To the Graduate Council:

I am submitting herewith a thesis written by Jeremy J Kovach entitled "Prediction of Impending Mechanical Seal Failure in Centrifugal Pumps." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Mechanical Engineering.

Masood Parang, Major Professor

We have read this thesis

and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor and

Dean of The Graduate School

PREDICTION OF IMPENDING MECHANICAL SEAL FAILURE IN CENTRIFUGAL PUMPS

A Thesis Presented for the Master of Science Degree

The University of Tennessee, Knoxville

Jeremy J Kovach

December 1999

ACKNOWLEDGMENTS

I am very appreciative to many people in the Mechanical Engineering Department for making my educational experience at the University of Tennessee so rewarding. I am particularly grateful to Dr. Masood Parang, my major professor, for his instruction, encouragement, and support for this research. I would also like to thank Dr. Stan Johnson and Dr. Wes Hines, both of whom served on my graduate committee as well as taught me many of the engineering skills that I possess today. I would also like to thank the Maintenance and Reliability center for sponsoring this research and introducing me to the many fine people whom have contributed to it. This research could not have been completed without the help of Dennis Higdon from the Mechanical Engineering Electronic Shop and Mark Lambert, a maintenance and reliability engineer from Tennessee Eastman Division. Mostly, I would like to thank my parents, without whom I would never have achieved this goal.

ABSTRACT

Mechanical seals on centrifugal pumps provide the junction between the rotating shaft and stationary housing on a pump assembly which prevents unnecessary fluid leakage. When these seals fail, the pumps no longer run at peak performance, and sometimes will not run at all. Since mechanical seals fail approximately 85% of the time as opposed to wear out, the early detection of their failure could assist in preventing system down time and possibly preventing the seal failure. The seals in question were instrumented to determine various operating conditions during different phases of the seal's performance spectrum. By training artificial intelligence computer codes to classify the relative performance of specific seals based on the information obtained by instrumentation, a prediction technique was modeled. These artificial intelligence codes were programmed using the general judgment criteria found from a literature survey as well as discussions with plant engineers directly associated with the failing seals. These codes are dependent on a variety of pump performance characteristics. These models for predictive analysis along with defined goals for increased data collection practices not only prove the feasibility of using artificial intelligence codes to help predict mechanical seal failure, but also lay the foundation for an expanded and robust failure analysis study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

AI	Artificial Intelligence
ANN	Artificial Neural Network
ANSI	American National Standards Institute
DOF	Degree of Fulfillment
MLP	Multi-Layer Perceptron Neural Network
MRC	Maintenance and Reliability Center
MSE	Mean Squared Error
MTBF	Mean Time Between Failure
NPSH	Net Positive Suction Head
PNN	Probabilistic Neural Network
PSD	Power Spectral Density
RBFN	Radial Basis Function Neural Network
RMS	Root Mean Squared
SSE	Sum of Squared Error
TED	Tennessee Eastman Division
LVQ	Learning Vector Quantization
<u>Units</u>	
[C]	degrees Celsius
[g]	acceleration of gravity
[Hz]	Hertz (1/second)
[μF]	micro-Farads
[psig]	pounds per square inch – gage
RPM	Revolutions Per Minute
[V]	volts

CHAPTER I

INTRODUCTION

Background

Centrifugal pumps provide an irreplaceable service in fluid transportation. Since the introduction of the centrifugal pump in 1846, they have become the most prevalently used type of pump in industry[1]. Centrifugal pumps employ centrifugal force in order to add energy to a fluid and thereby move it from one location to another. This process involves liquid being sucked into a chamber by an impeller that is being rotated on a shaft. This fluid is then accelerated by the vanes on the impeller and thrown out to the discharge by means of centrifugal force[2]. A simple diagram showing this process is shown below in Figure 1.

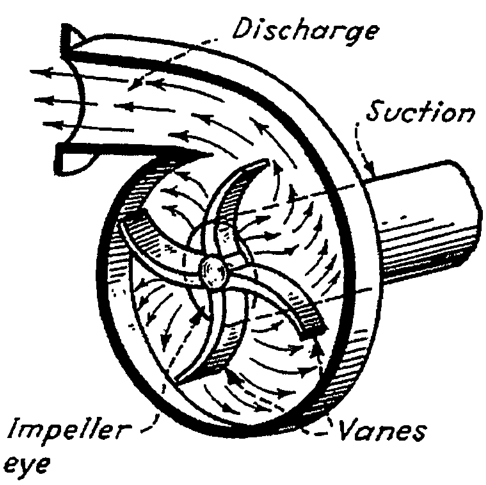


Figure 1 - Centrifugal Pump Impeller[2]

This process is dependent on the interaction between the rotating shaft and the stationary housing of the pump. In order to secure the junction between the rotating and stationary components of the centrifugal pump so that excess fluid is not lost, mechanical seals were developed. A schematic of all of a centrifugal pump's major components is shown below in Figure 2.

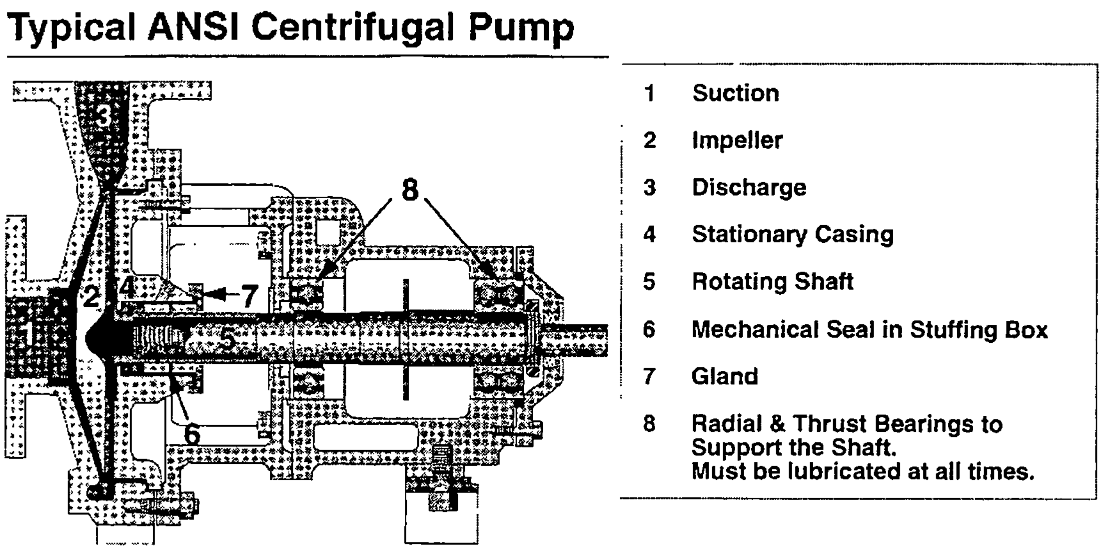


Figure 2 - Pump Schematic[3]

Mechanical seals are devices used to prevent fluid leakage by means of two sealing surfaces, one stationary and the other rotating. These two surfaces are compressed together by mechanical springs and prevent the excessive flow of fluid between them[4]. This seal is dependent on the presence of a thin layer of fluid to help lubricate this interaction. This thin layer of lubricating fluid is supplied by the pumped fluid and should never be so excessive as to be seen on the exterior of the seal. This fluid is circulated in a region called the stuffing box, where a small amount of the fluid acts as a lubricator at the seal junction. Whenever this fluid leaks, it is a direct indication that the mechanical seal has been compromised and has failed.

Problem Statement

As stated before, mechanical seals provide the junction between the rotating shaft and the stationary housing of centrifugal pumps. The dynamics of this form of interaction imply a constant process of degradation for mechanical seals. In fact, mechanical seals are not expected to have a very large life span, but the excessive failure of these seals has necessitated a study regarding their reliability and maintenance.

Seal failure is a significant problem for large chemical plants such as Tennessee Eastman Division (TED) of Eastman Chemical Company in Kingsport, TN. TED utilizes many thousands of pumps throughout their Kingsport facility. The most common pump used in this facility is the Durco Mark III Group II with a 1-7/8" shaft diameter. These are ANSI standard process pumps and number approximately 10,000 in the TED facility. Of these pumps, approximately 150 fail each month due to mechanical seal related issues, and the cost for replacement parts alone for the mechanical seals could reach $50,000 per month.

On average, mechanical seals fail approximately 85% of the time, as compared to the 15% that complete their life cycle and wear out[5]. Although most of these mechanical seals are designed to yield a ten-year life cycle, they only average lasting between five and seven years. This high rate of failure is compounded by the fact that some mechanical seals that are utilized with highly caustic fluids fail within several months of operation[5].

Research Motivation

The motivations in studying mechanical seal failure are primarily environmental and economical in nature. From a purely physical observation, whenever mechanical seals fail, the fluid that was being pumped leaks out of the pump into the environment. Depending on what fluid is being pumped, this could be a potential health or environmental risk. Although no mechanical seal will last forever, increasing their life span or knowing exactly when a seal will fail would undoubtedly decrease this potential hazard.

Aside from the potential environmental risks associated with seal failure, predicting mechanical seal failure or increasing mechanical seal life is economically beneficial as well. Despite the actual costs to replace failed mechanical seals; the costs associated with the process downtime could be reduced. In many instances where a mechanical seal fails, the entire fluid process has to be stopped while the pump or mechanical seal is exchanged for one that has not failed. This process downtime presents obvious and undesirable side effects.

Due to these problems associated with mechanical seal failure, it would be extremely beneficial to develop a strategy that would help determine a rational basis for the maintenance and replacement of mechanical seals and centrifugal pumps. This would involve moving from a reactive to a proactive maintenance schedule. Such a fundamental maintenance philosophy change would involve the replacement of mechanical seals prior to their failure at a convenient time of a scheduled process downtime as opposed to replacing the mechanical seals at a potentially undesirable time.

Research Objectives

The purpose of this research was two-fold. The first portion of this research involves providing a feasibility study for failure prediction analysis. This feasibility study lays the groundwork for future studies on mechanical seal failure prediction. By determining if specific physical parameters that correlate to varying seal conditions can be selected and unobtrusively measured, a framework for seal failure prediction can be established and implemented. This feasibility study would therefore determine if a failure prediction system based on a comprehensive data collection process could be created and effectively implemented on existing centrifugal pumps.

The second portion of this research involves developing a preliminary artificial intelligence code to facilitate this prediction analysis. Although no artificial intelligence code has been designed to predict premature mechanical seal failure in centrifugal pumps, artificial intelligence codes have been designed to perform many similar functions regarding prediction and classification. Such codes have been designed to classify anomalies in cooling systems, determine power system's reliability, classify aerosol particles, predict welding parameters, detect abnormalities in nuclear power plants, estimate tool wear, and classify image textures[6-12]. One of the purposes of this research is to design preliminary artificial intelligence codes to predict premature mechanical seal failure. These preliminary codes are based on the limited physical data that could be collected from a group of similarly functioning pumps at the TED facility. By developing several different artificial intelligence codes, the most effective and reliable code can be used as a model for future work. By developing a preliminary prediction analysis computer code, various physical aspects regarding the mechanical seal's performance can be evaluated as to their correlation to mechanical seal failure.

CHAPTER II

METHODOLOGY

The first step in this research program involved classifying the general causes for mechanical seal failure. This was accomplished through a literature survey as well as discussions with TED plant engineers and technicians who deal with mechanical seals and centrifugal pumps on a regular basis. This was done in an effort to gain a general understanding of the functioning and physical limitations and characteristics of mechanical seals.

Secondly, a data acquisition system was developed and assembled in a laboratory at the University of Tennessee. This experimental data acquisition system was designed from the information gained in the literature survey as well as discussions with plant engineers. This data acquisition system was developed and tested in the laboratory to assure proper functioning.

Thirdly, an ongoing data collection program was initiated on a group of similarly operating pumps at the TED facility. This data collection process involved recording the inlet and stuffing box fluid temperatures of the pump, as well as the vibration of the mechanical seal in two planes of motion. This was done to help determine if any of these parameters could be correlated to the performance of a mechanical seal. This data collection program was incorporated on ten pumps over a period of two months.

Fourthly, the information obtained from the data collection process was used to design several different artificial intelligence architectures. These AI architectures, although all different, were all designed to recognize premature mechanical seal failure. These different AI architectures include a radial basis function neural network, a multi-layer perceptron neural network, a probabilistic neural network, and a fuzzy logic classification system. From these different AI architectures, the most ideal method of prediction analysis can be determined.

Lastly, the strategy for a long term, laboratory controlled, test flow loop was conceived. By establishing a flow loop for a centrifugal pump in a laboratory environment, the effects on a mechanical seal relating to controlled and induced faults on a centrifugal pump can be determined. The conjunction of data collected from pumps in operation at TED with the information obtained in a laboratory environment will prove to yield an extremely powerful method for the prediction of premature mechanical seal failure.

CHAPTER III

SEAL FAILURE ANALYSIS

Causes for Seal Failure

Since this research involved the initiation of a new application regarding predictive maintenance analysis, the review and understanding of other similar studies were imperative. These studies included textbooks, journal articles, and technical manuals regarding centrifugal pumps and mechanical seals as well as a close cooperation with the reliability engineers and pump operators at the TED facility in Kingsport, TN. From these sources, a general understanding of mechanical seals was obtained, including their most common reasons for premature failure.

It quickly became obvious that an exact and irrefutable explanation for premature mechanical seal failure did not exist. Many people believe that these failures are based solely on pump operator error. Many others blame the mechanical seal design as the key source for failure. A significant number though believe that improper installation techniques lead to premature failure. Despite these varying viewpoints though, general circumstances that are believed to decrease the life of a mechanical seal can be summarized. These general causes can be divided into several categories.

The first such category involves the design or operational problems regarding the pump operation. These causes include an insufficient NPSH, too high of a suction pressure, the dry running of the pump (run with no fluid), running the pump off the design point, and pumping a fluid that has a higher than specified specific gravity and/or viscosity.

The next category regarding the causes of seal failure involves assembly problems. These causes include an excessive shaft misalignment (the rotating shaft is not aligned within 0.002" to its driver in all four planes), the bearings were incorrectly installed, the impeller is out of balance, there is an excessive shaft deflection, the seal was installed incorrectly, the couplings are out of balance, and the pump was assembled incorrectly[13].

Similar to assembly problems are the problems regarding the pump and piping placement. These causes for failure include: the baseplate (a rigid platform that holds the motor, shaft, and pump housing together) not installed properly, the piping not properly anchored, and the pump and/or driver not secured to the baseplate.

Another category involves failed components of the pump. These causes include the wearing of the shaft, the failure of the bearings, and the breaking of the impeller.

The last category involves physical problems with the mechanical seals. These causes for failures include the overheating of the seal faces and the lack of seal flush at the seal faces[14, 15].

As can be seen from these five categories, the agreed causes for premature mechanical seal failure can be attributed to at least 21 factors outlined here. Fortunately for the case at hand, the employees at the TED facility have taken considerable care to alleviate many of these causes for failure. Due to their reliability engineering practices, the previous list of possible causes for failure can be effectively reduced to seven circumstances. This refined list of possible causes for premature mechanical seal failure include: an excessive shaft misalignment, the overheating of seal faces, an excessive shaft deflection, the wearing of the shaft, the improper installation of the baseplate (although all of the pumps used in this subsequent study were installed properly), the failure of the bearings, and the poor anchoring of the piping. From this information survey, a general methodology for achieving the desired objectives of this research was developed.

Inspection of Mechanical Seals

The science of determining hazardous conditons for mechanical seals has been further compounded with visual correlations. Researchers have determined certain correlations between improper pump operating conditions and the wear patterns on the actual mechanical seal faces. Although these circumstances can only be determined after the mechanical seal has failed, this process of visual inspection does provide insightful information relating to mechanical seal failure. To help understand and explain this process of inspecting the mechanical seals to determine their motivations for failure, a more thorough understanding of a mechanical seal is required. A picture of a standard pusher-type cartridge seal is shown below in Figure 3, with its important features noted.

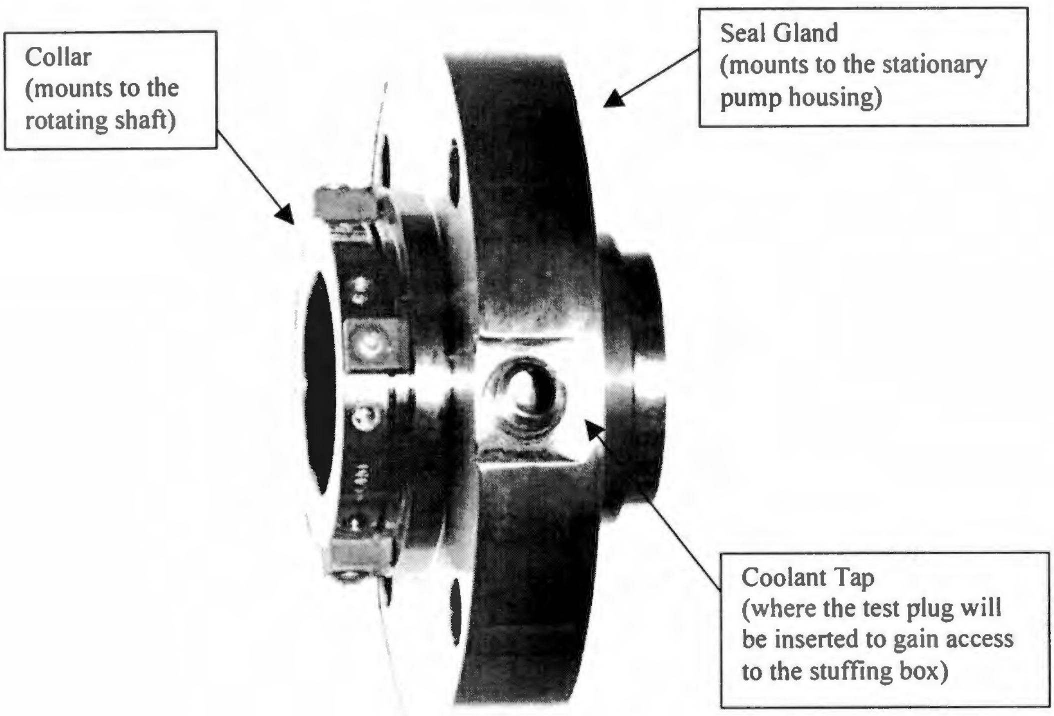


Figure 3 - Cartridge Seal

This type of cartridge seal is indicative of the type used at the TED facility. When this cartridge seal is taken apart, the interface between the moving and stationary seals can be seen. It is these seal faces that are used for the inspection process to determine why specific mechanical seals have failed. These seal faces and their interface can be seen more clearly below in Figure 4.

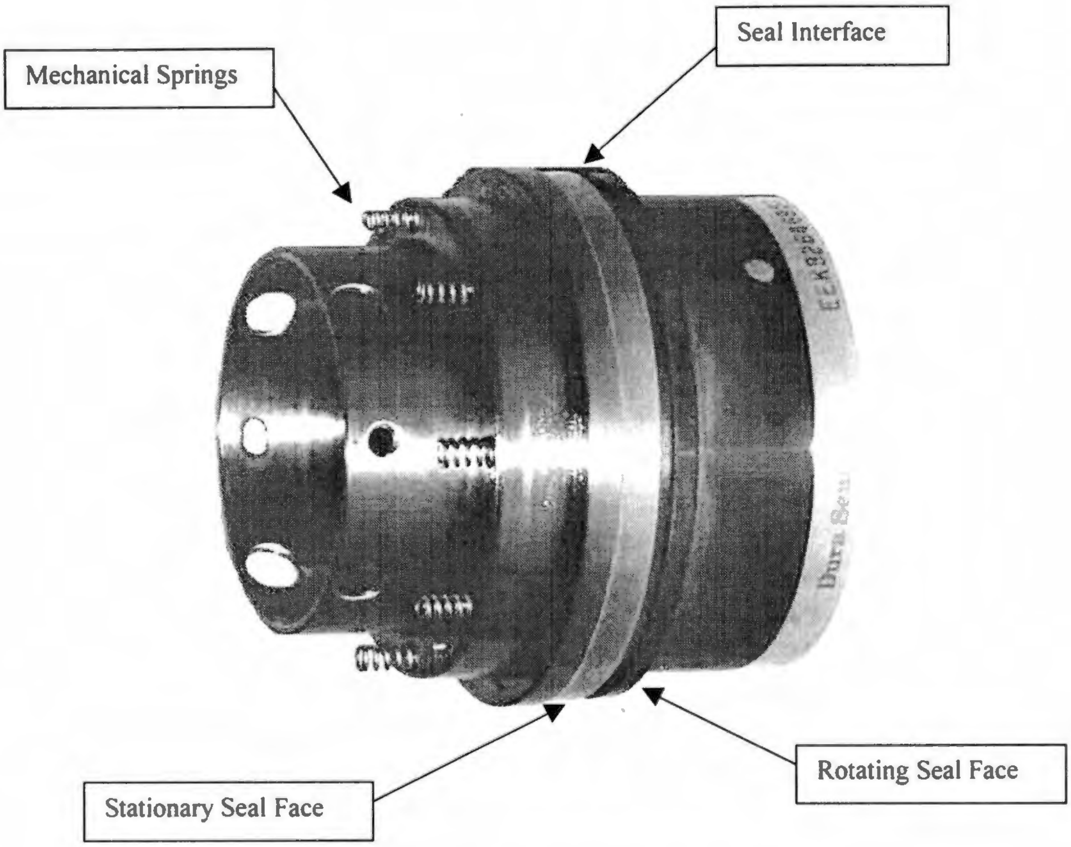


Figure 4 - Mechanical Seal Interface

The two seal faces that are compressed together with the mechanical springs (and forced apart by the fluid pressure) are of unequal widths. When these seals wear properly, the wear pattern on the wider seal is equal to the width of the narrow face; this wear pattern is shown below in Figure 5. It should also be noted that failures associated with this type of face wear are typically not related to seal face interactions.

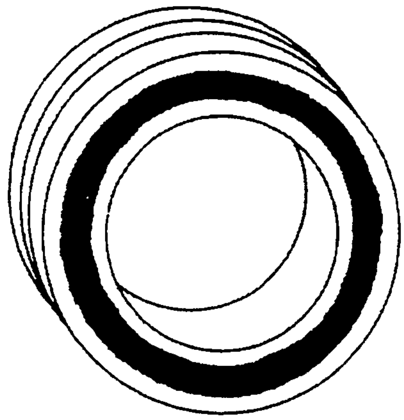


Figure 5 - Proper Wear Pattern[16]

If this wear pattern is narrower than the narrow seal width, it indicates that the thin lubricating film between the seal faces was inadequate. This type of wear pattern implies that the pressure of the pumped fluid was higher than the mechanical springs can counteract, and therefore an improper sealing surface was formed. This type of wear pattern is shown below in Figure 6.

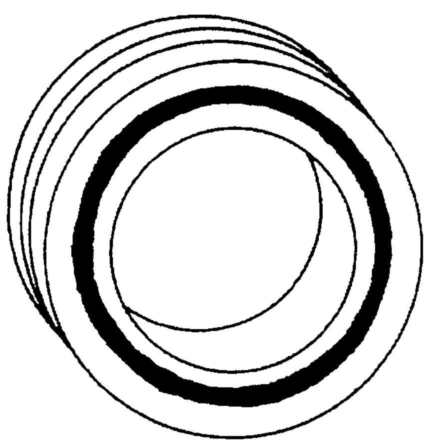


Figure 6 - Narrow Wear Pattern[16]

If the wear pattern is wider than the narrow seal face, several different faults can be attributed to the seal failure. These may entail pump misalignment, pipe strain, or pump cavitation. These all may cause the opening of the seal faces and cause too much lubricating film to be unevenly distributed along the seal interface. This wear pattern is shown below in Figure 7.

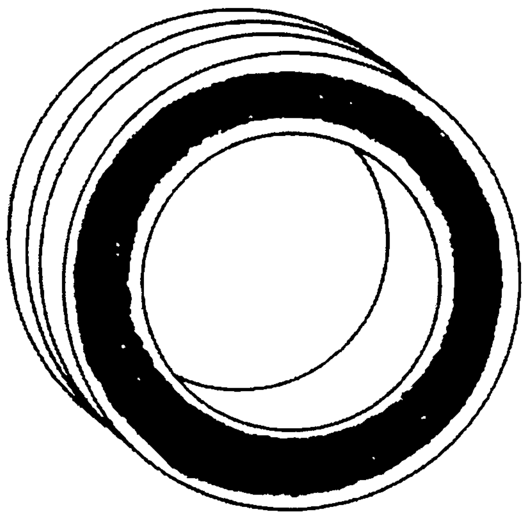


Figure 7 - Wide Wear Pattern[16]

If, on the other hand, there does not appear to be a wear pattern on the seal faces, this indicates that the rotary face is not turning against the stationary face. This could be attributed to the improper installation of the mechanical seal or the slippage of the rotary drive mechanism.

If the wear pattern on the seal faces is intermittent, then this indicates that the seal faces were not aligned exactly parallel with one another. This could be due to unflat seal faces, distortion of the stationary face when it was fastened to the pump housing, or an uneven tightening of the bolts that hold the seal in place. This wear pattern is shown below in Figure 8.

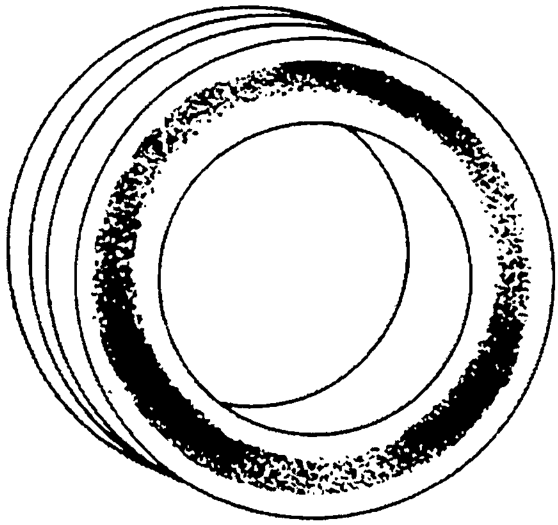


Figure 8 - Intermittent Wear Pattern[16]

The last wear pattern that has been correlated to mechanical seal failure involves the uneven rubbing of the seal faces. This form of wearing is usually indicative of an improper motor alignment, but may not always cause premature mechanical seal failure. This wear pattern is shown below in Figure 9.

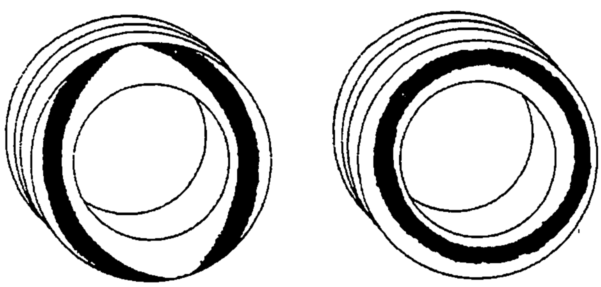


Figure 9 - Uneven Wear Pattern[16]

Along with examining the wear patterns of the seal faces, other abnormalities occurring on the seal faces have been correlated to mechanical seal failure. One such abnormality involves the overheating of the mechanical seal faces. This is often referred to as heat checking and is caused by the mechanical seal's inability to transfer generated heat away from the seal faces. Figure 10 shows how this method of failure affects the mechanical seal faces.

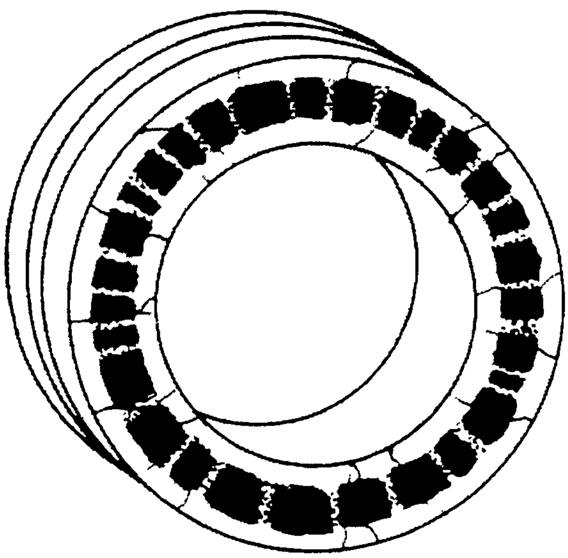


Figure 10 - Heat Checking[16]

Another result of improper temperature characteristics of the mechanical seal could be a cracked hard face. Sudden temperature spikes or mechanical shocks could cause the mechanical seal to fracture. This is shown in Figure 11.

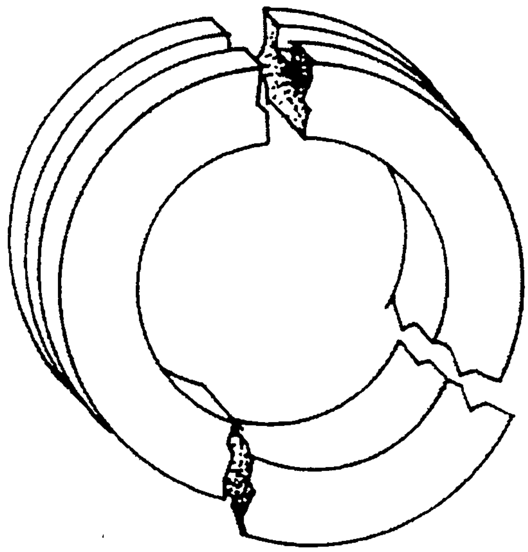


Figure 11 - Cracked Hard Face[16]

When pumps experience severe cavitation or pressure fluctuations, the edges of the mechanical seals could chip off. This chipping is a result of the mechanical seals being intermittently slammed against one another. Figure 12 shows how this phenomenon affects the seal faces.

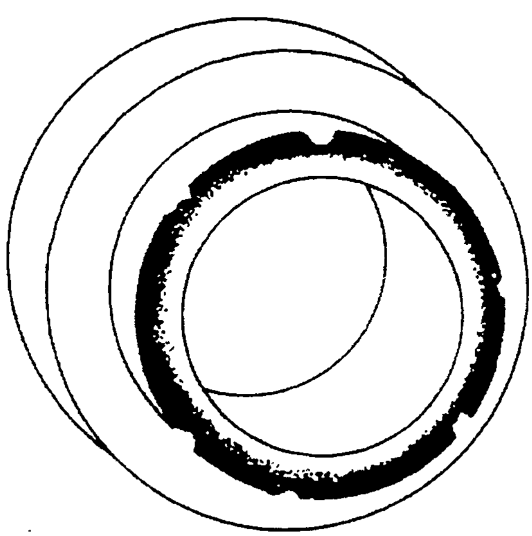


Figure 12 - Chipped Edges[16]

The faces of mechanical seals are often made by placing a hard facing (such as ceramic or tungsten carbide) onto a steel face. Whenever this coating is defective or there is a chemical attack on the seal faces, flaking and peeling could occur. This is represented in Figure 13 below.

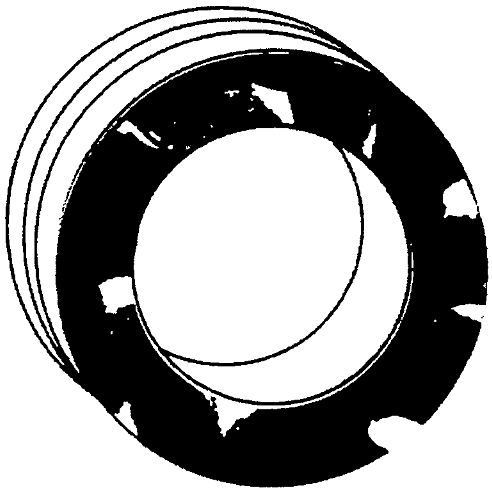


Figure 13 - Flaking and Peeling[16]

These chemical attacks are not limited to coated seal faces though. Whenever a solid mechanical seal face is attacked, pitting, blistering, or corrosion could result. Figure 14 below shows this phenomenon. Some chemicals may even begin to crystallize on the seal faces. This causes deep wear in the hard seal faces by the hard crystals. Figure 15 below shows how these crystals affect seal faces[16].

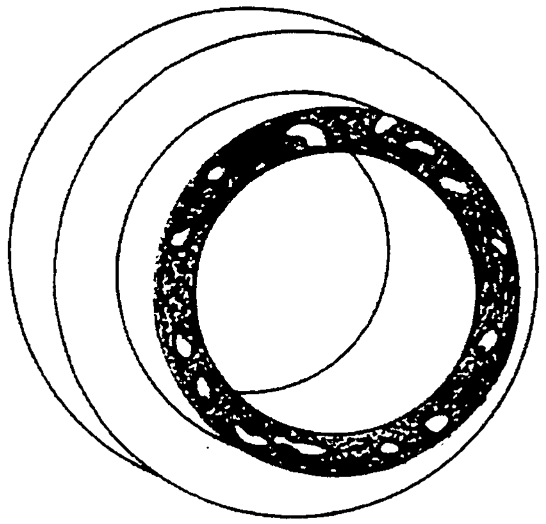


Figure 14 - Pitting, Blistering, Corrosion[16]

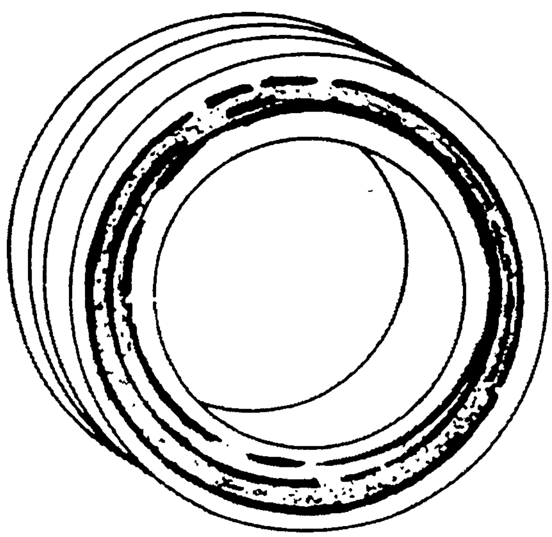


Figure 15 - Deep Wear in Hard Seal Face[16]

CHAPTER IV

EXPERIMENTAL SETUP

Equipment Selection

The first step involved in determining the components of the data acquisition system required knowing exactly which parameters of the mechanical seals should be monitored. These parameters were obtained from the information survey. These parameters include the vibration of the mechanical seal, the temperatures of the inlet and stuffing box fluids, the pressures of the inlet, outlet, and stuffing box fluids, the voltage and current drawn from the motor, and the dynamic alignment of the seal faces. Although all of these parameters were not monitored in this phase of the research, it was important to design a data acquisition system that would allow for the addition of any of these parameters.

The backbone of the data acquisition system is a CYDAS 1402HR data acquisition board. This board allows 100,000 samples to be taken per second, divided over 16 single ended or 8 differential inputs. These samples are taken with a 16-bit accuracy. This data acquisition board is run in an Intel Celeron 333a personal computer with 64 MB of RAM and a 6.4 GB hard drive. In order to manipulate the data acquisition board, LabVIEW, a graphical user interface data acquisition software package, is used. This combination of data acquisition board, personal computer, and software package offer an extremely efficient method of data acquisition which can be easily modified to incorporate additional parameters.

In order to begin the data collection process, the list of overall possible measurable parameters was reduced to a select few. This selection was made by determining the most important parameters within the imposed practicality constraints. The parameters selected include the temperatures of the inlet and stuffing box fluids, as well as the vibration of the stuffing box in two planes of motion, horizontal to the ground and along the vertical plane of the ground. These parameters could be measured unobtrusively and therefore normal operation of the pumps would not be affected during the data acquisition process.

In order to measure temperature effectively, a type-K thermocouple was introduced. This probe thermocouple was chosen due to its high temperature capabilities and low corrosive characteristics. In general, thermocouple's outputs often require amplification to provide accurate readings. For this purpose, an amplifier was incorporated to boost the output of the thermocouple to +/- 10 volts, to match the required input of the data acquisition board. The amplifier used was the Omni-Amp IV, which is powered by a standard 115 [V] power cord. This combination of thermocouple, power supply, data acquisition board, personal computer, and software package allowed for a least count of 0.017 [C] to be measured and recorded. A typical temperature response given by the thermocouple in application is shown in Figure 16.

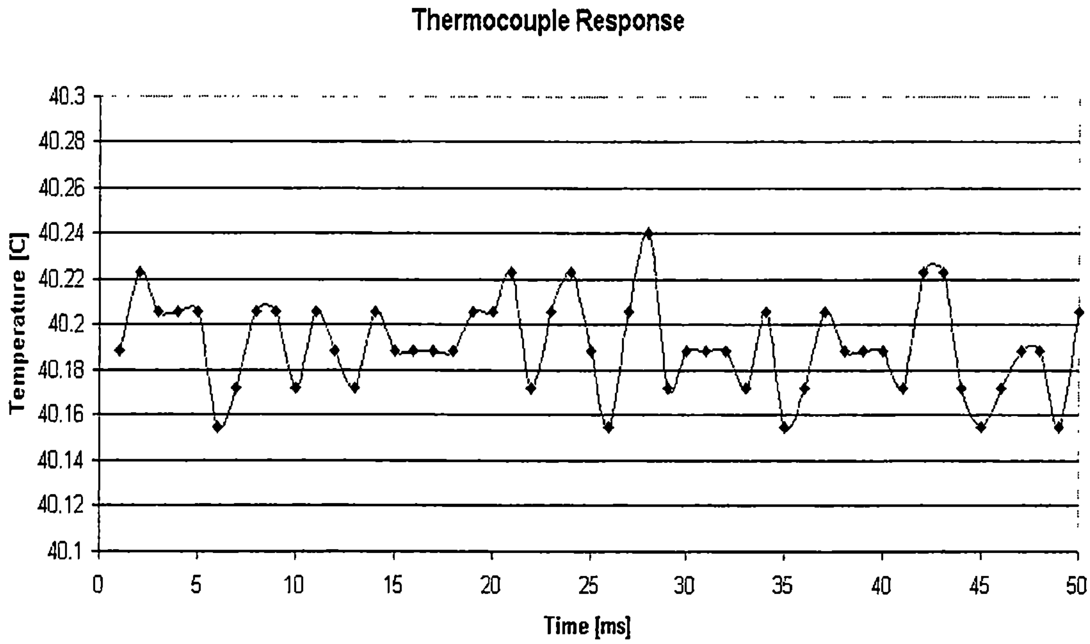


Figure 16 - Thermocouple Response

It should also be noted that the signals recorded by the thermocouple were subjected to a 0.8 [μF] capacitor to help filter out background noise recorded at the TED facility. Table 1 shows how the change in capacitance of the thermocouple output signal changes the average deviation of the count recorded by the data acquisition board. It should be noted that the capacitance is measured in micro-Farads and the count recorded by the data acquisition board needs to be mathematically manipulated in order to translate the raw data into a corresponding temperature.

[μF]	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Avg. Dev.	2.67	627.41	525.98	367.13	298.25	153.93	1.23	0.94	0.84	0.93	0.85

Table 1 - Thermocouple Stabilization

The vibration signal was measured with a piezoelectric accelerometer, which measures the change in acceleration of an object. Similar to the thermocouple, this output signal is too small to accurately record with the chosen data acquisition system, so the signal had to be amplified. This amplification was accomplished with a PCB PIEZOTRONIC model 480E09 ICP power unit, which allowed the 9.88 [mV/g] signal to be amplified by a factor of 100. This combination of equipment allowed the accelerometer signal to be measured at 1000 [Hz] with a least count of 0.08 [g]. Since the motors driving the pumps are rotating at 1750 RPM (29.17 [HZ]), the sample rate at 1000 [Hz] is sufficiently fast enough to adequately record all the vibration frequencies of the pump/seal assembly. A typical vibration reading from this assembly is shown below in Figure 17.

