

AD-A247 029



2

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

DTIC
ELECTE
MAR 05 1992
S B D

PASSIVE SONAR TARGET RECOGNITION
USING A BACK-PROPAGATING NEURAL NETWORK

by

DAVID FRANKLIN MOORE

JUNE 1991

Thesis Advisor:

Murali Tummala

Approved for public release; distribution is unlimited

92-05013



REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b RESTRICTIVE MARKINGS		
2a SECURITY CLASSIFICATION AUTHORITY			3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited		
2b DECLASSIFICATION/DOWNGRADING SCHEDULE					
4 PERFORMING ORGANIZATION REPORT NUMBER(S)			5 MONITORING ORGANIZATION REPORT NUMBER(S)		
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School		6b OFFICE SYMBOL (If applicable) EC/Tu	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School		
6c ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000			7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000		
8a NAME OF FUNDING/SPONSORING ORGANIZATION		8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c ADDRESS (City, State, and ZIP Code)					
			10 SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO	PROJECT NO	TASK NO
11 TITLE (Include Security Classification) PASSIVE SONAR TARGET RECOGNITION USING A BACK-PROPAGATING NEURAL NETWORK					
12 PERSONAL AUTHOR(S) MOORE, David Franklin					
13a TYPE OF REPORT Master's Thesis		13b TIME COVERED FROM _____ TO _____		14 DATE OF REPORT (Year, Month, Day) 1991 June	
15 PAGE COUNT 78					
16 SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.					
17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Neural Networks, Back-propagation, Passive sonar target recognition, Sonar target modeling		
FIELD	GROUP	SUB-GROUP			
19 ABSTRACT (Continue on reverse if necessary and identify by block number) The prompt and accurate processing of sonar data is essential in undersea warfare. The ability to quickly detect and classify sonar targets is crucial to the performance and survivability of all navy surface ships and submarines. With the advent of neural network technology, new opportunities have arisen which could greatly enhance current sonar target recognition capabilities. The main objective of this research is to demonstrate the practical usage of neural networks in recognizing the acoustic signatures of passive sonar targets using simulated-at-sea conditions. We will review the theory behind neural networks, the problems associated with recognizing acoustic signals in an underwater environment, and we will make a detailed case study of a neural network's performance using test data generated from simulated sonar targets.					
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS				21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a NAME OF RESPONSIBLE INDIVIDUAL Murali Tummala				22b TELEPHONE (Include Area Code) (408) 646-2645	
				22c OFFICE SYMBOL EC/Tu	

Approved for public release; distribution is unlimited

Passive Sonar Target Recognition
Using A Back-propagating Neural Network

by

David Franklin Moore
Lieutenant, United States Navy
B.S., University of California, 1982
B.S., California State University, 1986

Submitted in partial fulfillment of the
requirements for the degree of

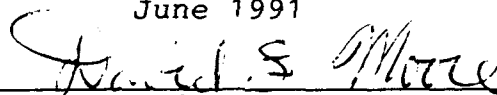
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

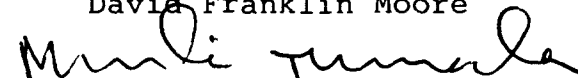
NAVAL POSTGRADUATE SCHOOL

June 1991

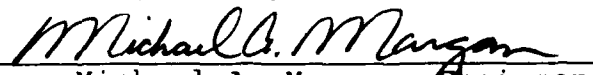
Author:


David Franklin Moore

Approved By:


Murali Tummala, Thesis Advisor


Chyan Yang, Second Reader


Michael A. Morgan, Chairman
Department of Electrical and
Computer Engineering

ABSTRACT

The prompt and accurate processing of sonar data is essential in undersea warfare. The ability to quickly detect and classify sonar targets is crucial to the performance and survivability of all navy surface ships and submarines. With the advent of neural network technology, new opportunities have arisen which could greatly enhance current sonar target recognition capabilities. The main objective of this research is to demonstrate the practical usage of neural networks in recognizing the acoustic signatures of passive sonar targets using simulated-at-sea conditions. We will review the theory behind neural networks, the problems associated with recognizing acoustic signals in an underwater environment, and we will make a detailed case study of a neural network's performance using test data generated from simulated sonar targets.

Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

TABLE OF CONTENTS

I. INTRODUCTION	1
II. NEURAL NETWORK FUNDAMENTALS	5
A. THE NEURAL NETWORK MODEL	5
1. The Goals of the Neural Network Model . . .	5
2. The Origin of the Neural Network Model . . .	6
3. Neurons - The Neural Network's Basic Processing Elements	7
B. THE GENERALIZED DELTA RULE	10
C. SUMMARY OF THE BACK-PROPAGATION ALGORITHM . . .	13
III. THE SONAR TARGET MODEL	15
A. THE PASSIVE SONAR EQUATION	15
B. MODELING THE TARGET SOURCE	18
1. CAVITATION NOISE	18
2. SCREW BLADE NOISE	20
3. OTHER RADIATED NOISE	22
C. TRANSMISSION LOSSES	26
D. AMBIENT NOISE LEVEL	26
IV. NEURAL NETWORK TRAINING AND TESTING	29
A. NEURAL NETWORK DESIGN CONSIDERATIONS	29

B. TESTING THE FIRST NEURAL NETWORK ARCHITECTURE .	35
C. REVISING THE NEURAL NETWORK ARCHITECTURE . . .	41
V. CONCLUSIONS	45
REFERENCES	48
APPENDIX A: TEST DATA	50
APPENDIX B: MATLAB PROGRAM	62
INITIAL DISTRIBUTION LIST	71

I. INTRODUCTION

Improving passive sonar target recognition continues to be an important problem for the U.S. Navy. Because of the swiftness and complexity of modern warfare, the prompt and accurate detection and classification of sonar targets is vital to the success and survivability of all modern navy ships. Any failure or delay in identifying and pursuing an opponent in wartime invites a surprise attack upon ones own ship. In the past, various methods were used to detect and classify sonar targets. The earliest method involved a single sonarman listening to an acoustic signal over a simple hull mounted hydrophone which he could manually steer to search for nearby targets. In this method, the expertise of the sonar operator determined the quality and accuracy of the target detection and classification. Unfortunately, human beings cannot discern many important characteristics of underwater signals because many key signals exist well below the normal 1 to 5 Khz sensitivity range of human ears [Ref. 1]. Since the analysis of low frequency signals is essential to passive sonar target recognition, the human operator is inherently ill-equipped to handle the task. Moreover, the faintest signals can be masked by the ship's own noise which can further hamper signal detection. Although human senses are limited, one thing the human operator does excel at is at

picking out patterns and discriminating between ambient sea noises and man-made noises. This important trait is also shared by neural networks.

Beginning in the early 1950's, as signal processing techniques and digital computers improved, better methods of sonar target detection were devised using towed arrays, newly advanced hull arrays, digital beamforming, and computer enhanced methods for spectral analysis [Ref. 2]. Most contemporary sonar systems are improved versions of these earlier systems. With the advent of neural network technology, it now becomes possible to combine human-like capabilities of pattern recognition with the power and speed of modern signal processing. Neural networks can lend three important qualities to the realm of acoustic signal processing. First, neural networks are excellent at forming generalizations about a given set of objects [Ref. 3]. In our research, such a set of objects will be comprised of the acoustic signatures of passive sonar targets. With such a set of acoustic signals, generalization means that a neural network can learn and recall the key traits of the signal even in the presence of ambient-noise and other signal distortions. Thus, a sonar target signal contaminated by undersea noise may still be recognizable as long as a few identifying features remain. A second advantage of neural networks is in their ability to discriminate between objects or signals which have very complex interrelationships [Ref. 3]. In sonar target

recognition, this means that any key features or relationships attributed to the acoustic signatures that may be non-linear and not easily solved by other conventional statistical techniques may be better solved by the application of a neural network which can solve such problems more quickly and efficiently. Lastly, and probably the most important advantage of neural networks lies in their ability to learn and extract key information from a given set of training examples [Ref.3]. Learning gives the neural network the ability to adapt to changes in its environment and build upon the information and associations already stored in its memory. Therefore, because the undersea environment is so dynamic and complex, an intelligent and adaptive sonar system regulated by a neural network would be very suitable for detecting and recognizing sonar targets.

The idea of using neural networks to solve sonar target recognition problems is not entirely new. In 1988, two neural network researchers, R. Paul Gorman and Terrence Sejnowski, successfully used a neural network to identify the echoes of undersea objects using active sonar [Ref. 4]. Thus inspired by the work of Gorman and Sejnowski, the goal of our research was to design an artificial neural network which could identify the acoustic signature of a given passive sonar target. Our goal was accomplished in three steps. First, we created a computer program that could realistically simulate a passive sonar target signal. Secondly, we trained a neural network to

memorize a simulated sonar target using a back-propagation learning algorithm. And finally, in our third step, we tested the neural network's memory to see if it could recognize the sonar target using two test scenarios generated by the sonar target model.

By following the above outlined steps, the thesis was organized in the following manner. Chapter two discusses the origins and special characteristics of neural networks and the benefits derived from their application. Also discussed is the nature of the back-propagation learning algorithm and how it may be used to train our neural network. Chapter three discusses the development of the sonar target model and some of its properties and assumptions. Chapter four describes the results of training and testing a neural network as applied to passive sonar target recognition. We also examined the advantages and disadvantages of two particular neural network architectures. Lastly, Chapter five discusses the conclusions of our results and considers the future applications of neural networks in passive sonar target recognition.

II. NEURAL NETWORK FUNDAMENTALS

A. THE NEURAL NETWORK MODEL

1. The Goals of the Neural Network Model

The primary goal of our investigation into passive sonar target recognition is to design a neural network that has the ability to:

1. Learn the acoustic signature of a passive sonar target.
2. Draw generalizations about certain passive sonar signals thus allowing the system to see through noise and other disturbances. In applying this property to sonar targets, this means the ability to identify a target even when it undergoes changes in range, speed, and aspect.
3. Identify sonar targets even when faced with complex interrelationships such as changing environments, multiple targets, and non-linearities in the signal. This includes learning the ability to distinguish among multiple targets and to interpret signals corrupted by ambient-noise or high regional shipping density.

Many recent successes in neural network applications have illustrated some of these special properties. Some applications have shown that neural networks can learn special kinship patterns among families. After memorizing the genealogies of several related families, the neural network demonstrated the ability to draw accurate conclusions about many complex family relationships [Ref. 5]. In addition, neural networks have shown success in recognizing targets that are illuminated by active sonar echoes [Ref. 4], and some

have demonstrated the ability to recognize samples of handwritten characters [Ref. 5: p. 136]. Each exercise reveals that neural networks can learn and can correctly generalize about complicated patterns and distinguish among objects even when inputs to the network are contaminated by noise or missing data. In our investigation, the neural network will be presented first with a set of simulated signals representing the acoustic signature of a given target. After the neural network is trained on these signals, it will then be tested to see if it can remember and thus recognize the target's acoustic signature even under noisy conditions.

2. The Origin of the Neural Network Model

The study of neural networks has been around since the 1950's and has coexisted with the development of the digital computer. Many early manifestations of neural networks, such as the Mark I Perceptron developed in 1957 by Frank Rosenblatt, gained much interest and notoriety. However, much of the neural network research came to an abrupt halt in 1969 with the publication of the book Perceptrons by Marvin Minsky and Seymour Papert who proved that perceptrons could not implement the simple EXCLUSIVE OR logic operation [Ref. 6]. At the time, most researchers abandoned neural networks and pursued the development of the very popular and increasingly powerful digital computer and related techniques. Little work was done on neural networks until the late 1970's and early

1980's when neural networks made a comeback with the help of John Hopfield and others who proved both theoretically and experimentally that massively parallel machines like neural networks could be designed to make intelligent decisions and perform useful tasks. Moreover, they showed that the neural network's human-like data processing style was beyond the capabilities of conventional digital computers. In 1986, David Rumelhart and James McClelland demonstrated a powerful learning algorithm known as the back-propagation rule which inserts a "hidden" layer of neurons into Rosenblatt's perceptron and thus corrects the deficiencies exposed by Minsky and Papert. The neural networks are now used in many practical applications. [Ref. 6]

3. Neurons - The Neural Network's Basic Processing Elements

Neural networks are parallel, distributed, information processing systems capable of learning and recalling given sets of data and recognizing complex associations among sets of related objects [Ref. 6]. Neural networks, also sometimes called connection machines, process data analogously to the way in which human brains process data. In the brain, neurons behave like tiny microprocessing elements which receive and combine input signals dispatched by other neurons (Figure 2.1). Each nerve cell receives input signals through input structures called dendrites. The arriving input signals are

transformed into charged particles which are accumulated to form a single voltage. If the cumulative voltage reaches a cell's activation threshold, the cell fires producing an action potential which propagates down an output path called an axon. When this output signal reaches the axon's terminal end, it is converted into chemical energy in the form of a neurotransmitter which is injected into the junction or synapse joining the axon terminal of the activated neuron with the dendritic inputs of the targeted neuron. As the passing

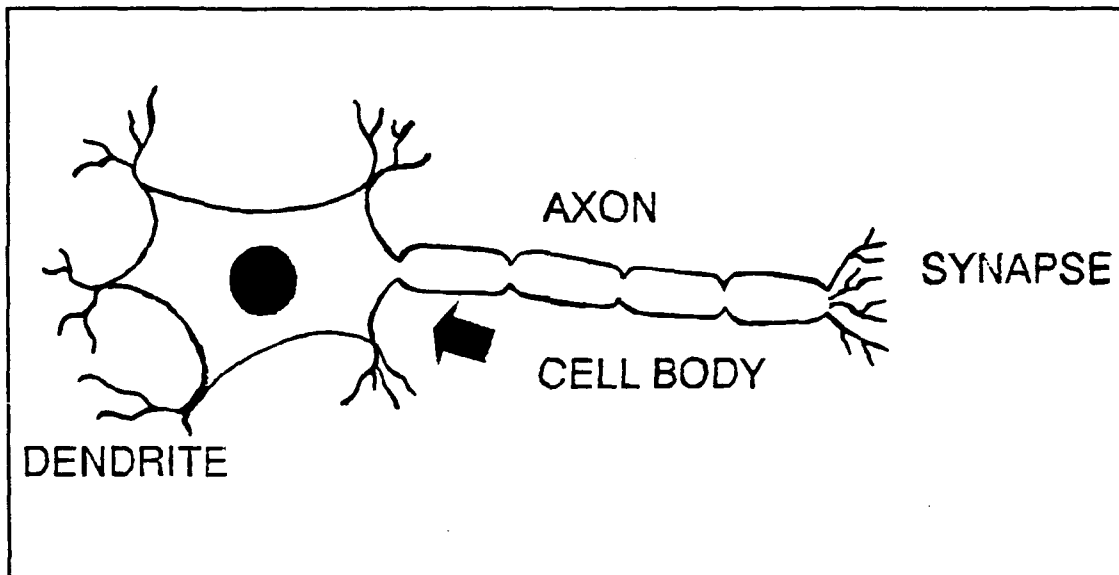


Figure 2.1: Diagram of a biological neuron.

neural signal travels between neurons, information is either being stored as occurs in training or it is being retrieved as occurs during recall. Consequently, the unique capabilities of a neural network are derived from these complex

interrelationships shown by the vast network of interconnected neurons. This complex neural architecture gives every neural network its special properties. Thus borrowing from the biological neuron, the neurons of artificial neural networks emulate the information processing behavior displayed by the human brain by interconnecting a large number of artificial neurons called processing elements (PE's). [Ref. 7]

Each neural processing element has four basic components (Figure 2.2).

