DEVELOPMENT OF ARTIFICIAL NEURAL NETWORK-BASED CLASSIFIERS TO IDENTIFY MILITARY IMPULSE NOISE

by

Brian Arthur Bucci

BS, Mechanical Engineering, University of Pittsburgh, 2005

Submitted to the Graduate Faculty of

School of Engineering in partial fulfillment

of the requirements for the degree of

Master of Science in Mechanical Engineering

University of Pittsburgh

2007

UNIVERSITY OF PITTSBURGH

SCHOOL OF ENGINEERING

This thesis was presented

by

Brian Arthur Bucci

It was defended on

October 17, 2007

and approved by

Jeffrey S. Vipperman, Associate Professor, Department of Mechanical Engineering and
Materials Science, Associate Professor, Department of Bioengineering

Mark Miller, Associate Professor, Department of Mechanical Engineering and Materials Science,
Associate Professor of Orthopedic Surgery, Drexel University

Daniel Cole, Assistant Professor, Department of Mechanical Engineering and Materials Science

Thesis Advisor: Jeffrey S. Vipperman, Associate Professor, Department of Mechanical
Engineering and Materials Science, Associate Professor, Department of Bioengineering

DEVELOPMENT OF ARTIFICIAL NEURAL NETWORK-BASED CLASSIFIERS TO IDENTIFY MILITARY IMPULSE NOISE

Brian Arthur Bucci, MS

University of Pittsburgh, 2007

Noise monitoring stations are in place around some military installations, to provide records that assist in processing noise complaints and damage claims. However, they are known to produce false positives and miss many impulse events. In this thesis, classifiers based on artificial neural networks were developed to improve the accuracy of military impulse noise identification. Two time-domain metrics, kurtosis and crest factor, and two custom frequency-domain metrics, spectral slope and weighted square error, were selected as inputs to the artificial neural networks. A separate effort attempted to identify military impulse noise by the shape of the recorded waveform. The classification algorithm achieved up to 100% accuracy on the training data and the validation data.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

1.0 INTRODUCTION

There is a need for an accurate method of recording impulse noise around military installations in order to monitor the impact on the surrounding communities and corroborate noise complaints and damage claims ("Joint Land Use Study", 1993, Luz, 2004, Luz, 1980). As urban environments develop around military installations, these concerns have hindered military exercises that produce high levels of impulse noise. By definition, military impulse noise consists of weapons firing projectiles larger than 20mm and all types of high explosive charges (e.g. demolitions, hand grenades, etc.). To maintain a good relationship with the surrounding community, steps have been taken to control the production of military impulse noise through computer modeling of sound propagation on ranges (Pater, 1999) and changes in training procedures (Pater, 1998). However, sound propagation models (USACHPPM) are not perfect and variations in terrain can further complicate sound level estimations (ANSI, 1996). It is also well known that atmospheric conditions influence the propagation of sound (ANSI, 1996) so, exercises are scheduled to minimize the disturbance to the surrounding community (Russell, 2001).

Since sound propagation can be such a complex problem and it is necessary to produce a record of noise levels in the event of a complaint or damage claim, monitoring stations are in place around many installations in an attempt keep a record of events that exceed a specified threshold. These systems have evolved into an on-line monitoring system that continuously

monitored C-weighted noise levels. C-weighted noise levels approximate the human ear's response to a 90 phon loudness level. This useful is in estimating the human response to high amplitude military impulse noise. When levels exceeded 115 dBC, an integration unit would switch on to measure sound exposure level (SEL), while the system also recorded the time, duration, and peak level. Since wind noise (Blevins, 2001) and blast noise (Benson, 1996, Attias et. al, 2004)) have predominately low-frequency content (<100 Hz), an unusually high amount (90%) of false-positive events were recorded, owing largely to the poor performance of windscreens at low-frequency. An improved, second-generation of monitoring stations operates on an algorithm that expects an impulse event to have a waveform much like that of a classic Friedlander wave (ANSI, 1993) and complying with a set of temporal conditions. Once again this system requires the peak sound levels in excess of 115 dBC to be triggered thus, any military impulse events with peak levels below 115 dBC are not considered. The detection threshold is set to this level in attempt to mitigate false positive detections due to wind noise. When impulse events have very high amplitude (>119 dBC) (ISO, 1996) and conditions are ideal (location of event and sensor (Schomer and Attenborough, 2005), weather patterns, absence of multi-path propagation effects (Chunchuzov et. al, 2005), and timing of impulse events) this approach can work quite well. Unfortunately, these stations marginally detect lower-amplitude impulse events (function reliably with peak levels above 119dBC) and still suffer from some false positive detection of impulse noise (as high as 10%). Most false positive detections are purportedly caused by wind noise, which can always pose a problem in an outdoor noise monitoring situation.

This describes an algorithm to detect military impulse noise in a civilian environment. An overview of the previous body of work is given to familiarize the reader with the current state

of technology. A history and derivation of various neural network structures is also given to introduce these topics. Finally the construction of the classifier is presented.

2.0 MILITARY IMPULSE NOISE MONITORING

The aim of this research is to develop a more accurate noise classifier that is able to also detect events with much lower peak levels (down to 100 dBC). Earlier efforts have proved effective in detection of snipers and other small arms fire (Liaw and Berger, 1998 and 2002, Danicki, 2006, Maroti et. al, 2004, Stoughton, 1997, Safety Dynamics). These projects are different from large ordinance monitors because the spectral content and propagation characteristics of small arm are not the same as those of large ordinance. Efforts have also been made in classification for urban battlefield type settings (Young, 2001). These cases differ from the civilian environment detection of military impulse noise in that they often use more complex systems, having an array of microphones, in order to precisely triangulate the location of the event while in motion.

The most common problems in identifying military impulse noise in a civilian type setting are misclassifications that arise due to wind noise. This is because an explosion and wind rushing over the diaphragm of a microphone often "sound" the same to a rudimentary detection algorithm. Several efforts have been made to mitigate the effects of wind noise while still detecting military impulse noise.

The use of wind screens on microphones and computing the correlation of the signals received at two separate microphones proved to be an effective method to reduce wind noise in blast noise measurements (Schomer, 1990). A wind screen placed over a measurement microphone reduced wind noise by 30 dBC. Additionally, since wind is theoretically made up of

4

numerous random vortices, then two acoustic measurements of wind noise conducted at different point in space should be theoretically uncorrelated. An algorithm utilizing this concept can filter out wind noise from acoustic measurements. This algorithm showed a 22 dBC reduction in wind noise. The two methods used in conjunction were able to reduce wind noise by 33 dBC.

In 1996, Benson investigated the use of correlation between two acoustic measurements as a means of rejecting wind noise from measurements containing possible blast noise (Benson, 1996). Although it was found that the correlation between two measurements did offer some insight into whether an event was blast noise or wind noise, it was determined that a detector based solely on this property would not provide adequate accuracy in the detection of blast noise and rejection of wind noise. Benson then analyzed the spectral content of both blast noise and wind noise. It was found that blast noise typically contained the greatest energy between 10 and 50 Hz, while wind noise typically contained the most energy below 10 Hz. Data were sampled at 2,048 Hz. From this data a 128 point FFT was computed across a 768 point window of the data. An "energy function" was then computed as,

$$EF(x) = \sum_{i=4}^{5} F_a(i)^2 F_b(i)^2 \qquad\qquad (2.1)$$

where i is the frequency bin, (in this case 16 and 32 Hz), of the FFT and F_a and F_b are the FFTs of the respective measurement locations, a and b, for a given window of the recorded waveform. This method served to combine the methods of spectral analysis and correlation functions. Event detection rates of up to 99.2% and false detection rates of 0.12% were claimed in a Monte Carlo simulation of the classifier. A field study was conducted at Fort Riley, Kansas, where winds averaged 6.5 mph. Two microphones were vertically spaced 1.5 and 5.5 m from the ground. The triggering threshold of the system was set to 100 dB. Of the 62 blast events observed all

5

were detected by the algorithm. Additionally 239 wind events were also recorded. The system misclassified 9 of the wind events as blast noise, a false positive rate of 3.7%.

The report CERL Noise Monitoring and Warning System (Sachs, Benson, Schomer, 1998) describes a full implementation of the concepts presented in Benson's paper. In this report, the need for an upgraded monitoring system based on the availability of replacement parts for older units and recent technological advancements is presented. In the case of servicing old units it had become very difficult to obtain comparable replacement parts for the stations originally commissioned in 1986. Additionally, higher performance components and communication links had been developed since the inception of the noise monitoring efforts. This new monitoring station appeared somewhat similar to the original model but, included a wind meter and faster modem. The faster modem enables more data collection and ability to provide software updates transmitted from a central location. The addition of the wind meter allows for suppression of data collection in the case of very windy conditions. Much like the first system this system utilizes two measurement microphones and the computed cross product of their signals. Data is sampled at 48 kHz and decimated to 2 kHz. For this system to register an impulse event, the following conditions must be satisfied: a C weighted peak exceeding 95 dB must be registered, a flat weighted peak of at least 100 dB must be registered, a cross correlation of at least 70% between the two measurement microphones must be computed for the duration of the event, and the spectral conditions proposed in Benson's paper must be satisfied. The detection characteristics are consistent with Benson's work.

In later work by Schomer a fuzzy classifier called *Expert* is described (Schomer et. al, 2000). The system produces a decision based on the combination of the output of five fuzzy tests. Test 1 is the *Self Test*. Based on the weapon being fired and the distance from the noise

source and the monitoring station, this test assigns a likelihood to an event being measured by a noise monitoring station. This test determines the extent to which a measurement can be trusted. The second test is the peak level minus C-weighted sound exposure level (L_{pk}−CSEL) test. L_{pk} is the peak sound level pressure, which is subtracted from CSEL, which is C-weighted sound exposure level. The output of the test is based on the ratio of the measured L_{pk}−CSEL compared to the theoretical L_{pk}−CSEL where, the theoretical L_{pk}−CSEL is defined as

$$P = 25 + D + W , \qquad\qquad (2.2)$$

where D is,

$$D = 2\log_2\left(\frac{1600}{d}\right), \qquad\qquad (2.3)$$

for which d is the distance from the source in meters. W is equal to zero for charges greater than the equivalent of 1.13kg of TNT and equal to

$$W = \log_2\left(\frac{2.5}{w}\right), \qquad\qquad (2.4)$$

where w is the weight of the charge, when the charge weight is less than the equivalent of 1.13kg of TNT. The third test is the *Duration Test*. The duration of a military impulse event is typically less than 2 seconds. From the calculated duration of the event a fuzzy output of the likelihood that an event is military impulse noise is computed. Test 4 is known as the *Windy Test*. This test is based on the number of sound pressure events exceeding the measurement threshold in a given period of time. Since military impulse events typically occur in isolated instances, in training exercises, a large number of events observed in a short period of time often indicates windy, false detection prone conditions. The fifth and final test is the *Source-microphone Test*. This test examines the network of noise monitoring stations and determines which stations are most likely to observe an event based on the location of the noise source. Stations closer to a military

impulse event are more likely to register the event. Conversely, an event observed at a far off station and no event registered at a nearby station may indicate a false positive detection. Weights are assigned to each of the five tests and the combined output is taken to be the decision of the classification algorithm. It is difficult to quantify the accuracy of this system and this algorithm requires many input parameters but, it does present a novel classification method for identifying military impulse noise that could be potentially computationally efficient.

In most of the previous body of work, related to this problem, the use of more than one measurement microphone is advocated. Additionally, some classification methods must use inputs of the specifics dealing with the training exercises to be conducted over a given period of time. The current state of the technology deployed around many military installations offers only one measurement microphone. Thus a system that uses a single measurement would be desirable. Also, unless the facility has a more advanced fire control system, it is often difficult to determine, within a reasonable time frame, which positions are firing and where the impacts will occur. Thus, in the interest of cost savings, it would be of interest to develop an algorithm that could be implemented as simple software update to existing systems, with minimal hardware updates. The algorithm would function autonomously, with little need for user input, and simply "listen" for military impulse events. The algorithm would also provide easily interpretable output that could be corroborated from what is known of the firing records. Although the algorithm could be implemented in it simplest form, (logging impulse events detected), it could also be combined with more advanced control operations and monitoring systems as they become available.

3.0 SIGNAL CLASSIFIERS

A classifier is systematic approach of applying rules to a data set to identify classes of data (Abe, 2001, Smith, 1993). Classification rules are developed either by expert analysis and intuition or by intelligent algorithms (Abe, 2001, Smith, 1993). In its most rudimentary form, a classifier consists of raw input data, some form of data preprocessing or feature extraction, and the actual classification algorithm (Smith, 1993, Youssif and Purdy, 2002). Most classifiers can be categorized in one of the following approaches (Jain, 2000): template matching approach, statistical approach, syntactic approach, or neural network approach.

3.1 NON-NEURAL NETWORK CLASSIFICATION METHODS

3.1.1 Template Matching

Template matching involves comparing a pattern directly to a known template (Jain, 2000). The template may be an actual object in the case of analysis of a 2-dimensional pattern. This template may also be derived from a set of training data. The similarity of patterns is often based on the correlation between the input and the template patterns. Template matching can often be computationally expensive and also lacks some generalization capabilities, which is most

9

apparent in the case of image processing. If an image becomes distorted or orientation of the image changes, template matching often fails.

3.1.2 Statistical Method

In a statistical approach to pattern classification a pattern is represented by a set number of features, n, in an n-dimensional space (Jain, 2000). The goal of this approach is to have different patterns occupy compact, disjoint regions in the n-dimensional space. The performance of such a classification system depends on how well the measured set of features can be separated, by class of pattern, when viewed in this n-dimensional space. Depending on the desired characteristics of the classifier, (acceptable accuracy rates for different classes of data), the decision boundaries are defined in this n-dimensional space. An example of a statistical approach to pattern classification is a Bayesian classifier (Bucci and Vipperman, 2007, Kim and Singh, 2005, Abe, 2001). In the Bayesian classifier, the data used to train the classifier are modeled as multi-dimensional statistical distributions. In most cases these distributions are multi-Gaussian in nature. The Bayesian classifier computes the likelihood ratio that a given data point is from a particular class of data. The likelihood ratio is then compared to the determined threshold of test. The threshold of test is a value derived from the prior probabilities of the observation of a particular class of data. This parameter has the effect of tailoring the classifier to be more or less sensitive to the detection of a particular class of data. If the likelihood ratio computed for an input data point with respect to a particular class of data is greater than the threshold of test, then the data point is classified as being part of the given class of data; otherwise, the data point is classified as not being a part of the given class of data.

3.1.3 Syntactic Method

A syntactic approach to pattern classification entails viewing the pattern as a combination of smaller sub-patterns called primitives (Jain, 2000). An analogy drawn to this approach is that of sentences being part of a language and sentences being constructed of letters. The sentences are viewed as patterns, the language that the sentence is written in is viewed as the class of pattern, and the letters composing the sentence are the primitives. The sentence can be classified into a language based on the arrangement or grammar of the primitives. However, it is noticeable with this approach that a very number of combinations of primitives must be investigated thus, very large training sets are necessary and the method can become very computationally expensive.

3.2 ARTIFICIAL NEURAL NETWORK METHOD

An artificial neural network (ANN) is a computational structure. In an ANN computation is performed by multiple processing nodes of set activation function. These processing nodes, or neurons, are interconnected with various strengths, or synaptic weights, to other neurons in the network. With proper configuration this system is capable of approximating any computable function (Abe, 2001, Haykin, 1999).

3.2.1 History of artificial neural networks

The modern concept of artificial neural networks was developed by McCulloch and Pitts (McCulloch and Pitts, 1943) in 1943 in their paper "A logical calculus of the ideas immanent in nervous activity". This paper proposed the McCulloch-Pitts model of a neuron, the hard-limit activation function. McCulloch and Pitts showed it was possible for a network constructed of these neurons to compute any computable function. *The Organization of Behavior* (Hebb 1949), presented the concept of Hebbian learning. This was the first method developed to modify the synaptic weights of a network from a random state to a state where the network could perform useful computation. "The Perceptron: a probabilistic model for information storage," helped to define how information is stored in a network and also introduced the concept of self-organization (Rosenblatt, 1958). This paper also provided the basis for supervised and unsupervised learning in networks. In 1960 Widrow and Hoff published "Adaptive Switching Circuits" (Widrow and Hoff, 1960) and also constructed the Adeline. The Adeline was a hardware implementation of the hard-limit neuron where a number of inputs could be connected to the device and an output of either 1 or −1 could be produced. Widrow and Hoff are also credited with developing the Least Mean Squares (LMS) learning algorithm, in which the synaptic weights of a network are adjusted to minimize the error of the network outputs over the entire set of training patterns. Much of the progress in the development of neural networks was brought to a halt in 1969 with the publishing of Minsky and Papert's book, *Perceptrons* (Minsky and Papert, 1969). In their book they stated, "…our intuitive judgment [is] that the extension [to multilayer perceptrons with hidden layers] is sterile." Although this statement was later proven to be false, at the time it was enough to cut funding from most all neural network projects. The

major advancement occurring in the 1970's was the first demonstration of the concept of self-organization by von der Marlsburg (von der Marlsburg, 1973). In a continuation in this direction of work, Kohonen introduced the concept of self-organizing maps in 1982. In 1986 the back propagation algorithm, an efficient method of training multilayer networks, was first demonstrated by Rumelhart, Hinton, and Williams (Rumelhart et. al, 1986). Although this concept was later traced back to Bryson and Ho's work in 1969 (Bryson and Ho, 1969), it represented the first actual implementation of this concept and first statement disproving the assumption of Minsky and Papert. The invention of the back-propagation algorithm lead to a resurgence of interest in researching the applications of neural networks, particularly multi-layer perceptrons. The latest major development in the field of neural networks was Vapnik's support vector machine in 1992 (Vapnik, 1992). This learning algorithm often offers the simplest and most interpretable training style and often outperforms most other neural network structures.

Given their abilities in making generalization, strength in pattern classification, wealth of easily obtainable data, and limited prior knowledge of the problem, the ANN structure was thought to be the ideal method for creating a noise classifier (Abe, 2001, Cabell et. al, 1992, Kermani et. al, 2005, Cabell et. al, 1998, Bucci and Vipperman, 2006, 2007). In this classification problem, although the features are extracted from a waveform that occurs in time, the features are not time dependent in the global sense. Classification also does not depend on the order or timing of any previous events. This classifier merely produces decisions based on scalar metrics exacted from a set section of a waveform where the sound pressure level exceeds a set value. Thus, this problem is most likened to time-invariant pattern classification problem rather than any type of problem dealing with time series classification or prediction. Accordingly, the ANN structures to be evaluated are selected from structures that have proven

useful in such situations. Since the problem is truly not time dependent, ANN structures containing time delay feedback or other time dependent features (Tsoi and Back, 1997) would not be applicable to this problem. Rather, structures to be tested are common in the engineering field and have worked well in similar static pattern classification applications. Three ANN structures with the desired characteristics were selected for evaluation: Radial Basis Function (RBF) network (Rafiq et. al, 2001, Haykin, 1999, Arain, 1996), a Self-Organizing Map (Haykin, 1999, Kohonen, 1990, Lavine et. al, 2004, Simela et. al, 1999, Grossberg and Bradski, 1999) (SOM), and a feed-forward Multi-Layer Perceptron (MLP) (Rafiq et. al, 2001, Haykin, 1999, Arain, 1996, Skapura, 1996) trained with back propagation (BP). For comparison purposes, the performance of these three classifiers is then compared to that of a least squares classifier to illustrate that the non-linear modeling capabilities of the ANN structures do indeed offer advantages over a linear classifier.

3.2.2 Benefits of using neural networks

There are several benefits to using ANNs to solve problems. Since the activation function of a neuron may be non-linear, ANNs are capable of non-linear modeling (Haykin, 1999). This is particularly useful in the case where a problem space is non-linear. ANNs are inherently adaptive structures. The training of an ANN to a particular problem space is based on the adaptation of the synaptic weights of the ANN. If the problem space in slightly altered it is often a trivial task to retrain the ANN to the new problem space. ANNs are often also very robust and fault tolerant. Even if a part of the network is damaged; for example the values of some of the synaptic waits are corrupted, the remainder of the network can often still provide marginal

14

function. In the case where there is a wealth of easily obtainable data and there is little expert knowledge of the relationships between the parameters of the problem, ANN structures are often the most efficient method of quantifying the problem.

3.2.3 Basic neuron models

As previously mentioned an ANN is made up of multiple neurons, with multiple connections, arranged in various configurations. To comprehend this model it is first necessary to understand the basic neuron model. Figure 1 shows the basic neuron model.
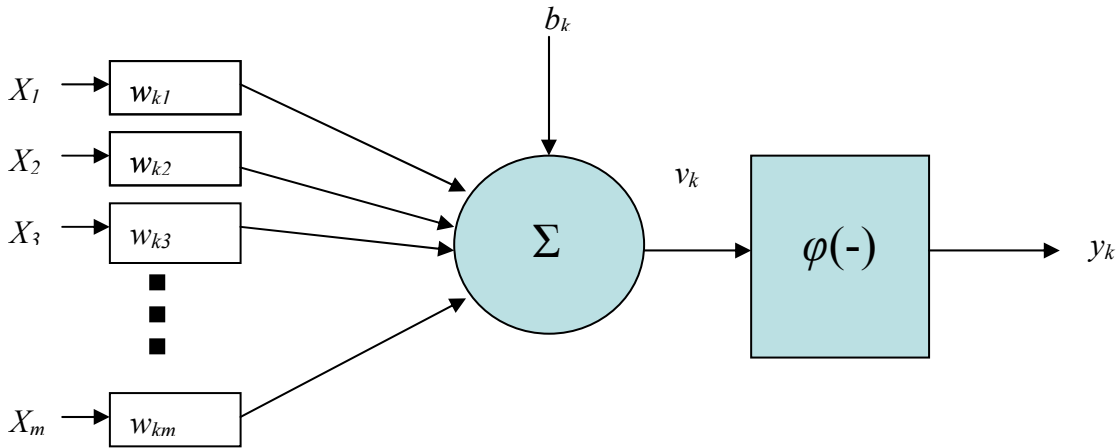
Figure 1. Basic neuron model for kth neuron of network

x_j denotes the jth input parameter (where j goes from 1 to m), w_{kj} is the jth synaptic weight to neuron k, and b_k is a bias value applied to the summing junction. If

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \qquad (3.1)$$

then v_k is computed by,

$$v_k = u_k + b_k. \qquad (3.2)$$

The function $\varphi(\cdot)$ is the activation function of the neuron. The output of the neuron, y_k, is

$$y_k = \varphi(v_k).$$ (3.3)

There are several common activation functions used for neurons. Graphical representations of these functions are provided in Figure 2. The threshold activation function is defined as

$$\varphi(v) = \begin{cases} 1, & \text{if } v \geq 0 \\ 0, & \text{if } v < 0 \end{cases}$$ (3.4)

Piecewise linear functions may also be used as activation functions. An example of a piecewise linear activation function would be

$$\varphi(v) = \begin{cases} 1, & \text{if } \quad v \geq \tfrac{1}{2} \\ v, & \text{if } \quad \tfrac{1}{2} > v > -\tfrac{1}{2} \\ 0, & \text{if } \quad v \leq -\tfrac{1}{2} \end{cases}$$ (3.5)

The most frequently used activation functions are sigmoid based functions. These functions are particularly useful because they exhibit a balance between both linear and non-linear behavior. One such sigmoid function is the logistic function, defined as

$$\varphi(v) = \frac{1}{1 + \exp(-av)}.$$ (3.6)

In this function a is the slope parameter. It can be seen that as a approaches infinity, the logistic function reduces to the threshold function. The logistic function produces values between 0 and 1. If the activation of the neuron is desired to be within the range of -1 to 1, which is desirable in some cases, a sigmoid function based upon the hyperbolic tangent function can be used. Such a function may have the form of

$$\varphi(v) = \tanh(v).$$ (3.7)

The sigmoidal functions are also constantly differentiable, a property that will be useful. There are also neurons with statistical based activation functions. Some of these will be discussed later.
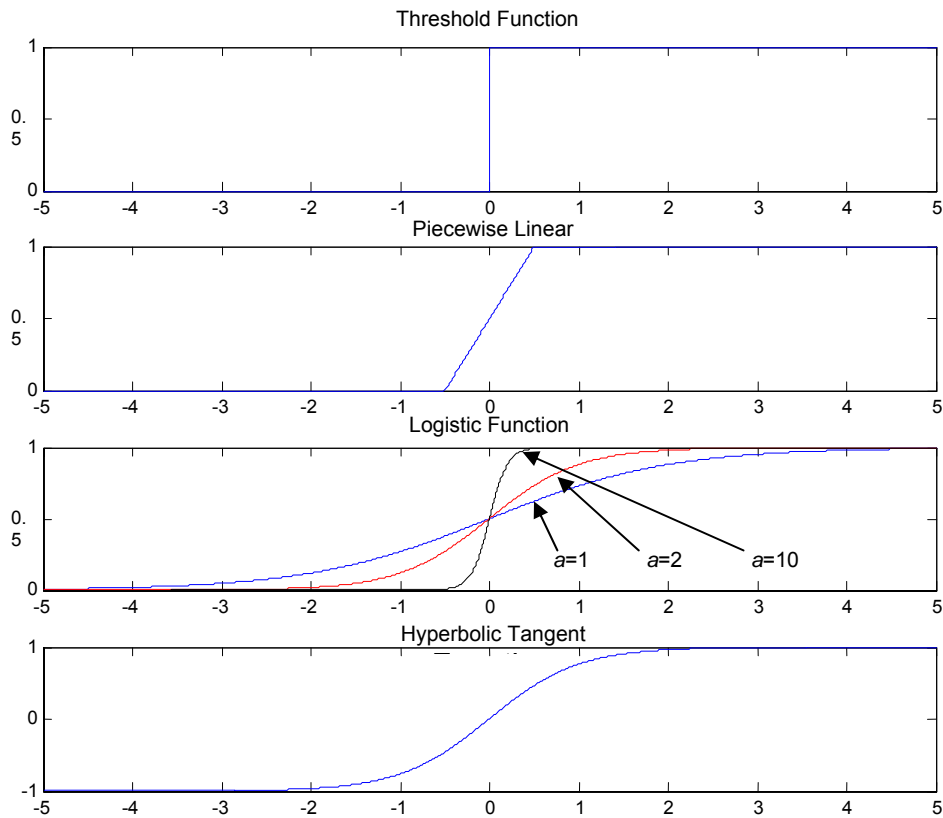
Figure 2. Plot of typical neuron activation functions.