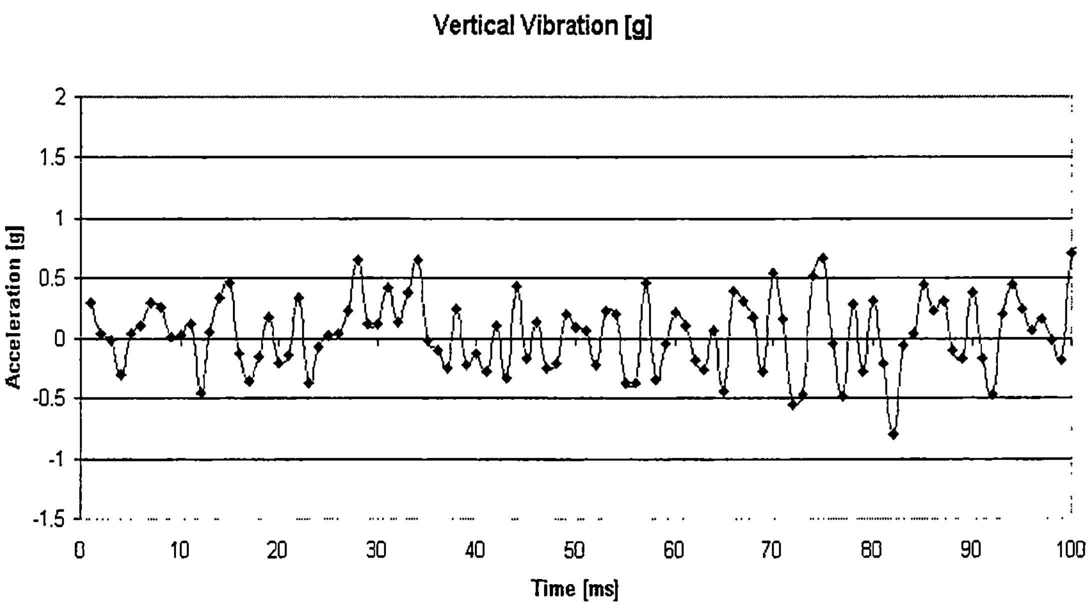


Figure 17 - Vibration Measurement

A schematic showing the instruments, amplification devices, and data acquisition system is shown below in Figure 18.

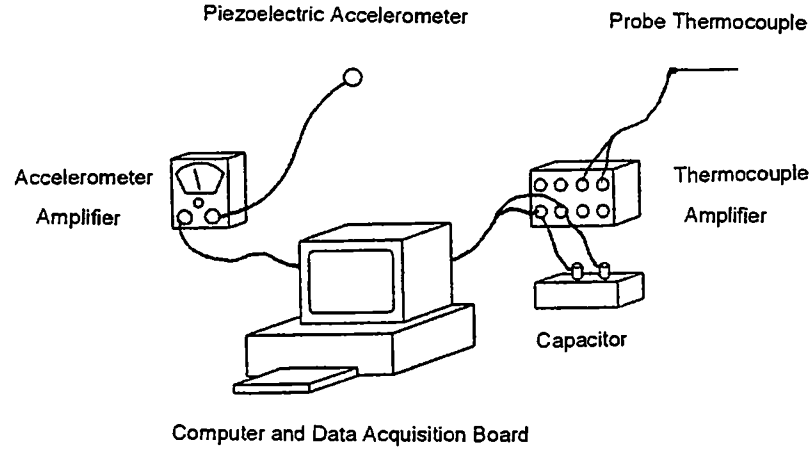


Figure 18 - Data Acquisition System

In order to determine the temperatures of the inlet and stuffing box fluids, a method of instrumenting the pump to make contact with the fluids had to be devised. This was accomplished by installing test plugs into the inlet piping and stuffing box of all of the test pumps. These test plugs consisted of two Nordel self-sealing seals that could withstand temperatures up to 140 [C] and pressures up to 1000 [psig]. Nordel is a neoprene type substance that is resistant to detergents, phosphates, esters, ketones, alcohols, and glycols but not resistant to petroleum products[17]. These test plugs allow the temperatures to be taken with the probe thermocouple and then the thermocouple can be removed so that each pump does not require a permanent thermocouple mount. These test plugs also allow the possibility of acquiring pressure data in the future with no additional pump modifications with the use of a pressure probe.

After all of the testing equipment was purchased and assembled into a complete data acquisition system, it was tested in the laboratory at the University of Tennessee. A Durco Mark III Group II pump was loaned from TED and installed at UT. This pump did not incorporate a flow loop, so the impeller was removed for all of the testing. The inlet and outlet flanges of the pump were sealed off and the stuffing box was filled with water to decrease the damage to the mechanical seal during the testing of the equipment.

By installing the test plug into the mechanical seal of the test pump, the stuffing box fluid temperature could be effectively measured. The vibration of the mechanical seal was also measured using the piezoelectric accelerometer with the amplifier. This testing at UT allowed for most of the problems associated with data collection to be corrected in the laboratory. This reduced the number of problems that would be encountered on site at TED where the testing would actually take place. A picture of this test pump is shown here as Figure 19.

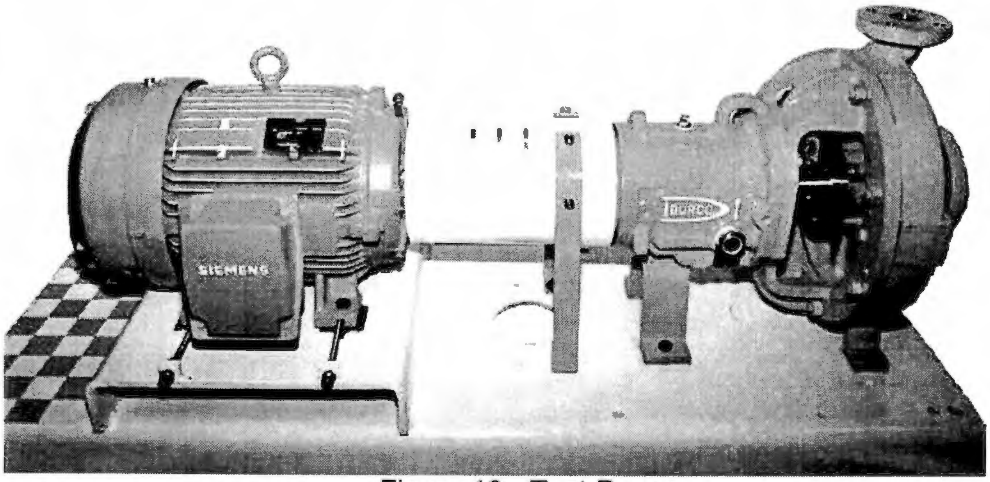


Figure 19 - Test Pump

Pump Selection

After all of the testing equipment was purchased, installed, and tested in the laboratory, the next phase of the research dictated using this equipment to take relevant data from pumps running at TED. Working with reliability engineers at TED, a group of ten pumps was selected to be used as test pumps for the data acquisition phase of the research. These ten test pumps were all located in the same general area of the plant and all performed similar functions. All of the pumps were used to pump water at a constant speed of 1750 RPM. There were some minor differences in the selected pumps though. For example, the pumps had different power ratings and incorporated different types of seals. Table 2 showing these ten pumps and their vital statistics follows:

Pump Tag	Description	Durco Size	Motor hp	Speed	Seal Type	Seal Installed
UK10	COOLING WATER	3X1-1/2-10/100	5	1750	Durametallic RO	Jan-98
UK11	COOLING WATER	3X1-1/2-13/130	15	1750	Durametallic RO	Dec-94
UK16	COOLING WATER	3X1-1/2-13/130	15	1750	Durametallic RO	Jan-98
WK10	COOLING WATER	3X1-1/2-10/100	5	1750	Durametallic RO	Jun-96
WK11	COOLING WATER	3X1-1/2-13/130	15	1750	AES CSS	May-99
WK16	COOLING WATER	3X1-1/2-13/130	15	1750	AES CSS	May-99
DC14A	TEMPERED WATER	4X3-10/90	10	1750	AES CSS	Feb-99
DC14B	TEMPERED WATER	4X3-10/90	10	1750	Durametallic RO	Dec-94
DK14A	TEMPERED WATER	3X2-13/92	20	1750	AES CSS (LB)	Jan-99
DK14B	TEMPERED WATER	3X2-13/92	20	1750	Durametallic RO	Dec-94

Table 2 - MRC Test pumps

Although these ten pumps are not exactly identical and do not perform identical functions with identical mechanical seals, they are similar enough in all of these categories so that the information obtained from one of the pumps should correlate with information from another. This uniformity of pumps and applications is an important aspect in the initial phase of this research. In order to develop a successful failure prediction system, initial similarity of testing criteria is essential. It would prove to be an impossible task to compare the mechanical seals of two pumps that operate under completely different circumstances.

CHAPTER V

ARTIFICIAL INTELLIGENCE

Artificial Intelligence Overview

The term "artificial intelligence" was first coined at the 1956 Dartmouth Conference sponsored by the Rockefeller Foundation. The scope of this conference was: "The potential use of computers and simulation in every aspect of learning and any other feature of intelligence[18,19]." In it's broadest sense, AI (artificial intelligence) is a combination of computer science, philosophy, and physiology which "encompasses a number of technologies that includes...expert systems, neural networks, genetic algorithms, fuzzy logic systems, cellular automata, chaotic systems, and anticipatory systems[18,19]." These technologies are primarily used in data and/or information processing environments to quantitatively evaluate relevant parameters and gain a general understanding of an unknown or uncertain phenomenon. In simpler terms, AI incorporates computer programming that is modeled on the behavior of the human brain to gain an understanding of situations that are often too complex for the human brain to fully comprehend.

This doesn't imply that artificial intelligence programs have the ability to "think", but rather to perform predesigned operations based on the human brain. The two basic approaches of AI research can be labeled as the bottom-up and top-down philosophies. The bottom-up philosophy of AI focuses on building electronic replicas of the human brain's complex network of neurons while the top-down philosophy focuses on mimicking the behavior of the brain with computer programs[19]. The difference between these two philosophies has led to the many different forms of artificial intelligence research being conducted throughout the world. For the task of predicting premature mechanical seal failure, two of these basic forms have been selected for consideration. These two major methods of artificial intelligence include artificial neural networks and fuzzy logic systems. They are the most prevalent methods of AI used for prediction analysis, yet are completely different in internal architecture and method of manipulating information. Despite these differences though, they are each very capable of performing a prediction analysis of failing mechanical seals.

Artificial Neural Networks

Artificial neural networks (ANN) subscribe to the bottom-up theory of AI research. This implies that effective AI architectures can be developed based on the biological infrastructure of the brain. The human brain is composed of a web of billions of cells called neurons which by themselves do not characterize intelligence, but when these neurons are grouped together, they are able to pass electrical signals through networks. Scientists believe that this method of network interaction is the cornerstone for intelligent behavior[19].

The fundamental cellular unit of the brain's nervous system is the *neuron*. These neurons receive and combine signals given off by other neurons by means of *dendrites.* If these signals are strong enough, they are transmitted to other *dendrites* by means of an *axon.* Before this transfer of signal takes place though, the signal is altered by the *neuron* as it passes through the *synapse.* This *synapse* accelerates or slows down the flow of the electrical charge before it is transmitted by the *axon.* It is this process of accelerating and slowing down of the electrical charges that yields the brain's ability to learn and remember. This process of learning occurs until the appropriate synaptic strengths between each connected neuron are adjusted. Figure 20 below shows a biological neuron and it's major components.

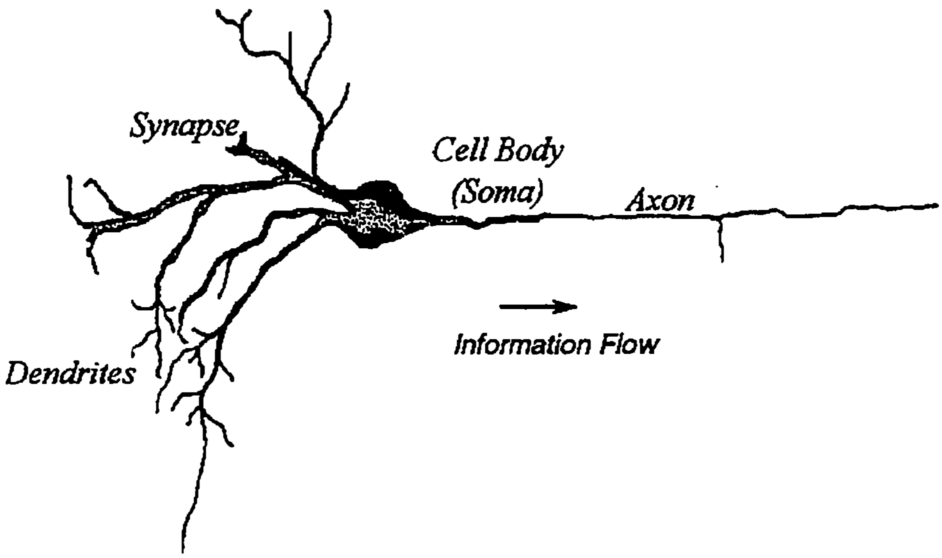


Figure 20 - Biological Neuron[18]

This basic understanding of a biological neuron led to the development of an artificial counterpart to be used for artificial intelligence research. Warren McCulloch and Walter Pitts, a medical doctor and a mathematician, showed how neurons could be considered as devices for processing binary numbers (1's and 0's)[19]. This observation proved the backbone for future research into the field, for computers were also run on such binary principles.

Artificial neurons were then designed based on their biological counterparts. Figure 21 below shows an example of an artificial neuron. The input signals to the neuron from other neurons are characterized by the symbols x_0, x_1, ...x_n, while the output signal is characterized by y_j. Similar to the synapse response in a biological neuron, these input signals are altered by weights. These weights actually alter the input signals by multiplying the signal by its corresponding weight. These weighted inputs are then summed together. This new value is sent through an activation function that dictates what type of response is passed as an output to other neurons.

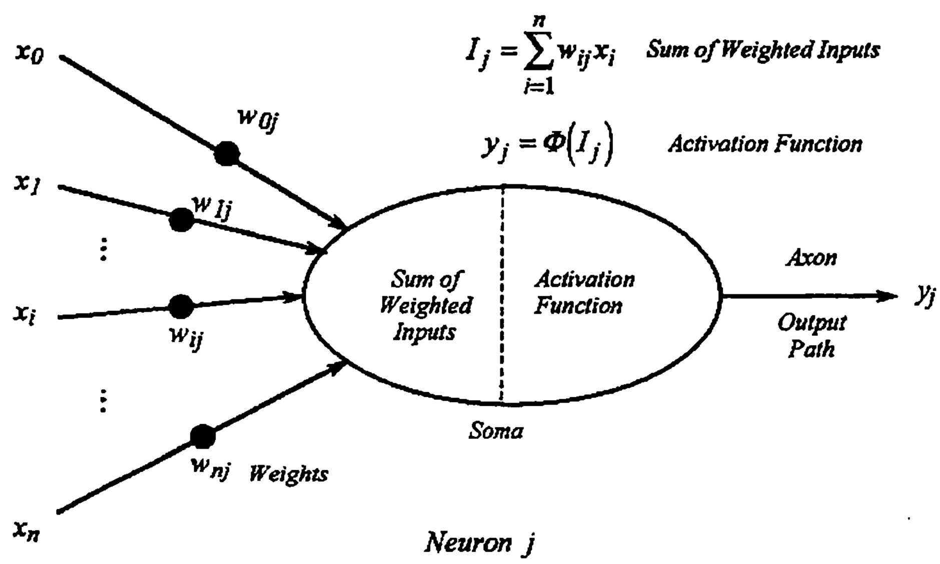


Figure 21 - Artificial Neuron[18]

As stated before, single neurons do not facilitate intelligence, but networks of them do. An artificial neural network can therefore be defined as: "A data processing system consisting of a large number of simple, highly interconnected processing elements (artificial neurons) in an architecture inspired by the structure of the cerebral cortex of the brain[18]." To accomplish such a network, these artificial neurons are grouped into layers, consisting of an input layer, and output layer, and at least one hidden layer. These layers allow the logical transferal of information between subsequent neurons. The most standard models of ANN are fully connected, this means that each neuron is connected to every possible neuron in adjacent layers. Each of these connections is accompanied by a weight, which must be trained in order for the ANN to yield a response from the output layer that corresponds to the input in the input layer. A schematic of a neural network that is not fully connected is shown below in Figure 22.

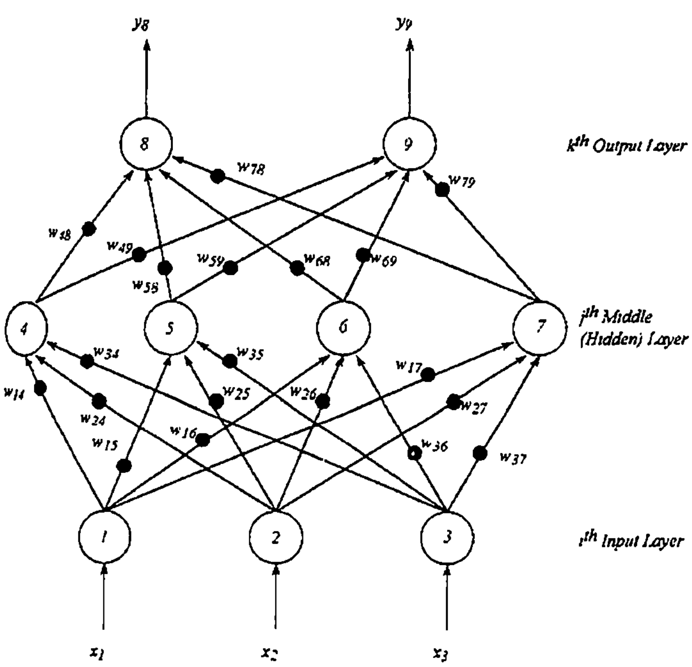


Figure 22 - Artificial Neural Network[18]

It is important to understand how an ANN with the described general architecture "learns". The two methods for training an ANN are supervised learning and unsupervised learning. Supervised learning, the most common in engineering applications, involves knowing the inputs and desired outputs of the artificial neural network. By supplying the neural network with known parameters for input and the desired output corresponding to the inputs, the ANN can internally adjust the weights connecting the various neurons. This process of adjusting the weights is analogous to the process of human learning. Each connection between the various neurons is given an initial random value, from which the ANN adjusts to make the outputs correspond to the inputs. The addition of neurons and hidden layers allows for more complex phenomenon to be modeled. As might be expected, this form of "learning" (called training) can sometimes take an extraordinary amount of time, and sometimes, based on the internal architecture of the ANN, might never be successfully achieved. This could be a result of the initial randomness of the weights, or too few or too many neurons or hidden layers.

Unsupervised learning on the other hand lets the ANN develop its own output response to a given set of data. Although this might seem pointless, it sometimes proves useful. These ANN are supplied with inputs and usually some sort of additional training criteria to accomplish some preset goal. This form of training requires the same method of weight distribution and adjustment as does supervised learning. Since this form of training requires the same basic training techniques, the same pitfalls for possible lengthy training times are present.

In general, "neural networks approach the modeling representation [of a system] by using precise inputs and outputs which are used to 'train' a generic model which has sufficient degrees of freedom to formulate a good approximation of the complex relationship between the inputs and the outputs[18]." Neural networks gain their main strength by rationalizing many different sets of data into understandable outputs, which the human brain is unable to do. This rationalization is very useful in the classification of multi-dimensional data into predetermined groups.

The main drawback of using an ANN involves the amount of data available concerning a phenomenon. As can be imagined, if only one set of input and output values is used to train the ANN, the ANN will only be able to recall those results. In order for an ANN to be able to represent a real world situation well, it must be trained with many data sets encompassing many aspects of the phenomenon. In specific terms, the ANN for this research must be trained with data sets (temperature differences, horizontal vibration, and vertical vibration) from several different pumps that are operating with different degrees of failing mechanical seals. Due to this constraint for a properly functioning ANN, ANN are considered to be data driven techniques. It is due to this data driven constraint relating to ANN that other methods of AI technology need to be investigated for determining a rational basis for mechanical seal failure prediction.

Fuzzy Logic Systems

As stated before, ANN were developed from the bottom-up philosophy of AI research, which require a large data set to properly train the network. Fuzzy logic systems on the other hand subscribe to the top-down philosophy of AI research. This means that instead of requiring a large data set to model a phenomenon, an expert knowledge of the phenomenon's parameters is required. Although fuzzy logic systems can be designed to classify phenomenon, like neural networks are able to do, they incorporate different architectures and philosophies to solve problems. Each technique of artificial intelligence provides unique capabilities for information processing. Unlike ANN, in fuzzy logic systems "the input and output variables are encoded in "fuzzy" representations, while their interrelationships take the form of well-defined *if / then* rules[18]." In general terms, fuzzy logic systems define input and output relations in linguistic terms and then use fuzzy modifiers and rule sets to correlate those relations.

The basic premise of a fuzzy logic system involves the understanding of the fuzzy set theory. In the classical set theory, parameters either belong to a prescribed set or are excluded from it, while in the fuzzy set theory, parameters are assigned a membership to the set. This fuzzy set theory allows for an amount of uncertainty and judgement to be rationally handled. This also allows for linguistic terms that lack exact definition to be incorporated into mathematical models. For instance, words like "very", "small", "hot", etc. can have a definite and specific meaning that is subject to change based on the situation at hand. It is important though to place a range onto which these fuzzy sets may act. "The elements of all the sets under consideration in a given situation belong to an invariable, constant set, called the ... universe of discourse[18]." This universe of discourse sets the minimum and maximum values that are to be considered in a fuzzy set.

As stated before, the fuzzy set theory allows for a membership to be placed on a parameter to determine its "belonging" into the set. Mapping these memberships along the universe of discourse allows for the formulation of a membership function, whereas every element of the universe of discourse yields a membership in the interval of 0 – 1.0. These membership functions represent subjective notions of a vague class. To help visualize this phenomenon, Zadeh diagrams for fuzzy sets were developed. Figure 23 below shows how someone might represent the phrase "very young" with a membership function.

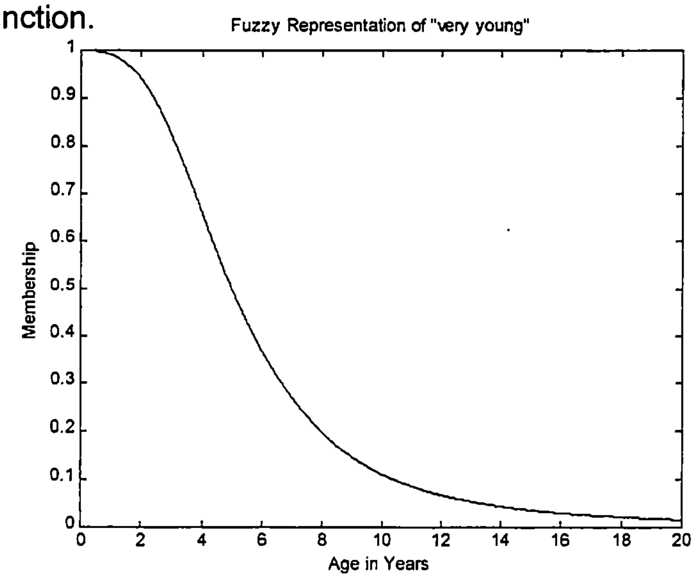


Figure 23 - Membership Function

Note how the universe of discourse is set to 0 – 20 years old and each age corresponds to a specific membership, ranging from 0 – 1.0. In this representation, higher memberships correspond to younger ages.

Fuzzy sets can also be used to describe numbers. Such descriptions take into account the possibility of ambiguity and uncertainty in data. Representation of fuzzy numbers can also take into account such linguistic terms as "nearly, "about", "almost" etc. Such a representation is shown below by comparing the two following figures. The first, Figure 24, describes a crisp number 5, while Figure 25 describes a fuzzy number 5.

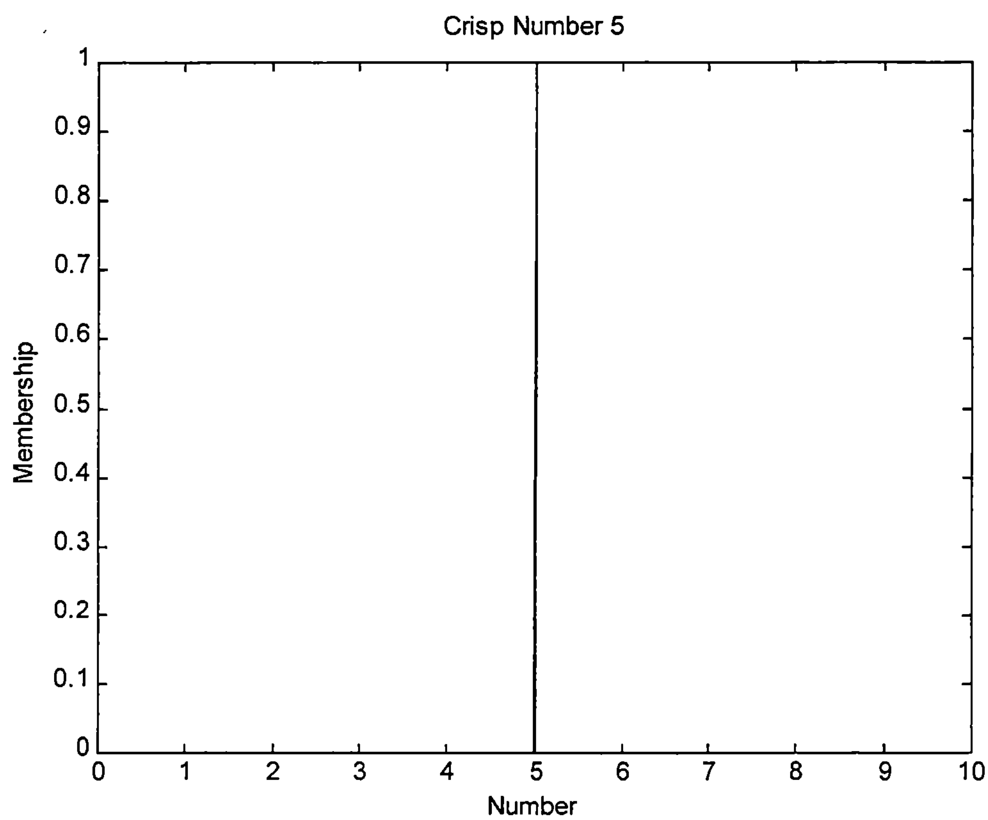


Figure 24 - Crisp Number

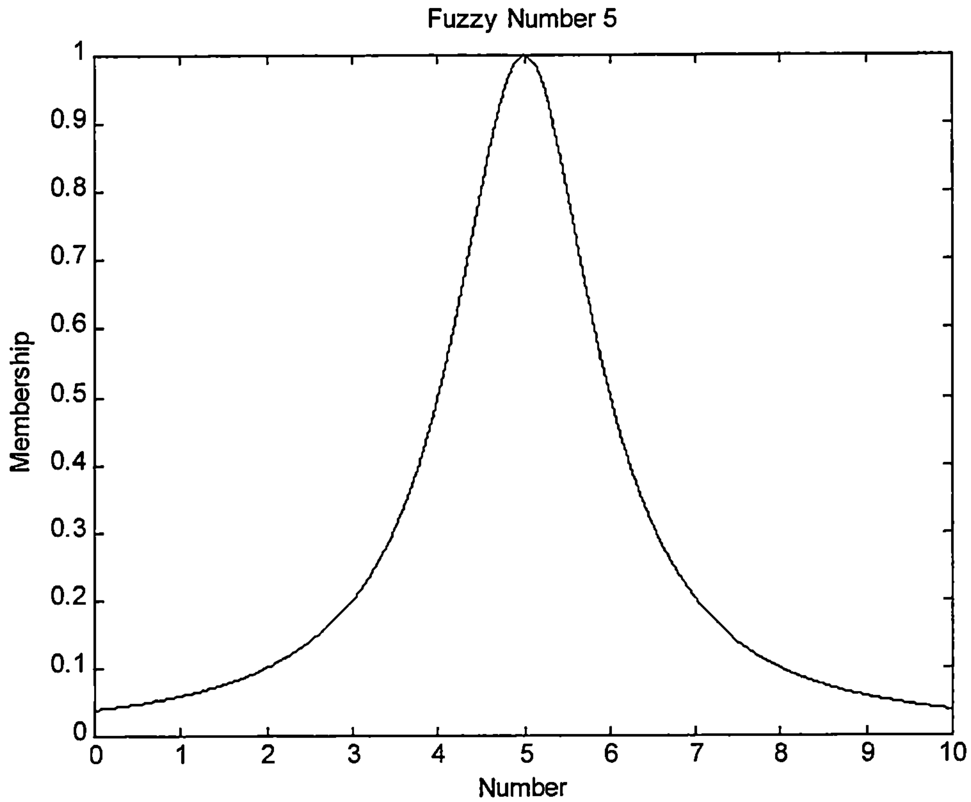


Figure 25 - Fuzzy Number

The true power of such representations occurs when multiple membership functions are placed on the same universe of discourse to describe the behavior of a system. Such a combination of membership functions allows for linguistic descriptions of a situation to yield adjustable and meaningful results. For instance, imagine a model of a room's temperature, where three different membership functions described the temperature. These membership functions could describe cool, moderate, and hot temperatures. Figure 26 below shows a Zadeh diagram of the temperatures and membership functions acting on the universe of discourse.

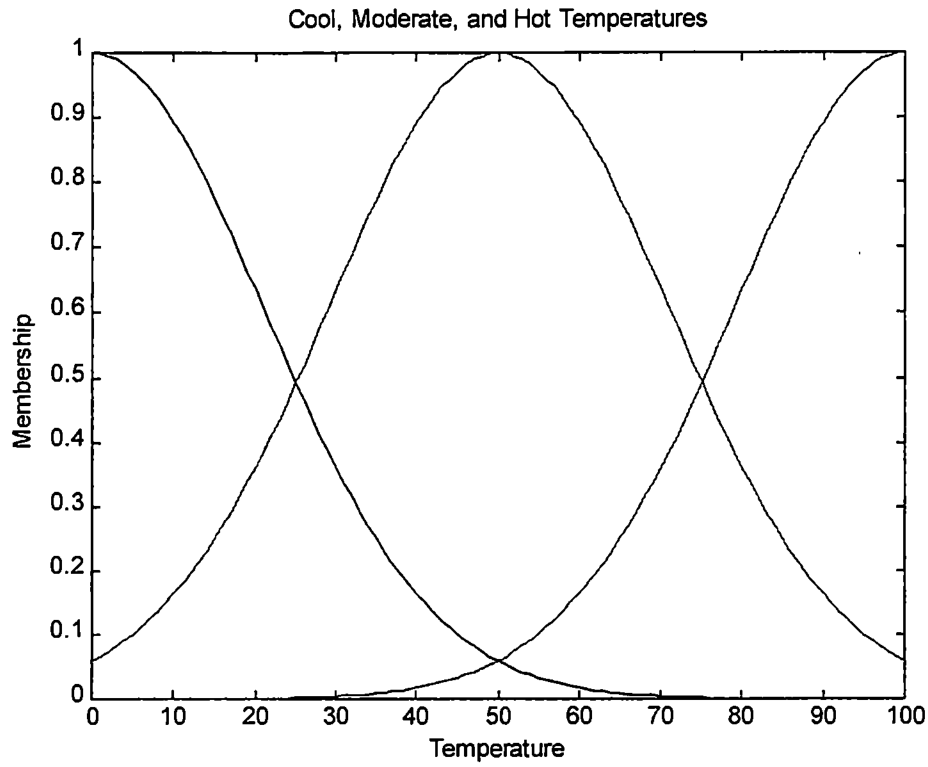


Figure 26 - Multiple Membership Functions

It is important to note how the membership functions overlap each other. This is an important characteristic in obtaining information about the system. Because these membership functions overlap one another, each temperature is represented by more than one membership function.

The next step involved in implementing a fuzzy logic systems involves determining the degree of fulfillment (DOF) of an input to each of the membership functions in the input's universe of discourse. In the above example, the DOF would determine how "hot", "moderate", or "cool" the room was. For instance, if the temperature from the above example is 30 degrees, then the "cool" membership function would yield a membership of approximately 0.36 and the "moderate" membership function would yield a membership of approximately 0.63. The "hot" membership function would not be triggered and therefore would yield a membership of 0.00. This can be seen in Figure 27 below. Once the DOF for each membership function has been determined, then these values are combined together in some way to yield a single fuzzy set describing the input.

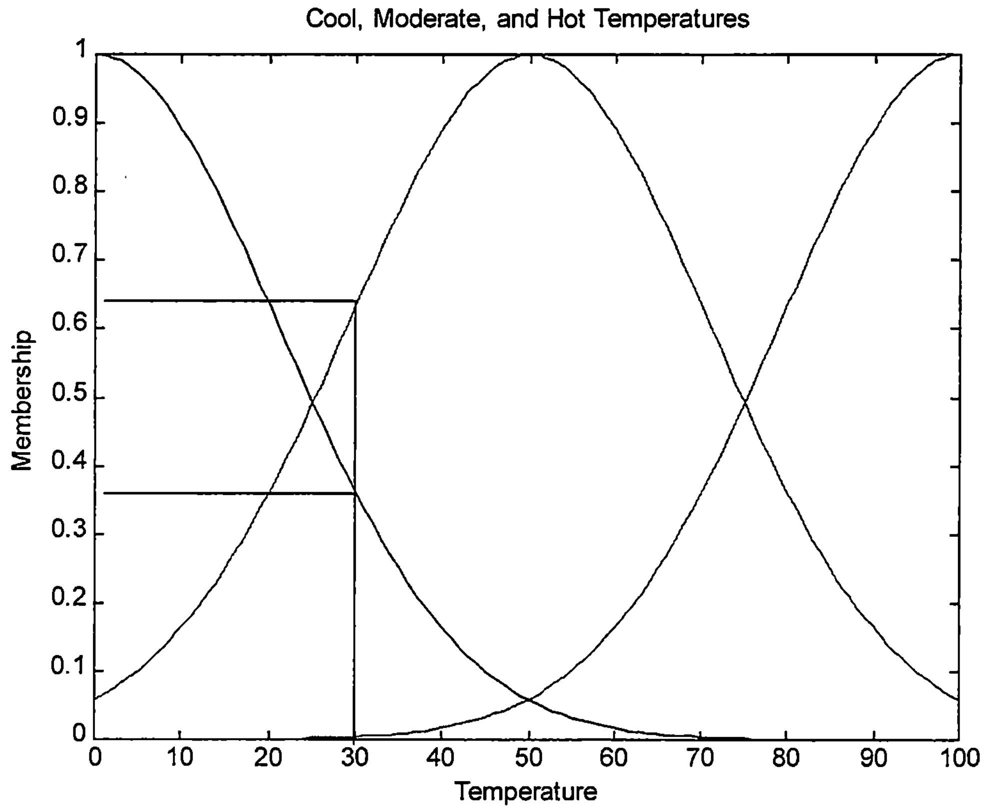


Figure 27 - Degree of Fulfillment

After the inputs were fuzzified by their respective membership functions, fuzzy operators were applied to the fuzzy values. The fuzzy operators contain the rule sets to which the fuzzy classification system is based. For instance, if the above system was used to control an air-conditioner based on the room temperature and the speed of a fan in the room, then the rules would relate the room temperature and fan speed. A typical rule would be "If the fan speed is low and the room is hot, then the air-conditioner is run on high." These fuzzy rules determine the overall behavior of the system.

After each of the fuzzy rules had been applied to the fuzzy inputs, they are implicated together. This process implies the consequents from the degree of fulfillment determined by the antecedents. This implication process provides a mathematical basis for which the rule sets are followed.

The next step involves combining these implicated functions into one fuzzy set. This fuzzy set is based on the rule sets and implication technique used to describe the situation and the inputs entered into the problem. After the various inputs have been altered by the various membership functions and rule sets, then one fuzzy set is formed. This fuzzy set is then de-fuzzified over a predefined universe of discourse to yield a crisp output, which corresponds to the crisp inputs. This defuzzification process can be accomplished in several ways, such as by taking the centroid of the area, the center of sums, or the mean of the maxima of the fuzzy output set. Regardless of the technique used, each process converts a fuzzy output into a crisp number. Such a process of de-fuzzification concludes the fuzzy logic system's operation.

Due to the design of fuzzy logic systems, they have the ability to make decisions in uncertain conditions without precisely defining the complex phenomenon involved. This form of AI lacks the data dependency that can plague an ANN, but it has weaknesses of its own. Fuzzy logic systems require expert knowledge of a system to adequately define its rule set. This rule set might be simply defined over two unknown but understood variables or it could be dependent on hundreds. As systems become more complex, the expert knowledge of systems grows exponentially due to the correlation and interdependency of the variables. As can be seen, ANN and fuzzy logic systems perform very differently, but each has been proven to have the ability to adequately classify the behavior of a system.

Another important differing characteristic regarding ANN and fuzzy logic involves the outputs. Artificial neural networks only allow for the output of a classification system to fall into a predefined group. For the air-conditioner example used above, an ANN would only allow the air-conditioner to be run at "high", or "low", or whatever other group the ANN was specifically designed to classify into. Fuzzy logic on the other hand, allows the outputs to be scaled. This means that the output could be any value from 0 to 1.0, where 1.0 implies that the air-conditioner is run on high and 0 implies that the air-conditioner is turned off. This difference is another important characteristic that should be noted when determining an optimal AI architecture to model a phenomenon. One of the objectives of this research is to determine which artificial neural network or fuzzy logic system could most accurately and adequately predict premature mechanical seal failure based on a set number of input parameters.

CHAPTER VI

ARTIFICIAL INTELLIGENCE SYSTEM DESIGN

Theoretical Design Considerations

As stated in the previous chapter, artificial neural networks and fuzzy logic systems each possess a limiting characteristic regarding their successful implementation. For the task of predicting premature mechanical seal failure, these limitations become a potentially fatal hindrance. For ANN, a large data set encompassing every aspect of mechanical seal operation is required to train the network to yield acceptable and meaningful results. Fuzzy logic systems on the other hand require an expert knowledge regarding the relationships between each of the measured parameters as to how they relate to mechanical seal failure.

Due to the limited amount of information that could be obtained from the mechanical seals in a two-month period, both of these limitations prove to inhibit the design of a working AI predictive system. Since the seals were only monitored for a two-month period, sufficient data relating to the mechanical seal's life cycle could not be obtained. To be more specific, no mechanical seals failed during this time, so the characteristics of a failed mechanical seal could not be observed. The fact that no failed mechanical seal was monitored prohibits any effective classification system, for no information relating to a failed seal was obtained. For ANN and fuzzy logic systems, the information from a failed seal is just as important as the information from a properly functioning seal. It is the combination of these two types of data that allows a proper classification system to be developed.

Despite this lack of sufficient data and expert knowledge though, several different theoretical AI architectures can still be developed. The AI architectures considered for this research include a multi-layer perceptron neural network (MLP), radial basis function neural network, (RBFN), probabilistic neural network (PNN), and a fuzzy logic system (FLS). Although sufficient data has not been obtained to yield a successful and accurate AI architecture, these four different AI architectures can be trained with a set of theoretical data that simulate actual data. This method of simulated training can help to show the limitations, benefits, and general architectures of each of the four AI systems. This simulated data set was based on seven input parameters (corresponding to the same seven parameters that will be incorporated in the actual AI systems when enough data is available). These seven inputs include: the temperature difference between the fluid in the stuffing box and the inlet fluid to the pump, the horsepower of the pump, the age of the installed mechanical seal, the RMS power of the vibration signal of the mechanical seal in the horizontal and vertical planes, and the maximum minus minimum vibrational acceleration of the mechanical seal in the horizontal and vertical planes.

To maintain consistency, the same data set was used to train each of the three ANN systems and the FLS. These data sets were then minutely altered to determine the ability of each of the four AI systems to generalize the data. This determines whether the AI architecture modeled the exact data presented or the general trends that the data represented. After the AI systems were tested for generalization, they were subjected to data sets that did not correspond to data that the AI systems had been trained with. This determined how well the AI system could process information for which it was not programmed to classify.

