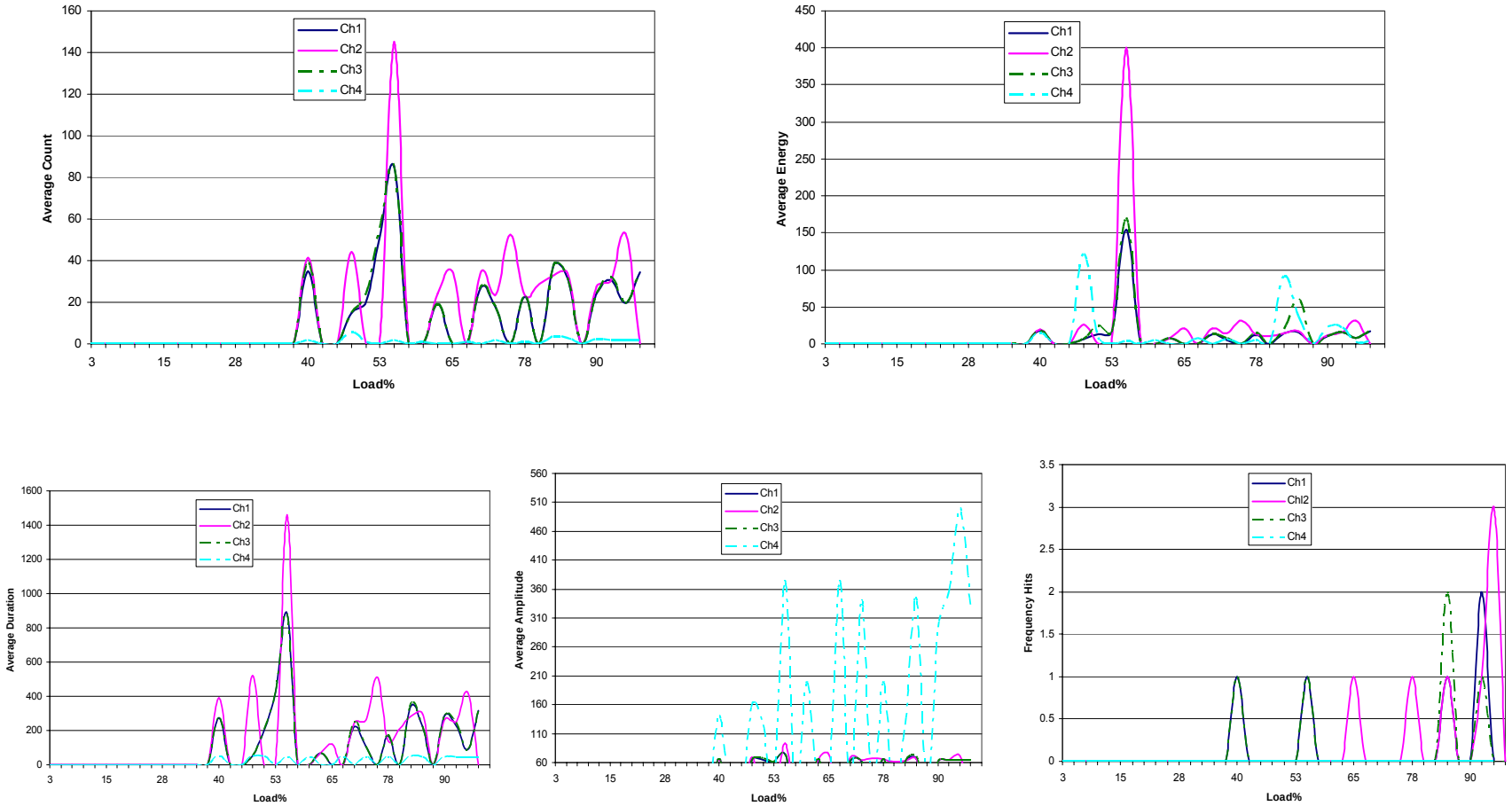


Figure 4.112 Test BendFat, Average AE parameters vs. Load %

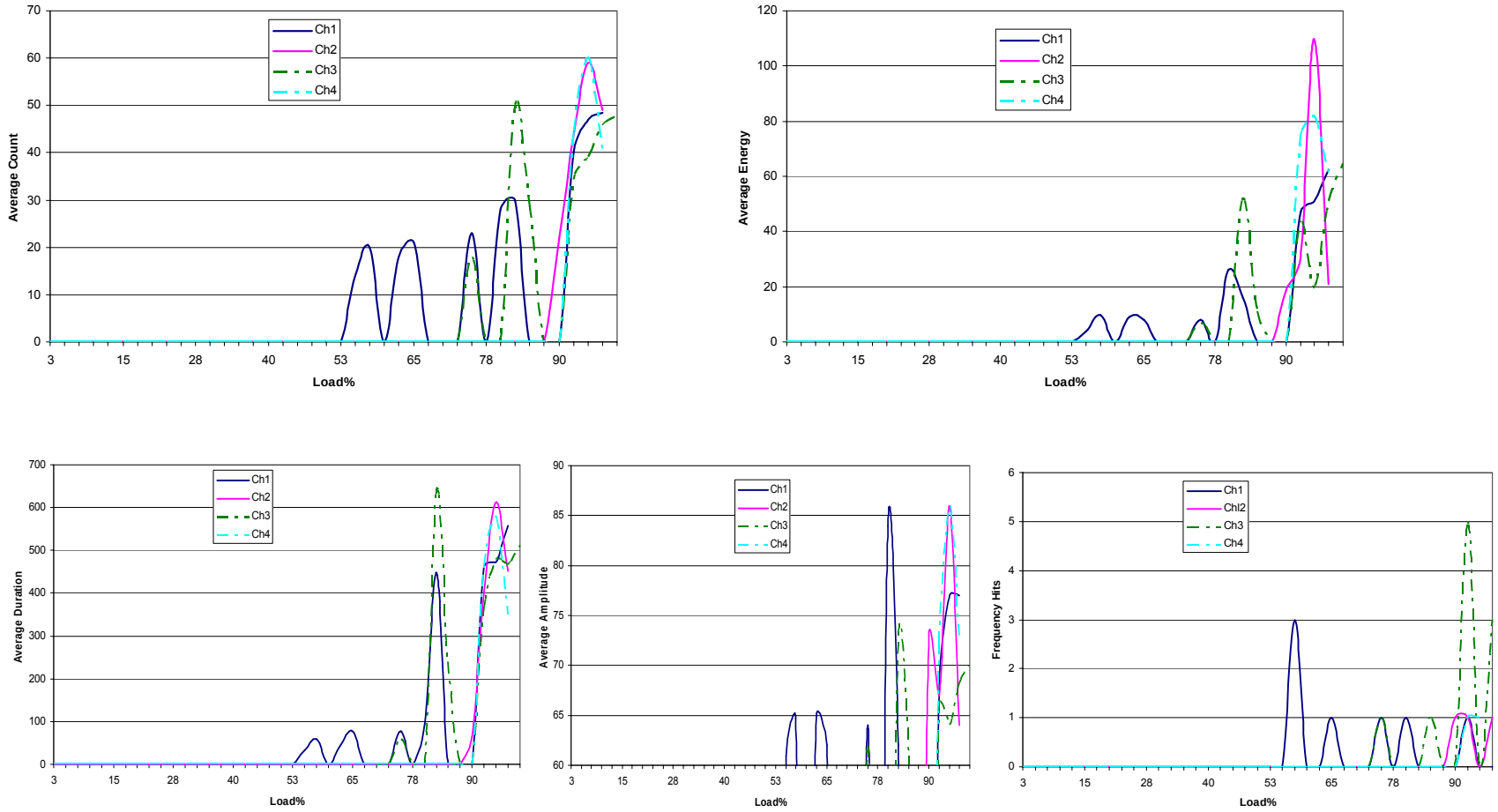


Figure 4.113 Test FatBend1, Average AE parameters vs. Load %

Chapter 5

Application of Program

5.1 Neural Network Analysis

5.1.1 The Program

A C program to do Neural Network analysis was developed by at WVU (C.L.Chen 1992). The program uses a multi-layer NN system that consists of a set of highly interconnected, nonlinear Process Units (PU's) that operate in parallel. The connection between any two PU's consists of an adjustable, continuous value termed weight. A positive or excitatory, a negative or inhibiting or a zero value can be the weight. The number of hidden numbers in the hidden layers can be arbitrary. The method to determine the hidden layer number is by trial and error (Chen and Wissawapaisal, 2000).

5.1.2 Input Matrices

The input files that have all the training set matrices and the test matrix should meet a specific criterion to be understood by the NN program.

At the beginning of the file there are three numbers. The first number is the total number of spots in the body of each matrix. The second number is the number of slots at the end of each matrix for results, or in other words, the second number defines the number of results. The final number is total number of matrices in the file.

For bending experiments done previously (Arnold, 2004), each matrix was 10 by 10 in size, therefore the first number in input the file is 100. If we are trying to predict the loading quarter, then the second number which stores the number

of results is one. We had 6 specimens, therefore the third parameter would be 24 (6 specimens x 4 quarters = 24 matrices).

5.1.3 Application of Program

Previously, each of these matrices had to be prepared manually from enormous amounts of raw data coming from the LOCAN AT. But now, using the application program we have developed through this study (Chapter 3), we can prepare the 4 matrices for each specimen with minimal effort simply by running an executable and giving very few number of parameters.

The complete assembly of the input file can also be automated by modifying the existing program. However, this should be done at the point where all the laboratory research is done and the complete structural health monitoring system is ready to be deployed in the field.

5.1.4 Loading Quarter Prediction

If we train the Neural Network program with a large number of input matrices listed with a result, which is the loading quarter number, then we can use the trained NN program to predict the loading quarter of a specimen by giving its input matrix as a test set.

The data file from the experiments is given as an input to the program. The program stores the NN input matrix in files named by the user. These input matrices are gathered into one file along with the loading quarter number and the three parameters at the top of the file. This constitutes the training set file for the NN program.

Once the training is done. The NN program establishes the connection weights. Then, the test set matrix, which is obtained from the data conversion program with the loading quarter specified, is given to the NN program. The result given by the program is the prediction of the loading quarter based on the matrix and training given to the program. Figures 5.2 and 5.3 show the execution of the NN program, where b2.txt is the training set and b2t.txt is the test set.

5.2 Results

5.2.1 Testing Procedure

In Section 3.6 it has been shown that the application program developed through this study can calculate the average values of AE parameters as good as those calculated manually. All of the data manipulations explained in chapter 3 are devised to prepare the raw AE data for Neural Network analysis. Comparisons of Neural Network predictions obtained between manually prepared input matrices and those prepared using the data conversion program are shown in Tables 5.1 and 5.2.

5.2.2 Training Matrix Preparation

Training of Neural Network is a process that adjusts the connection weights according to a learning procedure. The adjustment is processed by introducing a set of training patterns to the network. When the training process converges, the network achieves the required mapping, and weights would represent knowledge that the network has acquired (Chen and Wissawapaisal, 2000).

To prepare the training set matrix to predict the loading quarter of the tension or bending specimen, the normalized AE parameters from both sensors (sensor 1 and sensor 2) are used. Table 3.1 shows the elements to be included in the first matrix for any test for one sensor (5 rows). The first matrix for one test should include data from both sensors (10 rows), therefore each matrix has 100 values (10X10). All 4 matrices (4 quarters), from all the tests except the one currently being predicted, are included in the training set matrix, along with results for each matrix and three numbers at the beginning of file as explained in Section 5.1.2.

From the output of the data conversion program, we get matrices for one sensor in one file. We have to combine those matrices together to make the test and training matrix for Neural Network analysis. This can be easily done using text file editor such as Notepad in a PC, a sample input file is shown in Figure 5.4(a).

5.2.3 Test Matrix Preparation

A test matrix just has the four matrices from the test being predicted along with the results and the three numbers explained in Section 5.1.2. The Neural Network program then gives the correction rate, the prediction of result and the error in the prediction. A sample test file, b2t.txt is shown in Figure 5.4(b)

5.2.4 Executing the Neural Network Program

The Neural Network executable is called NBP.exe; this is placed in a folder along with the training and test matrix files. In command prompt, the

directory is changed into the folder and the executable is run by giving train and test matrix file names as command line arguments as shown below:

```
C:\nn\nbp train.txt test.txt
```

Also, the user has to give some parameters for Neural Network analysis such as the number of layers. Three layered networks are used for analysis. The layer[0] is the first layer, which is the input matrix; the dimensions of layer[0] therefore are 10 rows and 10 columns. Then, the second layer layer[1], which is a hidden layer, is defined. It was decided by trial and error that 6 hidden units should be used for the analysis. Therefore layer[1] dimensions for this study were 1 row and 6 columns. The row scope values for Plane[1] can be left as default. The third layer, layer[2], is the result matrix. As we are trying to predict the loading quarter, the result is just one number, therefore, layer[2] dimensions are 1 row and 1 column. Figure 5.1 schematically shows the three layered Neural Network used in this study.

After giving all the layer definitions and establishing the input and output ranges for the data, the Neural Network program allows us to select from the Main Menu. From this menu, we have to first train the Neural Network and Neural Net simulator then Recall/test to see the prediction. The screen captures of the Neural Network program being executed are shown in Figures 5.2 and 5.3.

This procedure explained above is performed for each test, once using manually prepared matrix, and once using the matrix prepared by using data conversion program.

5.2.5 Manual vs. Program

Prediction performance of the NN is determined by the Correct Rate (CR) defined as

$$CR = \frac{\text{(Number of patterns correctly classified)}}{\text{(Total number of patterns of the data sets)}} \text{ X } 100\%$$

We have tried to predict the loading quarter of the specimen, so each correctly predicted quarter gives a CR of 25%. The prediction is valid if the value predicted is ±40% of the actual value; that is, if the NN predicted a value of 1.40 or 0.60 for first quarter then the prediction is acceptable.