1. Input Connections (dendrites) through which the neuron receives activation signals from other neurons [Ref. 8].
2. Summation Function which combines the input signals from many sources into a single activation signal [Ref. 8].
3. Threshold Function that converts the summation of input signals into an output activation signal [Ref. 8].
4. Output Connections (axons) represent the path followed by the output activation signal produced by the threshold function [Ref. 8].

Each connection that joins two processing elements is assigned a numerical weight which quantifies the strength of that neural connection. Each weight modulates the strength of the incoming input signals and thus helps to determine the contribution of that connection as it reaches the targeted processing element. When all the weighted input signals arrive at the targeted processing element, they are added together by a summing function. The resulting sum of weighted inputs is then passed to an awaiting threshold function. If

the combined level of activation exceeds the given threshold as defined by the threshold function, then the targeted processing element is activated and an output signal is sent to other neurons. Thus, unlike conventional computers which store data at discrete addresses, the information stored in a neural network resides in the weighted connections joining the processing elements. [Ref. 8]

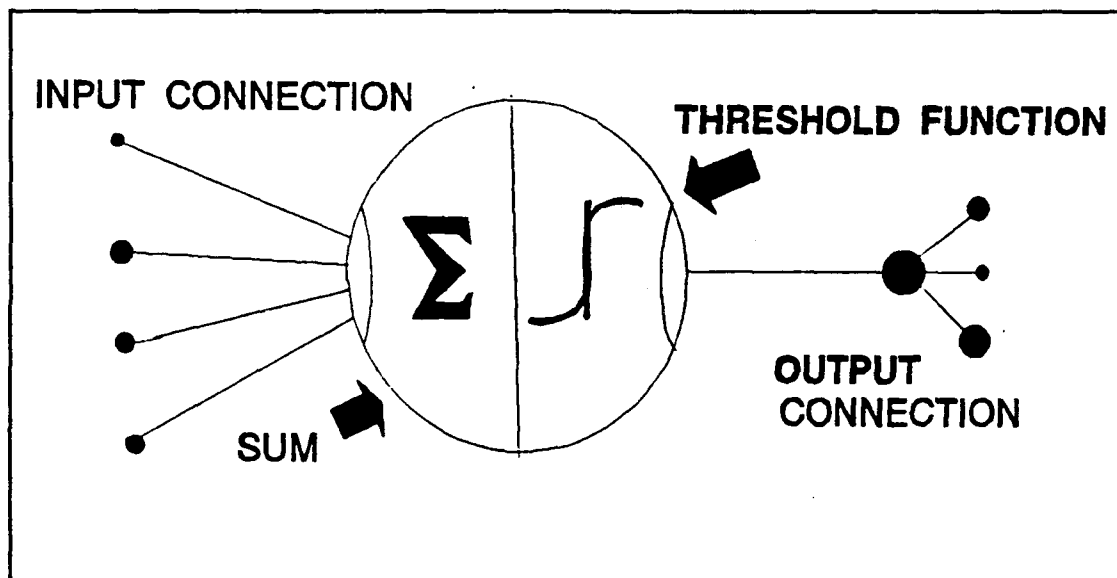


Figure 2.2: Diagram of an artificial neuron or processing element.

B. THE GENERALIZED DELTA RULE

Before a neural network is ready to be used, its connection weights must be predetermined through the process of training. In neural networks, there are two important types of learning. One is competitive learning in which the network

is presented with a series of input examples and the processing elements compete among themselves to establish a final equilibrium among their connection weights. This method requires fairly complex network architectures and learning algorithms in order to be implemented. The generalized delta-rule, however, follows another type of learning called supervised learning in which the network is trained using a series of input examples which are associated with a desired target output. In this scheme, the network's connection weights align themselves to produce the target output when fed the proper input signal. To implement the delta-rule, all connection weights are initialized using small random values. As training proceeds, the network is presented with training pairs, one representing an input pattern, and the other representing the desired target output. The training process continues until each input pattern generates its desired output response. During any given training event, the input nodes of the network are presented with an input pattern I_i which invokes an output response O_j from the network. Comparing the output response O_j to the desired target output T_j produces a delta. Following each training event, the connection weights are updated to reflect the discrepancy between the neural network's own generated output and the ideal target output [Ref. 8]. This quantity is expressed in

the following equation

$$\Delta w_{ij} = \eta (T_j - O_j) I_i \quad . \quad (2.1)$$

The constant η represents a momentum term that can accelerate the convergence of the network to its final connection weight values. With the delta-rule, the most active inputs in an input pattern cause the greatest weight modification. The delta-rule uses a gradient heuristic, meaning that connection weights tend to change in directions that maximize the change in an error term that sums the squares of output deltas [Ref. 8]. Thus the delta-rule produces a steepest descent algorithm that converges quickly in the beginning when the differences between the network generated output and target output are greatest but then converges more slowly as the generated output nears the targeted output. Although the delta-rule provides an effective training method, it has some limitations. First, the generalized delta-rule is most effective in networks using a "hidden" layer of processing elements. A hidden layer is a separate layer of processing elements positioned between the input and output layers. Hidden layers are used because single unit networks are unable to perform important operations like computing the EXCLUSIVE OR which was the main drawback of the perceptrons. One final drawback of the delta-rule is that it can sometimes be unstable. When using the delta-rule, the input parameters and learning momentum term must be well behaved, otherwise, the

algorithm can become unstable and convergence may never occur [Ref. 8]. To counter this limitation, input data and training data sometimes need to be modified or normalized in order to prevent instability from occurring. Finding the best momentum term is often a matter of trial and error.

C. SUMMARY OF THE BACK-PROPAGATION ALGORITHM

One of the direct benefits of the generalized delta-rule is the back-propagation algorithm which is basically a systematic implementation of the delta-rule. The back-propagation algorithm can be broken into five separate steps. Each pass through the five steps constitutes one iteration in the training cycle.

Step 1. Calculate the sum of products (N_j) of the connection weight matrix (W_{ji}) with its respective input vector (I_i). Where n equals the number of elements in the input vector during the t th training cycle. [Ref. 9]

$$N_j = \sum_{i=1}^n W_{ji}(t) I_i \quad (2.2)$$

Step 2. Calculate the output signal (O_j) by passing the sum of products (N_j) through the threshold function $f_j(N_j)$. The threshold function must abide by two rules. First, the output of the function must be nondecreasing and second the function must be differentiable. Two commonly used threshold functions are the sigmoid and the hyperbolic tangent functions. [Ref. 9]

$$O_j = f_j(N_j) \quad (2.3)$$

Step 3. Calculate the delta term (δ_j) by comparing the output signal (O_j) with the desired target output (T_j) and multiply that difference with the derivative of the threshold function. [Ref. 9]

$$\delta_j = (T_j - O_j) f'_j(N_j) \quad (2.4)$$

Step 4. This is the step where the back-propagation occurs. The new delta term which is based on the current output is sent back around to update the connection weight matrix $W_{ji}(t+1)$. The η variable represents the momentum term which can be increased or decreased in order to facilitate the training process. [Ref. 9]

$$W_{ji}(t+1) = W_{ji}(t) + \eta \delta_j I_i^T \quad (2.5)$$

Step 5. Return to step 1 and repeat the process until the connection weights can generate the desired output response when given the appropriate input vector [Ref. 9].

III. THE SONAR TARGET MODEL

A. THE PASSIVE SONAR EQUATION

Before we can properly design and train a neural network to recognize the acoustic signature of a given sonar target, a realistic sonar signal must be created that can simulate a sonar target as a function of range, speed and aspect, and can account for ambient undersea noises including man-made shipping noises. In order to design a proper target model we must follow the basic principles of the passive sonar equation [Ref. 2: p. 21] which can be expressed as

$$SL - TL = NL - DI + DT \quad . \quad (3.1)$$

Each term in the passive sonar equation is expressed in decibels. The first term, *SL*, is the source level of a given target representing the total amount of acoustic energy emitted from the target of interest. It includes such things as cavitation noise, screw blade noise, and a variety of narrow-band tonals representing a family of mechanically generated signals that radiate from auxiliary machinery, pumps, generators, and certain types of flow noises [Ref.2: pp.328-351].

The second term, *TL*, represents the transmission losses of a signal as it travels from the source location to the

receiving hydrophone. Transmission loss consists of several key components. The first component is spreading loss. As sound propagates through a medium, the acoustic intensity of the wave decreases as a function of the square of the distance between the source and the receiver. The second component of transmission loss is attenuation which is a function of frequency as well as range. In general, as frequency increases, the attenuation of an acoustic signal also increases. Therefore, one can expect higher frequencies to propagate shorter distances than lower frequencies. For purposes of the sonar target model discussed here, attention will be directed toward the lower end of the frequency spectrum where the effects of attenuation are not as strongly felt. The entire frequency range covered by our model will be from 1 Hz to 256 Hz. In addition to spreading and attenuation losses, other factors such as scattering losses also contribute to the overall transmission loss. Sound scattering occurs when sound waves come in contact with underwater barriers such as the undersea bottom, stratified thermal layers, or the air-sea interface at the surface. Sometimes even an obstructing school of fish can cause sound scattering. To eliminate the effects of scattering in the sonar target model, it can be assumed that all passive sonar signals are direct path which means that the acoustic signals travel directly from the sound source to the receiver without encountering any scattering obstacles [Ref.2: pp.99-285].

The third term in the passive sonar equation, NL , refers to the ambient-noise level present at all times under the ocean. Ambient-noise includes noise caused by the wind, rain, currents, and wave action. Much of the ambient-noise is measured in sea state. The higher the sea state level, the higher the ambient-noise present. In the sonar target model, the ambient-noise level will be expressed in terms of sea state. Each time the sea state increases by one, the ambient-noise level doubles. In addition to environmental ambient-noise, there are ocean noises attributable to shipping or biological organisms such as fish or shrimp. The sonar target model will include only the shipping density as a factor in calculating the total ambient-noise [Ref.2: pp.202-223].

The last two factors expressed in the passive sonar equation, DI and DT , are both dependent on the transducer receiving the acoustic signal. DI represents the directivity index of the transducer. The directivity index depends on the physical arrangement of the transducer array which when oriented in specific direction can exhibit a large gain or sensitivity for a given signal. This type of directivity is often used in assessing the bearing to a sonar target [Ref.2: p.22]. For purposes of this model, we can assume that our transducer is omni-directional and thus exhibits a gain of one in all directions. Thus, DI will be computed as 0 dB in the passive sonar equation. The last factor, DT , is the detection threshold of the receiving transducer and is

dependent on the overall sensitivity of a given transducer. In the sonar target model a detection threshold of 0 dB SNR will be selected. This means that all signals received from the target source that are above 0 dB SNR will be considered as a target present and all other signals below 0 dB SNR will be considered as noise, in which case, only ambient-noise will be detected. After revising the passive sonar equation for the sonar target model, we obtain

$$SL - TL - NL > DT \quad (3.2)$$

which reflects the fact that for all signals received from the sonar target above 0 dB, our model will detect the target.

B. MODELING THE TARGET SOURCE

1. CAVITATION NOISE

The target source for our model will exhibit three types of signals: cavitation noise, blade rate noise, and narrow band tonals. Cavitation noise covers a broad spectrum with its center frequency located around 100 Hz (Figure 3.1) [Ref. 2: p. 334-339]. Cavitation noise is generated by a ship's propeller. As the propeller blade churns the water, a trail of bubbles is generated. As these bubbles form and then collapse, the resulting noise created is called cavitation. Cavitation noise can vary with respect to a target's speed and

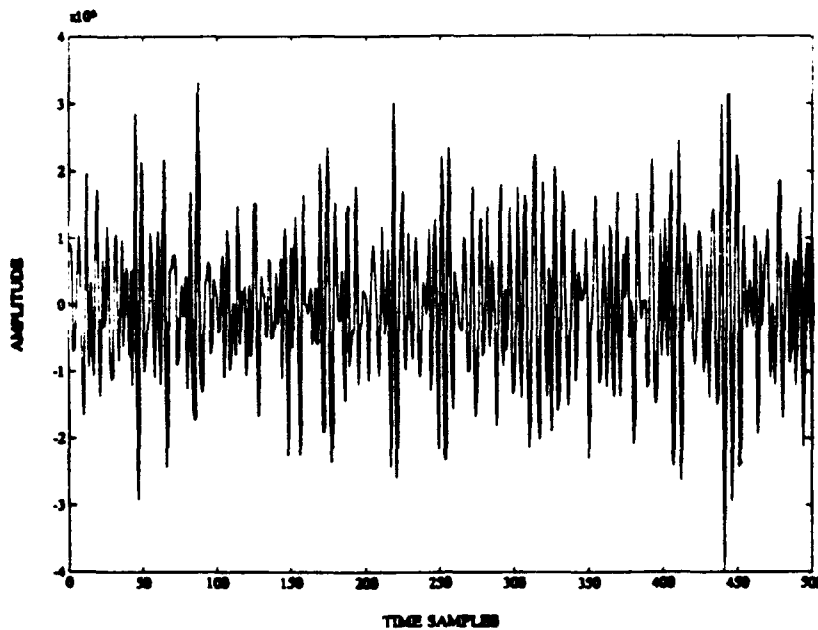


Figure 3.1 (a): Plot of the actual cavitation noise signal as produced by the sonar target. The amplitude is referenced to $1 \mu\text{Pa}$.