To be more specific, this theoretical data set consisted of 104 different "patterns" of data. Of these, 8 patterns represented "good" pump operation, 40 patterns represented "ok" pump operation, and 56 patterns represented "bad" pump operation. Each pattern consisted of five data sets that were generated from MATLAB's random number generator. These values were mathematically altered so that they would fall into specific ranges for each variable. This combination of ranges determined the foundations for the "good", "ok", and "bad" operating conditions. These same data sets were reinitialized and used to test the AI architectures for their ability to generalize the data. This is an acceptable means of testing for generalization since the data sets were based on random values that were confined to specific ranges. A complete listing of the 104 different data set patterns that comprised the 520 data sets used as the theoretical data is shown in the Appendix as Table A1. Along with these 520 data sets, the AI architectures were also subjected to data sets that did not follow any of the predefined theoretical patterns. These data sets determine how well the different AI architectures classify pump operation for cases that they were not trained to recognize.

Multi-Layer Perceptron Neural Network

The most basic and simple type of neural network is a multi-layer perceptron neural network (MLP). A perceptron is another name for a single neuron with adjustable weights and biases used for AI computer programming. These artificial neurons function in the same fashion as biological neurons, and are therefore grouped together to maximize their performance. By placing these neurons in consecutive layers, this interconnectivity can be achieved. A MLP consists of a series of multiple layers: an input layer, an output layer, and at least one hidden layer. The number of neurons (perceptrons) in the input layer corresponds to the number of input parameters for the AI architecture, while the output layer contains the same number of perceptrons as output parameters. The hidden layer (or layers) contain a varying number of perceptrons that are connected to both the input and output perceptrons.

For the task of classifying premature mechanical seal failure, a specific MLP architecture was tested based on the theoretical data set. The most adept MLP contained 7 input neurons, 15 hidden layer neurons located in one hidden layer, and 1 output neuron. This MLP used hyperbolic tangent activation functions in the hidden layer as well as the output layer. These activation functions mandate that the outputs of the neurons lie within the range of –1.0 to 1.0 despite the value of the input. This hyperbolic tangent function is the most widely used activation function for classification purposes due to the specific output values ranging from –1.0 to 1.0. Figure 28 below shows basic functioning of a hyperbolic tangent activation function. Notice that despite the value that the function corresponds to on the x-axis, the function always yields a value between –1.0 and 1.0 in the y-axis.

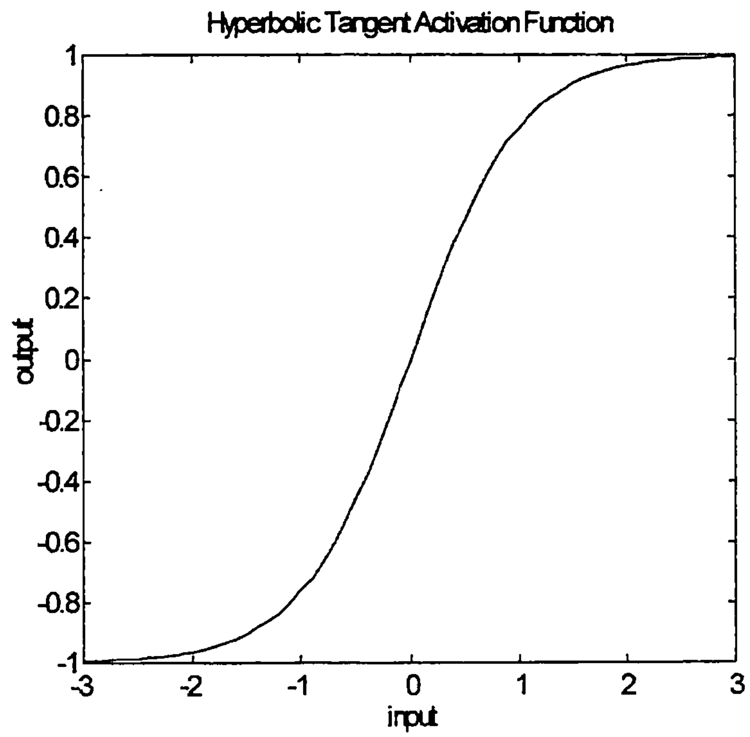


Figure 28 - Hyperbolic Tangent Activation Function[20]

This MLP was trained with the 520 data sets that were created based on the information obtained from the pumps that are being monitored at TED. These data sets were divided into 3 categories, corresponding to "good", "ok", and "bad" seal performance. These categories were used to train the MLP by having the "good" performance correspond to an output value of 0.95, the "ok" performance correspond to an output value of 0.00, and the "bad" performance correspond to an output value of –0.95. This training follows the perceptron learning rule, which implies that if the outputs do not correspond to the inputs, then the weights and biases of the individual perceptrons are updated. If the outputs do correspond to the inputs however, the weights and biases of the perceptrons are unchanged. As stated before, the weights corresponding to the neurons are multiplied by the inputs of the neuron, which is similar to the function of the synapse in the biological neuron. A bias is a mathematical addition placed on the input in conjunction with the weights. This bias often serves to break a linear dependency of a system that is being modeled.

Radial Basis Function Network

Similar to the architecture of a MLP, radial basis function networks (RBFN) also consist of a pattern of layers. In the case of a RBFN though, this number of layers is always three: an input layer, a hidden layer, and an output layer. The main distinguishing characteristic between a MLP and a RBFN involves the activation functions used. Unlike MLP that typically incorporate logistic or tangential activation functions, RBFN use the Gaussian kernel function. This function acts like a standard Gaussian distribution function with the width of the function being alterable. Figure 29 below shows a typical Gaussian activation function. Notice how the values in the x-axis correspond to the values in the y-axis in much the same way as the membership functions do in a fuzzy logic system. Despite the input to the function, the output of the function is always a value between 0.0 and 1.0, which is centered in a Gaussian fashion around a single value. The utilization of this method of activation function allows for classification techniques to be enhanced.

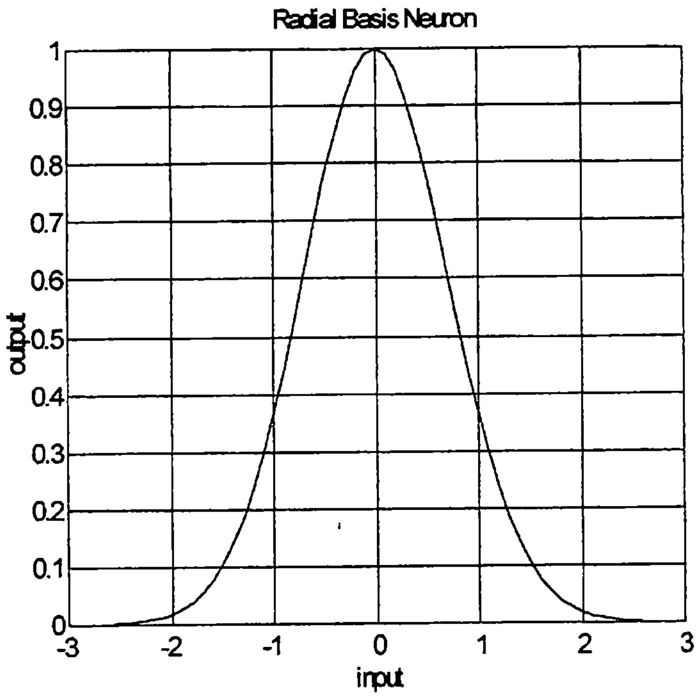


Figure 29 - Radial Basis Gaussian Kernel Function[20]

Due to this series of gaussian activation functions, the radial basis functions "thus produce localized, bounded, and radially symmetric activations – that is, activations rapidly decreasing with the distance from the function's centers (in contrast, the...sigmoidal activation functions [used in MLP] produce global and unbounded activations[18]." In simpler terms, RBFN are more adept to classifying systems than MLP. The RBFN used for this research incorporated 7 input neurons and 1 output neuron, the same as did the MLP. Unlike the MLP though, the RBFN does not require a pre-specified number of hidden layer neurons, it incorporates a method of unsupervised learning to determine exactly how many hidden radbas neurons are required to sufficiently classify the situation. This unsupervised learning is based on the spread constant of the Gaussian activation functions (which act as a means for centering these classifications on important characteristics). This RBFN was trained with the same 520 data sets as the MLP.

Probabilistic Neural Network

The last neural network designed was a probabilistic neural network (PNN). PNN use the training data to "develop distribution functions that are in turn used to estimate the likelihood of a feature vector being within several given categories[19]." Basically, PNN utilize a supervised learning technique to assign a probability that the given input will lie in each of the known possible classification. The classification with the greatest statistical probability "wins" the classification. PNN are primarily used to classify systems based on a known classification scheme. The only data provided to a PNN are the data sets, output classifications, and a width parameter (similar to that used in the RBFN). PNN have an advantage of always determining the probability distribution of the data sets, with no possibility of an untrained NN. This is due to the fact that PNN are based on the principles of Bayesian probability, which dictates that a certain probability exists relating a data set to a specific output.

PNN are composed of four layers, unlike the previous AI architectures that only contained three. These four layers are the input layer, pattern layer, summation layer, and output layer. A diagram showing the basic interactions of a PNN is shown in Figure 30 below.

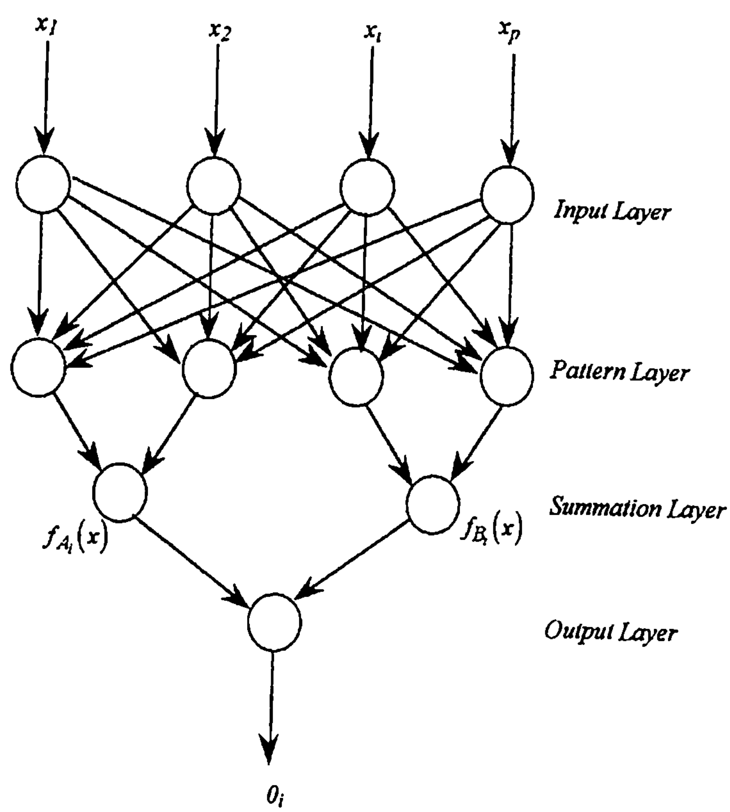


Figure 30 - Probabilistic Neural Network[18]

The internal architecture of this type of neural network is similar to those previously discussed though. The input layer contains a neuron for each of the input types, in the case of classifying mechanical seals, 7 (like used in the MLP and RBFN). The pattern layer contains one neuron for each neuron in the input layer, and the neurons are all fully connected to the input layer. The third layer, the summation layer, contains one neuron for each category that the PNN is directed to classify into. These layers simply add the values from the pattern layers together to their respective classifications. The final layer interprets the results from the summation layer and yields a single output value, which corresponds to the classification of the input.

Another major difference in the PNN architecture from those previously discussed involves the activation function. As opposed to the hyperbolic tangent or Gaussian kernel activation functions used earlier, the PNN uses an exponential activation function. This function essentially yields an output of zero for small inputs, and very large outputs for inputs greater than 1.0. The probabilistic properties of the PNN result from these activation functions in conjunction with the general architecture of the neural network.

The PNN used for this research was trained with the same 520 data sets as was the MLP and RBFN. This PNN contained 7 input neurons in the input layer, 7 pattern neurons in the pattern layer, 104 neurons in the summation layer, and 1 neuron in the output layer. Although there were 104 neurons in the summation layer (which would indicate 104 different classifications), the PNN was actually trained to yield outputs similar to those of the MLP and RBFN. PNN work best when they classify N^{th}-dimension problems into M classes, where M>N. In this research, that would imply classifying a seventh dimensional problem into three classes. To counteract this limitation, a pseudo-learning vector quantization (LVQ) algorithm was implemented into the output. This pseudo-LVQ technique allowed the PNN to classify the data into the 104 possible patterns for which it was trained, and then re-classify those patterns into three sets, "good", "ok", and "bad". Actual LVQ algorithms involve the addition of an extra layer of neurons to help decipher the outputs from the output layer neurons, but the pseudo-LVQ method used simply reclassified the output data.

Fuzzy Logic System

Along with the three neural network systems designed to predict premature mechanical seal failure, a fuzzy logic system was also designed. The fuzzy logic system was based on a complex set of 105 fuzzy if/then rules. These 105 fuzzy rules mirror the 104 different patterns of data that were used to train the theoretical neural network systems. A typical rule would be: 'If the motor is 5 hp *AND* the seal is new *AND* the temperature difference between the inlet fluid and stuffing box fluid is small *AND* the RMS vibrational power in the horizontal direction is low *AND* the RMS vibrational power in the horizontal direction is low *AND* the maximum-minimum vibration in the horizontal direction is large *AND* the maximum-minimum vibration in the vertical direction is low, *THEN* the seal is bad.' It should be noted here that the combination of 7 input parameters yields the possibility of 1944 fuzzy rules, but the majority of these were not programmed due to ambiguity or the unlikelihood that certain combinations of input parameters would arise.

The first step taken in designing the fuzzy logic system involved defining the universe of discourse and membership functions for each of the input and output parameters. The universe of discourse for the motor horsepower ranged from 0 – 25 [hp], with four singleton membership functions located at 5, 10, 15, and 20 [hp]. Singleton membership functions were used to guarantee that the motor horsepower would only be considered in one membership function at a time. A graphical representation of the motor horsepower's universe of discourse is shown below as Figure 31.

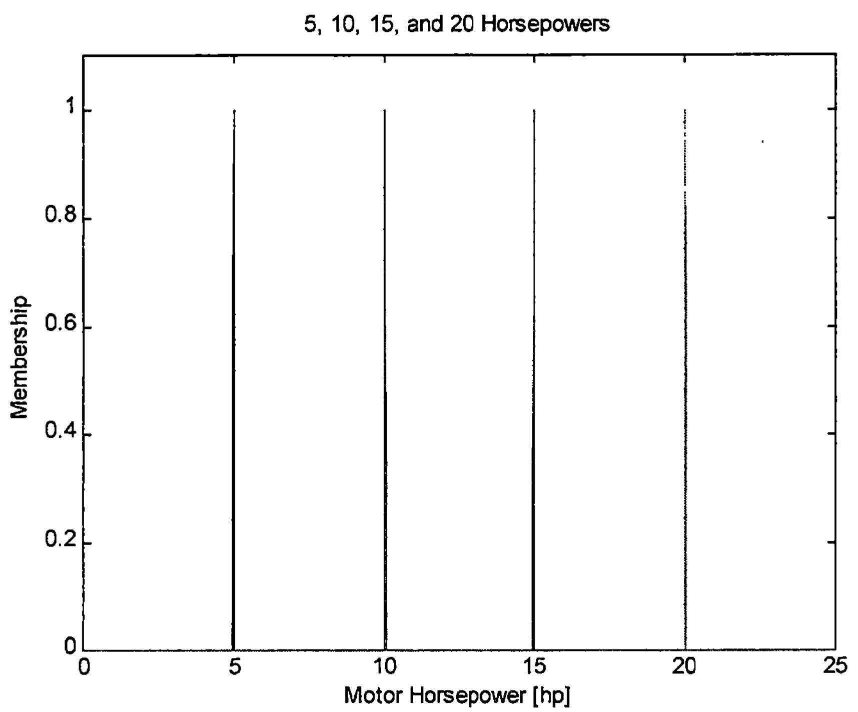


Figure 31 – Motor Horsepower Universe of Discourse

The next input variable concerning the mechanical seal performance is the age of the mechanical seal. This age is represented in decimal terms to describe how many years the mechanical seal has been installed in the pump. The universe of discourse for the seal age ranges from 0.00 – 10.0 [years]. This universe of discourse is divided into two regions by triangular membership functions. These two triangular membership functions represent how "new" or "old" the mechanical seal is. Figure 32 below depicts the universe of discourse for the age of the mechanical seal.

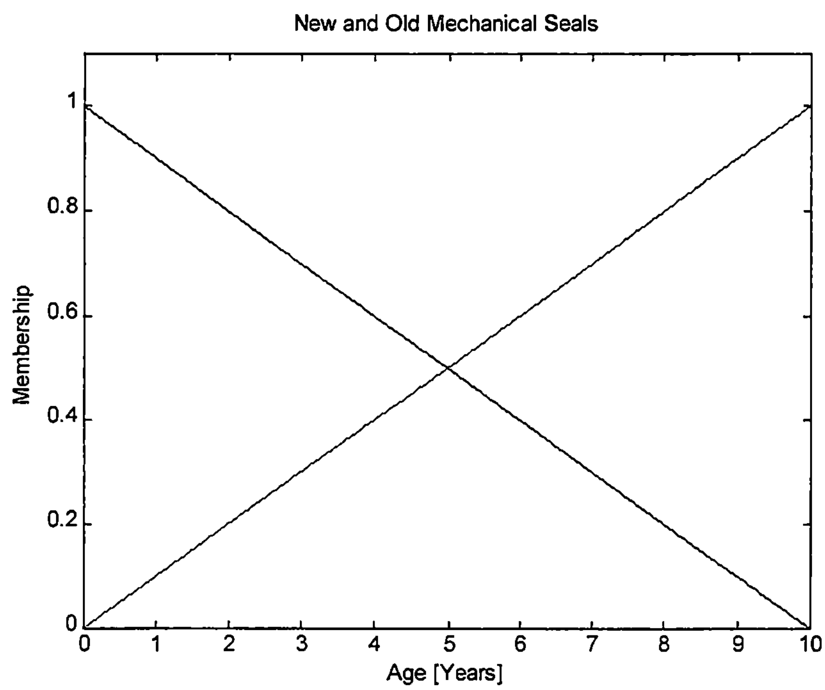


Figure 32 – Seal Age Universe of Discourse

The next input variable involved the change in temperature between the inlet fluid and fluid in the stuffing box. This universe of discourse was set from 0.50 – 9.50 [C] with three Gaussian membership functions to divide the universe of discourse into three regions. These three regions are "cool", "moderate", and "hot" temperature differences. This universe of discourse is shown below in Figure 33.

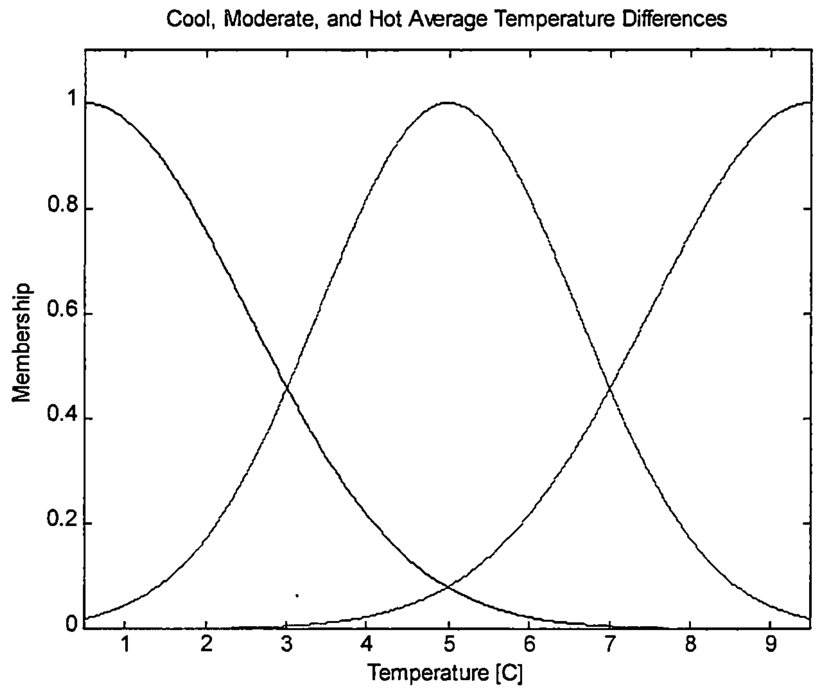


Figure 33 – Temperature Difference Universe of Discourse

The universes of discourse for the RMS horizontal and vertical vibrational power were identical, with identical membership functions. Three Gaussian membership functions were used to divide the universe of discourse, which ranges from 0.00 – 1.00 [g], into "light", "moderate", and "high" RMS powers. These two universes of discourse are depicted below in Figure 34.

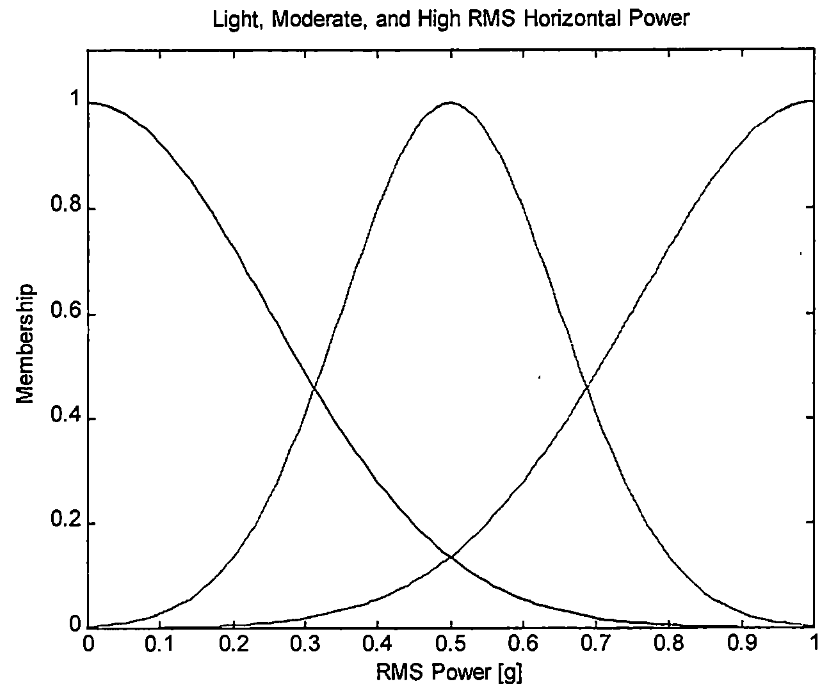


Figure 34 – RMS Power Universe of Discourse

Similarly, the universes of discourse depicting the maximum – minimum horizontal and vertical vibration were identical. These universes of discourse ranged from 0.00 – 8.00 [g] and were divided by three Gaussian membership functions depicting "low", "moderate", and "high" vibrations. This configuration is shown below in Figure 35.

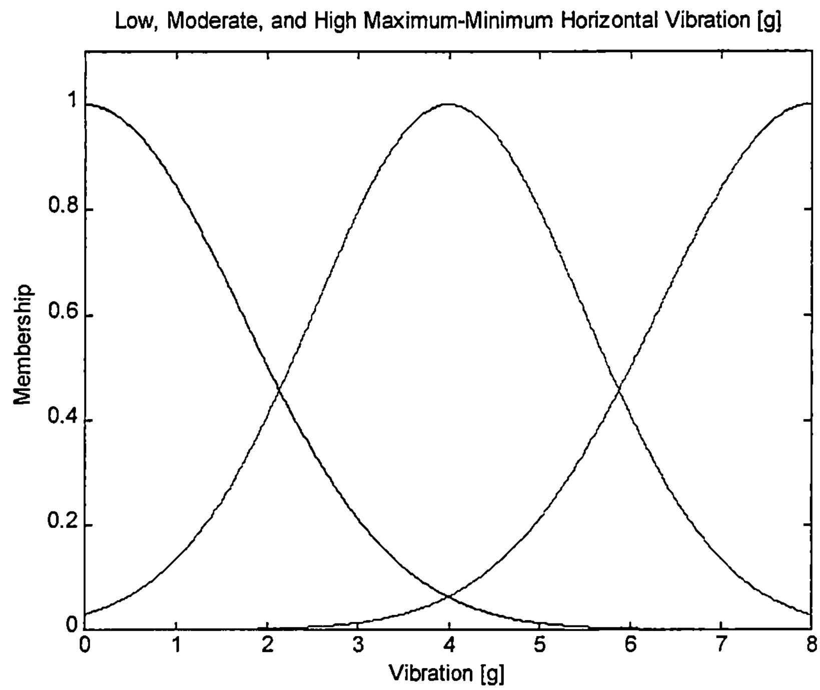


Figure 35 – Maximum – Minimum Vibration Universe of Discourse

Along with the universes of discourse designed for the input parameters, the output parameter's universe of discourse (seal performance) also had to be determined. The output universe of discourse was ranged from 0.00 – 1.00, depicting a scaled performance of the seal. A value of 0.00 indicates an extremely poorly performing seal, while a value of 1.00 indicates a perfectly performing seal. This universe of discourse was divided into three regions with Gaussian membership functions. These three regions depict "low", "moderate", and "high" seal performance. This output universe of discourse can be seen below in Figure 36.

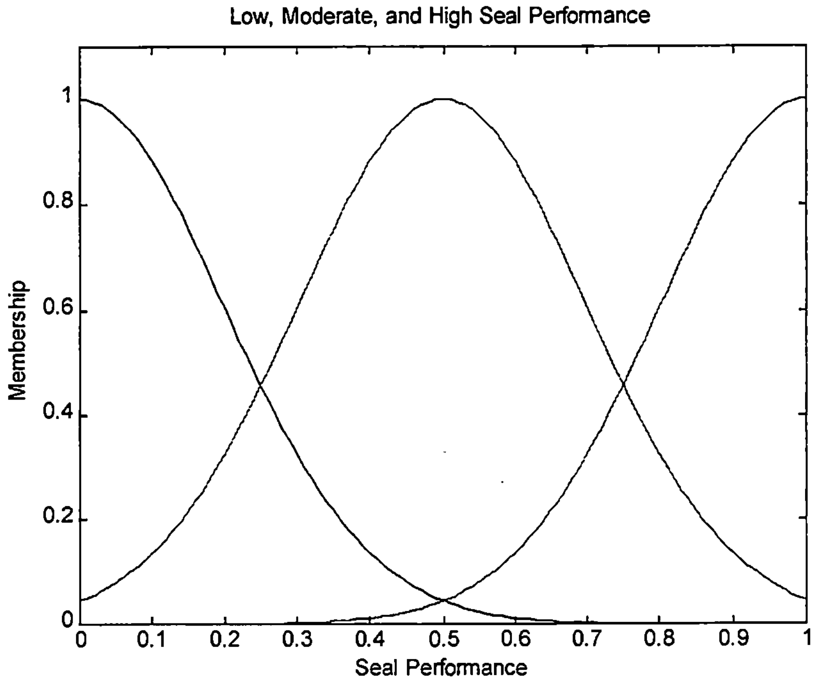


Figure 36 – Seal Performance Universe of Discourse

The combination of the 105 fuzzy rules acting over these eight universes of discourse allowed for the theoretical system to be adequately modeled. This FLS allowed for the mechanical seal's performance to be predicted based on expert knowledge of the seal's behavior. It is important to note here that since enough data could not be collected from pumps in operation at TED, an expert knowledge of the phenomenon's characteristics could not be achieved. This lack of expert knowledge was overcome here by implementing theoretical expert knowledge that was based on the theoretical data used to train the neural network systems.

CHAPTER VII

LABORATORY TESTING

As stated previously, no data from failed mechanical seals could be obtained during the process of this research. The lack of this data made it impossible to develop a working AI code to predict premature mechanical seal failure. In fact, it could be several years until one of the monitored pumps at TED experiences a seal failure. This hindrance can be alleviated and even exploited upon through the initiation of a new test platform. The current research focused on periodically monitoring centrifugal pumps in operation at TED to record the changes in operating conditions over time. In theory, as the mechanical seals begin to fail, their measured parameters would change corresponding to their specific mode of failure. This monitoring process is limited to the operating circumstances of the existing pumps. A more controlled and systematic approach would be to develop a test platform in a laboratory setting to induce faults in centrifugal pumps, and monitor how those faults affect the mechanical seals.

By placing the existing Durco test pump into a closed flow loop, this method of testing could take place. This flow loop would allow the test pump to be operated in a fashion similar to that of an actual pump used in industry. With this flow loop established for the test pump, known faults could be induced onto the pump to monitor the adverse affects received by the mechanical seal. These faults should include all of the major faults mentioned previously that are believed to contribute to mechanical seal failure.

These faults include: depriving the pump of the required NPSH, running the pump with too high of a suction pressure, running the pump dry, running the pump off the design point, and pumping a fluid that has too high of a specific gravity or viscosity. Along with the design or operationally induced faults, faults due to the pump assembly could also be induced. These faults include: misaligning the shaft, installing the bearings incorrectly, deflecting the shaft, installing the impeller out of balance, installing the coupling out of balance, and installing the mechanical seal incorrectly. Other faults that could be effectively induced in the pump include: installing the baseplate incorrectly, not anchoring the piping well, overheating the seal faces, and separating the seal faces to prevent seal flush.

The majority of these faults could be induced with a minimum amount of time or effort. Placing restriction valves into the flow loop and preventing the required amount of fluid from entering or leaving the pump could induce the faults associated with NPSH and pressure. Dry running the pump and running the pump with fluids that have a high viscosity or specific gravity would however require draining the flow loop. After the flow loop was drained, the pump would have to be run with no fluid, or run with a with the new properties desired. The faults relating to the imbalance of parts could be induced by simply adding weights to certain portions of the part and monitoring the effects this had on the mechanical seal. Likewise, inducing faults relating to misaligned parts could be accomplished by establishing a test bed to mount the pump and motor on. By adjusting the position of the motor in relation to the pump on this test bed, known misalignments to the rotating shaft could be induced and monitored. The motor and/or pump could be loosened from this test bed in order to simulate the improper assembly of the baseplate.

The last faults that would need to be induced and monitored involve the mechanical seals. It would be fairly simple to install the mechanical seal incorrectly, but the other faults could be slightly more difficult to induce. The seals could be artificially overheated by placing a heating device in a close proximity to the mechanical seal, or by pumping extremely hot fluids. Obviously, pumping the hot fluids might provide more accurate responses, but this would require a much more elaborate, expensive, and possibly hazardous flow loop. The addition of a heating device, such as a small space heater, directed onto the mechanical seal, could serve the same purpose. The lack of seal flush could be simulated by tampering with the mechanical springs on the backside of the mechanical seals. These springs counteract the pressure forces of the fluid in the flow loop, but if some of the springs were more or less stiff than others, a lack of seal flush could be obtained. Flow straighteners could also be installed into the flow loop to monitor the effects of excessively turbulent flow on the overall vibration of the system.

Similar to the information obtained from the pump monitoring at TED, the information obtained from this testing procedure would undoubtedly assist in predicting premature mechanical seal failure. Unlike the information obtained from TED however, the laboratory testing would allow the characteristics of failing mechanical seals to be directly attributed to specific failure causes. This correlation could eventually be incorporated with the information obtained from the test pumps at TED. The combination of the laboratory test data with the actual data recorded from operational pumps could not only determine when a mechanical seal was going to fail, but could also determine what was causing the failure. This would serve to not only repair or replace the mechanical seal prior to failure, but also to service the pump/seal assembly to alleviate the mode of failure. It is this comparative analysis that strengthens the value of the information obtained from the laboratory testing of induced faults.

CHAPTER VIII

RESULTS AND DISCUSSION

Data Collection Procedure

As stated previously, the data collection process on the ten monitored pumps at the TED facility has only been performed for a limited time. This lack of long-term data prevents the successful correlation of seal performance to the measured parameters of the mechanical seals. This is not to say that the data-collection process thus far has been unfruitful. The collected data has laid the foundation for the design of several different AI architectures, as well as served to initiate a long-term data-collection process. From the collected data, seven different mechanical seal characteristics could be determined. These seven characteristics include: the age of the mechanical seal, the horsepower of the motor, the average change in temperature between the inlet and stuffing box fluids, the RMS vibrational power in the horizontal and vertical directions, and the maximum-minimum vibration in the horizontal and vertical directions. These seven characteristics, as well as the pump tag, type of mechanical seal, date of seal installation, date of data acquisition, and leak information pertaining to the specific mechanical seal has been tabulated in Table 3. It should be noted that all of the pumps are run at 1750 RPM and pump water.

Pump	Seal Type	Seal Installed	Date Taken	Seal Life	Leak Info	Motor hp	ΔT [C]	Vert. Vib. RMS Pwr. [g]	Hor. Vib. RMS Pwr. [g]	Vert. Vib. Max-Min [g]	Hor. Vib. Max-Min [g]
DC-14a	AES CSS	02/01/99	06/02/99	0.33	-	10	6.205	0.072	0.058	0.516	0.434
DC-14a	AES CSS	02/01/99	07/08/99	0.43	-	10	4.103	0.097	0.074	0.746	0.541
DC-14b	Duro RO	12/01/94	05/19/99	4.47	-	10	5.337	0.154	0.117	1.206	0.757
DC-14b	Duro RO	12/01/94	06/02/99	4.50	-	10	4.950	0.093	0.081	0.722	0.629
DC-14b	Duro RO	12/01/94	07/08/99	4.60	-	10	3.255	0.113	0.083	0.851	0.638
DK-14a	AES CSS (LB)	01/01/99	05/19/99	0.38	-	20	-0.535	0.052	0.054	0.397	0.397
DK-14a	AES CSS (LB)	01/01/99	06/02/99	0.42	-	20	0.017	0.044	0.046	0.369	0.335
DK-14a	AES CSS (LB)	01/01/99	07/08/99	0.52	-	20	0.933	0.037	0.043	0.259	0.306
DK-14b	John Crane 715	12/01/94	06/02/99	4.50	-	20	5.290	0.109	0.106	0.759	0.690
DK-14b	John Crane 715	12/01/94	07/08/99	4.60	-	20	6.256	0.126	0.094	0.904	0.616
UK-10	John Crane 715	01/01/98	06/02/99	1.42	Lite Leak	5	8.579	0.326	0.279	2.651	2.048
UK-10	John Crane 715	01/01/98	07/08/99	1.52	Lite Leak	5	8.172	0.370	0.330	2.790	2.326
UK-11	Duro RO	12/01/94	06/02/99	4.50	Lite Leak	15	1.274	0.598	0.590	4.094	5.298
UK-11	Duro RO	12/01/94	07/08/99	4.60	Lite Leak	15	0.424	0.580	0.724	4.405	5.581
UK-16	John Crane 715	01/01/98	06/02/99	1.42	-	15	9.359	0.670	0.643	4.877	5.289
UK-16	John Crane 715	01/01/98	07/08/99	1.52	-	15	3.222	0.593	0.500	4.001	3.270
WK-10	Duro RO	06/01/96	06/02/99	3.00	-	5	2.034	0.681	0.681	4.962	4.162
WK-10	Duro RO	06/01/96	06/20/99	3.05	-	5	2.085	0.649	0.691	4.909	4.618
WK-10	Duro RO	06/01/96	07/08/99	3.10	-	5	0.525	0.697	0.641	4.916	4.826
WK-11	AES CSS	05/01/99	06/02/99	0.09	-	15	1.407	0.234	0.241	2.787	1.904
WK-11	AES CSS	05/01/99	06/20/99	0.14	-	15	0.966	0.209	0.274	2.730	2.292
WK-11	AES CSS	05/01/99	07/08/99	0.19	-	15	1.390	0.290	0.264	2.714	1.866
WK-16	AES CSS	05/01/99	06/02/99	0.09	-	15	1.441	0.713	0.602	5.379	4.866
WK-16	AES CSS	05/01/99	07/08/99	0.19	-	15	0.865	0.218	0.288	1.759	2.241

Table 3 - Data Acquisition Results

The first column, pump tag, is the serial number designation of the pump. This tag is indicative of the type of pump used and it's location in the TED facility. The second column reveals the type of mechanical seal used in the pump. As can be seen, there are four different mechanical seals used for these pumps: AES CSS, Duro RO, John Crane 715, and AES CSS LB (Large Bore). Pump DK-14a incorporates the only large bore mechanical seal, a newer seal type used in newer pump specifications. The large bore mechanical seal is installed in pumps that have an enlarged stuffing box in order to allow the compatibility of larger mechanical seal configurations. Most new pump designs incorporate this large bore design, but unfortunately for this study, only one of the pumps was manufactured in the past 15 years. As will be shown later, this type of mechanical seal has the potential to yield data that varies vastly from other mechanical seal designs.

The next three columns involve the age of the mechanical seal. The third column indicates the approximate date at which the mechanical seal was installed into the pump. The fourth column indicates the date the data was recorded from the pump, while the fifth column indicates the difference between these two dates in terms of years. The sixth column deals with the leakage history of the particular mechanical seal. From the ten pumps being currently monitored, two such pumps have shown signs of a small amount of leakage in the past. This leakage does not seem to be the result of a failed mechanical seal, for the two mechanical seals function adequately. This leakage could have resulted from a dry running of the pump, a pressure spike (hammer) in the flow, or several other possibilities. The next column on the table reveals the horsepower rating of the pump motor. It should be noted here that the pumps are run at four different horsepowers: 5, 10, 15, and 20 [hp].

The last five columns (8 through 12) involve the data that was collected from this list of ten pumps. The eighth column shows the average temperature difference between the inlet and stuffing box fluids. This information was obtained by recording the temperature of the inlet fluid and stuffing box fluids with the type-K thermocouple via the test plugs that were installed into the mechanical seals and pipe inlet flanges. These temperatures were recorded at 1000 [Hz], with 50 temperature points being averaged together to determine the average temperatures for the inlet and stuffing box fluids. The differences between these two temperatures (ΔT) were calculated to determine the temperature rise of the fluid in the stuffing box (area surrounding the mechanical seals). The fluid temperature difference is a more important parameter than the absolute flow temperature for several reasons. First, some of the pumps handle fluids that are vastly different in temperature from other pumps. Second, most of these pumps are located outside, so the piping is subject to the ambient temperature of the environment. These differences between fluid property and environmental temperature is a real measure of the energy added to the fluid because of the mechanical seals. As can be seen from the data from pump DK-14a though, these temperature differences are not always predictable. Pump DK-14a not only incorporates an enlarged stuffing box (which allows the transfer of heat away from the mechanical seals more efficiently), but also insulation around the inlet piping. Due to the combination of insulated piping, ambient weather conditions, and an enlarged stuffing box, the first recorded temperature difference on May 19, 1999 indicated that the fluid in the stuffing box was cooler than the inlet fluid. Such physical differences in pump design can thus vastly alter the "normal" temperature response of the system.

The last four columns in the table involve the vibration of the mechanical seal. These vibrations were determined by placing piezo-electric accelerometer directly onto the mechanical seal, in the horizontal and vertical planes. The measurements from this accelerometer were then recorded at a rate of 1000 [Hz] and analyzed. The three meaningful ways to interpret vibration data involve the RMS power of the vibration signal, the maximum – minimum amplitude of the vibration signal, and the analysis of the vibration data in the frequency domain. The ninth and tenth columns in Table 3 indicate the RMS power of the vibration signal in the horizontal and vertical planes, while the eleventh and twelfth columns indicate the maximum – minimum vibrational amplitude in these two planes. The information from the frequency domain analysis of the vibration data was not incorporated into this table due to the complexity of the information.