There are two large classifications of conventional ANN structures, feed-forward and recurrent. In a feed-forward ANN data from the input space enter the network and are processed to produce output. This output is not influenced in any way by data that has been input put to the network at any other time. An example of a feed-forward structure is shown in Figure 3.
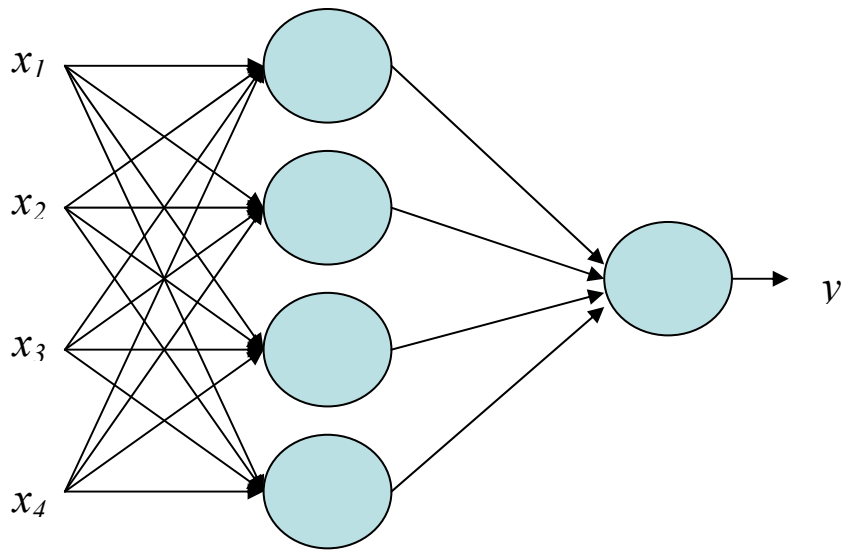
Figure 3. Feed-forward ANN structure

In contrast, a recurrent ANN is one with some sort time step dependent feedback. It is trivial to see that the output, y, in Figure 4 will be influenced by computations occurring within the network at previous time steps.
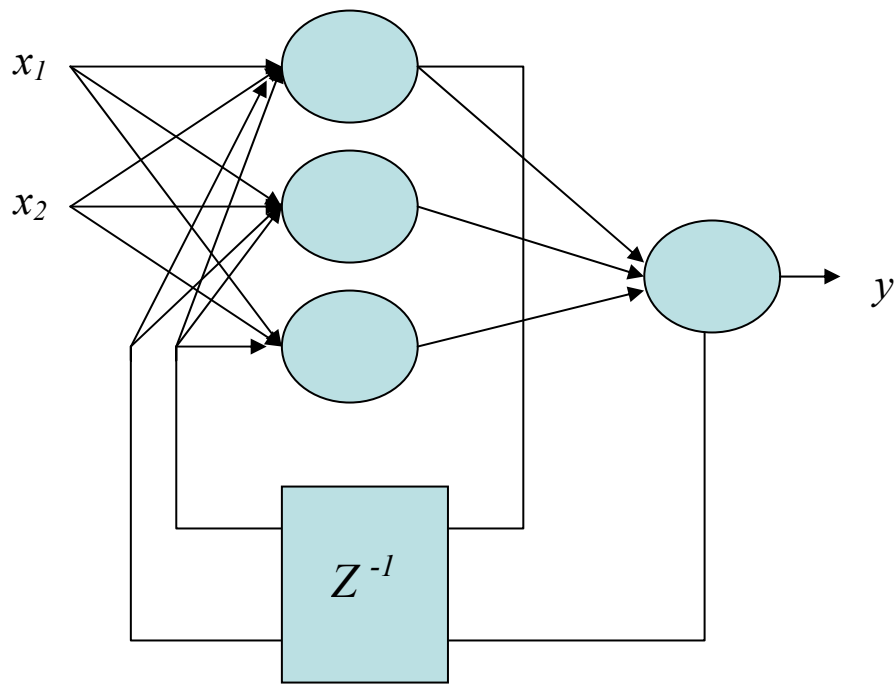
Figure 4. Recurrent ANN structure

In Figure 4 the symbol, Z^{-1}, denotes the unit delay operator. Specifics of several ANN structures will be discussed in detail as each type of structure is analyzed.

When an ANN is created, there are an infinite number of configurations in which the neurons could be arranged. At initialization all of the synaptic weights are set to random values. For the ANN to perform some useful task, the network will have to be trained to produce a meaningful output for a given set of input data. Training an ANN involves modifying the synaptic weights to minimize the error between the desired output of the network and the actual output of the network.

The two main learning methods are supervised learning and unsupervised learning (Haykin, 1999). Supervised learning involves comparing the output of the network to a desired output and adjusting the network weights accordingly. Unsupervised learning involves the network adapting itself to the data space and identifying similar data points. The construction and training of the types of neural network considered for this problem are presented in the following section.

3.2.4 Multi-layer perceptron

The most common class of ANN is the feed-forward multi-layer perceptron (MPL) (Haykin, 1999). The type of ANN uses multiple layers of multiple interconnected neurons, as shown in Figure 5.
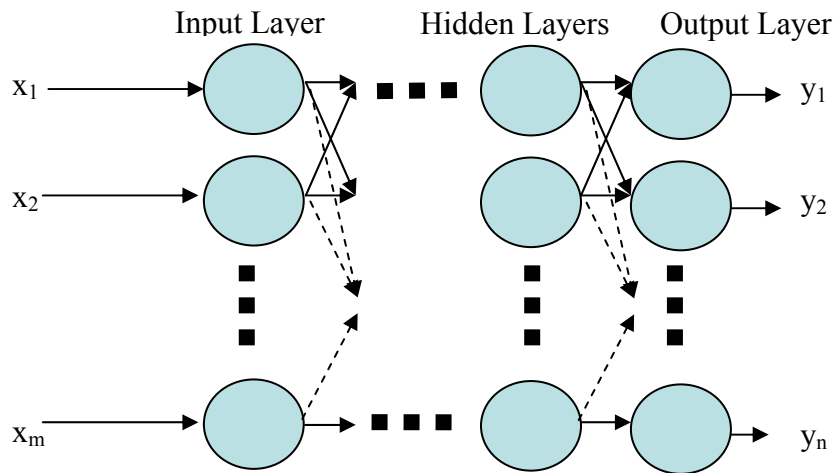
Figure 5. Multi-layer Perceptron

Often every neuron in a given layer is connected, by a synaptic weight, to every neuron in the next layer. This is known as a fully connected network. The activation functions of the neurons can be linear but, in most cases a sigmoidal non-linear function is used. Common choices for the activation function of the neurons are the logistic function and the hyperbolic tangent function. The layer of neurons where the data enter the network is known as the input layer. Analogously, the layer of neurons producing the output of the network is the output layer. Multiple layers of neurons can reside between these two layers. These are called hidden layers. As the number and size of hidden layers increases, computational power and ability of the network to perform complex tasks increases. However, computational expense of training and operating the network also increases. It is difficult to explicitly state the optimal size of a network. When constructing a MLP it is common practice to increase the size of the network until no significant performance gains are made or operation has become too computationally expensive. To reduce computational expense, "pruning", or removing some neurons or synaptic weights, is also available. These will be discussed in further sections.

3.2.5 Back-propagation algorithm

The MLP produces an output based on equations 3.1 through 3.3 computed at each node of the network (Haykin, 1999). Computation is done layer by layer beginning with the input layer, through the hidden layer(s), and ending at the output layer. As previously mentioned, when the network is initialized the weights are set to random values. Thus, when a piece of data is fed into the newly initialized network the output is most likely meaningless. The iterative supervised learning method called back propagation can be applied in an attempt to make the network useful at performing a task. Similar to how data propagates through the network, an error signal can be made to propagate back through the network, as seen in Figure 6.
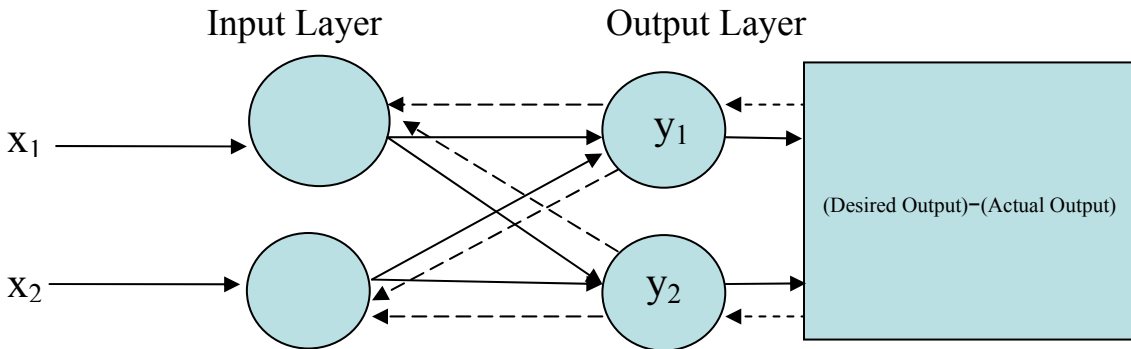
Figure 6. Error propagation back through MLP

The error at an output neuron can be computed as

$$e_j(n) = d_j(n) - y_j(n),$$
(3.8)

where e_j is the error computed, d_j is the desired output of the network, y_j is the actual output of the network, and n is iteration or training example number. The total error computed for a training example is defined as

21

$$\varepsilon(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n),$$ (3.9)

where C is the space of all output neurons. When presenting training data to the network the training data contains N training examples or patterns. Presentation of the entire training set of data to the network once is referred to as one epoch. The average error over one epoch is defined as

$$\varepsilon_{av} = \frac{1}{N} \sum_{n=1}^{N} \varepsilon(n).$$ (3.10)

This parameter reflects the overall accuracy to which the network has approximated the data. To derive the back propagation weight adjustment algorithm, the adjustment of the weights feeding into an output neuron j first must be considered. If the neuron model in Figure 7 is used, $v_j(n)$ is

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) y_i(n),$$ (3.11)

where there are m inputs to neuron j. The value of i begins at 0 to consider the DC bias term or synaptic weight w_{j0}. Output of neuron j at iteration n is

$$y_{j(n)} = \varphi_j(v(n)).$$ (3.12)
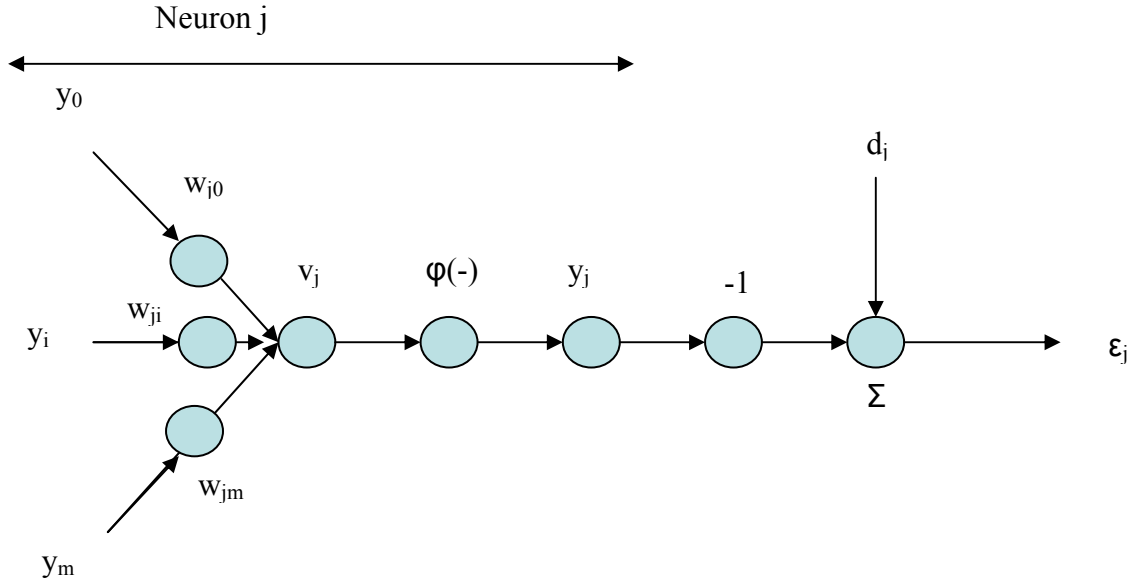
Figure 7. Neuron Model

To correct the output of the neuron, some change of synaptic weight w_{ji} must be applied. This correction is proportional to the change in error computed over some number of training examples with respect to the change in the value of the synaptic weight. This concept expanded via the chain rule yields

$$\frac{\partial \varepsilon(n)}{\partial w_{ji}(n)} = \frac{\partial \varepsilon(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}, \tag{3.13}$$

where ε is the error computed at iteration n. Solving for each of the terms in equation 13 yields

$$\frac{\partial \varepsilon(n)}{\partial e_j(n)} = e_j(n) \tag{3.14}$$

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \tag{3.15}$$

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \varphi'(v_j(n)) \tag{3.16}$$

23

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n).$$ (3.17)

At this point, it is important to note that by using sigmoidal activation functions in the neurons, problems with solving equation 3.16 are avoided, since sigmoidal activation functions are C^1 continuous. Thus, equation 3.13 becomes

$$\frac{\partial \varepsilon(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi'_j(v_j(n))y_i(n).$$ (3.18)

Since the change in a given synaptic weight is proportional to equation 3.18 the change in a given weight is

$$\Delta w_{ji}(n) = -\eta \frac{\partial \varepsilon(n)}{\partial w_{ji}(n)},$$ (3.19)

where η is known as the learning rate parameter. A larger learning rate parameter will cause the training of the network to occur faster, with the accompanying risk of network instabilities. A smaller learning rate parameter will cause the training of the network to take more time. This is the back-propagation weight adjustment rule for an output neuron.

Next, consider the case where the weights of neuron j, located in a hidden layer adjacent to the output layer, whose neurons are denoted by k, are to be adjusted. This case is illustrated in Figure 8.
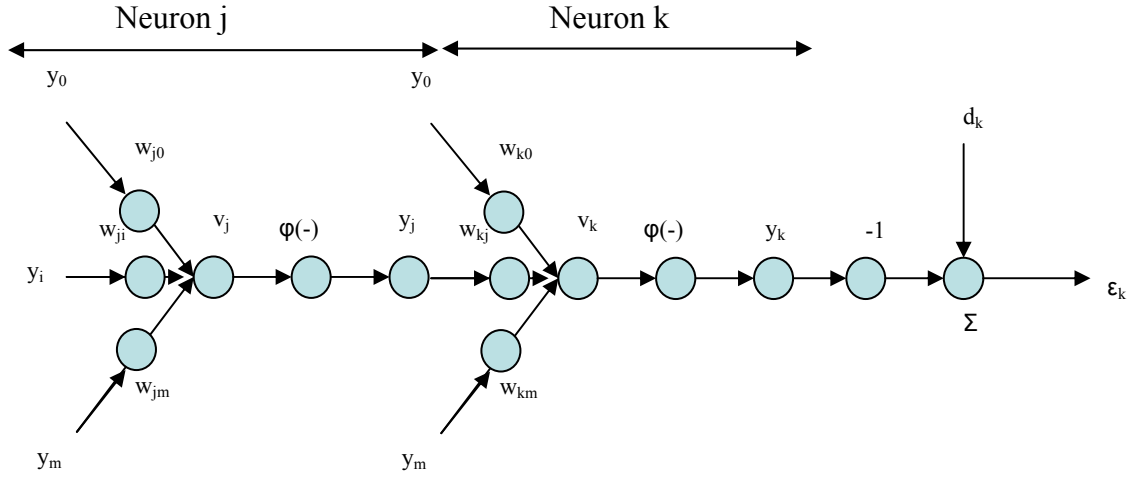
Figure 8. Model of Hidden Neuron

For convenience, let a function known as the local gradient be defined as

$$\delta_j(n) = -\frac{\partial \varepsilon(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)}.$$

(3.20)

It is important to note that

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n).$$

(3.21)

The error from output neurons $k \in C$ is

$$\varepsilon(n) = \frac{1}{2} \sum_{k \in C} e_k^2(n).$$

(3.22)

Differentiating this function with respect to the output of neuron j yields

$$\frac{\partial \varepsilon(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial y_j(n)}.$$

(3.23)

By applying the chain rule, the differential term in equation 23 can be split into

$$\frac{\partial \varepsilon(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial v_k(n)} \frac{\partial v_k(n)}{\partial y_j(n)}.$$

(3.24)

In much the same procedure as equations 15 and 16 it is found that

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi'(v_k(n)).$$

(3.25)

In solving for the second term, recall that

$$v_k(n) = \sum_{j=0}^{m} w_{kj}(n) y_j(n).$$

(3.26)

Thus,

$$\frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n).$$

(3.27)

If the local gradient function is defined at the kth neuron as

$$\delta_k(n) = e_k(n)\varphi_k'(v_k(n)),$$

(3.28)

the change in error at neuron j with respect to the change in y_j is now

$$\frac{\partial \varepsilon(n)}{\partial y_j(n)} = -\sum \delta_k(n) w_{kj}(n).$$

The local gradient at neuron j becomes

$$\delta_j(n) = \varphi_j'(v_j(n)) \sum_k \delta_k(n) w_{kj}(n).$$

(3.29)

This result can then be placed in equation 21 to calculate weight adjustment Δw_{kj}. This concept can be extended to compute the weight adjustment for any neuron in the network. However, it is easy to see how this process can become very computationally intensive for a network with many layers.

3.2.6 Levenberg-Marquardt algorithm

In addition to the basic back-propagation algorithm, there are several other training algorithms available for training the synaptic weights of a neural network. Since training an ANN involves minimizing the error of the output vector as compared to some target vector, many of these algorithms have their origin in function optimization methods. The Levenberg-Marquardt algorithm has been used to train the MLP in this problem. The Levenberg-Marquardt algorithm is defined as

$$\underline{x}_{k+1} = \underline{x}_k - \left(\underline{F}(\underline{x}_k) + \mu_k \underline{I} \right)^{-1} \underline{g}_k, \tag{3.30}$$

where $\underline{x}$ is a vector of parameters at a time step, $\underline{F}(\cdot)$ is the Hessian matrix, μ is an algorithm parameter greater than or equal to zero, and g is the gradient of the network error (Chong, 2001). The Levenberg-Marquardt algorithm attempts to capture the best optimization properties of both gradient descent and Newton's methods (Chong, 2001). When parameters are far from the global minimum, gradient decent algorithms often work best to quickly move the optimization in the correct direction. However, as the optimization moves close to the global minimum, gradient descent algorithms are often very slow to converge. Conversely, Newton's algorithms can be unreliable when the optimization is far from the global minimum, but they are usually very quick to converge as the optimization nears the global minimum. In equation 3.30, if μ is comparatively large, the algorithm behaves like a gradient descent method. As μ approaches zero, the algorithm reduces to Newton's method. When the Levenberg-Marquardt algorithm is initiated, the μ begins at a value greater than zero and is reduced in some fashion, but never becomes less than zero. Choice of this parameter and method of reduction often requires skill

and knowledge of the problem. In the case of the algorithm used to train the MLP, the Levenberg-Marquardt algorithm becomes

$$\underline{w}_{k+1} = \underline{w}_k - \left(\underline{J}_k^T \underline{J}_k + \mu_k \underline{I}\right)\underline{J}_k \underline{e}_k,$$ (3.31)

$\underline{w}$ is the weight vector of the network, $\underline{J}$ is the Jacobian of the network errors with respect to the weights, and $\underline{e}$ is the vector of network errors. An approximation of the Hessian matrix,

$$\underline{F}_k \approx \underline{J}_k^T \underline{J}_k,$$ (3.32)

is used to reduce total computation. Also, the gradient is defined as

$$\underline{g}_k = \underline{J}_k \underline{e}_k.$$ (3.33)

The Levenberg-Marquardt algorithm utilizes the back-propagation algorithm to compute error parameters of the network thus, it is an improvement on the basic back-propagation algorithm.

3.2.7 Network pruning

Network pruning is a technique that can be applied to reduce the complexity and thus the computational expense of operating a neural network (Haykin, 1999, Karnin, 1990). First define a cost function as

$$R(\underline{w}) = \varepsilon_s(\underline{w}) + \lambda \varepsilon_c(\underline{w}),$$ (3.34)

where ε_s is a measure of network performance, ε_c is a measure of network complexity, and λ is known as the regularization parameter. The regularization parameter controls the influence of network complexity in the optimization. In many cases, networks still perform to a high degree of accuracy even when the complexity of the network has been greatly reduced. In addition, the network may possess greater generalization capability when complexity is reduced, as a highly

complex network may actually be over fit to the training data and not capable of making generalizations.

An effective method of network pruning is Hessian based pruning. This approach begins by computing the average error about a given operating point and approximating the error around the point by a Taylor series expansion about the point. This cost function is defined as,

$$\varepsilon_{av}\left(\underline{w}+\underline{\Delta w}\right)=\varepsilon_{av}\left(\underline{w}\right)+\underline{g}^{T}\left(\underline{w}\right)\underline{\Delta w}+\frac{1}{2}\underline{\Delta w}^{T}\underline{H}\underline{\Delta w}+O\left(\|\underline{\Delta w}\|^{3}\right),$$ (3.35)

where $\underline{w}$ is the weight vector, $\underline{\Delta w}$ a perturbation in the weight vector, g is the gradient evaluated for the weight vector, and $\underline{H}$ is the Hessian of the weight vector. In this process, two assumptions to be made: 1) the network has converged to a global minimum of error 2) that the error function is approximately quadratic at this global minimum. Under these assumptions it is possible to state that,

$$\begin{aligned}\Delta\varepsilon_{av} &= \varepsilon\left(\underline{w}+\underline{\Delta w}\right)-\varepsilon\left(\underline{w}\right)\\ &\cong\frac{1}{2}\underline{\Delta w}^{T}\underline{H}\underline{\Delta w}\end{aligned}.$$ (3.36)

The objective of pruning the network is to set selected weights to zero. This is stated as

$$\underline{1}_{i}^{T}\underline{\Delta w}+w_{i}=0,$$ (3.37)

where $\underline{1}_{i}$ is a unit vector in which the ith element equals 1 and all others are 0. This problem can now be stated as the constrained optimization

$$S=\frac{1}{2}\underline{\Delta w}^{T}\underline{H}\underline{\Delta w}-\lambda\left(\underline{1}_{i}^{T}\underline{\Delta w}+w_{i}\right),$$ (3.38)

where λ is a Lagrange multiplier. By taking the derivative of S, the Lagrangian, and applying constraint equation 3.37 the optimal change in the weight vector becomes

$$\underline{\Delta w}=-\frac{w_{i}}{[\underline{H}^{-1}]_{i,i}}\underline{H}^{-1}\underline{1}_{i}.$$ (3.39)

Thus the optimum value of the Langrangian for weight w_i becomes

$$S_i = \frac{w_i^2}{2[\underline{H}^{-1}]_{i,i}} \; .$$
(3.40)

The value of the Lagrangian S_i is also referred to as saliency. Saliency is a measure of the increase in the mean square error of the network if the selected weight is deleted. Thus removing weights with low saliency will not greatly affect the performance of the network.

Network pruning is an iterative process. First the saliency of all of the weights in the network is computed. Next, the weight with the lowest saliency is removed from the network. The network is then tested to verify that it will still perform to the desired level of accuracy. The process is then repeated until the network has been pruned to the desired size or the network performance has degraded to an unsatisfactory level. It is good practice to retrain the network after pruning has been completed.

3.2.8 Self-organizing map

The concept of the self organizing map (SOM) was proposed by Rosenblatt in 1958. Many of the advancements in the technique of SOMs are attributed to Teuvo Kohonen. In fact, SOMs are sometimes referred to as Kohonen Network or Kohonen Self-Organizing Maps. The SOM is a powerful tool for visualization of high-dimensional data (Kohonen, 1998). This is accomplished by mapping high-dimensional data into a low dimensional grid-like structure. In most cases the output space is usually one or two dimensional, although mappings of higher dimensions are possible. A visual representation of a SOM is presented in Figure 9.
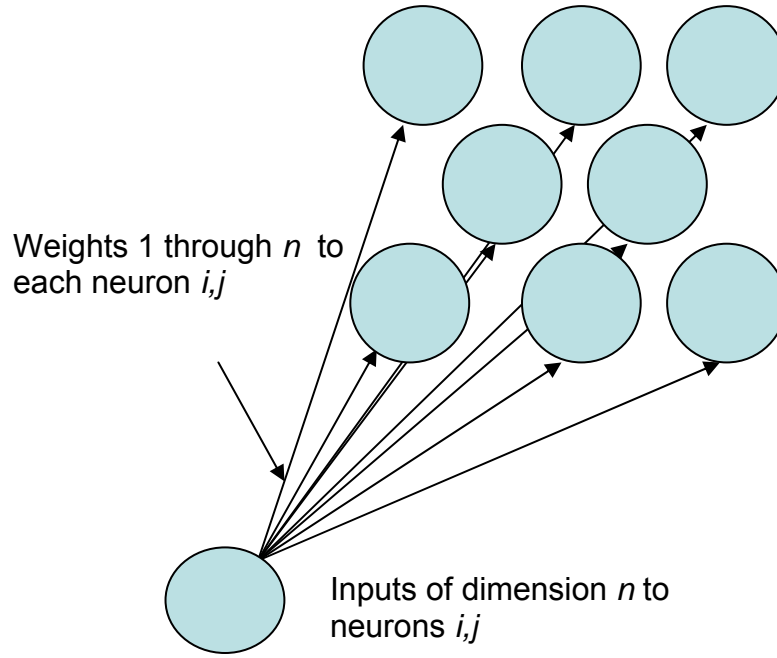
Figure 9. Diagram of SOM

SOMs consist of set of inputs, of dimension *n*, fed into a single output layer of neurons. Each of the output neurons has *n* synaptic weights, one for each of the inputs. Each of the output neurons has a set "location" in the output space. In the case of the one dimensional SOM, location is defined by a single coordinate. For the two dimensional SOM location is defined by two coordinates and so on. The utility of this coordinate system will soon become apparent. Classification by SOM is a competitive process. When input pattern $\underline{x}$ of dimension *n* is presented to the network, the function

$$\left\| \underline{x} - \underline{m}_i \right\|, \tag{3.41}$$

is computed at all neurons, where in equation 3.41, $\underline{m}$ denotes the synaptic weights of neuron *i*.

The "winning" neuron or decision of the network is

$$\min_i \left\{ \left\| \underline{x} - \underline{m}_i \right\| \right\}. \tag{3.42}$$

This can be stated as the neuron whose synaptic weights most closely match the values of the input vector.

As is the case with many ANN structures the SOM's synaptic weights are initialized to a random orientation. At this point it is necessary to train the network to produce any form of useful output. In the case of the SOM, similar data that are desired to be clustered together in the map. Thus, it is desirable to only modify the synaptic weights in a particular area of the map at a given time step. Let N_c be defined as a "neighborhood" function. Also let c be defined as the neuron winning the competition in equation 3.42. The neighborhood function refers the distance other neurons are to the winning neuron. Accordingly, the synaptic weights of the neurons in the neighborhood of the winning neuron are to be modified and the neurons not within the neighborhood of the winning neuron are not to be modified during a given time step of training. This is defined as

$$\underline{m}_i(t+1) = \begin{cases} \underline{m}_i(t) + \Delta \underline{m}_i, & \text{if } i \in N_c(t) \\ \underline{m}_i(t), & \text{if } i \notin N_c(t) \end{cases}. \tag{3.43}$$

The synaptic weights of the neurons closer within the neighborhood function and closer to the winning neuron should be modified by a greater magnitude than those further away from the winning neuron. To accomplish this we define a kernel function as

$$h_{ci} = h_o \exp\left(\frac{-\left\| r_i - r_c \right\|^2}{\sigma^2} \right), \tag{3.44}$$

where h_o is a constant, r_i is the location of neuron i, r_c is the location of the winning neuron, c, and σ is a function determining neighborhood width. Equation 3.44 has the effect of centering a

q-dimensional Gaussian distribution at the winning neuron, where q is equal to the dimension of the space of the output neurons, most often one or two.