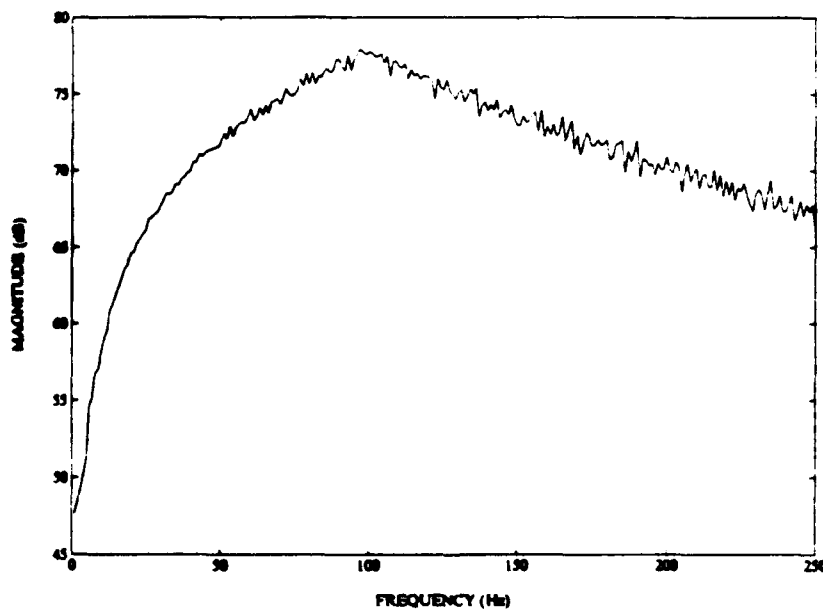


Figure 3.1 (b): Plot of the cavitation noise spectrum as produced by the sonar target model. Notice that the center frequency of this spectrum is approximately 100 Hz.

depth. For the purposes of this model, we can assume that the target's depth remains constant at the surface. Therefore, cavitation noise will vary only in response to changes in target speed. From the standpoint of target detection and classification, this means that as the target ship's speed increases, the cavitation noise created will tend to mask fainter tonals making the target signal more difficult to identify. However, since neural networks are capable of recognizing important features from noisy and distorted signals, our neural network should be well equipped to see through cavitation-type noise. [Ref.2: pp.334-339]

2. SCREW BLADE NOISE

Screw blade noise is the second component of the target source's acoustic signature. Screw blade or propeller noise depends on several factors including the target's speed, the number of blades on each screw, and the turns per knot (TPK) of the screw itself. As a ship's propeller churns the water, an acoustic signal or "blade rate" is generated [Ref. 2:p. 348]. The propeller noise is an amplitude-modulated signal and contains discrete spectral blade rate components that can be detectable in the low frequency end of the acoustic noise spectrum (Figure 3.2). The frequencies of the blade rate can be determined by the formula

$$f_n = nSTN_{BLADES} \quad (3.3)$$

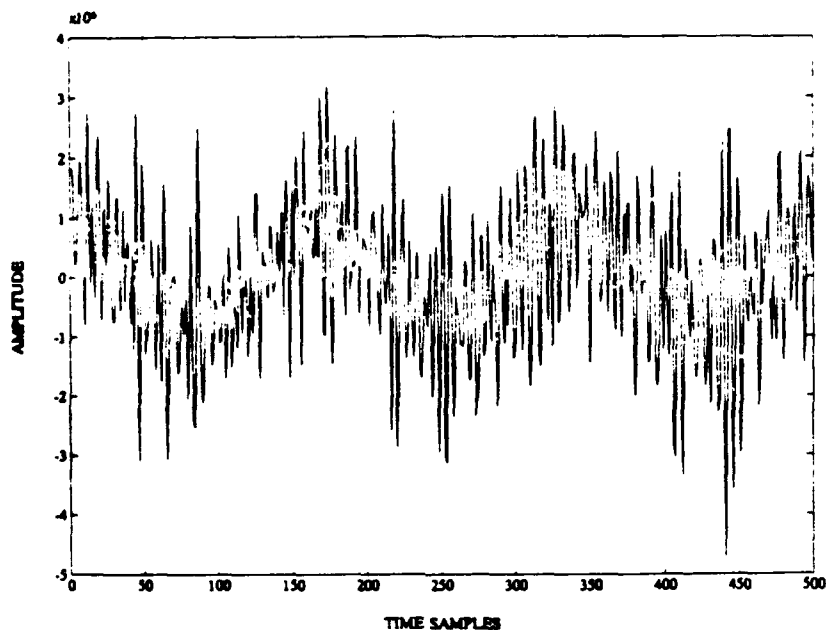


Figure 3.2 (a): Plot of the actual cavitation noise signal with screw blade noise added. This signal was produced by the sonar target model. The signal's amplitude is referenced to 1 μPa .

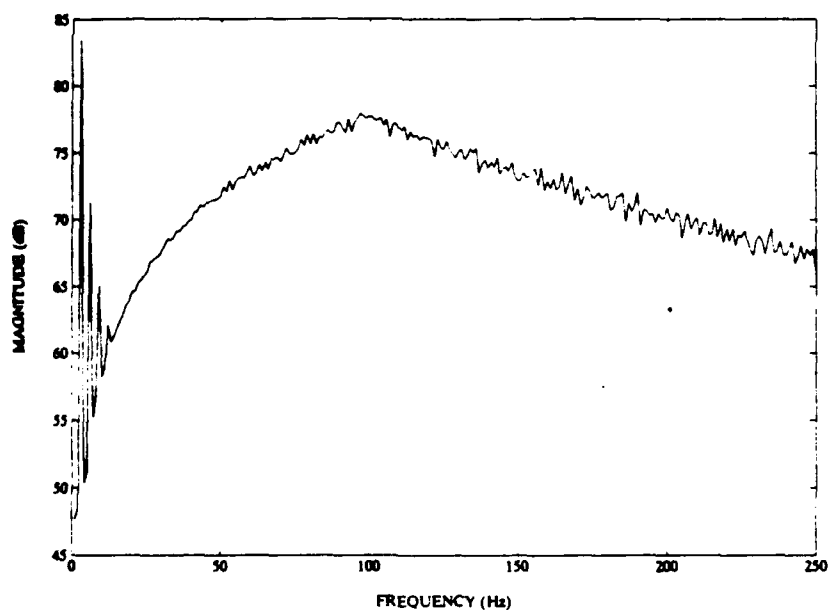


Figure 3.2 (b): Plot of the cavitation noise spectrum with screw blade noise added. This signal is from the sonar target model. Notice the "blade rate" lines below 25 Hz.

where n is the n th harmonic of the fundamental blade rate frequency f_0 , S is the ship's speed in knots, T represents the number of propeller turns per knot (TPK), and N_{BLADES} is the number of blades per propeller. The blade rate spectra are best observed at low frequencies below 50 hz. Higher frequency harmonics tend to be weaker and they attenuate faster. Moreover, weaker signals tend to be masked by louder signals such as cavitation and ambient-noise. Once again, the neural network's ability to generalize and identify objects even when such information is missing makes them useful in identifying a target with or without the presence of screw blade noise. [Ref.2: p.348]

3. OTHER RADIATED NOISE

The third component of the target source signal involves the remaining forms of radiated noise emitted by the target in the form of narrow band tonals. Tonals are normally seen as very narrow spectral lines occurring in the ship's acoustic signature spectrum. Tonals are created from a variety of sources including auxiliary machinery, flow noise, reduction gears, propellers, and other mechanical sources. The presence of tonals can act as a ship's fingerprint. Figure 3.3 shows a target's signal spectrum where tonals appear at 53, 56 and 203 Hz. If a ship has a bad bearing in its bilge pump, for example, the tonal radiating from that bad bearing can help single out that ship from others (See Table 3.1).

Another factor affecting the detection of tonals is determined by the target ship's aspect. Each tonal can have a certain directivity (Figure 3.4). The strongest tonals travel along the most direct path from the mechanical noise source to the ship-sea interface where it is emitted into the surrounding undersea environment. Longer propagation paths can often absorb and muffle mechanical vibrations before they can be radiated as a tonal. By training our neural network to recognize tonals and associate those tonals with a given target, we can effectively teach the neural network a sonar target's acoustic signature [Ref.2: p.341].

TABLE 3.1: A LIST OF TONALS USED TO SIMULATE A SONAR TARGET

Sonar Target Signal	Signal Source	Frequency (Hz)
Tonal A	Motor Bearing	88 Hz @ 160 dB
Tonal B	Reduction Gear	53-57 Hz @ 165 dB
Tonal C	Flow Noise	203 Hz @ 169 dB
Tonal D	Generator	60, 120, 180, 240 Hz @ 164 dB
Tonal E	Fuel Oil Pump	135 Hz @ 158 dB

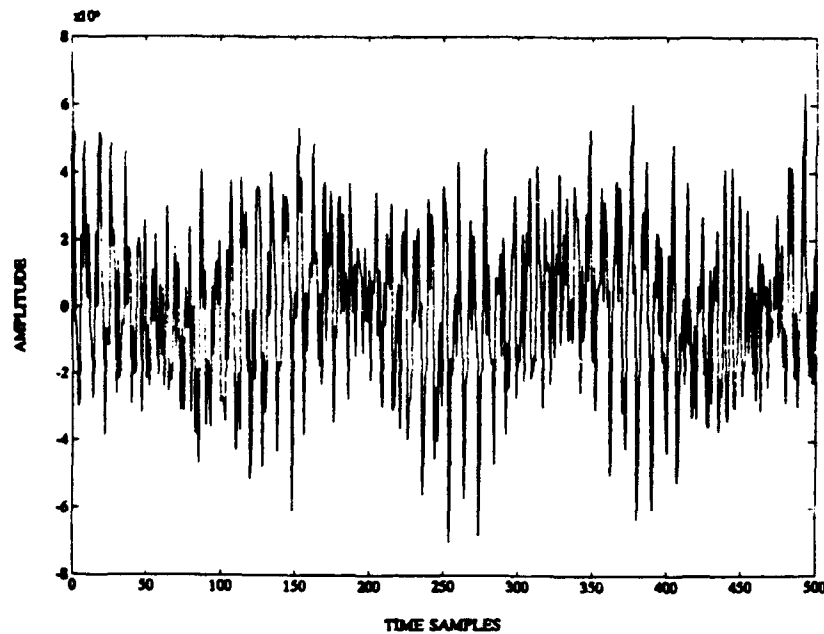


Figure 3.3 (a): This is a plot of the simulated acoustic signal produced by a single sonar target. Cavitation noise, blade rate noise, and narrow-band tonals are included.

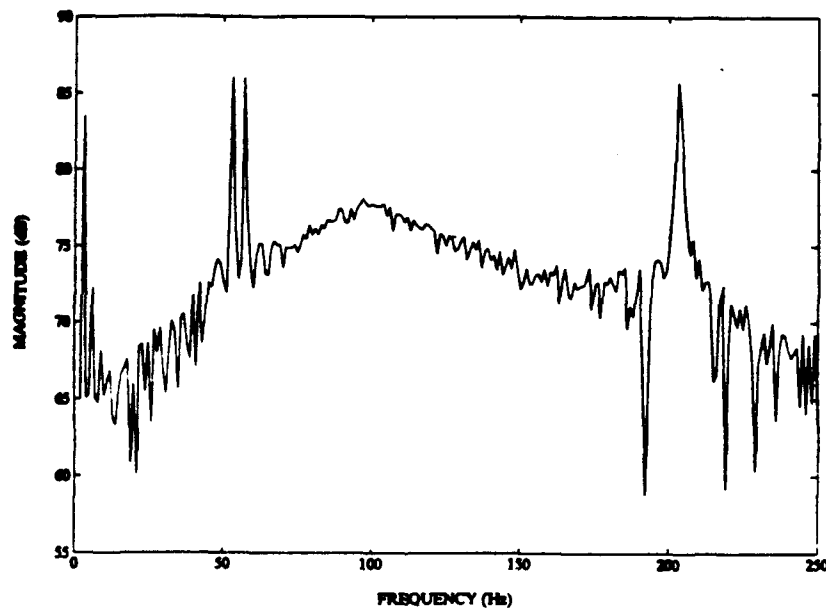


Figure 3.3 (b): This plot is the frequency spectrum of Figure 3.3 (a). This spectrum was created by a target traveling at 8 knots, at a range of 5000 yds, and with a 10 degree aspect (Refer to Figure 3.4). Ambient-noise was not included.

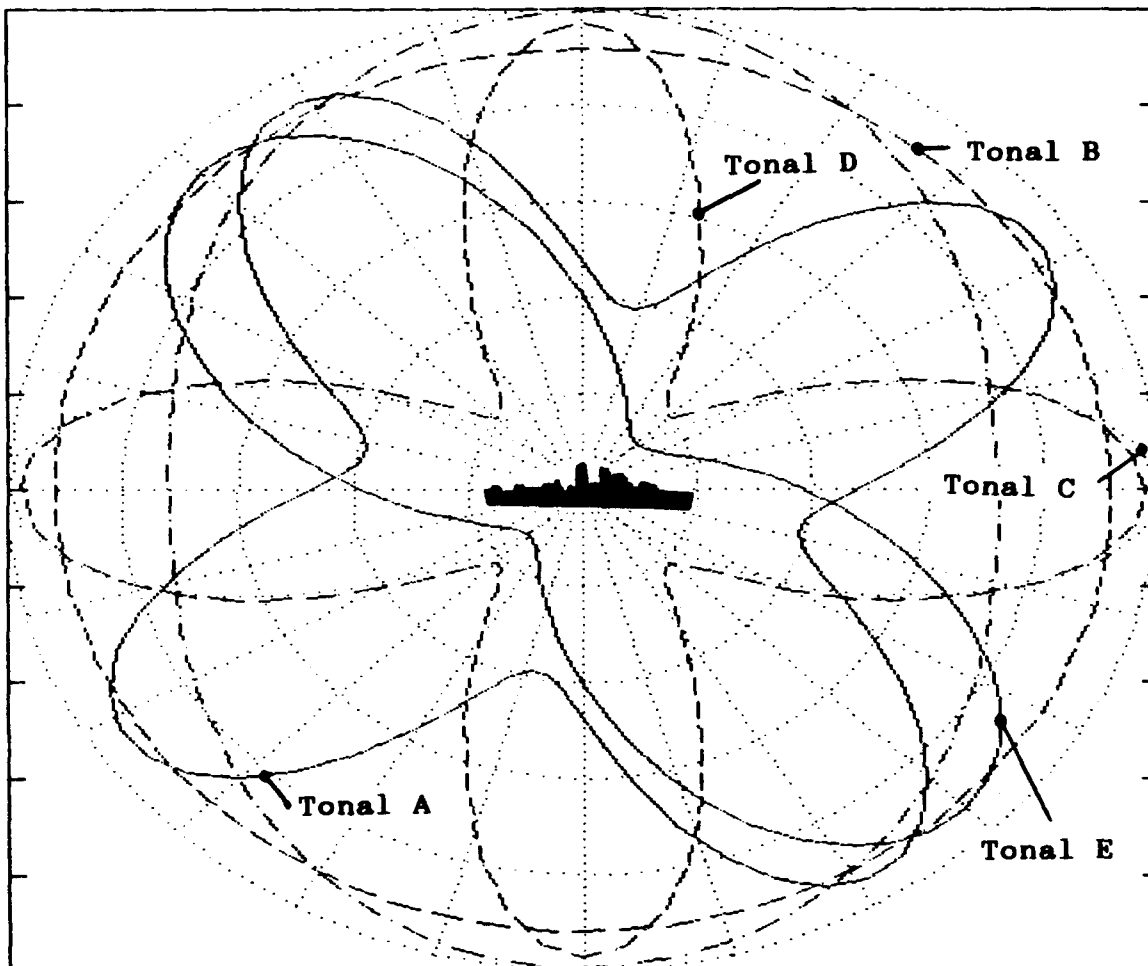


Figure 3.4: This diagram shows the directivities of the tonals in Table 3.1 as a function of aspect. Note that at a 10 degree aspect, the most prominent tonals are the 203 Hz and 53-57 Hz tonals accompanied by the blade rate spectra. Compare these tonals to the spectrum of Figure 3.3 (b).

C. TRANSMISSION LOSSES

In the sonar target model, transmission losses will be a function of range and frequency. The spreading losses previously described will be a function of range while attenuation will be a function of range as well as frequency. An expression for the attenuation coefficient was developed from empirical data measured by W.H Thorpe in 1967 [Ref. 2: p. 108]. This particular attenuation coefficient assumes a sea water temperature of 39° F and a depth of 3000 feet. The expression is stated as

$$\alpha = \frac{0.1F^2}{1 + F^2} + \frac{40F^2}{4.100 + F^2} + 2.75 \times 10^{-4}F^2 + 0.003 \quad (3.4)$$

where α is the attenuation coefficient expressed in dB per kiloyard [Ref.2: p.108]. Using this attenuation coefficient, the transmission loss equation is given by

$$TL = 20 \log r + \alpha r \times 10^{-3} \quad (3.5)$$

where TL is expressed in dB and r is the range to the target in yards [Ref.2: p.111].