The effectiveness of the NN input matrices prepared from raw AE data can be measured by looking at the CR values from NN prediction and the error values. One can clearly see from Tables 5.1 and 5.2 that the input matrices prepared by using the program have equal or greater Correct Rates when compared to manually prepared matrices. Tables 5.1 and 5.2 show the results from the specimens tested by Arnold (2003) and the AE data were shown in Chapter 3 for the comparison between manual and program prepared AE parameters, 9 bending specimens and 6 tension specimens were selected for the NN analysis. The specimen numbering is identical as described by Arnold (2003). One more observation made while executing the NN program is that the data from the conversion program reaches convergence in about half the number of cycles taken by manually prepared matrices. The input matrices prepared by

the application program are also more consistent than those prepared manually. This is the main reason for better results from the NN analysis.

5.3 Summary

- A comparison of Neural Network prediction obtained between manually prepared input matrices and those prepared by using the application is presented (Tables 5.1 and 5.2).
- The Neural Network program predicts more accurately and efficiently when the input matrix is prepared using the application software developed through this study.

B2				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	1.04	-0.04	0.83	0.17
2	1.91	0.09	2.28	-0.28
3	3.87	-0.87	3.27	-0.27
4	3.86	0.14	3.81	0.19
CR	75		100	

B3				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	0.74	0.26	1.09	-0.09
2	2.36	-0.36	2.62	-0.62
3	3.86	-0.86	3.53	-0.53
4	3.71	0.30	3.87	0.13
CR	75		75	

B4				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	1.31	-0.31	1.04	-0.04
2	2.79	-0.79	2.23	-0.23
3	3.57	-0.57	3.61	-0.61
4	3.89	0.11	3.75	0.25
CR	50		75	

B5				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	0.83	-0.17	0.90	0.10
2	2.26	-0.26	2.35	-0.35
3	3.20	-0.20	2.62	0.38
4	2.41	1.59	3.57	0.43
CR	75		75	

Table 5.1a Neural Network Result Comparisons for bending data

B6				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	1.14	-0.14	1.37	-0.37
2	2.35	-0.35	3.25	-1.25
3	1.96	1.04	2.97	0.03
4	3.56	0.44	3.85	0.15
CR	50		75	

B7				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	1.16	-0.16	0.91	0.09
2	2.64	-0.64	2.28	-0.28
3	2.67	0.33	3.02	-0.02
4	3.83	0.17	3.89	0.11
CR	75		100	

B8				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	3.03	-2.03	1.53	-0.53
2	3.18	-1.18	2.92	-0.92
3	3.35	-0.35	2.79	-0.21
4	3.82	0.18	3.82	0.18
CR	50		50	

B9				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	0.63	0.37	0.98	0.02
2	1.25	0.75	1.86	0.14
3	2.53	0.43	2.36	0.64
4	3.77	0.23	3.90	0.10
CR	50		75	

Table 5.1b Neural Network Result Comparisons for bending data

B10				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	0.63	0.37	0.98	0.02
2	1.25	0.75	1.86	0.14
3	2.53	0.43	2.36	0.64
4	3.77	0.23	3.90	0.10
CR	50		75	

Table 5.1c Neural Network Result Comparisons for bending data

T1				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	1.68	-0.68	2.04	-1.04
2	1.58	0.42	1.61	0.39
3	2.85	0.15	2.90	0.10
4	3.61	0.39	3.70	0.30
CR	50		75	

T2				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	0.68	0.32	1.04	-0.04
2	1.90	0.10	2.08	-0.08
3	2.71	0.29	2.32	0.68
4	3.56	0.44	3.83	0.17
CR	75		75	

Table 5.2a Neural Network Result Comparisons for tension data

T3				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	1.38	-0.38	1.49	-0.49
2	2.79	-0.79	2.49	-0.49
3	3.52	-0.52	3.26	-0.26
4	3.83	0.17	3.88	0.12
CR	50		50	

T4				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	1.22	-0.22	1.00	0
2	2.58	-0.58	2.54	-0.54
3	2.95	-0.05	3.06	-0.06
4	3.26	0.74	3.77	0.23
CR	50		75	

T5				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	1.37	-0.37	1.18	-0.18
2	2.70	-0.70	2.49	-0.49
3	3.36	-0.36	3.42	-0.42
4	3.86	0.14	3.83	0.17
CR	75		50	

T6				
QR	Manual		Program	
	Predicted	Error	Predicted	Error
1	0.48	0.52	0.57	0.43
2	2.10	-0.10	1.68	0.32
3	3.49	-0.49	2.75	0.25
4	3.89	0.11	3.81	0.19
CR	50		75	

Table 5.2b Neural Network Result Comparisons for tension data

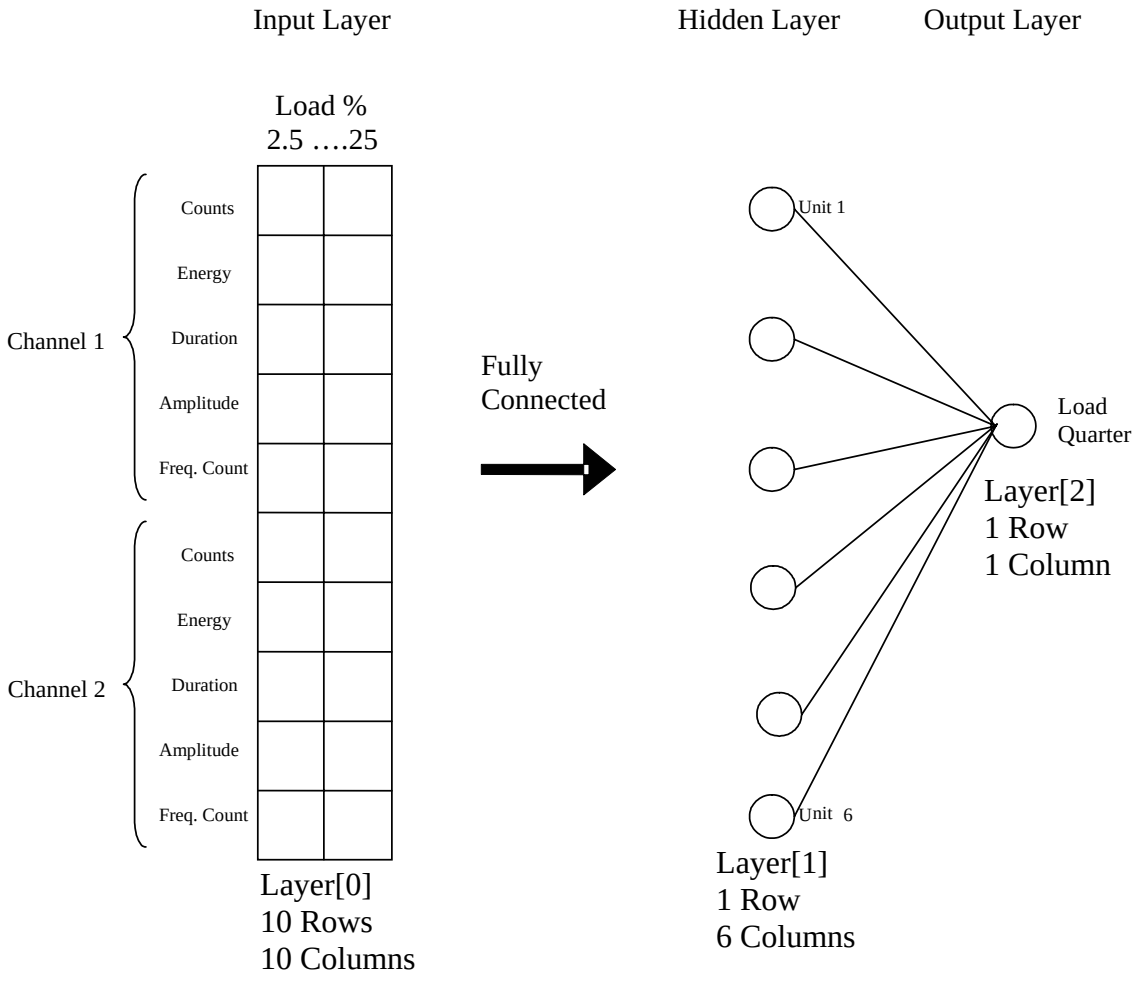


Figure 5.1 NN Architecture for Load Quarter Predictor (Arnold, 2003)