In the process of network training, error of the mapping to the output space will start at a large value and, if the training proceeds as planned, become smaller. Accordingly, as the SOM is trained, it would be desirable to make large changes in synaptic weights over a large neighborhood at the beginning of the training and smaller changes in the synaptic weights over smaller neighborhoods as the SOM converges to its optimal state. Thus, a good choice of neighborhood function becomes

$$\sigma(t) = \sigma_o \exp\left(-\frac{t}{\tau_1}\right),$$ (3.45)

where σ_o is a neighborhood width constant, t is the integer time step of training, and τ_1 is the time constant of the exponential decay. This function will decrease the size of the neighborhood effected by each set of weight changes, but still define a neighborhood of at least one neuron, (the winning neuron will always be included). In a very similar fashion, h_o can be set to

$$h_o = \eta \exp\left(-\frac{t}{\tau_2}\right),$$ (3.46)

where η is a constant learning rate parameter and τ_2 is the time constant of the exponential decay. Equation 3.46 has the effect of reducing the magnitude of weight changes as training of the SOM proceeds. The final SOM weight update equation is now defined as

$$m_i(t+1) = m_i(t) + \eta \exp\left(-\frac{t}{\tau_2}\right) \exp\left(\frac{-\|r_i - r_c\|^2}{\left(\sigma_o \exp\left(-\frac{t}{\tau_1}\right)\right)^2}\right).$$ (3.47)

In actual practice, the training of the SOM occurs in two distinct phases, the first of which is the ordering phase and the later the convergence phase. In the ordering phase of training, large adjustments are made to the synaptic weights to achieve a topological order from an initially chaotic state. This phase may take up to 1000 iterations of training data. A reasonable choice of η would be 0.1 and 1000 for τ_2. The initial size of the neighborhood function should be approximately equal to the radius of the output lattice (i.e. if the output neurons are arranged in a 4 by 4 square with sides of length 4, then the radius of the lattice is the square root of 2) with its time constant set to

$$\tau_1 = \frac{1000}{\log_{10}\sigma_o}.$$
(3.48)

In the convergence phase of training, smaller adjustments to the synaptic weights are to be made fine-tune the map to the data space. It is recommended that this phase be conducted for 500 times the total number of neurons in the network iterations. The value of h_o can be maintained around 0.01 during this phase. The neighborhood function should enter the convergence phase including the nearest neighbors of the winning neuron and finish including only the winning neuron. By this method, effective training of a SOM is possible.

3.2.9 Support vector machine

The development of the support vector machine is credited to the work of Vapnik (Abe, 2001, Cortes and Vapnik, 1995). Consider a system with n classes or types of data. The support vector machine works by converting an n class classification problem into n 2-class classification problems. An optimal hyperplane is then computed, in each 2-class problem, separating the data

of a certain class from rest of the data. The derivation below follows. For a single 2-class problem where the data is linearly separable and for class 1 y_i=1 and for class 2 $y_i = -1$, a decision function can be defined as

$$D(x) = \underline{w}^T \underline{x} + b .$$ (3.49)

In equation 3.49 $\underline{w}$ is a weight vector of dimension m, b is a scalar bias term, and for values of i from 1 to M

$$\underline{w}^T \underline{x}_i + b \begin{cases} \geq +1 & \text{for} \quad y_i = +1 \\ \leq -1 & \text{for} \quad y_i = -1 \end{cases} .$$ (3.50)

A hyperplane separating these two classes of data is

$$D(\underline{x}) = \underline{w}^T \underline{x} + b = c \quad \text{where} \quad -1 < c < 1.$$ (3.51)

Figure 10 illustrates the construction of hyperplanes to separate data into two classes and also shows the optimal hyperplane of separation.
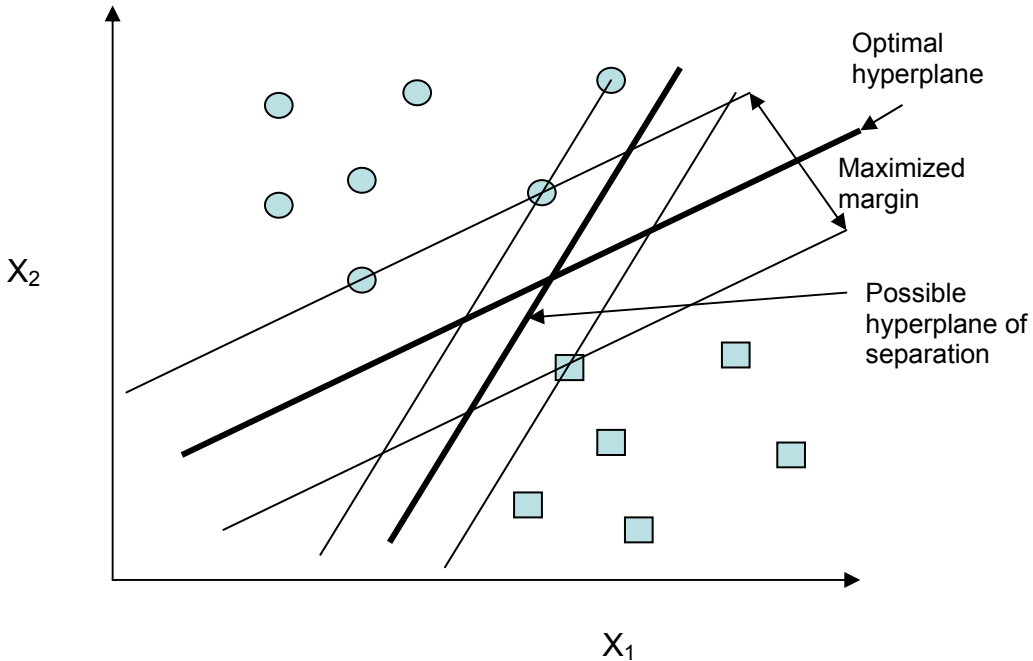

Figure 10. Examples of separating and optimal hyperplanes in two-dimensional space

The optimal separating hyperplane is defined as $c=0$ in equation 3.51 and the margin, or distance perpendicular from the optimal hyper plane to the nearest data point, has been maximized. In Vapnik's theory, risk of classification error with probability of at least $1-\eta$ is bounded by

$$R(\underline{w},b) \leq R_{emp}(\underline{w},b) + \phi, \tag{3.52}$$

where R_{emp} is the empirical risk and ϕ is the confidence interval of unknown data. Empirical risk is the classification error on training data, of data set size M, and confidence interval is

$$\phi = \frac{\varepsilon}{2}\left(1 + \sqrt{1 + \frac{4R_{emp}(\underline{w},b)}{\varepsilon}}\right), \tag{3.53}$$

where,

$$\varepsilon = 4 \frac{h\left[\ln\left(\frac{2M}{h}\right)+1\right] - \ln\left(\frac{\eta}{4}\right)}{M}. \tag{3.54}$$

The parameter h is defined as the Vapnik-Cortes (VC) dimension of a set of hyperplanes. The VC dimension of the set of classification functions $\{D(\underline{x},\underline{w}) : \underline{w} \in W\}$, where W is the space of the weight vectors, is the maximum number of training examples that can be learned by the machine without error for all possible binary labelings of the classification functions (Haykin, 1999). Since the set of m-dimensional hyperplanes can separate at most $m+1$ samples, the VC dimension is $m+1$. It is now possible to solve for the optimal hyperplane.

Since the Euclidean distance of a data point, x, to the decision boundary is $|D(\underline{x})|/\|\underline{w}\|$ and δ can be used to define the margin, the network training data satisfies the condition

$$\frac{y_i D(x_i)}{\|\underline{w}\|} \geq \delta \qquad \text{for} \qquad i=1 \text{ through } M. \tag{3.55}$$

To prevent multiple solutions to this equation let the constraint

$$\delta\|\underline{w}\| = 1 \tag{3.56}$$

be imposed. The optimal hyperplane can now be solved for by minimizing

$$\frac{1}{2}\|\underline{w}\|^2$$
(3.57)

with respect to $\underline{w}$ and b subject to

$$y_i\left(\underline{w}^T\underline{x}_i + b\right) \geq 1 \qquad \text{for} \qquad i=1 \text{ through } M.$$
(3.58)

This however, is a convex optimization problem which can prove quite computationally intensive for the large dimensional problems. The convex problem can be converted into the unconstrained problem (Abe, 2001, Chong, 2001)

$$Q(\underline{w}, b, \alpha) = \frac{1}{2}\underline{w}^T\underline{w} - \sum_{i=1}^{M} \alpha_i \left\{y_i\left(\underline{w}^T\underline{x}_i + b\right) - 1\right\},$$
(3.59)

where α_i is a Langrange multiplier. The optimal solution is the point where equation 3.59 is minimized with respect to $\underline{w}$ and b and maximized with respect to α (for $\alpha \geq 0$). The optimal solution would satisfy the Kuhn-Tucker conditions

$$\frac{\partial Q\left(\underline{w}^*, b^*, \alpha^*\right)}{\partial b} = 0$$
(3.60)

and

$$\frac{\partial Q\left(\underline{w}^*, b^*, \alpha^*\right)}{\partial \underline{w}} = 0,$$
(3.61)

where w^*, b^*, α^* are the optimized parameters. Equations 3.59 through 3.61 can be reduced to

$$\sum_{i=1}^{M} \alpha_i^* y_i = 0$$
(3.62)

and

$$\sum_{i=1}^{M} \alpha_i^* y_i \underline{x}_i = \underline{w}^*.$$
(3.63)

Substituting these quantities back into equation 3.59 yields

$$Q(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=0}^{M} \alpha_i \alpha_j y_i y_j \underline{x}_i^T \underline{x}_j .$$

(3.64)

The constraint equations are

$$\sum_{i=1}^{M} y_i \alpha_i = 0, \quad \alpha_i \geq 0 .$$

(3.65)

The optimal value of b^* is given by

$$b^* = -\frac{1}{2}\left(\underline{w}^T \underline{s}_1 + \underline{w}^T \underline{s}_2 \right),$$

(3.66)

where s_1 and s_2 are arbitrary support vectors from data classes 1 and 2 respectively. Thus b^* can be written as

$$b^* = -\frac{1}{2}\sum y_i \alpha_i^* \left(\underline{s}_1^T \underline{x}_i + \underline{s}_2^T \underline{x}_i \right).$$

(3.67)

This defines the optimal hyperplane when data are linearly separable. In the case where data are not linearly separable, the concept of a slack variable is introduced. Equation 3.58 becomes

$$y_i\left(\underline{w}^T \underline{x}_i + b \right) \geq 1 - \xi_i \quad \text{for} \quad i=1 \text{ through } M,$$

(3.68)

where ξ_i is a non-negative slack variable. When $0 < \xi < 1$, the data do not have the maximum margin, but are still correctly classified. If $\xi_i \geq 1$, the data are misclassified by the optimal hyperplane. The function to minimize is now

$$\frac{1}{2}\|\underline{w}\|^2 + C\sum_{i=1}^{M} \xi_i ,$$

subject to,

$$y_i\left(\underline{w}^T \underline{x}_i + b \right) \geq 1 - \xi_i .$$

(3.69)

The constant C is somewhat like a regularization parameter, in that it controls complexity of the classifier versus accuracy of fit to training data. Similar to equation 3.59, the problem can be posed as an unconstrained optimization as

$$Q(\underline{w}, b, \xi, \alpha, \beta) = \frac{1}{2} \| \underline{w} \|^2 + C \sum_{i=1}^{M} \xi_i - \sum \alpha_i \left(y_i \left(\underline{w}^T \underline{x} + b \right) - 1 + \xi_i \right) - \sum_{i=1}^{M} \beta_i \xi_i , \quad (3.70)$$

where β_i is also a Langrange multiplier. In the optimal case

$$\frac{\partial Q \left(\underline{w}^*, b^*, \xi^*, \alpha^*, \beta^* \right)}{\partial b} = 0 , \qquad (3.71)$$

$$\frac{\partial Q \left(\underline{w}^*, b^*, \xi^*, \alpha^*, \beta^* \right)}{\partial \underline{w}} = 0 , \qquad (3.72)$$

and

$$\frac{\partial Q \left(\underline{w}^*, b^*, \xi^*, \alpha^*, \beta^* \right)}{\partial \xi} = 0 \qquad (3.73)$$

will hold true. By utilizing equations 3.70 through 3.73, it is now possible to state that

$$\sum_{i=1}^{M} \alpha_i^* y_i = 0 , \quad \alpha_i^* \geq 0 , \qquad (3.74)$$

$$\sum_{i=1}^{M} \alpha_i^* y_i \underline{x}_i = \underline{w}^* , \qquad \alpha_i^* \geq 0 , \qquad (3.75)$$

and

$$\alpha_i + \beta_i = C , \quad \alpha_i^*, \beta_i^* \geq 0 . \qquad (3.76)$$

Equation 3.70 can now be restated as

$$Q(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j \underline{x}_i^T \underline{x}_j \qquad (3.77)$$

subject to the constraints

$$\sum_{i=1}^{M} y_i \alpha_i = 0 \quad \text{and} \quad 0 \le \alpha_i \le C .$$
(3.78)

The decision function is expressed as

$$D(\underline{x}) = \sum_{i=1}^{M} \alpha_i^* y_i \underline{x}_i^T \underline{x} + b^* .$$
(3.79)

When D($\underline{x}$) >0, the input data point is classified as class 1, otherwise, it is classified as coming from class 2. Determining the optimal hyperplane in this method produces a classifier with good generalization abilities if the data are linearly separable. However, if the data are not linearly separable, the generalization ability of the network can be improved by mapping the data into a high-dimensional dot product space called a feature space. If the non-linear function $g(\underline{x})$ =$(g_1(\underline{x}),...,g_q(\underline{x}))^T$ is defined to map the p-dimensional input vector, $\underline{x}$, to the q-dimensional feature space, then the linear decision function can be expressed as

$$D(\underline{x}) = \underline{w}^T g(\underline{x}),$$
(3.80)

where $\underline{w}$ is a vector of dimension q. In equation 3.80 the constant term has been omitted, since $g_1(\underline{x})$=1. The decision function can be rewritten as

$$D(\underline{x}) = \sum \alpha_i y_i g(\underline{x}_i)^T g(\underline{x}).$$
(3.81)

The Hilbert-Schmidt theory (Abe, 2001) states that the dot product in feature space can be expressed by a symmetric kernel function

$$H(\underline{x},\underline{x}') = \sum_{j=1}^{q} g_j(\underline{x}) g_j(\underline{x}'),$$
(3.82)

if

$$\int\int H(\underline{x},\underline{x}') h(\underline{x}) h(\underline{x}') d\underline{x} d\underline{x}' \ge 0$$
(3.83)

is satisfied for all the square integrable functions $h(\underline{x})$ in the compact subset of the input space $\left(\int h^2(\underline{x})d\underline{x} < \infty\right)$. Thus the optimization to be solved is now to maximize

$$Q(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j H\left(\underline{x}_i, \underline{x}_j\right) \qquad (3.84)$$

subject to the constraints

$$\sum_{i=1}^{M} y_i \alpha_i = 0 \quad \text{and} \quad 0 \le \alpha_i \le C. \qquad (3.85)$$

The optimal hyperplane is now

$$\sum_{i=1}^{M} \alpha_i y_i H\left(\underline{x}, \underline{x}'\right) = 0. \qquad (3.86)$$

The optimal weight vector, w_o is defined as

$$\underline{w}_o = \sum_{i=1}^{M} \alpha_{o,i} y_i g\left(\underline{x}_i\right), \qquad (3.87)$$

where α_o denotes the optimum values of α. In the case of the Radial Basis Function network, the kernel function is

$$H\left(\underline{x}, \underline{x}'\right) = \exp\left(-\frac{1}{2\sigma^2}\|\underline{x} - \underline{x}'\|^2\right), \qquad (3.88)$$

where σ is the kernel width parameter specified by the user.

4.0 TRADITIONAL ACOUSTIC METRICS

In acoustic monitoring, several metrics are commonly used to quantify a noise source. The metrics, power spectral density, peak sound pressure level, sound exposure level, equivalent continuous sound pressure level, crest factor, kurtosis, number of positive samples, and number of negative samples were computed for the signals to be analyzed in this noise classification problem. Power spectral density (PSD) is a common metric used in the analysis of signals. This derivation of the PSD was taken from Norton (Norton, 2003). First, define an arbitrary signal in time, $x(t)$. The auto-correlation, or correlation of the signal to itself, is

$$R_{xx}(\tau) = E[x(t)x(t+\tau)] = \lim_{T\to\infty} \frac{1}{T} \int_0^T x(t)x(t+\tau)dt , \qquad (4.1)$$

where E[-] is the expected value operator, τ is the separation in time, and T is the length of the signal. The auto-spectral density of the signal is obtained by taking the Fourier transform of the correlation, shown as

$$S_{xx}(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} R_{xx}(\tau)\exp(-i\omega\tau)d\tau , \qquad (4.2)$$

where ω indicates the frequency domain. The auto-spectral density is defined for frequencies ranging from $-\infty$ to ∞ and is symmetric about the y-axis. In real applications, it is often not convenient to deal with negative frequencies so, the PSD is defined as

$$G_{xx}(\omega) = 2S_{xx}(\omega)|_0^\infty . \qquad (4.3)$$

Another acoustic metric computed was the peak sound pressure level (L_{pk}) (ISO, 1996). L_{pk} is defined as

$$L_{pk} = \max\left(10\log\frac{p(t)^2}{p_0^2}\right), \tag{4.4}$$

where $p(t)$ is sound pressure as a function of time, p_0 is the standard reference pressure of 20×10^{-6} Pa. Lpk, along with many other acoustic metrics, is measured with the decibel (dB) scale. Sound exposure level (SEL) was also computed across the data set (ISO, 1996). SEL has symbol L_E and is computed as

$$L_E = 10\log\left(\frac{E}{E_0}\right), \tag{4.5}$$

where

$$E = \int_T p^2(t)dt \tag{4.6}$$

and

$$E_0 = \int_T p_0{}^2 dt. \tag{4.7}$$

SEL can be computed over any length of time but, in both equations 4.6 and 4.7 the standard value of $T=1$ was used. SEL is also expressed in decibels and the unit of SEL is Pa^2s. Another conventional acoustic metric calculated was equivalent continuous sound pressure level (L_{eq}). L_{eq} is defined as

$$L_{eq} = 10\log\left(\frac{1}{T}\int_T\frac{p^2(t)}{p_0{}^2}dt\right), \tag{4.8}$$

where T is the time that the L_{eq} is computed over (ISO, 1996). A common form of L_{eq} is L_{8eq}, where the L_{eq} is computed over eight hours, an average shift length at most jobs. Another metric investigated was crest factor. Crest factor, CF, is

$$CF = \frac{L_{pk}}{p_{rms}}, \tag{4.9}$$

where p_{rms} is the root mean squared of the sound pressure over some period of time. Crest factor tends to be a good indicator of impulsiveness in a signal (Norton, 2003). The statistical metric kurtosis is also a good indicator of impulsiveness within a signal. Kurtosis is the fourth statistical moment and is given by

$$\text{Kurtosis} = \frac{E[x^4]}{\sigma^4} = \frac{1}{\sigma^4 T}\int_0^T x^4 dt, \tag{4.10}$$

where x is the signal to be measured, σ is the variance of the signal, and T is the time frame over which kurtosis is computed (Norton, 2003). In the case of digital signals, the number of samples with positive and negative values can easily be counted. The number of positive and negative valued samples can also be taken as a metric. All of these metrics were computed via the Enhanced Sound Level Meter Matlab code developed at the University of Pittsburgh. (Smith, 2007)

5.0 DATA COLECTION

To develop this noise classifier, it was first necessary to collect a library of measurements containing military impulse noise and potential false positives noise sources for the classifier, as no such previous library was available. No commercially available data collection system could completely provide the functions desired to perform the data collection, so a composite system was assembled from several different components. Since the bulk of the energy for most of the measured noise sources is within the 0 to 100 Hz bandwidth, a B&K 4193 infrasonic microphone with bandwidth of 70 mHz to 20 kHz was used. The microphone was connected to a Larson Davis 824 Sound Level Meter (SLM). The AC output of the LD 824 SLM was in turn connected to a National Instruments (NI) DAQCard-6036E data acquisition card through a NI BNC-2110 input/output board. The DAQCard was installed into a Dell Latitude laptop with a Pentium IV processor. Figure 11 shows the measurements instrumentation set up in the field.

Figure 11. Instrumentation set up in field

A Virtual Instrument (VI) was created to capture waveform data. The VI enables an "automatic triggering/pre-triggering mode," where data are recorded when the signal exceeds a specified threshold, (used to automatically record impulse events – this corresponds to exceeding a certain L_{pk} value). Although the threshold can be set to record at or above any desired L_{pk} level, it was typically adjusted to just above ambient noise levels in order to record as much useful data as possible. In our data collection, a 0.1-0.25 second pre-trigger was coupled with an additional 2 seconds of recorded data for each record. By pre-triggering, the entire event was able to be stored. A manual (continuous) triggering mode was also possible, which was used to record longer or continuous events such as wind, aircraft noise, traffic, and engine noise. During multiple successive trigger events, the "automatic mode" also triggered nearly continuously. Although most of the military impulse noise energy would be in the very low-frequency range (0-100 Hz), some noise sources (aircraft) have significant energy up to 2 to 3 kHz. Consequently, data were sampled at 10 kHz. Figure 12 shows a screen shot of the VI.
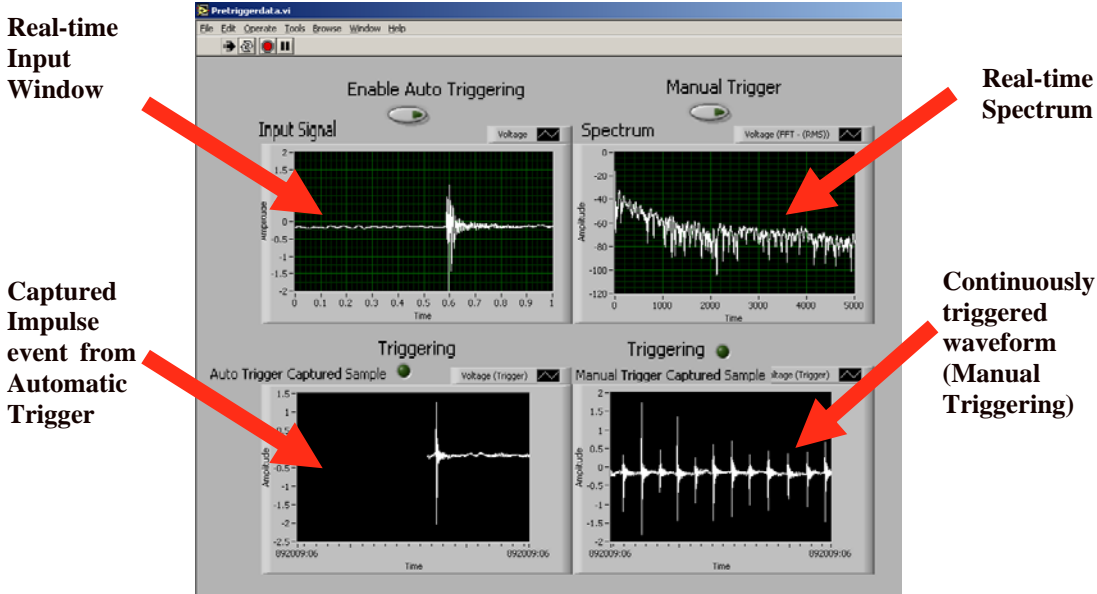
Figure 12. Screen-shot of virtual instrument

Approximately 1,000 usable waveforms were collected (330 military impulse and 670 non-impulse noise). The military impulse noise recordings consisted of 155mm Howitzers, 81mm mortars, 60mm rockets, M67 hand grenades, and Bangalore Torpedoes (strings of 3, (27lbs HE)). The military impulse noise records had Lpk values ranging from 80 to 138dB. The non-impulse noise records were wind noise and aircraft noise (F-16, A-10, C-130). Table I. shows a summary of the records of military impulse noise along with the number of impulse events that occurred during the 2.2 to 2.5 second recordings. As seen from the table, some recordings contained 2, 3, or even 4 events. Accurate classification in the midst of multiple events is a desired trait of a proposed algorithm.

Table I. Summary of the recorded ordinance and number of events occurring in each recording

	Number of Impulse Events During Recording			
	1	**2**	**3**	**4**
Types of Noise Sources				
155mm Howitzer Firing	116	20	5	1
155mm Howitzer Firing and High Explosive Event From 155mm Different Projectile	0	21	0	0
155mm High Explosive Projectile	36	0	0	0
60 mm Rockets	0	0	1	1
81 mm Mortar Impact	95	14	3	0
M 67 Hand Grenade	5	0	0	0
Bangalore Torpeo	4	0	0	0
155mm Howitzer Firing with F-16 Flyover	5	0	0	0
81mm Mortar Impact with F-16 Flyover	3	0	0	0

The military impulse noise recordings were made at ranges between 1.5 km and 6 km from the noise sources. Locations for the recording sessions were selected to provide the widest spectrum of different terrain types. Since operations were often carried out beyond the line of sight of the research team, a schedule of the weapons to be fired, locations of the active firing positions, and active impact areas was provided to the research team by the fire control officers prior to the research team entering the field. Sessions were conducted in flat open fields, mountainous dense forest, over soft sandy soil, on top of rocky terrain, tops of ridge-lines, valley bottoms, and

beside large bodies of water. During the recording sessions the temperature ranged from 75 to 100 F due to the summer season in which the measurements were made. The relative humidity also varied from 44% to 100% (and raining). Data collection performed under these various environmental conditions is necessary to develop a robust noise classifier.

6.0 DATA PROCESSING

Preliminary processing of the collected waveforms was conducted at the University of Pittsburgh using the Enhanced Sound Level Meter (ESLM) MATLAB code (Smith, 2007). The ESLM code computed the scalar metrics equivalent sound level (L_{eq}), peak sound level (L_{pk}), sound exposure level (SEL), crest factor (CF), 8-hour equivalent sound level (L_{8eq}), kurtosis (Kurt), duration (Dur), number of positive samples (Pos), and number of negative samples (Neg) and also the power spectral density (PSD) (Norton, 2003, ISO, 1996).

6.1 TIME-DOMAIN METRICS

Of the before mentioned scalar metrics, kurtosis and crest factor offered the best insight into whether or not a waveform contained military impulse noise. Kurtosis and Crest factor are often good indicators of impulsiveness within a signal (Norton, 2003, Lin and Zou, 2003, Dron et. al, 2004), as they tend to produce comparatively larger values for waveforms that are impulsive in nature. These metrics have been used successfully in the past in bearing fault analysis (Norton, 2003, Lin and Zou, 2003, Dron et. al, 2004), which is a somewhat similar problem to that of military impulse noise identification. In both cases it is necessary to detect an impulse that occurs within a measured waveform. For kurtosis to have a true statistical meaning, the measured process must be ergodic. We assume that the process of sound measurement is ergodic

in each recorded waveform. In the case of this study, kurtosis is not being used in a strict statistical sense, rather it is believed that the procedure of computing kurtosis will yield a useful metric in describing the content of a waveform. For these reasons it was believed that kurtosis and crest factor would be good metrics to aid in the detection of military impulse noise. Although the other metrics were not believed to be useful in identifying military impulse noise their respective statistical distributions are investigated to validate this assumption.