D. AMBIENT-NOISE LEVEL

The last important component of the sonar target model is the ambient-noise level which represents all of the background noise in the underwater environment. The several factors contribute to ambient-noise. First, wind, rain and wave action make up a large part of the ambient-noise, and secondly,

shipping density and biological noises contribute to the remaining part of ambient-noise (Figure 3.5). The ambient-noise component of the sonar target model will provide the neural network with a broad range of complex relationships, and will aid in the neural network's generalization process [Ref. 10]. Our goal is to train the neural network with as many examples as possible from the sonar target model. After the neural network is trained, it will be tested using realistic scenarios generated by the sonar target model.

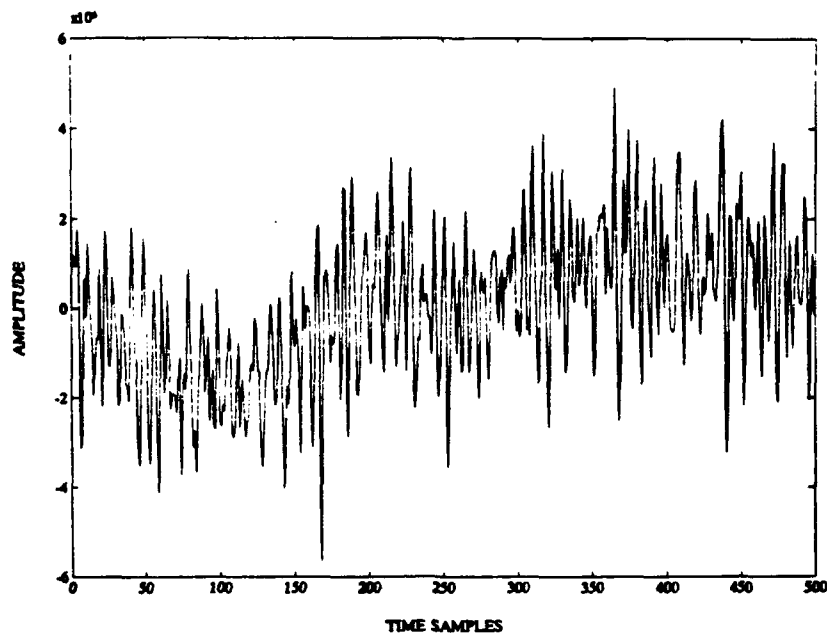


Figure 3.5 (a): This is a plot of the ambient-noise signal generated in a sea state one.

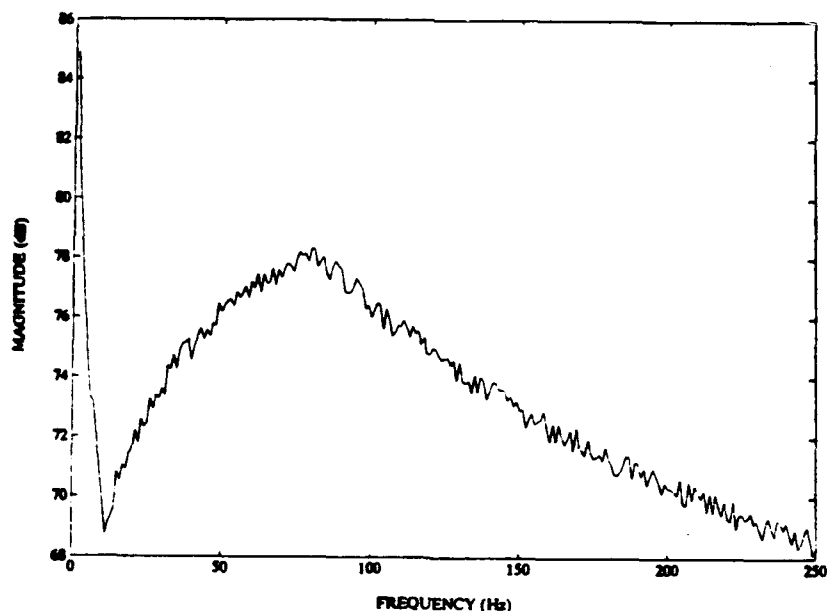


Figure 3.5 (b): This plot is the frequency spectrum of Figure 3.5 (a). Note that there are two clear peaks. The first peak lies below 10 Hz and represents low frequency noise caused by waves and currents. The second peak, centered at 80 Hz, represents noise caused mostly by wind.

IV. NEURAL NETWORK TRAINING AND TESTING

A. NEURAL NETWORK DESIGN CONSIDERATIONS

Once the sonar target model was completed and it conveyed an accurate simulation of a passive sonar target, the neural network was ready to be trained and tested. As a basis for our neural network design, we decided to train the neural network to recognize four different signals. The first signal would be ambient-noise. By providing the neural network with samples of the ambient-noise spectrum, we could aid the neural network in establishing a reference level whereby it could judge either the presence or the absence of sonar targets. When the neural network recognizes only ambient-noise, no detectable targets are present. The remaining three signal spectrums which we call target one, target two, and target three represent the three sonar targets the neural network was trained to recognize. When one or all of these targets approach a detectable range, the neural network should indicate their presence, otherwise, the neural network would only indicate the presence of ambient-noise.

Once we had decided upon a testable criteria for our neural network, we needed an efficient design. Unlike other design methods in which the design path is clearly marked, the path to designing a neural network is less apparent. There is

no single or correct approach for conjuring the perfect neural network arrangement. Often much experimentation and trial and error is required to arrive at an optimal arrangement of input, hidden and output layers, and the best combination of training vectors and learning parameters. In designing a neural network, there are three basic aspects of the design that must be considered:

1. Choosing an appropriate learning rule.
2. Arranging an effective neural network architecture.
3. Selecting some suitable training examples for educating the neural network.

For the first step, one must choose the best learning rule. For this design, the back-propagation algorithm was chosen because of its reputation for producing very effective neural networks. Choosing the learning rule often dictates the choices for the other design considerations. By choosing the back-propagation algorithm, we must use an accommodating neural network architecture that leads to the best results. As we saw in Chapter two Section B, the best arrangement for a neural network using back-propagation is a multi-layered network with input, hidden and output layers. Therefore, in the second design step, we decided upon a hidden layer style architecture which left us to choose the best layer arrangement. Previous experience with layered architectures has shown [Ref. 10] that as the ratio of input elements to

hidden elements increases, the better the neural network becomes at generalization. Therefore, we had to choose more input elements in relation to hidden elements. In order to find the best number of processing elements for the hidden layer, we looked at two rules-of-thumb which were commonly used by neural network designers. The first rule-of-thumb states that "the more complex the relationship between the input data and the desired output, the more PEs (processing elements) are normally required in the hidden layer" [Ref.11]. The second rule-of-thumb concerning the number of hidden layer units, h , can be best expressed in the formula [Ref. 11]

$$h = \frac{K}{C_f * (m + n)} \quad (4.1)$$

where the numerator, K , represents the number of training vectors available to train the network, and the variables m and n represent the number of processing elements occupying the input and output layers, respectively. Note that C_f represents the data complexity factor. Based on previous experience, it has been observed that as the complexity or noise level of the training data increases, the higher the data complexity coefficient needs to be in order to create an effective neural network [Ref. 11]. Generally, for relatively clean data, the C_f coefficient should be less than ten, and for noisy data one would expect a coefficient C_f greater than ten [Ref.11]. One other factor to consider in choosing the

number of hidden elements is the computation time required for training. The more processing elements there are to train, the longer it takes for the neural network to converge to its final training state. Based on these considerations and a few preliminary test results, the decision was made to divide the 256 Hz frequency spectrum (Figure 4.1) into 64 4Hz bins thus forming the input vector for the neural network. Using 64 instead of 256 input bins helped to reduce computation time while maintaining most features in the frequency spectrum (Figure 4.2). Since the neural network accommodates the recognition of three sonar targets plus ambient-noise, the decision was made to use 32 processing elements in the hidden layer. This number would tend to balance the complexity involved in training for three targets plus ambient-noise with the need to generalize the network. Finally, four processing elements were chosen for the output layer (Figure 4.3). The final output elements represented the ambient sea state noise, target one, target two, and target three, respectively.

The last design consideration concerns the type of training data needed for the best training results. From experiments conducted in the literature, it was apparent that noisy training samples were better suited for training neural networks than clean samples because they force the neural network to generalize [Ref. 10]. Without generalization, the neural network would only be able to recognize inputs that strongly resembled the training data, thus making the network

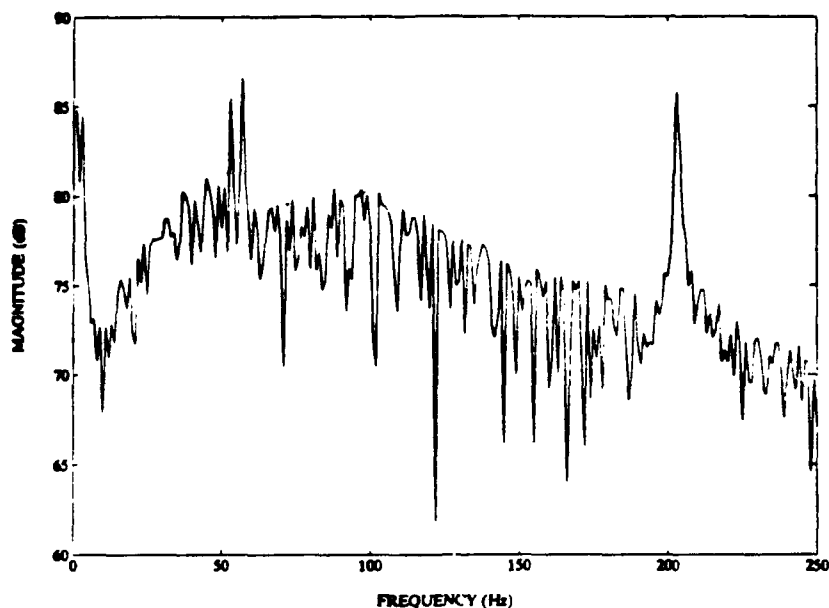


Figure 4.1: This plot is the frequency spectrum of a simulated sonar target signal as seen at the receiver. This plot is the signal in Figure 3.3 after it was contaminated by ambient-noise.

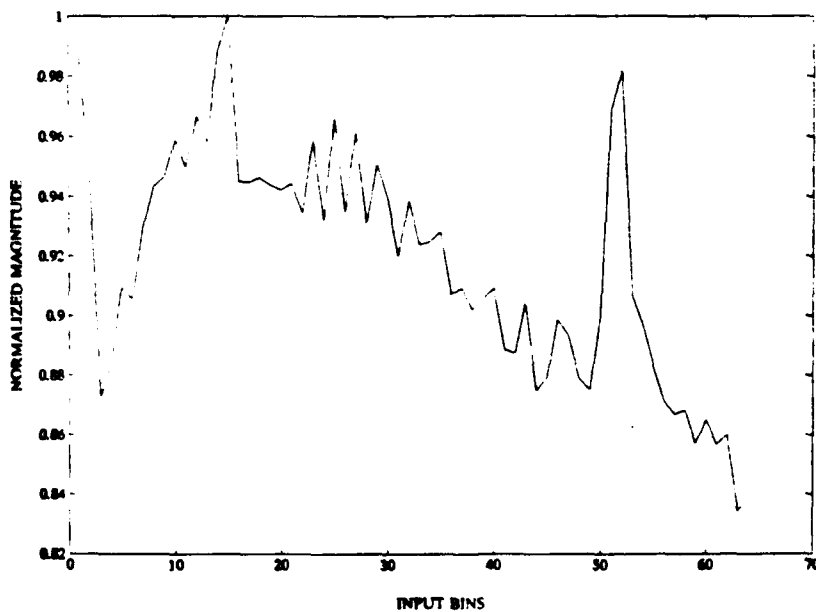


Figure 4.2: This plot is the frequency spectrum of Figure 4.1 after it was condensed into 64 frequency bins. These bins were used as inputs to the neural network. Note that the 64 bins retain the spectrum's key characteristics.

useless for identifying targets presented in diverse environments. Thus, the decision was made to include noise in the training data to enhance the neural network's overall performance.

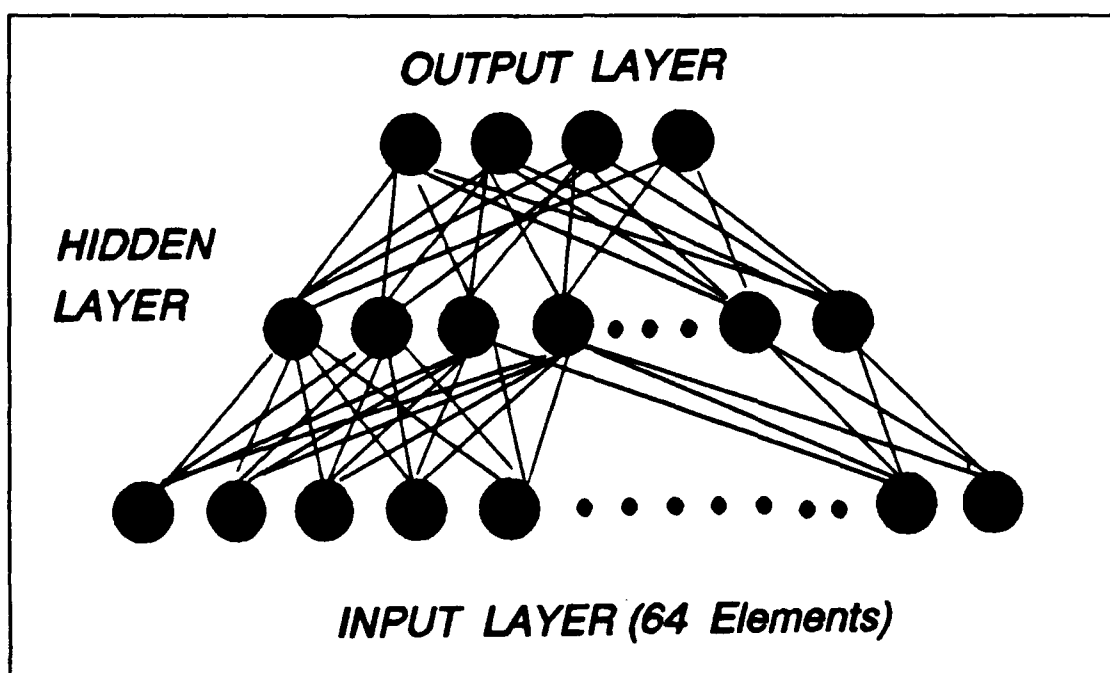


Figure 4.3: This is a diagram of the first neural network architecture with 64 processing elements in the input layer, 32 in the hidden layer, and four processing elements in the output layer.

Lastly, we used 50 samples of each target. The samples represented each target at random ranges, speeds, and aspects. The neural network was trained to produce an output value of one if a desired signal was present or a zero if it was absent. In preparing the training data, we discovered one note of caution. As training data is gathered, one must limit the

number of training examples used to avoid overtraining, otherwise, the neural network might recognize anything, including false targets. Thus, one must be selective in choosing the quality and quantity of training data because overtraining degrades the neural network's performance.