Such frequency domain analyses of the vibration data were obtained by performing a fast Fourier transformation of the data and then converting this information into a power spectral density plot. These power spectral density (PSD) plots determine at which frequencies the vibrational data has the most power. These PSD plots can be compared between different pumps to determine if different dominant frequencies exist between them. Certain dominant frequencies indicate certain known faults with the operation of rotating machinery. As can be seen in Figure 37, the PSD plot of this particular pump indicates a high power at 148 [Hz]. This frequency corresponds to the vane pass frequency of the impeller on the pump. The pumps are run at 1750 RPM (29.17 [Hz]) and contain impellers that incorporate five vanes. Therefore, the vane pass frequency should be approximately 146 [Hz], assuming that the pumps are run at exactly 1750 RPM.

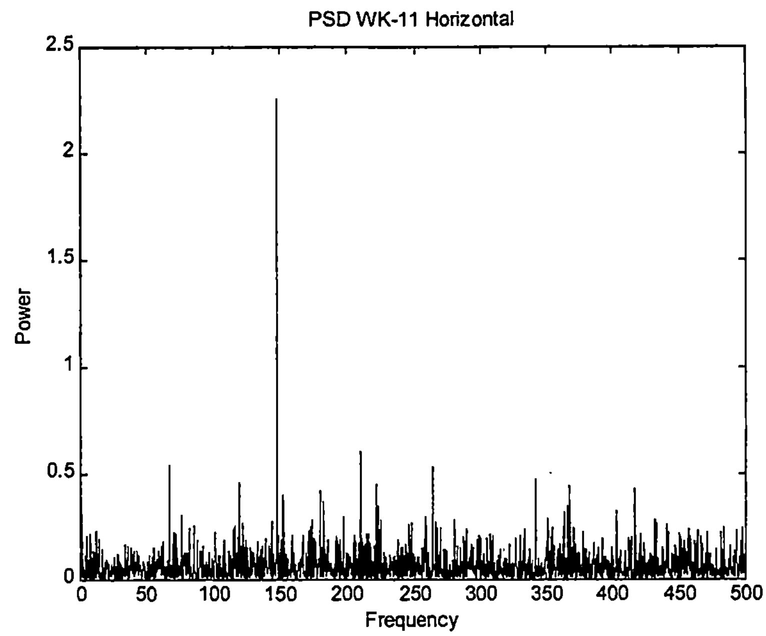


Figure 37 – PSD Plot

A complete survey of all the PSD plots obtained from the ten pumps over the course of their testing can be found in the Appendix. It should also be noted here that in common practice, it is not recommended that rotational machinery (such as pumps) be operated in environments where they are subjected to a maximum – minimum vibration of more than 2.0 [g]. As can be seen from the tabulated data, approximately six of the ten pumps operate under such circumstances. Accordingly, these same six pumps yield PSD plots that indicate significantly more power in many more frequencies than the other four pumps. As the maximum – minimum vibration increases over 2.0 [g], the PSD plots increase accordingly. The previous PSD plot involved pump WK-11, which yielded a maximum – minimum vibration of 1.866 [g]. Figure 38 below shows the PSD plot of pump WK-10, which has a maximum – minimum vibration of 4.826 [g].

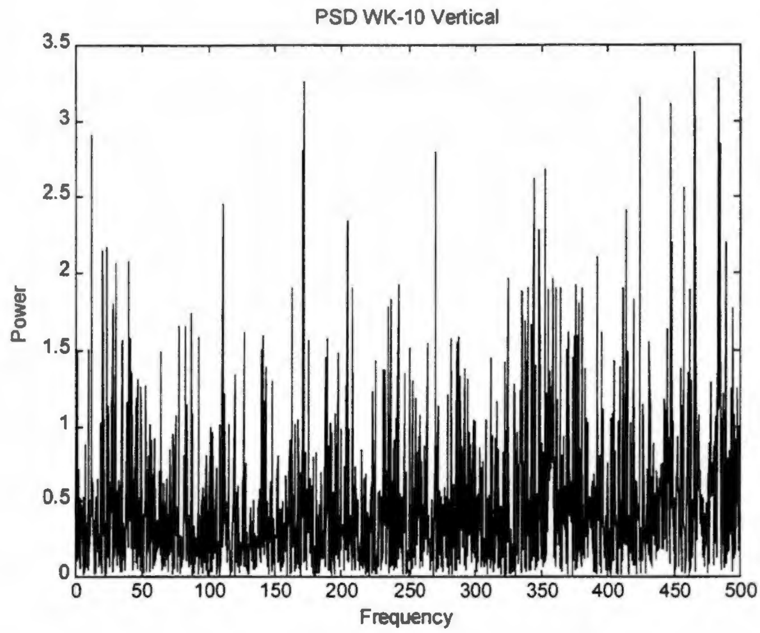


Figure 38 – Erratic PSD Plot

All such PSD plots from pumps with high overall amplitudes of vibration yield similar results. The lack of definite dominant frequencies combined with the overall increase in power would most likely indicate noise in the data. Unfortunately though, this is most likely not the case. As can be seen from Table 3, the consistency of the excessive amplitudes is obvious. This consistency, accompanied by the fact that some of the pumps that are generating these unusually high vibration amplitudes are located next to pumps that appear to be functioning within acceptable conditions, rules out the possibility of electrical noise associated with the location of the pumps. One possible explanation for these erratic signals could be the result of turbulent flow in the fluid.

Linear Regression Analysis

After the data was collected, a linear regression was performed on six of the parameters to determine how successfully the seventh parameter could be predicted from the other six. This method of linear regression would determine how correlated the sets of data were to one another. In the linear regression performed, the age of the mechanical seal was predicted based on the other six parameters. First, the data was divided into two groups, a training group and a testing group. Each group contained at least one data set from each mechanical seal. Next, the linear regression algorithm was performed on all six variables, then each variable independently, and finally with combinations of any two variables. The ages of the mechanical seals ranged from 0.09 – 4.6 years, with an average age of 1.94 years old. When the linear regression was performed with all six variables, the average error of the test set for predicting the age of the mechanical seal was 1.44 years. When this regression technique was limited to the most significant variable, the average error of the test set was 1.64 years. This singificant variable was the horsepower of the motor. When a combination of two variables was used in the test set though, the average error of the age was reduced slightly to 1.53 years. This was accomplished by incorporating the horizontal and vertical RMS vibrational of the seal. It should also be noted that when the age of the mechanical seal was predicted solely on the average of the training set, an average error of 1.68 years was attained. This method of linear regression indicates that from the preliminary data, the age of the mechanical seal cannot be effectively predicted from the other six variables. Although this technique does not relate directly to mechanical seal performance, it implies that other methods might need to be incorporated in order to accurately predict premature mechanical seal failure.

Theoretical Artificial Intelligence Performance

Since there was not enough valuable information obtained from the data acquisition process performed on the ten test pumps to adequately train artificial intelligence architectures, the four different AI architectures had to be trained and tested with theoretical data. This theoretical data was however based on the information obtained from the actual data. For instance, the typical ranges set for the seven pump characteristics were dictated by the actual data. These data sets were used to train three different neural network systems as well as program one expert system. From this process, the different AI architectures could be tested over similar data to determine how effectively they were able to classify premature mechanical seal failure.

Before the AI architectures were trained, a linear regression was performed on the theoretical data set to serve as a benchmark for classification performance. Linear regression techniques represent the most basic method of prediction. When the theoertical data set was subjected to the linear regression technique, the technique was able to properly classify the data 77.12% of the time, which corresponds to a 22.88% error. From this classification technique, the performance of the various AI architectures can be effectively compared

The first AI architecture trained and tested was the multi-layer perceptron neural network (MLP). This ANN proved most successful at predicting the generalized data set when it contained 15 neurons in the hidden layer. This particular configuration took 29 epochs to adequately train the ANN to a mean squared error (MSE) of 10% of the original scaled data set. This MSE equals 0.0361 ($[0.1 \times (0.95 - {}^{-}0.95)]^2$) and corresponds to a sum of squared error (SSE) of 18.772 (0.0361 x 520). With this configuration of 15 hidden layer neurons and a 10% MSE, the MLP was able to successfully classify the training data with a 7.69% error and the generalized test data with an 18.08% error. Other configurations of MLP were tested, but they were not able to generalize the test data as well as the previously mentioned configuration. Table 4 shows the training history of the various MLP neural networks.

# hidden neurons	MSE %	MSE	SSE	training epochs	training data %	test data %	out of range data %
15	7%	0.0177	9.1983	53	2.88%	26.34%	0.00%
18	7%	0.0177	9.1983	42	3.27%	26.35%	0.00%
20	7%	0.0177	9.1983	44	2.70%	25.96%	0.00%
25	7%	0.0177	9.1983	41	2.50%	25.57%	0.00%
50	7%	0.0177	9.1983	26	2.69%	25.00%	0.00%
15	10%	0.0361	18.7720	29	7.69%	18.08%	0.00%
18	10%	0.0361	18.7720	26	7.50%	19.81%	0.00%
20	10%	0.0361	18.7720	50	6.35%	18.63%	0.00%
25	10%	0.0361	18.7720	12	6.92%	19.23%	0.00%
50	10%	0.0361	18.7720	13	6.39%	18.65%	0.00%

Table 4 – MLP Training History

As can be seen from this table, the test data could only be classified with a minimum of 18.08% error, and this AI architecture is unable to determine when the input data lies outside of its trained region. By increasing the percent error tolerable from 7% to 10%, the networks were able to generalize the information better. This mean that the network was better able to model the trends that the data represented as opposed to the data itself.

After the MLP was trained and tested, the RBFN was subjected to similar trials. Since the number of hidden radbas neurons in the RBFN is determined from an unsupervised learning algorithm, the spread constant of the Gaussian activation functions along with an acceptable error goal are required to train the networks. The RBFN most adept at classifying the generalized data set contained 247 radbas hidden neurons, a spread constant of 0.20, and a MSE of 0.0361 (10%). This RBFN was able to properly classify the test data with an 11.9% error, while classifying the training data with a 7.11% error. An advantage that this type of AI architecture has over other forms is its ability to recognize data that does not conform to any of the classification patterns for which it was trained. Although not perfect, the RBFN were able to recognize data that lied outside of the trained region approximately 46% of the time. This value would undoubtedly increase if the input data sets varied more from the trained data sets. Table 5 shows the training history for the RBFN.

spread	MSE %	MSE	SSE	# radbas neurons	training data %	test data %	out of range data %
0.20	7%	0.0177	9.1983	247	2.50%	14.23%	46.00%
0.20	10%	0.0361	18.7720	180	7.11%	11.90%	46.00%
0.25	5%	0.0090	4.6930	286	0.00%	20.57%	30.00%
0.25	7%	0.0177	9.1983	232	2.69%	16.15%	46.00%
0.25	10%	0.0361	18.7720	160	8.27%	17.30%	50.00%

Table 5 – RBFN Training History

Similar to the performance of the MLP, as the MSE is decreased, the ability of the ANN to classify the test data increases while the ability of the ANN to classify generalized data decreases. It is due to this characteristic of ANN that the MSE of 0.0361 (10%) generalizes better than architectures which incorporate smaller error goals.

The last ANN trained and tested was the probabilistic neural network (PNN). This ANN, unlike the others examined, determines a statistical basis for classifying the data sets. This statistical basis means that no error goal has to be specified for the ANN to successfully train, for the ANN will automatically train to the best error goal possible based on the training data set. The PNN used for this research was able to successfully classify the trained data with an average error of 6.73% and the test data with an average error of 11.15%. This AI architecture was unable to determine when a data set lied outside of the trained region though. It simply determined a classification based on the statistical probability of belonging to certain classes. As stated before, this PNN required a pseudo-LVQ algorithm in order to be implemented properly. A similar pseudo-LVQ algorithm would have to be designed based on the actual data in order for this method of classification to be utilized.

After the ANN classification systems were designed, trained, and tested, a fuzzy logic system was designed. As stated before, this FLS utilized singleton, triangular, and Gaussian membership functions to divide the various universes of discourse and define the fuzzy input and output parameters. Unlike the ANN, the FLS did not require a training set of data, but a fuzzy rule set governing the behavior of the system. This inherent difference between the two systems made a comparison between the systems difficult. After the FLS was designed with a rule set that mimicked the theoretical data sets used for the ANN training, it was tested with a theoretical data set. This classification system however did not directly classify the performance of the mechanical seals into three categories. Instead, the FLS scaled the output performance of the mechanical seal onto a range of 0.00 – 1.00 (representing bad – good performance). For instance, when the input parameters were a 10 [hp] motor, a seal age of 2.2 [years], a ΔT of 1.4 [C], an RMS vertical of 0.12 [g], an RMS horizontal of 0.16[g], a max-min vertical of 0.43, and a max-min horizontal of 0.52, the effective seal performance was 0.929. This output value of 0.929 corresponds to a very well performing mechanical seal, which corresponds to the theoretical data set describing a "good" seal and the fuzzy rule set for a "good" seal. This output effective performance can be seen in Figure 39.

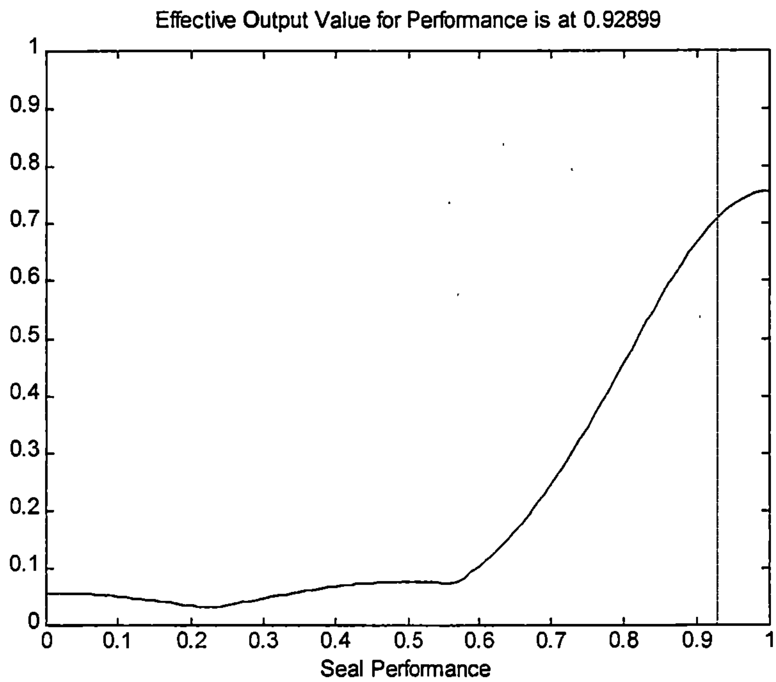


Figure 39 – FLS Output Performance

It should be noted here that this is an effective output response for the FLS. Since a centroid de-fuzzification method with Gaussian output membership functions was used to interpret the crisp output value of the system, values equal to exactly 0.00 or 1.00 could not be generated. In fact, with this particular FLS, the minimum attainable value was 0.1564 and the maximum attainable crisp output was 0.8222. The crisp outputs were therefore scaled to allow the minimum and maximum crisp outputs to be 0.00 and 1.00 respectively. It should also be noted that since there were many more rule sets describing "bad" seals compared to "good" seals (57 to 8), the max operator was used in the aggregation process to give an equal value to the "good" and "bad" rule sets. If the sum operator had been used, then the unequal number of rule sets would have given more importance to the "bad" seal characteristics. Also, the product operator was used to determine the consequents of the fuzzy rules from the antecedents. This was done as to avoid losing any of the information concerning the fuzzy rules. This combination of fuzzy operators and fuzzy rules allowed for a very powerful method of classifying the performance of mechanical seals. When this FLS was used to classify the same data that was used to test the three NN architectures and the linear regression technique, it performed moderately. This FLS correctly classified 76.73% of the data, which corresponds to a 23.27% error. This large error can be attributed to a lack of an expert knowledge of the system. The programs used to design, train, and test the four different AI architectures are given in the Appendix.

CHAPTER IX

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE STUDY

Although there was only a limited amount of data collected from the ten test pumps, several conclusions can be made. The first conclusion in this research involves the determination of the principle parameters of the mechanical seal that should be monitored. These parameters were chosen based on the known and perceived symptoms and results of mechanical seal failure. For future studies, this list of five measured physical parameters (temperature difference between the inlet and stuffing box fluid, RMS power of the vibration signal in the horizontal and vertical planes, and maximum – minimum vibrational amplitude in the horizontal and vertical planes) should be increased. This increased list of measured parameters should include a tachometer reading of the rotating shaft to determine the actual rotating speed of the impeller as well as pressure transducer readings from the stuffing box and inlet and outlet fluids. These pressure readings will determine if the mechanical seal is operating under designed conditions and if any pressure spikes occur in the flow. Pressure fluctuations in the stuffing box could also indicate an inadequate flushness of the seal faces.

The flushness of the seal faces could also be determined by monitoring the springs on the backside of the stationary seal face. By comparing the dynamic compression of the springs located on opposite sides of the seal face, the imbalance of force acting on the seal faces could be determined. Research has also been performed that monitors the thin layer of lubricating fluid between the seal faces by means of an active ultrasonic transmission technique[21]. This technique measures the frequency of reflected ultrasonic waves to determine the thickness of the lubricating layer. This information would also prove to be a powerful indicator for premature seal failure. Regardless of the new physical parameters monitored, the same non-invasive strategy must be implemented to insure the ease of data acquisition. Up to this point, nothing had to be permanently altered with the operation of the mechanical seal, and data could be taken without the interruption of the pump's operation.

Along with the addition of physical parameters that should be measured, additional filtering might be required. The addition of an anti-aliasing filter to the accelerometer will prevent the mirroring effect onto the PSD plot that can result from frequencies that occur beyond that limited by the Nyquist criteria. For this research, any vibration that occurred at a frequency greater than 500 [Hz] would be reflected back onto the range of 0 – 500 [Hz]. This reflected information is indistinguishable from the actual vibrational data after the signal has been recorded.

After the physical parameters of the pumps were measured, several conclusions regarding this data can be made. There is a definite correlation between excessive maximum – minimum vibrational amplitude and erratic PSD plots. Although this correlation could be the result of a large amount of random noise in the acquisition system, the repeatability and selectivity of these measurements suggests otherwise. The two PSD plots shown earlier, Figures 37-38, show distinctly different frequency patterns for the pumps, yet the two pumps illustrated are located directly next to one another. The close proximity and relatively consistent maximum – minimum vibrational amplitudes recorded from these two pumps indicates general patterns of behavior as opposed to a random interference of the signals. As stated previously though, these signals could be the result of an excessive amount of turbulence in the flow. The pipe configuration, inner pipe wall roughness, or improper flow conditions could cause such turbulent flow.

These PSD plots, when generated with the data from pumps with small (less than 2.0 [g]) vibrational amplitudes, do reveal the general rotational behaviors of the systems. Many of the PSD plots indicate a strong (if not maximum) power at the rotational frequency of the impeller, while other plots indicate such phenomenon as the vane pass frequency. Further study is required to correlate each of the dominant frequencies in the PSD plots to known faults with mechanical seals though.

It also became apparent that monitoring pumps rated at four different powers with four different mechanical seals might not be the best means for predicting premature mechanical seal failure for a specific pump. It is true that in order to model the general behavior of centrifugal pumps; information obtained from many different pumps is necessary. The fault with this method of prediction lies in the inherent differences between the different power ratings and different mechanical seals. It might prove more successful to base prediction systems on pumps with a specific power rating or a specific mechanical seal instead of grouping all of the information into one prediction method. This would allow the prediction systems to focus on the differences of the measured physical parameters as opposed to the differences in the type and rating of the pump.

When enough data has been collected from differing pumps in various stages of performance, perhaps a regression technique such as linear regression or principle component analysis should be performed on the data. This regression analysis would determine how relevantly each of the measured parameters correlated to the performance of the mechanical seal. By excluding the non-correlated parameters, a prediction system based on a reduced number of parameters could be designed.

If the theoretical data that was generated to train and test the four different AI architectures was truly indicative of the performance of mechanical seals, then mechanical seal failure could be predicted approximately 88% of the time. This would be based on a radial basis function network. The RBFN was not only able to classify the generalized data correctly 88% of the time, but it was able to recognize when as input data set could not be classified accurately. Although the MLP and PNN classified all of the data that it was presented, they were unable to determine when data that the ANN had not been trained with was used. The MLP and PNN both classified the data to the best extent that they could, which does not mean that the classifications were wrong, but that they were unpredictable. The FLS, on the other hand, not only could determine if the seal was failing or not, but could give a scaled output corresponding to the seal performance. This was accomplished with a scaling technique that assigned a mechanical seal a value corresponding to how "good" or "bad" the seal was performing. The main drawback of this technique involves the expert knowledge required of the system. Until a more complex understanding of the collected data is achieved, this expert knowledge will be based on the collected data. Basing the expert knowledge on the data therefore weakens the effectiveness of the FLS, for the expert knowledge regarding seal failure was attained from the relationships perceived from the data. These relationships can be much more accurately modeled with ANN systems, such as the MLP, RBFN, or PNN. The exploration of these various AI architectures shows how they are more adept at classifying the mechanical seal failure than standard linear regression techniques. Table 6 shows the comparisons between the linear regression technique and the four AI architectures.

	Linear Regression	MLP	RBFN	PNN	FLS
Correctly Classified	77.12%	81.92%	88.10%	88.85%	76.73%
out of bounds determined	NO	NO	YES	NO	NO

Table 6 – Classification Comparison

Despite the method of prediction used, more information is required pertaining to premature mechanical seal failure. This information can be obtained three ways. The first means of acquiring data has already been initiated, and continues with the monitoring of the ten test pumps at the TED facility. The second means of acquiring data involves establishing a test flow loop for a centrifugal pump. This test loop can be manipulated to generate faulted mechanical seals. The information obtained from such laboratory testing could very effectively correlate trends in data to specific mechanical seal behaviors. This correlation could then be combined with the information obtained from the ten test pumps to help determine the performance of the pumps at the TED facility. The last means of acquiring data involves expanding the expert knowledge relating to mechanical seal failure. This can be accomplished by combining the information obtained from the test flow loop laboratory testing with the information obtained from the visual inspection of failed mechanical seals at TED and in the laboratory. The visual inspection of the mechanical seals provides the final link in connecting the experimental testing of pumps at TED with the controlled fault induced laboratory testing. The combination of all of this information would not only establish a prediction system that could determine when a mechanical seal would fail, but could determine the reasons for the failure. This expert knowledge could then be used to program an expert system similar to the FLS system programmed before. This knowledge could effectively reduce the number of future mechanical seal failures.

In conclusion, the research presented here represents the first phase of a project intended to predict the premature failure of mechanical seals in centrifugal pumps. As shown, a considerable number of steps have been taken to establish an effective method for this prediction. Principle parameters relating to seal failure were chosen and monitored on a group of ten test pumps over a period of several months. Four different theoretical artificial intelligence architectures were designed and tested to show how effectively the performance of a mechanical seal could be predicted. Lastly, a long term and robust method for data and information acquisition was discussed. It is the authors deepest hope that the information presented here will form the basis of a comprehensive method for predicting mechanical seal failure in centrifugal pumps.

References

1. Greene, Arthur, *Pumping Machinery*, Second Edition, John Wiley & Sons, Inc., 1919.
2. Carter, Roy and Igor Karassik, *Pump Questions And Answers*, McGraw Hill Book Company, 1949.
3. INTECH, *Mechanical Seal Technology and Installation*, 1993.
4. Matley, Jay, *Fluid Movers*, McGraw Hill Publications Co, 1979.
5. Burke, W., "Extending Seal and Pump Maintenance Intervals," *Plant Engineering*, July 10, 1995, pp 83-86.
6. Meneganti, M and Francesco Saviello, "Fuzzy Neural Networks for Classification and Detection of Anomalies," *IEEE Transactions on Neural Networks*, vol. 9 no. 5, Sept. 1998, pp 848-860.
7. Amjady, Nima and Mehdi Ehsan, "Evaluation of Power Systems Reliability by an Artificial Neural Network," *IEEE Transactions on Power Systems*, vol. 14 no. 1, Feb. 1999, pp 287-292.
8. Song, Xin-Hua and Philip Hopke, "Classification of Single Particles Analyzed by ATOFMS Using an Artificial Neural Network, ART-2A," *Analytical Chemistry*, vol. 71 no. 4, Feb. 1999, pp 860-865.
9. Moon, Hyeong-Soon and Suck-Joo Na, "Optimum Design Based on Mathematical Model and Neural Network to Predict Weld Parameters for Fillet Joints," *Journal of Manufacturing Systems*, vol. 16 no. 1, 1997, pp 13-23.

10. Uhrig, Robert, "Artificial Neural Networks in Nuclear Power Plants," *Nuclear News*, July 1994, pp 38-40.

11. Raghu, P. and B. Yegnanarayana, "Supervised Texture Classification Using a Probabilistic Neural Network and Constraint Satisfaction Model," *IEEE Transactions on Neural Networks*, vol. 9 no. 3, May 1998, pp 516-522.

12. Kou, R.J. and P.H. Cohen, "Multi-Sensor Integration for On-line Tool Wear Estimation Through Radial Basis Function Networks and Fuzzy Neural Network," *Neural Networks*, vol. 12 no. 2, March 1999, pp 355-369.

13. Taylor, Jeff, "Data, Questions, and Thoughts Concerning the Reliability Issues of 33 Dowtherm Pumps, Floor 5, Building 255," report submitted to Kevin Bellamy at Tennessee Eastman Division of Eastman Chemical Company on June 10, 1998.

14. Nepa, W. A. and R. J. Urwin, "Centrifugal Pump Failure Analysis," *Plant Engineering*, Sept. 23, 1993, pp 56-58.

15. Foszcz, J., "Increasing Centrifugal Pump Reliability," *Plant Engineering*, June 1998, pp 58-64.

16. Mechanical Seal Handbook, *Fluid Sealing Association*, 1991.

17. Omega Engineering, *The Pressure Strain And Force Handbook*, Vol. 29, 1995.

18. Tsoukalas, Lefteri and Robert Uhrig, *Fuzzy and Neural Approaches in Engineering*, John Wiley and Sons, 1997.

19. Think Quest, Internet site, *An Introduction to Artificial Intelligence* http://library.advanced.org/2705/basics.html 6-16-99.

20. Hines, J. Wesley, *MATLAB SUPPLEMENT TO: Fuzzy and Neural Approaches in Engineering,* John Wiley and Sons, 1997.

21. Anderson, W. and Richard Salant, "*Ultrasonic Detection of Lubricating Film Collapse in Mechanical Seals,*" Presented at the 54th annual meeting of the Society of Tribologists and Lubricating Engineers, May 23-27 1999.

APPENDIX

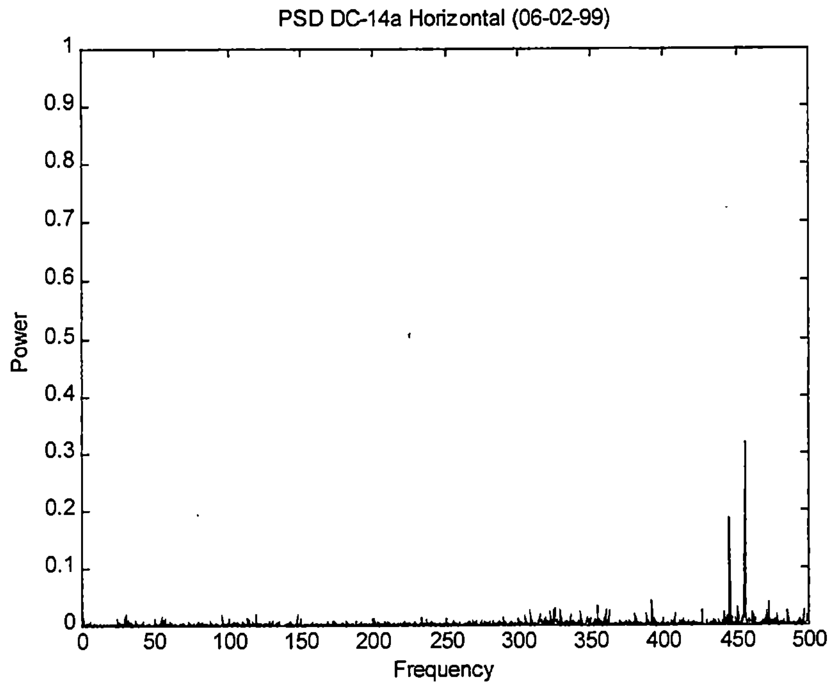
Table A1 – Theoretical Data Set

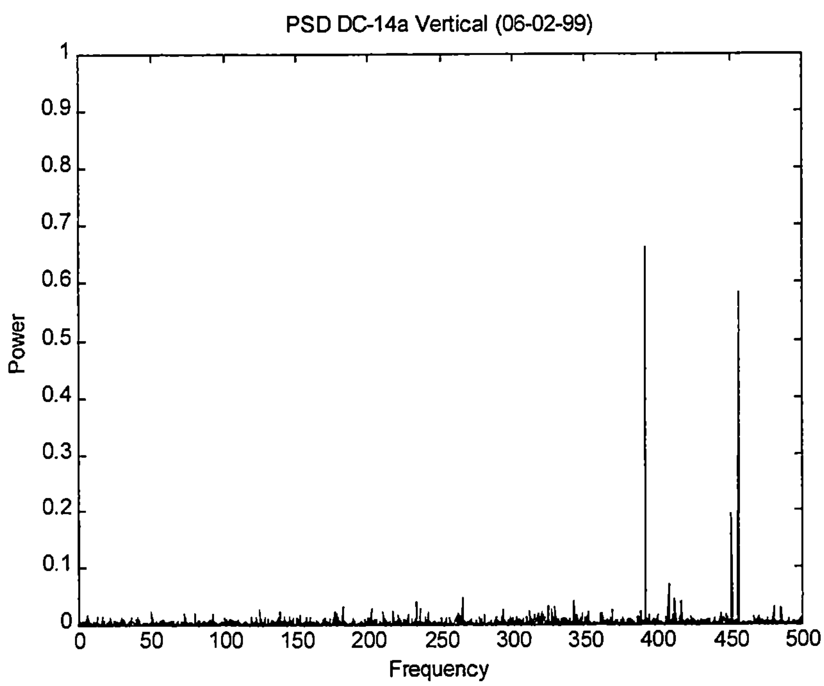
seal condition	Motor Power	age [years]	Δt [C]	Power Hor. [g]	Power Ver. [g]	Max-Min Hor. [g]	Max-Min Ver. [g]
good	5	0-5	0.5-3.5	0-0.2	0-0.2	0-1	0-1
good	5	5-10	0.5-3.5	0-0.2	0-0.2	0-1	0-1
good	10	0-5	0.5-3.5	0-0.3	0-0.3	0-2	0-2
good	10	5-10	0.5-3.5	0-0.3	0-0.3	0-2	0-2
good	15	0-5	0.5-3.5	0-0.4	0-0.4	0-3	0-3
good	15	5-10	0.5-3.5	0-0.4	0-0.4	0-3	0-3
good	20	0-5	0.5-3.5	0-0.5	0-0.5	0-4	0-4
good	20	5-10	0.5-3.5	0-0.5	0-0.5	0-4	0-4
ok	5	0-5	0.5-3.5	0.2-0.4	0-0.2	0-1	0-1
ok	5	0-5	0.5-3.5	0-0.2	0.2-0.4	0-1	0-1
ok	5	0-5	0.5-3.5	0-0.2	0-0.2	0-1	0-1
ok	5	0-5	0.5-3.5	0-0.2	0-0.2	1-2	0-1
ok	5	0-5	3.5-6.5	0-0.2	0-0.2	0-1	1-2
ok	5	5-10	0.5-3.5	0.2-0.5	0-0.2	0-1	0-1
ok	5	5-10	0.5-3.5	0-0.2	0.2-0.5	0-1	0-1
ok	5	5-10	0.5-3.5	0-0.2	0-0.2	1-3	0-1
ok	5	5-10	0.5-3.5	0-0.2	0-0.2	0-1	1-3
ok	5	5-10	3.5-6.5	0-0.2	0-0.2	0-1	0-1
ok	10	0-5	0.5-3.5	0.3-0.5	0-0.3	0-2	0-2
ok	10	0-5	0.5-3.5	0-0.3	0.3-0.5	0-2	0-2
ok	10	0-5	0.5-3.5	0-0.3	0-0.3	2-3	0-2
ok	10	0-5	0.5-3.5	0-0.3	0-0.3	0-2	2-3
ok	10	0-5	3.5-6.5	0-0.3	0-0.3	0-2	0-2
ok	10	5-10	0.5-3.5	0.3-0.6	0-0.3	0-2	0-2
ok	10	5-10	0.5-3.5	0-0.3	0.3-0.6	0-2	0-2
ok	10	5-10	0.5-3.5	0-0.3	0-0.3	2-4	0-2
ok	10	5-10	0.5-3.5	0-0.3	0-0.3	0-2	2-4
ok	10	5-10	3.5-6.5	0-0.3	0-0.3	0-2	0-2
ok	15	0-5	0.5-3.5	0.4-0.6	0-0.4	0-3	0-3
ok	15	0-5	0.5-3.5	0-0.4	0.4-0.6	0-3	0-3
ok	15	0-5	0.5-3.5	0-0.4	0-0.4	3-4	0-3
ok	15	0-5	0.5-3.5	0-0.4	0-0.4	0-3	3-4

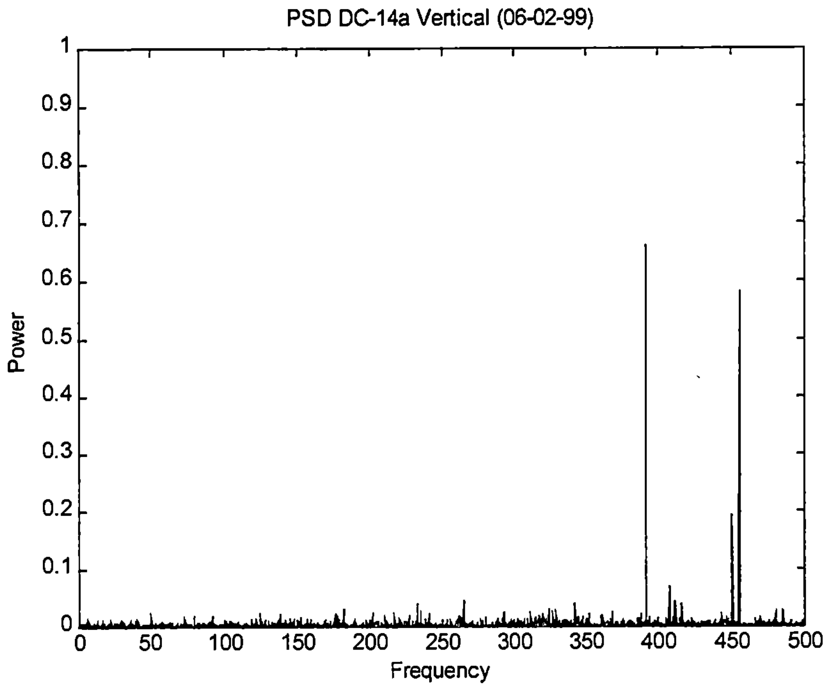
ok	15	0-5	3.5-6.5	0-0.4	0-0.4	0-3	0-3
ok	15	5-10	0.5-3.5	0.4-0.7	0-0.4	0-3	0-3
ok	15	5-10	0.5-3.5	0-0.4	0.4-0.7	0-3	0-3
ok	15	5-10	0.5-3.5	0-0.4	0-0.4	3-5	0-3
ok	15	5-10	0.5-3.5	0-0.4	0-0.4	0-3	3-5
ok	15	5-10	3.5-6.5	0-0.4	0-0.4	0-3	0-3
ok	20	0-5	0.5-3.5	0.5-0.7	0-0.5	0-4	0-4
ok	20	0-5	0.5-3.5	0-0.5	0.5-0.7	0-4	0-4
ok	20	0-5	0.5-3.5	0-0.5	0-0.5	4-5	0-4
ok	20	0-5	0.5-3.5	0-0.5	0-0.5	0-4	4-5
ok	20	0-5	3.5-6.5	0-0.5	0-0.5	0-4	0-4
ok	20	5-10	0.5-3.5	0.5-0.8	0-0.5	0-4	0-4
ok	20	5-10	0.5-3.5	0-0.5	0.5-0.8	0-4	0-4
ok	20	5-10	0.5-3.5	0-0.5	0-0.5	4-6	0-4
ok	20	5-10	0.5-3.5	0-0.5	0-0.5	0-4	4-6
ok	20	5-10	3.5-6.5	0-0.5	0-0.5	0-4	0-4
bad	5	0-5	0.5-3.5	0.2-1.0	0.2-1.0	0-1	0-1
bad	5	0-5	0.5-3.5	0-0.2	0-0.2	1-8	1-8
bad	5	0-5	0.5-3.5	0.5-1.0	0-0.2	0-1	0-1
bad	5	0-5	0.5-3.5	0-0.2	0.5-1.0	0-1	0-1
bad	5	0-5	0.5-3.5	0-0.2	0-0.2	3-8	0-1
bad	5	0-5	0.5-3.5	0-0.2	0-0.2	0-1	3-8
bad	5	0-5	6.5-9.5	0-0.2	0-0.2	0-1	0-1
bad	5	5-10	0.5-3.5	0.2-1.0	0.2-1.0	0-1	0-1
bad	5	5-10	0.5-3.5	0-0.2	0-0.2	1-8	1-8
bad	5	5-10	0.5-3.5	0.6-1.0	0-0.2	0-1	0-1
bad	5	5-10	0.5-3.5	0-0.2	0.6-1.0	0-1	0-1
bad	5	5-10	0.5-3.5	0-0.2	0-0.2	4-8	0-1
bad	5	5-10	0.5-3.5	0-0.2	0-0.2	0-1	4-8
bad	5	5-10	6.5-9.5	0-0.2	0-0.2	0-1	0-1
bad	10	0-5	0.5-3.5	0.3-1.0	0.3-1.0	0-2	0-2
bad	10	0-5	0.5-3.5	0-0.3	0-0.3	2-8	2-8
bad	10	0-5	0.5-3.5	0.6-1.0	0-0.3	0-2	0-2
bad	10	0-5	0.5-3.5	0-0.3	0.6-1.0	0-2	0-2
bad	10	0-5	0.5-3.5	0-0.3	0-0.3	4-8	0-2
bad	10	0-5	0.5-3.5	0-0.3	0-0.3	0-2	4-8