6.2 FREQUENCY-DOMAIN METRICS

Two custom scalar metrics were also developed to improve the accuracy of the noise classifier. These metrics are based on commonalities observed in the PSD of the waveforms of similar noise sources. As will be shown, spectrally-based metrics are less affected by multiple events that occur in a very short period of time and are less susceptible to propagation effects that can strongly influence temporal characteristics of a waveform. Although the primary interest is in discerning between military impulse and non-impulse noise, overall accuracy is enhanced by taking into account all classes of noise. The major classes of noise (military impulse noise and non-impulse noise), with L_{pk} values exceeding 80dB, are military impulse noise, wind noise and aircraft noise (the latter two of which are non-impulse noise) (Bucci and Vipperman, 2006, 2007). Figure 13 shows the PSD of a typical sample of wind noise.
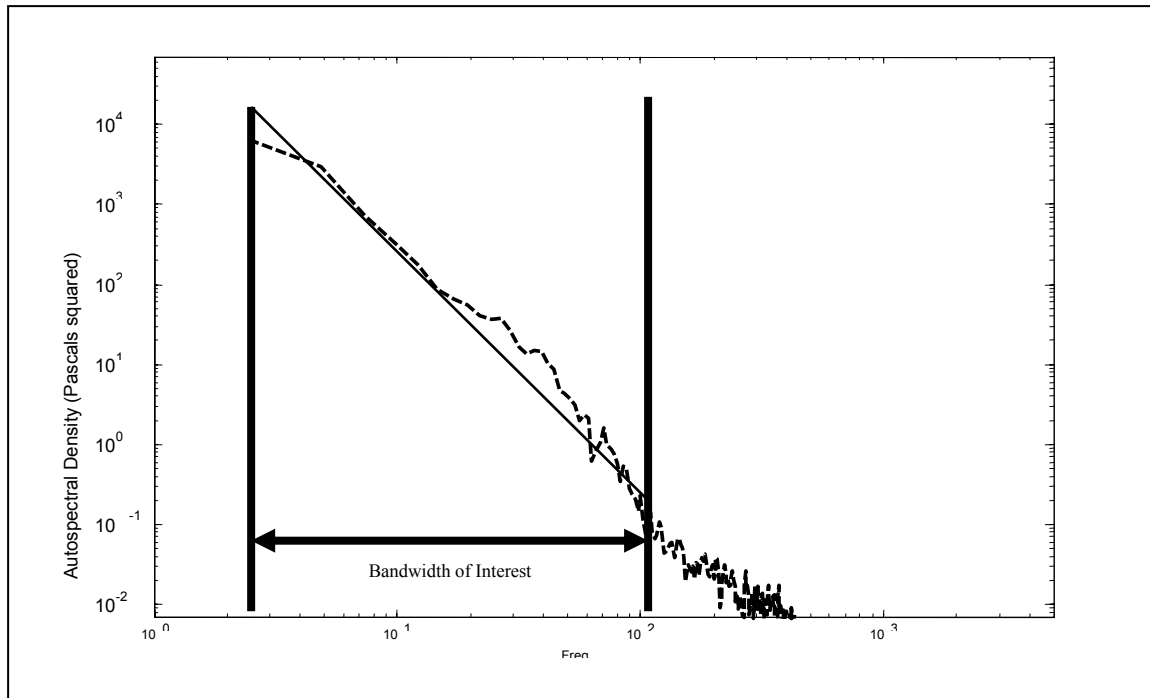
Figure 13. PSD curve of typical wind noise

It is important to note the typical linear shape and negative slope of the PSD curve as plotted on a log-log scale (Blevins, 2001). In the case of wind noise, most of the signal energy is below 10 Hz (Schomer, 1990). Figure 14 shows the PSD plots for some typical military impulse noise sources (81mm mortar (top), Bangalore torpedo (middle), and 155m Howitzer (bottom)).
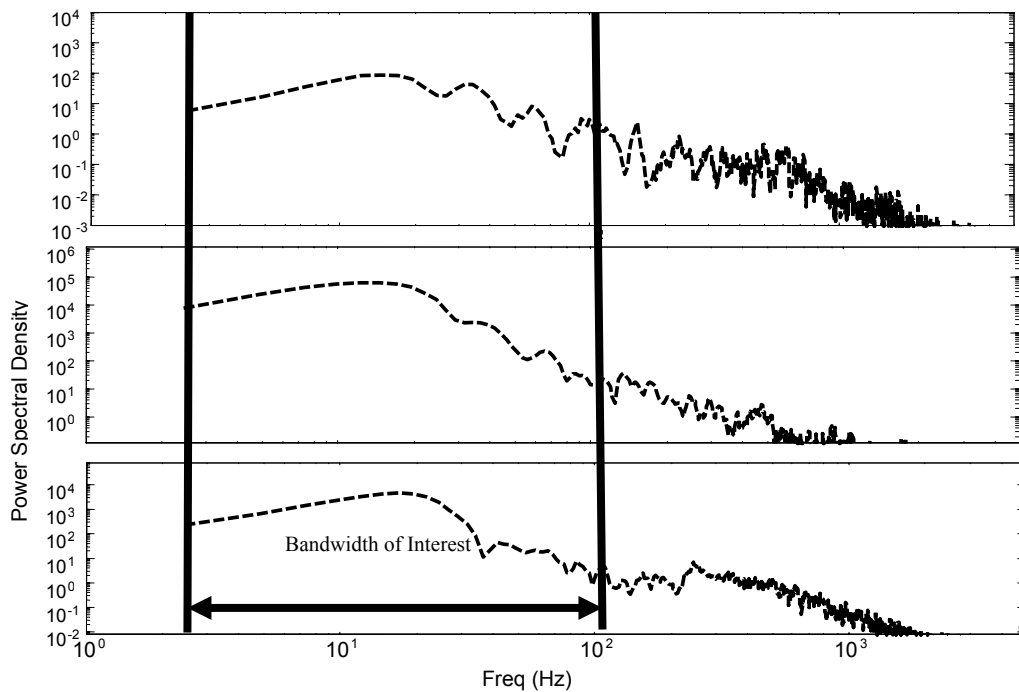
Figure 14. PSD curves of typical military impulse noise, (top) 81mm mortar, (middle) Bangalore torpedo, (bottom) 155mm Howitzer (firing)

Although these three PSD plots may appear somewhat different, all three PSD functions seem to begin with an increasing slope, peak with the greatest energy between 10 and 30 Hz, and the PSD tends to roll off at higher frequencies (Attias et. al, 2004, Benson, 1996). These trends are observed in each of the three plots in Figure 14. These three regions of the PSD are in contrast to the linear trend noted for wind noise.

Figure 15 illustrates an example of typical aircraft noise. Note that aircraft noise seems to have a relatively flat PSD out to around 400 Hz. By defining a "bandwidth of interest" (BWI) from 0 to 100 Hz, it is determined that sufficient generalization about each noise source can be made with the data points residing in this region.
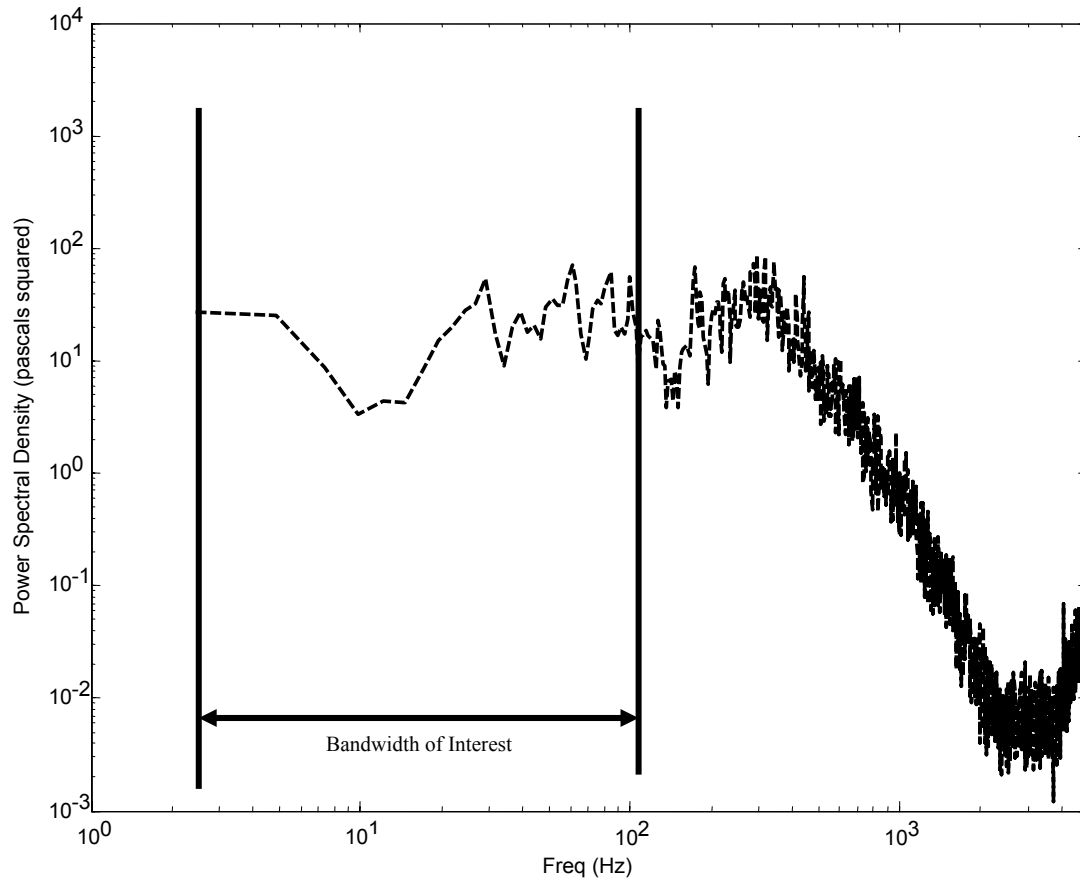
Figure 15. PSD typical aircraft noise (F-16)

It is desirable to capture the above generalizations about the PSD for various noise sources in an efficient manner. Using all frequency bins within the BWI would add considerable complexity to the classifier. Thus, a pair of scalar metrics were designed to work in tandem. These metrics are referred to as spectral slope (*m*) and weighted square error (*WSE*).

6.2.1   Spectral slope

Spectral Slope (m), is computed by creating a least-squares fit to a line,

$$\hat{y} = mx + b, \tag{6.1}$$

where $\hat{y} = \log_{10}(\text{PSD})$ is the base-10 logarithm of the power spectral density and $x = \log_{10}(f)$ is the base-10 logarithm of frequency. The fit is conducted over the frequencies between 2.5 and 100 Hz. Although the impulse and aircraft noise sources are poorly characterized by a linear trend, the slope is still useful for differentiating from other types of noise, as will be illustrated later.

6.2.2 Weighted square error

There are 42 frequency bins from 2.5-100Hz for the given spectral resolution (4,096 point FFT), and the Weighted Square Error *(WSE)* is computed from the first 41 of these bins. To complement metric m, the "goodness" of the linear fit is assessed with the *WSE*, which is computed as

$$WSE = \sum_{i=1}^{41} [y_i - \hat{y}_i]^2 [f_{i+1} - f_i], \tag{6.2}$$

where y_i is based upon the $\log_{10}(\text{PSD}_i)$ of the ith frequency bin, $\hat{y}_i$ is the estimate of y_i from the linear curve fit, and f_i is the log base 10 of the ith frequency. Squaring the quantity $[y_i - \hat{y}_i]$ allows *WSE* to remain positive and also reflects the total magnitude of the error. The term $[f_{i+1} - f_i]$ serves to add greater weight to the error at the lower frequency bins. This is done because the best features for identifying military impulse noise from non-impulse noise occur at

the lower reaches of the bandwidth of interest. Also, higher frequency bins within the bandwidth

of interest are more susceptible to contamination from other sources of environmental noise.   In

order to scale the metrics with respect to signal energy, the logarithmic PSD terms, y_i, are

normalized, between [0,1], which correspond to the minimum and maximum values of

$\log_{10}$(PSD), respectively. Thus, y_i is computed as

$$y_i = \frac{\log_{10}(PSD_i) - \min[\log_{10}(PSD)]}{\max[\log_{10}(PSD)] - \min[\log_{10}(PSD)]}. \qquad (6.3)$$

It is important to note to the values of $\hat{y}_i$ are not normalized. They are simply the curve fit of y_i.

In summary, PSD functions that do not follow a linear trend within the bandwidth of interest will

produce much larger values of *WSE* than PSD functions that exhibit a linear trend within the

bandwidth of interest, namely wind.

6.3 SIGNAL METRICS RESULTS AND DISCUSSION

Figures 16 through 23 show the statistical box plots for L_{pk}, L_{eq}, SEL, L_{8eq}, Pos, Neg, kurtosis,

and crest factor, respectively, for the three classes of noise. The upper and lower edges of the

box show the 25th and 75th percentile of the data respectively. This is known as the interquartile

range (IQR). The line in the center of the box shows the median of the data. The "whiskers"

that extend from the top and bottom of the boxes show the range of the distribution of data.

They are drawn to either the minimum or maximum value of the data or to a distance of 1.5

times the IQR. Data farther than 1.5 times the IQR from the box are regarded as outliers.

Figures 16 through 21 show the statistical box plots for L_{pk}, L_{eq}, SEL, L_{8eq}, Pos, and Neg

respectively. As seen in these plots, there are not any clear trends that would provide a method

for differentiating the noise sources by using these metrics. This is because of severe overlap of the IQR between military impulse noise and the other noise sources. In addition, the nature of the problem and the quantity being measured must also be considered. In the case of this problem, L_{pk} is not a good choice of metric because the levels of the sources being measured are all in the same approximate range (80-138dB), due to the adjustable threshold of the measurement device (and ultimately the noise monitor). The original blast monitoring systems performed inadequately because they simply looked for high L_{pk} values. The metrics L_{eq}, SEL, and L_{8eq} also do not yield any significant discrepancies amongst the classes of noise, since they are basically integrating the waveform across a set length of time. As such, these metrics do not offer any specific insight into the nature of the waveform. In summary, the total energy in a waveform is highly likely to be quite similar given the nature of the measured sources. Although kurtosis (Fig. 22) has some overlap between the IQR of aircraft and wind noise, crest factor (Fig. 23) has no overlap of the IQR for all three classes, making it particularly appealing as a classification metric. For all box plots, a few mild outliers are indicated with "+" symbols. From these figures, it is noticeable that kurtosis and crest factor tend to produce clearly larger values in the case of impulsive noise sources.
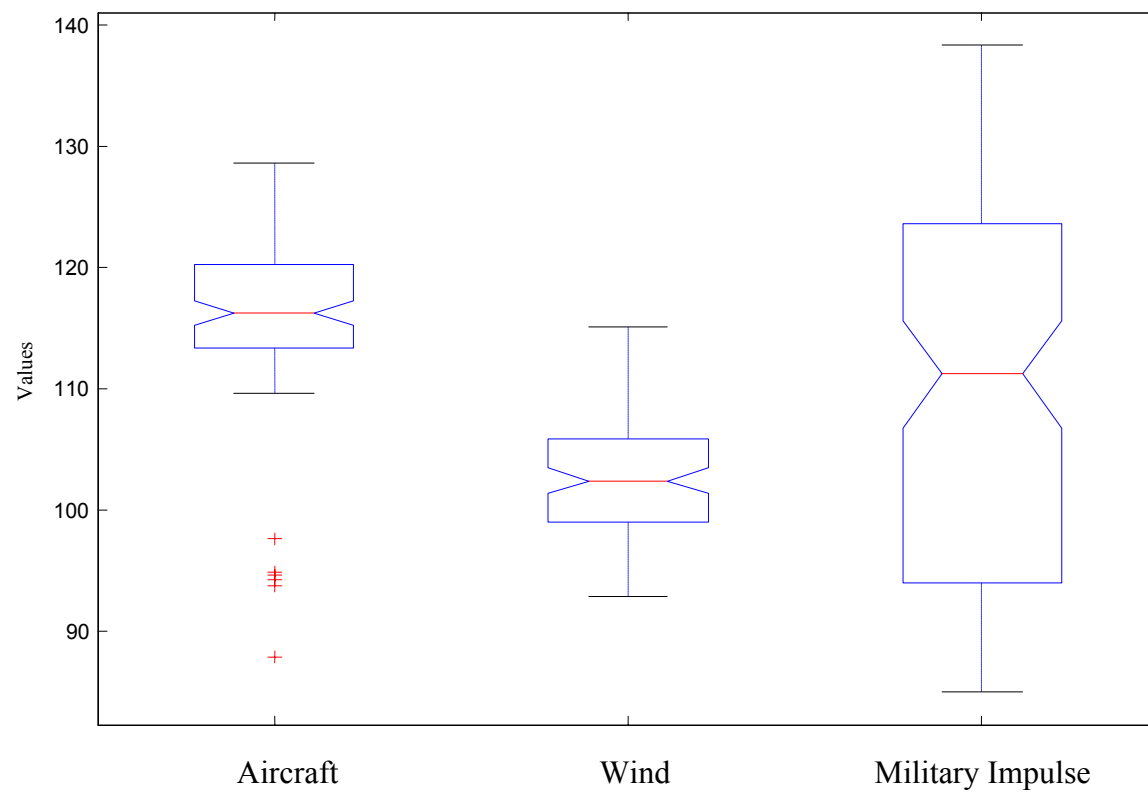
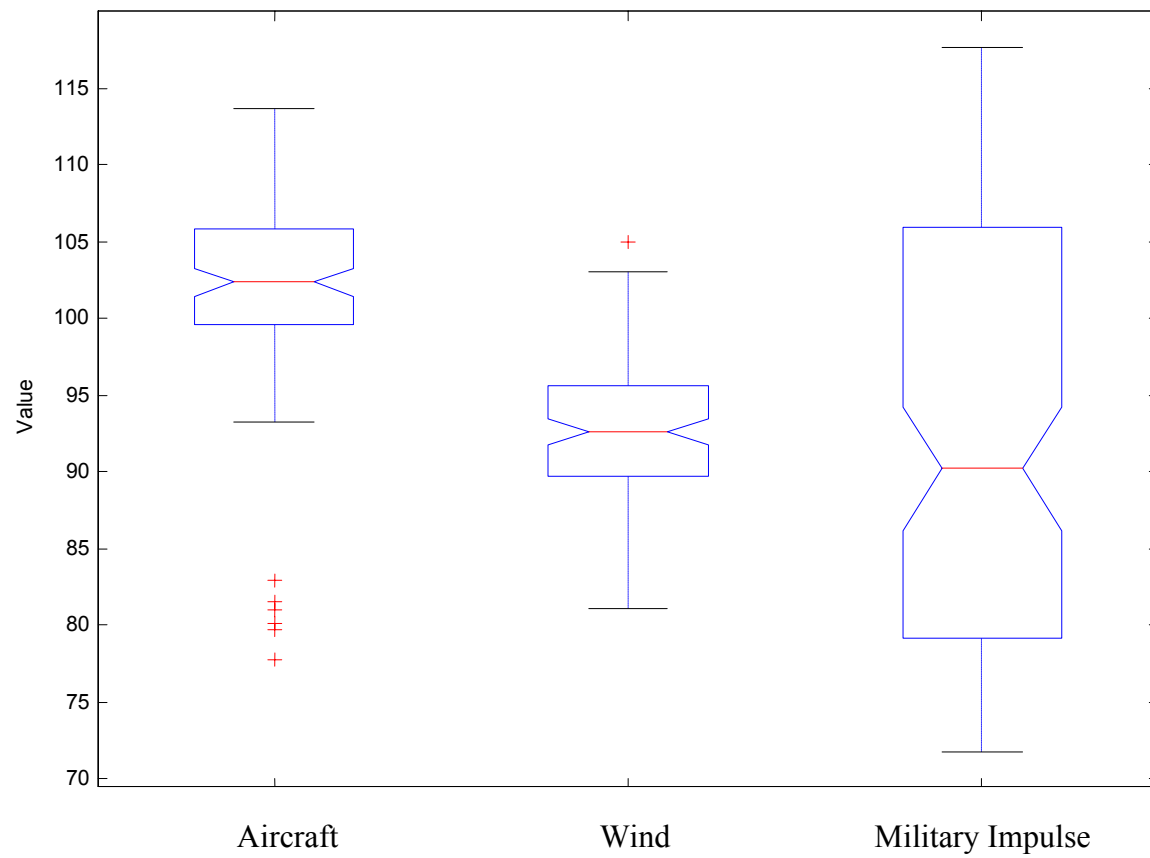**Figure 16.** Box plot of L_{pk} for noise sources

Figure 17. Box plot of L$_{eq}$ for noise sources
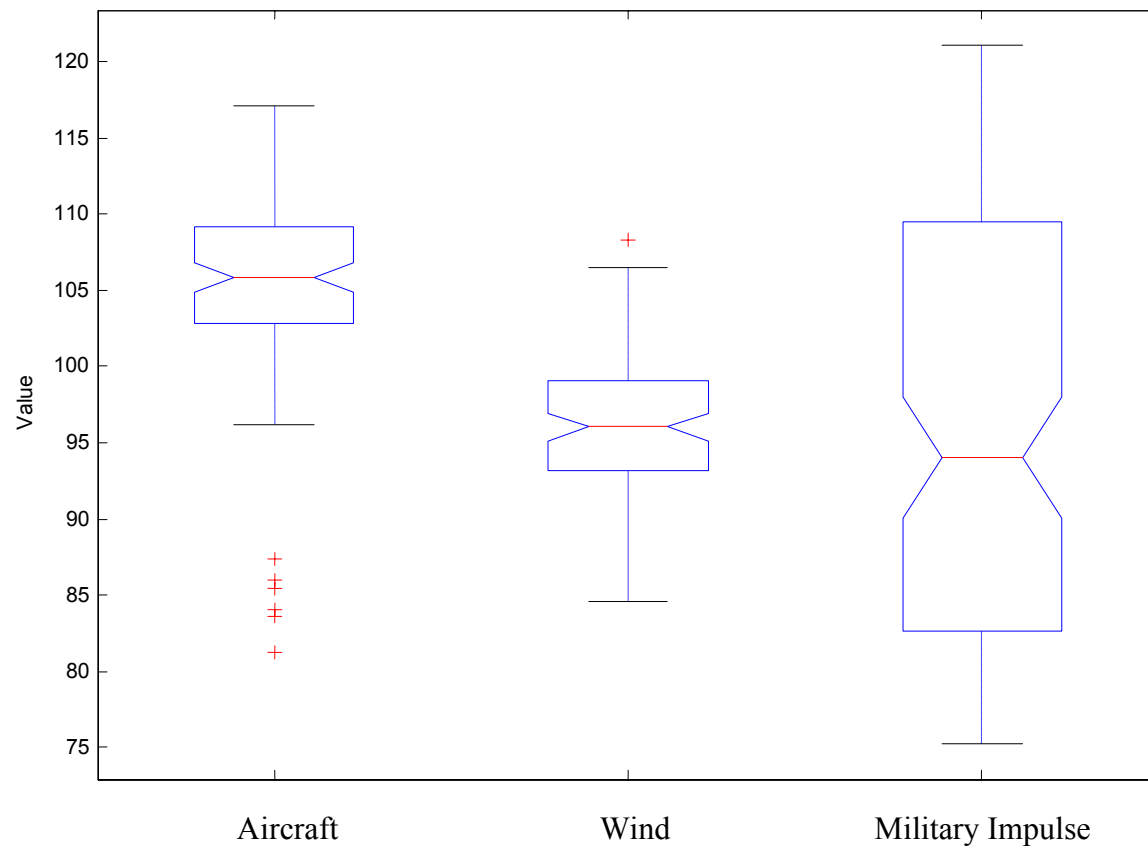
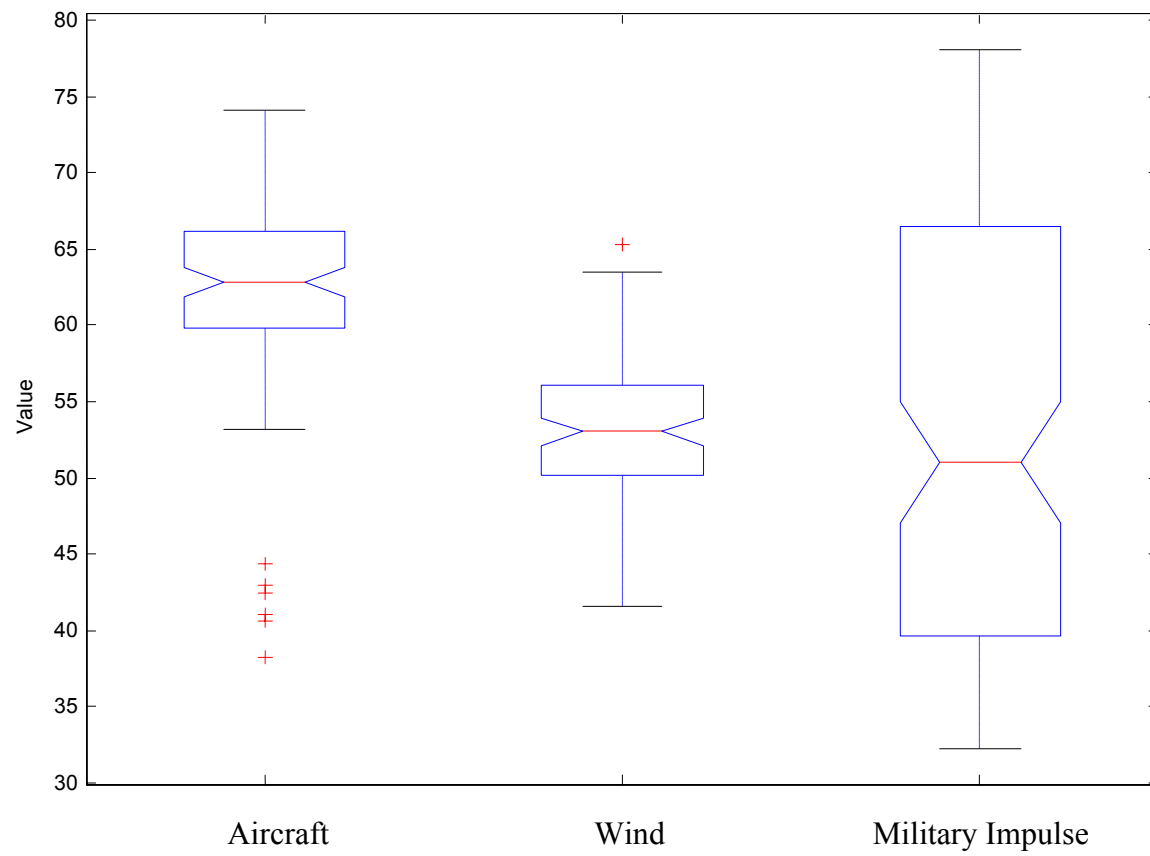**Figure 18.** Box plot of SEL for noise sources

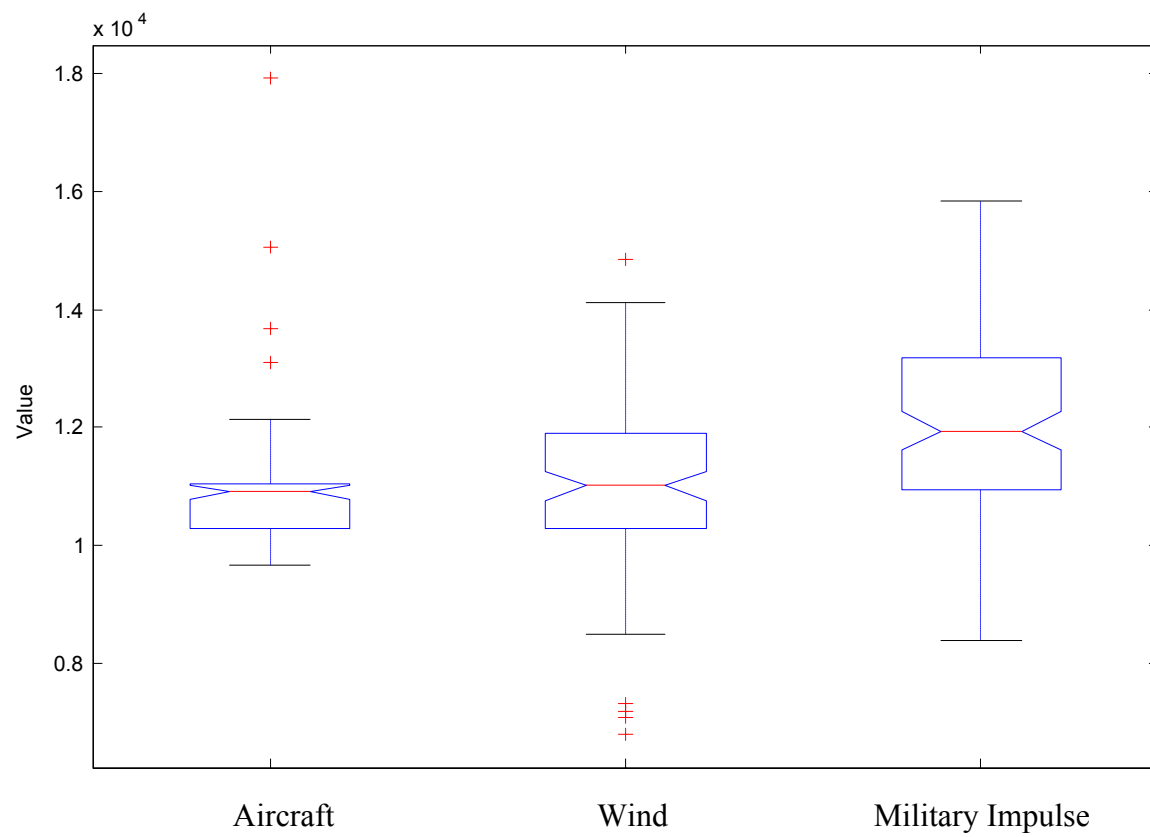**Figure 19.** Box plot of L_{8eq} for noise sources