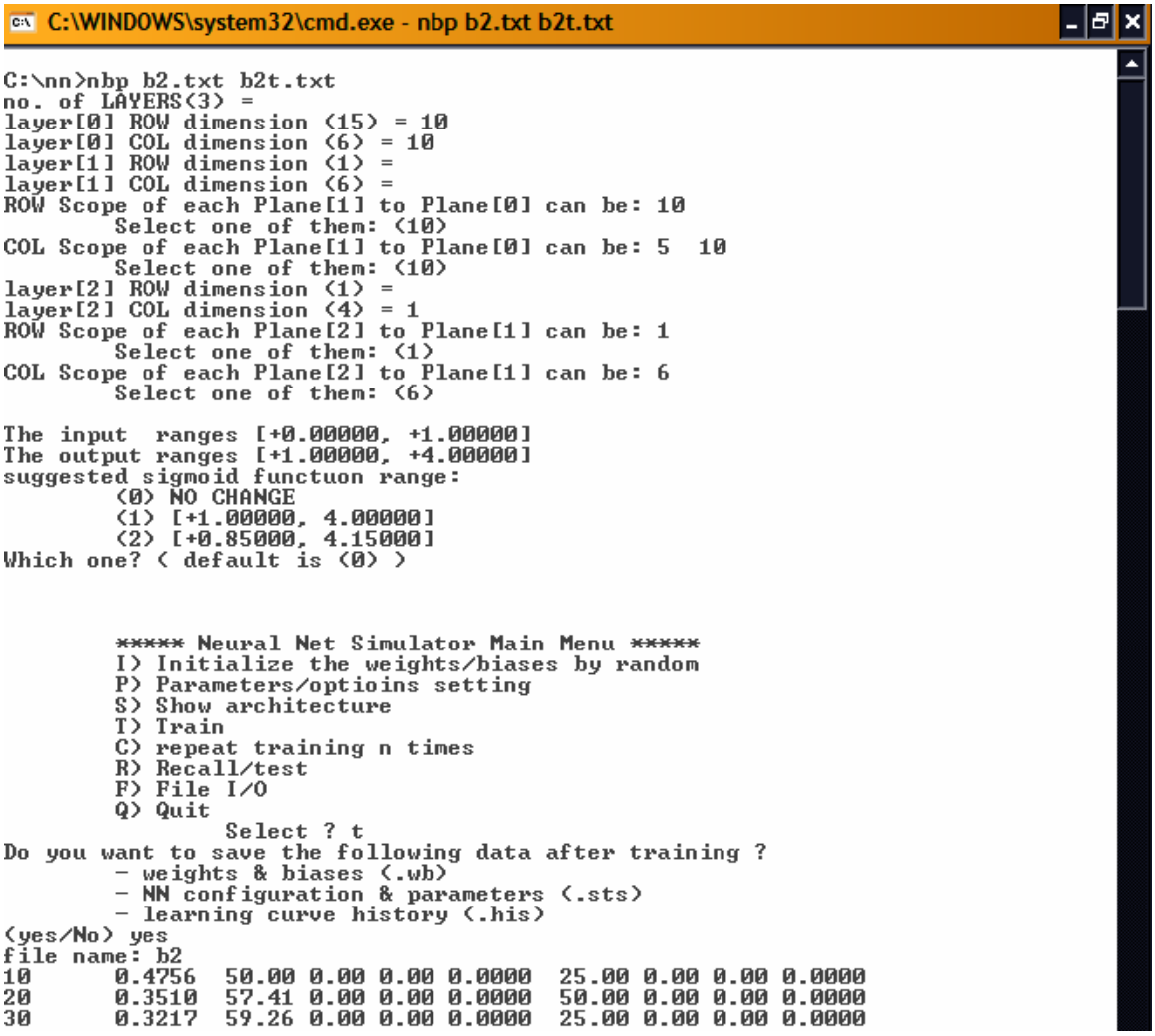


Figure 5.2 First screen capture of NN program being executed

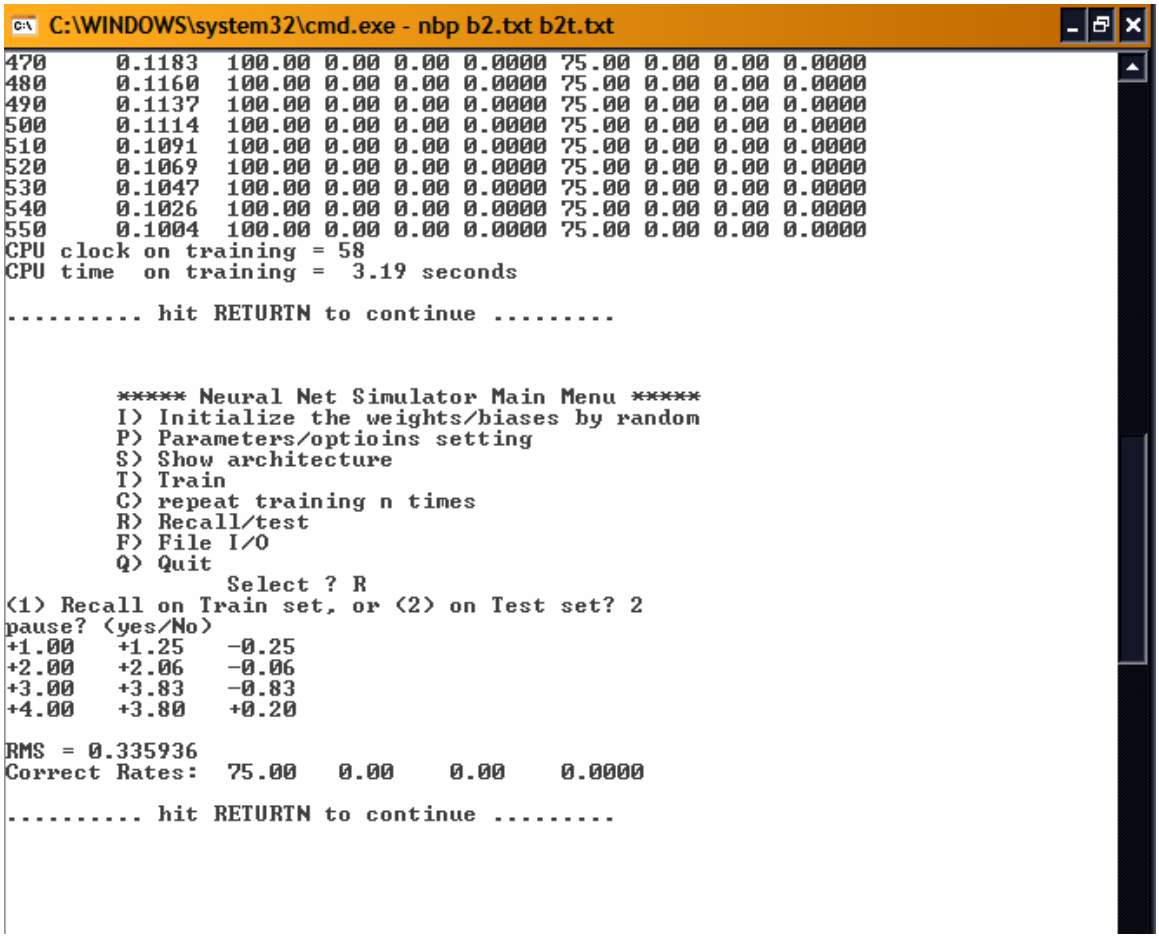
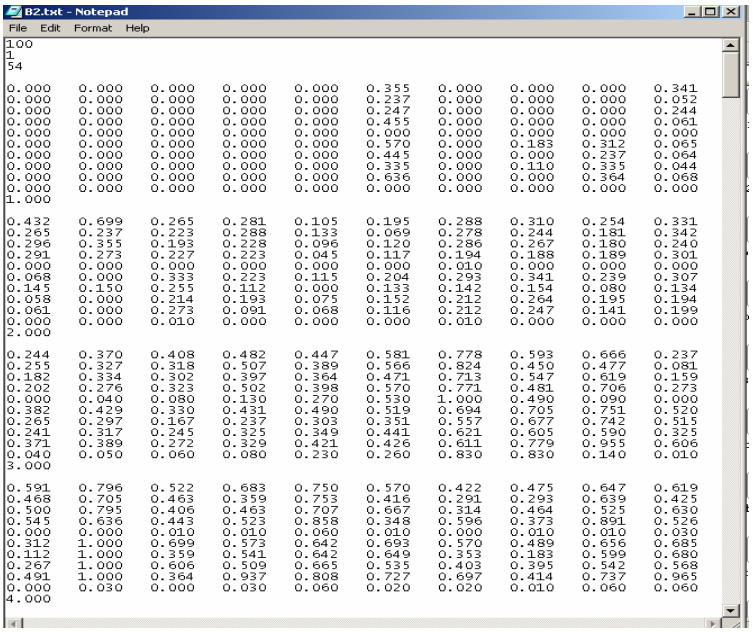
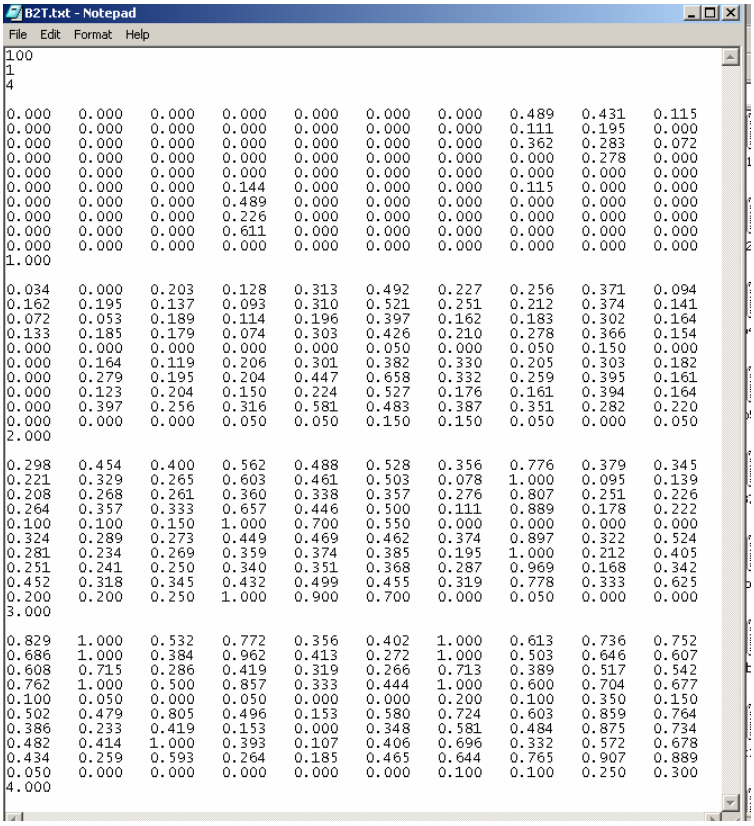


Figure 5.3 Second screen capture of NN program being executed



(a)



(b)

Figure 5.4 NN input matrices (a) B2.txt (b) B2T.txt

Chapter 6

Conclusions, Applications & Recommendations

6.1 Conclusions

To extensively use composite materials in the construction industry, the development of a real-time non-destructive evaluation technique is essential. It has been proven by previous research that usage of acoustic emission evaluation along with Neural Network analysis is a good non-destructive evaluation technique for FRP composite material.

Some of the conclusions reached from this research are listed below

1. awk programs were developed to automate the computations done on raw data collected from LOCAN AT.
2. Noise elimination algorithms were developed in order to eliminate the signals generated by loading arms.
3. Seven individual awk scripts should be run before the raw data transforms into 4 sets of 5 X 10 matrices, which will be the input to Neural Network program.
4. A C program was developed to automate the process of calling the awk scripts in sequence and obtaining the experimental parameters from user.
5. The average AE parameter values obtained from application software are in good agreement with those obtained by manual conversion (Refer Figures 3.11 to 3.90).
6. Two tension tests were conducted on FRP samples made by using a resin mixed with nano-engineered clay material.

7. Observation of many short duration signals having high amplitude values in the *Amplitude Vs. Duration* plots can be thought of as increased brittleness in the FRP material due to the addition of nano-engineered clay material in the resin. More experiments should be conducted to confirm this.
8. Three bending tests were conducted on 2 inch wide Prodeck4 samples. The samples failed in shear stress induced by the webs close to the supports.
9. The Neural Network program correctly predicted the third loading quarter for all the samples. This proves that AE testing along with the application developed, and the NN program can be used for non-destructive structural evaluation of FRP materials.
10. Two fatigue tests were conducted on 6 feet long, single celled, square cross sectioned FRP bridge decks.
11. AE parameters could predict the final failure of the fatigue specimens.
12. The NN program predicts more accurately and efficiently when the input matrix is prepared by using the application software developed through this study (Tables 5.1 and 5.2).
13. The AE data depends on the material more than the design of structure being tested.
14. Damage level prediction can be done more effectively by using the application software developed in this study to convert raw AE data into a better format which can be used as an input matrix for NN program.

6.2 Applications

The conclusions made from this research can be applied in the following future applications:

1. The application program developed through this research can be used to develop real-time structural health monitoring system for structures made of structure made of FRP composite materials.
2. The application program developed can be used for tension tests, bending tests, and fatigue tests.
3. The conversion of AE data to Neural Network input matrix is fast and more flexible, as the awk scripts can be very easily modified.

6.3 Recommendations

The following recommendations are made for future research:

1. More fatigue tests should be conducted and analyzed.
2. There has been a tremendous advancement in the field of Neural Network algorithms in the past 10 years. Usage of feed forward networks can be considered for future research.
3. A real-time structural health monitoring system can be developed by connecting the C program developed through this research to the AE data acquisition program and the Neural Network program.
4. Research is needed to develop wireless sensors to record AE data from structures.

References

1. Arnold, Ryan Edward "Acoustic Emission Evaluation of FRP Composite Specimens in Tension and Bending", Masters Thesis, West Virginia University May, 2003.

2. ASTM Standards.

3. Chen, C. L., "Training the Multi-layer Neural Nets by Setting the Initial Weights," Ph.D. Dissertation, West Virginia University, Morgantown, 1992.

4. Chen, H. L., and Chen, C. L., " Applying Neural Network to Acoustic Emission Signal Processing", *Fourth International Symposium on Acoustic Emission From Composite Materials*, The American Society for Nondestructive Testing, Seattle, Washington, pp. 273-281, July, 1992.

5. Chen, H. L., Cheng, C. T., Chen, S. E., "Determination of Fracture Parameters of Mortar and Concrete Beams by Using Acoustic Emission", *Materials Evaluation*, pp. 888-894, July, 1992.

6. Chen, H. L., He, Yidong, and Superfesky, Michael, "Acoustic Surface Waveguides for Acoustic Emission Monitoring of Fiber-Reinforced Plastic Structures', *Materials Evaluation*, Vol. 52, No. 9, pp. 1112-1116, September, 1994.

7. Chen, H. L., Sami, Z., and GangaRao, H. V., " Identifying Damages in Stressed Aramid FRP Bars Using Acoustic Emission", *Dynamic Characterization of Advanced Materials*, ASME, NCA-Vol. 16/ AMD-Vol. 172, pp. 171-178, 1993.

8. Chen, H. L., and Wissawapaisal, Komwut, "Study of Acoustic Surface Waveguides on Reinforced Concrete Slabs" *Journal of Nondestructive Evaluation*, Vol. 19, No. 4, 2000.

9. Chen, H. L., Zhou, H. W., and GangaRao, H. V., "Characterization of West Virginia Hardwood Using Acoustic Emission and Vibration Method", *Review of Progress in Quantitative Nondestructive Evaluation*,Vol. 11, Thompson, D. O., and Chimenti, D. E. ed., Plenum Press, New York, 1992.

10. Choi, J. H., and Chen, H. L., "Design Considerations of GFRP-Reinforced CRCP," accepted for *ACI Special Publication for Sessions on "Field Applications of FRP Reinforcement: Case Studies" for the Boston Convention, Sept. 27-Oct. 1, 2003*.

11. Choi, Jeong-Hoon, “The Fracture Analysis and Remaining Life Estimation of the AVLB Sub-Components”, Master’s Thesis, West Virginia University, Morgantown, November 2000.

12. Dale Dougherty and Arnold Robbins, “sed and awk 2nd edition”, O’reilly publications, March, 1997.

13. Eberhart, Russ, Simpson, Pat, and Dobbins, Roy, *Computational Intelligence PC Tools*, An Indispensible Resource for the Latest in: Fuzzy Logic, Neural Networks, Evolutionary Computing, AP Professional, Orlando, Florida, 1996.

14. *Fiber Reinforced Polymer Composite Bridges of West Virginia*, FHWA Pub. No. FHWA-ERC-2-002, Compiled by Constructed Facilities Center, West Virginia University, Morgantown, West Virginia, 2001.

15. “FRP Materials, Manufacturing Methods and Markets”, *Composites Technology YellowPages*, pg. 6, 1999

16. Fultineer, R. D., “Study of Fatigue Cracks in Steel Bridge Components Using Acoustic Emission”, Master’s Thesis, West Virginia University, Morgantown, 1997.

17. Grabec, Igor, Sachse, Wolfgang, and Govekar, Edvard. “Solving AE Problems by a Neural Network”, *Acoustic Emission: Current Practice and Future Directions*, ASTM STP 1077, Sachse/Roget/Yamaguchi ed., pp. 165-182, 1991.

18. Graff, Karl F., *Wave Motion in Elastic Solids*, Dover Publications, 1991.

19. Harrold, R. T., and Sanjana, Z. N., “Acoustic Waveguide Monitoring of the Cure and Structural Integrity of Composite Materials”, *Polymer Engineering and Science*, Vol. 26, No. 5, mid-March, 1986.

20. Hawkins, Neil M., McCabe, W. Martin, and Nobuta, Yoshinobu, “Use of Acoustic Emission to Detect Debonding of Reinforcing Bars in Concrete”, *Progress in Acoustic Emission IV*, The Japanese Society for NDI, 1988.

21. Higo, Yakichi, and Hidehiro, Inaba, “The General Problems of AE Sensors”, *Acoustic Emission: Current Practice and Future Directions*, ASTM STP 1077, Sachse/Roget/Yamaguchi ed., pp. 7-24, 1991.

22. McIntire, Paul, *Nondestructive Testing Handbook*, 2nd ed., Vol. 5 Acoustic Emission Testing, ASNT, 1987.

23. Mitchell, Tom M., *Machine Learning*, WCB/McGraw-Hill, Portland, Oregon, 1997.

24. O'Connor, Jerome S., "New York's Experience with FRP Bridge Decks", *Polymer Composites II: Applications of Composites in Infrastructure Renewal and Economic Development*, Creese, Robert C., and GangaRao, Hota ed., Organized by Constructed Facilities Center, West Virginia University, Morgantown, West Virginia, 2001.

25. PCI, *PCI Design Handbook: Precast and Prestressed Concrete*, Fourth Edition, Chicago Illinois, 1992.

26. Pollock, A. A., "Practical Guide to Acoustic Emission Testing", *LOCAN AT User's Manual*, REV. 1.0, 1988.

27. Reeve, Scott R., "FRP Composite Bridge Decks: Barriers to Market Development",*Polymer Composites II: Applications of Composites in Infrastructure Renewal and Economic Development*, Creese, Robert C., and GangaRao, Hota ed., Organized by Constructed Facilities Center, West Virginia University, Morgantown, West Virginia, 2001.