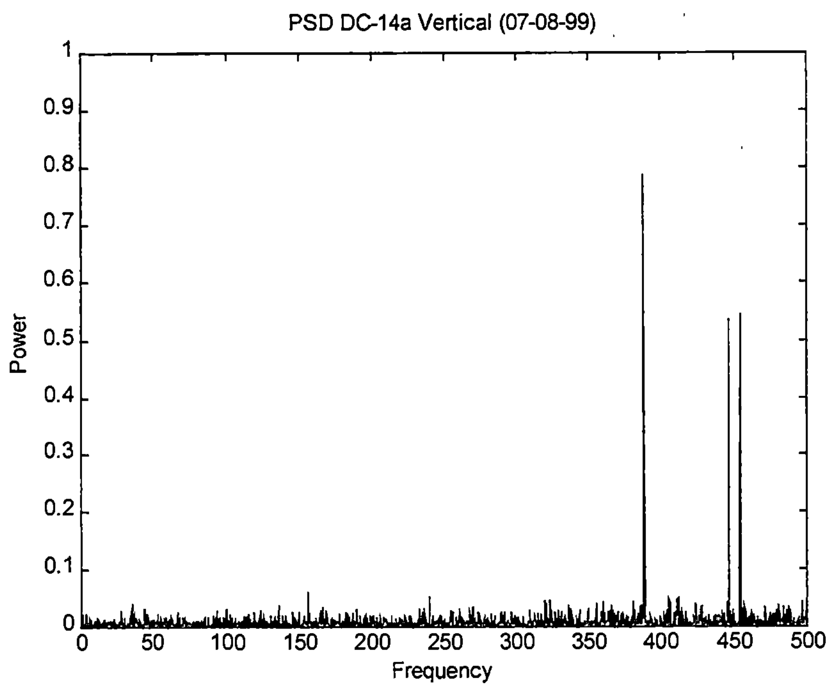
bad	10	0-5	6.5-9.5	0-0.3	0-0.3	0-2	0-2
bad	10	5-10	0.5-3.5	0.3-1.0	0.3-1.0	0-2	0-2
bad	10	5-10	0.5-3.5	0-0.3	0-0.3	2-8	2-8
bad	10	5-10	0.5-3.5	0.7-1.0	0-0.3	0-2	0-2
bad	10	5-10	0.5-3.5	0-0.3	0.7-1.0	0-2	0-2
bad	10	5-10	0.5-3.5	0-0.3	0-0.3	5-8	0-2
bad	10	5-10	0.5-3.5	0-0.3	0-0.3	0-2	5-8
bad	10	5-10	6.5-9.5	0-0.3	0-0.3	0-2	0-2
bad	15	0-5	0.5-3.5	0.4-1.0	0.4-1.0	0-3	0-3
bad	15	0-5	0.5-3.5	0-0.4	0-0.4	3-8	3-8
bad	15	0-5	0.5-3.5	0.7-1.0	0-0.4	0-3	0-3
bad	15	0-5	0.5-3.5	0-0.4	0.7-1.0	0-3	0-3
bad	15	0-5	0.5-3.5	0-0.4	0-0.4	5-8	0-3
bad	15	0-5	0.5-3.5	0-0.4	0-0.4	0-3	5-8
bad	15	0-5	6.5-9.5	0-0.4	0-0.4	0-3	0-3
bad	15	5-10	0.5-3.5	0.4-1.0	0.4-1.0	0-3	0-3
bad	15	5-10	0.5-3.5	0-0.4	0-0.4	3-8	3-8
bad	15	5-10	0.5-3.5	0.8-1.0	0-0.4	0-3	0-3
bad	15	5-10	0.5-3.5	0-0.4	0.8-1.0	0-3	0-3
bad	15	5-10	0.5-3.5	0-0.4	0-0.4	6-8	0-3
bad	15	5-10	0.5-3.5	0-0.4	0-0.4	0-3	6-8
bad	15	5-10	6.5-9.5	0-0.4	0-0.4	0-3	0-3
bad	20	0-5	0.5-3.5	0.5-1.0	0.5-1.0	0-4	0-4
bad	20	0-5	0.5-3.5	0-0.5	0-0.5	4-8	4-8
bad	20	0-5	0.5-3.5	0.8-1.0	0-0.5	0-4	0-4
bad	20	0-5	0.5-3.5	0-0.5	0.8-1.0	0-4	0-4
bad	20	0-5	0.5-3.5	0-0.5	0-0.5	6-8	0-4
bad	20	0-5	0.5-3.5	0-0.5	0-0.5	0-4	6-8
bad	20	0-5	6.5-9.5	0-0.5	0-0.5	0-4	0-4
bad	20	5-10	0.5-3.5	0.5-1.0	0.5-1.0	0-4	0-4
bad	20	5-10	0.5-3.5	0-0.5	0-05	4-8	4-8
bad	20	5-10	0.5-3.5	0.9-1.0	0-0.5	0-4	0-4
bad	20	5-10	0.5-3.5	0-0.5	0.9-1.0	0-4	0-4
bad	20	5-10	0.5-3.5	0-0.5	0-0.5	7-8	0-4
bad	20	5-10	0.5-3.5	0-0.5	0-0.5	0-4	7-8
bad	20	5-10	6.5-9.5	0-0.5	0-0.5	0-4	0-4

PSD Plots for MRC Test Pumps

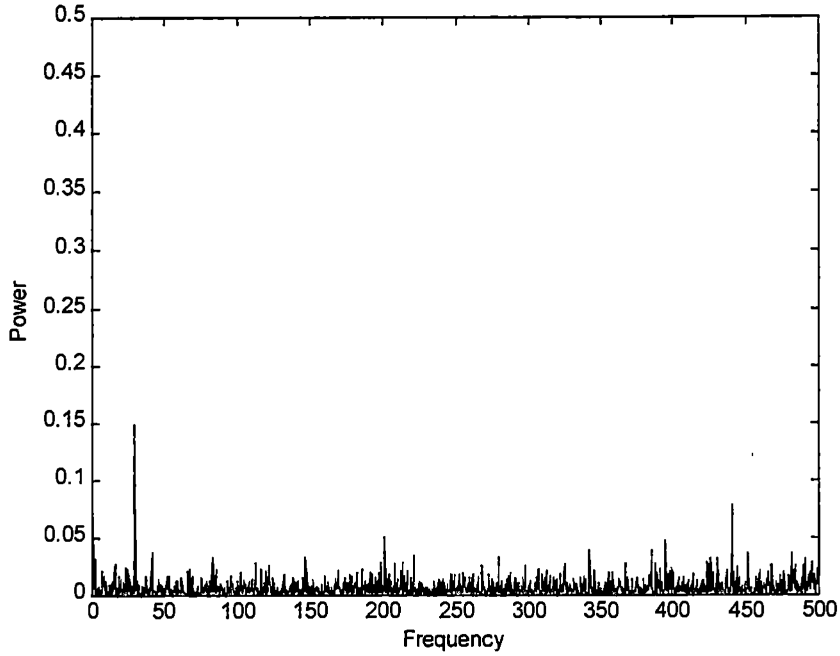




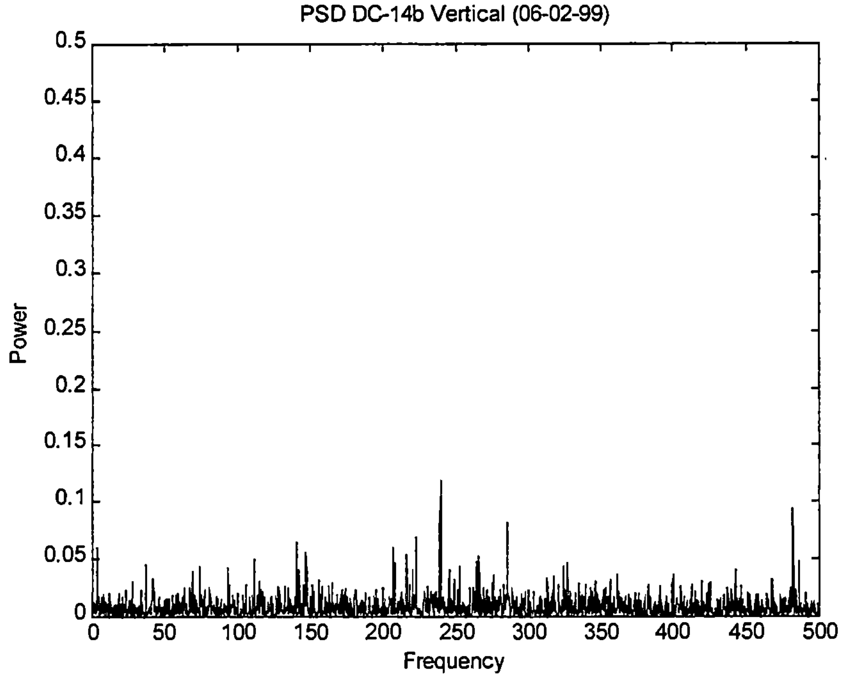

PSD DC-14a Vertical (06-02-99)
Power
Frequency
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
50
100
150
200
250
300
350
400
450
500

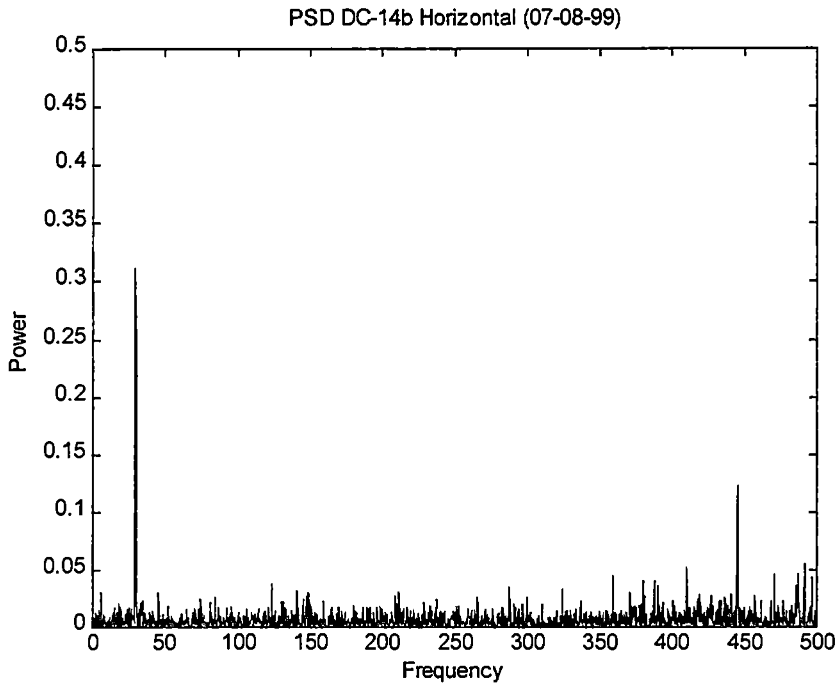

PSD DC-14a Vertical (07-08-99)
Power
Frequency
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
50
100
150
200
250
300
350
400
450
500

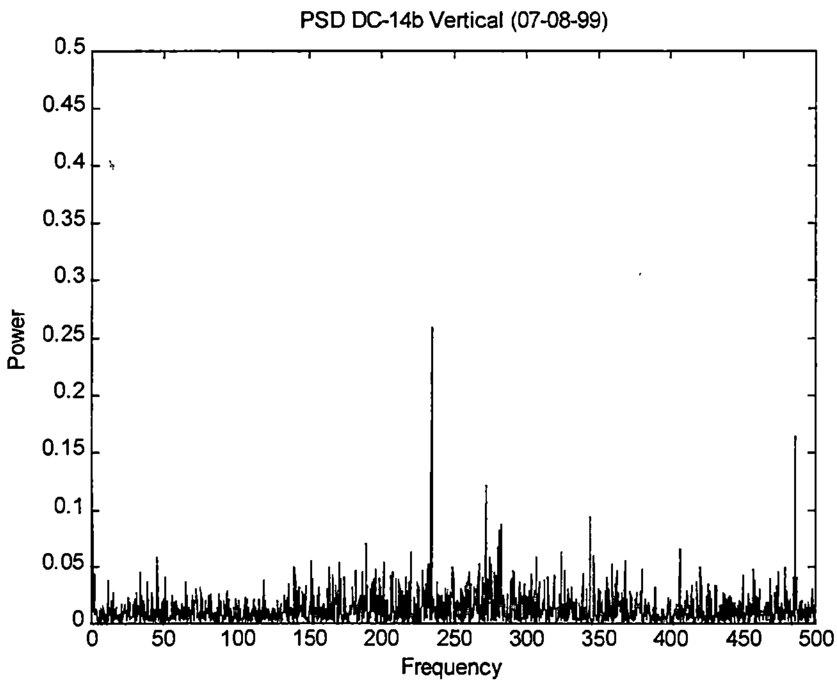
PSD DC-14b Horizontal (06-02-99)

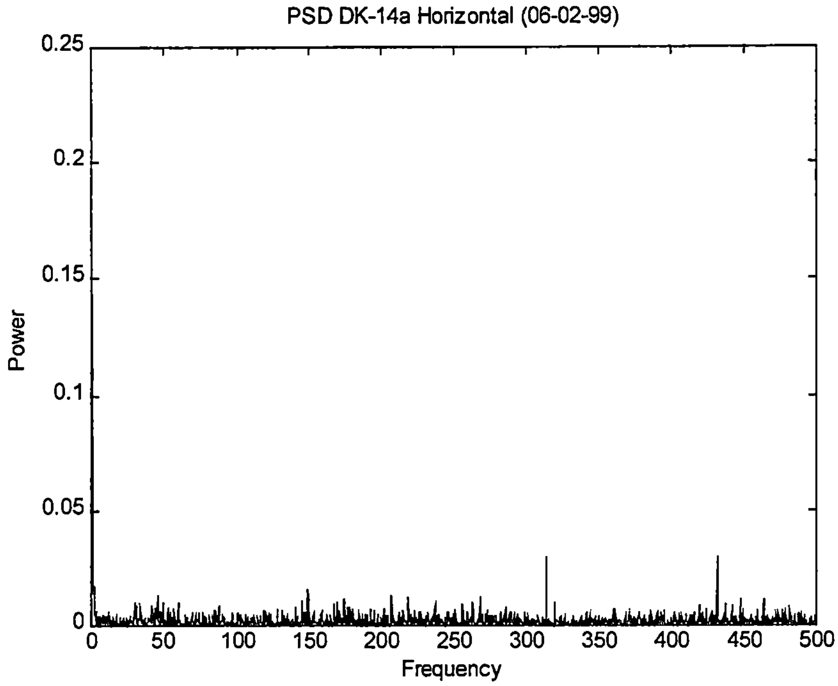


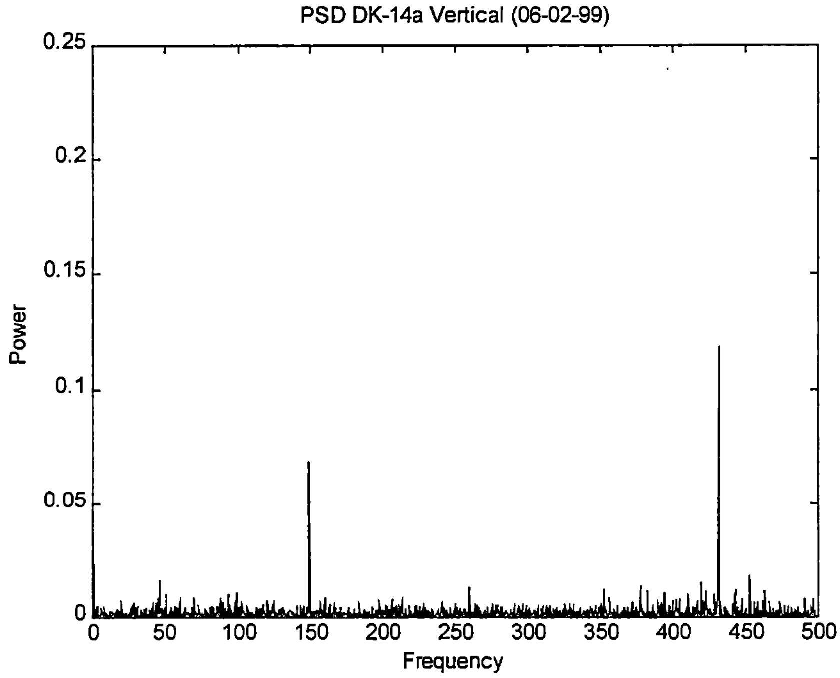
PSD DC-14b Vertical (06-02-99)

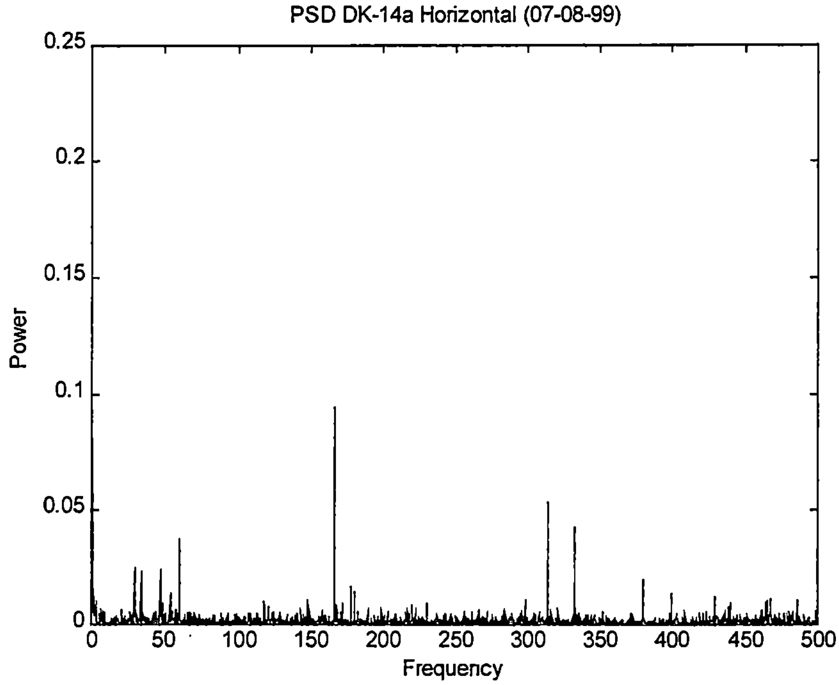


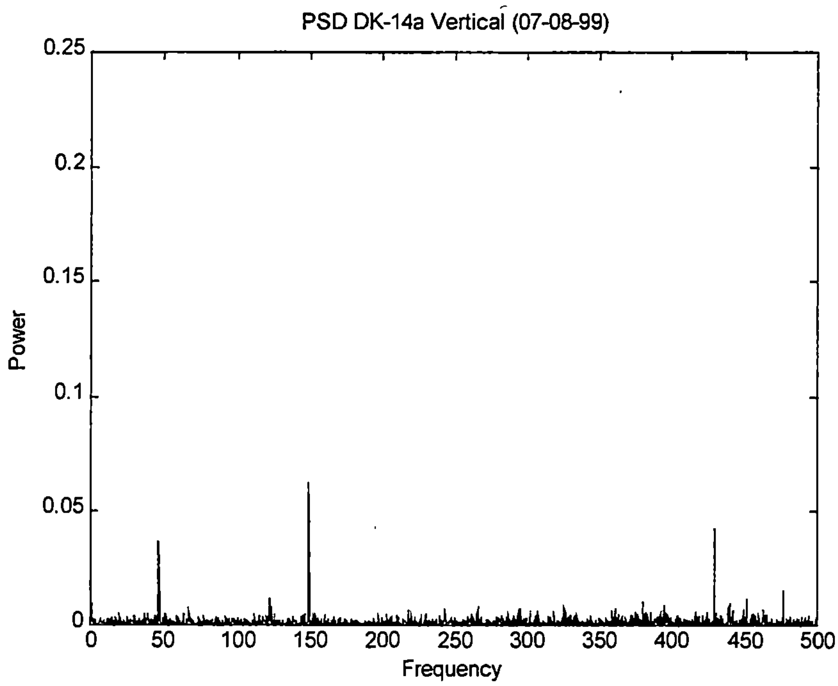

PSD DC-14b Horizontal (07-08-99)
Power
Frequency
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
0
50
100
150
200
250
300
350
400
450
500

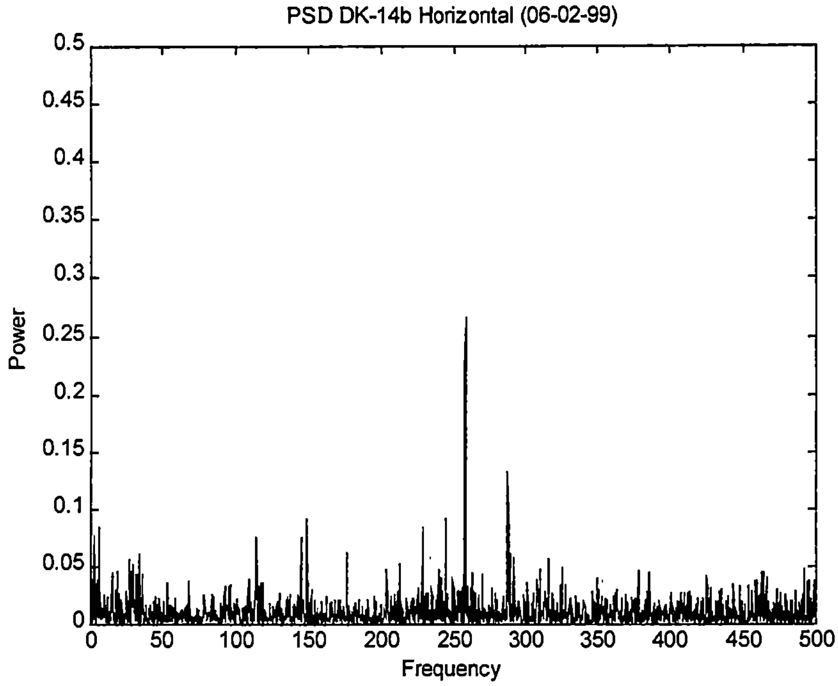

PSD DC-14b Vertical (07-08-99)
Power
Frequency
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
0
50
100
150
200
250
300
350
400
450
500

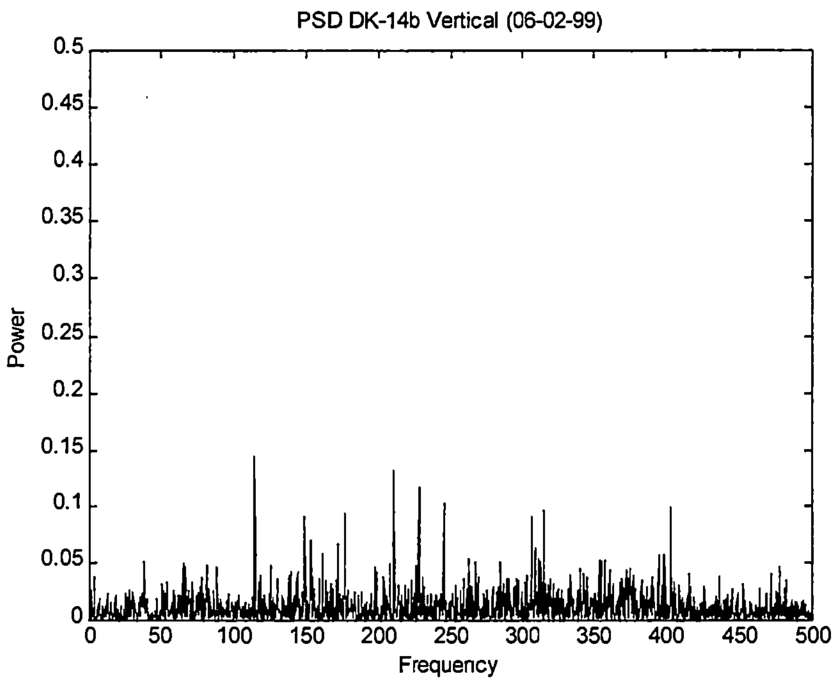

PSD DK-14a Horizontal (06-02-99)
Power
Frequency
0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300
350
400
450
500


PSD DK-14a Vertical (06-02-99)
Power
Frequency
0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300
350
400
450
500

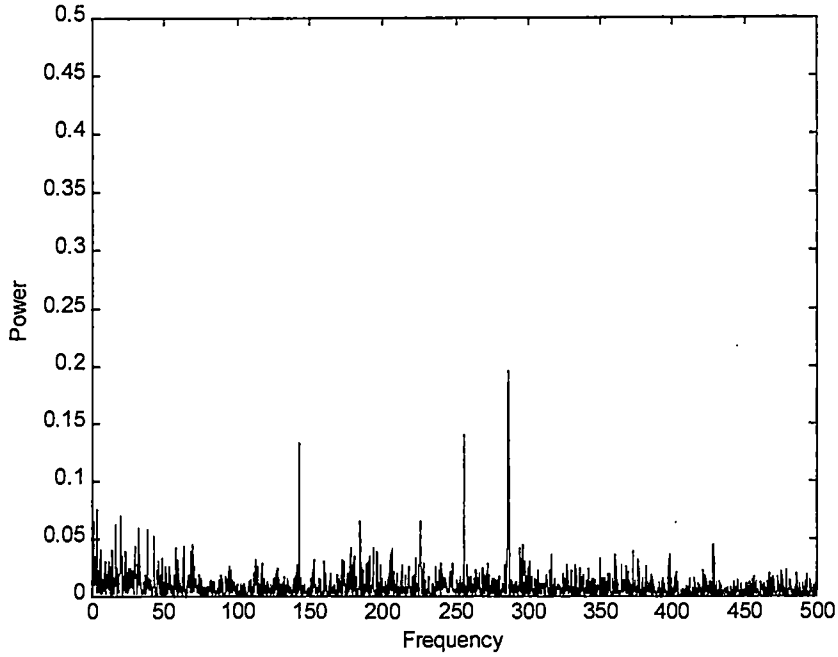

PSD DK-14a Horizontal (07-08-99)
Power
0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300
350
400
450
500
Frequency


PSD DK-14a Vertical (07-08-99)
Power
0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300
350
400
450
500
Frequency

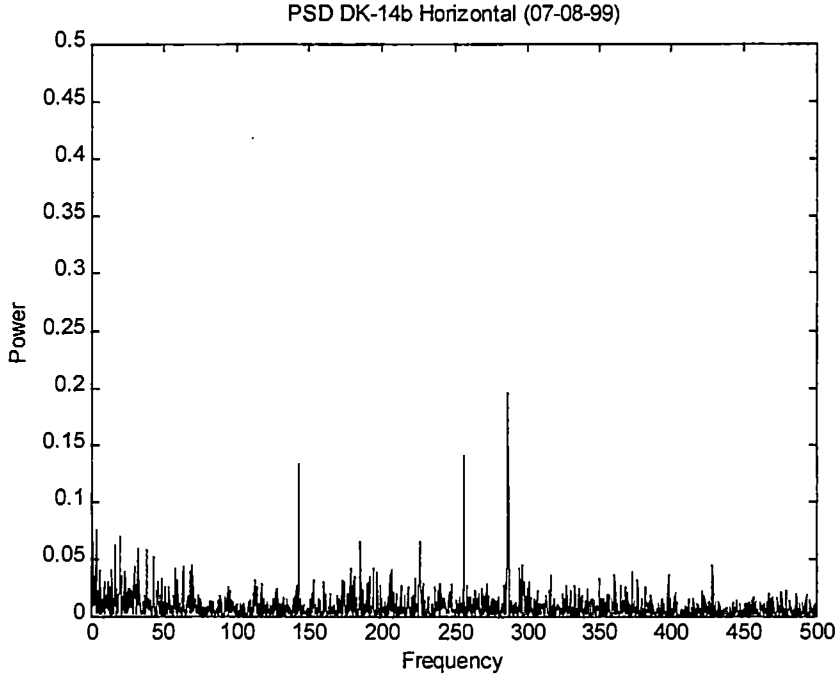

PSD DK-14b Horizontal (06-02-99)
Power
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300
350
400
450
500
Frequency

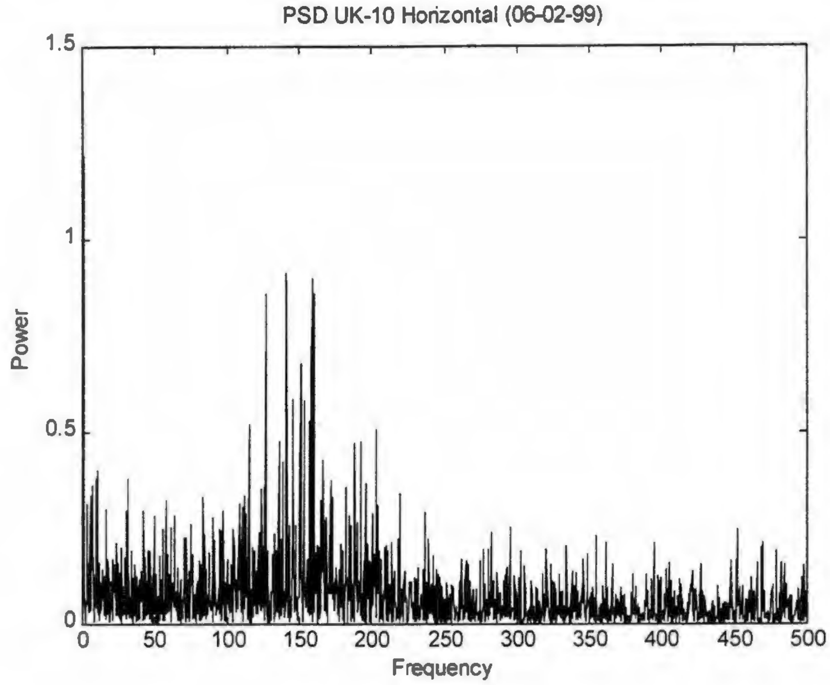

PSD DK-14b Vertical (06-02-99)
Power
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
50
100
150
200
250
300
350
400
450
500
Frequency

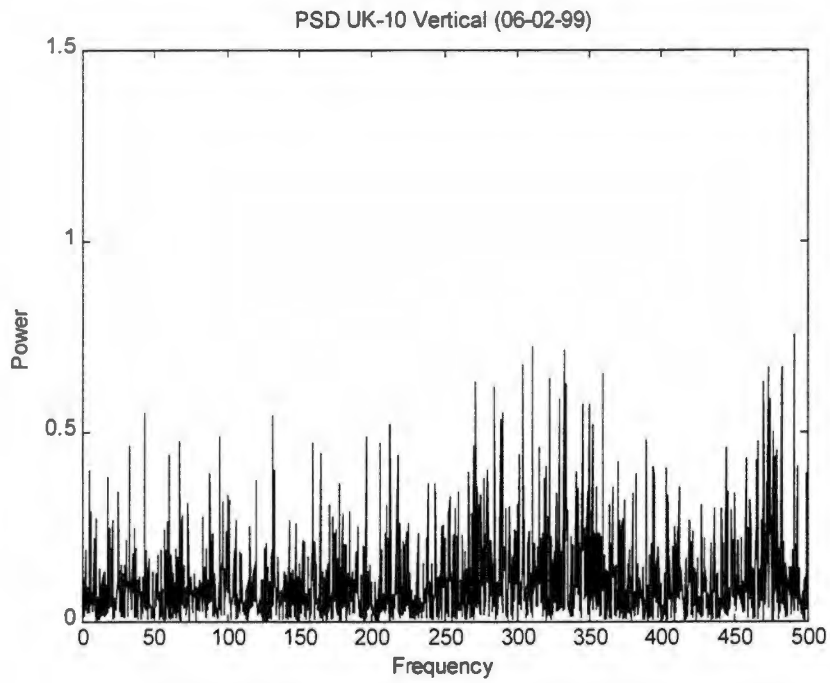
PSD DK-14b Horizontal (07-08-99)

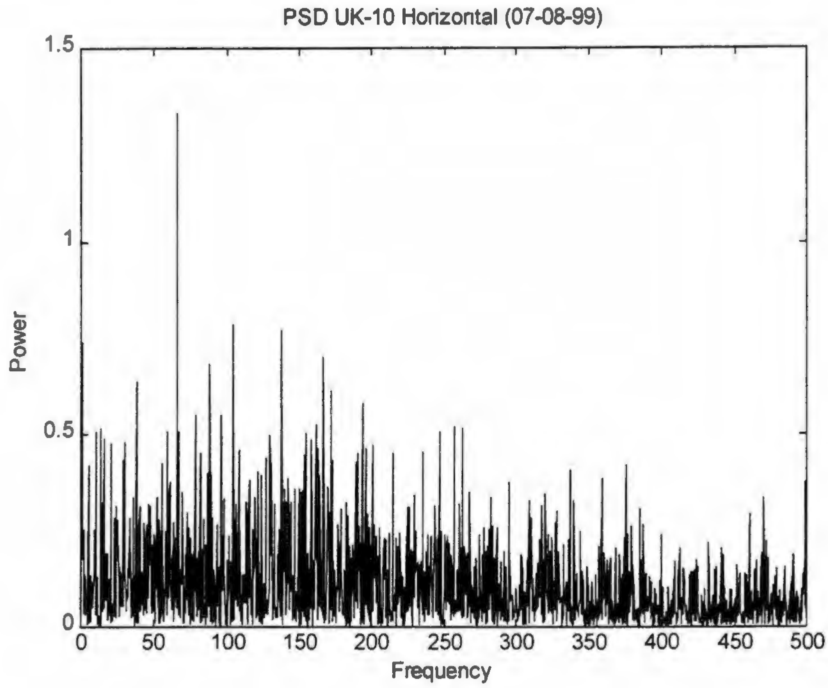


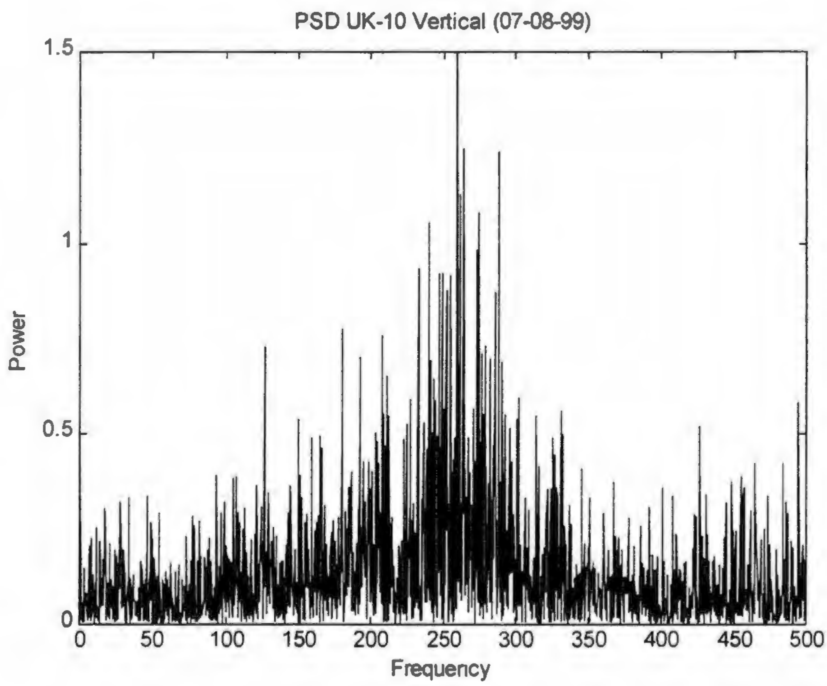
PSD DK-14b Horizontal (07-08-99)

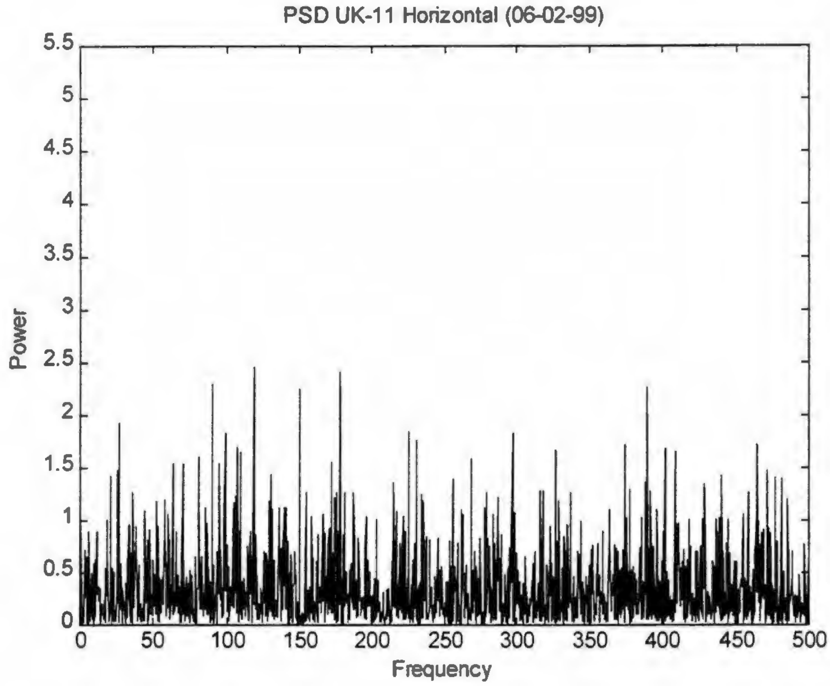


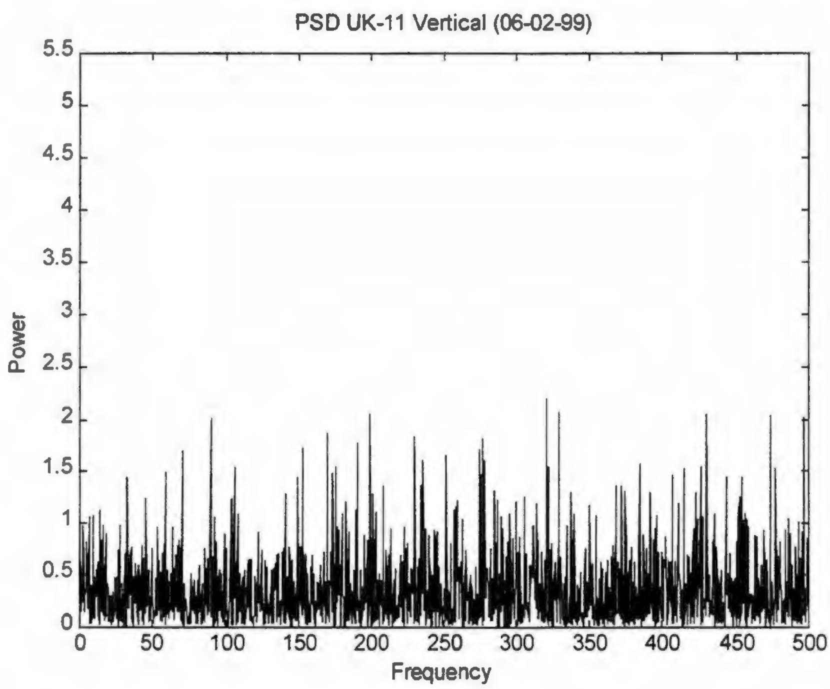

PSD UK-10 Horizontal (06-02-99)
Power
Frequency
0
0.5
1
1.5
0
50
100
150
200
250
300
350
400
450
500

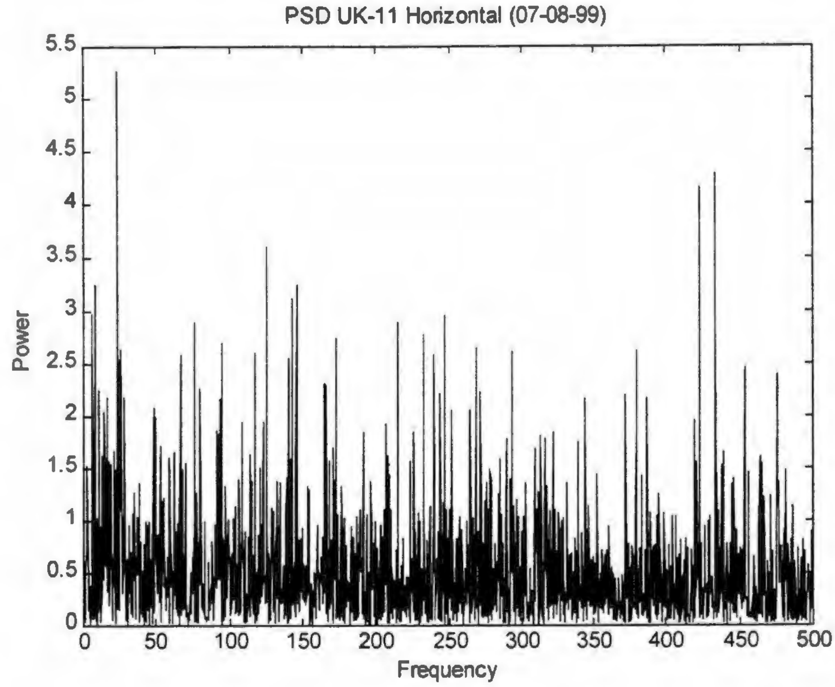

PSD UK-10 Vertical (06-02-99)
Power
Frequency
0
0.5
1
1.5
0
50
100
150
200
250
300
350
400
450
500