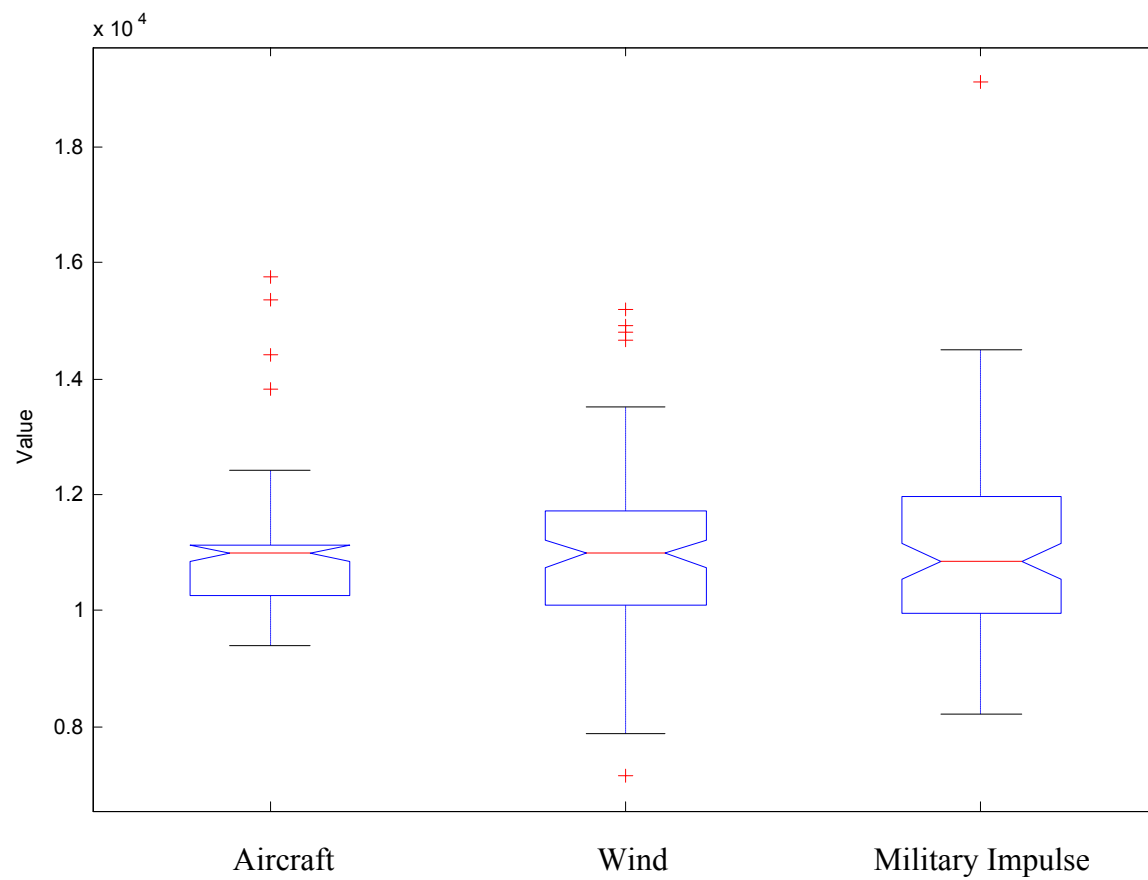**Figure 20.** Box plot of Pos for noise sources
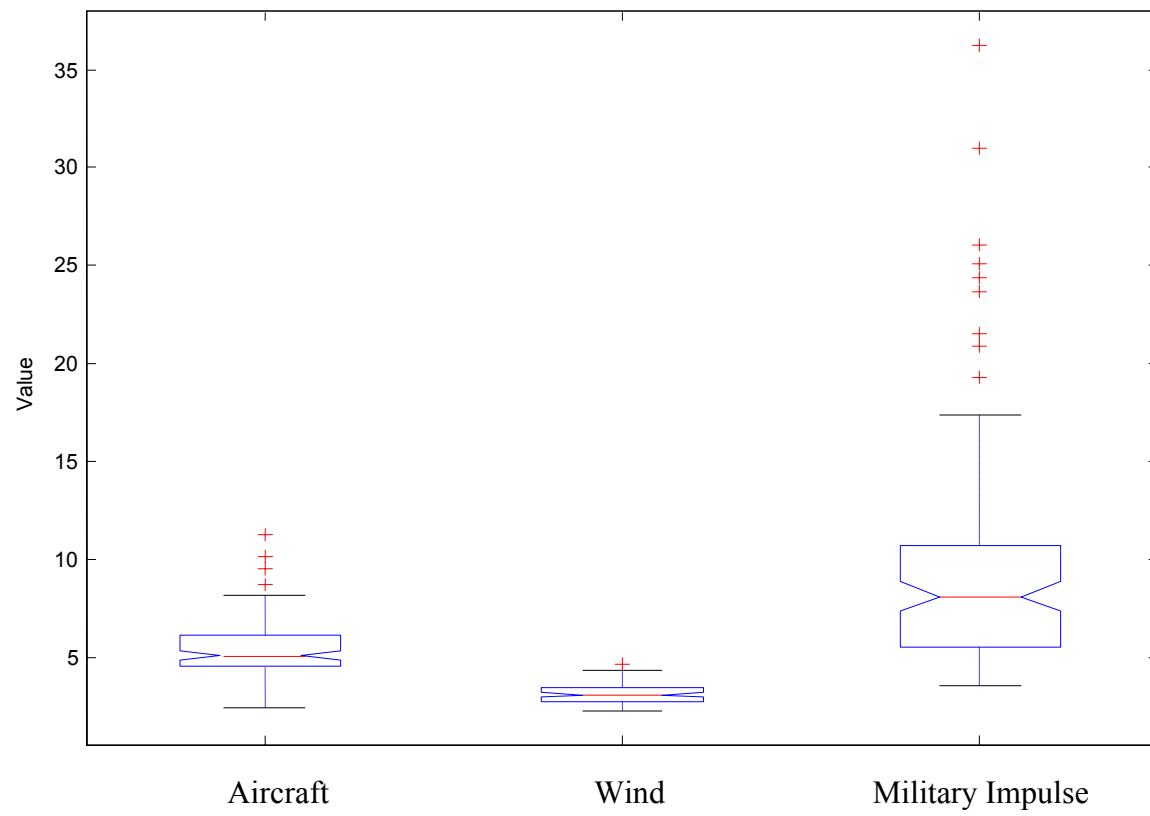
Figure 21. Box plot of Neg for noise sources

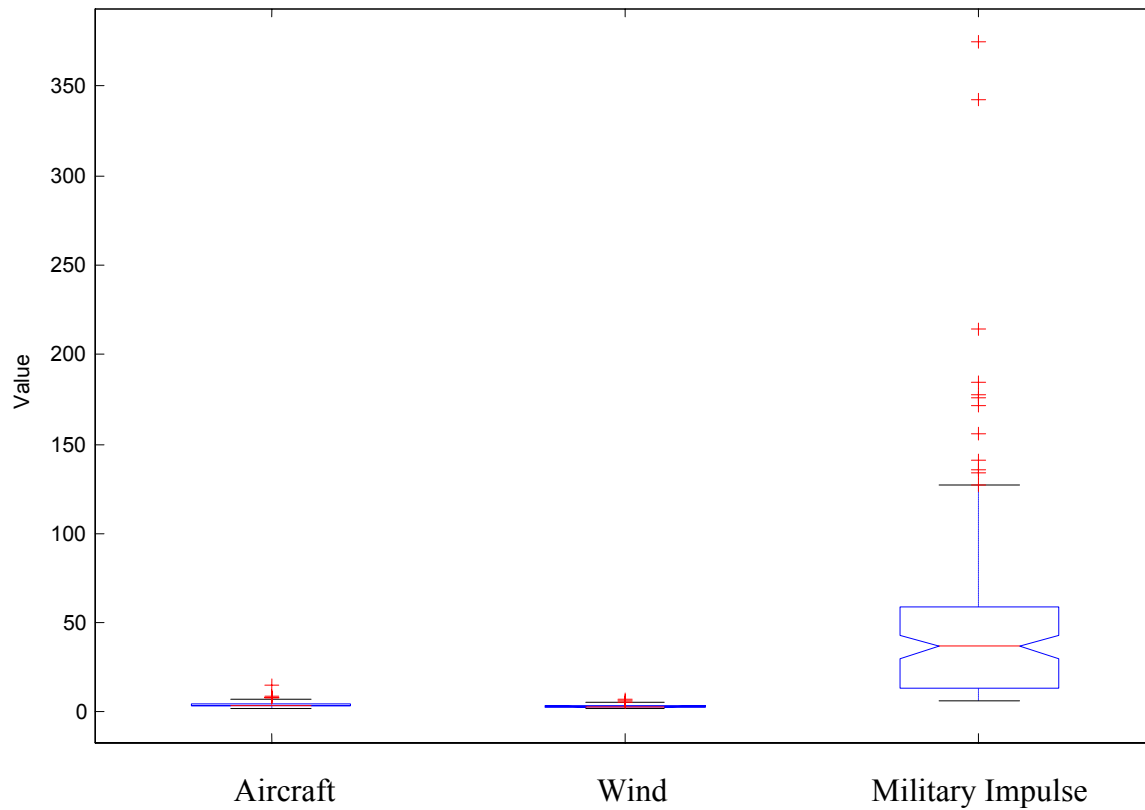**Figure 22.** Box plot of Crest Factor for noise sources

Figure 23. Box plot of Kurtosis for noise sources

Further, the data center about different values for each of the three sources. Definite trends are observable within the kurtosis and crest factor data sets. Kurtosis has no overlap between the IQR of military impulse noise and the other noise sources. Given this observation, it is believed that kurtosis would provide the best insight into the origin of a waveform of the traditional acoustic metrics. Crest factor has only a slight overlap of the IQR when comparing military impulse noise to aircraft noise and no overlap of IQR when comparing military impulse noise to wind noise. Thus, it is believed that crest would at least perform well in separating military impulse noise from wind noise and possible provide some separation of military impulse noise from aircraft noise. Figures 24 and 25 show the distribution of *m* and *WSE,* respectively.
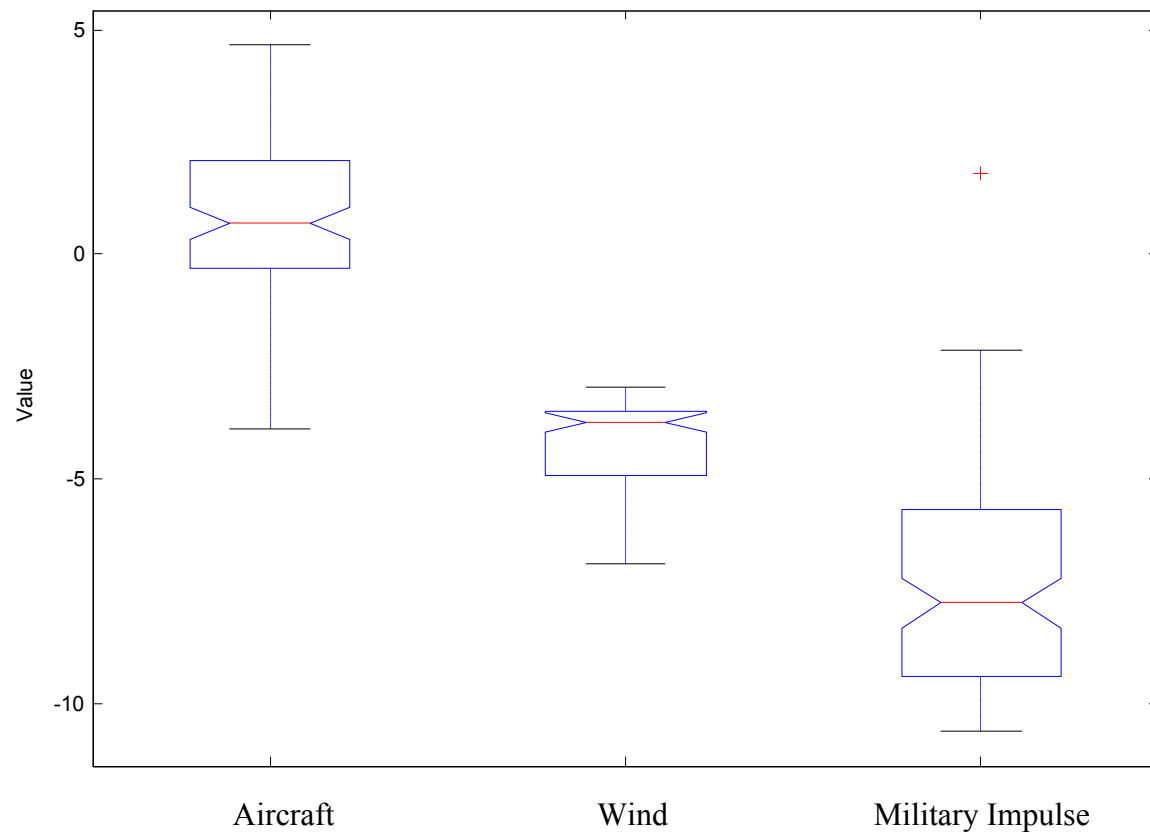
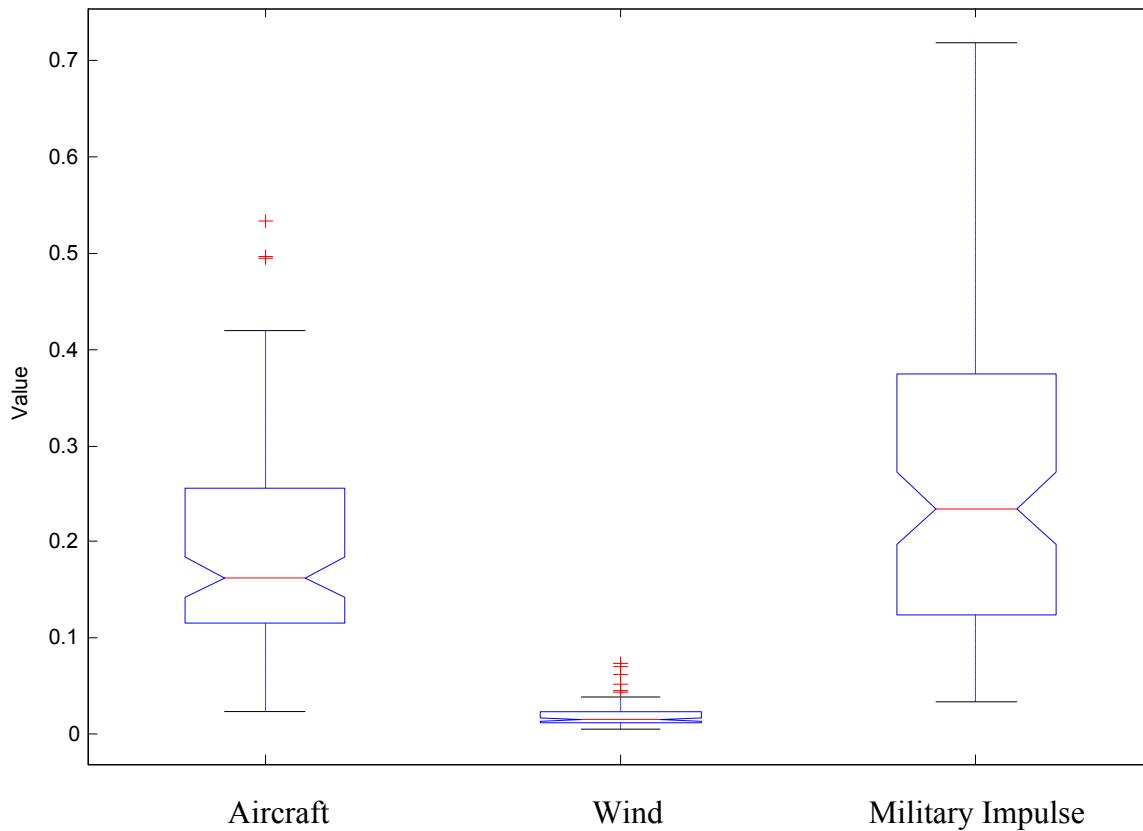**Figure 24.** Box plot of Spectral Slope for noise sources

Figure 25. Box plot of Weighted Square Error for noise sources

In figure 24 it is seen that *m* makes a strong differentiation between aircraft noise and both wind noise and military impulse noise, as only the whisker of aircraft noise only partially overlaps both military impulse noise and wind noise. There is some differentiation between military impulse noise and wind noise but their respective IQRs are still very close to one another. Figure 25 show a strong difference between wind noise and both military impulse and aircraft noise, as there is no IQR overlap between the respective sources. However, there is a large IQR overlap between aircraft noise and military impulse noise in this case. As these two metrics were designed to work in tandem, by applying some deductive reasoning, it is possible to answer: Is the recording aircraft noise or not aircraft noise? and, Is the recording wind noise or not wind noise? If the recording is not aircraft noise and not wind noise and we are only considering three

noise sources, then the recording is military impulse noise. From figures 16 through 25, kurtosis, crest factor, m, and WSE had the most statistical difference, but overlap of whiskers indicate no single metric is capable of perfect classification. Thus, a combination of metrics may provide an increased classification accuracy. Although the feature space is 4-dimensional (crest factor, kurtosis, m, WSE), some insight can be gained by examining the relationship between pairs of metrics. Figures 26 through 31 shows each of the metric combinations plotted against each other. In these plots, **x** signifies the point the given metrics map to for a recording containing military impulse noise and **o** shows the corresponding point for a recording not containing military impulse noise. In the case of useful metrics, it is desired to distinct clusters for each of the data types. A division of data types is observed in each plotted pair of metrics, however there is overlap in each case. A line has been added to each of these plots illustrating that there are indeed distinct clusters of data. A box has also been added to draw attention to the regions where the clusters overlap. In laboratory examination of the data space 3-d plots were constructed but they do not translate well to the paper format.
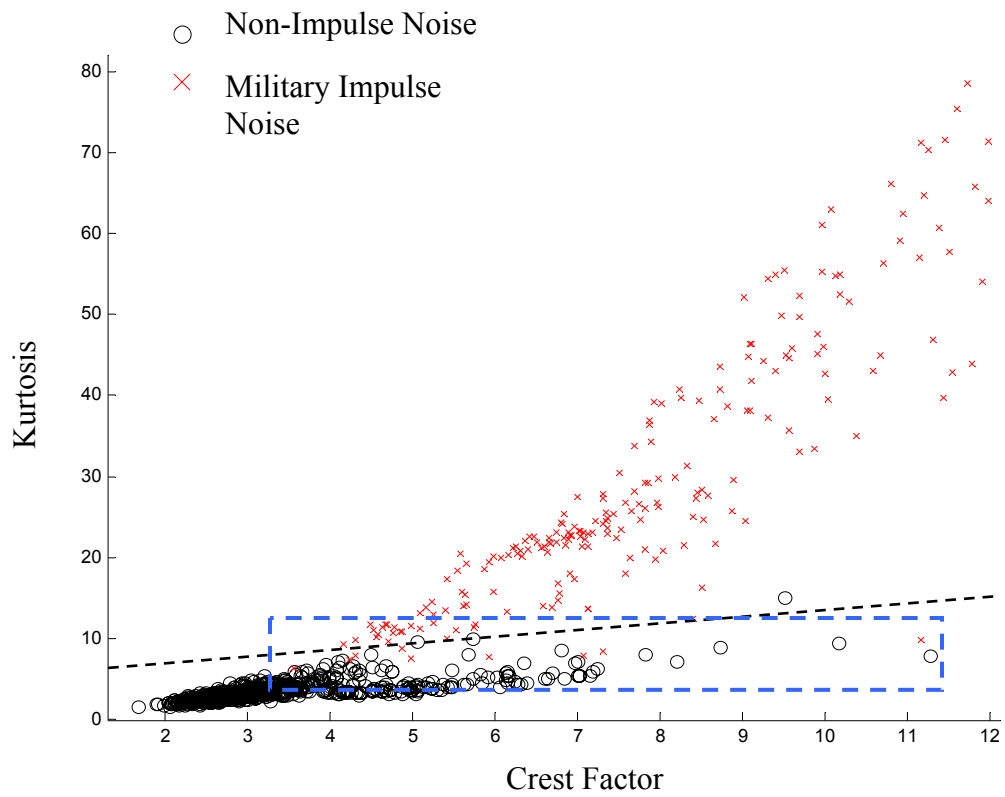
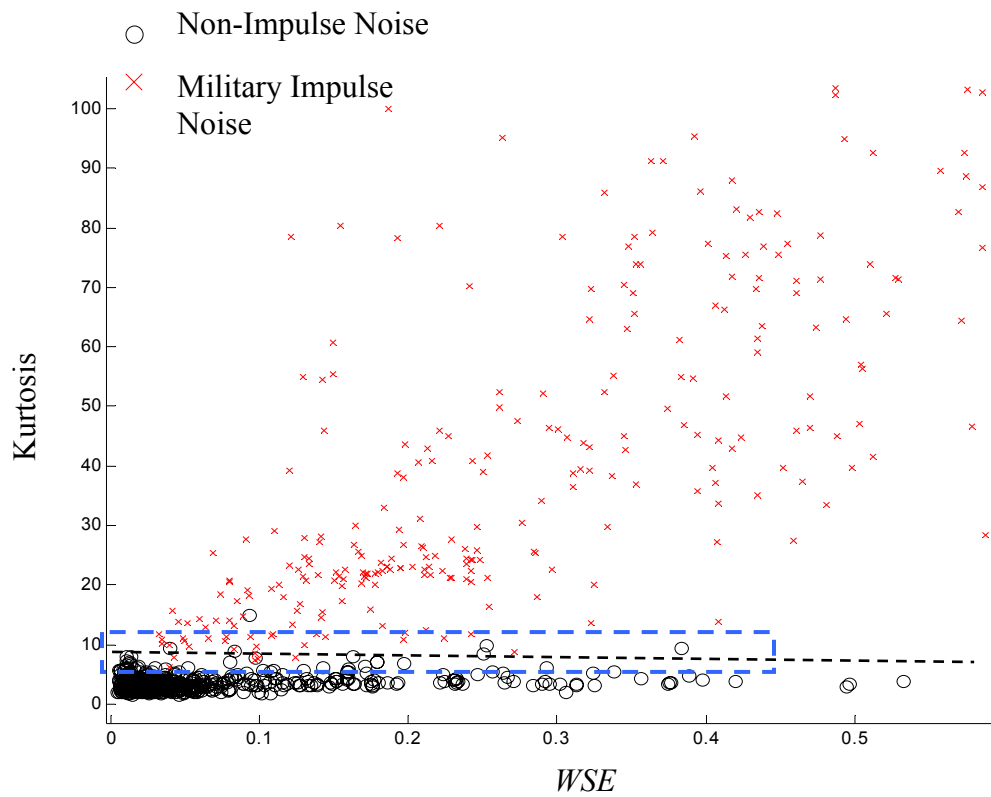**Figure 26.** Kurtosis plotted against Crest Factor for all data points

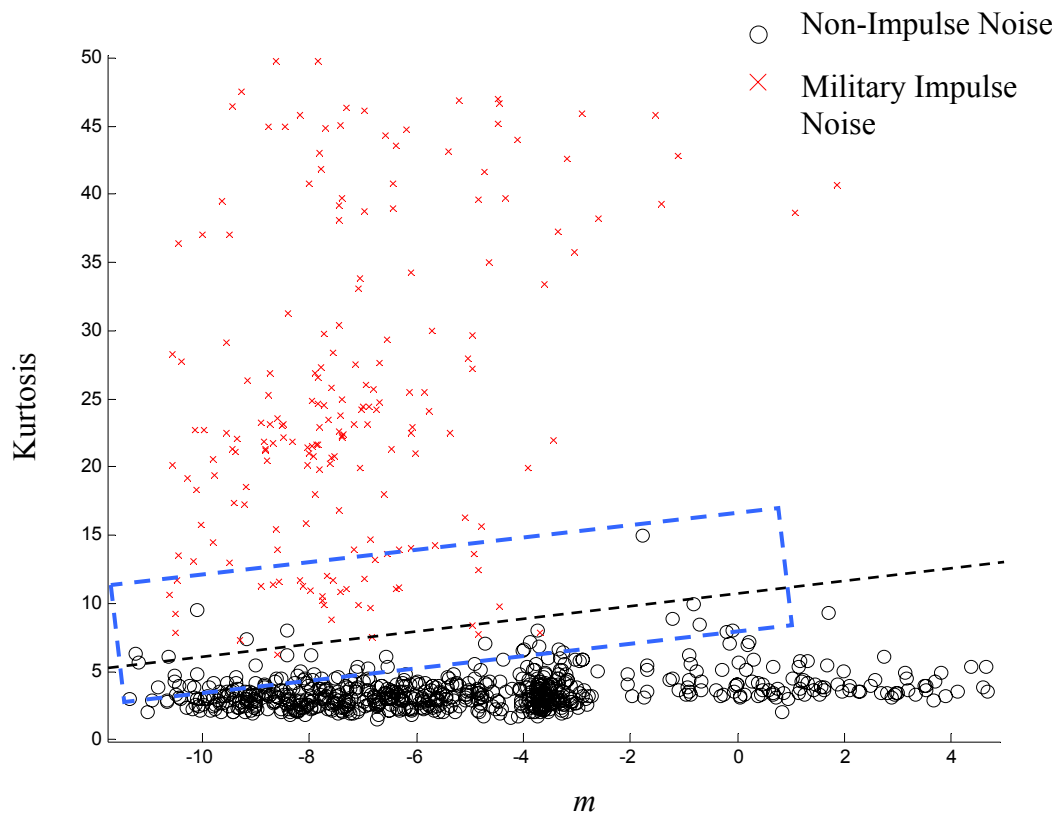**Figure 27.** Kurtosis plotted against *WSE* for all data points