B. TESTING THE FIRST NEURAL NETWORK ARCHITECTURE

After considering the design parameters, we trained the neural network using the back-propagation algorithm provided by NeuralWare software [Ref. 11]. The network was instructed to converge to a mean-square error threshold of 0.005 requiring 550,000 training iterations. A sigmoid function was used as the threshold function. To test the network's memory, two scenarios were created. In the first scenario, the neural network was presented with a single target, target one (Figure 4.4). In the second scenario, target one was accompanied by the two other targets, target two and target three (Figure 4.5). In both scenarios, the targets were initially placed at a distance of 16,000 yards from a receiving transducer acting as the reference point. All targets maintained a constant course and speed throughout the test. In the multi-target scenario, target two led target one by 2000 yards, and target one led target three by 2000 yards. With this arrangement, one should expect the receiver to detect target two first, followed by target one and then target three. As the targets approach the receiver, we should expect times when two or more

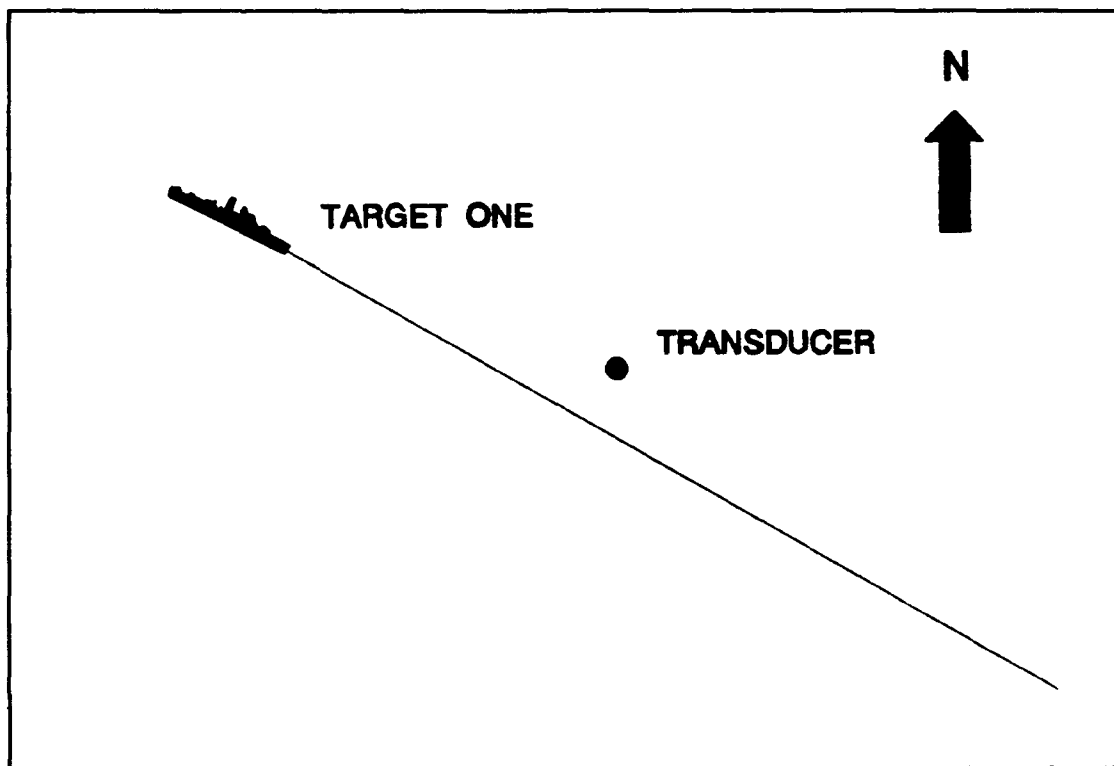


Figure 4.4: This is a diagram of scenario one. Initially, target one is stationed 16000 yards from the receiver traveling on a course of 130 degrees at 8 knots. The CPA is 3000 yards.

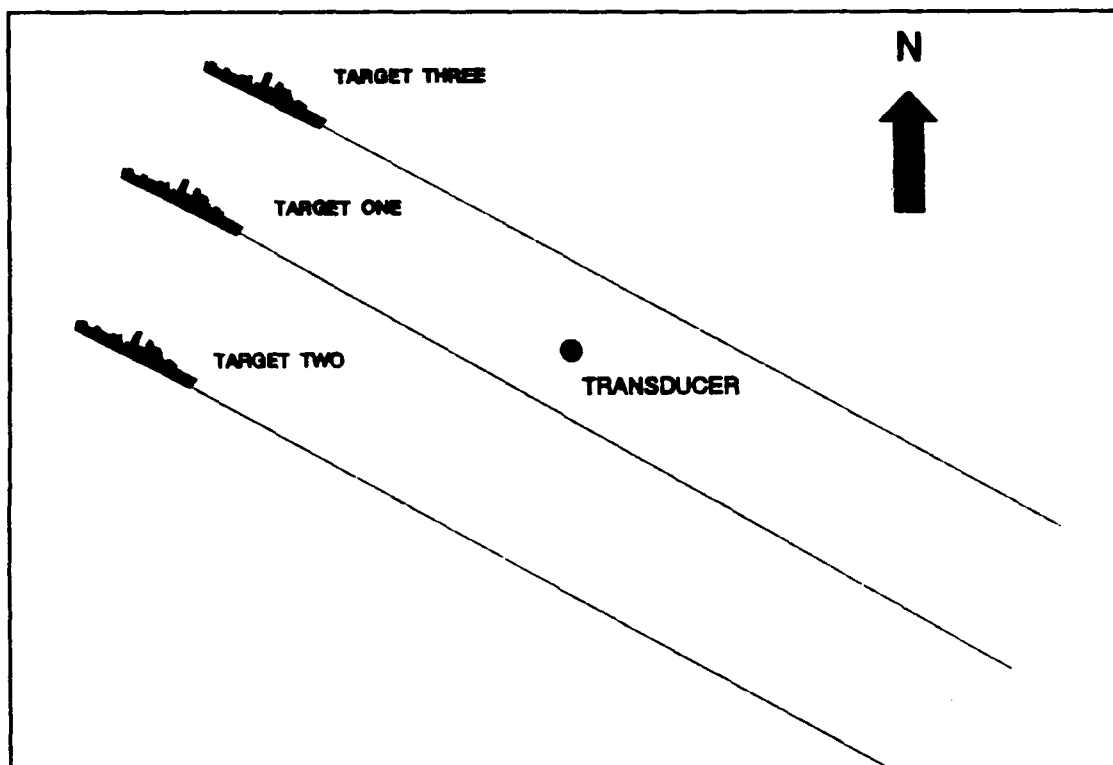


Figure 4.5: This is a diagram of scenario two. All targets are initialized at the same course and speed as in scenario one. The CPAs for targets one, two and three are 3000, 7000, and 4000 yards, respectively.

targets are detected and identified simultaneously. In the absence of any targets, we should expect the neural network to respond solely to ambient-noise. The goal of these test scenarios was to evaluate the neural network's ability to detect and identify a passive sonar target or a group of targets even when those targets are changing in range and aspect. In order to simplify the test, we limited the overall ambient-noise to a sea state of one and a low shipping density environment.

After the tests were conducted, the single target scenario was evaluated first and several key observations were made. In viewing the data, we decided to accept all output values above 0.90 as a signal detection, and all those below 0.10 as a signal not present (Appendix A: Figure A.1 (a) (b) (c) (d)). All output values between 0.10 and 0.90 were considered as indeterminate. Initially, the neural network behaved as expected. For the first four observation times, only ambient-noise was detectable because target one was at a great distance away. Between observation times five and seven, the network began approaching an indeterminate state as if it were trying to decide whether target one or ambient-noise was the strongest signal present. As target one drew closer, it appears the neural network was building a stronger impression. By observation time eight, target one was clearly identified. However, at observation time nine, the neural network made a surprising switch. Suddenly, target two emerged and target one

disappeared. At observation time ten, target one returned and target two disappeared. Gradually, as target one passes its closest point of approach (CPA) and continues on, the neural network lost contact. By observation time 14, target one was no longer detected and ambient-noise became the dominant signal. The performance of the neural network in this test raised two important questions. First, why did the network flip-flop from target one to target two and then back to target one? Secondly, when target two appeared why did target one totally disappear? In answer to the first question, it seems likely that at certain aspect angles, two different targets may appear alike. This would be analogous to a human being trying to recognize someone at a particular angle or distance away, only to find as he gets closer and sees more identifying traits, the person he thought he recognized was someone else. Similarly, the neural network can exhibit the same behavior. An answer to this problem is to give the network more training on its deficiencies or move the neural network to another location so it can get a better look. It is possible that a wide array of interlinked transducers could give the neural network a better consensus. In answer to the second question, the fact that target one disappeared when target two emerged showed that certain processing elements are trained to dominate other processing elements depending on the strength of a given input signal. Thus, it appears that at least one target, target two in particular, had been trained

to dominate over the other targets. Both of these problems need to be solved before the neural network can function correctly. As we evaluated the second scenario, we found that it had similar drawbacks as witnessed in the first scenario. This time, however, all three targets were present in the input signal simultaneously (Appendix A: Figure A.2 (a)(b)(c)(d)). As the scenario began, the neural network seemed to be focusing on the targets. The detection level of the ambient-noise was very low. In observation times two and three the neural network detected and clearly identified target three. By observation time four, the neural network began a transition between target two and target three. By observation time five, target two was the dominant target sensed by the network. The network maintained contact with target two until observation time ten when there was another transition period between targets two and three. This time target three became dominant at about the time it reached CPA. From observation times 13 to 19 the targets were traveling out of range and the neural network clearly identified the presence of ambient-noise. As in the first scenario the neural network displayed the tendency to choose one dominant target over the others. In other words, the weaker targets were inhibited by the stronger target. Only one target was allowed to be detected and identified at a time. To make matters worse, certain targets, like target one, were completely dominated and never detected at all. These

inherent weaknesses had to be corrected so that all the targets could be detected and identified concurrently.

C. REVISING THE NEURAL NETWORK ARCHITECTURE

After evaluating the deficiencies of the first neural network, the decision was made to revise it using some key alterations. The modified network still retained 64 input elements, 32 hidden elements and four output elements. However, the connection scheme for those elements were changed. Instead of connecting all elements in the hidden layer to each element in the output layer, the hidden layer was divided into four separate regions, each servicing one output element (Figure 4.8). Each eight element region in the resulting neural network was specifically trained as an expert on either target one, target two, target three, or ambient-noise. In this scheme, each output element could be physically isolated from the others thus preventing any inhibitory cross-talk. Training the neural network was conducted as before using the same set of training vectors.

After the second neural network architecture was retrained and implemented, it was tested using the same previous scenarios. In scenario one, once again, only target one was present while traveling at a constant course and speed. This time, however, target one was detected earlier at observation time seven (Appendix A: Figure A.3 (a)(b)(c)(d)). Moreover, the contact was not broken once it was made. In other words,

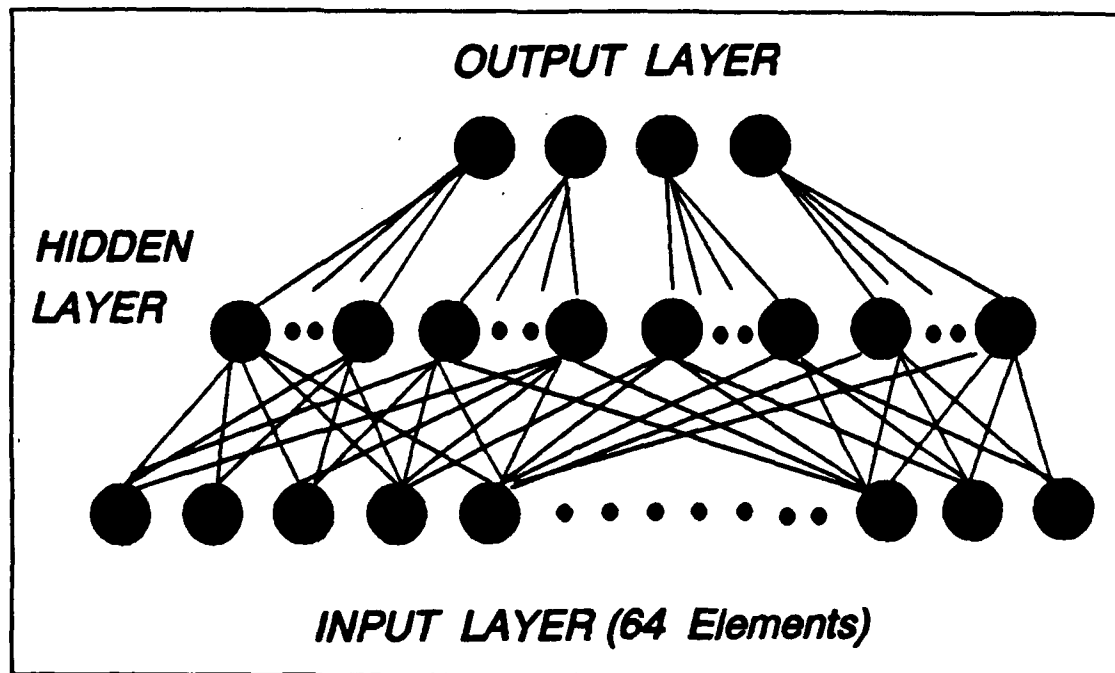


Figure 4.8: This is a diagram of the modified neural network architecture. Notice that the output elements have been isolated to their respective expert regions.

target two did not suddenly intervene. Even though target two did appear briefly during observation time nine, the network no longer sensed it as strongly as it did before and most importantly, its presence no longer interfered with the network's perception of target one. The modified architecture made each output element more sensitive to its respective target and removed the damaging cross-talk that previously interfered with the target's detection and classification.

In the second scenario, all three targets were presented to the neural network as before. This time, however, it was apparent that no one target dominated the neural network

(APPENDIX A: Figure A.4 (a)(b)(c)(d)). Acoustically, target two seemed to interfere with target one and target three by masking their respective signals. However, as a result of the modified neural architecture, all three targets were expressed more clearly and concurrently. Target one was no longer suppressed by target two.

The modified neural network architecture appears to be the best approach toward solving the problem of passive sonar target recognition for several reasons. First, by dividing the hidden layer into separate expert regions, we removed the cross-talk that inhibited the output elements from revealing a target detection, even though the target was present and should have been detectable. Furthermore, by intentionally training each expert region to regard the inputs of other targets as non-detections, we created a training scheme whereby false targets are actively ignored and the sensitivity towards the desired target is reinforced and improved. This added sensitivity removed the neural network's propensity for spurious guessing which generally resulted in erroneous detections. A second advantage of the improved neural architecture was that by decreasing the number of hidden units, we could speed up computation time and better enhance the generalization property of the neural network. Lastly, by using the modified neural architecture, we can easily add new targets to the neural network without changing or effecting any of the existing processing elements. For instance, a new

region of hidden elements devoted to a new target could be separately trained and then added to the existing network at some future time. This can be accomplished by simply grafting one more region of hidden elements plus an additional output element onto the existing neural network. One can imagine such a neural network architecture to be very justifiable for sonar system applications.

V. CONCLUSIONS

The main objective of our research was to design a neural network which could identify the acoustic signature of a given passive sonar target. To achieve this objective, we followed these basic steps. First, we created a computer program using MATLAB [Ref. 12] (APPENDIX B) that could simulate a realistic passive sonar target. Next, using the target simulations from our passive sonar target model, we trained the neural network to recognize the target's simulated acoustic signature by using the back-propagation algorithm. When training was completed, we tested the neural network scheme by providing two separate scenarios which helped us judge the effectiveness of the neural network's performance.