28. Robert, Jeffrey L., "Replacement of the Bridge on MD 24 over Deer Creek in Harford County, Maryland, Utilizing a Fiber-Reinforced Polymer (FRP) Deck", *Polymer Composites II: Applications of Composites in Infrastructure Renewal and Economic Development*, Creese, Robert C., and GangaRao, Hota ed., Organized by Constructed Facilities Center, West Virginia University, Morgantown, West Virginia, 2001.

29. Soneji, J., Hu, C., Chaudhri, M., Faqiri, A. Gillespie, J. W., Jr., Eckel, D. A., II, Mertz, D. R., and Chajes, M.J., "Use of Glass-Fiber-Reinforced Composite Panels to Replace the Superstructure for Bridge 351 on N387A over Muddy Run", *Polymer Composites II: Applications of Composites in Infrastructure Renewal and Economic Development*, Creese, Robert C., and GangaRao, Hota ed., Organized by Constructed Facilities Center, West Virginia University, Morgantown, West Virginia, 2001.

30. Tong, L., Bannister, M., Mouritz, A., and Oakeley, A., *3D Fibre Reinforced Polymer Composites*, 1st ed., Elsevier Health Sciences, 2002.

31. Werts, William M., "PennDOT's First FRP Deck, *Polymer Composites II: Applications of Composites in Infrastructure Renewal and Economic Development*, Creese, Robert C., and GangaRao, Hota ed., Organized by Constructed Facilities Center, West Virginia University, Morgantown, West Virginia, 2001.

32. Wevers, M., Verpoest, I., De Meester, P., and Aernoudt, E., “Identification of Fatigue Failure Modes in Carbon Fibre Reinforced Composites with the Energy Discriminating Acoustic Emission Method”, *Acoustic Emission: Current Practice and Future Directions*, ASTM STP 1077, Sachse/Roget/Yamaguchi ed., pp. 416-423, 1991.

33. Yeh, Shu-Kai, and Cho, Eung Ha, personal communication, 2003.

Appendix A

Bending.awk

```
BEGIN{
}
{
printf("%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n",$1,$2,$4,$5,$6,$7,$8,$9,$10) >
"bend.txt"
}
END{
}
```

Signalnumbering.awk

```
BEGIN {i=1
}
{
#NR is the Row NUmber
if(NR==1)
time=$1
if(NR==2)
prev=$1
if(NR>2)
 {
  diff=$1-prev;
  if(diff<time)
  {
   i=i+1;
   prev=$1;
  }
  else
  prev=$1;

printf("%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n",$1,$2,$3,$4,$5,$6,$7,$8,$9,i) >
"awk1.txt"
}

}
END{
}
```

Eliminatenoise.awk

```
BEGIN{
signal=0;
}
{
if(NR==1)
{
one=$1
two=$2
}
else if(NR>2)
{
#only channel 6 and channel 2 are sensors on the grips
if($3==one||$3==two)
signal=$10;
else if($10>signal)
printf("%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n",$1,$2,$3,$4,$5,$6,$7,$8,$9) >
"awk2.txt"
}
}
END{
}
```

Code1.awk

```
BEGIN{
}

{

if(NR==1)
{
m=$1
i=$2
threshold=$3
avgfailload=$4
channel=$5
}
else
{
if($8>=threshold)
```

```
{
if($3==channel) #Eliminating all Channels except one
{
#Converting the Voltage(parameter1) to Load
a=$2*m + i;

#Converting Load into Load percentage
a = (a/avgfailload)*100;

#if load % drops morethan 2.5% stop proceeding this indicates that the material
already failed or experimental error #(unloading between loading)
if((previousload-a)>=2.5)
exit;

printf("%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n",a,$4,$5,$6,$7,$8,$9) > "awk3.txt"
previousload=a;
}
}
}

}

End{
}
```

Code2.awk

```
BEGIN{m=1;
}
{
if(NR==1)
{
watchamp=$1;
freqlower=$2;
frequpper=$3;
}
else
{
if(NR==2&&$1>2.5)
{
 while($1>m*2.5)
  {
   printf("%f\t0.000000\t0.000000\t0.000000\t0.000000\t0.000000\n",m*2.5) >
"awk4.txt"
   m=m+1
```

```
  }
}
bool=1;
do
{
#Dividing the data into every 2.5% of load percentage
 if($1<=(m*2.5))
 {
  load=$1;
  #When coming from if loop donot add count=count+$3
  if(bool!=0)
  {
   if($6>=watchamp && $7>=freqlower && $7<=frequpper)
   hit=hit+1;
   count=$3+count;     #Sum of Counts
   energy=$4+energy;
   duration=$5+duration;
   amplitude=$6+amplitude;
  }
  n=n+1; #No of records to use in finding avg()
  bool=1;
 }
 else
  {

  if(n==0)
  n=1;
  count=count/n;
  energy=energy/n;
  duration=duration/n;
  amplitude=amplitude/n;

  #If load is jumping from one step to like 2 or more steps like 7.8 to 19.882
  #we should stop printing 7.8 record over and over again, That is done by this if
statement
  if($1==printingload)
  {
   printf("%f\t0.000000\t0.000000\t0.000000\t0.000000\t0.000000\n",m*2.5) >
"awk4.txt"
   }
  else if(NR!=2)
  {
  printingload=$1;
#this is actual values into the matrix
  printf("%f\t%f\t%f\t%f\t%f\t%d\n",m*2.5,count,energy,duration,amplitude,hit) >
"awk4.txt"
```

```
  }

  hit=0;
  count=$3;
  energy=$4;
  duration=$5;
  amplitude=$6;
  if(amplitude>=watchamp && $7>=freqlower && $7<=frequpper)
  hit=1;
  m=m+1;
  n=0;
  bool=0;
  }

}while(bool==0)
}
}
END{
}
```

Mamin.awk

```
BEGIN{
mincount=100
minenergy=100
mindur=100
minampl=100
minhit=100
}
{

if($2>maxcount)
maxcount=$2
if($3>maxenergy)
maxenergy=$3
if($4>maxdur)
maxdur=$4
if($5>maxampl)
maxampl=$5
if($6>maxhit)
maxhit=$6
if($2!=0&&$2<mincount)
mincount=$2
if($3!=0&&$3<minenergy)
minenergy=$3
```

```
if($4!=0&&$4<mindur)
mindur=$4
if($5!=0&&$5<minampl)
minampl=$5
if($6!=0&&$6<minhit)
minhit=$6

}
END{
printf("%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f",maxcount,mincount,maxenergy,min
energy,maxdur,mindur,maxampl,minampl,maxhit,minhit) > "maxmins.txt"
}
```

Matrix.awk

```
BEGIN{
}
{
if (NR==1)
{
maxcount=$1
mincount=$2
maxener=$3
minener=$4
maxdur=$5
mindur=$6
maxamp=$7
minamp=$8
maxhit=$9
minhit=$10
file=$11
}

a[NR,1]=($2-mincount)/(maxcount-mincount)
if (a[NR,1] < 0)
a[NR,1]=0
else if (a[NR,1] > 1)
a[NR,1]=1
a[NR,2]=($3-minener)/(maxener-minener)
if (a[NR,2] < 0)
a[NR,2]=0
else if (a[NR,2] > 1)
a[NR,2]=1

a[NR,3]=($4-mindur)/(maxdur-mindur)
```

```
if (a[NR,3] < 0)
a[NR,3]=0
else if (a[NR,3] > 1)
a[NR,3]=1

a[NR,4]=($5-minamp)/(maxamp-minamp)
if (a[NR,4] < 0)
a[NR,4]=0
else if (a[NR,4] > 1)
a[NR,4]=1

a[NR,5]=($6-minhit)/(maxhit-minhit)
if (a[NR,5] < 0)
a[NR,5]=0
else if (a[NR,5] > 1)
a[NR,5]=1

max=NR

}
END{

i=2
j=1
k=11
swap=2


while(k<42)
{
  while (j<6)
   {
     while (i<=k)
       {
         printf("%f\t",a[i,j]) > file
         i=i+1
       }
    i=swap
    j=j+1
    printf("\n") > file
   }
 j=1
 i=k+1
 k=k+10
 swap=i
 printf("\n") > file
```

```
}
}
```

Dataconvert.c

```
#include<stdio.h>
main()
{
 char s[80],ch,c;
 int i,fat;
 FILE *finp1,*fs,*finp2,*finp3,*finp4,*fbuff,*fbuff2;
 fs=fopen("inp1.txt","w+");
 fclose(fs);
 fs=fopen("inp2.txt","w+");
 fclose(fs);
 fs=fopen("inp3.txt","w+");
 fclose(fs);
 fs=fopen("inp4.txt","w+");
 fclose(fs);
 fs=fopen("buffer.txt","w+");
 fclose(fs);
 fs=fopen("buffer2.txt","w+");
 fclose(fs);
 fs=fopen("bend.txt","w+");
 fclose(fs);

 printf("Press b if Bending, Press t if Tension and Press f if Fatigue:");
 scanf("%c",&c);
 gets(s);

 if(c=='b'||c=='B')
 {
 printf("\nYour Input File should have following field values separated by tab
without any
headings:\nTime,Load,Location,Channel,Rise,Count,Energy,Duration,Amplitude,
AvgFrequency\nPlease Enter the Input file with    Extension:");
  gets(s);
  fs=fopen(s,"r");
  finp1=fopen("inp1.txt","a+");
  while(1)
 {
  ch=fgetc(fs);
  if(ch==EOF)
  break;
  else
  fputc(ch,finp1);
  }
```

```
 fclose(fs);
 fclose(finp1);
 system("awk -f bending.awk inp1.txt");
 fs=fopen("bend.txt","r");
 finp2=fopen("inp2.txt","w+");
  while(1)
 {
  ch=fgetc(fs);
  if(ch==EOF)
  break;
  else
  fputc(ch,finp2);
  }
 fclose(finp2);
 finp1=fopen("inp1.txt","w+");
 fclose(finp1);
 }
 else if(c=='t'||c=='T')
{
 printf("Your Input File should have following field values separated by tab without
any
headings:\nTime,Load,Channel,Rise,Count,Energy,Duration,Amplitude,AvgFrequ
ency\nPlease Enter the Input file with     Extension:");
 finp1=fopen("inp1.txt","a+");
 gets(s);
 fs=fopen(s,"r");
 finp2=fopen("inp2.txt","w+");
  while(1)
 {
  ch=fgetc(fs);
  if(ch==EOF)
  break;
  else
  fputc(ch,finp2);
  }
fclose(finp2);
}
 else
{

fat=1;
 printf("Your Input File should have following field values separated by tab without
any
headings:\nTime,Cycles,Channel,Rise,Count,Energy,Duration,Amplitude,AvgFre
quency\nPlease Enter the Input file with    Extension:");
 finp1=fopen("inp1.txt","a+");
```

```
 gets(s);
 fs=fopen(s,"r");
 finp2=fopen("inp2.txt","w+");
  while(1)
 {
  ch=fgetc(fs);
  if(ch==EOF)
  break;
  else
  fputc(ch,finp2);
  }
 fclose(finp2);
}


printf("\nIt is a good practice to attach a sensor on each loading grip to eliminate
noise from the loading grips");
printf("Did you use the noise sensors for your experiment:<Press Y if Yes>");
scanf("%c",&c);
gets(s);
if(c=='Y'||c=='y')
{
printf("Please Enter the Time(in seconds) an Acoustic Wave takes to Travel From
Noise Sensor to Nearest Material Sensor:");
 gets(s);
 fputs(s,finp1);
 fputs("\n",finp1);
 while(1)
 {
  ch=fgetc(fs);
  if(ch==EOF)
  break;
  else
  fputc(ch,finp1);
  }
 fclose(finp1);
 fclose(fs);
 system("awk -f signalnumbering.awk inp1.txt");
 fs=fopen("awk1.txt","r");
 finp2=fopen("inp2.txt","a+");
 printf("Please Enter First Noise Channel:");
 gets(s);
 fputs(s,finp2);
 fputs("\t",finp2);
 printf("Please Enter Second Noise Channel:");
 gets(s);
```

```
 fputs(s,finp2);
 fputs("\n",finp2);
 while(1)
 {
  ch=fgetc(fs);
  if(ch==EOF)
  break;
  else
  fputc(ch,finp2);
  }
 fclose(finp2);
 fclose(fs);
 system("awk -f eliminatenoise.awk inp2.txt");
 fs=fopen("awk2.txt","r");
 finp2=fopen("inp2.txt","w");
 while(1)
 {
  ch=fgetc(fs);
  if(ch==EOF)
  break;
  else
  fputc(ch,finp2);
  }
 fclose(fs);
 fclose(finp2);
 fs=fopen("inp2.txt","r");
}


 fbuff=fopen("buffer.txt","a+");
 fbuff2=fopen("buffer2.txt","a+");
 printf("*****************Load Calibration Parameters*******************\n");
 printf("If Calibration already done in LOCAN-AT your Multiplier Should be 1 and
Offset should be 0\n");
 printf("\n\tPlease Enter Multiplier:");
 gets(s);
 fputs(s,fbuff);
 fputs("\t",fbuff);
 printf("\n\tPlease Enter Offset:");
 gets(s);
 fputs(s,fbuff);
 fputs("\t",fbuff);
 printf("\n\nPlease Input the Threshold Value:");
 gets(s);
 fputs(s,fbuff);
 fputs("\t",fbuff);
```

```
if(fat==1)
printf("\n\nPlease Input the Failing Cycle Value:");
else
printf("\n\nPlease Input the Failure Load Value:");

 gets(s);

 fputs(s,fbuff);
 fputs("\t",fbuff);
 printf("\n\nPlease Enter the Watch Area Starting Amplitude:");
 gets(s);
 fputs(s,fbuff2);
 fputs("\t",fbuff2);
 printf("\n\nPlease Enter the Watch Area Lower Limit Freuency Value:");
 gets(s);
 fputs(s,fbuff2);
 fputs("\t",fbuff2);
 printf("\n\nPlease Enter the Watch Area Upper Limit Frequency Value:");
 gets(s);
 fputs(s,fbuff2);
 fputs("\t",fbuff2);
 printf("\n\nPlease Enter Number of Channels Which are Used in the
Experiment:");
 scanf("%d",&i);
 gets(s);
 fclose(fbuff);
 fclose(fbuff2);

while(i>=1)
 {
  fbuff=fopen("buffer.txt","r");
  fbuff2=fopen("buffer2.txt","r");
  finp3=fopen("inp3.txt","w+");
  fclose(fs);
  fs=fopen("inp2.txt","r");
  while(1)
  {
  ch=fgetc(fbuff);
  if(ch==EOF)
  break;
  else
  fputc(ch,finp3);
  }

  printf("\nEnter Channel Number:");
  gets(s);
```

```
 fputs(s,finp3);
 fputs("\n",finp3);

  while(1)
 {
 ch=fgetc(fs);
 if(ch==EOF)
 break;
 else
 fputc(ch,finp3);
 }

system("awk -f code1.awk inp3.txt");
 fclose(finp3);
 finp4=fopen("inp4.txt","w+");
 while(1)
 {
 ch=fgetc(fbuff2);
 if(ch==EOF)
 break;
 else
 fputc(ch,finp4);
 }
fclose(fs);
fs=fopen("awk3.txt","r");
 while(1)
 {
 ch=fgetc(fs);
 if(ch==EOF)
 break;
 else
 fputc(ch,finp4);
 }
 system("awk -f code2.awk inp4.txt");
 fclose(finp4);
 //take the input from user for the output file, do the normalization and make it
ready for Neural Network.
 system("awk -f mamin.awk awk4.txt");
 finp4=fopen("inp4.txt","w+");
 fclose(finp4);
 finp4=fopen("inp4.txt","a+");
 fclose(fs);
 fs=fopen("maxmins.txt","r");
 while(1)
 {
  ch=fgetc(fs);
```

```
  if(ch==EOF)
  break;
  else
  fputc(ch,finp4);
  }
 printf("\nEnter Output file to store Matrix from Channel %s :",s);
 gets(s);
 fputs("\t",finp4);
 fputs(s,finp4);
 fputs("\n",finp4);
 fclose(fs);
 fs=fopen("awk4.txt","r");
 while(1)
 {
 ch=fgetc(fs);
 if(ch==EOF)
 break;
 else
 fputc(ch,finp4);
  }

  system("awk -f matrix.awk inp4.txt");


 i=i-1;
 fclose(fbuff);
 fclose(fbuff2);
 }

 fclose(finp4);
 fclose(fs);
 printf("**********DONE DEAL!**********");
 getchar();
 exit(0);
}
```