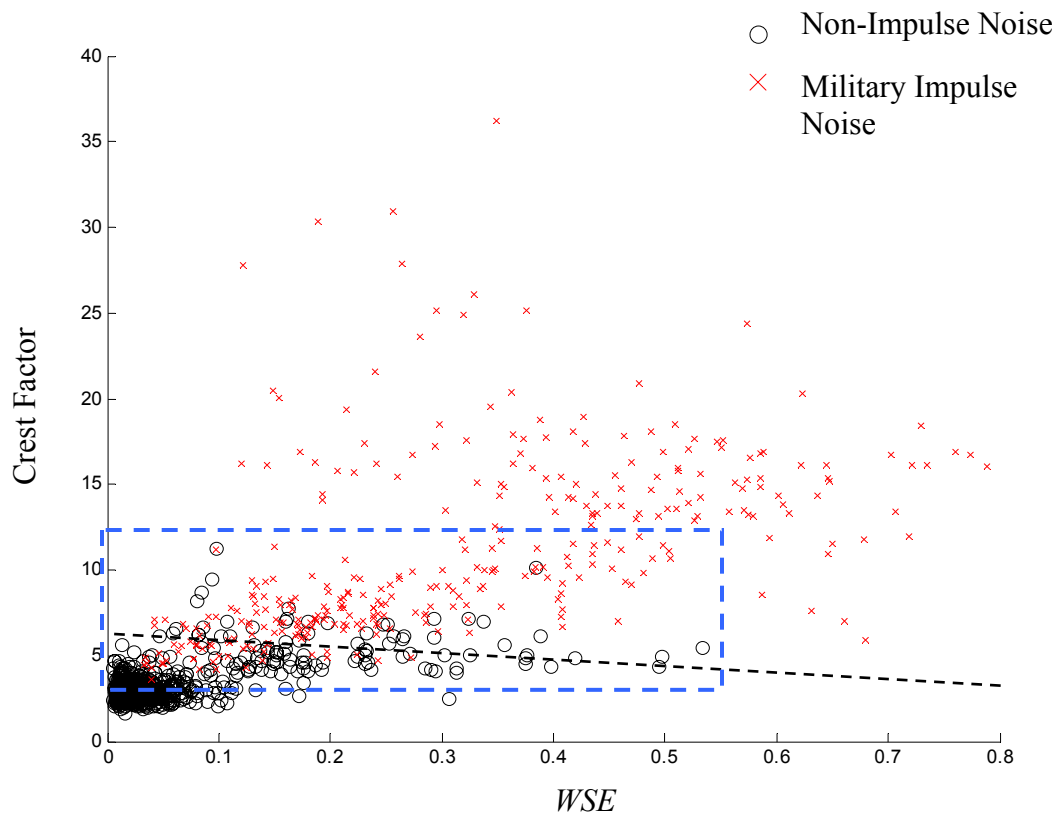**Figure 28.** Kurtosis plotted against *m* for all data points

Figure 29. Crest Factor plotted against *WSE* for all data points
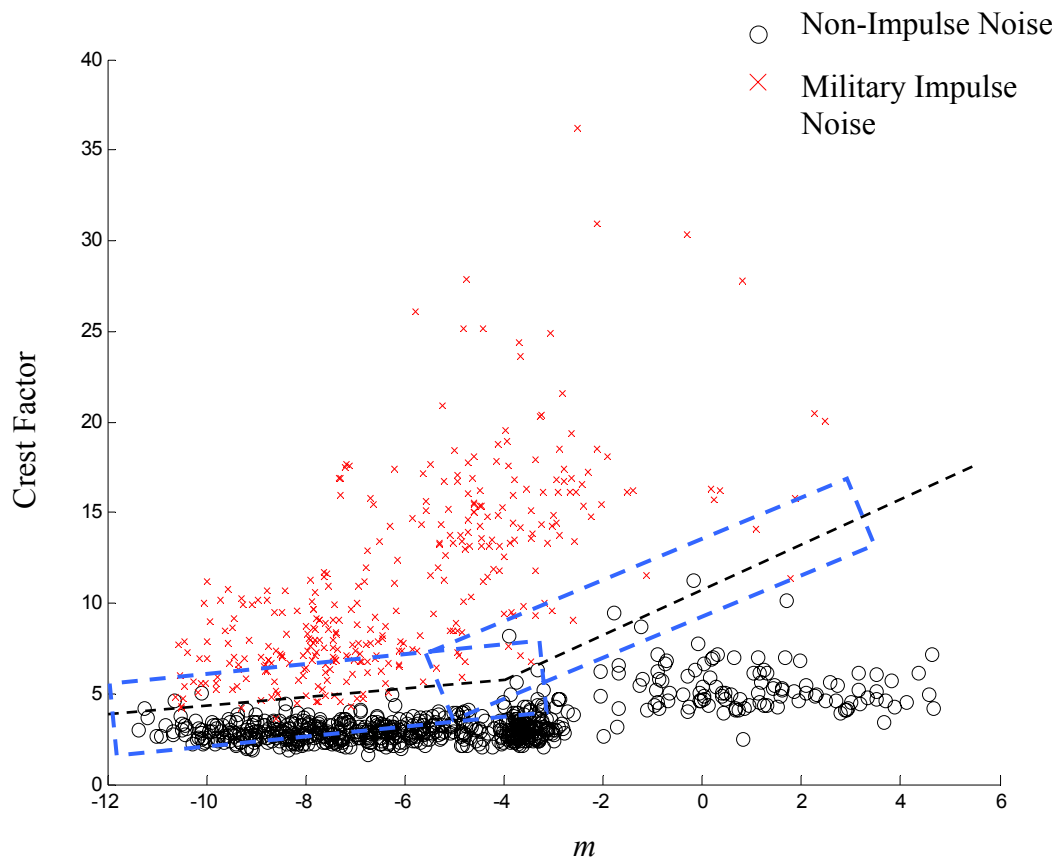
Figure 30. Crest Factor plotted against *m* for all data points
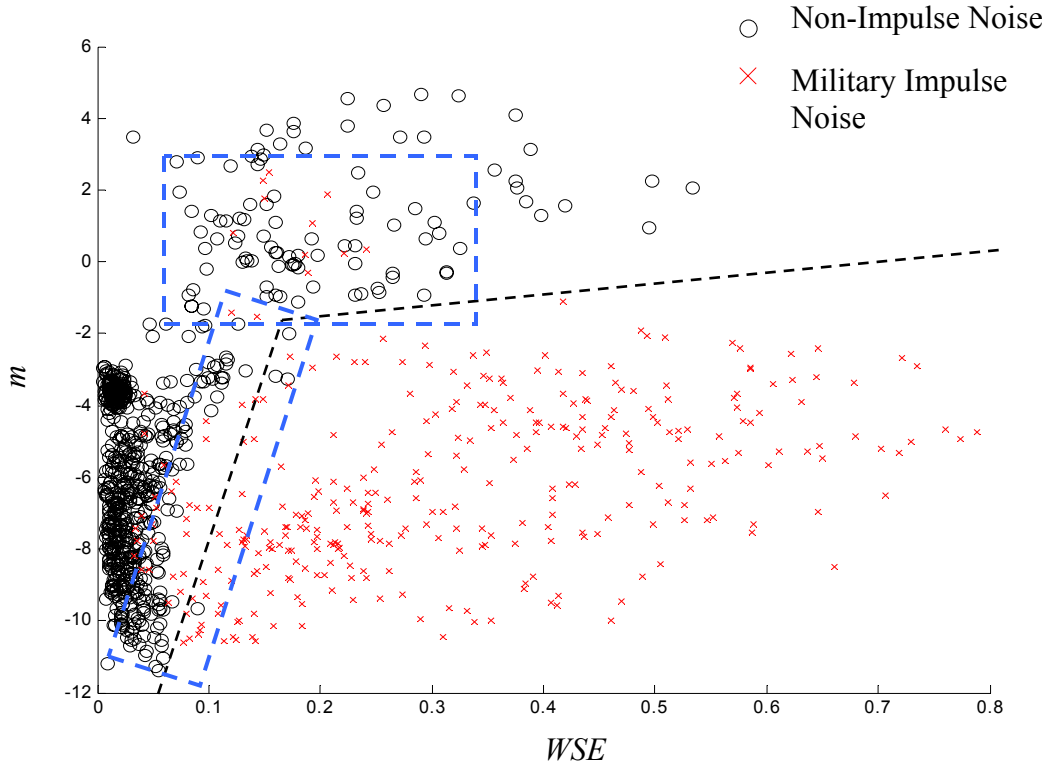
Figure 31. *m* plotted against *WSE* for all data points

In the case of plots dealing with kurtosis and/or crest factor (Figures 26 through 39), the chief cause of overlap between the data clusters were multiple events occurring in one record, for military impulse events, and aircraft noise and transient wind behavior (gusts etc.), for non-impulse events. The most interest pair of metrics is *m* and *WSE*. Figure 32 presents a more in depth evaluation of the values of *m* and *WSE* plotted against each other for the entire set of collected waveforms.

Figure 32. Spectral Slope (*m*) plotted against Weighted Square Error (*WSE*) for all data points

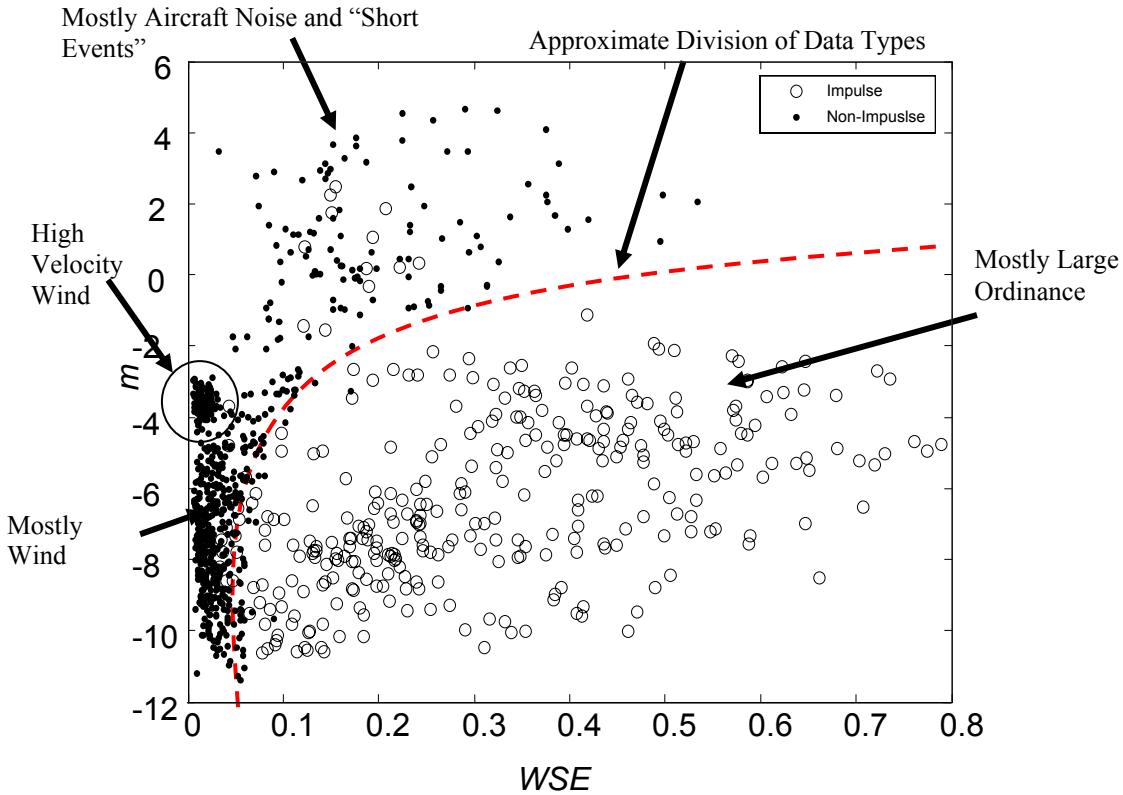The locations of various categories of military impulse and non-impulse noise classes are indicated in Figure 32. As seen in Figure 32, the decision boundary between the two classes of noise is non-linear. Thus, the non-linear modeling capabilities of the ANN will be useful in constructing this decision boundary. There is however, cluster overlap within this data set. These can occur, for example, with military impulse noise when the event is particularly short and quick to peak. These very short events, such as 60 mm rockets breaking the sound barrier, can have a PSD that is much closer to flat within the bandwidth of interest, much like an ideal impulse (equal energy at all frequencies). They do not share as many similarities of their PSDs with large ordinance, such as 155 mm Howitzers. However, these events have a very large value for kurtosis and crest factor, making these complementary to the frequency-domain metrics.

Figure 33 shows a processed waveform containing several 60 mm rocket shots, illustrating the aforementioned exception.
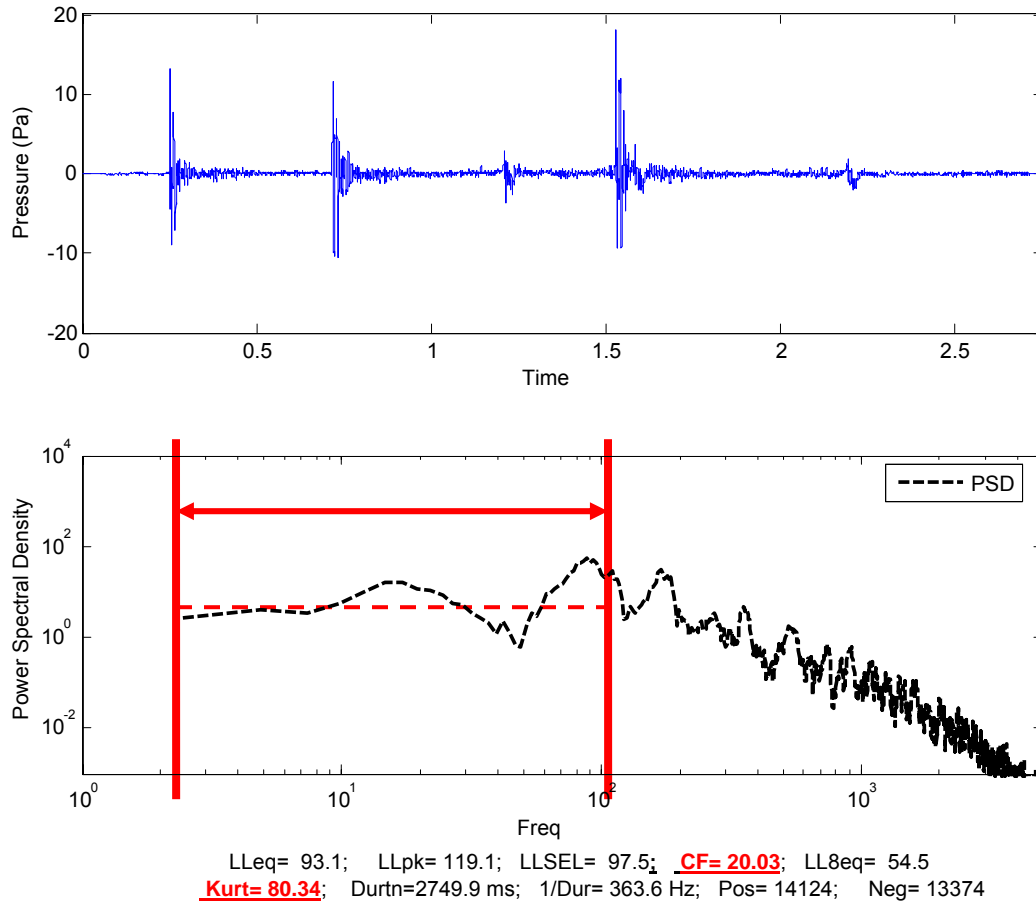
Figure 33. Processed waveform of 60mm rocket noise

As predicted, the PSD trend is relatively flat within the BWI yet, comparatively large values for kurtosis and crest factor have been computed. The ANN structures will work well to take into account cases were some of the metrics strongly indicate the presence or absence of impulse noise while the other metrics may produce inconclusive results. A similar case can occur with the outliers in the non-impulse noise section of the data. These events often had comparatively low L_{pk} values and less than ideal shapes for their PSDs. In an actual implementation of these algorithms in an actual monitoring system, these events will probably be of little interest due to their sufficiently low amplitude (<100dB).

The frequency-domain metrics become particularly useful in the case of multiple events occurring in a very short period of time. Since the temporal and statistical characteristic of a waveform can change significantly for these cases, comparatively lower values for kurtosis and crest factor could be calculated for a waveform of given length. In comparison, as shown in Figure 34, the PSD calculated over a recording containing multiple events will retain a similar shape to the PSD calculated for a single impulse event (Figure 14).

LLeq= 97.9;   LLpk= 111.7;   LLSEL= 102.3;   **CF= 4.87**;   LL8eq= 59.3
Kurt= 8.75;   Durtn=2749.9 ms;   1/Dur= 363.6 Hz;   Pos= 14031;   Neg= 13467
m= -3.2998, WSE=0.1380
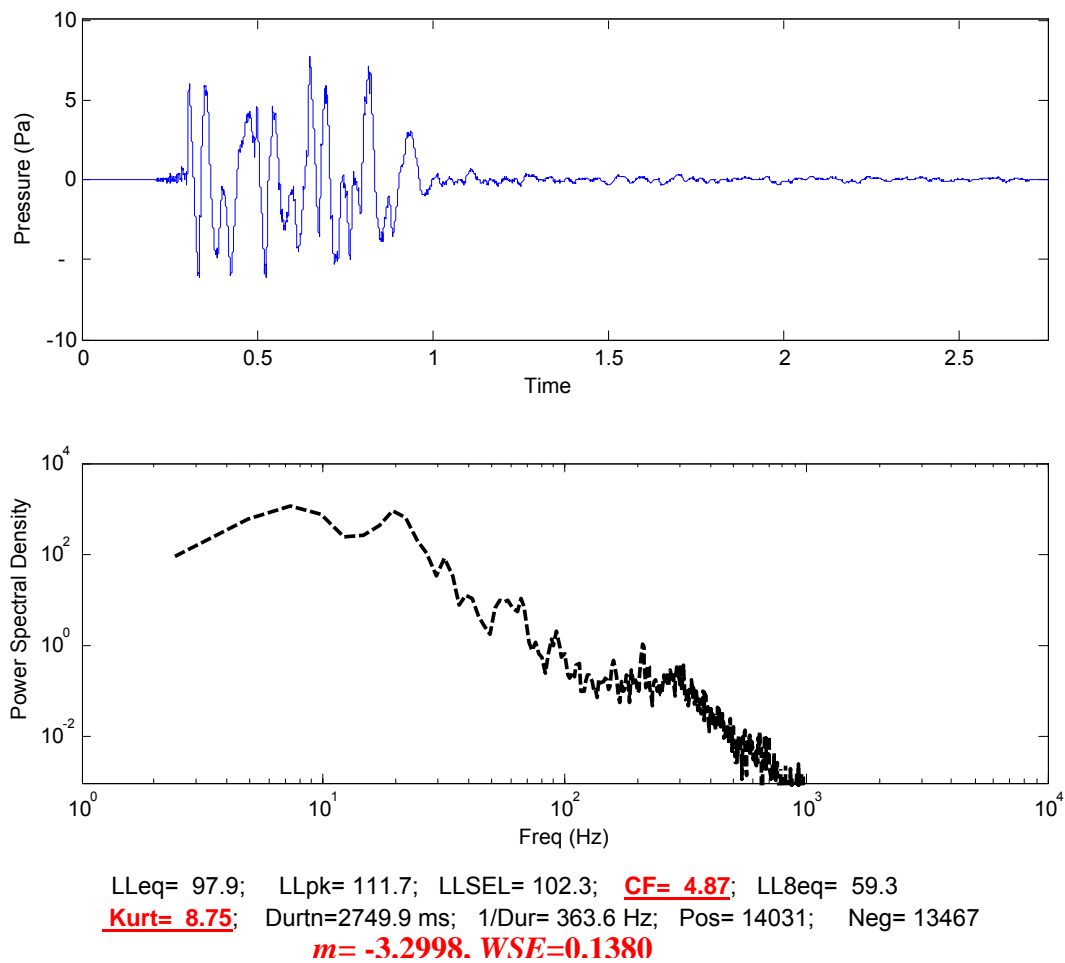
Figure 34. Processed waveform of three 81mm mortar shell exploding nearly simultaneously

Figure 34 shows the processed waveform of three 81 mm mortar shell exploding nearly simultaneously. The computed values for kurtosis and crest factor are comparatively lower, to the point where classification errors may occur. However, the values for *m* and *WSE* fall within the range of military impulse noise. With this input, the classifier is now able to make a correct decision. Thus, by combining frequency-domain and time-domain metrics, the detection of impulse noise is improved.

7.0 CLASSIFIER CONSTRUCTION

To train and validate the classifiers, 1,000 sets of metrics were computed from processing each of the 1,000 waveforms. Two-thirds of the data were selected in a stratified-random sample to serve as the training data, and the remaining one-third were used for evaluation. The one-third of the data used for evaluation was divided into two equal parts, half to be used for training validation and half for test data. The classifiers were first tested with a single input of kurtosis or crest factor to test the hypothesis that these metrics were a good indicator of the presence of military impulse noise within a waveform. *WSE* and *m*, were also tested to determine the utility of these metrics, which were designed to act as a pair. The classifiers were then trained and tested with both kurtosis and crest factor used as inputs. Finally, kurtosis and crest factor plus the two spectral metrics, *WSE* and *m*, were evaluated using the sequential forward selection method to investigate whether the combination of these metrics would produce a classifier with greater accuracy. By using this method, the utility of each metric is observable along with the utility of the combinations of metrics. The classifiers to be evaluated are the support vector machine with radial basis function kernel, multi-layer perceptron, and self-organizing map. The configuration of each experiment in terms of the metrics used is presented in Table II.

Table II. Summary of Metric Configuration for Each Experiment

Experiment Label	Number of Inputs to Classifier	Metrics Used as Inputs
A	1	Kurtosis
B	1	Crest Factor
C	2	*WSE* and *m*
D	2	Kurtosis and Crest Factor
E	4	Kurtosis, Crest Factor, *WSE*, and *m*

7.1 SUPPORT VECTOR MACHINE WITH RADIAL BASIS FUNCTION KERNEL

The SVM Networks began with 1, 2, or 4 input neurons, 667 neurons in the hidden layer, and 2 output neurons (one indicating military impulse noise and one indicating non-impulse noise), as the network was created as an exact fit to the training data. The network was then pruned to 28 hidden neurons. Since the final network contains 28 hidden neurons, the entire distribution of data modeled with a total of 28 multi-Gaussian distributions. A Gaussian RBF spread factor of 10 was found to produce the best results.   The Gaussian spread factor is a parameter that determines the width of the multi-Gaussians placed at each point of training data. Any data point that lies within a Euclidean distance of 10 or less from a given neuron will produce an activation of at least 0.5 at the given neuron.

7.2 SELF-ORGANIZING MAP

The SOM consisted of 16 neurons arranged in a 4-by-4 square lattice. This configuration was chosen because it was the smallest size network that produced a satisfactory result. The network learning parameters were set per recommended procedures of training SOMs (Haykin, 1999). During the ordering phase, the initial learning rate parameter was set to 0.1 with a time constant of 1,000 and the initial size of the neighborhood function was set to 2 (the approximate radius of the lattice) with a time constant of 1000/log(2). The ordering phase was conducted for 1,000 iterations. During the lattice convergence phase, the learning rate parameter is held at its value at the end of the ordering phase and the neighborhood function continued to decrease. The lattice convergence phase was conducted for 20,000 iterations.

7.3 MULTI-LAYER PERCEPTRON

The structure of the MPL was heuristically optimized, as is common practice. The networks began with 1, 2, or 4 input neurons and 2 output neurons. Additional nodes and layers were added to the network until no further performance gains could be achieved (Haykin, 1999). The final MPL architectures consists of 1, 2, and 4 input neurons, layers 3 of 4 hidden neuron with tan-sig activation functions, and 2 output neurons with log-sig activation functions. The network was trained using the Reduced Memory Levenberg-Marquardt algorithm with a learning rate of 0.001. Training was conducted for 1,000 epochs.

7.4 LEAST SQUARES CLASSIFIER

For comparison purposes, linear least squares classifiers using between 1 and 20 terms were tested on the data set. It was found that the linear classifiers performed almost identically regardless of the number of terms used. The accuracy of the least squares classifier was taken to be the configuration of terms that produced the maximum accuracy, a marginal improvement over the mean accuracy evaluated over all term configurations.