What we discovered was that a neural network could be implemented to recognize an acoustic signature by observing the following restrictions. First, training a neural network can be a very lengthy process, however, by optimizing the number of processing elements, adjusting the learning parameters and using an appropriate threshold function, we could speed up the overall training process. In addition, we found that by choosing the best neural network architecture and by using the best combination of training examples, we could greatly improve overall performance of the neural network. In our investigation, the best performing neural

architecture used 64 input elements connected to a hidden layer which was sub-divided into four separate regions. Each region was a trained expert servicing one output element. This connection scheme prevented one target from dominating the others and allowed more than one target to be recognized or identified simultaneously. Lastly, we noted that neural network's are not 100% foolproof. As observed in scenario one, it is possible for one target to resemble another target under certain conditions just as a person might mistakenly recognize someone at a distance. This limitation can be improved by retraining a deficient neural network with better training examples or by providing the neural network with better observations from which to judge. In addition, the preprocessing of input data using appropriate signal processing techniques to improve the SNR of one target over another can help to enhance the neural network's performance.

Neural networks can be useful in situations where the qualities of human-like pattern recognition capabilities are required but where humans beings cannot be deployed, such as in deep sea SOSUS stations, incorporated into the guidance systems of weapons such as torpedoes or mines, or when employed in small sensors such as sonobuoys. Neural network technology presents a viable option for improving contemporary sonar systems. In future investigations, it would be challenging to design neural network schemes that were connected to a large array of sonar transducers. By increasing

the number of transducers, we could provide the neural network with multiple observations taken from different angles. This scheme could greatly enhance the neural network's ability to identify and draw clearer conclusions about the objects it senses. Other areas and applications that can be investigated are numerous. Some of them include the development of better neural network architectures, the use of real sonar targets vice simulated sonar targets as training data, and the implementation of other more highly adaptive learning algorithms such as those using competitive learning rules.

REFERENCES

1. Douglas O'Shaughnessy, Speech Communication, p. 142, Addison-Wesley Publishing Company, Reading, MA, 1990.
2. Robert J. Urick, Principles of Underwater Sound, 3rd ed., McGraw-Hill Book Company, New York, NY, 1983.
3. Leon N. Cooper, "Adaptive Pattern Recognition: Neural Networks In Real World Applications", Tutorial IEEE/INNS International Joint Conference On Neural Networks, June 18, 1989, IEEE Publishing Services, New York, NY, 1989.
4. R. Paul Gorman and Terrence J. Sejnowski, "Learned Classification of Sonar Targets Using A Massively Parallel Network", IEEE Transactions On Acoustic Speech and Signal Processing, Vol. 36, No. 7, pp. 1135-1140, July 1988.
5. William F. Allman, Apprentices of Wonder: Inside the Neural Network Revolution, pp. 130-131, Bantam Books, New York, NY, 1989.
6. Robert Hecht-Nielsen, Neurocomputing, pp. 1-19, Addison-Wesley Publishing Company, Reading, MA, 1990.
7. NeuralWare, Inc., Neural Computing: NeuralWorks Professional II/PLUS and NeuralWorks Explorer, pp. NC-3 to NC-8, NeuralWare, Inc. Technical Publications Group, Pittsburgh, PA, 1991.
8. William P. Jones and Josiah Hoskins, "Back-Propagation: A Generalized Delta Learning Rule", BYTE, October 1987, Vol. 12, No. 11, pp. 155-162, McGraw-Hill, New York, NY, 1987.
9. David E. Rumelhart and James L. McClelland and the PDP Research Group, "Learning Internal Representations by Error Propagation", Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol.1: Foundations, David E. Rumelhard and James L. McClelland, eds., pp. 318-362, The MIT Press, Cambridge, MA, 1988.
10. Jocelyn Sietsmar and Robert J.F. Dow, "Creating Artificial Neural Networks That Generalize", Neural Networks, Vol. 4, pp. 67-79, Pergamon Press plc, 1991.

11. NeuralWare, Inc., Using NWorks: An Extended Tutorial For NeuralWorks Professional II/PLUS and NeuralWorks Explorer, pp. UN-18 to UN-21, NeuralWare, INC. Technical Publications Group, Pittsburgh, PA, 1991.
12. The MathWorks, Inc., PRO-MATLAB, Version 3.5h, The MathWorks, Inc., South Natick, MA, 1990.

APPENDIX A: TEST DATA

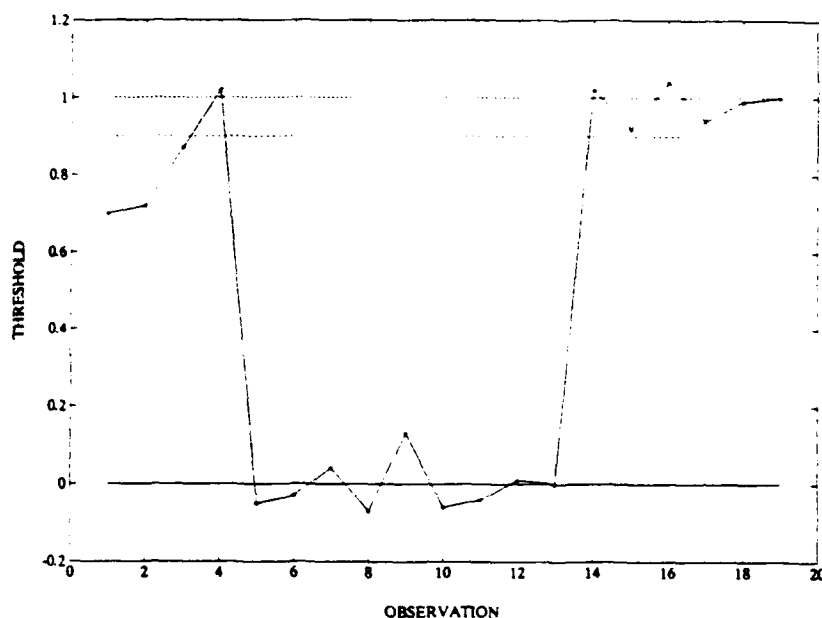


Figure A.1 (a): The activity level of the ambient-noise processing element, located in the output layer, during scenario one. This was part of the evaluation of the first neural network architecture (See Table A.1).

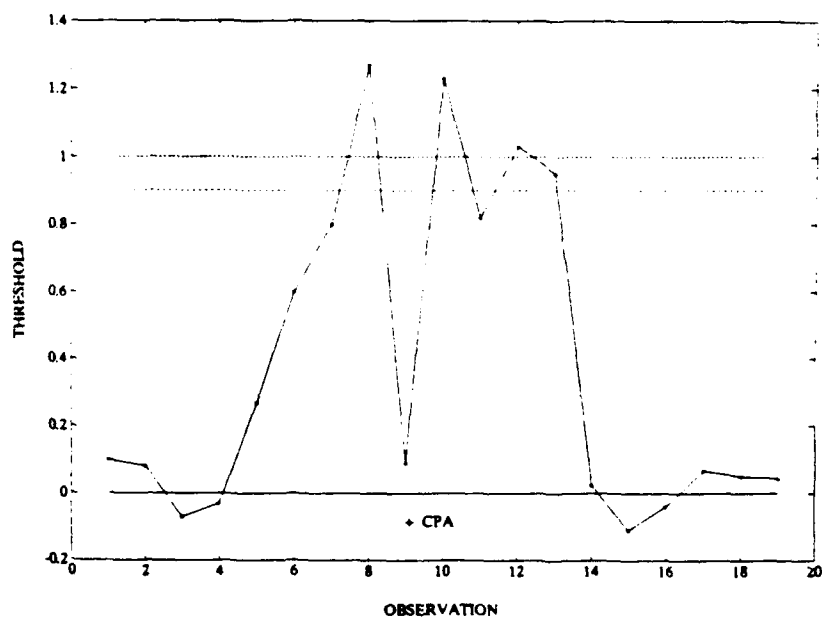


Figure A.1 (b): The activity level of target one's processing element, located in the output layer, during scenario one. This was part of the evaluation of the first neural network architecture (See Table A.1).

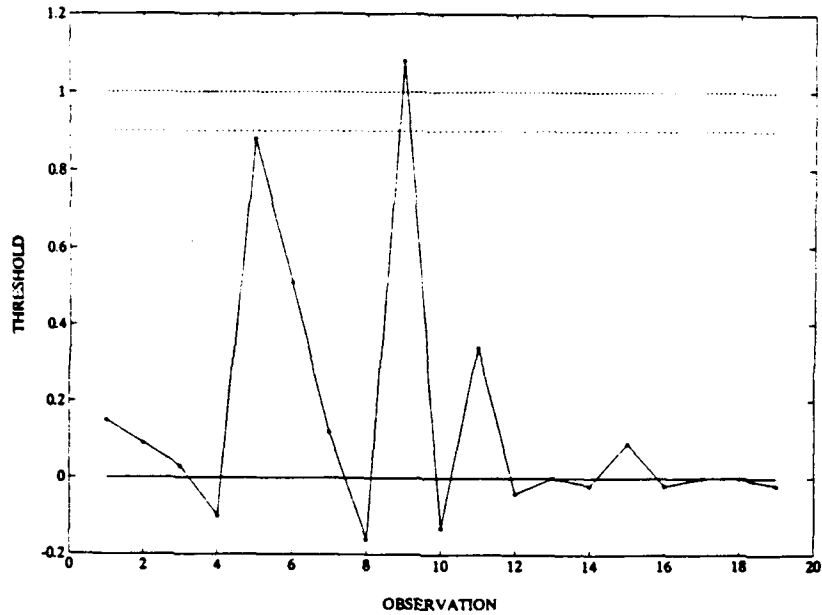


Figure A.1 (c): The activity level for target two's processing element, located in the output layer, during scenario one. This was part of the evaluation of the first neural network architecture (See Table A.1).

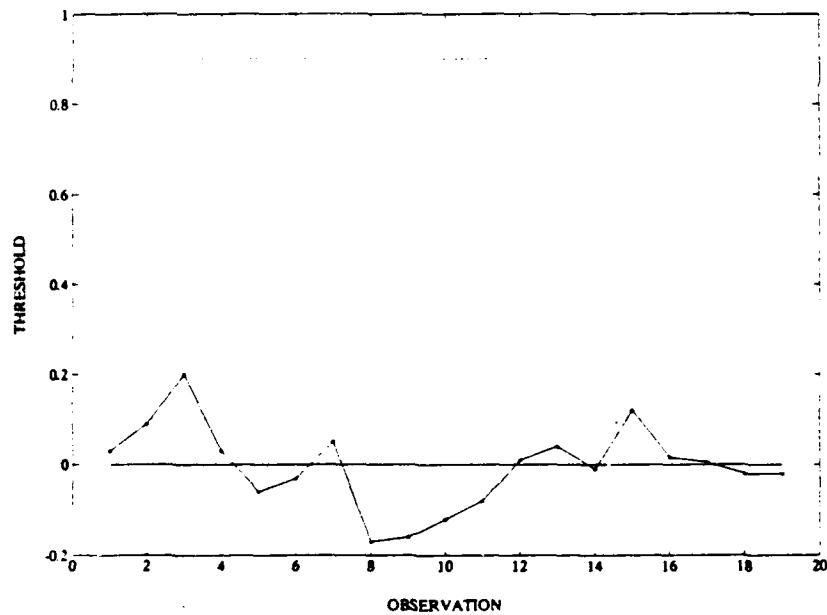


Figure A.1 (d): The activity level of target three's processing element, located in the output layer, during scenario one. This was part of the evaluation of the first neural network architecture (See Table A.1).

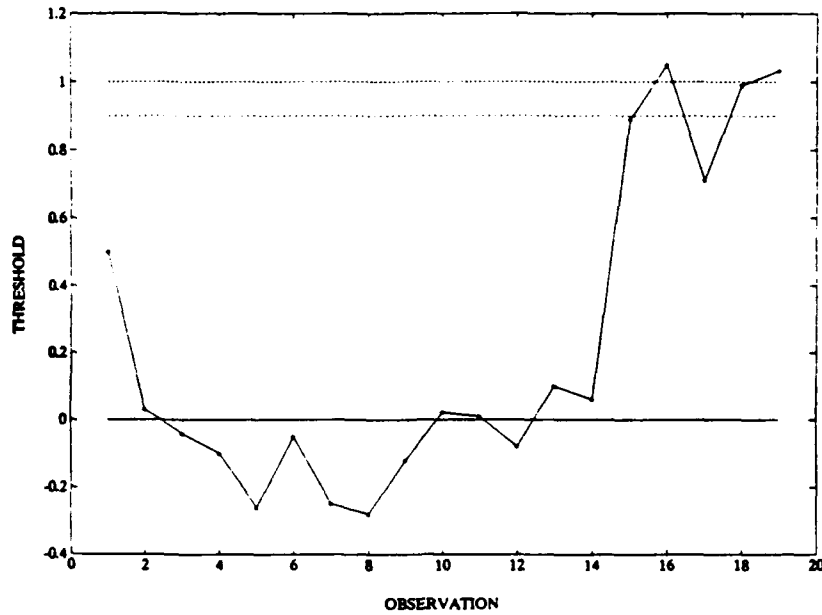


Figure A.2 (a): The activity level of the **ambient-noise** processing element, located in the output layer, during **scenario two**. This was part of the evaluation for the **first** neural network architecture (See Table A.2).

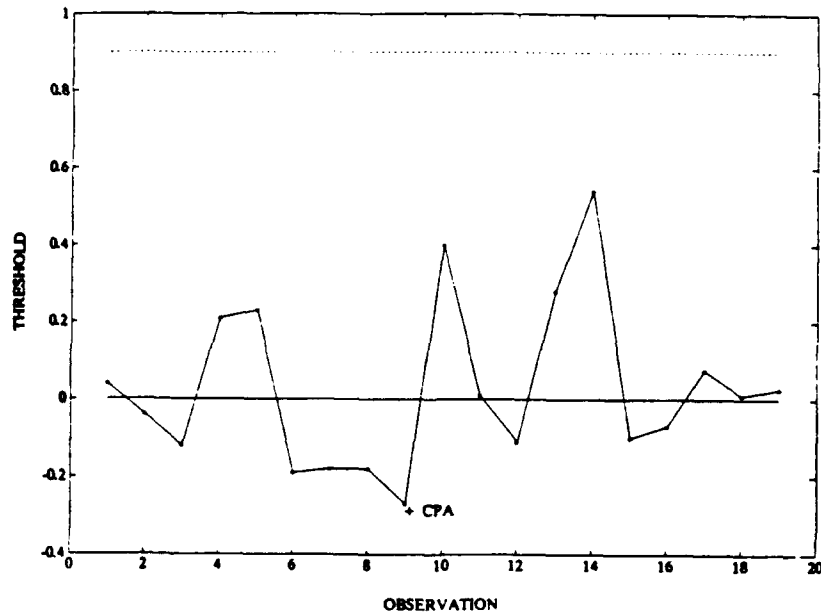


Figure A.2 (b): The activity level of **target one's** processing element, located in the output layer, during **scenario two**. This was part of the evaluation of the **first** neural network architecture (See Table A.2).