Appendix B

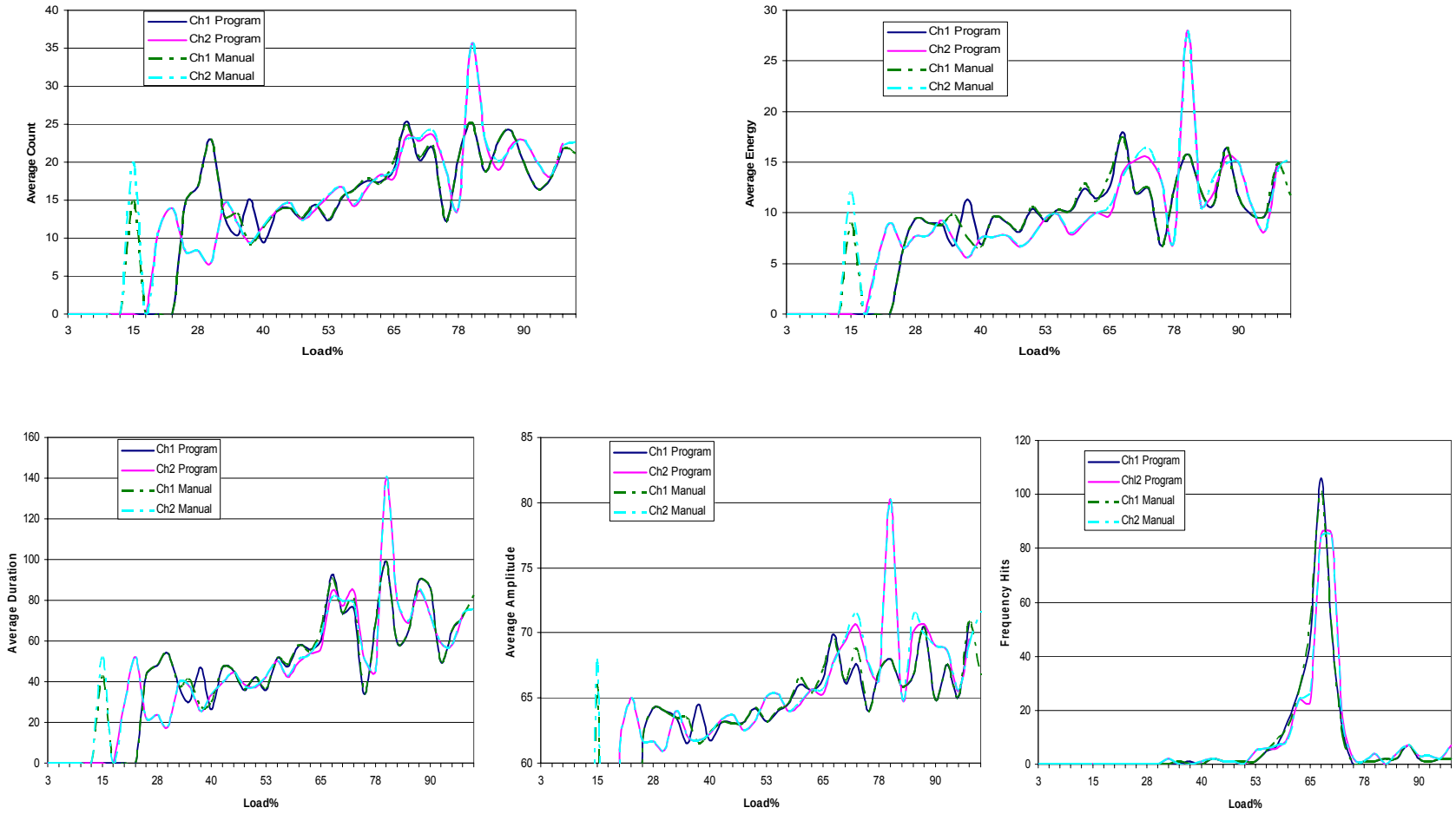


Figure A1 Test B3, Average AE parameters vs. Load%

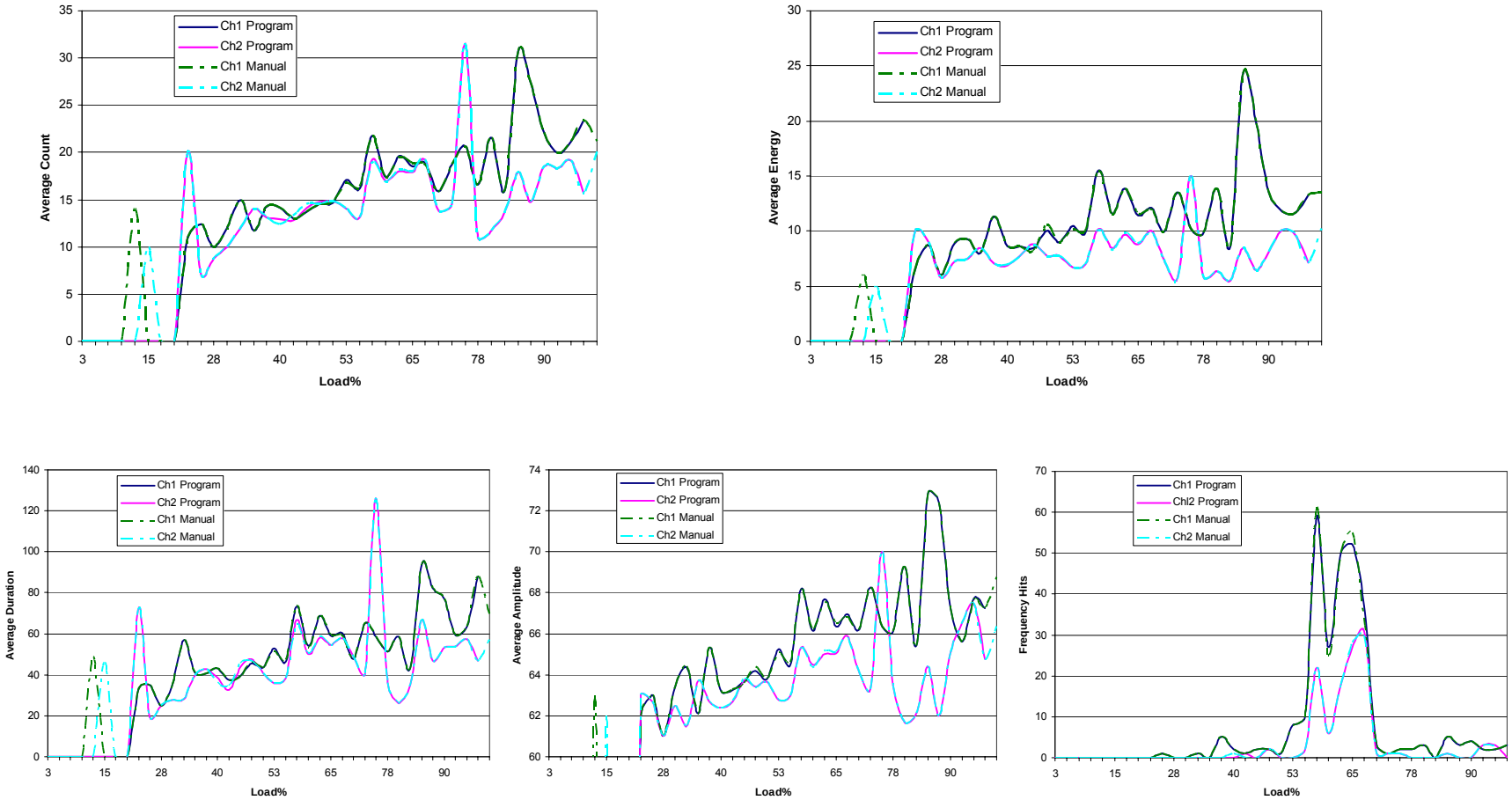


Figure A2 Test B4, Average AE parameters vs. Load%

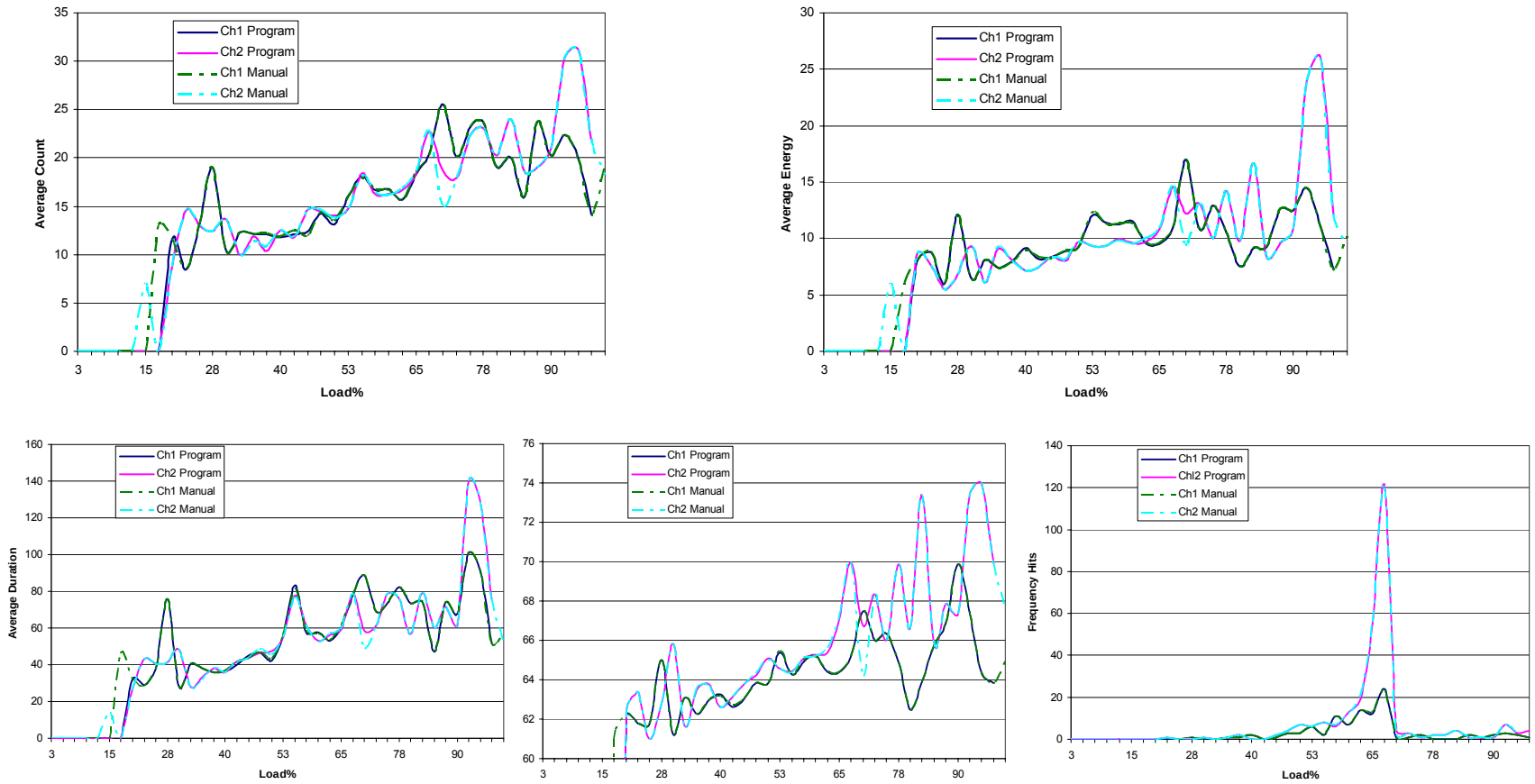


Figure A3 Test B5, Average AE parameters vs. Load%

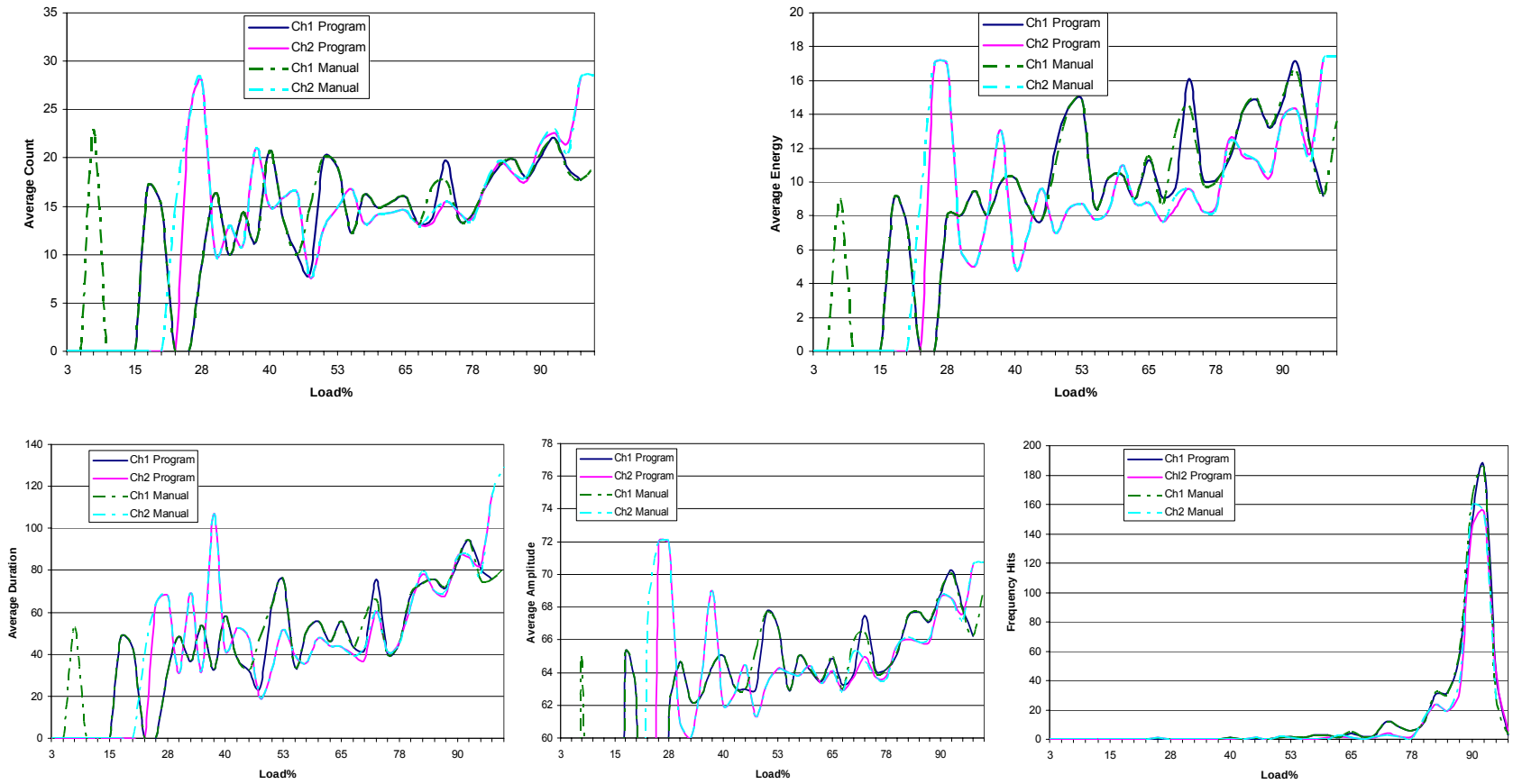


Figure A4 Test B6, Average AE parameters vs. Load%

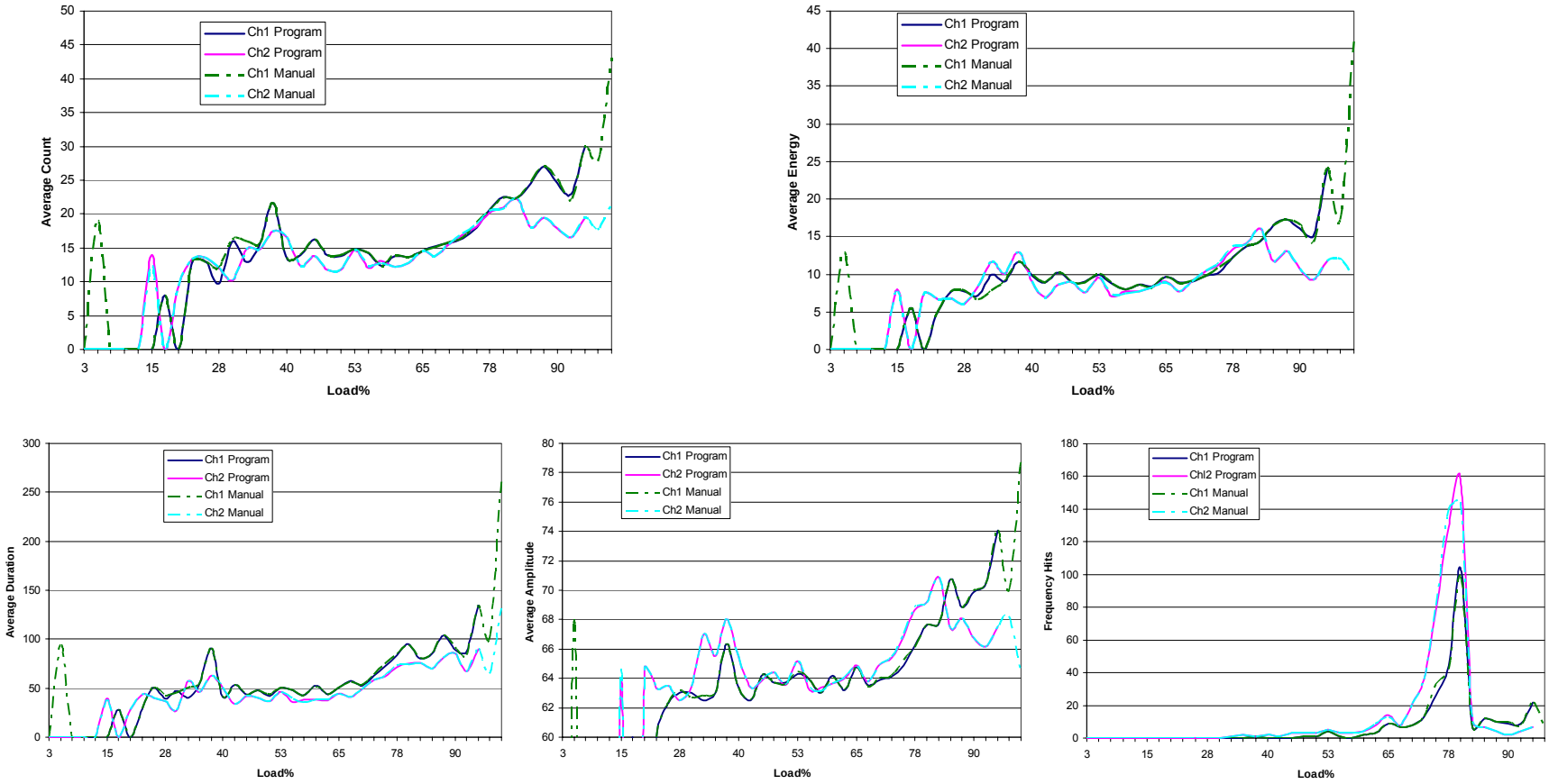


Figure A5 Test B7, Average AE parameters vs. Load%

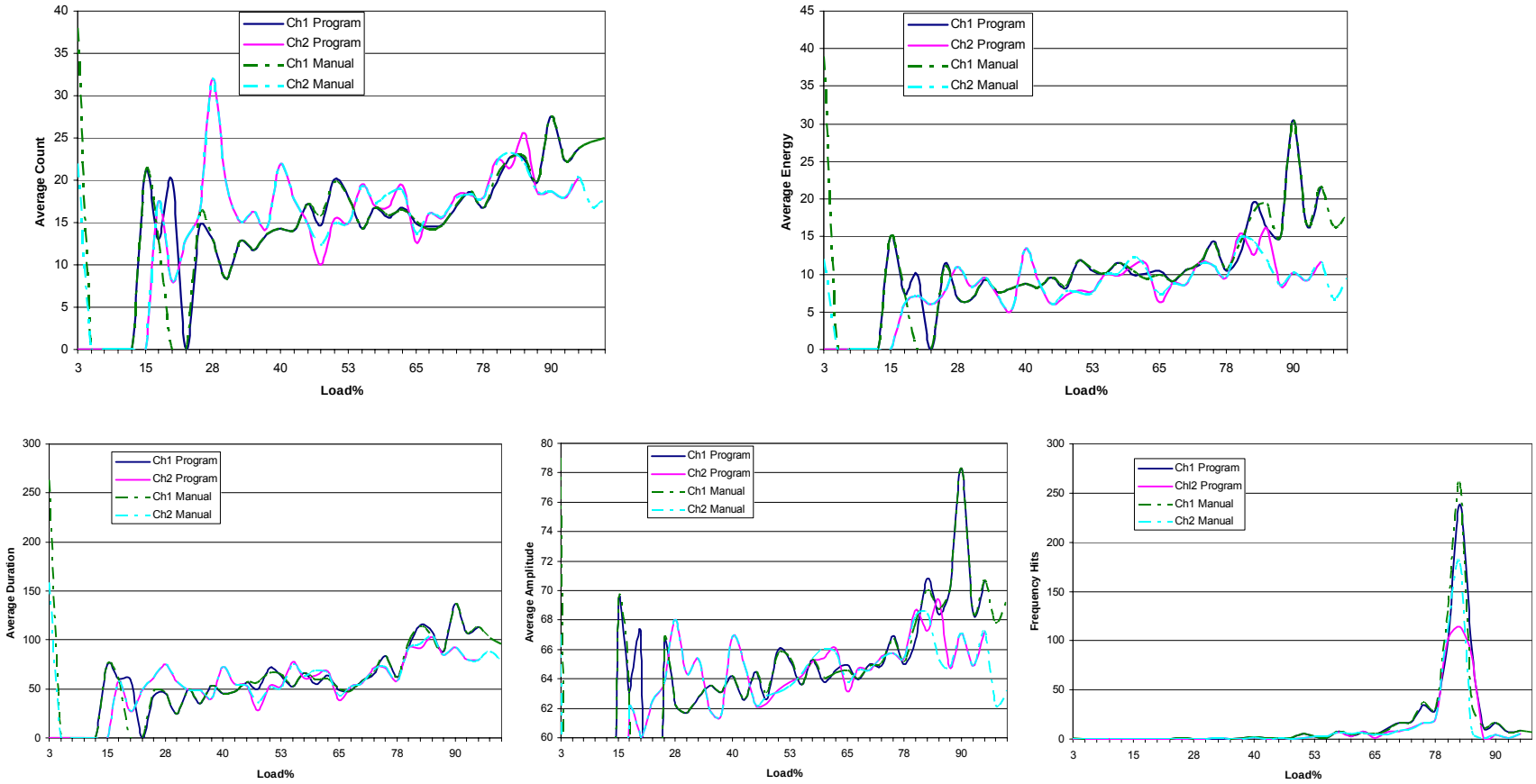


Figure A6 Test B8, Average AE parameters vs. Load%

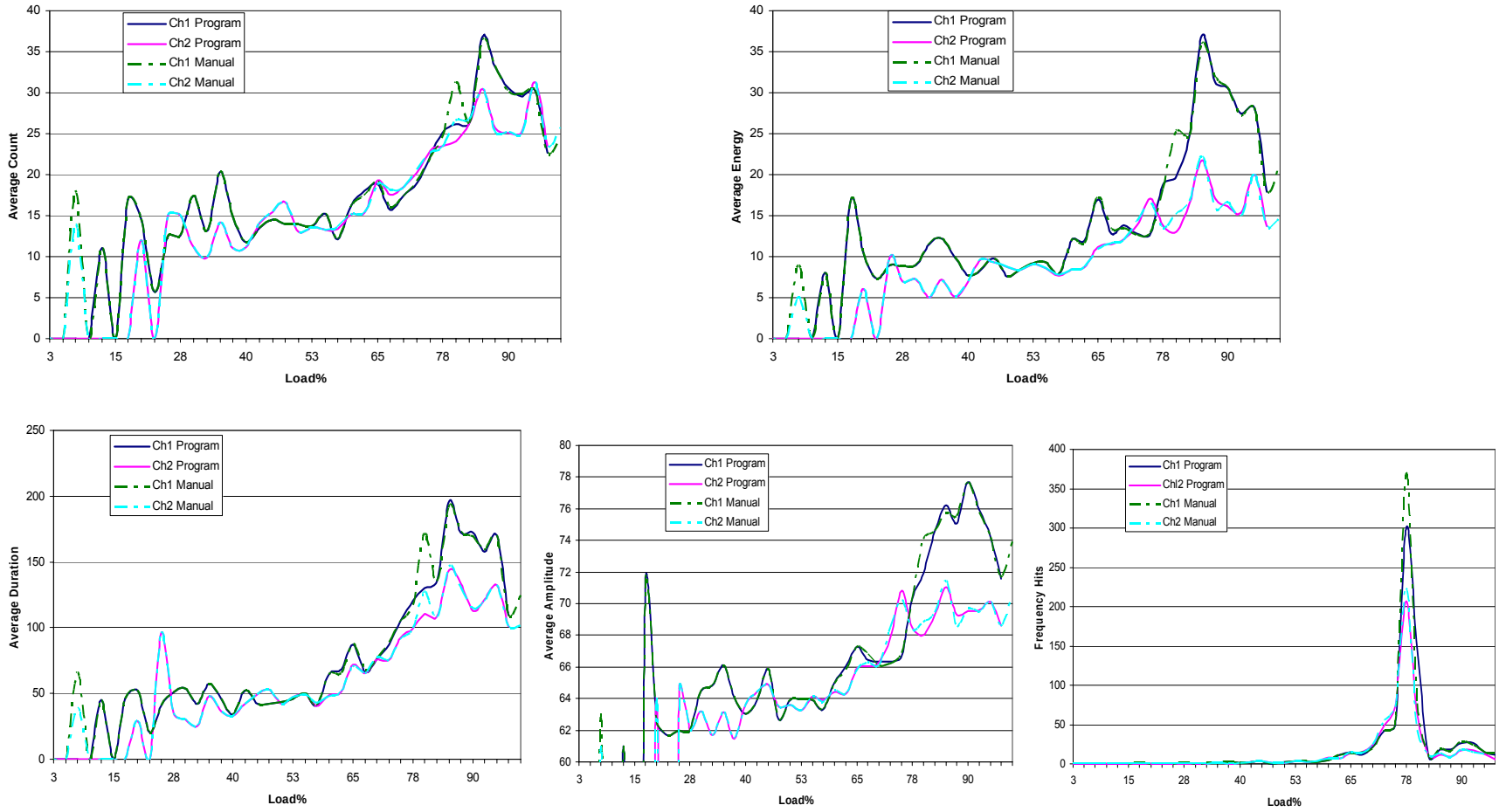


Figure A7 Test B9, Average AE parameters vs. Load%

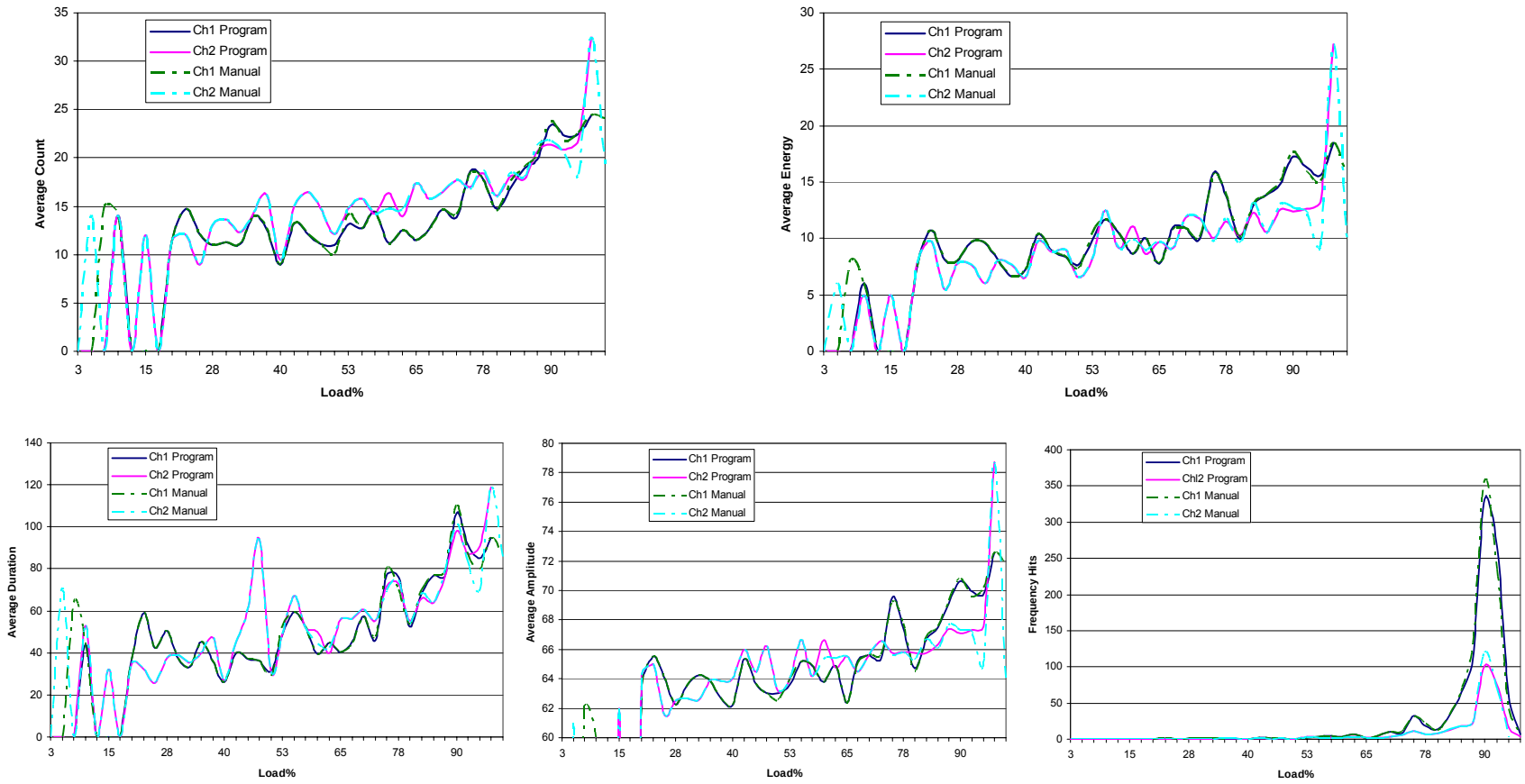


Figure A8 Test B10, Average AE parameters vs. Load%

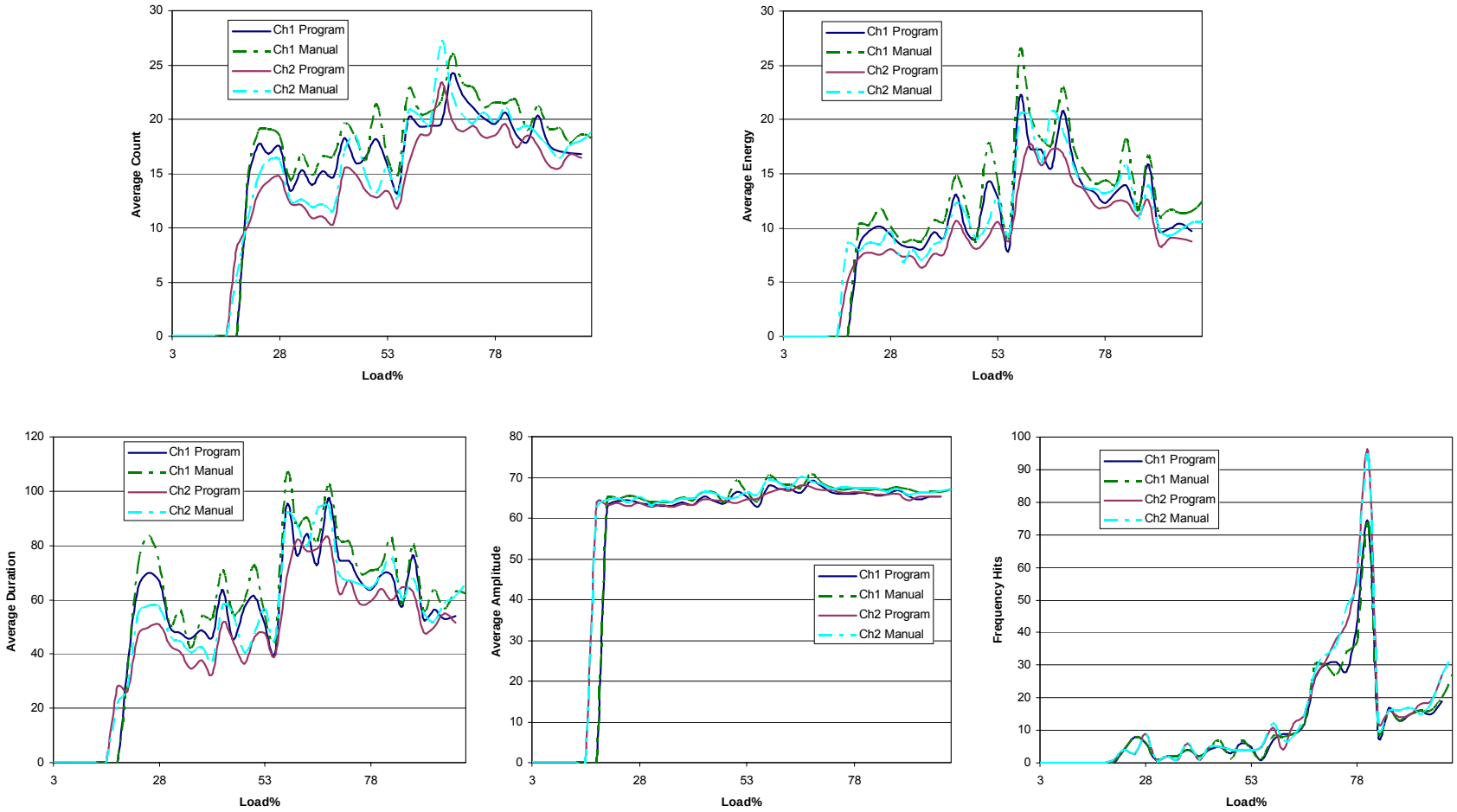


Figure A9 Test 2t, Average AE parameters vs. Load%

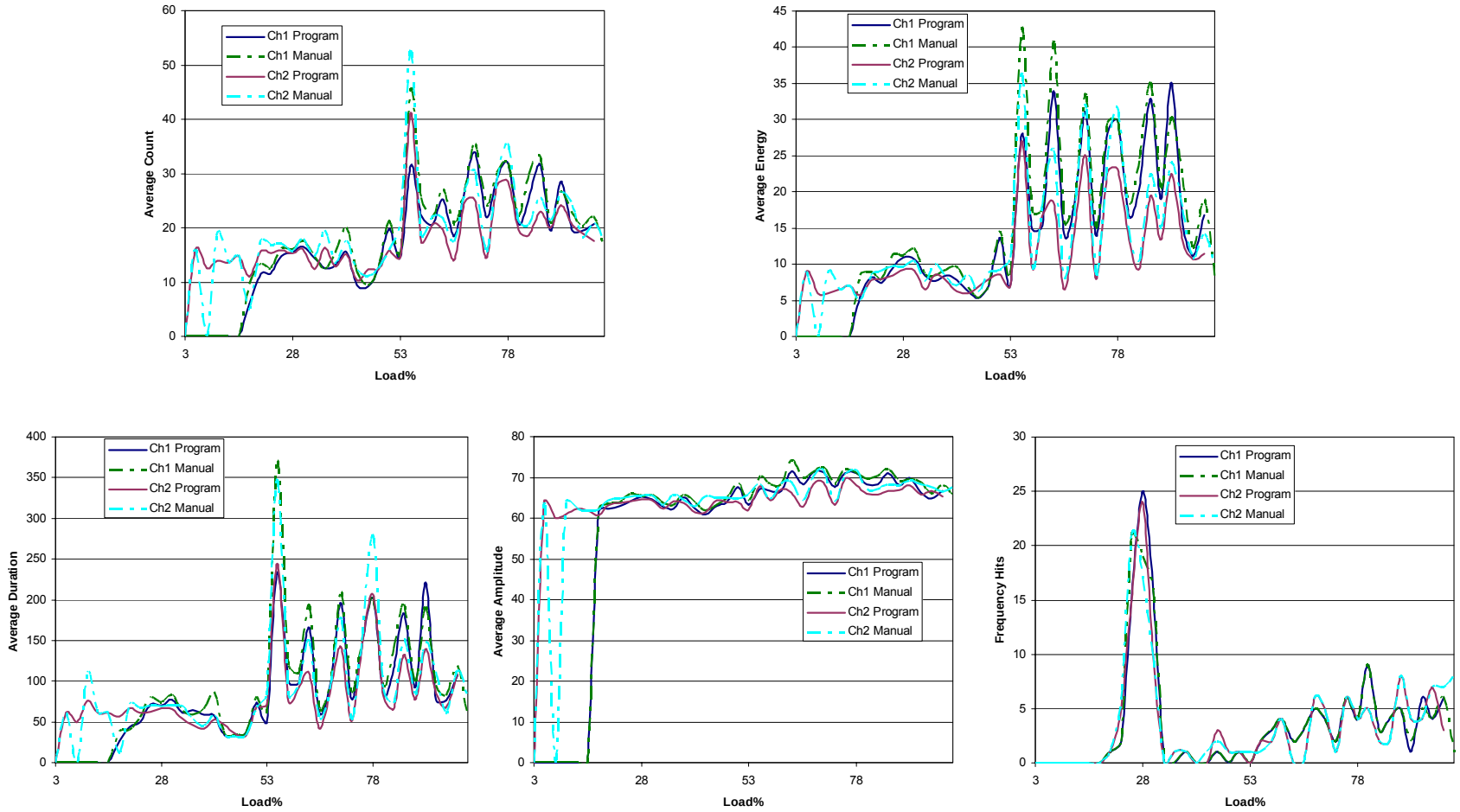


Figure A10 Test 4t, Average AE parameters vs. Load%

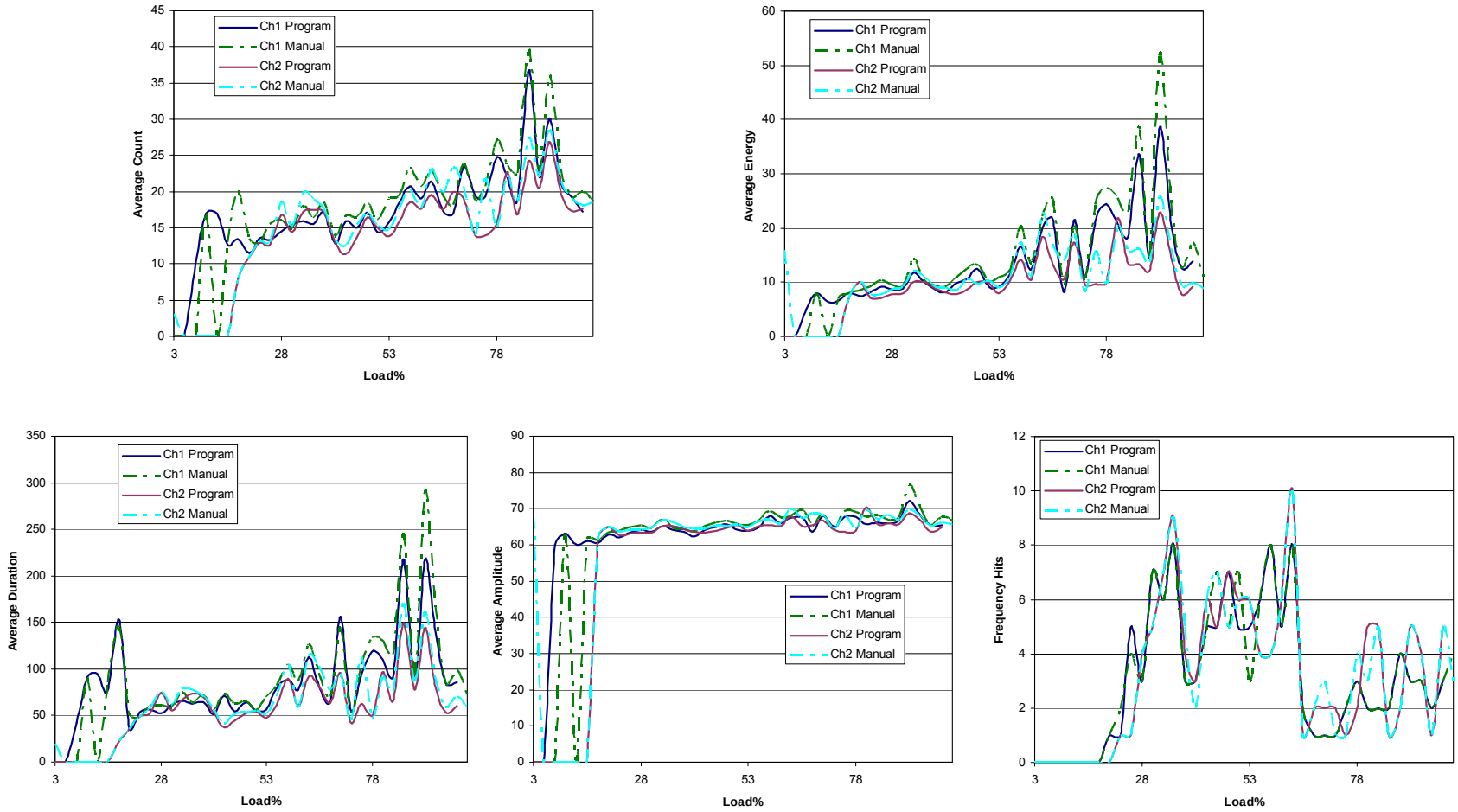


Figure A11 Test 5t, Average AE parameters vs. Load%

471 A, Inglewood Blvd
Morgantown,
WV-26505

AVINASH KAZA

avinash@mail.wvnet.edu
Office: 304-293-5192 X 229
Cell: 304-216-4927

SUMMARY

- More than 4 years of IT experience
- Extensively used **ColdFusionMX 6.0 and 6.1** to develop data driven secure websites providing multiple levels of access, visibility, monitoring, and control
- Used advanced SQL queries and PLSQL blocks to implement business logic
- Good working experience in administration of web-application and database servers
- Worked with database programs like **Oracle 8i/9i, SQL Server** and **MS Access**

Copyrights

- Obtained copyrights for the code developed for WVHEFIS project, REG: Txu1-150-050, Txu1-182-953

Presentations

- Presented at the SRAPPA 2004 conference held at New Orleans and received 2nd prize
- Presented the HEFIS system at the West Virginia Higher Education CFO/CPO (Chief Financial Officers / Chief Purchase Officers) Spring 2004 conference
- Presented the HEFIS system to West Virginia Higher Education Policy Commission chancellor J.Michael Mullen, as part of yearly mile stone meeting of the project

EDUCATION

Diploma In Advanced Software Technology **June 2002**
CMC, A government of India enterprise with 4.0 cumulative GPA

BS in Mechanical Engineering **June 2002**
MVSR Engineering College, India with 3.6 cumulative GPA

Master of Science In Engineering **April 2005**
West Virginia University, Morgantown with 3.66 cumulative GPA, 3.88 major GPA

TECHNICAL SKILLS

Web Designing	:	**HTML, CFML, MS FrontPage, DreamWeaver MX, Flash MX, Fireworks**
Web Application Servers	:	**ColdFusion MX 6.0/6.1, J2EE, .NET Framework**
Databases	:	**Oracle 8i/9i, MS Access, SQL Server**
Packages	:	**Visual Basic 6.0, Visual C++ 6.0, Crystal Reports 9, SmartDraw, Adobe Photoshop 6.0, Adobe ImageReady 3.0, Adobe Illustrator 10**
Languages	:	**C, C++, VB 6.0, VB.NET, JAVA, UNIX shell programming, SQL, PLSQL, SED and AWK (UNIX power utility tools)**
Operating Systems	:	**Windows 9x/NT/2000/XP, MAC, Redhat Linux 6.0 and UNIX**

WORK EXPERIENCE

WVNET - West Virginia Network **April 2005 to Present**
Working as Senior Database Administrator for the WVHEFIS project funded by the HEPC chancellor's office, West Virginia