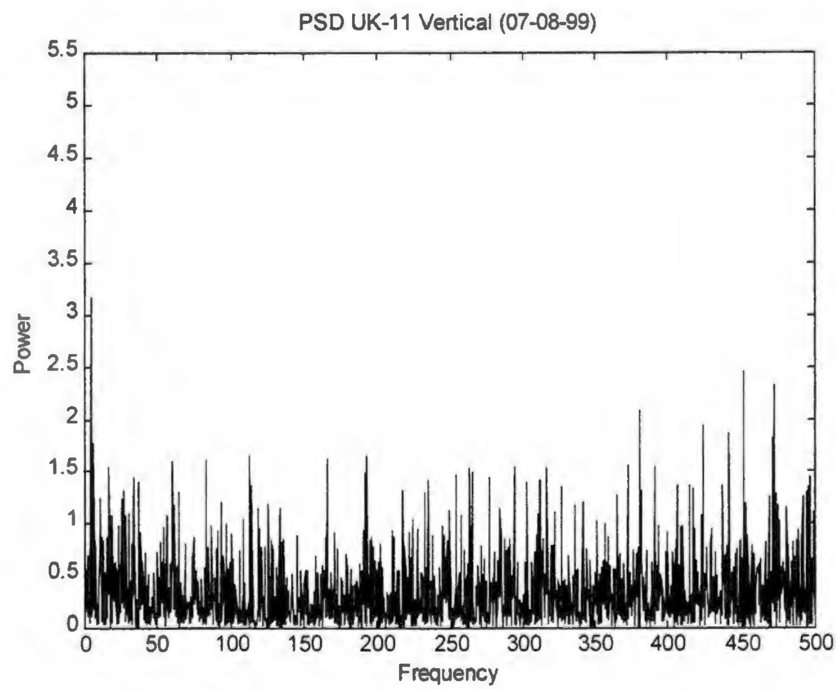

PSD UK-10 Horizontal (07-08-99)
Power
Frequency
1.5
1
0.5
0
0
50
100
150
200
250
300
350
400
450
500

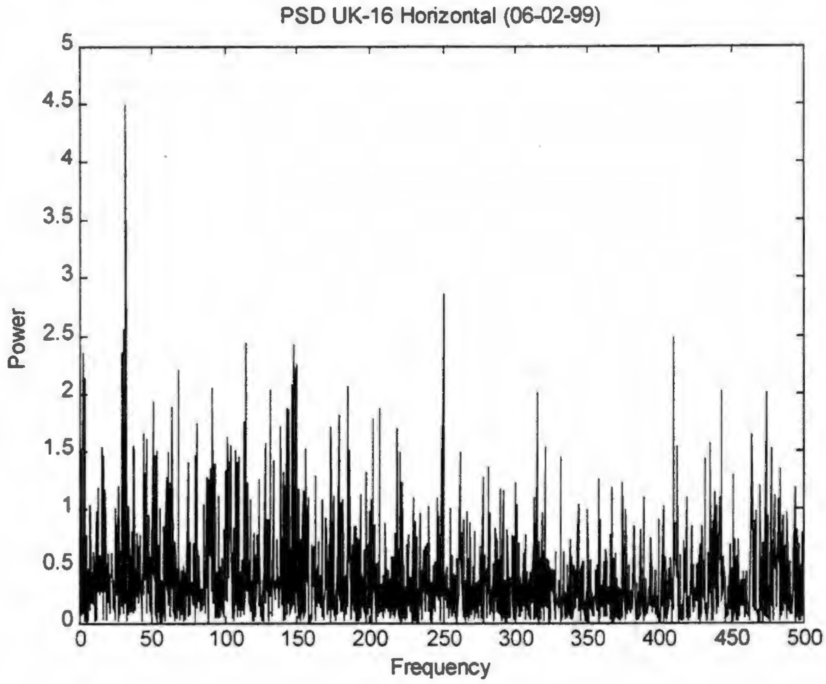

PSD UK-10 Vertical (07-08-99)
Power
Frequency
1.5
1
0.5
0
0
50
100
150
200
250
300
350
400
450
500

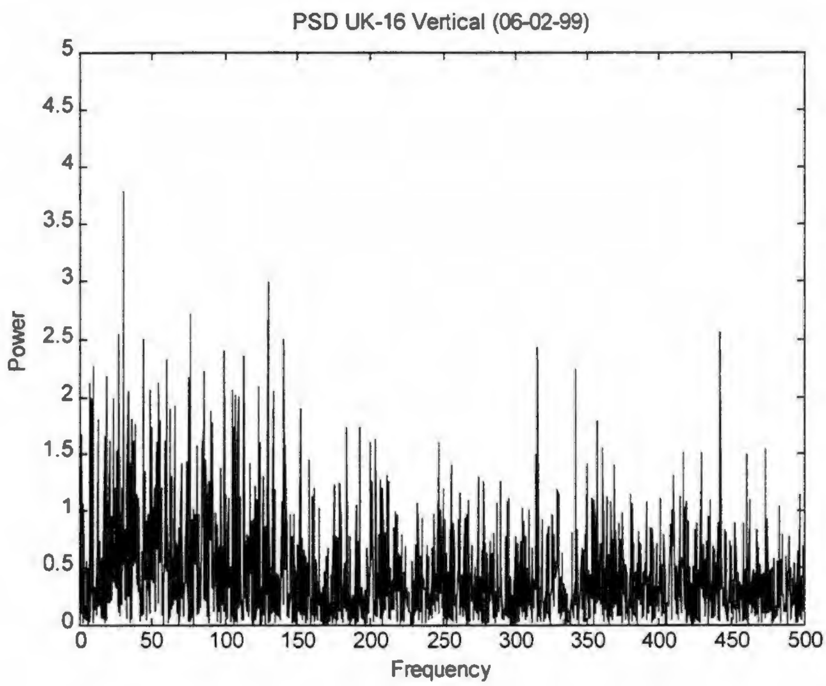

PSD UK-11 Horizontal (06-02-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
0
50
100
150
200
250
300
350
400
450
500

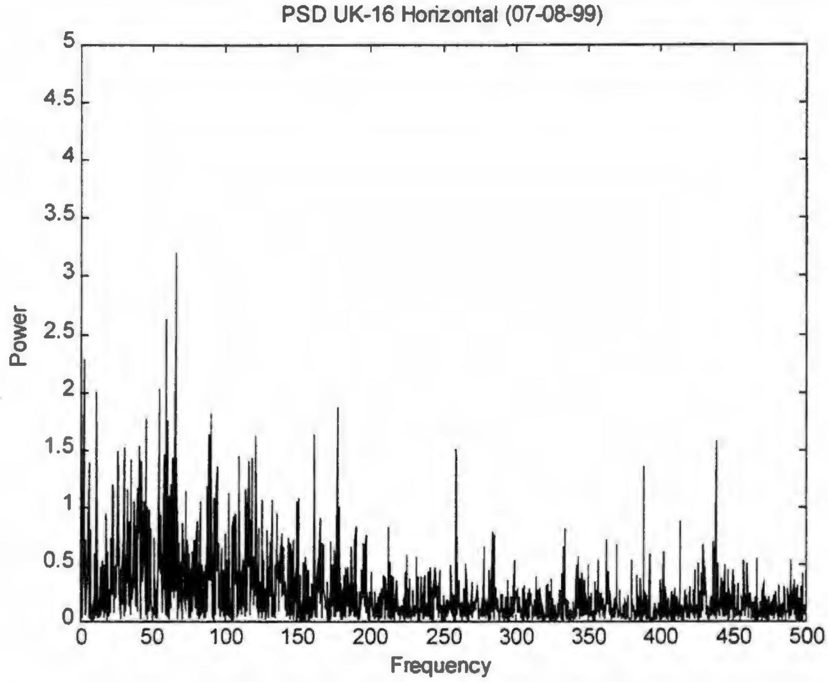

PSD UK-11 Vertical (06-02-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
0
50
100
150
200
250
300
350
400
450
500

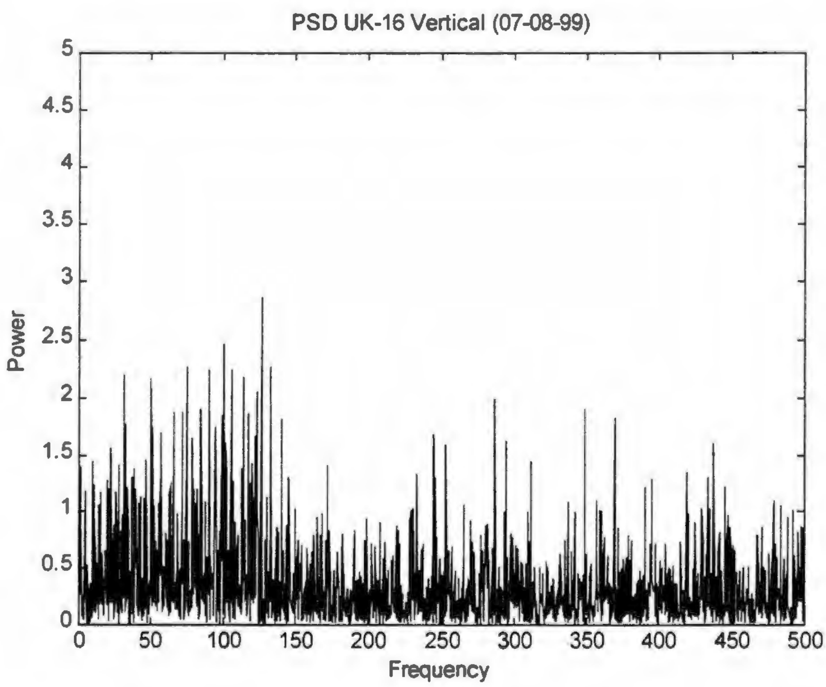

PSD UK-11 Horizontal (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
0
50
100
150
200
250
300
350
400
450
500

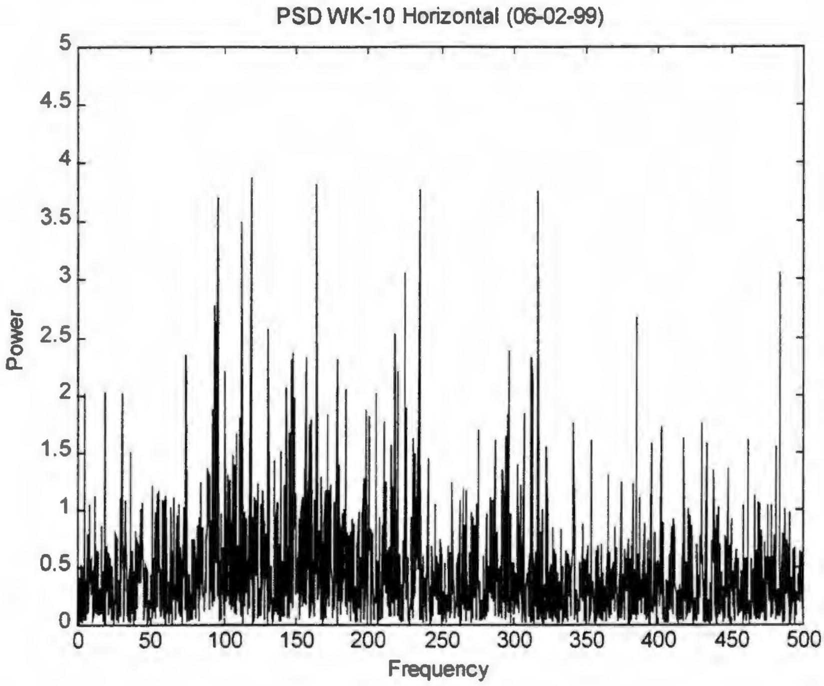

PSD UK-11 Vertical (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
0
50
100
150
200
250
300
350
400
450
500


PSD UK-16 Horizontal (06-02-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500

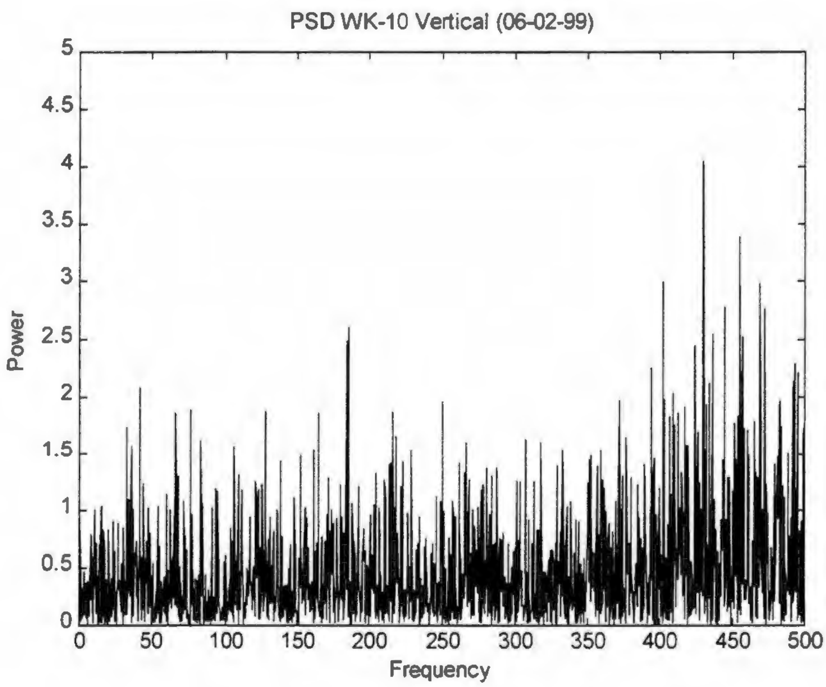

PSD UK-16 Vertical (06-02-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500


PSD UK-16 Horizontal (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500

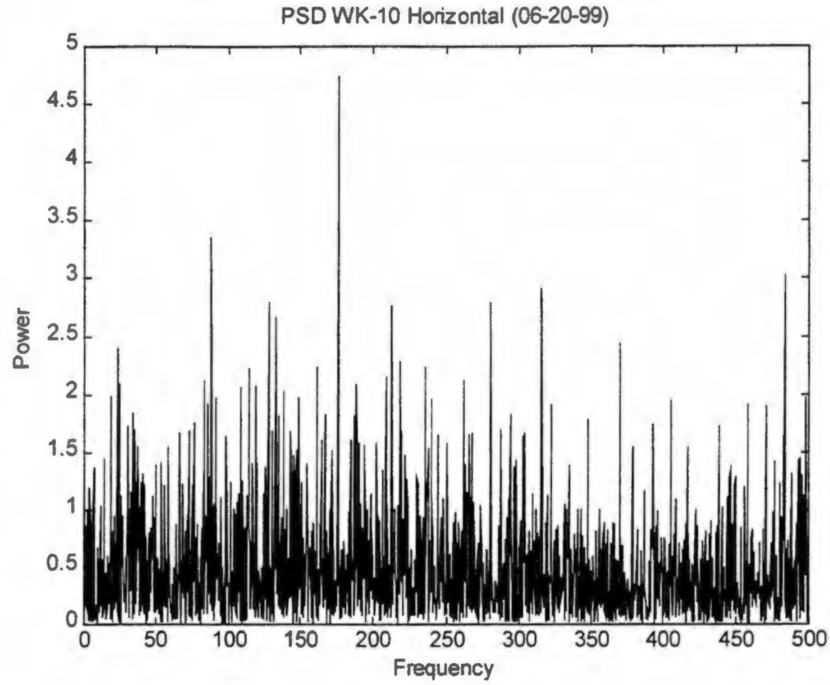

PSD UK-16 Vertical (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500

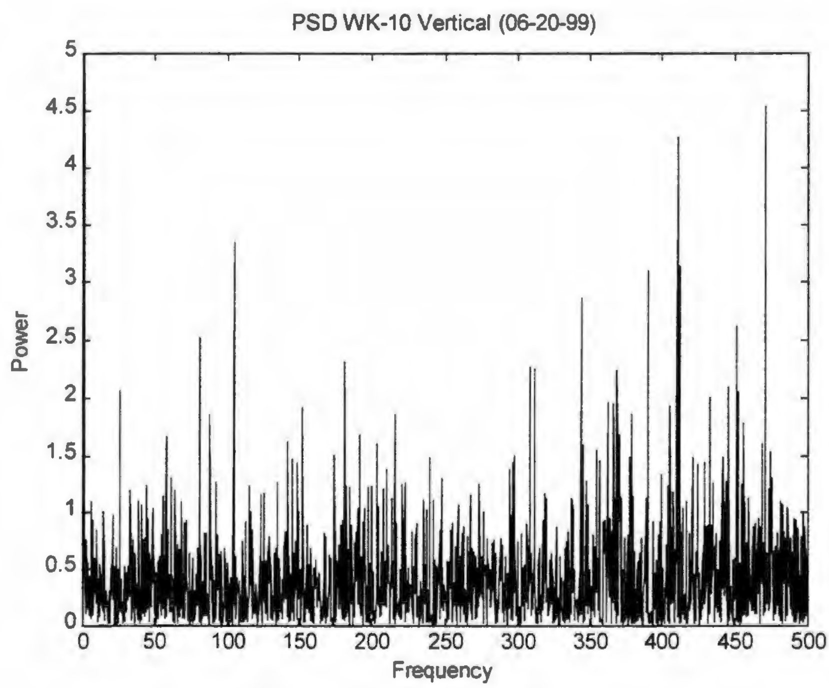


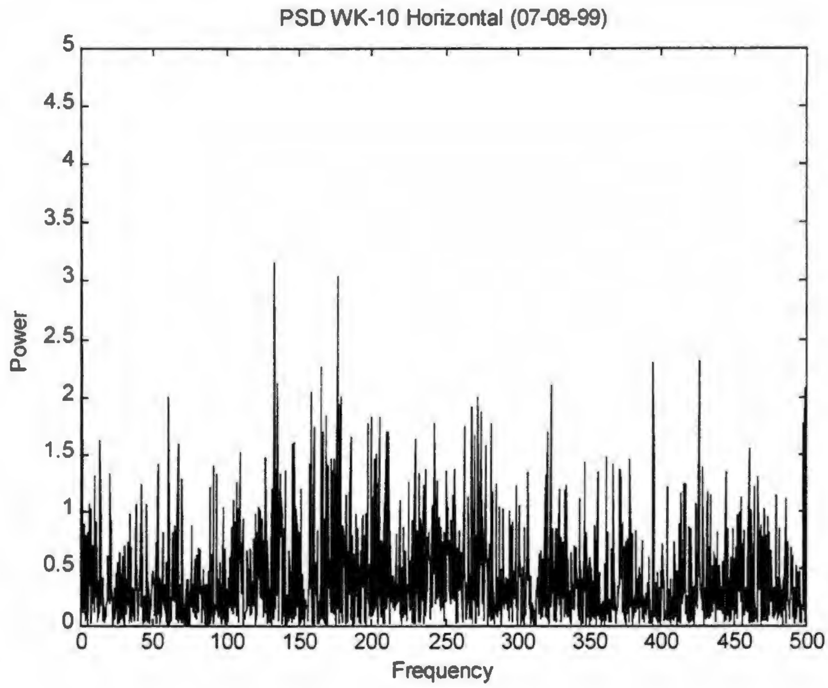
PSD WK-10 Horizontal (06-02-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500

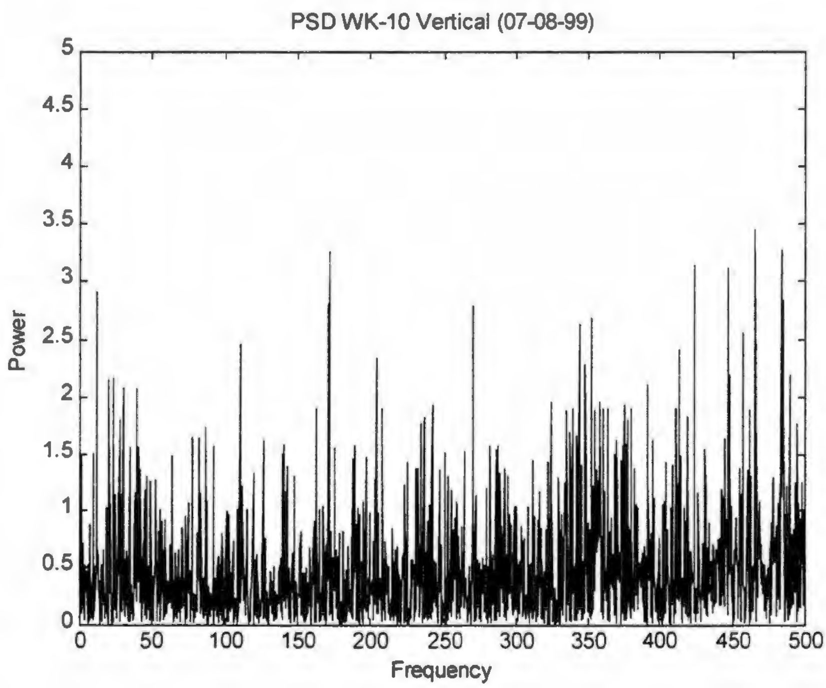


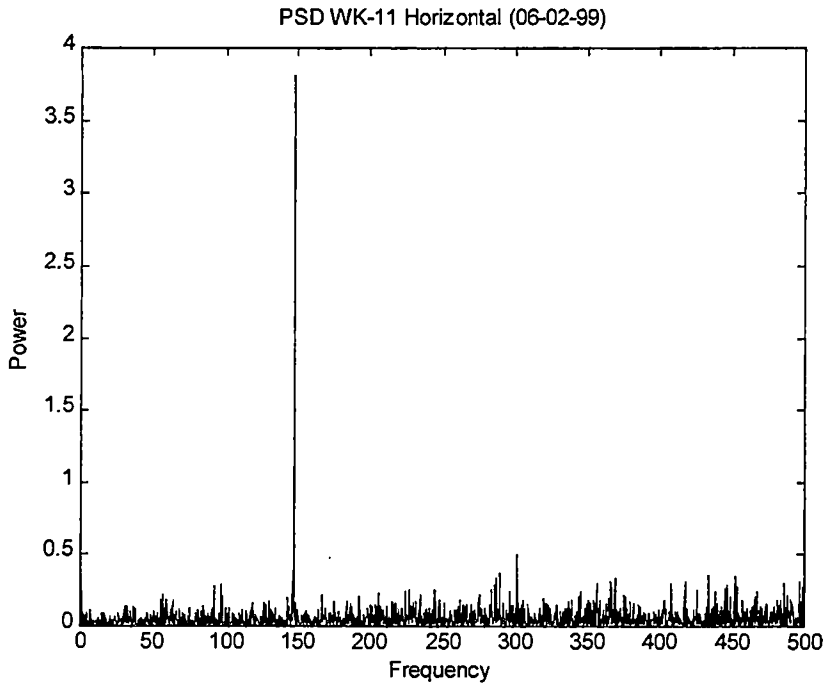
PSD WK-10 Vertical (06-02-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500


PSD WK-10 Horizontal (06-20-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500

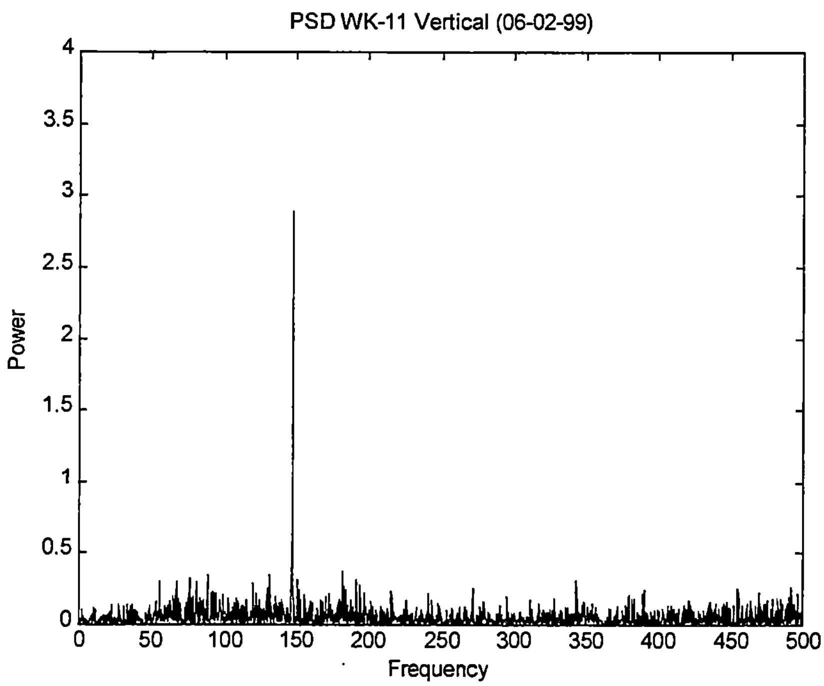

PSD WK-10 Vertical (06-20-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500


PSD WK-10 Horizontal (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500


PSD WK-10 Vertical (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500

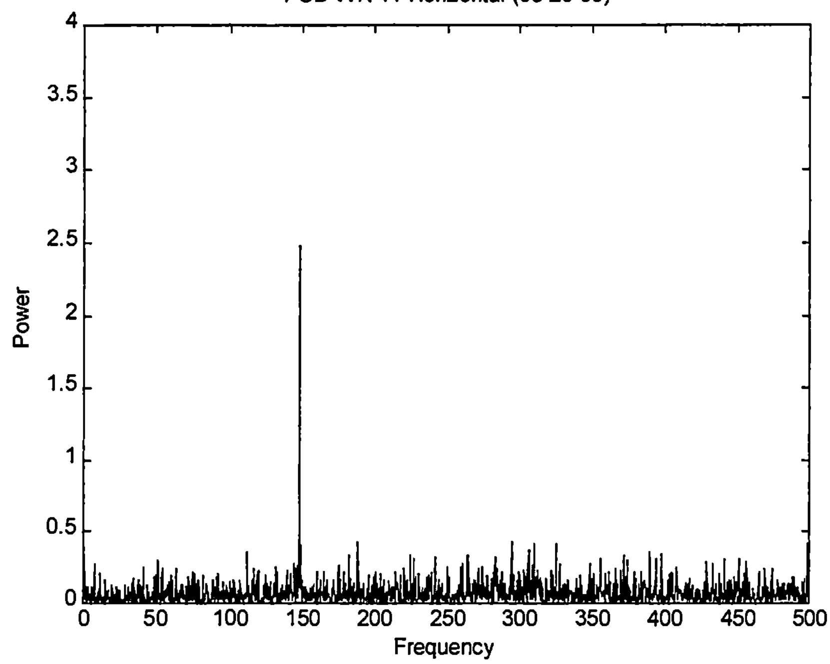


PSD WK-11 Horizontal (06-02-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
0
50
100
150
200
250
300
350
400
450
500

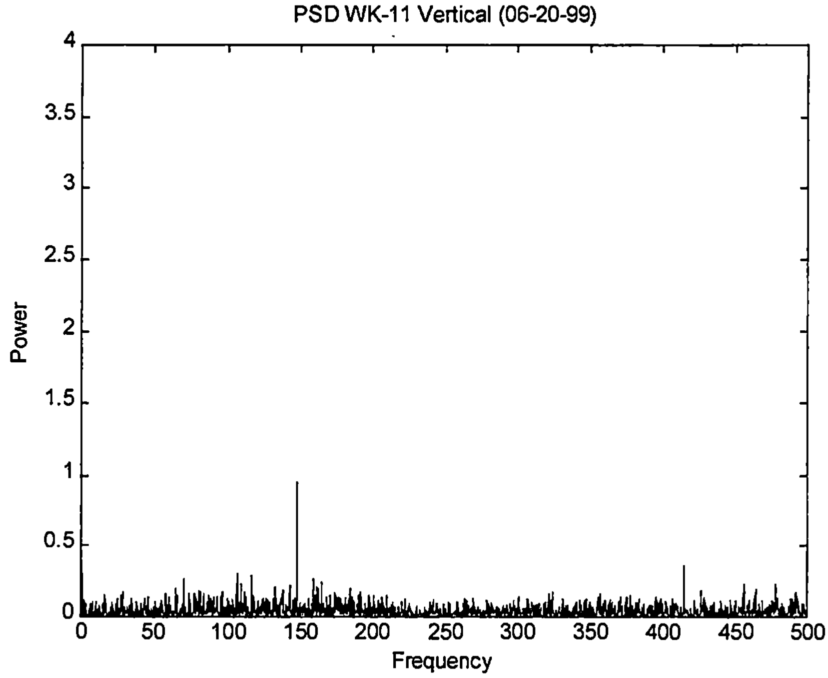


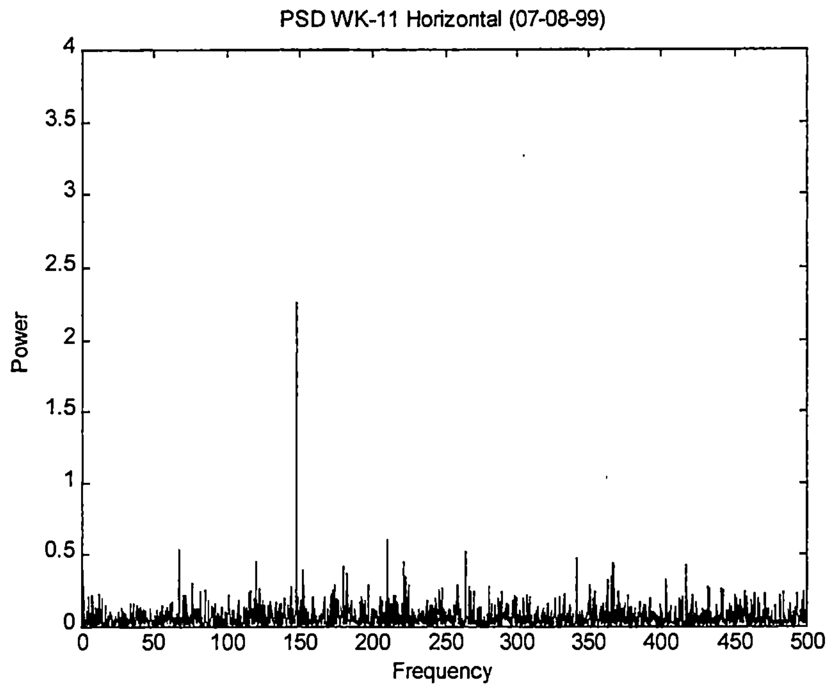
PSD WK-11 Vertical (06-02-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
0
50
100
150
200
250
300
350
400
450
500

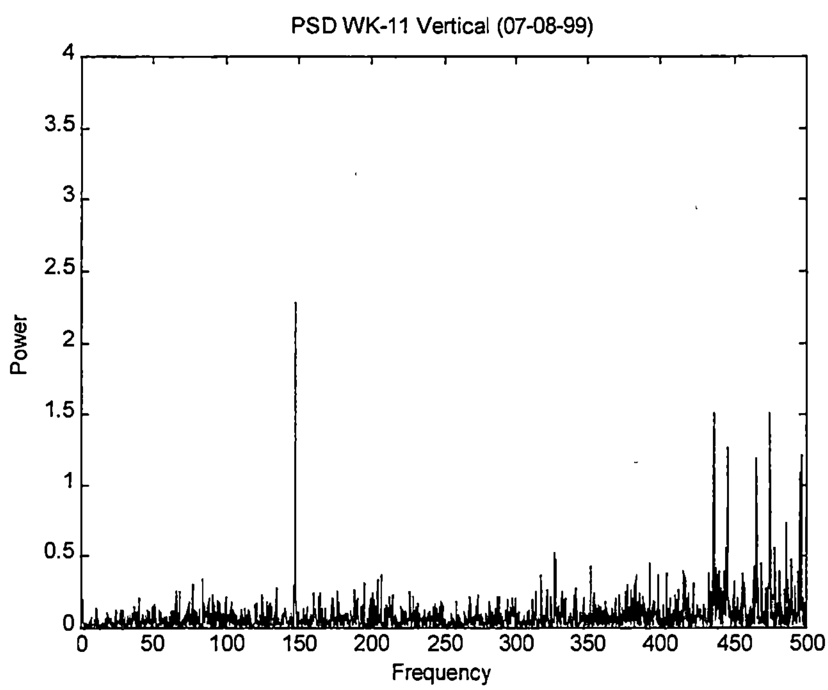
PSD WK-11 Horizontal (06-20-99)

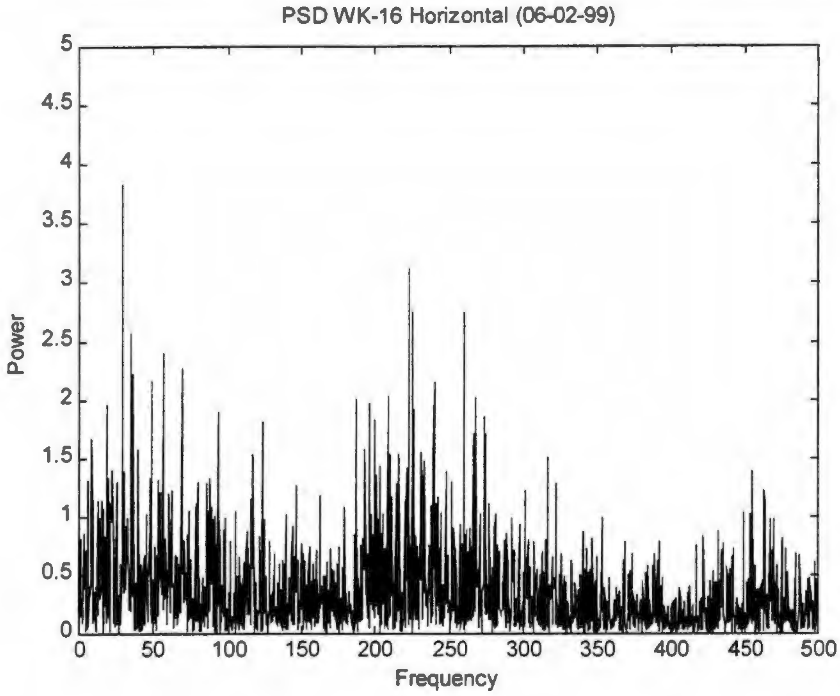


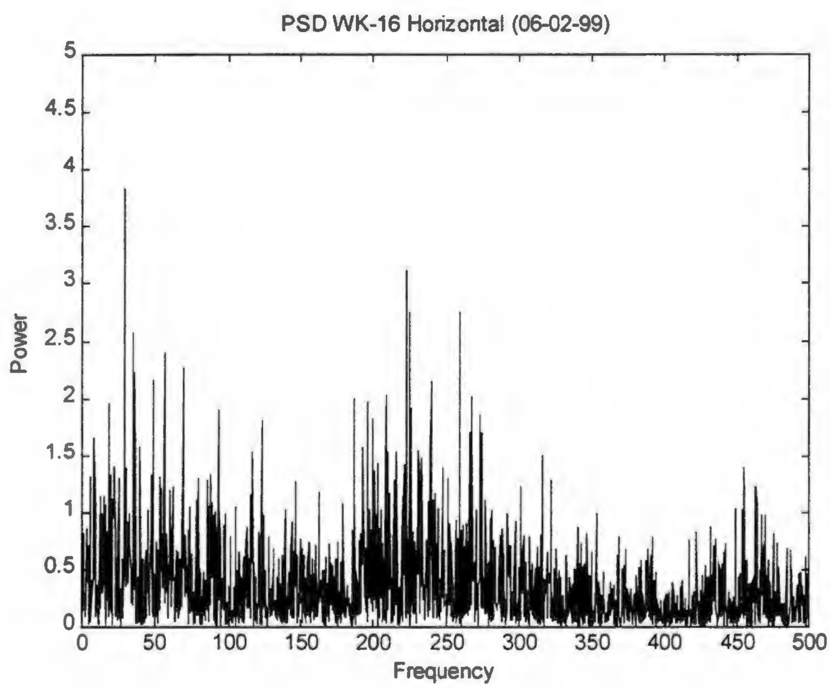
PSD WK-11 Vertical (06-20-99)

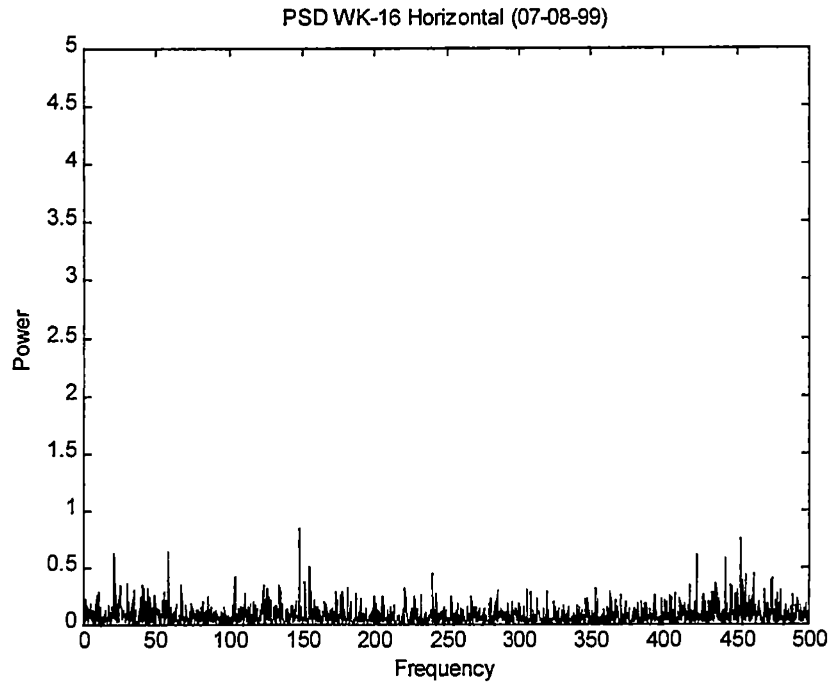


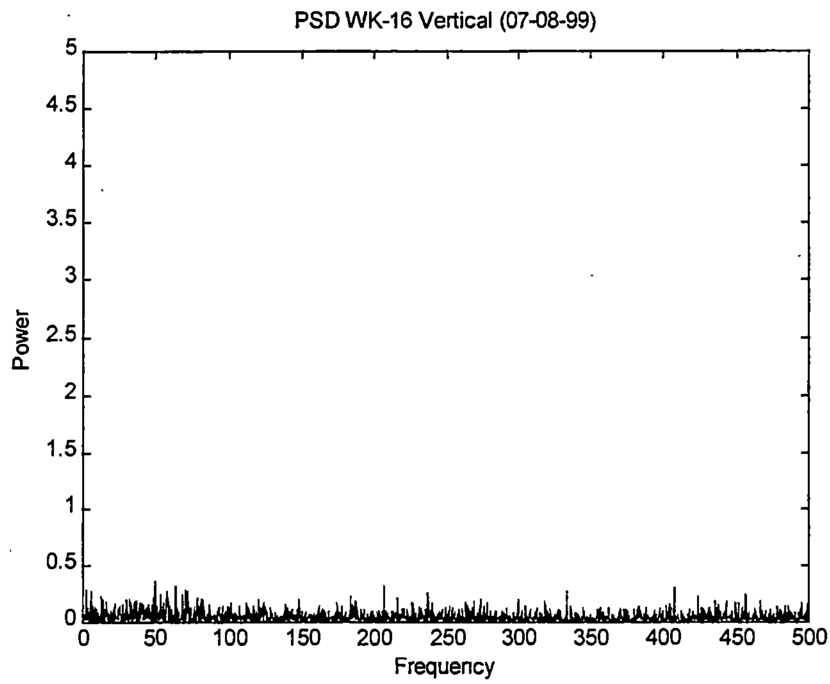

PSD WK-11 Horizontal (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
0
50
100
150
200
250
300
350
400
450
500


PSD WK-11 Vertical (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
0
50
100
150
200
250
300
350
400
450
500


PSD WK-16 Horizontal (06-02-99)
Power
Frequency


PSD WK-16 Horizontal (06-02-99)
Power
Frequency


PSD WK-16 Horizontal (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500


PSD WK-16 Vertical (07-08-99)
Power
Frequency
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
50
100
150
200
250
300
350
400
450
500