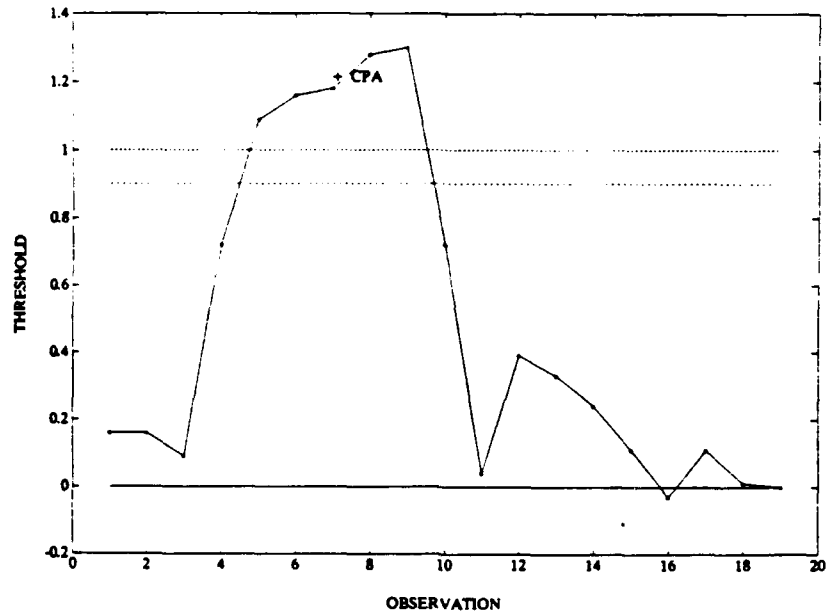


Figure A.2 (c): The activity level of target two's processing element, located in the output layer, during scenario two. This was part of the evaluation of the first neural network architecture (See Table A.2).

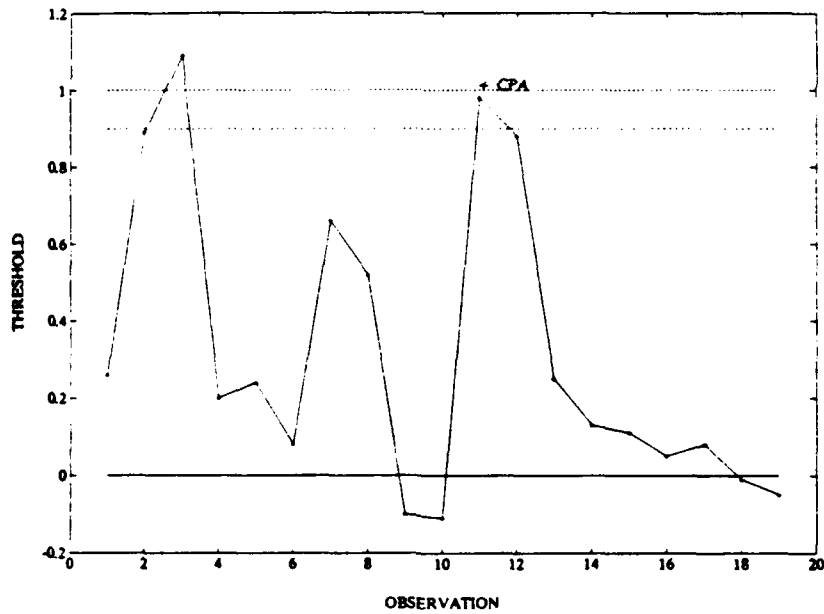


Figure A.2 (d): The activity level of target three's processing element, located in the output layer, during scenario two. This was part of the evaluation of the first neural network architecture (See Table A.2).

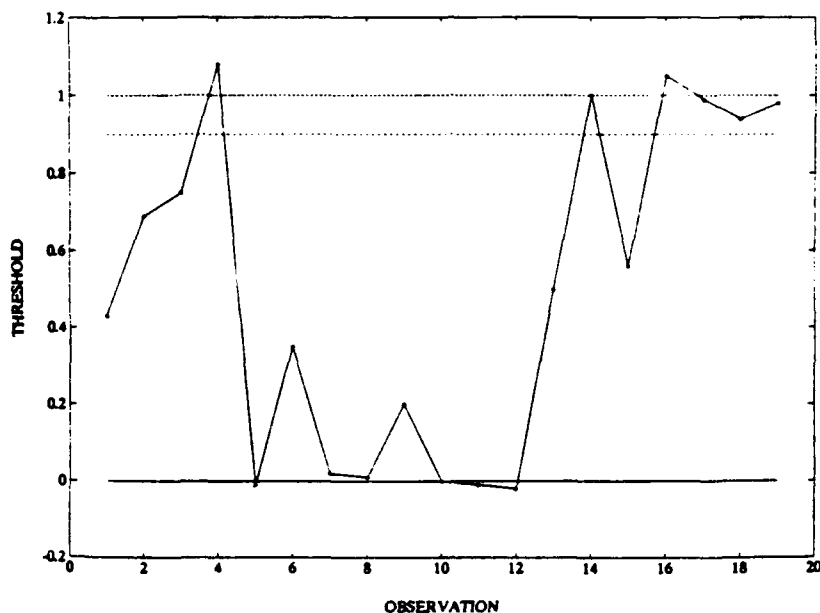


Figure A.3 (a): The activity level of the ambient-noise processing element, located in the output layer, during scenario one. This was part of the evaluation of the modified neural network (See Table A.3).

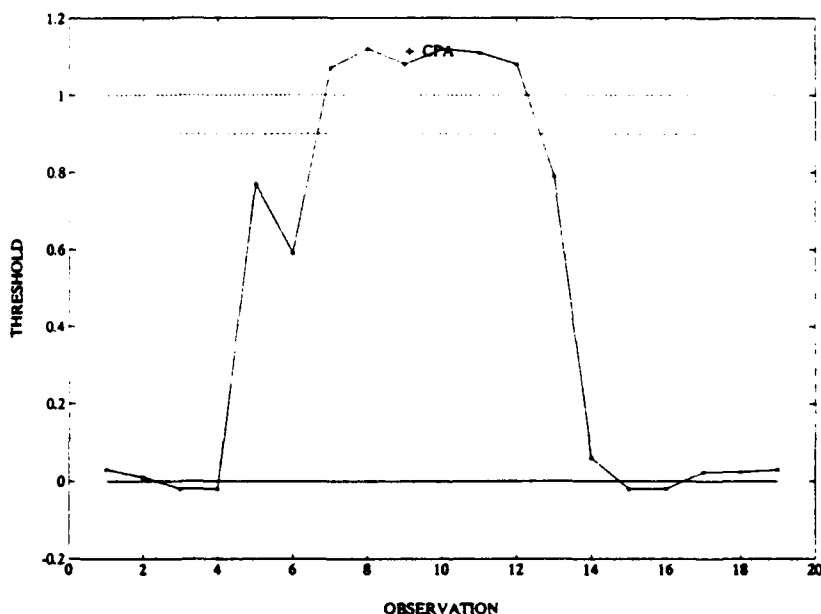


Figure A.3 (b): The activity level of target one's processing element, located in the output layer, during scenario one. This was part of the evaluation of the modified neural network (See Table A.3).

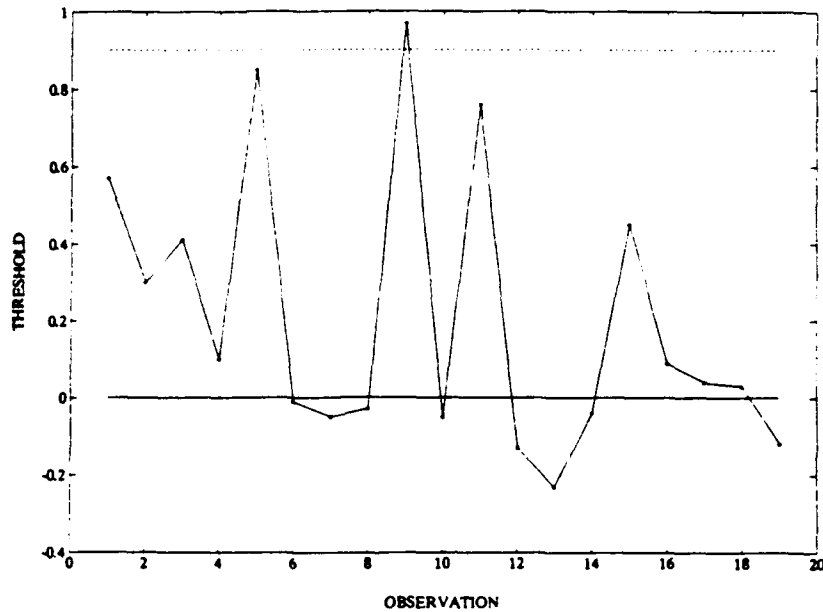


Figure A.3 (c): The activity level of target two's processing element, located in the output layer, during scenario one. This was part of the evaluation of the modified neural network (See Table A.3).

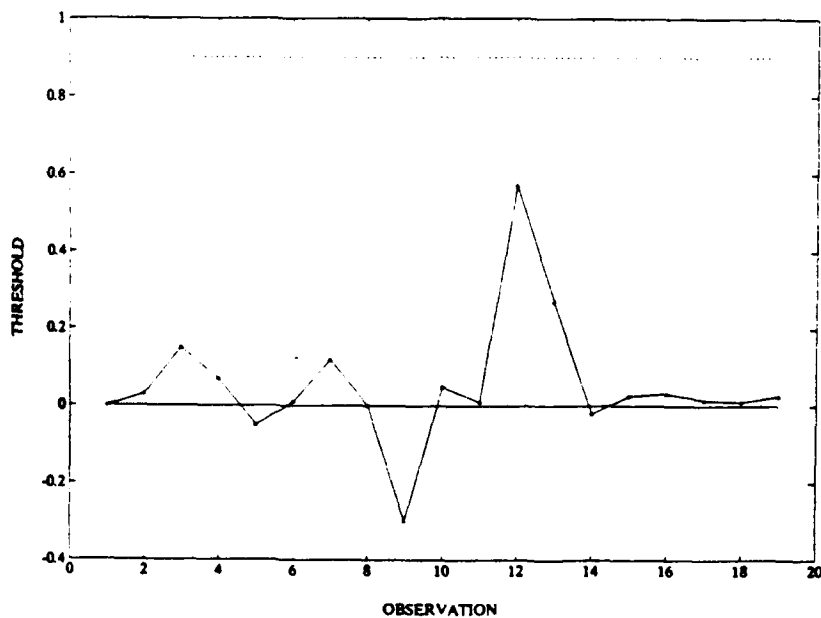


Figure A.3 (d): The activity level of target three's processing element, located in the output layer, during scenario one. This was part of the evaluation of the modified neural network (See Table A.3).

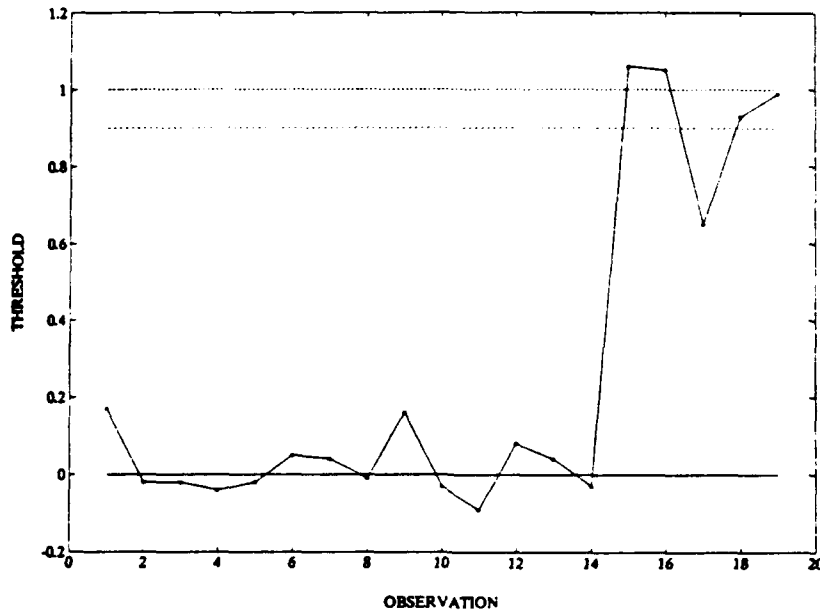


Figure A.4 (a): The activity level of the ambient-noise processing element, located in the output layer, during scenario two. This is part of the evaluation of the modified neural network (See Table A.4).

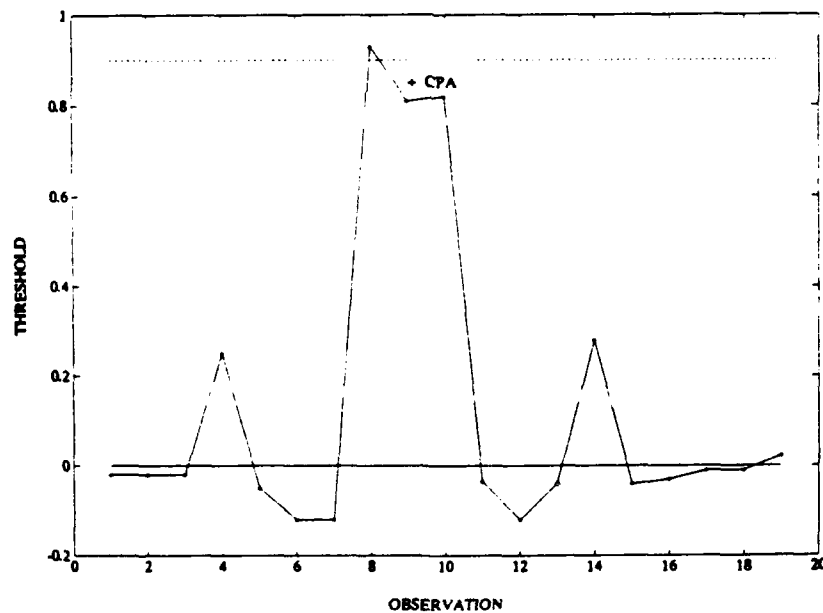


Figure A.4 (b): The activity level of target two's processing element, located in the output layer, during scenario two. This was part of the evaluation of the modified neural network (See Table A.4).

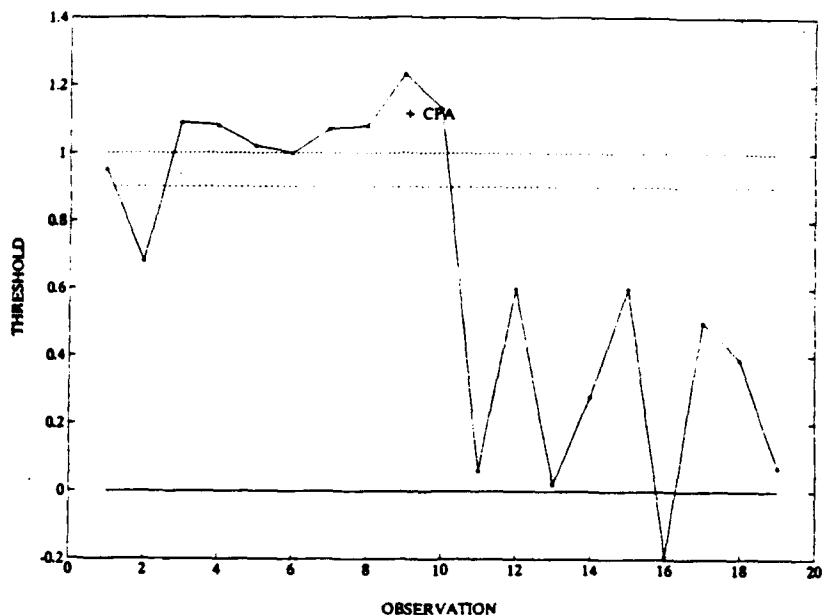


Figure A.4 (c): The activity level of target two's processing element, located in the output layer, during scenario two. This was part of the evaluation of the modified neural network (See Table A.4).