West Virginia University Facilities and Services April 2003 to April 2005
Worked as System Designer for *West Virginia Higher Education Facilities Information System (WVHEFIS)* project funded by HEPC chancellor's office, West Virginia
Description:
Design of an efficient database, which can hold the facilities information (Buildings & Rooms & Vehicles & Housing) and course schedule information of all higher education institutions in the state of West Virginia. Development of a website with the ability to search, make real time alterations and output reports at institutional and HEPC levels.
Responsibilities:

- Installed and Configured ColdFusion MX 6.0 on Windows 2000 server and later upgraded ColdFusion to 6.1
- Designed an MS Access database with user-friendly forms (developed in VBA) for data population, to be used by the data acquisition team
- Programmed in different access levels to data based up on username and password
- Dynamic data querying was made possible by creating search forms for facilities information

- Developed ability to create dynamic Excel, PDF or CSV files
- Course scheduling information was obtained from the institutions in different formats as text or excel files, which were then translated into a standard format by using AWK scripts
- The course data was interlinked with facilities information to get a wide variety of very useful utilization reports and graphs at different levels
- After data collection was accomplished for all the institutions, merged the individual MS Access databases into a single Oracle9i database, which was configured on another windows 2000 machine
- Added ability to do all database maintenance tasks like Add, Update and Delete records through the website
- All changes done to the database are tracked by storing the modification and user info in a separate table
- Added Tools module, which helps in facilities planning by calculating Sherman-Dergis and Formula Budgeting values for each higher education facility in the state of West Virginia
- Separated community college component at wvu-tech institute for charge backs, using enrollment breakdown by college
- Vehicles and housing information for all institutions in West Virginia were added to the database and all data maintenance tasks were made web based along with some reports requested by HEPC

URL of website: http://fishepc.wvu.edu/
Tools Used:
ColdFusion MX 6.0 (later upgraded to 6.1), Windows 2000 Server, DreamWeaver MX, MS Access 2002, AWK, Oracle 9i, Oracle SQL *Plus, Oracle SQL *Loader, Oracle EXP, Oracle IMP and other Oracle tools like RMAN, Adobe Photoshop 6.0, Adobe ImageReady 3.0, SmartDraw, Adobe Illustrator 10, Oracle 9i, Oracle SQL *Plus, Oracle SQL *Loader, Oracle EXP, Oracle IMP and other Oracle tools like RMAN etc

West Virginia University Facilities and Services August 2004 to present
Working as Facilities Information System's *TMA and ISD* Database and Web Administrator
Description:
This project used TMA Systems, to track and maintain work orders for WVU online
Responsibilities:
- Worked on setting up TMA clients and TMA databases on windows machines
- Installed I Service Desk and administered it foe WVU