8.0 CLASSIFIER RESULTS AND DISCUSSION

The results for the classifiers respective accuracies are presented in Figure 35. The ANN structures performed comparably to the linear least-squares classifier in the cases of single metrics (kurtosis or crest factor) used as inputs.
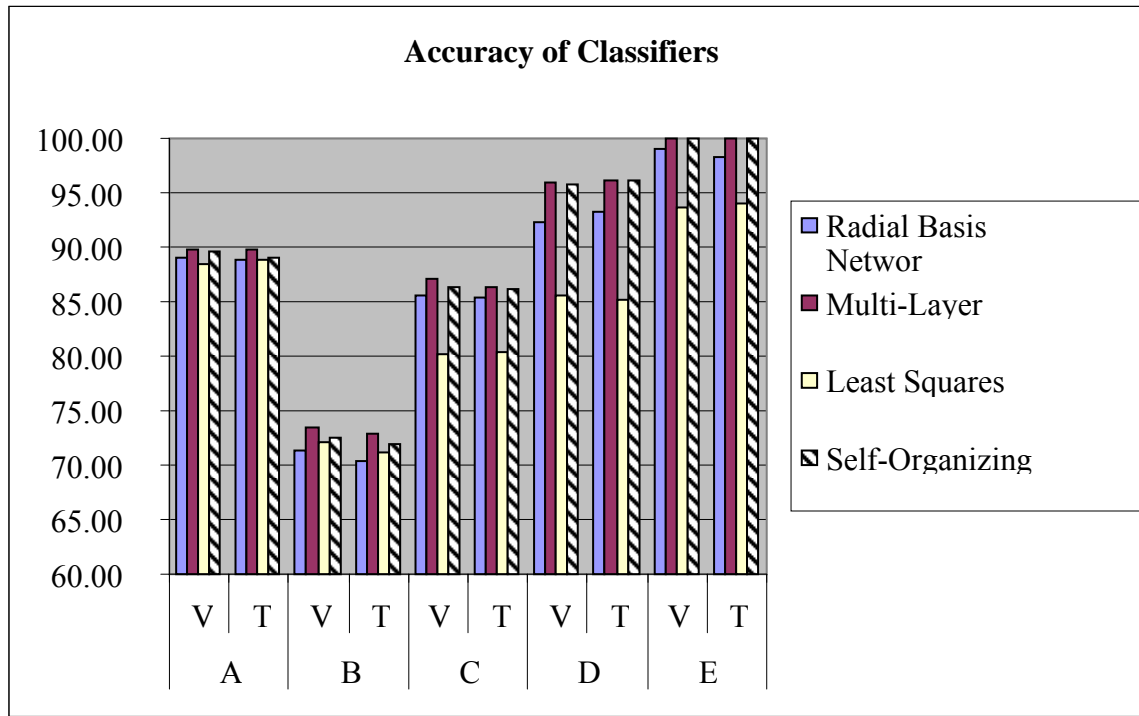

Figure 35. Summary of the accuracy of classifiers for different configurations of metrics: (A) kurtosis, (B) crest factor, (C) *WSE* and *m*, (D) kurtosis and crest factor, (E) kurtosis, crest factor, *WSE*, and *m* (V-Validation Data, T-Testing Data)

This is expected since all of the classifiers are simply computing a set threshold for this configuration. In all but the single metric classifiers, cases A and B, the ANN structures did out perform the linear least-squares classifier significantly. In case C, where *WSE* and *m* were used as inputs all of the classifiers performed approximately equally. It is seen that the ANN structures out-performed the least-squares classifier when using either the kurtosis and crest factor data set or the four-metric data set (kurtosis, crest factor, *WSE*, and *m*). It is also noticeable that all of the classifiers performed better, with respect to themselves, with the four-metric data set, as compared to the data sets with fewer metrics. For the four-metric classifier, the SOM and MPL both achieved 100% classification accuracy on the validation and test data while the RBF network achieved very similar performance with an accuracy of 99.1% on the test data and 98.2% on the test data. Figure 36 summarizes the false positive detection rates for the classifiers.
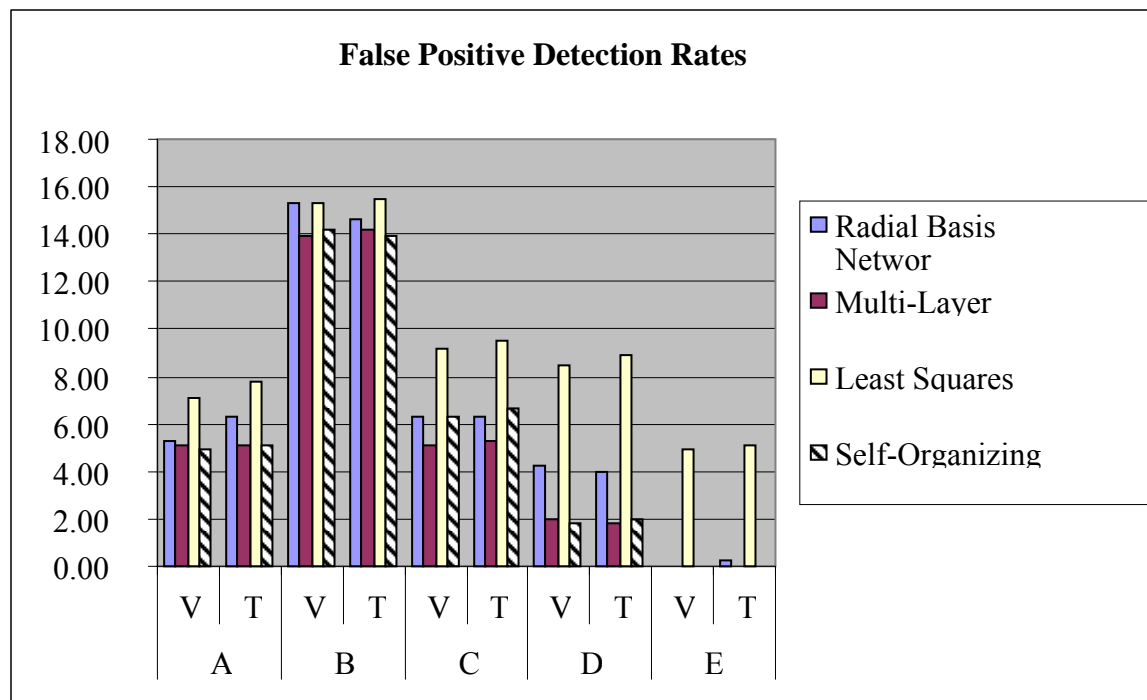
Figure 36. Summary of false positive detection of classifiers for different configurations of metrics: (A) kurtosis, (B) crest factor, (C) *WSE* and *m*, (D) kurtosis and crest factor, (E) kurtosis, crest factor, *WSE*, and *m* (V-Validation Data, T-Testing Data)
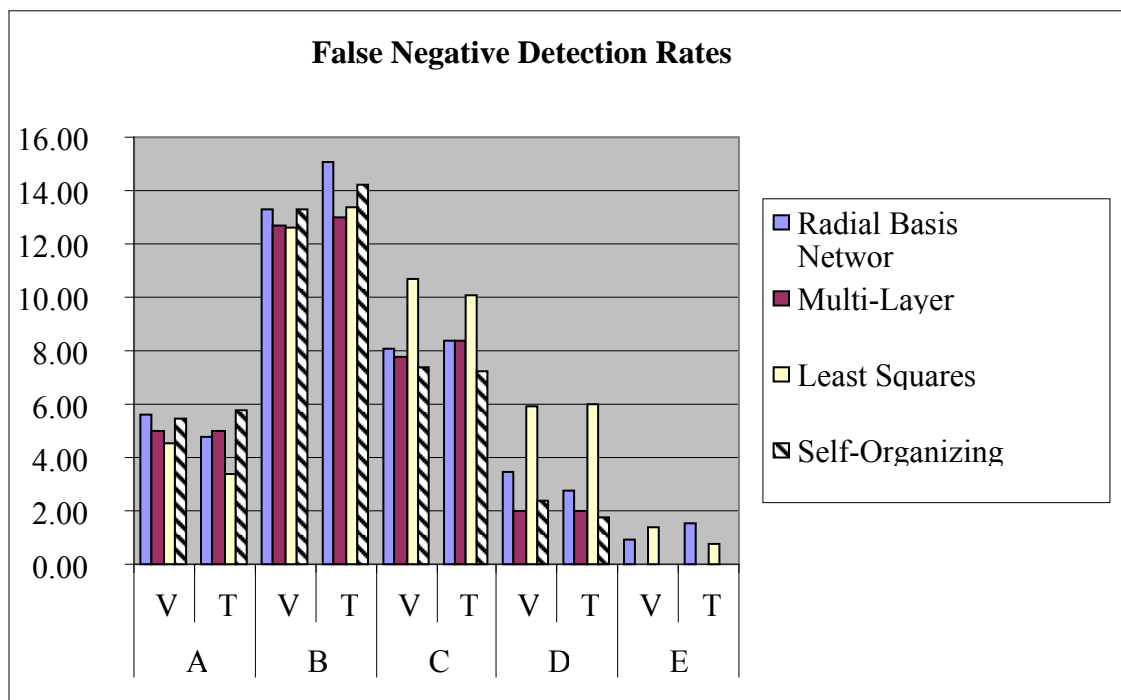

Figure 37. Summary of false negative detection of classifiers for different configurations of metrics: (A) kurtosis, (B) crest factor, (C) *WSE* and *m*, (D) kurtosis and crest factor, (E) kurtosis, crest factor, *WSE*, and *m* (V-Validation Data, T-Testing Data)

Obviously, the false positive and false negative detection rates for the classifiers are inversely proportional to the accuracy of the respective classifiers. In most cases, the false positive and false negative classifications were approximately evenly divided. This is not surprising since the threshold of test was set to 0.5 for the classifier trials. If the classifier was desired to be more sensitive to military impulse noise, then the detection threshold could be lowered. Increasing the sensitivity of the classifier comes with the cost of a greater chance of false positive detection. A visual representation of the classifier outputs, using the four-metric data set, is presented in Figures 38-40. In each of the figures, the military impulse noise events are represented by circles and non-impulse events (wind and aircraft) by squares. In Figures 38 and 39 a dashed line is used to indicate the decision boundary of the classifier.
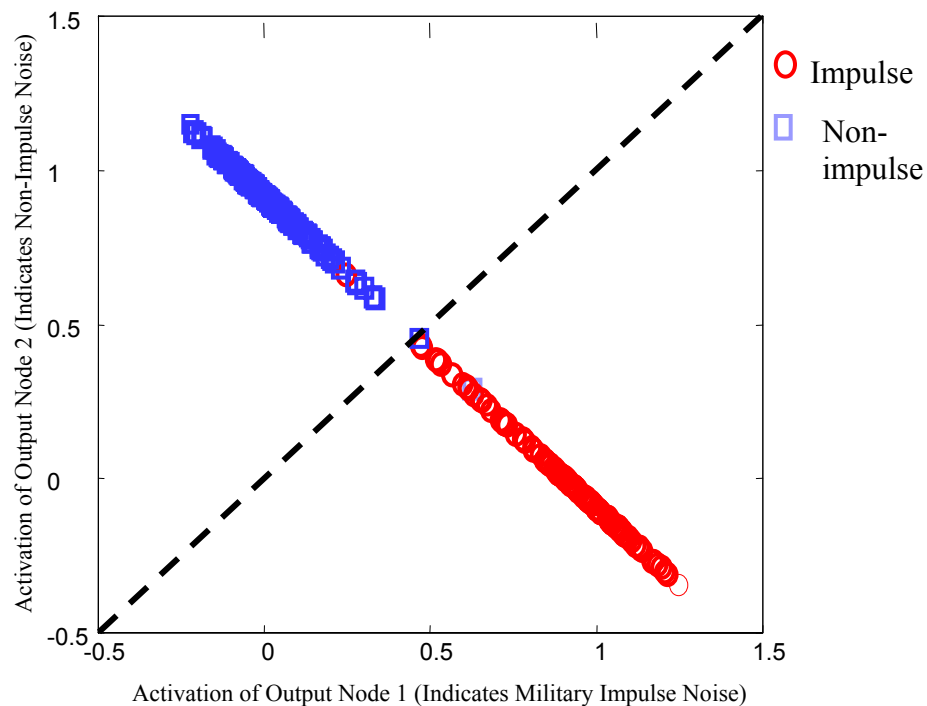

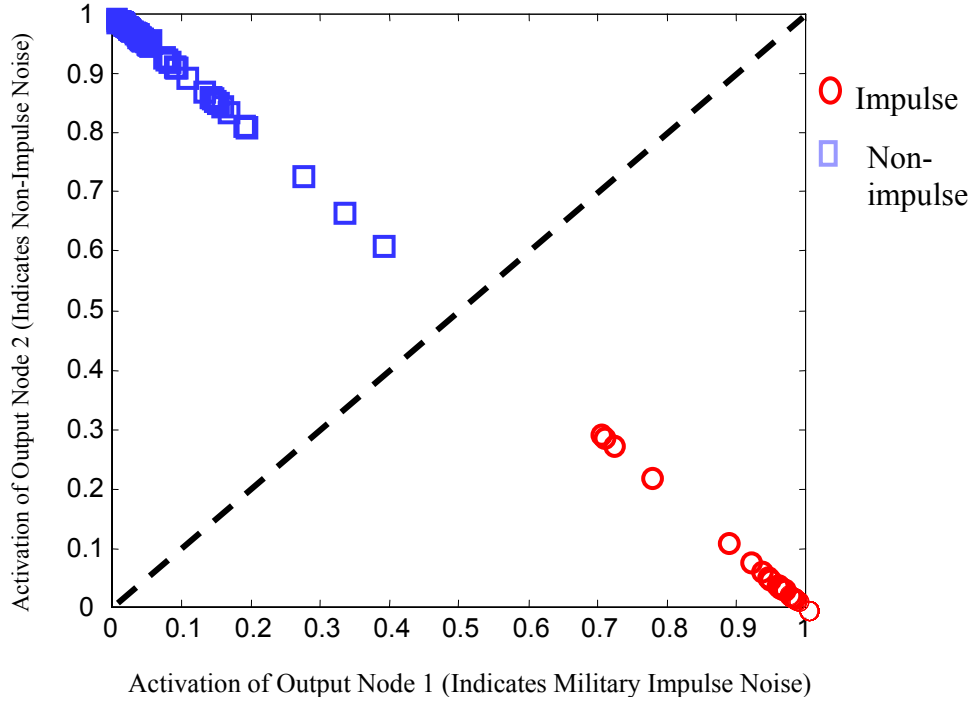**Figure 38.** Graphic Output of RBF Network using four-metric data set

Figure 39. Graphic Output of MLP using four-metric data set

When comparing the performance of the RBF network and the MLP (Figures 38 and 39 respectively), it is noticeable that the MPL produces a much clearer decision boundary than does the RBF network, which is consistent with the higher accuracy of the MLP. In fact, no matter where the data are separated in Figure 38, misclassification will occur. In Figure 40 it is noticeable how neatly the classes of data have been grouped within the SOM. The large group neurons on the left side of the map contain wind noise, the two neurons at the right center of the map contain the military impulse noise, and the single neuron on the upper right corner contains the aircraft noise. Thus, good performance of this classifier is also expected.
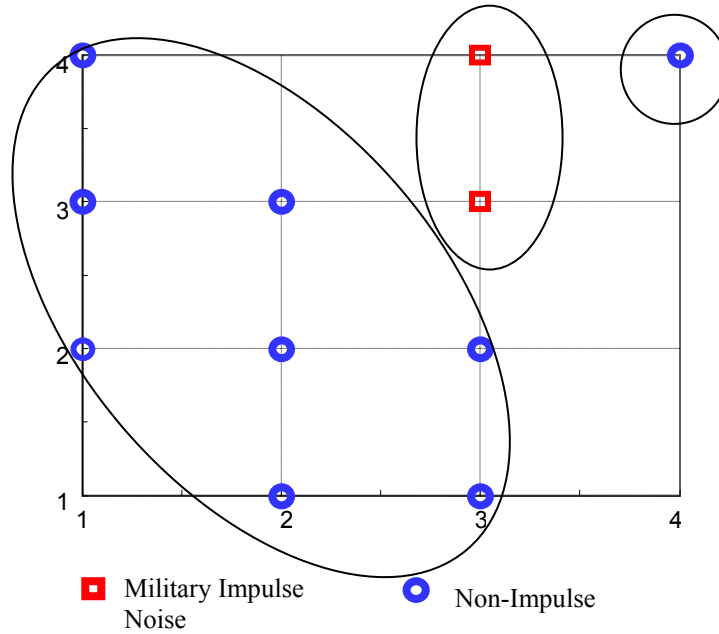
Figure 40. Graphic Output of SOM using four-metric data set

Any additional classes of data could be handled by increasing the number of outputs in the RBF and MLP (Bucci, 2006) and increasing the size of the SOM. Receiver operating curves for these classifiers are not presented, as they are deemed to be rather uninteresting, since in most cases, they would be nothing more than a rectangle given the high degree of accuracy for the ANNs.

9.0 TEMPORAL ANN APPROACH

Upon visual examination of the waveforms in this library, it became apparent that, after viewing the plotted waveform of a few examples of each type of noise source, an observer could easily identify which class of noise to which a waveform belonged based on the shape of the waveform. It was then hypothesized that if this task could be easily carried out by a human operator with little training, an algorithm based on the same principles could be developed.

9.1 OBSERVATION AND GENERALIZATIONS FROM COLLECTED WAVEFORMS

Figures 41 through 43 show typical examples of military impulse noise, wind noise, and aircraft noise. Casual observation of a few examples of each type of would most likely yield highly accurate classification of any additional waveforms presented to a human observer. While this classification is a simple task for humans to perform, development of a set of rules for a classification algorithm to abide by may not be quite so trivial. A human observer does not rely specifically on each data point within a waveform to produce a judgment. Alternatively they make a generalization based on the approximate distribution of data points. It is hypothesized that an artificial neural network can be taught to do the same.
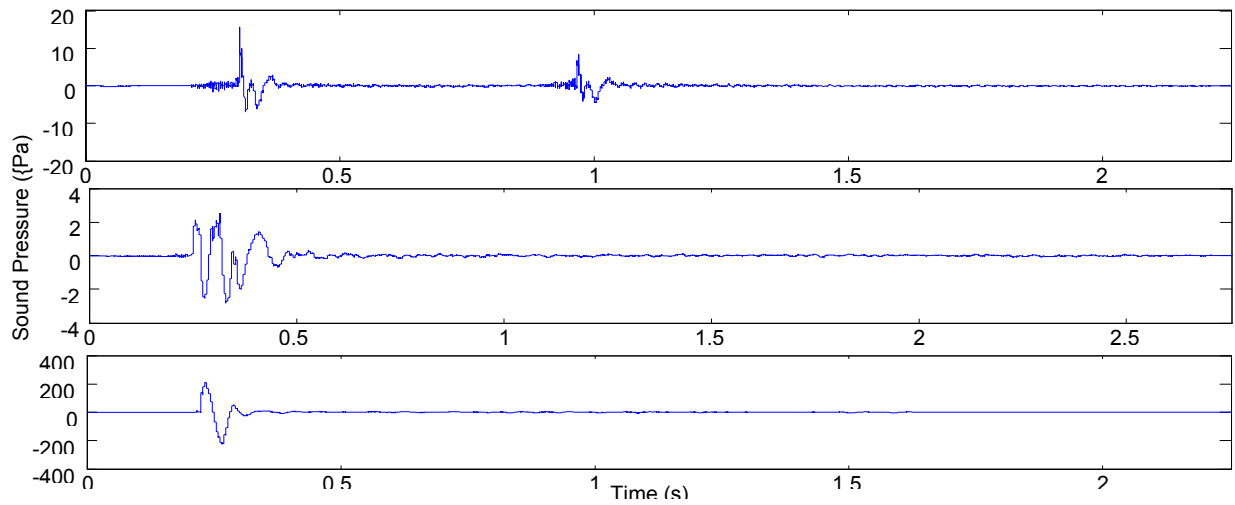
Figure 41. (top) 2 81mm detonations at 2km, (middle) 155mm high explosive round detonation at 6 km, (bottom) 3 Bangalore torpedoes at 2.5 km
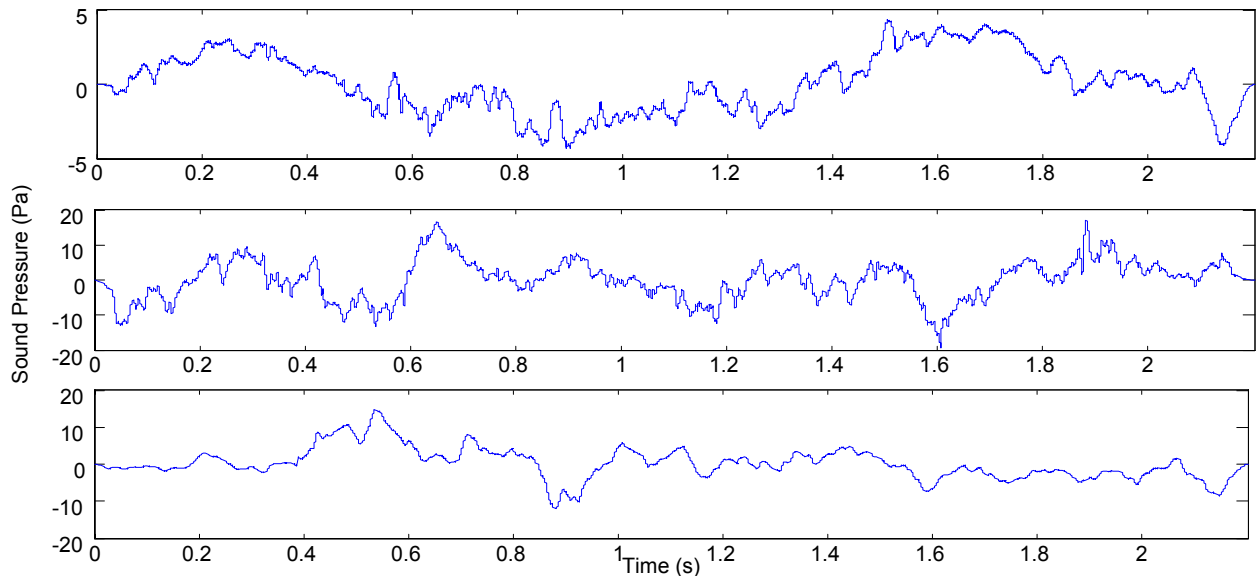

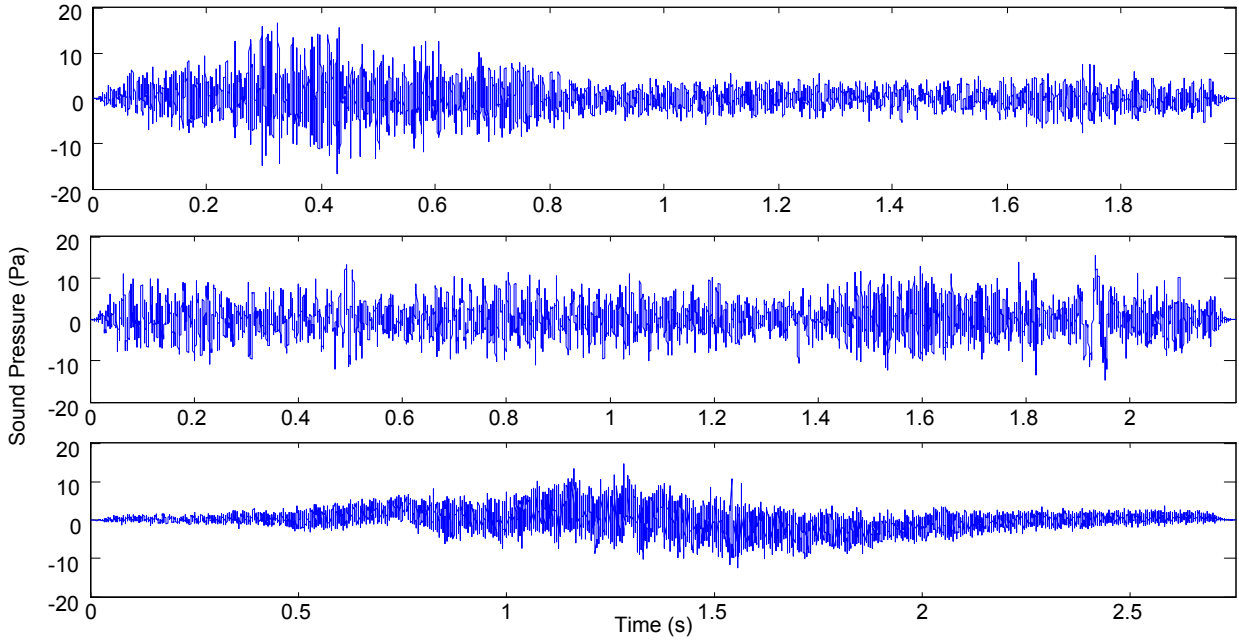**Figure 42.** Three recordings of wind noise

Figure 43. (top) F-16 flyover, (middle) F-16 flyover, (bottom) A-10 flyover

9.2 DATA PRE-PROCESSING

As previously mentioned, the data were originally sampled at 10kHz to verify that no features within the higher frequencies were being passed over. For this step, the waveforms were first decimated by a factor of 10 to simplify processing while retaining the shape of the waveform. The new sampling frequency was chosen since the current noise monitoring hardware samples at 1kHz. Should the technique work, it could be implemented on exsisting hardware with just a software upgrade. When a waveform is decimated from 10kHz to 1kHz the Nyquist frequency is moved from 5kHz to 0.5kHz. To verify the assumption that most of the signal features would be retained in the decimation process, the amount of signal energy retained after decimation was estimated using equation 9.1.

91

$$E = \frac{\sum_{w=1}^{M} PSD(w_i)}{\sum_{w=1}^{N} PSD(w_i)} \qquad (9.1)$$

In equation 9.1 E is the percent of the signal energy retained after decimation, PSD denotes the discrete power spectral density, M is the number of spectral bins from 0 to 500 Hz, and N is the number of spectral bins from 0 to 5000Hz. It was found that decimating recordings of wind noise and military impulse noise caused an average estimated loss of 0.1% of the total signal energy and decimation of recordings of aircraft noise caused an average estimated loss of 20-35% of signal energy. While there was considerably more energy lost in the decimation of waveforms of aircraft noise, the shapes of the waveforms still provided sufficient information. Figure 44 shows the un-decimated and decimated waveforms of a typical recording of military impulse noise. The decimated and un-decimated plots are virtually indistinguishable. Thus, there is no perceptible difference in waveforms after executing this operation.
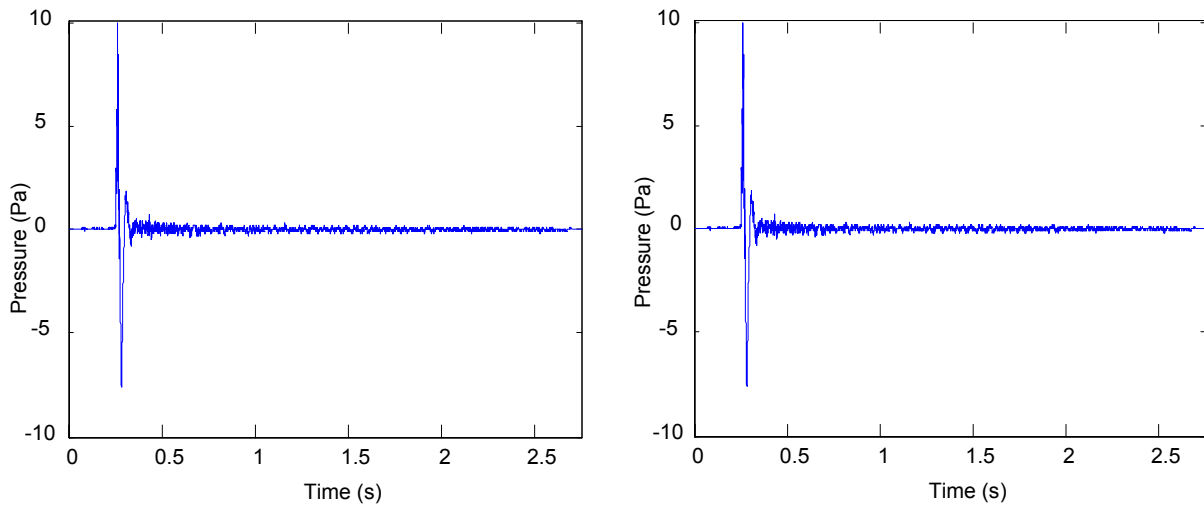

Figure 44. (Left) Recording of 155mm Howitzer at 1km sampled at 10kHz, (right) Same waveform after being decimated to 1 kHz

Temporal processing of the data set involves processing the input waveforms as an image. Although the input waveforms must exceed some set peak level to trigger the system, the input

waveforms have arbitrary amplitudes. The next step is to normalize the image. This is done by

finding the point in the waveform with maximum magnitude of sound pressure level and

dividing values of the rest of the points in the waveform by this magnitude. This yields an image

with a possible maximum value of 1 and possible minimum value of −1. The classifier will then

have the ability to generalize waveforms of arbitrary peak level values. The final step is to

discretize the normalized waveform into 100 bin grid placed over the waveform. The bin grid is

10 by 10 in structure and has equal widths in the horizontal and vertical directions (height of

each bin is the scaled value of 0.2 and the width is 0.2 seconds). The feature set to be input to

the artificial neural network is the number of data points that have mapped to each bin for a

particular waveform. Figure 45 shows the computation of the feature set for a typical recording

of military impulse noise.
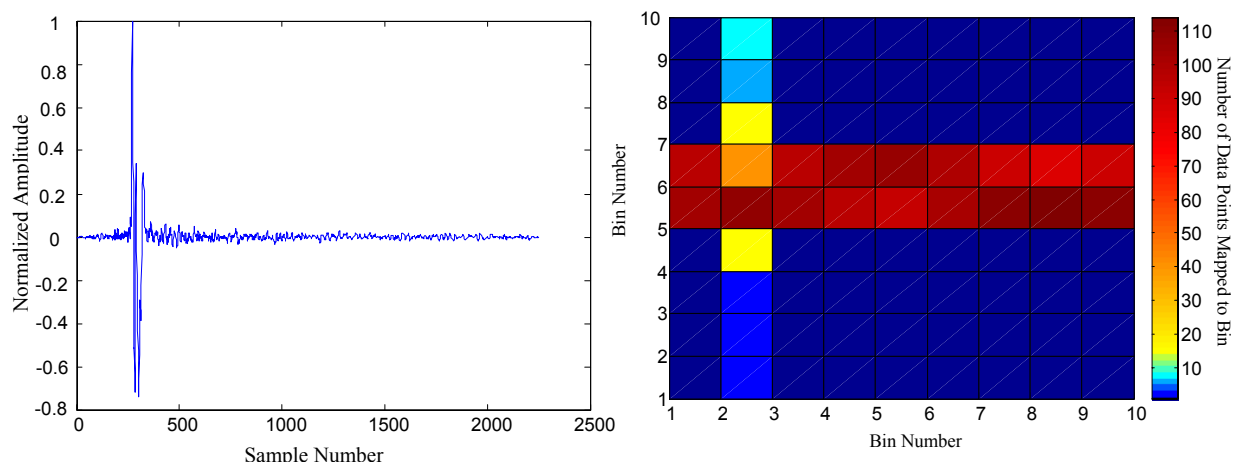
Figure 45. Computation of feature set for 81mm mortar recording

To control the number of inputs to the neural network structure and provide consistency across

the slightly varying lengths of the recordings, only the first 2 seconds of each recording were

considered for processing. Most considered events included a 0.25 second pretrigger thus, the

effective lengths of the processed recordings were 1.75 seconds. This time length scale is

deemed sufficient since most all military impulse events are completed well within 2 seconds

(Schomer, et. al., 2000). There is a possibility that the ends of some extremely large events could be missed by this method but, no such events existed in the library of recordings that were collected and the shape of the event that occurs within the set time window may already be sufficient to make a correct classification. In future efforts the time window may be extended.