Frequency PSD Program (freqpsd.m)

```
clear all

load wk16ver.dat
filename = wk16ver;

average = mean(filename);
filename = filename - average;

Fs = 1000;
X = fft(filename);
nfft = length(filename);
Pxx = (abs(X).^2)/nfft;

select = (1:nfft/2 +1)';
PSD = Pxx(select);
freq = (select - 1)*Fs/nfft;
plot(freq,PSD);title('PSD WK-16 Vertical (07-08-99)');
xlabel('Frequency');ylabel('Power');
axis([0 500 0 5])

maxval1 = find(PSD == max(PSD));
maxfreq1 = maxval1*Fs/nfft;
maxpower1 = max(PSD);
PSD(maxval1) = PSD(maxval1)*0;
maxval2 = find(PSD == max(PSD));
maxfreq2 = maxval2*Fs/nfft;
maxpower2 = max(PSD);
PSD(maxval2) = PSD(maxval2)*0;
maxval3 = find(PSD == max(PSD));
maxfreq3 = maxval3*Fs/nfft;
maxpower3 = max(PSD);
PSD(maxval3) = PSD(maxval3)*0;
maxval4 = find(PSD == max(PSD));
maxfreq4 = maxval4*Fs/nfft;
maxpower4 = max(PSD);
PSD(maxval4) = PSD(maxval4)*0;
maxval5 = find(PSD == max(PSD));
maxfreq5 = maxval5*Fs/nfft;
maxpower5 = max(PSD);


%maxpower = [maxpower1 maxpower2 maxpower3 maxpower4 maxpower5]
%maxfreq = [maxfreq1 maxfreq2 maxfreq3 maxfreq4 maxfreq5]
```

Data for Artificial Intelligence Programs (data.m)

```
clear all

% The first 40 data sets are "good", the next 200 are "ok", the last
280
% are "bad".

% =       [HP Age(years) DeltaT(C)   RMS-H(g)  RMS-V(g) Max-Min-H(g)
Max-Min-V(g)]
x(1,:)  = [5    rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(2,:)  = [5    rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(3,:)  = [5    rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(4,:)  = [5    rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(5,:)  = [5    rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(6,:)  = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(7,:)  = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(8,:)  = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(9,:)  = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(10,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
rand  ];
x(11,:) = [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(12,:) = [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(13,:) = [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(14,:) = [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(15,:) = [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(16,:) = [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(17,:) = [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(18,:) = [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(19,:) = [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
x(20,:) = [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
```

```
x(21,:) = [15   rand*5   0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(22,:) = [15   rand*5   0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(23,:) = [15   rand*5   0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(24,:) = [15   rand*5   0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(25,:) = [15   rand*5   0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(26,:) = [15  5+rand*5  0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(27,:) = [15  5+rand*5  0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(28,:) = [15  5+rand*5  0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(29,:) = [15  5+rand*5  0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(30,:) = [15  5+rand*5  0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
x(31,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(32,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(33,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(34,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(35,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(36,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(37,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(38,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(39,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
x(40,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
%  =      [HP Age(years) DeltaT(C)   RMS-H(g)  RMS-V(g) Max-Min-H(g)
Max-Min-V(g)]
x(41,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
x(42,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
x(43,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
x(44,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
x(45,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
x(46,:) = [5    rand*5   0.5+rand*3` rand/5     rand/5+.2  rand
rand  ];
```

```
x(47,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5+.2  rand
rand  ];
x(48,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5+.2  rand
rand  ];
x(49,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5+.2  rand
rand  ];
x(50,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5+.2  rand
rand  ];
x(51,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand+1
rand  ];
x(52,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand+1
rand  ];
x(53,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand+1
rand  ];
x(54,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand+1
rand  ];
x(55,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand+1
rand  ];
x(56,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand
rand+1];
x(57,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand
rand+1];
x(58,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand
rand+1];
x(59,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand
rand+1];
x(60,:) = [5    rand*5    0.5+rand*3  rand/5       rand/5     rand
rand+1];
x(61,:) = [5    rand*5    3.5+rand*3  rand/5       rand/5     rand
rand  ];
x(62,:) = [5    rand*5    3.5+rand*3  rand/5       rand/5     rand
rand  ];
x(63,:) = [5    rand*5    3.5+rand*3  rand/5       rand/5     rand
rand  ];
x(64,:) = [5    rand*5    3.5+rand*3  rand/5       rand/5     rand
rand  ];
x(65,:) = [5    rand*5    3.5+rand*3  rand/5       rand/5     rand
rand  ];
x(66,:) = [5   5+rand*5   0.5+rand*3  rand*.3+.2 rand/5       rand
rand  ];
x(67,:) = [5   5+rand*5   0.5+rand*3  rand*.3+.2 rand/5       rand
rand  ];
x(68,:) = [5   5+rand*5   0.5+rand*3  rand*.3+.2 rand/5       rand
rand  ];
x(69,:) = [5   5+rand*5   0.5+rand*3  rand*.3+.2 rand/5       rand
rand  ];
x(70,:) = [5   5+rand*5   0.5+rand*3  rand*.3+.2 rand/5       rand
rand  ];
x(71,:) = [5   5+rand*5   0.5+rand*3  rand/5       rand*.3+.2 rand
rand  ];
x(72,:) = [5   5+rand*5   0.5+rand*3  rand/5       rand*.3+.2 rand
rand  ];
x(73,:) = [5   5+rand*5   0.5+rand*3  rand/5       rand*.3+.2 rand
rand  ];
```

```
x(74,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand*.3+.2 rand
rand  ];
x(75,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand*.3+.2 rand
rand  ];
x(76,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand*2+1
rand  ];
x(77,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand*2+1
rand  ];
x(78,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand*2+1
rand  ];
x(79,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand*2+1
rand  ];
x(80,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand*2+1
rand  ];
x(81,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
x(82,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
x(83,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
x(84,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
x(85,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
x(86,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand  ];
x(87,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand  ];
x(88,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand  ];
x(89,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand  ];
x(90,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand  ];
x(91,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand  ];
x(92,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand  ];
x(93,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand  ];
x(94,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand  ];
x(95,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand  ];
x(96,:) = [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand  ];
x(97,:) = [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand  ];
x(98,:) = [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand  ];
x(99,:) = [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand  ];
x(100,:)= [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand  ];
```

```
x(101,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand  ];
x(102,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand  ];
x(103,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand  ];
x(104,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand  ];
x(105,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand  ];
x(106,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
x(107,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
x(108,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
x(109,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
x(110,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
x(111,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(112,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(113,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(114,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(115,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(116,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
x(117,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
x(118,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
x(119,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
x(120,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
x(121,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
x(122,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
x(123,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
x(124,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
x(125,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
x(126,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
x(127,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
```

```
x(128,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
x(129,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
x(130,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
x(131,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
x(132,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
x(133,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
x(134,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
x(135,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
x(136,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(137,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(138,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(139,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(140,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
x(141,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
x(142,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
x(143,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
x(144,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
x(145,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
x(146,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
x(147,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
x(148,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
x(149,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
x(150,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
x(151,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
x(152,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
x(153,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
x(154,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
```

```
x(155,:)= [15   rand*5    0.5+rand*3  rand*.4     rand*.4     rand+3
3*rand  ];
x(156,:)= [15   rand*5    0.5+rand*3  rand*.4     rand*.4     rand*3
rand+3];
x(157,:)= [15   rand*5    0.5+rand*3  rand*.4     rand*.4     rand*3
rand+3];
x(158,:)= [15   rand*5    0.5+rand*3  rand*.4     rand*.4     rand*3
rand+3];
x(159,:)= [15   rand*5    0.5+rand*3  rand*.4     rand*.4     rand*3
rand+3];
x(160,:)= [15   rand*5    0.5+rand*3  rand*.4     rand*.4     rand*3
rand+3];
x(161,:)= [15   rand*5    3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(162,:)= [15   rand*5    3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(163,:)= [15   rand*5    3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(164,:)= [15   rand*5    3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(165,:)= [15   rand*5    3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(166,:)= [15  5+rand*5   0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
x(167,:)= [15  5+rand*5   0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
x(168,:)= [15  5+rand*5   0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
x(169,:)= [15  5+rand*5   0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
x(170,:)= [15  5+rand*5   0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
x(171,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
x(172,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
x(173,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
x(174,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
x(175,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
x(176,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
x(177,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
x(178,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
x(179,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
x(180,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
x(181,:)= [15  5+rand*5   0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+3];
```

```
x(182,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*3
2*rand+3];
x(183,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*3
2*rand+3];
x(184,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*3
2*rand+3];
x(185,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*3
2*rand+3];
x(186,:)= [15  5+rand*5  3.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(187,:)= [15  5+rand*5  3.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(188,:)= [15  5+rand*5  3.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(189,:)= [15  5+rand*5  3.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(190,:)= [15  5+rand*5  3.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(191,:)= [20   rand*5   0.5+rand*3  rand/5+.5  rand*.5    rand*4
4*rand  ];
x(192,:)= [20   rand*5   0.5+rand*3  rand/5+.5  rand*.5    rand*4
4*rand  ];
x(193,:)= [20   rand*5   0.5+rand*3  rand/5+.5  rand*.5    rand*4
4*rand  ];
x(194,:)= [20   rand*5   0.5+rand*3  rand/5+.5  rand*.5    rand*4
4*rand  ];
x(195,:)= [20   rand*5   0.5+rand*3  rand/5+.5  rand*.5    rand*4
4*rand  ];
x(196,:)= [20   rand*5   0.5+rand*3  rand*.5    rand/5+.5  rand*4
4*rand  ];
x(197,:)= [20   rand*5   0.5+rand*3  rand*.5    rand/5+.5  rand*4
4*rand  ];
x(198,:)= [20   rand*5   0.5+rand*3  rand*.5    rand/5+.5  rand*4
4*rand  ];
x(199,:)= [20   rand*5   0.5+rand*3  rand*.5    rand/5+.5  rand*4
4*rand  ];
x(200,:)= [20   rand*5   0.5+rand*3  rand*.5    rand/5+.5  rand*4
4*rand  ];
x(201,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand+4
4*rand  ];
x(202,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand+4
4*rand  ];
x(203,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand+4
4*rand  ];
x(204,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand+4
4*rand  ];
x(205,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand+4
4*rand  ];
x(206,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4
rand+4];
x(207,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4
rand+4];
x(208,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4
rand+4];
```

```
x(209,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
rand+4];
x(210,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
rand+4];
x(211,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(212,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(213,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(214,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(215,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(216,:)= [20 5+rand*5   0.5+rand*3  rand*.3+.5 rand*.5     rand*4
4*rand  ];
x(217,:)= [20 5+rand*5   0.5+rand*3  rand*.3+.5 rand*.5     rand*4
4*rand  ];
x(218,:)= [20 5+rand*5   0.5+rand*3  rand*.3+.5 rand*.5     rand*4
4*rand  ];
x(219,:)= [20 5+rand*5   0.5+rand*3  rand*.3+.5 rand*.5     rand*4
4*rand  ];
x(220,:)= [20 5+rand*5   0.5+rand*3  rand*.3+.5 rand*.5     rand*4
4*rand  ];
x(221,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.3+.5 rand*4
4*rand  ];
x(222,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.3+.5 rand*4
4*rand  ];
x(223,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.3+.5 rand*4
4*rand  ];
x(224,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.3+.5 rand*4
4*rand  ];
x(225,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.3+.5 rand*4
4*rand  ];
x(226,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*2+4
4*rand  ];
x(227,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*2+4
4*rand  ];
x(228,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*2+4
4*rand  ];
x(229,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*2+4
4*rand  ];
x(230,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*2+4
4*rand  ];
x(231,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+4];
x(232,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+4];
x(233,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+4];
x(234,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+4];
x(235,:)= [20 5+rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+4];
```

```
x(236,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
x(237,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
x(238,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
x(239,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
x(240,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
%  =      [HP Age(years) DeltaT(C)   RMS-H(g)  RMS-V(g) Max-Min-H(g)
Max-Min-V(g)]
x(241,:)= [5    rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(242,:)= [5    rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(243,:)= [5    rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(244,:)= [5    rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(245,:)= [5    rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(246,:)= [5    rand*5   0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
x(247,:)= [5    rand*5   0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
x(248,:)= [5    rand*5   0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
x(249,:)= [5    rand*5   0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
x(250,:)= [5    rand*5   0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
x(251,:)= [5    rand*5   0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
x(252,:)= [5    rand*5   0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
x(253,:)= [5    rand*5   0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
x(254,:)= [5    rand*5   0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
x(255,:)= [5    rand*5   0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
x(256,:)= [5    rand*5   0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
x(257,:)= [5    rand*5   0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
x(258,:)= [5    rand*5   0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
x(259,:)= [5    rand*5   0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
x(260,:)= [5    rand*5   0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
x(261,:)= [5    rand*5   0.5+rand*3  rand*.2    rand*.2    rand*5+3
rand  ];
```

```
x(262,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand*5+3
rand  ];
x(263,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand*5+3
rand  ];
x(264,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand*5+3
rand  ];
x(265,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand*5+3
rand  ];
x(266,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand
5*rand+3];
x(267,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand
5*rand+3];
x(268,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand
5*rand+3];
x(269,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand
5*rand+3];
x(270,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand
5*rand+3];
x(271,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
x(272,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
x(273,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
x(274,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
x(275,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
x(276,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(277,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(278,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(279,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(280,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
x(281,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
x(282,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
x(283,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
x(284,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
x(285,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
x(286,:)= [5   5+rand*5   0.5+rand*3  rand*.4+.6 rand*.2     rand
rand  ];
x(287,:)= [5   5+rand*5   0.5+rand*3  rand*.4+.6 rand*.2     rand
rand  ];
x(288,:)= [5   5+rand*5   0.5+rand*3  rand*.4+.6 rand*.2     rand
rand  ];
```

```
x(289,:)= [5   5+rand*5  0.5+rand*3  rand*.4+.6 rand*.2    rand
rand  ];
x(290,:)= [5   5+rand*5  0.5+rand*3  rand*.4+.6 rand*.2    rand
rand  ];
x(291,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.4+.6 rand
rand  ];
x(292,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.4+.6 rand
rand  ];
x(293,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.4+.6 rand
rand  ];
x(294,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.4+.6 rand
rand  ];
x(295,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.4+.6 rand
rand  ];
x(296,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand*4+4
rand  ];
x(297,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand*4+4
rand  ];
x(298,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand*4+4
rand  ];
x(299,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand*4+4
rand  ];
x(300,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand*4+4
rand  ];
x(301,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
4*rand+4];
x(302,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
4*rand+4];
x(303,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
4*rand+4];
x(304,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
4*rand+4];
x(305,:)= [5   5+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
4*rand+4];
x(306,:)= [5   5+rand*5  6.5+rand*3  rand*.2    rand*.2    rand
rand  ];
x(307,:)= [5   5+rand*5  6.5+rand*3  rand*.2    rand*.2    rand
rand  ];
x(308,:)= [5   5+rand*5  6.5+rand*3  rand*.2    rand*.2    rand
rand  ];
x(309,:)= [5   5+rand*5  6.5+rand*3  rand*.2    rand*.2    rand
rand  ];
x(310,:)= [5   5+rand*5  6.5+rand*3  rand*.2    rand*.2    rand
rand  ];
x(311,:)= [10   rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
x(312,:)= [10   rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
x(313,:)= [10   rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
x(314,:)= [10   rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
x(315,:)= [10   rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
```

```
x(316,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(317,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(318,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(319,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(320,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(321,:)= [10   rand*5   0.5+rand*3  rand*.4+.6 rand*.3      rand*2
2*rand  ];
x(322,:)= [10   rand*5   0.5+rand*3  rand*.4+.6 rand*.3      rand*2
2*rand  ];
x(323,:)= [10   rand*5   0.5+rand*3  rand*.4+.6 rand*.3      rand*2
2*rand  ];
x(324,:)= [10   rand*5   0.5+rand*3  rand*.4+.6 rand*.3      rand*2
2*rand  ];
x(325,:)= [10   rand*5   0.5+rand*3  rand*.4+.6 rand*.3      rand*2
2*rand  ];
x(326,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.4+.6 rand*2
2*rand  ];
x(327,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.4+.6 rand*2
2*rand  ];
x(328,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.4+.6 rand*2
2*rand  ];
x(329,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.4+.6 rand*2
2*rand  ];
x(330,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.4+.6 rand*2
2*rand  ];
x(331,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*4+4
2*rand  ];
x(332,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*4+4
2*rand  ];
x(333,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*4+4
2*rand  ];
x(334,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*4+4
2*rand  ];
x(335,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*4+4
2*rand  ];
x(336,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*2
4*rand+4];
x(337,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*2
4*rand+4];
x(338,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*2
4*rand+4];
x(339,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*2
4*rand+4];
x(340,:)= [10   rand*5   0.5+rand*3  rand*.3     rand*.3     rand*2
4*rand+4];
x(341,:)= [10   rand*5   6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(342,:)= [10   rand*5   6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
```

```
x(343,:)= [10   rand*5    6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(344,:)= [10   rand*5    6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(345,:)= [10   rand*5    6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(346,:)= [10  5+rand*5   0.5+rand*3  rand*.7+.3  rand*.7+.3  rand*2
2*rand  ];
x(347,:)= [10  5+rand*5   0.5+rand*3  rand*.7+.3  rand*.7+.3  rand*2
2*rand  ];
x(348,:)= [10  5+rand*5   0.5+rand*3  rand*.7+.3  rand*.7+.3  rand*2
2*rand  ];
x(349,:)= [10  5+rand*5   0.5+rand*3  rand*.7+.3  rand*.7+.3  rand*2
2*rand  ];
x(350,:)= [10  5+rand*5   0.5+rand*3  rand*.7+.3  rand*.7+.3  rand*2
2*rand  ];
x(351,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(352,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(353,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(354,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(355,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
x(356,:)= [10  5+rand*5   0.5+rand*3  rand*.3+.7  rand*.3     rand*2
2*rand  ];
x(357,:)= [10  5+rand*5   0.5+rand*3  rand*.3+.7  rand*.3     rand*2
2*rand  ];
x(358,:)= [10  5+rand*5   0.5+rand*3  rand*.3+.7  rand*.3     rand*2
2*rand  ];
x(359,:)= [10  5+rand*5   0.5+rand*3  rand*.3+.7  rand*.3     rand*2
2*rand  ];
x(360,:)= [10  5+rand*5   0.5+rand*3  rand*.3+.7  rand*.3     rand*2
2*rand  ];
x(361,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3+.7  rand*2
2*rand  ];
x(362,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3+.7  rand*2
2*rand  ];
x(363,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3+.7  rand*2
2*rand  ];
x(364,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3+.7  rand*2
2*rand  ];
x(365,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3+.7  rand*2
2*rand  ];
x(366,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*3+5
2*rand  ];
x(367,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*3+5
2*rand  ];
x(368,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*3+5
2*rand  ];
x(369,:)= [10  5+rand*5   0.5+rand*3  rand*.3     rand*.3     rand*3+5
2*rand  ];
```

```
x(370,:)= [10  5+rand*5  0.5+rand*3  rand*.3     rand*.3     rand*3+5
2*rand  ];
x(371,:)= [10  5+rand*5  0.5+rand*3  rand*.3     rand*.3     rand*2
3*rand+5];
x(372,:)= [10  5+rand*5  0.5+rand*3  rand*.3     rand*.3     rand*2
3*rand+5];
x(373,:)= [10  5+rand*5  0.5+rand*3  rand*.3     rand*.3     rand*2
3*rand+5];
x(374,:)= [10  5+rand*5  0.5+rand*3  rand*.3     rand*.3     rand*2
3*rand+5];
x(375,:)= [10  5+rand*5  0.5+rand*3  rand*.3     rand*.3     rand*2
3*rand+5];
x(376,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(377,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(378,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(379,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(380,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
x(381,:)= [15   rand*5   0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(382,:)= [15   rand*5   0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(383,:)= [15   rand*5   0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(384,:)= [15   rand*5   0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(385,:)= [15   rand*5   0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(386,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
x(387,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
x(388,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
x(389,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
x(390,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
x(391,:)= [15   rand*5   0.5+rand*3  rand*.3+.7 rand*.4     rand*3
3*rand  ];
x(392,:)= [15   rand*5   0.5+rand*3  rand*.3+.7 rand*.4     rand*3
3*rand  ];
x(393,:)= [15   rand*5   0.5+rand*3  rand*.3+.7 rand*.4     rand*3
3*rand  ];
x(394,:)= [15   rand*5   0.5+rand*3  rand*.3+.7 rand*.4     rand*3
3*rand  ];
x(395,:)= [15   rand*5   0.5+rand*3  rand*.3+.7 rand*.4     rand*3
3*rand  ];
x(396,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.3+.7 rand*3
3*rand  ];
```

```
x(397,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.3+.7 rand*3
3*rand  ];
x(398,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.3+.7 rand*3
3*rand  ];
x(399,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.3+.7 rand*3
3*rand  ];
x(400,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.3+.7 rand*3
3*rand  ];
x(401,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3+5
3*rand  ];
x(402,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3+5
3*rand  ];
x(403,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3+5
3*rand  ];
x(404,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3+5
3*rand  ];
x(405,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3+5
3*rand  ];
x(406,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
x(407,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
x(408,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
x(409,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
x(410,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
x(411,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(412,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(413,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(414,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(415,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
x(416,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(417,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(418,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(419,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(420,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
x(421,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*5+3
5*rand+3];
x(422,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*5+3
5*rand+3];
x(423,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*5+3
5*rand+3];
```

```
x(424,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*5+3
5*rand+3];
x(425,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*5+3
5*rand+3];
x(426,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8  rand*.4     rand*3
3*rand  ];
x(427,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8  rand*.4     rand*3
3*rand  ];
x(428,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8  rand*.4     rand*3
3*rand  ];
x(429,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8  rand*.4     rand*3
3*rand  ];
x(430,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8  rand*.4     rand*3
3*rand  ];
x(431,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8  rand*3
3*rand  ];
x(432,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8  rand*3
3*rand  ];
x(433,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8  rand*3
3*rand  ];
x(434,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8  rand*3
3*rand  ];
x(435,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8  rand*3
3*rand  ];
x(436,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+6
3*rand  ];
x(437,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+6
3*rand  ];
x(438,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+6
3*rand  ];
x(439,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+6
3*rand  ];
x(440,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+6
3*rand  ];
x(441,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+6];
x(442,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+6];
x(443,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+6];
x(444,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+6];
x(445,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+6];
x(446,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(447,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(448,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(449,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
x(450,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
```

```
x(451,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
x(452,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
x(453,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
x(454,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
x(455,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
x(456,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4+4
4*rand+4];
x(457,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4+4
4*rand+4];
x(458,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4+4
4*rand+4];
x(459,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4+4
4*rand+4];
x(460,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4+4
4*rand+4];
x(461,:)= [20   rand*5   0.5+rand*3  rand*.2+.8 rand*.5    rand*4
4*rand  ];
x(462,:)= [20   rand*5   0.5+rand*3  rand*.2+.8 rand*.5    rand*4
4*rand  ];
x(463,:)= [20   rand*5   0.5+rand*3  rand*.2+.8 rand*.5    rand*4
4*rand  ];
x(464,:)= [20   rand*5   0.5+rand*3  rand*.2+.8 rand*.5    rand*4
4*rand  ];
x(465,:)= [20   rand*5   0.5+rand*3  rand*.2+.8 rand*.5    rand*4
4*rand  ];
x(466,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.2+.8 rand*4
4*rand  ];
x(467,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.2+.8 rand*4
4*rand  ];
x(468,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.2+.8 rand*4
4*rand  ];
x(469,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.2+.8 rand*4
4*rand  ];
x(470,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.2+.8 rand*4
4*rand  ];
x(471,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*2+6
4*rand  ];
x(472,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*2+6
4*rand  ];
x(473,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*2+6
4*rand  ];
x(474,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*2+6
4*rand  ];
x(475,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*2+6
4*rand  ];
x(476,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4
2*rand+6];
x(477,:)= [20   rand*5   0.5+rand*3  rand*.5    rand*.5    rand*4
2*rand+6];
```

```
x(478,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+6];
x(479,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+6];
x(480,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+6];
x(481,:)= [20   rand*5   6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(482,:)= [20   rand*5   6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(483,:)= [20   rand*5   6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(484,:)= [20   rand*5   6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(485,:)= [20   rand*5   6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
x(486,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5  rand*.5+.5  rand*4
4*rand  ];
x(487,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5  rand*.5+.5  rand*4
4*rand  ];
x(488,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5  rand*.5+.5  rand*4
4*rand  ];
x(489,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5  rand*.5+.5  rand*4
4*rand  ];
x(490,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5  rand*.5+.5  rand*4
4*rand  ];
x(491,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
x(492,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
x(493,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
x(494,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
x(495,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
x(496,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9  rand*.5     rand*4
4*rand  ];
x(497,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9  rand*.5     rand*4
4*rand  ];
x(498,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9  rand*.5     rand*4
4*rand  ];
x(499,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9  rand*.5     rand*4
4*rand  ];
x(500,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9  rand*.5     rand*4
4*rand  ];
x(501,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9  rand*4
4*rand  ];
x(502,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9  rand*4
4*rand  ];
x(503,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9  rand*4
4*rand  ];
x(504,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9  rand*4
4*rand  ];
```

```
x(505,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.1+.9 rand*4
4*rand  ];
x(506,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand+7
4*rand  ];
x(507,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand+7
4*rand  ];
x(508,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand+7
4*rand  ];
x(509,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand+7
4*rand  ];
x(510,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand+7
4*rand  ];
x(511,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand*4
rand+7];
x(512,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand*4
rand+7];
x(513,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand*4
rand+7];
x(514,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand*4
rand+7];
x(515,:)= [20  5+rand*5  0.5+rand*3  rand*.5   rand*.5    rand*4
rand+7];
x(516,:)= [20  5+rand*5  6.5+rand*3  rand*.5   rand*.5    rand*4
4*rand  ];
x(517,:)= [20  5+rand*5  6.5+rand*3  rand*.5   rand*.5    rand*4
4*rand  ];
x(518,:)= [20  5+rand*5  6.5+rand*3  rand*.5   rand*.5    rand*4
4*rand  ];
x(519,:)= [20  5+rand*5  6.5+rand*3  rand*.5   rand*.5    rand*4
4*rand  ];
x(520,:)= [20  5+rand*5  6.5+rand*3  rand*.5   rand*.5    rand*4
4*rand  ];

traindata = x';

%% 40  "good" parameters
%% 200 "ok"   paremeters
%% 280 "bad"  parameters

number_good = 40;
number_ok = 200;
number_bad = 280;
good = ones(1,(number_good))*0.95;
ok = zeros(1,(number_ok));
bad = ones(1,(number_bad))*(-.95);

%good = 0.95;
%ok = 0;
%bad = -0.95;

y = [good ok bad]';

% =       [HP Age(years) DeltaT(C)   RMS-H(g)  RMS-V(g) Max-Min-H(g)
Max-Min-V(g)]
```

```
tx(1,:)  = [5    rand*5    0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(2,:)  = [5    rand*5    0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(3,:)  = [5    rand*5    0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(4,:)  = [5    rand*5    0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(5,:)  = [5    rand*5    0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(6,:)  = [5   5+rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(7,:)  = [5   5+rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(8,:)  = [5   5+rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(9,:)  = [5   5+rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(10,:) = [5   5+rand*5   0.5+rand*3  rand/5     rand/5     rand
rand  ];
tx(11,:) = [10   rand*5    0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(12,:) = [10   rand*5    0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(13,:) = [10   rand*5    0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(14,:) = [10   rand*5    0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(15,:) = [10   rand*5    0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(16,:) = [10  5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(17,:) = [10  5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(18,:) = [10  5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(19,:) = [10  5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(20,:) = [10  5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand*2];
tx(21,:) = [15   rand*5    0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(22,:) = [15   rand*5    0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(23,:) = [15   rand*5    0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(24,:) = [15   rand*5    0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(25,:) = [15   rand*5    0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(26,:) = [15  5+rand*5   0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(27,:) = [15  5+rand*5   0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
```

```
tx(28,:) = [15  5+rand*5  0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(29,:) = [15  5+rand*5  0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(30,:) = [15  5+rand*5  0.5+rand*3  rand/2.5   rand/2.5   rand*3
rand*3];
tx(31,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(32,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(33,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(34,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(35,:) = [20   rand*5   0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(36,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(37,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(38,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(39,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
tx(40,:) = [20  5+rand*5  0.5+rand*3  rand/2     rand/2     rand*4
rand*4];
%  =       [HP Age(years) DeltaT(C)   RMS-H(g)  RMS-V(g) Max-Min-H(g)
Max-Min-V(g)]
tx(41,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
tx(42,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
tx(43,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
tx(44,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
tx(45,:) = [5    rand*5   0.5+rand*3  rand/5+.2  rand/5     rand
rand  ];
tx(46,:) = [5    rand*5   0.5+rand*3  rand/5     rand/5+.2  rand
rand  ];
tx(47,:) = [5    rand*5   0.5+rand*3  rand/5     rand/5+.2  rand
rand  ];
tx(48,:) = [5    rand*5   0.5+rand*3  rand/5     rand/5+.2  rand
rand  ];
tx(49,:) = [5    rand*5   0.5+rand*3  rand/5     rand/5+.2  rand
rand  ];
tx(50,:) = [5    rand*5   0.5+rand*3  rand/5     rand/5+.2  rand
rand  ];
tx(51,:) = [5    rand*5   0.5+rand*3  rand/5     rand/5     rand+1
rand  ];
tx(52,:) = [5    rand*5   0.5+rand*3  rand/5     rand/5     rand+1
rand  ];
tx(53,:) = [5    rand*5   0.5+rand*3  rand/5     rand/5     rand+1
rand  ];
```

```
tx(54,:) = [5    rand*5    0.5+rand*3  rand/5      rand/5      rand+1
rand  ];
tx(55,:) = [5    rand*5    0.5+rand*3  rand/5      rand/5      rand+1
rand  ];
tx(56,:) = [5    rand*5    0.5+rand*3  rand/5      rand/5      rand
rand+1];
tx(57,:) = [5    rand*5    0.5+rand*3  rand/5      rand/5      rand
rand+1];
tx(58,:) = [5    rand*5    0.5+rand*3  rand/5      rand/5      rand
rand+1];
tx(59,:) = [5    rand*5    0.5+rand*3  rand/5      rand/5      rand
rand+1];
tx(60,:) = [5    rand*5    0.5+rand*3  rand/5      rand/5      rand
rand+1];
tx(61,:) = [5    rand*5    3.5+rand*3  rand/5      rand/5      rand
rand  ];
tx(62,:) = [5    rand*5    3.5+rand*3  rand/5      rand/5      rand
rand  ];
tx(63,:) = [5    rand*5    3.5+rand*3  rand/5      rand/5      rand
rand  ];
tx(64,:) = [5    rand*5    3.5+rand*3  rand/5      rand/5      rand
rand  ];
tx(65,:) = [5    rand*5    3.5+rand*3  rand/5      rand/5      rand
rand  ];
tx(66,:) = [5  5+rand*5    0.5+rand*3  rand*.3+.2 rand/5      rand
rand  ];
tx(67,:) = [5  5+rand*5    0.5+rand*3  rand*.3+.2 rand/5      rand
rand  ];
tx(68,:) = [5  5+rand*5    0.5+rand*3  rand*.3+.2 rand/5      rand
rand  ];
tx(69,:) = [5  5+rand*5    0.5+rand*3  rand*.3+.2 rand/5      rand
rand  ];
tx(70,:) = [5  5+rand*5    0.5+rand*3  rand*.3+.2 rand/5      rand
rand  ];
tx(71,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand*.3+.2 rand
rand  ];
tx(72,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand*.3+.2 rand
rand  ];
tx(73,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand*.3+.2 rand
rand  ];
tx(74,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand*.3+.2 rand
rand  ];
tx(75,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand*.3+.2 rand
rand  ];
tx(76,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand/5      rand*2+1
rand  ];
tx(77,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand/5      rand*2+1
rand  ];
tx(78,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand/5      rand*2+1
rand  ];
tx(79,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand/5      rand*2+1
rand  ];
tx(80,:) = [5  5+rand*5    0.5+rand*3  rand/5      rand/5      rand*2+1
rand  ];
```

```
tx(81,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
tx(82,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
tx(83,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
tx(84,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
tx(85,:) = [5   5+rand*5  0.5+rand*3  rand/5     rand/5     rand
2*rand+1];
tx(86,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand ];
tx(87,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand ];
tx(88,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand ];
tx(89,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand ];
tx(90,:) = [5   5+rand*5  3.5+rand*3  rand/5     rand/5     rand
rand ];
tx(91,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand ];
tx(92,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand ];
tx(93,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand ];
tx(94,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand ];
tx(95,:) = [10   rand*5   0.5+rand*3  rand/5+.3  rand*.3    rand*2
2*rand ];
tx(96,:) = [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand ];
tx(97,:) = [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand ];
tx(98,:) = [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand ];
tx(99,:) = [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand ];
tx(100,:)= [10   rand*5   0.5+rand*3  rand*.3    rand/5+.3  rand*2
2*rand ];
tx(101,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand ];
tx(102,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand ];
tx(103,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand ];
tx(104,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand ];
tx(105,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand+2
2*rand ];
tx(106,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
tx(107,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
```

```
tx(108,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
tx(109,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
tx(110,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
rand+2];
tx(111,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(112,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(113,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(114,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(115,:)= [10   rand*5   3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(116,:)= [10 5+rand*5   0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
tx(117,:)= [10 5+rand*5   0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
tx(118,:)= [10 5+rand*5   0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
tx(119,:)= [10 5+rand*5   0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
tx(120,:)= [10 5+rand*5   0.5+rand*3  rand*.3+.3 rand*.3    rand*2
2*rand  ];
tx(121,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
tx(122,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
tx(123,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
tx(124,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
tx(125,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3+.3 rand*2
2*rand  ];
tx(126,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
tx(127,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
tx(128,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
tx(129,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
tx(130,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2+2
2*rand  ];
tx(131,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
tx(132,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
tx(133,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
tx(134,:)= [10 5+rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
```

```
tx(135,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
2*rand+2];
tx(136,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(137,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(138,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(139,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(140,:)= [10  5+rand*5  3.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(141,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
tx(142,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
tx(143,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
tx(144,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
tx(145,:)= [15   rand*5   0.5+rand*3  rand/5+.4  rand*.4    rand*3
3*rand  ];
tx(146,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
tx(147,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
tx(148,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
tx(149,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
tx(150,:)= [15   rand*5   0.5+rand*3  rand*.4    rand/5+.4  rand*3
3*rand  ];
tx(151,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
tx(152,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
tx(153,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
tx(154,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
tx(155,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+3
3*rand  ];
tx(156,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+3];
tx(157,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+3];
tx(158,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+3];
tx(159,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+3];
tx(160,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+3];
tx(161,:)= [15   rand*5   3.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
```

```
tx(162,:)= [15   rand*5   3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
tx(163,:)= [15   rand*5   3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
tx(164,:)= [15   rand*5   3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
tx(165,:)= [15   rand*5   3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
tx(166,:)= [15  5+rand*5  0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
tx(167,:)= [15  5+rand*5  0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
tx(168,:)= [15  5+rand*5  0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
tx(169,:)= [15  5+rand*5  0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
tx(170,:)= [15  5+rand*5  0.5+rand*3  rand*.3+.4  rand*.4     rand*3
3*rand  ];
tx(171,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
tx(172,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
tx(173,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
tx(174,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
tx(175,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.3+.4  rand*3
3*rand  ];
tx(176,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
tx(177,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
tx(178,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
tx(179,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
tx(180,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*2+3
3*rand  ];
tx(181,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+3];
tx(182,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+3];
tx(183,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+3];
tx(184,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+3];
tx(185,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4     rand*3
2*rand+3];
tx(186,:)= [15  5+rand*5  3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
tx(187,:)= [15  5+rand*5  3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
tx(188,:)= [15  5+rand*5  3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
```

```
tx(189,:)= [15  5+rand*5  3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
tx(190,:)= [15  5+rand*5  3.5+rand*3  rand*.4     rand*.4     rand*3
3*rand  ];
tx(191,:)= [20   rand*5   0.5+rand*3  rand/5+.5   rand*.5     rand*4
4*rand  ];
tx(192,:)= [20   rand*5   0.5+rand*3  rand/5+.5   rand*.5     rand*4
4*rand  ];
tx(193,:)= [20   rand*5   0.5+rand*3  rand/5+.5   rand*.5     rand*4
4*rand  ];
tx(194,:)= [20   rand*5   0.5+rand*3  rand/5+.5   rand*.5     rand*4
4*rand  ];
tx(195,:)= [20   rand*5   0.5+rand*3  rand/5+.5   rand*.5     rand*4
4*rand  ];
tx(196,:)= [20   rand*5   0.5+rand*3  rand*.5     rand/5+.5   rand*4
4*rand  ];
tx(197,:)= [20   rand*5   0.5+rand*3  rand*.5     rand/5+.5   rand*4
4*rand  ];
tx(198,:)= [20   rand*5   0.5+rand*3  rand*.5     rand/5+.5   rand*4
4*rand  ];
tx(199,:)= [20   rand*5   0.5+rand*3  rand*.5     rand/5+.5   rand*4
4*rand  ];
tx(200,:)= [20   rand*5   0.5+rand*3  rand*.5     rand/5+.5   rand*4
4*rand  ];
tx(201,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand+4
4*rand  ];
tx(202,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand+4
4*rand  ];
tx(203,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand+4
4*rand  ];
tx(204,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand+4
4*rand  ];
tx(205,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand+4
4*rand  ];
tx(206,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
rand+4];
tx(207,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
rand+4];
tx(208,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
rand+4];
tx(209,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
rand+4];
tx(210,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5     rand*4
rand+4];
tx(211,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
tx(212,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
tx(213,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
tx(214,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
tx(215,:)= [20   rand*5   3.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
```

```
tx(216,:)= [20  5+rand*5  0.5+rand*3  rand*.3+.5 rand*.5    rand*4
4*rand  ];
tx(217,:)= [20  5+rand*5  0.5+rand*3  rand*.3+.5 rand*.5    rand*4
4*rand  ];
tx(218,:)= [20  5+rand*5  0.5+rand*3  rand*.3+.5 rand*.5    rand*4
4*rand  ];
tx(219,:)= [20  5+rand*5  0.5+rand*3  rand*.3+.5 rand*.5    rand*4
4*rand  ];
tx(220,:)= [20  5+rand*5  0.5+rand*3  rand*.3+.5 rand*.5    rand*4
4*rand  ];
tx(221,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.3+.5 rand*4
4*rand  ];
tx(222,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.3+.5 rand*4
4*rand  ];
tx(223,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.3+.5 rand*4
4*rand  ];
tx(224,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.3+.5 rand*4
4*rand  ];
tx(225,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.3+.5 rand*4
4*rand  ];
tx(226,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*2+4
4*rand  ];
tx(227,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*2+4
4*rand  ];
tx(228,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*2+4
4*rand  ];
tx(229,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*2+4
4*rand  ];
tx(230,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*2+4
4*rand  ];
tx(231,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
2*rand+4];
tx(232,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
2*rand+4];
tx(233,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
2*rand+4];
tx(234,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
2*rand+4];
tx(235,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
2*rand+4];
tx(236,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
tx(237,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
tx(238,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
tx(239,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
tx(240,:)= [20  5+rand*5  3.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
%  =      [HP Age(years) DeltaT(C)   RMS-H(g)  RMS-V(g) Max-Min-H(g)
Max-Min-V(g)]
tx(241,:)= [5    rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
```

```
tx(242,:)= [5    rand*5    0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
tx(243,:)= [5    rand*5    0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
tx(244,:)= [5    rand*5    0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
tx(245,:)= [5    rand*5    0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand  ];
tx(246,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
tx(247,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
tx(248,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
tx(249,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
tx(250,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*7+1
7*rand+1];
tx(251,:)= [5    rand*5    0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
tx(252,:)= [5    rand*5    0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
tx(253,:)= [5    rand*5    0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
tx(254,:)= [5    rand*5    0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
tx(255,:)= [5    rand*5    0.5+rand*3  rand/2+.5  rand*.2    rand
rand  ];
tx(256,:)= [5    rand*5    0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
tx(257,:)= [5    rand*5    0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
tx(258,:)= [5    rand*5    0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
tx(259,:)= [5    rand*5    0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
tx(260,:)= [5    rand*5    0.5+rand*3  rand*.2    rand/2+.5  rand
rand  ];
tx(261,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*5+3
rand  ];
tx(262,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*5+3
rand  ];
tx(263,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*5+3
rand  ];
tx(264,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*5+3
rand  ];
tx(265,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand*5+3
rand  ];
tx(266,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand
5*rand+3];
tx(267,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand
5*rand+3];
tx(268,:)= [5    rand*5    0.5+rand*3  rand*.2    rand*.2    rand
5*rand+3];
```

```
tx(269,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand
5*rand+3];
tx(270,:)= [5    rand*5    0.5+rand*3  rand*.2     rand*.2     rand
5*rand+3];
tx(271,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand ];
tx(272,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand ];
tx(273,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand ];
tx(274,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand ];
tx(275,:)= [5    rand*5    6.5+rand*3  rand*.2     rand*.2     rand
rand ];
tx(276,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand ];
tx(277,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand ];
tx(278,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand ];
tx(279,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand ];
tx(280,:)= [5   5+rand*5   0.5+rand*3  rand*.8+.2 rand*.8+.2 rand
rand ];
tx(281,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
tx(282,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
tx(283,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
tx(284,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
tx(285,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.2     rand*7+1
7*rand+1];
tx(286,:)= [5   5+rand*5   0.5+rand*3  rand*.4+.6 rand*.2     rand
rand ];
tx(287,:)= [5   5+rand*5   0.5+rand*3  rand*.4+.6 rand*.2     rand
rand ];
tx(288,:)= [5   5+rand*5   0.5+rand*3  rand*.4+.6 rand*.2     rand
rand ];
tx(289,:)= [5   5+rand*5   0.5+rand*3  rand*.4+.6 rand*.2     rand
rand ];
tx(290,:)= [5   5+rand*5   0.5+rand*3  rand*.4+.6 rand*.2     rand
rand ];
tx(291,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.4+.6 rand
rand ];
tx(292,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.4+.6 rand
rand ];
tx(293,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.4+.6 rand
rand ];
tx(294,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.4+.6 rand
rand ];
tx(295,:)= [5   5+rand*5   0.5+rand*3  rand*.2     rand*.4+.6 rand
rand ];
```

```
tx(296,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand*4+4
rand  ];
tx(297,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand*4+4
rand  ];
tx(298,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand*4+4
rand  ];
tx(299,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand*4+4
rand  ];
tx(300,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand*4+4
rand  ];
tx(301,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand
4*rand+4];
tx(302,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand
4*rand+4];
tx(303,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand
4*rand+4];
tx(304,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand
4*rand+4];
tx(305,:)= [5    5+rand*5  0.5+rand*3  rand*.2     rand*.2     rand
4*rand+4];
tx(306,:)= [5    5+rand*5  6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
tx(307,:)= [5    5+rand*5  6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
tx(308,:)= [5    5+rand*5  6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
tx(309,:)= [5    5+rand*5  6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
tx(310,:)= [5    5+rand*5  6.5+rand*3  rand*.2     rand*.2     rand
rand  ];
tx(311,:)= [10    rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(312,:)= [10    rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(313,:)= [10    rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(314,:)= [10    rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(315,:)= [10    rand*5   0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(316,:)= [10    rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
tx(317,:)= [10    rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
tx(318,:)= [10    rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
tx(319,:)= [10    rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
tx(320,:)= [10    rand*5   0.5+rand*3  rand*.3     rand*.3     rand*6+2
6*rand+2];
tx(321,:)= [10    rand*5   0.5+rand*3  rand*.4+.6 rand*.3     rand*2
2*rand  ];
tx(322,:)= [10    rand*5   0.5+rand*3  rand*.4+.6 rand*.3     rand*2
2*rand  ];
```

```
tx(323,:)= [10   rand*5   0.5+rand*3  rand*.4+.6 rand*.3    rand*2
2*rand  ];
tx(324,:)= [10   rand*5   0.5+rand*3  rand*.4+.6 rand*.3    rand*2
2*rand  ];
tx(325,:)= [10   rand*5   0.5+rand*3  rand*.4+.6 rand*.3    rand*2
2*rand  ];
tx(326,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.4+.6 rand*2
2*rand  ];
tx(327,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.4+.6 rand*2
2*rand  ];
tx(328,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.4+.6 rand*2
2*rand  ];
tx(329,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.4+.6 rand*2
2*rand  ];
tx(330,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.4+.6 rand*2
2*rand  ];
tx(331,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*4+4
2*rand  ];
tx(332,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*4+4
2*rand  ];
tx(333,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*4+4
2*rand  ];
tx(334,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*4+4
2*rand  ];
tx(335,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*4+4
2*rand  ];
tx(336,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
4*rand+4];
tx(337,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
4*rand+4];
tx(338,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
4*rand+4];
tx(339,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
4*rand+4];
tx(340,:)= [10   rand*5   0.5+rand*3  rand*.3    rand*.3    rand*2
4*rand+4];
tx(341,:)= [10   rand*5   6.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(342,:)= [10   rand*5   6.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(343,:)= [10   rand*5   6.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(344,:)= [10   rand*5   6.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(345,:)= [10   rand*5   6.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
tx(346,:)= [10  5+rand*5  0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(347,:)= [10  5+rand*5  0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(348,:)= [10  5+rand*5  0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(349,:)= [10  5+rand*5  0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
```

```
tx(350,:)= [10  5+rand*5  0.5+rand*3  rand*.7+.3 rand*.7+.3 rand*2
2*rand  ];
tx(351,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*6+2
6*rand+2];
tx(352,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*6+2
6*rand+2];
tx(353,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*6+2
6*rand+2];
tx(354,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*6+2
6*rand+2];
tx(355,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*6+2
6*rand+2];
tx(356,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.7 rand*.3    rand*2
2*rand  ];
tx(357,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.7 rand*.3    rand*2
2*rand  ];
tx(358,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.7 rand*.3    rand*2
2*rand  ];
tx(359,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.7 rand*.3    rand*2
2*rand  ];
tx(360,:)= [10  5+rand*5  0.5+rand*3  rand*.3+.7 rand*.3    rand*2
2*rand  ];
tx(361,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.7 rand*2
2*rand  ];
tx(362,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.7 rand*2
2*rand  ];
tx(363,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.7 rand*2
2*rand  ];
tx(364,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.7 rand*2
2*rand  ];
tx(365,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3+.7 rand*2
2*rand  ];
tx(366,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*3+5
2*rand  ];
tx(367,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*3+5
2*rand  ];
tx(368,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*3+5
2*rand  ];
tx(369,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*3+5
2*rand  ];
tx(370,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*3+5
2*rand  ];
tx(371,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
3*rand+5];
tx(372,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
3*rand+5];
tx(373,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
3*rand+5];
tx(374,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
3*rand+5];
tx(375,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand*.3    rand*2
3*rand+5];
tx(376,:)= [10  5+rand*5  6.5+rand*3  rand*.3    rand*.3    rand*2
2*rand  ];
```

```
tx(377,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
tx(378,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
tx(379,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
tx(380,:)= [10  5+rand*5  6.5+rand*3  rand*.3     rand*.3     rand*2
2*rand  ];
tx(381,:)= [15   rand*5   0.5+rand*3  rand*.6+.4  rand*.6+.4  rand*3
3*rand  ];
tx(382,:)= [15   rand*5   0.5+rand*3  rand*.6+.4  rand*.6+.4  rand*3
3*rand  ];
tx(383,:)= [15   rand*5   0.5+rand*3  rand*.6+.4  rand*.6+.4  rand*3
3*rand  ];
tx(384,:)= [15   rand*5   0.5+rand*3  rand*.6+.4  rand*.6+.4  rand*3
3*rand  ];
tx(385,:)= [15   rand*5   0.5+rand*3  rand*.6+.4  rand*.6+.4  rand*3
3*rand  ];
tx(386,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
tx(387,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
tx(388,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
tx(389,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
tx(390,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4    rand*5+3
5*rand+3];
tx(391,:)= [15   rand*5   0.5+rand*3  rand*.3+.7  rand*.4     rand*3
3*rand  ];
tx(392,:)= [15   rand*5   0.5+rand*3  rand*.3+.7  rand*.4     rand*3
3*rand  ];
tx(393,:)= [15   rand*5   0.5+rand*3  rand*.3+.7  rand*.4     rand*3
3*rand  ];
tx(394,:)= [15`  rand*5   0.5+rand*3  rand*.3+.7  rand*.4     rand*3
3*rand  ];
tx(395,:)= [15   rand*5   0.5+rand*3  rand*.3+.7  rand*.4     rand*3
3*rand  ];
tx(396,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.3+.7  rand*3
3*rand  ];
tx(397,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.3+.7  rand*3
3*rand  ];
tx(398,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.3+.7  rand*3
3*rand  ];
tx(399,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.3+.7  rand*3
3*rand  ];
tx(400,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.3+.7  rand*3
3*rand  ];
tx(401,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4     rand*3+5
3*rand  ];
tx(402,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4     rand*3+5
3*rand  ];
tx(403,:)= [15   rand*5   0.5+rand*3  rand*.4     rand*.4     rand*3+5
3*rand  ];
```

```
tx(404,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3+5
3*rand  ];
tx(405,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3+5
3*rand  ];
tx(406,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
tx(407,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
tx(408,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
tx(409,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
tx(410,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
3*rand+5];
tx(411,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
tx(412,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
tx(413,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
tx(414,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
tx(415,:)= [15   rand*5   6.5+rand*3  rand*.4    rand*.4    rand*3
3*rand  ];
tx(416,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
tx(417,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
tx(418,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
tx(419,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
tx(420,:)= [15  5+rand*5  0.5+rand*3  rand*.6+.4 rand*.6+.4 rand*3
3*rand  ];
tx(421,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*5+3
5*rand+3];
tx(422,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*5+3
5*rand+3];
tx(423,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*5+3
5*rand+3];
tx(424,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*5+3
5*rand+3];
tx(425,:)= [15  5+rand*5  0.5+rand*3  rand*.4    rand*.4    rand*5+3
5*rand+3];
tx(426,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8 rand*.4    rand*3
3*rand  ];
tx(427,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8 rand*.4    rand*3
3*rand  ];
tx(428,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8 rand*.4    rand*3
3*rand  ];
tx(429,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8 rand*.4    rand*3
3*rand  ];
tx(430,:)= [15  5+rand*5  0.5+rand*3  rand*.2+.8 rand*.4    rand*3
3*rand  ];
```

```
tx(431,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8 rand*3
3*rand  ];
tx(432,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8 rand*3
3*rand  ];
tx(433,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8 rand*3
3*rand  ];
tx(434,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8 rand*3
3*rand  ];
tx(435,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.2+.8 rand*3
3*rand  ];
tx(436,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*2+6
3*rand  ];
tx(437,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*2+6
3*rand  ];
tx(438,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*2+6
3*rand  ];
tx(439,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*2+6
3*rand  ];
tx(440,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*2+6
3*rand  ];
tx(441,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*3
2*rand+6];
tx(442,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*3
2*rand+6];
tx(443,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*3
2*rand+6];
tx(444,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*3
2*rand+6];
tx(445,:)= [15  5+rand*5  0.5+rand*3  rand*.4     rand*.4    rand*3
2*rand+6];
tx(446,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4    rand*3
3*rand  ];
tx(447,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4    rand*3
3*rand  ];
tx(448,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4    rand*3
3*rand  ];
tx(449,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4    rand*3
3*rand  ];
tx(450,:)= [15  5+rand*5  6.5+rand*3  rand*.4     rand*.4    rand*3
3*rand  ];
tx(451,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(452,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(453,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(454,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(455,:)= [20   rand*5   0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(456,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5    rand*4+4
4*rand+4];
tx(457,:)= [20   rand*5   0.5+rand*3  rand*.5     rand*.5    rand*4+4
4*rand+4];
```

```
tx(458,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
tx(459,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
tx(460,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
tx(461,:)= [20    rand*5    0.5+rand*3  rand*.2+.8 rand*.5     rand*4
4*rand  ];
tx(462,:)= [20    rand*5    0.5+rand*3  rand*.2+.8 rand*.5     rand*4
4*rand  ];
tx(463,:)= [20    rand*5    0.5+rand*3  rand*.2+.8 rand*.5     rand*4
4*rand  ];
tx(464,:)= [20    rand*5    0.5+rand*3  rand*.2+.8 rand*.5     rand*4
4*rand  ];
tx(465,:)= [20    rand*5    0.5+rand*3  rand*.2+.8 rand*.5     rand*4
4*rand  ];
tx(466,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.2+.8 rand*4
4*rand  ];
tx(467,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.2+.8 rand*4
4*rand  ];
tx(468,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.2+.8 rand*4
4*rand  ];
tx(469,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.2+.8 rand*4
4*rand  ];
tx(470,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.2+.8 rand*4
4*rand  ];
tx(471,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*2+6
4*rand  ];
tx(472,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*2+6
4*rand  ];
tx(473,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*2+6
4*rand  ];
tx(474,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*2+6
4*rand  ];
tx(475,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*2+6
4*rand  ];
tx(476,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+6];
tx(477,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+6];
tx(478,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+6];
tx(479,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+6];
tx(480,:)= [20    rand*5    0.5+rand*3  rand*.5     rand*.5     rand*4
2*rand+6];
tx(481,:)= [20    rand*5    6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
tx(482,:)= [20    rand*5    6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
tx(483,:)= [20    rand*5    6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
tx(484,:)= [20    rand*5    6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
```

```
tx(485,:)= [20   rand*5   6.5+rand*3  rand*.5     rand*.5     rand*4
4*rand  ];
tx(486,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(487,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(488,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(489,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(490,:)= [20  5+rand*5  0.5+rand*3  rand*.5+.5 rand*.5+.5 rand*4
4*rand  ];
tx(491,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
tx(492,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
tx(493,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
tx(494,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
tx(495,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4+4
4*rand+4];
tx(496,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9 rand*.5     rand*4
4*rand  ];
tx(497,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9 rand*.5     rand*4
4*rand  ];
tx(498,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9 rand*.5     rand*4
4*rand  ];
tx(499,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9 rand*.5     rand*4
4*rand  ];
tx(500,:)= [20  5+rand*5  0.5+rand*3  rand*.1+.9 rand*.5     rand*4
4*rand  ];
tx(501,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9 rand*4
4*rand  ];
tx(502,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9 rand*4
4*rand  ];
tx(503,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9 rand*4
4*rand  ];
tx(504,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9 rand*4
4*rand  ];
tx(505,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.1+.9 rand*4
4*rand  ];
tx(506,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand+7
4*rand  ];
tx(507,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand+7
4*rand  ];
tx(508,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand+7
4*rand  ];
tx(509,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand+7
4*rand  ];
tx(510,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand+7
4*rand  ];
tx(511,:)= [20  5+rand*5  0.5+rand*3  rand*.5     rand*.5     rand*4
rand+7];
```

```
tx(512,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
rand+7];
tx(513,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
rand+7];
tx(514,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
rand+7];
tx(515,:)= [20  5+rand*5  0.5+rand*3  rand*.5    rand*.5    rand*4
rand+7];
tx(516,:)= [20  5+rand*5  6.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
tx(517,:)= [20  5+rand*5  6.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
tx(518,:)= [20  5+rand*5  6.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
tx(519,:)= [20  5+rand*5  6.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
tx(520,:)= [20  5+rand*5  6.5+rand*3  rand*.5    rand*.5    rand*4
4*rand  ];
%  =      [HP Age(years) DeltaT(C)   RMS-H(g)  RMS-V(g) Max-Min-
H(g) Max-Min-V(g)]
tx(521,:)= [30 10+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
rand  ];
tx(522,:)= [5    rand*5      rand-4.5 rand*.2    rand*.2    rand
rand  ];
tx(523,:)= [10  5+rand*5  0.5+rand*3  rand+2     rand*.3    rand*2
2*rand  ];
tx(524,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand+2     rand*2
2*rand  ];
tx(525,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+12
3*rand  ];
tx(526,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+12];
tx(527,:)= [30 10+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
rand  ];
tx(528,:)= [5    rand*5      rand-4.5 rand*.2    rand*.2    rand
rand  ];
tx(529,:)= [10  5+rand*5  0.5+rand*3  rand+2     rand*.3    rand*2
2*rand  ];
tx(530,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand+2     rand*2
2*rand  ];
tx(531,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+12
3*rand  ];
tx(532,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+12];
tx(533,:)= [30 10+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
rand  ];
tx(534,:)= [5    rand*5      rand-4.5 rand*.2    rand*.2    rand
rand  ];
tx(535,:)= [10  5+rand*5  0.5+rand*3  rand+2     rand*.3    rand*2
2*rand  ];
tx(536,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand+2     rand*2
2*rand  ];
tx(537,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+12
3*rand  ];
```

```
tx(538,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+12];
tx(539,:)= [30 10+rand*5  0.5+rand*3  rand*.2    rand*.2    rand
rand  ];
tx(540,:)= [5    rand*5      rand-4.5 rand*.2    rand*.2    rand
rand  ];
tx(541,:)= [10  5+rand*5  0.5+rand*3  rand+2     rand*.3    rand*2
2*rand  ];
tx(542,:)= [10  5+rand*5  0.5+rand*3  rand*.3    rand+2     rand*2
2*rand  ];
tx(543,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand+12
3*rand  ];
tx(544,:)= [15   rand*5   0.5+rand*3  rand*.4    rand*.4    rand*3
rand+12];

testdata = tx';
```