- Added code to I Service Desk pages to let ISD users to go into wvhefis site and see the schedules, Drawings or reports
- Enabled ISD users to add, edit or delete buildings and rooms in WVHEFIS database
- Migrated two TMA databases from SQL Server to Oracle9i using TMA's e-tools and Oracle Migration Workbench
- Migrated one Database from Oracle9i to SQL Server

Tools Used:
Oracle Migration Workbench, Windows 2000 Server, Dream Weaver MX, SQL Server 2000, Oracle Exp, Oracle Imp utilities

West Virginia University Facilities and Services April 2003 to present
Working as Facilities Information System's *Key Control Management System (FISKCMS)* Developer
Description:
This project aimed at facilitating the building supervisors in WVU to access the keys database through a website developed using ColdFusion MX.
Responsibilities:
- Modified existing MS Access keys databases, for each building, making them simple and consistent throughout
- Designed a website allowing users to login and access their keys information
- Users with appropriate rights can assign or un-assign keys based on room, key number or key holder
- Created a logs table that records the MAC Address, IP Address of the system, username and password used for login, in addition to the changes made to the database thereby giving administrators the ability to track back any errors committed by users
- Configured SMTP server and used for signup of new users

URL of website: http://fiskcms.wvu.edu
Tools Used:
ColdFusion MX 6.0/6.1, Windows 2000 Server, DreamWeaver MX, MS Access 2002

West Virginia University Plant and Soil Sciences November 2002 to March 2003

Worked as Web Database Designer and Administrator for *International Culture Collection of (Vesicular) Arbuscular Mycorrhizal (INVAM)* Fungi funded by National Science Foundation

Description:

This project was initiated to let users, across the world, search for different cultures and fungi that are grown in INVAM laboratory. I developed a powerful web based, password protected user interface for the INVAM staff to maintain the oracle database.

- Installed and configured Coldfusion MX web server and Oracle 9i database server
- Corrected the erroneous data in MS Access database, which was populated using memo data types having return chars inside, by writing a C program
- Migrated MS Access Database to Oracle 9i
- Developed WebPages to search and retrieve records from the database
  http://invam.caf.wvu.edu/cultures/accessions.htm
  http://invam.caf.wvu.edu/cultures/CultSearch.htm
- Developed a Search interface to search the website
  http://invam.caf.wvu.edu/invam_search_form.cfm
- Developed feedback forms to receive feedback from users
  http://invam.caf.wvu.edu/feedback.cfm
- Developed password protected dynamic data reports and web pages to insert, update or delete records from database for INVAM staff

Tools Used:

ColdFusion MX 6.0, Windows NT Server, DreamWeaver MX, MS Access 2000, C Programming, Oracle 9i, Oracle SQL *Plus, Oracle SQL *Loader

West Virginia University Plant and Soil Sciences September 2002 to October 2002

Worked as Data Programmer for *the role of plant diversity in disease incidents in organic tomato production*

Description:

Collected weather data for different regions of interest from a government website and translated it to disease severity values for use in a blitecast model.

- Weather data was first downloaded from http://www.ncdc.noaa.gov
- Used already collected data from different farmers and used combination of severity models to program the data to give the disease severity values for use in development of blitecast model

Tools Used:

MS Access 2000, VBA, MS Excel

M.V.S.R Engineering College, INDIA December 2001 to May 2002

Developed a Computer Interface and Control of a Mechanical Device, in partial fulfillment for requirements of the degree of engineering, funded by Matrusri Education Society

Description:

A physical parameter (*Temperature*) is sensed into a computer and a device (*pump*) is controlled according to the value of the parameter. To do this a GUI application was developed using Visual Basic 6.0. The software shows the temperature sensed for every half a second with a resolution of ***0.195^{o}C*** and switches on/off the pump according to the conditions given by the user.

- Hardware interface circuits were designed to read analog temperature value from a thermocouple and digitize the data in order to be read by the software application.
- A transistorized switching circuit was designed to take input from the application and switch on or switch off a cooling pump of a concentric tube counter-flow heat exchanger.
- The digital input and output between the application and circuit is made possible by programming an 8255 Programmable Peripheral Interface (*PPI*) card.
- To program the 8255 PPI card wrote a DLL (Dynamic Linked Library) file to use the port access functions available in VC++.

Tools Used:

Visual Basic 6.0, Visual C++ 6.0

CMC Limited, Hyderabad, INDIA June 2000 to December 2000

Developed a complete education system package in partial fulfillment for requirements of the diploma in advanced software technology

Description:

An application was developed using Visual Basic 6.0 as front end and Oracle 8i as back end to computerize all the important processes in a general education institution

- A dummy database was created with student data, course data and faculty data.
- The student grades can be fed into the system using forms and progress reports are automatically generated. Ability is given to the faculty members to write comments on student's progress reports.
- The system is made capable of maintaining fee transaction records from students and giving reports to users about dues.
- Graphs are made available to be printed out for each student showing the progress he made since his admission.

Tools Used:

Visual Basic 6.0, Oracle 8i

Thesis Related Works

- Developed seven independent **AWK** scripts for converting experimentally collected raw data into an input matrix to a neural network program. To relieve future users from possessing the knowledge of syntax to call AWK scripts, developed a **C program** that takes experimental inputs from user, calls relevant AWK scripts, passes variables to them and saves the output matrix by a name specified by user
- Worked with **Wireless Sensor Networks** using MICA MOTES hardware and TINYOS operating system and applications like TINYDB and TASK