9.3 ARTIFICIAL NEURAL NETWORK PROCESSING

This classification problem is most likened to a static image processing type of problem, such as the printed character recognition problem (Burr, 1986, Jagota, 1990). This problem has been addressed in a variety of methods with a variety of different networks. In the case of this problem, the type of network used is the feed-forward multi-layer perceptron trained with the back-propagation algorithm (Abe, 2001, Haykin, 1999, Chong, 2001). This network is the most simple type of network and is deemed to be a good starting point for the investigation. Networks constructed of sigmoidal and saturating linear activation functions are also investigated (Abe, 2001). The reasoning behind the use of saturating linear neurons was that if a successful network could be constructed with sigmoidal neurons then, it may be possible to construct a network with saturating linear neurons. In an actual DSP implementation of this algorithm, saturating linear neurons will prove to be more computationally efficient than sigmoidal neurons because the saturating linear neuron network will only require the multiplication and addition operations and the artificial realization of the sigmoid neuron network will require more complicated operations such as the estimation of exponentials.

Both networks consisted of 100 inputs, 2 hidden layers of 100 neurons each, and 1 output neuron. The training, validation, and testing data was formulated in the same fashion as the metric-based ANN study. The targets for the network consisted of values of 1 for records containing military impulse noise and values of −1 for records of non-impulse noise (wind and aircraft). In the case of both classifiers, the decision threshold is set to zero. In an actual implementation of this algorithm, the decision threshold could be increased to make the classifier less susceptible to false positives, with the cost of more missed events, or the decision could be decreased to make the classifier more sensitive to military impulse noise, with the cost of more false positive detections. Both networks were trained for 2000 epochs using the gradient descent with momentum and adaptive back-propagation algorithm (Chong, 2001). Table III shows the accuracy of the two configurations of artificial neural networks on the training, validation and testing datasets. Both networks performed extremely well (up to 100% accuracy) on all three datasets. This indicates that the proposed classifier is capable of making accurate generalization about the origin of an acoustic waveform and has also not been over-fit to the set of training data. It is also seen that the sigmoidal-neuron neural network and the saturating linear neuron neural network performed to approximately the same degree of accuracy, for the current noise library. Thus the use of a saturating linear neuron neural network provides a comparable and possibly more computationally-efficient solution to this classification problem.

Table III. Accuracy of artificial neural network classifiers with decision threshold set to zero

Neuron Type	Training Accuracy	Validation Accuracy	Testing Accuracy
Sigmoidal	100%	99.6%	99.6%
Saturating Linear	100%	98.8%	100%

Figures 46 through 48 show the activation of the output neuron of each network for each of the training, validation, and testing datasets for the sigmoidal neural network. Figures 49 through 51

show the same results for the saturating linear neural network. Training patterns containing military impulse noise are denoted by × and those training patterns of non-impulse noise are represented by O. The decision boundary for the classifier is denoted by the dashed line. In all cases a noticeable division of data types is seen. The greatest degree of separation of data types is seen in the training datasets. This is expected as the weights of the network are adjusted to minimize the error witnessed in this dataset specifically. However, the division of data types is still seen in the validation and testing datasets. This indicates that the networks are capable of making generalizations based on input patterns that are different than those used to train the networks. Figures 47, 48, and 50 contain one false negative detection each and figure 50 also contains two false positive detections. As before, the network's decision threshold could, in most case, be adjusted to make the networks more or less sensitive to the detection of military impulse noise depending on the cost of misclassification in either case.

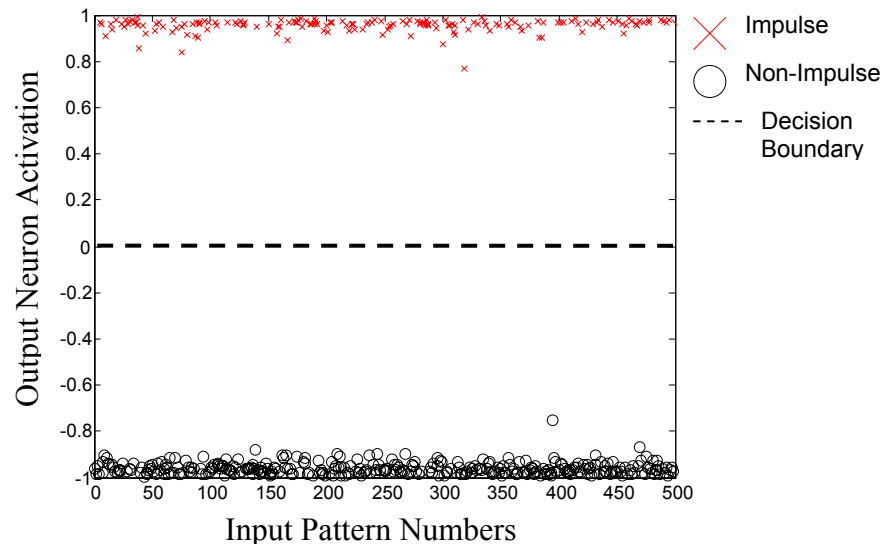**Figure 46.** Output training patterns presented to sigmoidal neuron ANN

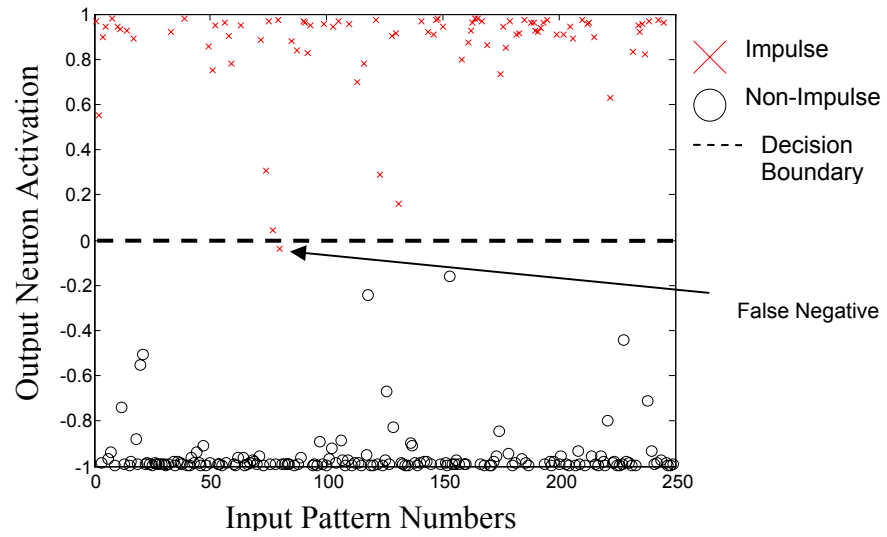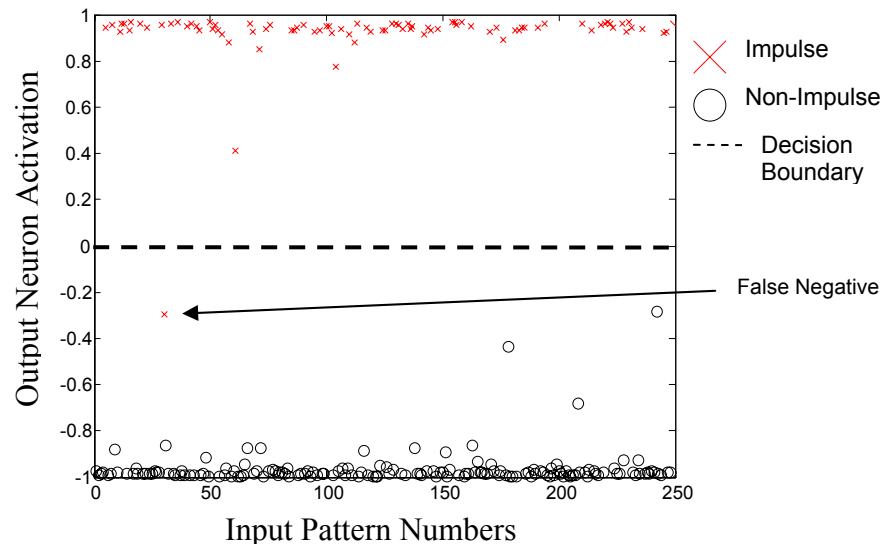**Figure 47.** Output validation patterns presented to sigmoidal neuron ANN

Figure 48. Output test patterns presented to sigmoidal neuron ANN
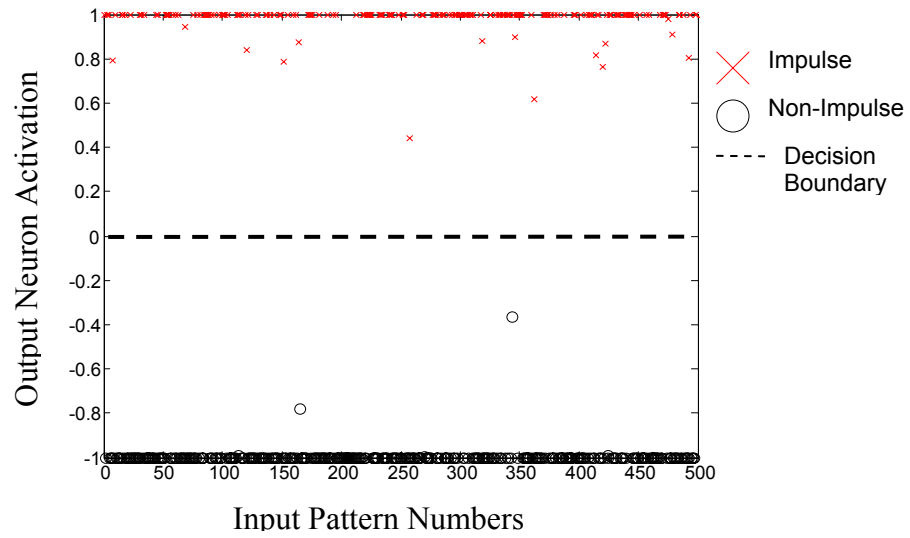
97

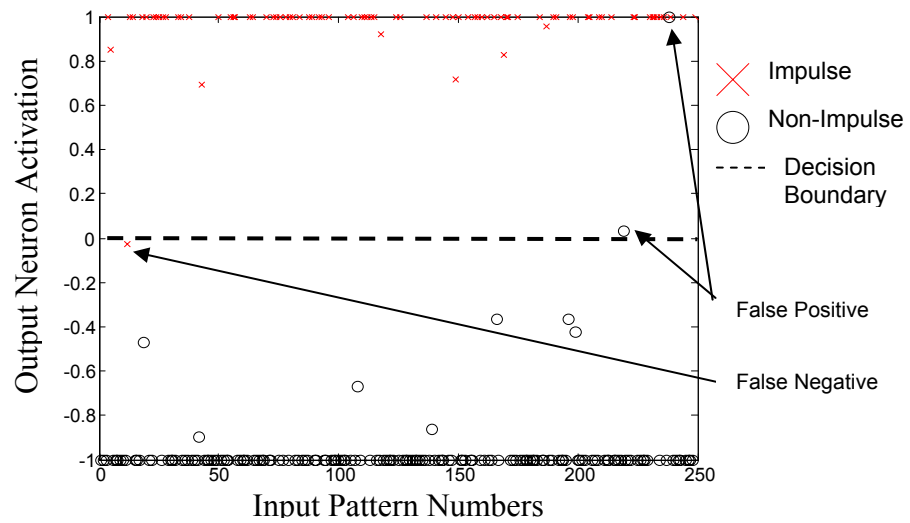**Figure 49.** Output training patterns presented to saturating linear neuron ANN

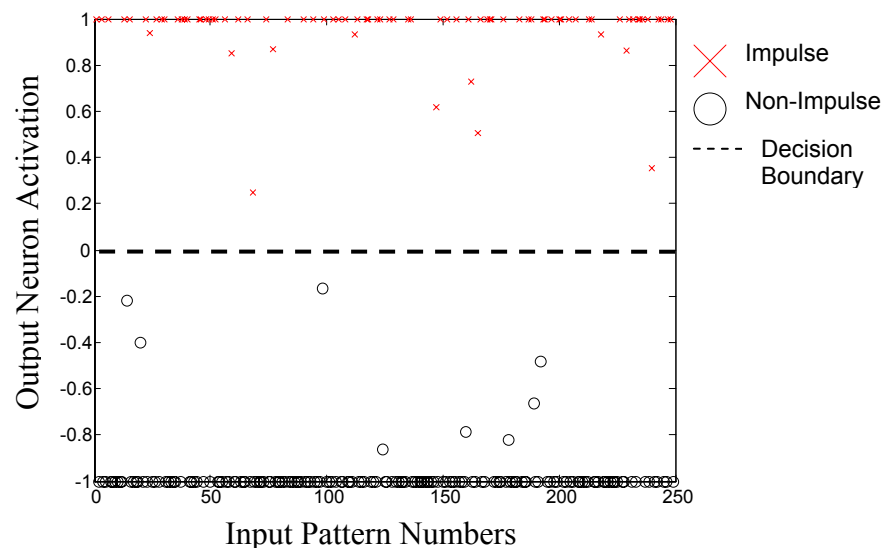**Figure 50.** Output validation patterns presented to saturating linear neuron ANN

Figure 51. Output test patterns presented to saturating linear neuron ANN

10.0 CONCLUSION

This thesis summarizes an effort to develop a more accurate and robust method of identifying high-amplitude military impulse noise from other noise sources. Three distinct noise classes were found: military impulse, wind, and aircraft noise, the latter two of which are non-impulsive. A library of military impulse noise and potential false-positive noise sources (wind and aircraft) was collected. The collected waveforms were then processed to extract the conventional signal metrics that were considered as signal features for the classifier. In addition, signal PSD was computed.   It was found that the time-domain metrics, kurtosis and crest factor, were most useful in identifying military impulse noise. Classification was first attempted with just kurtosis and crest factor as signal features to validate the assumption that kurtosis and crest factor would be good indicators of military impulse noise. To complement these metrics, two frequency-domain metrics, spectral slope (m) and weighted square error (WSE), were developed.

Finally, three ANN structures and a least squares classifier, each using the four signal metrics as inputs, were trained and evaluated. The ANN structures performed to 98.2-100% accuracy in identifying military impulse noise, while the least squares classifier performed to only 94.1% accuracy. This illustrates that the non-linear modeling capabilities of the ANN provide additional benefit over a linear classifier. When comparing the single metric classifiers to the kurtosis and crest factor classifier, an improvement of 4.3% to 24.3% in accuracy was achieved by using kurtosis and crest factor together as inputs.   The performances of the

classifiers were further improved, by between 3.8% and 8.5%, with the addition of the complementary frequency-domain custom metrics, *WSE* and *m*. The performance improvement illustrates that the two custom metrics are indeed useful in identifying military impulse noise. Utilizing the full set of metrics allowed the classifiers to achieve up to 100% accuracy in some cases.

In a continuation of previous work based on computed statistical and spectral metrics, a new artificial neural network based algorithm has been designed to operate from the temporal characteristics of collected waveforms. The artificial neural network is capable of making generalizations about the origin of a waveform based on the approximate shape of the waveform. In addition to evaluating the performance of a network constructed of sigmoidal activation function neurons, the performance of a possibly more computationally efficient saturating linear neuron neural network was compared to the former network. It was found that both temporal-based networks are capable of nearly producing 100% accuracy on training, validation, and test datasets. Although not presented, the classifiers were also able to identify impulse events with L_{pk} level down to 80 dB, where current monitors can only reliably detect events with $L_{pk} > 119$ dBC, indicating a measurement threshold improvement of 40 dB. As indicated by the number of events occurring in each recording documented in Table I., this algorithm was also found to work well in classifying multiple events occurring in a short span of time and in the presence of background noise.

11.0 FUTURE WORK

Extension of this effort would involve investigating the actual implementation of these algorithms on DSP boards accompanied by presenting the recorded waveforms to these self-contained systems and evaluating the systems. Eventually, an actual live fire field test would also be necessary. Additional data collection would also help to ensure construction of robust classification algorithms. Although the current set of computed metrics seems to provide adequate classification accuracy at this juncture, the addition of metrics should not be ruled out. These may become particularly useful if specific types of ordinance are added to the noise library resulting in degraded accuracy using the current set of metrics.

BIBLIOGRAPHY

S. Abe. *Pattern Classification: Neuro-fuzzy Methods and Their Comparisons*, Springer-Verlag, London, England, 2001.

Anonymous. Office of Economic Adjustment, Office of Assistant Secretary of Defense, and Economic Security. "Joint Land Use Study." November 1993, Office of Economic Adjustment, DUSD(I&E), Suite 200, 400 Army Navy Drive, Arlington, VA 22202-2884, (703)604-6020.

ANSI S12.17-1996. Impulse sound propagation for environmental noise assessment. New York, NY: Acoustic Society of America.

ANSI S12.17-1996. Estimating airblast characteristics for single point explosions in air, with a guide to evaluation of atmospheric propagation and effects. New York, NY: Acoustic Society of America.

ANSI S12.7-1986 (R1993). Methods for measurement of impulse noise. New York, NY: Acoustic Society of America.

M. Arain. "Modeling and controller design of an EGR solenoid valve using neural networks." Ed.: J.G. Taylor. *Neural Networks and Their Applicatoins.* John Wiley and Sons, West Sussex, England, 1996.

J. Attias, A.Y. Duvdevany, I. Reshef-Haran, Michal Zilberg, and Nageris Beni. (2004) Military noise induced hearing loss. In: Handbook of effects of noise on man. Luxon, L (Ed), London.

Jonathan W. Benson. "A real-time blast noise detection and wind rejection system." *Noise Control Engineering Journal*, **44** (6), November-December 1996, p. 306-314.

Robert D. Blevins (2001). *Flow-Induced Vibration*, Krieger Publishing Co., Melbourne, FL, 2/e reprint, 2001.

A. Bryson and Y. Ho. *Applied Optimal Control*, Blaisedale. (Revised Printing, 1975, Hemisphere Publishing, Washington, D.C.).

Brian Bucci and Jeffrey Vipperman. "Artificial neural network military impulse noise classifier." IMECE2006-14065, Proceedings of IMECE 06: 2006 ASME International Mechanical Engineering Congress, November 5-10, 2006, Chicago, Illinois.

Brian Bucci and Jeffrey Vipperman. "Performance of artificial neural network classifiers to identify military impulse noise." *Journal of the Acoustic Society of America*, submitted June 2007, (in press).

Brian Bucci and Jeffrey Vipperman. "Bayesian military impulse noise classifier." IMECE2007-41700, Proceedings of IMECE 07: 2007 ASME International Mechanical Engineering Congress, November 11-15, 2007, Seattle, Washington.

Brian Bucci and Jeffrey Vipperman. "Artificial neural network based temporal processing of waveforms to detect military impulse noise." Proceedings of Noise Con 07: Institute of Noise Control Engineering Conference, October 22-24, 2007, Reno, Nevada.

David Burr. "A neural network digit recognizer." IEEE Conference Procedings, Atlanta, GA, Oct. 1986, pp 1621-1625.

Randolph H. Cabell, Chris R. Fuller, and Walter F. O'Brien. "Identification of helicopter noise using a neural network." *AIAA Journal*, **30**(3), March 1992. p. 624-630.

R. H. Cabell, C. R. Fuller, and W. F. O'Brien. "Neural network modeling of oscillatory loads and fatigue damage estimation of helicopter components." *Journal of Sound and Vibration*, **209**(2), 1998, p. 329-342.

Igor Chunchuzov, Sergey Kulichov, Alexander Otrezov, and Vitaly Perepelkin. "Acoustic pulse propagation through a fluctuating stably stratified atmospheric boundary layer." *J. Acoustic Society of America*, **117** (4), April 2005, p. 1868-1879.

C. Cortes and V. Vapnik. "Support-Vector Networks." Machine Learning, **20**(3), September 1995, p.273-297.

E. Danicki. "The shock wave-based acoustic sniper localization." *Nonlinear Analysis*, **65**, 2006, p. 956-962.

J. Dron, F. Bolaers, and I. Rasolofondraibe. "Improvement of the sensitivity of the scalar indicators (crest factor, kurtosis) using a de-noising method by spectral subtraction: application to the detection of defects in ball bearings." *Journal of Sound and Vibration*, 270, 2004, p. 61-73.

R. Eberhart and R. Dobbins. "Early neural network development history: The age of Camelot." IEEE Engineering in Medicine and Biology, 9, 1990, p. 15-18.

S. Grossberg and G. Bradski. "A self-organizing architecture for invariant 3-d object learning and recognition from multiple 2-d views." Eds.: L. Jain and V. Vemuri. *Industrial applications of neural networks.* CRC Press, Boca Raton, Florida, 1999.

Simon Haykin. *Neural Networks: A Comprehensive Foundation*, Prentice Hall, Upper Saddle River, New Jersey, 1999.

Donald Hebb. *The Organization of Behavior*, Wiley: New York, New York, 1949.

ISO (1996-1). Acoustic-Description, measurement and assessment of environmental noise-Part 1: Basic quantities and procedures. ISO copyright office, Case postale 56, CH-1211 Geneva 20, ISO 1996-1:2003(E).

ISO (1996). Acoustics-Description and measurement of environmental noise-Part 1: Basic quantities and procedures. UDC 534.61 Ref. No. ISO 1996/1-1982(E)

Arun Jagota. "Applying a Hopfield-style network to degraded text recognition." International Joint Confernce on Neural Networks, San Diego, CA, June 1990, pp. 27-32.

Ehud Karnin. "A simple procedure for prunig back-propagation trained neural networks." *IEEE Transactions on Neural Networks*. **1**(2), June 1990, p. 239-243.

Bahram G. Kermani, Susan S. Schiffman, and H. Troy Nagle. (2005). "Performance of the Levenberg-Marquardt neural network training method in electronic nose applications." *Sensors and Actuators, B: Chemical*, **110**(1), September 2005, p. 13-22.

H. Kim and C. Singh. "Power system probabilistic security assessment using Bayes classifier." Electric Power Systems Research, Vol. 74, 2005, p. 157-165.

T. Kohonen. "Self-organized formulation of topologically correct feature maps." *Biological Cybernetics*, Vol. 43, p. 59-69.

T. Kohonen. "The self-organizing map." Procedings of the IEEE (1990), **78**(9), p. 1464-1480.

B. Lavine, C. Davidson, and D. Westover. "Spectral pattern recognition using self-organizing maps." *Journal of Chemical Information and Computer Sciences*, **44**(3), 2004, p. 1056-1064.

J. Liaw and T. Berger. "Dynamic synapse for signal processing in neural networks." U. S. Patent 6,363,369, June 11, 1998.

J. Liaw and T. Berger. "Dynamic synapse for signal processing in neural networks." U. S. Patent 6,643,627, March 26, 2002.

J. Lin and M. Zuo. "Gearbox fault diagnosis using adaptive wavelet filter." *Mechanical Systems and Signal Processing*, **17**(6), 2003, p. 1259-1269.

George A. Luz, Edward T. Nykaza, Catherine M. Stewart, Larry L. Pater. The role of sleep disturbance in predicting community response to the noise of heavy weapons. Final Report Work Unit CNN-T352, US Army Corps of Engineers Engineer Research and Development Center, ERDC/CERL TR-04-26, November 2004.

George A. Luz. "Suggested procedures for recording noise complaints at army installations." United States Army Environmental Hygiene Agency, Aberdeen Proving Ground, MD 21010-5422. HSE-08/WP Technical Guide, April 1, 1980.

Michael Norton, Denis Karczub. *Fundamentals of Noise and Vibration Analysis for Engineers*, Cambridge University Press, New York, NY, 2nd ed. 2003.

M. Maroti, G. Simon, A. Ledeczi, and J. Sztipanovits. "Shooter localization in urban terrain." *Computer*. **37** (9), August 2004, p. 60-61.

Warren McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity." *The Bulletin of Mathematical Biophysics*, Vol. 5, 1943.

Marvin Minsky and Seymour Papert. *Perceptrons*, MIT Press 1969.

N. H. Parsons. "Performance of a neural network based transient classifier at monitoring an acoustic perimeter intruder detection system." IEEE Annual International Carnahan Conference on Security Technology, Proceedings, 1995, p 9-13.

Larry Pater, Pamela Woof, Diane Rhoads, Michael White. Getting started guide for the small arms range noise assessment model (SARNAM). US Army Corps of Engineers Construction Engineering Research Laboratory, CERL ADP Report 99/48, May 1999.

Larry Pater, Anthony Krempin. Development of a muffler for small arms range noise mitigation. US Army Corps of Engineers Construction Engineering Research Laboratory, USACERL Techincal Report 98/126, September 1998.

M. Rafiq, G. Bugman, and D. Easterbrook. "Neural network design for engineering applications." *Computers and Structures,* **79**, 2001, p. 1541-1552.

Frank Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain." *Psychological Review*, **65**(6), 1958, p. 386-408.

William A. Russell Jr., George A. Luz, *et al.*, (2001), "Environmental Noise Management: An Orientation Handbook for Army Facilities," U.S. Army Center for Health Promotion and Preventive Medicine (USACHPPM).

Safety Dynamics Inc. Company Information, This is available online at
http://www.safetydynamics.net/.

Paul Schomer and Keith Attenborough, "Basic results from full-scale tests at Fort Drum", *Noise Control Engineering Journal*, **53** (3), May-June 2005, p. 94-109.

P. Schomer, M. Bandy, J. Lamb, and H. Van Slooten. "Using fuzzy logic to validate blast noise monitor data." *Noise Control Engineering Journal*, **48**(6), November-December 2000, p. 193-205.

O. Simula, P. Vasara, J. Vesanto, and R. Helminen. "The self-organizing map in industry analysis." Eds.: L. Jain and V. Vemuri. *Industrial applications of neural networks.* CRC Press, Boca Raton, Florida, 1999.

D. Skapura. *Building Neural Networks*. AMC Press, New York, New York, 1996.

A. Smith. "Enhanced sound level Matlab code." University of Pittsburgh Department of Mechanical Engineering 1097 Project Report: 00001, 2007.

M. Smith. *Neural Networks for Statistical Modeling.* Van Nostrand Reinhold, New York, New York, 1993.

R. Stoughton. "Measurements of small-caliber ballistic shock waves in air." *Journal of the Acoustic Society of America,* **102**(2), August 1997 781-787.

A. Tsoi and A. Back. "Discrete time recurrent neural network architectures: A unifying review." *Neurocomputing*, 15, 1997, p. 183-223.

USACHPPM. "Large arm noise."
http://chppm-www.apgea.army.mil/dehe/morenoise/large_arms.aspx

Vladimir Vapnik. "Principles of risk minimization for learning theory." *Advances in Neural Information Processing Systems*, Vol. 4, p. 831-838.

C. Von der Malsburg. "Self-organization sensitive cells in the striate cortex." *Kybernetik*, **14**(85), 1973.

Bernard Widrow and Marian Hoff. "Adaptive switching circuits". In *IRE WESCON*, New York. Convention Record, p. 96-104

S. Young. "Robotic vehicle uses acoustic array for detection and localization in urban environments." Proceedings of SPIE-The International Society for Optical Engineering, 4364, 2001, p. 264-273.

R. Youssif and C. Purdy. "A multi-strategy signal pattern classifier." Midwest Symposium on Circuits and Systems, 3, 2002, p. III304-III307.