Multi-Layer Perceptron Neural Network (thesismlp.m)

```
data

save data x y

vecfile = 'data';
numhid = input('How many neurons in the hidden layer? ');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%        Set up default parameters.                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fprintf('\n\nThe default variables are:\n\n');

fprintf('Learning rate = .03\n');
fprintf('Momentum = 0.9\n');
fprintf('Weights stored in "input_file"wt.mat\n');
fprintf('Output MSE error tolerence = 1\n');
fprintf('Errors plotted in progress every 10 epochs.\n');
fprintf('Maximum number of training epochs = 3000.\n');

lr = 0.03;
momentum = 0.9;
weightfile = [vecfile 'wt'];
tolerance = 0.0361;
epoch_disp = 10;
plotflag = 1;
maxepochs=3000;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Load training data.                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

eval(['load ',vecfile])
[nov1, numin] = size(x);    % x are input vectors
[nov2,numout] = size(y);    % y are target vectors

if nov1 ~= nov2
error('The number of input vectors and target vectors has to be the
same.')
end

fprintf('\nThis network has:\n\n');
fprintf(' %0.f input neurons\n',numin);
fprintf(' %0.f neurons in the hidden layer\n',numhid);
fprintf(' %0.f output neurons\n\n',numout);
fprintf('There are %0.f input/output pairs in this training
set.\n\n',nov1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  linearly scaled input                                           %
%  Transpose input/output vectors to conform to MATLAB standard.   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
   [xn, xm, xs] = scale(x);

save mlpunscale xn xm xs

   xn=xn';   % Input vector
   yn=y';   % Target vector

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     Set training parameters.                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

      % Hidden layer selection
  F1 = 'tansig';

F2 = 'tansig'; % Second layer selection
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Initialize weight vectors.                                     %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[W1,B1,W2,B2]=initff(xn,numhid,F1,numout,F2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Train the network using adaptive learning and momentum.      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

tolerance = tolerance*length(x);
TP=[epoch_disp maxepochs tolerance 0.001 0.001 10 .1 1e10];
[W1,B1,W2,B2,epochs,TR]=trainlm(W1,B1,F1,W2,B2,F2,xn,yn,TP);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     Tolerance criteria not met by BP.                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if epochs == maxepochs
   fprintf('\n\n***  Training complete, no answer found!')
   fprintf('\n***  SSE = %f    \n\n',min(TR(1,:)))
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      Tolerance criteria met by BP.                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fprintf('\n\n***  Training complete after %0.f epochs!
***\n',epochs)


save mlpweight W1 W2 B2 B2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%           The end                                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Radial Basis Function Network (thesisrb.m)

```
data

[x,xm,xs] = scale(x);

save data x y
save rbunscale x xm xs

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                %
%  rbtrain:  A program to create and train RBF neural networks.  %
%                                                                %
%  written by J. Wesley Hines                                    %
%  modified by Jeremy J Kovach                                   %
%                                                                %
%  The University of Tennessee                                   %
%                                                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Input parameters.                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

vecfile = 'data';
spread=0.2;
max_neurons=1000;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%        Set up default parameters.         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fprintf('\n\nThe default variables are:\n\n');
fprintf('MSE Output error tolerence = .017689\n');
fprintf('Errors plotted in progress every 2 epochs.\n');

%tolerance is the mean squared error desired.
tolerance = .017689;

epoch_disp = 2;plotflag = 1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     Allow user to change default parameters.   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

choice = input('Are you satisfied with these selections?
[y,n]:','s');

if choice == 'n'
tolerance = input('Output error tolerance (rms): ');
epoch_disp = input('How often to display errors (# of epochs): ');
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Load training data.                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
eval(['load ',vecfile])
[nov1, numin] = size(x);   % x are input vectors
[nov2,numout] = size(y);   % y are target vectors

if nov1 ~= nov2
error('The number of input vectors and target vectors has to be the
same.')
end

fprintf('\nThis network has:\n\n');
fprintf(' %0.f input neurons\n',numin);
fprintf(' %0.f output neurons\n\n',numout);
fprintf('There are %0.f input/output pairs in this training
set.\n\n',nov1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Transpose input/output vectors to conform to MATLAB standard. %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

P=x'; % Input vector
T=y'; % Target vector

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Train RBF Network                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   tolerance = tolerance*length(x);

   TP = [epoch_disp max_neurons tolerance spread];
   [W1,B1,W2,B2,neurons,TR]=solverb(P,T,TP);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     Tolerance criteria not met.      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if TR(neurons) > tolerance
   fprintf('\n\n***  Training complete, no answer found!')
   fprintf('\n***  SSE = %f    \n\n',TR(neurons))
else

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      Tolerance criteria met.              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

   fprintf('***  Training complete, SSE = %f   ***\n',TR(neurons))
   fprintf('***  There are %0.f radbas neurons.  ***\n',neurons)
   fprintf('***  Weights stored rbweights.mat      ***\n')
   eval(['save rbweights W1 W2 B1 B2']);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%     The end                               %
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Probabilistic Neural Network (thesispnn.m)

```
data

y = [1;1;1;1;1;2;2;2;2;2;3;3;3;3;3;4;4;4;4;4;5;5;5;5;5;6;6;6;6;6;
7;7;7;7;7;8;8;8;8;8;9;9;9;9;9;10;10;10;10;10;11;11;11;11;11;
12;12;12;12;12;13;13;13;13;13;14;14;14;14;14;15;15;15;15;15;
16;16;16;16;16;17;17;17;17;17;18;18;18;18;18;19;19;19;19;19;
20;20;20;20;20;21;21;21;21;21;22;22;22;22;22;23;23;23;23;23;
24;24;24;24;24;25;25;25;25;25;26;26;26;26;26;27;27;27;27;27;
28;28;28;28;28;29;29;29;29;29;30;30;30;30;30;31;31;31;31;31;
32;32;32;32;32;33;33;33;33;33;34;34;34;34;34;35;35;35;35;35;
36;36;36;36;36;37;37;37;37;37;38;38;38;38;38;39;39;39;39;39;
40;40;40;40;40;41;41;41;41;41;42;42;42;42;42;43;43;43;43;43;
44;44;44;44;44;45;45;45;45;45;46;46;46;46;46;47;47;47;47;47;
48;48;48;48;48;49;49;49;49;49;50;50;50;50;50;51;51;51;51;51;
52;52;52;52;52;53;53;53;53;53;54;54;54;54;54;55;55;55;55;55;
56;56;56;56;56;57;57;57;57;57;58;58;58;58;58;59;59;59;59;59;
60;60;60;60;60;61;61;61;61;61;62;62;62;62;62;63;63;63;63;63;
64;64;64;64;64;65;65;65;65;65;66;66;66;66;66;67;67;67;67;67;
68;68;68;68;68;69;69;69;69;69;70;70;70;70;70;71;71;71;71;71;
72;72;72;72;72;73;73;73;73;73;74;74;74;74;74;75;75;75;75;75;
76;76;76;76;76;77;77;77;77;77;78;78;78;78;78;79;79;79;79;79;
80;80;80;80;80;81;81;81;81;81;82;82;82;82;82;83;83;83;83;83;
84;84;84;84;84;85;85;85;85;85;86;86;86;86;86;87;87;87;87;87;
88;88;88;88;88;89;89;89;89;89;90;90;90;90;90;91;91;91;91;91;
92;92;92;92;92;93;93;93;93;93;94;94;94;94;94;95;95;95;95;95;
96;96;96;96;96;97;97;97;97;97;98;98;98;98;98;99;99;99;99;99;
100;100;100;100;100;101;101;101;101;101;102;102;102;102;102;
103;103;103;103;103;104;104;104;104;104]';

[xn,xm,xs] = scale(x);

save pnnunscale xn xm xs
```

Multi-Layer Perceptron Test Program (mlptest.m)

```
load mlpunscale.mat
load mlpweight.mat


for i = 1:size(traindata,2)

   pps = traindata(:,i)'.*xm +xs;
   ppsn = pps';
   trial(i) = simuff(ppsn,W1,B1,'tansig',W2,B2,'tansig');

end

for j = 1:size(traindata,2)
   if trial(j) < -0.3;
      trial(j) = -1;
   elseif trial(j) > 0.3;
      trial(j) = 1;
   else trial(j) = 0;
end
end

trial


for i = 1:size(testdata,2)

   pps = testdata(:,i)'.*xm +xs;
   ppsn = pps';
   generalize(i) = simuff(ppsn,W1,B1,'tansig',W2,B2,'tansig');

end

for j = 1:size(testdata,2)
   if generalize(j) < -0.3;
      generalize(j) = -1;
   elseif generalize(j) > 0.3;
      generalize(j) = 1;
   else generalize(j) = 0;
end
end

generalize
```

Radial Basis Function Test Program (rbtest.m)

```
load rbweights.mat
load rbunscale.mat

for i = 1:size(traindata,2)
   pps = traindata(:,i)'.*xm +xs;
   ppsn = pps';
   trial(i) = simurb(ppsn,W1,B1,W2,B2);
end

for j = 1:size(traindata,2)
   if trial(j) < -0.3;
      trial(j) = -1;
   elseif trial(j) > 0.3;
      trial(j) = 1;
   else trial(j) = 0;
end
end
trial

for i = 1:size(testdata,2)
   pps = testdata(:,i)'.*xm + xs;
   ppsn = pps';
   generalize(i) = simurb(ppsn,W1,B1,W2,B2);
end

out_of_trained_region =  find(abs(generalize-B2)<0.001)

for j = 1:size(testdata,2)
   if generalize(j) < -0.3;
      generalize(j) = -1;
   elseif generalize(j) > 0.3;
      generalize(j) = 1;
   else generalize(j) = 0;
end
end
generalize
```

Probabilistic Neural Network Test Program (pnntest.m)

```
load pnnunscale.mat

for i = 1:size(traindata,2)

   pps = traindata(:,i)'.*xm +xs;
   a = 0.25;
   classes = 104;

   [trial(i),prob] = pnn(xn,y,classes,pps,a);
   if trial(i) < 9
      trial(i) = 1;
   elseif trial(i) > 48
      trial(i) = -1;
   else trial(i) = 0;
end

end

trial

for i = 1:size(testdata,2)

   pps = testdata(:,i)'.*xm +xs;
   a = 0.25;
   classes = 104;

   [generalize(i),prob] = pnn(xn,y,classes,pps,a);
   if generalize(i) < 9
      generalize(i) = 1;
   elseif generalize(i) > 48
      generalize(i) = -1;
   else generalize(i) = 0;
end
end
generalize
```

Fuzzy Logic System (thesisfls.m)

```
%      Jeremy J Kovach
%      410-49-7488
%      Prediction of Impending Mechanical Seal Failure
%      Thesis
%      07-15-99


% Average Temperature Difference [degrees C] (0.50 - 9.50)
% avgtemp = Universe of Discourse for Average Temperature
% ATL, ATM, ATH = Membership functions for low, medium, and high
average temps
clear all

avgtemp = [0.5:0.01:9.5];
ATL = mf_gauss(avgtemp,[0.5 2.0]);
ATM = mf_gauss(avgtemp,[5 1.6]);
ATH = mf_gauss(avgtemp,[9.5 2.0]);
comb_avgtemp = [ATL;ATM;ATH];

% Age of Mechanical Seal [years] (0.00 - 10.00)
% age = Universe of Discourse for Age
% AL, AH = Membership functions for low and high ages

age = [0:0.01:10];
AL = triangle(age,[0 0 10]);
AH = triangle(age,[0 10 10]);
comb_age = [AL;AH];

% Maximum-Minimum Vibrational Horizontal Acceleration [g] (0.00 -
8.00)
% maxvibh = Universe of Discourse for Horizontal Vibration
% MVHL, MVHM, MVHH = Membership functions for low, medium, and high
vibrations

maxvibh = [0:.01:8];
MVHL = mf_gauss(maxvibh,[0 1.7]);
MVHM = mf_gauss(maxvibh,[4 1.5]);
MVHH = mf_gauss(maxvibh,[8 1.7]);
comb_maxvibh = [MVHL;MVHM;MVHH];

% Maximum-Minimum Vibrational Vertical Acceleration [g] (0.00 -
8.00)
% maxvibv = Universe of Discourse for Vertical Vibration
% MVVL, MVVM, MVVH = Membership functions for low, medium, and high
vibrations

maxvibv = [0:.01:8];
MVVL = mf_gauss(maxvibv,[0 1.7]);
MVVM = mf_gauss(maxvibv,[4 1.5]);
MVVH = mf_gauss(maxvibv,[8 1.7]);
comb_maxvibv = [MVVL;MVVM;MVVH];

% RMS Power Vibrational Horizontal Acceleration [g] (0.00 - 1.00)
```

```
% rmsh = Universe of Discourse for Horizontal RMS Vibration
% RMSHL, RMSHM, RMSHH = Membership functions for low, medium, and
high RMS power

rmsh = [0:.01:1];
RMSHL = mf_gauss(rmsh,[0 0.25]);
RMSHM = mf_gauss(rmsh,[.5 .15]);
RMSHH = mf_gauss(rmsh,[1 0.25]);
comb_rmsh = [RMSHL;RMSHM;RMSHH];

% RMS Power Vibrational Vertical Acceleration [g] (0.00 - 1.00)
% rmsv = Universe of Discourse for Vertical RMS Vibration
% RMSVL, RMSVM, RMSVH = Membership functions for low, medium, and
high RMS power

rmsv = [0:.01:1];
RMSVL = mf_gauss(rmsv,[0 0.25]);
RMSVM = mf_gauss(rmsv,[.5 .15]);
RMSVH = mf_gauss(rmsv,[1 0.25]);
comb_rmsv = [RMSVL;RMSVM;RMSVH];

% Horsepower of the motor [hp] (5, 10, 15, 20)
% hp = Universe of Discourse for power
% hp5, hp10, hp15, hp20 = Membership functions for horsepower

hp = [0:0.01:25];
hp5  = triangle(hp,[4.97 5 5.03]);
hp10 = triangle(hp,[9.97 10 10.03]);
hp15 = triangle(hp,[14.97 15 15.03]);
hp20 = triangle(hp,[19.97 20 20.03]);
comb_hp = [hp5;hp10;hp15;hp20];

% Seal Performance (0.00 - 1.00)
% SP = Universe of Discourse for Seal Performance
% SPL, SPM, SPH = Membership Functions for low, med. and high seal
Performance

SP = [0:0.01:1];
SPL = mf_gauss(SP,[0 .2]);
SPM = mf_gauss(SP,[.5 .2]);
SPH = mf_gauss(SP,[1 .2]);
comb_sp = [SPL;SPM;SPH];

horsepower = input('Enter the Horsepower of the Pump Motor [hp]
(5,10,15,20): ');
sealage = input('Enter the Age of the Mechanical Seal [year] (0.00 -
10.00): ');
avgtemperature = input('Enter the Average Temperature Difference [C]
(0.50 - 9.50): ');
rmspowerh = input('Enter the RMS Horizontal Power of the Vibration
[g] (0.00 - 1.00): ');
rmspowerv = input('Enter the RMS Vertical Power of the Vibration [g]
(0.00 - 1.00): ');
maxvibrationh = input('Enter the Maximum-Minimum Horizontal
Vibration [g] (0.00 - 8.00): ');
```

```
maxvibrationv = input('Enter the Maximum-Minimum Vertical Vibration
[g] (0.00 - 8.00): ');

% Fuzzifying the inputs
% dofhp = degree of fulfillment of average temperature
% dofage = degree of fulfillment of mechanical seal age
% doftemp = degree of fulfillment of temperature difference
% dofrmsh = degree of fulfillment of horizontal RMS power
% dofrmsv = degree of fulfillment of vertical RMS power
% dofmaxh = degree of fulfillment of horizontal max-min vibration
% dofmaxv = degree of fulfillment of vertical max-min vibration

dofhp = interp1(hp',comb_hp',horsepower')';
dofage = interp1(age',comb_age',sealage')';
doftemp = interp1(avgtemp',comb_avgtemp',avgtemperature')';
dofrmsh = interp1(rmsh',comb_rmsh',rmspowerh')';
dofrmsv = interp1(rmsv',comb_rmsv',rmspowerv')';
dofmaxh = interp1(maxvibh',comb_maxvibh',maxvibrationh')';
dofmaxv = interp1(maxvibv',comb_maxvibv',maxvibrationv')';

DOF1 = dofhp;
DOF2 = dofage;
DOF3 = doftemp;
DOF4 = dofrmsh;
DOF5 = dofrmsv;
DOF6 = dofmaxh;
DOF7 = dofmaxv;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%




antecedent_DOF = [
DOF3(3)
min([DOF6(3),DOF7(3)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(2),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(2)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(3),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(1),DOF5(3),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(3),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(3)])
min([DOF1(1),DOF2(1),DOF3(3),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(2),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(2)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(3),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(1),DOF5(3),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(3),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(3)])
min([DOF1(1),DOF2(2),DOF3(3),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(2),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(2)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(3),DOF5(1),DOF6(1),DOF7(1)])
```

```
min([DOF1(2),DOF2(1),DOF3(1),DOF4(1),DOF5(3),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(3),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(3)])
min([DOF1(2),DOF2(1),DOF3(3),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(2),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(2)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(3),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(1),DOF5(3),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(3),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(3)])
min([DOF1(2),DOF2(2),DOF3(3),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(2),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(2)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(3),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(1),DOF5(3),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(3),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(3)])
min([DOF1(3),DOF2(1),DOF3(3),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(2),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(2)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(3),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(1),DOF5(3),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(3),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(3)])
min([DOF1(3),DOF2(2),DOF3(3),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(2),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(2)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(3),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(1),DOF5(3),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(3),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(3)])
min([DOF1(4),DOF2(1),DOF3(3),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(2),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(2)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(3),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(1),DOF5(3),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(3),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(3)])
min([DOF1(4),DOF2(2),DOF3(3),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(2),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(1),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(2)])
min([DOF1(1),DOF2(1),DOF3(2),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(2),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(1),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(2)])
min([DOF1(1),DOF2(2),DOF3(2),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(2),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(1),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(2)])
min([DOF1(2),DOF2(1),DOF3(2),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(2),DOF5(1),DOF6(1),DOF7(1)])
```

```
min([DOF1(2),DOF2(2),DOF3(1),DOF4(1),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(2)])
min([DOF1(2),DOF2(2),DOF3(2),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(2),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(1),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(2)])
min([DOF1(3),DOF2(1),DOF3(2),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(2),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(1),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(2)])
min([DOF1(3),DOF2(2),DOF3(2),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(2),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(1),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(2)])
min([DOF1(4),DOF2(1),DOF3(2),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(2),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(1),DOF5(2),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(2),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(2)])
min([DOF1(4),DOF2(2),DOF3(2),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(1),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(2),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(3),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(1),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])
min([DOF1(4),DOF2(2),DOF3(1),DOF4(1),DOF5(1),DOF6(1),DOF7(1)])];      %good

consequent = [comb_sp(1,:) %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad     total of 56 bad cases
comb_sp(1,:)          %bad
```

```
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(1,:)          %bad
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
```

```
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok       total of 40 ok cases
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(2,:)          %ok
comb_sp(3,:)          %good
comb_sp(3,:)          %good
comb_sp(3,:)          %good
comb_sp(3,:)          %good
comb_sp(3,:)          %good     total of 8 good cases
comb_sp(3,:)          %good
comb_sp(3,:)          %good
comb_sp(3,:)];        %good

Consequent = product(consequent,antecedent_DOF);
aggregation = max(Consequent)/max(sum(Consequent));
output = centroid(SP,aggregation);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Plotting the graphs                                    %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

plot(hp,comb_hp)
title('5, 10, 15, and 20 Horsepowers')
xlabel('Motor Horsepower [hp]')
ylabel('Membership')
axis([0 25 0 1.1])

plot(age,comb_age)
title('New and Old Mechanical Seals')
xlabel('Age [Years]')
ylabel('Membership')

plot(avgtemp,comb_avgtemp)
title('Cool, Moderate, and Hot Average Temperature Differences')
xlabel('Temperature [C]')
ylabel('Membership')
```

```
plot(rmsh,comb_rmsh)
title('Light, Moderate, and High RMS Horizontal Power')
xlabel('RMS Power [g]')
ylabel('Membership')

plot(rmsv,comb_rmsv)
title('Light, Moderate, and High RMS Vertical Power')
xlabel('RMS Power [g]')
ylabel('Membership')

plot(maxvibh,comb_maxvibh)
title('Low, Moderate, and High Maximum-Minimum Horizontal Vibration
[g]')
xlabel('Vibration [g]')
ylabel('Membership')

plot(maxvibv,comb_maxvibv)
title('Low, Moderate, and High Maximum-Minimum Vertical Vibration
[g]')
xlabel('Vibration [g]')
ylabel('Membership')


plot(SP,comb_sp)
title('Low, Moderate, and High Seal Performance')
xlabel('Seal Performance')
ylabel('Membership')

plot(SP,Consequent)
axis([min(SP) max(SP) 0 1.0])
title('Consequent of Fuzzy Rules')
xlabel('Seal Performance Scale')
ylabel('Membership')

plot(SP,aggregation,'r')
axis([min(SP) max(SP) 0 1.0])
title('Aggregation of Fuzzy Rule Outputs')
xlabel('Seal Performance Scale')
ylabel('Membership')

c_plot(SP,aggregation,output,'Crisp Output Value for Performance')
axis([min(SP) max(SP) 0 1.0])
xlabel('Seal Performance')

effective_output = (output-0.1564)/.6658;

if effective_output <0.001
   effective_output=0;
end
effective_output

c_plot(SP,aggregation,effective_output,'Effective Output Value for
Performance')
axis([min(SP) max(SP) 0 1.0])
xlabel('Seal Performance')
```

VITA

Jeremy J Kovach (JJ) was born in Memphis, Tennessee on July 16, 1975 and grew up in and around the suburbs of Memphis. He attended Germantown and Mt. Pisgah Elementary schools before entering Houston High School. He graduated from Houston High School in Germantown, Tennessee in 1993.

JJ entered the University of Tennessee in the fall semester of 1993 seeking a Bachelor of Science degree in Mechanical Engineering, which he obtained in May of 1998. He enrolled in the graduate school at the University of Tennessee later that fall in pursuit of a Master of Science degree in Mechanical Engineering, with a concentration in thermal science. During his graduate work, the author was employed by the Maintenance and Reliability Center at UT. Through this assistanceship, the research and knowledge required to successfully complete this thesis was attained.

The author was an active member of Lambda Chi Alpha Fraternity, as well as the Student Engineering Council, the American Society of Heating, Refrigeration, and Air Conditioning Engineers, The Society of Automotive Engineers, and the Engineering Peer Mentor Program.