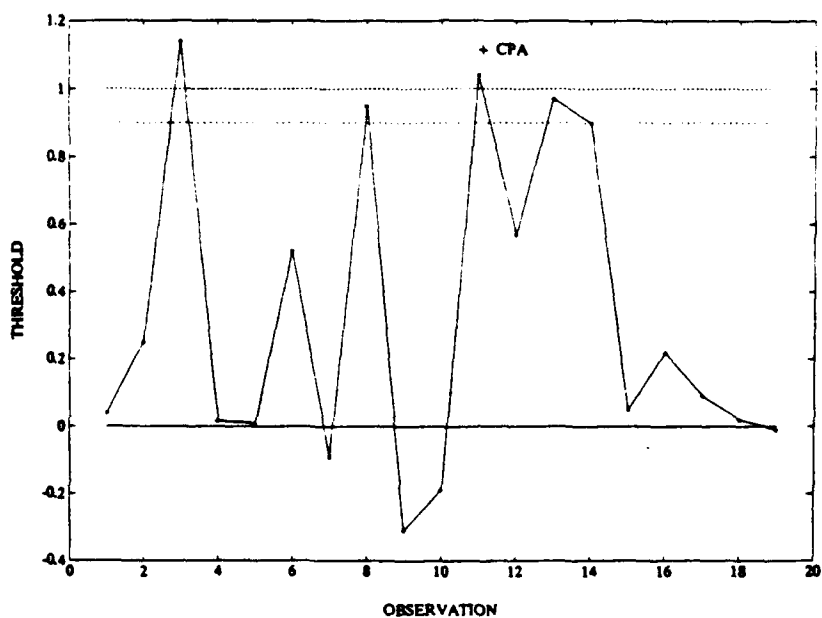


Figure A.4 (d): The activity level of target three's processing element, located in the output layer, during scenario two. This was part of the evaluation of the modified neural network (See Table A.4).

TABLE A.1: A LIST OF DATA USED IN FIGURE A.1

OBSERVATION	AMBIENT	TARGET 1	TARGET 2	TARGET 3
TIME 1	0.70	0.10	0.15	0.03
TIME 2	0.72	0.08	0.09	0.09
TIME 3	0.88	-0.08	0.04	0.20
TIME 4	1.02	-0.03	-0.10	0.03
TIME 5	-0.05	0.25	0.88	-0.08
TIME 6	-0.04	0.59	0.52	-0.04
TIME 7	0.05	0.79	0.12	0.05
TIME 8	-0.08	1.28	-0.17	-0.17
TIME 9	0.13	0.09	1.09	-0.16
TIME 10	-0.07	1.24	-0.15	-0.14
TIME 11	-0.06	0.83	0.31	-0.09
TIME 12	0.01	1.03	-0.05	0.01
TIME 13	0.00	0.95	0.00	0.04
TIME 14	1.02	0.03	-0.02	-0.01
TIME 15	0.93	-0.11	0.09	0.12
TIME 16	1.04	-0.04	-0.02	0.02
TIME 17	0.94	0.06	0.00	0.01
TIME 18	0.99	0.05	0.00	-0.03
TIME 19	1.00	0.04	-0.02	-0.03

TABLE A.2: A LIST OF DATA USED IN FIGURE A.2

OBSERVATION	AMBIENT	TARGET 1	TARGET 2	TARGET 3
TIME 1	0.50	0.04	0.16	0.25
TIME 2	0.03	-0.04	0.16	0.89
TIME 3	-0.04	-0.14	0.09	1.08
TIME 4	-0.10	0.21	0.73	0.19
TIME 5	-0.26	0.24	1.08	0.22
TIME 6	-0.05	-0.19	1.15	0.07
TIME 7	-0.25	-0.18	1.16	0.66
TIME 8	-0.28	-0.18	1.26	0.52
TIME 9	-0.13	-0.24	1.29	-0.12
TIME 10	0.03	0.38	0.70	-0.13
TIME 11	0.02	0.02	0.04	0.97
TIME 12	-0.10	-0.12	0.39	0.87
TIME 13	0.10	0.27	0.34	0.24
TIME 14	0.06	0.73	0.23	0.14
TIME 15	0.89	-0.11	0.11	0.12
TIME 16	1.04	-0.09	-0.04	0.05
TIME 17	0.70	0.08	0.12	0.07
TIME 18	0.98	0.02	0.02	-0.02
TIME 19	1.04	0.04	0.00	-0.08

TABLE A.3: A LIST OF DATA USED IN FIGURE A.3

OBSERVATION	AMBIENT	TARGET 1	TARGET 2	TARGET 3
TIME 1	0.44	0.04	0.58	0.00
TIME 2	0.69	0.01	0.30	0.03
TIME 3	0.73	-0.02	0.41	0.15
TIME 4	1.08	-0.02	0.10	0.07
TIME 5	-0.02	0.76	0.85	-0.06
TIME 6	0.34	0.58	-0.01	0.01
TIME 7	0.03	1.08	-0.06	0.12
TIME 8	0.02	1.12	-0.04	0.00
TIME 9	0.18	1.09	0.97	-0.30
TIME 10	0.00	1.12	-0.07	0.05
TIME 11	-0.02	1.11	0.75	0.01
TIME 12	-0.04	1.09	-0.15	0.77
TIME 13	0.47	0.79	-0.24	0.25
TIME 14	1.00	0.06	-0.04	-0.03
TIME 15	0.54	-0.02	0.44	0.03
TIME 16	1.05	-0.02	0.08	0.03
TIME 17	0.98	0.02	0.04	0.02
TIME 18	0.94	0.02	0.03	0.01
TIME 19	0.98	0.03	-0.14	0.03

TABLE A.4: A LIST OF DATA USED IN FIGURE A.4

OBSERVATION	AMBIENT	TARGET 1	TARGET 2	TARGET 3
TIME 1	0.17	-0.03	0.95	0.04
TIME 2	-0.03	-0.03	0.66	0.24
TIME 3	-0.03	-0.03	1.09	1.14
TIME 4	-0.05	0.25	1.08	0.02
TIME 5	-0.03	-0.07	1.02	0.01
TIME 6	0.05	-0.12	1.00	0.52
TIME 7	0.04	-0.12	1.06	-0.09
TIME 8	-0.01	0.93	1.07	0.95
TIME 9	0.16	0.81	1.21	-0.31
TIME 10	-0.04	0.02	1.11	-0.19
TIME 11	-0.10	-0.04	0.06	1.04
TIME 12	0.08	-0.12	0.58	0.58
TIME 13	0.04	-0.05	0.02	0.97
TIME 14	-0.04	0.27	0.25	0.90
TIME 15	1.07	-0.04	0.62	0.05
TIME 16	1.06	-0.03	-0.19	0.18
TIME 17	0.62	-0.01	0.50	0.09
TIME 18	0.93	-0.01	0.39	0.02
TIME 19	0.99	0.03	0.07	-0.01

APPENDIX B: MATLAB PROGRAM

```
function SIGNAL = ship(SPEED,RANGE,ASPECT,SS,SD)
%
%   The SHIP.M function generates a simulated passive sonar
%   signal resembling the acoustic signature of a surface ship.
%   The sonar signal behaves in accordance with common surface
%   ship parameters and the passive sonar equation [Ref. 2]:
%
%            $SL - TL = NL - DI + DT$ 
%
%   SL = SOURCE LEVEL (of TARGET)
%   TL = TRANSMISSION LOSS
%   NL = AMBIENT NOISE LEVEL
%   DI = DIRECTIVITY INDEX (of TRANSDUCER)
%   DT = DETECTION THRESHOLD (of TRANSDUCER)
%
%   The program takes the SPEED and ASPECT angle of a target
%   provided by the user, and calculates the total source level
%   of the target. Next, using the RANGE provided, the program
%   calculates the expected transmission losses and then modifies
%   the predetermined source level signal to reflect the losses.
%   Lastly, the program uses the sea-state level (SS) and the
%   shipping density level (SD) to find the overall ambient-noise
%   which is then added to the modified source level signal. The
%   resulting source level signal presented to the user.
%
% STEP (1) DETERMINE THE SOURCE LEVEL
%
%   First determine the noise level based on the speed of the
%   and aspect angle of the target source.
%
%   Center frequency for cavitation noise spectrum
%
%   CFREQ = 100;
%
%   Total frequency bandwidth covered
%
%   FREQ = 256;
%
%   STEP_01 = 'CALCULATE MAXIMUM SOURCE LEVEL'
%
% Calculate the maximum signal strength, MAX, (in dB).
%
%   if SPEED <= 10
%       MAX = 150 + .5*SPEED;
%   end
%
%   if SPEED > 10
%       MAX = 155 + 1.2*SPEED;
%   end
```

```

%
% DETERMINE THE CAVITATION NOISE SPECTRUM
%
STEP_02 = 'DETERMINE CAVITATION NOISE'

CAVIT_SIG = cavit(FREQ, CFREQ, RANGE);

% Determine the cavitation noise spectrum
% Find the normalized fft of the cavitation noise signal
NORMFFT = normal(fft(CAVIT_SIG));

% Adjust the amplitude of the spectrum to match source level.
% (1) Determine spreading loss in dB
SP_LOSS = 20*log(RANGE)/log(10);

% (2) Determine adjusted amplitude
ADJ_AMP = MAX - SP_LOSS - atten(CFREQ)*RANGE*1E-3;

% Calculate the adjusted cavitation noise signal
% (1) Find the modified fft of the cavitation noise spectrum
ADJ_SIG = exp(0.23*ADJ_AMP)*NORMFFT;

% (2) Extract the simulated cavitation noise signal with ifft.
TARGET_SIG = real(ifft(ADJ_SIG));

% (3) Calculate and plot the cavitation noise spectrum

clf
subplot(211)
SPECTRUM = db(abs(ADJ_SIG(512:-1:1)));
plot(TARGET_SIG(1:500)),title('CAVITATION NOISE SIGNAL')
xlabel('TIME SAMPLES'),ylabel('AMPLITUDE')

plot(SPECTRUM(1:250)),title('CAVITATION NOISE SPECTRUM')
xlabel('FREQUENCY (Hz) '), ylabel('MAGNITUDE (dB)')
pause

```

```

STEP_03 = 'CALCULATE BLADE RATE'

% DETERMINE THE BLADE RATE FREQUENCY

% Assume 5 bladed screw

BLADE = 5;

% Propeller turns per knot

TPK = .12;

% Propeller rotations per second

RPS = SPEED*TPK;

% Calculate the blade rate frequency and the harmonics

f0 = BLADE*RPS;
f1 = 2*BLADE*RPS;
f2 = 3*BLADE*RPS;
f3 = 4*BLADE*RPS;

pi = 6.28319;

% Sampling frequency (fs)

fs = 512;

% Calculate blade rate directivity based on aspect

DI_BLADE = di(25,f0,3.14,ASPECT);

% Determine the transmission loss for blade noise.

TL_A0 = SP_LOSS + atten(f0)*RANGE*1E-3;
TL_A1 = SP_LOSS + atten(f1)*RANGE*1E-3;
TL_A2 = SP_LOSS + atten(f2)*RANGE*1E-3;
TL_A3 = SP_LOSS + atten(f3)*RANGE*1E-3;

% Calculate the amplitude for each frequency.

A0 = exp(0.23*DI_BLADE*(MAX - 17.37*log(f0) - TL_A0));
A1 = exp(0.23*DI_BLADE*(MAX - 17.37*log(f1) - TL_A1));
A2 = exp(0.23*DI_BLADE*(MAX - 17.37*log(f2) - TL_A2));
A3 = exp(0.23*DI_BLADE*(MAX - 17.13*log(f3) - TL_A3));

STEP_04 = 'DETERMINE SCREW NOISE'

PHI0 = 2*pi*f0/fs; PHI1 = 2*pi*f1/fs; PHI2 = 2*pi*f2/fs; PHI3 = 2*pi*f3/f

for t = 1:fs
    SCREW(t) = A0*cos(PHI0*t)+A1*cos(PHI1*t)+A2*cos(PHI2*t)+A3*cos(PHI3*t);
end

```

```

% Combine screw blade noise with source level signal

    TARGET_SIG = TARGET_SIG + SCREW;

% DETERMINE OTHER RADIATED NOISE

%           TONE_A = Motor Bearing @ 88 Hz 185 db
%           TONE_B = Reduction Gear @ 53-57 Hz 150 dB
%           TONE_C = Flow noise @ 203 Hz 200 dB
%           TONE_D = Generator @ 60,120 180,240 Hz 190 dB(MAX)
%           TONE_E = Fuel oil Pump @ 135 Hz 160 dB

% CALCULATE ADJUSTMENT FOR DOPPLER EFFECT
%
% ASSUME SPEED OF SOUND IN WATER IS 2925 KNOTS
%

STEP_05 = 'DETERMINE DOPPLER'

DOPPLER = 2925/(2925 + SPEED*cos(ASPECT));

    TONE_A = 88*DOPPLER;
    TONEB1 = 53*DOPPLER;
    TONEB2 = 57*DOPPLER;
    TONE_C = 203*DOPPLER;
    TONE_D1 = 60*DOPPLER;
    TONE_D2 = 120*DOPPLER;
    TONE_D3 = 180*DOPPLER;
    TONE_D4 = 240*DOPPLER;
    TONE_E = 135*DOPPLER;

STEP_06 = 'CALCULATE TRANSMISSION LOSSES'

    TL_A = SP_LOSS + atten(TONE_A)*RANGE*1E-3;
    TL_B = SP_LOSS + atten(TONE_B1)*RANGE*1E-3;
    TL_C = SP_LOSS + atten(TONE_C)*RANGE*1E-3;
    TL_D1 = SP_LOSS + atten(TONE_D1)*RANGE*1E-3;
    TL_D2 = SP_LOSS + atten(TONE_D2)*RANGE*1E-3;
    TL_D3 = SP_LOSS + atten(TONE_D3)*RANGE*1E-3;
    TL_D4 = SP_LOSS + atten(TONE_D4)*RANGE*1E-3;
    TL_E = SP_LOSS + atten(TONE_E)*RANGE*1E-3;

% CALCULATE THE SIGNAL LEVEL BASED ON ASPECT

```

```

%% CALCULATE DIRECTIVITY FACTOR
%
% DIRECTIVITY = DI(SOURCE QUALITY, FREQUENCY, ASPECT, SOURCE DIRECTIVITY)

STEP_07 = 'CALCULATE DIRECTIVITY'

DI_A = di(50,88,2.57,ASPECT) + di(50,88,1.0,ASPECT);
DI_B = di(90,53,3.14,ASPECT) + di(100,53,1.57,ASPECT);
DI_C = di(25,203,3.14,ASPECT);
DI_D1 = di(25,60,1.57,ASPECT);
DI_D2 = di(25,120,1.57,ASPECT);
DI_D3 = di(25,180,1.57,ASPECT);
DI_D4 = di(25,240,1.57,ASPECT);
DI_E = di(25,135,0.8,ASPECT);

STEP_08 = 'CALCULATE THE AMPLITUDE OF EACH TONAL'

A(1) = exp(0.23*DI_A*(160 - TL_A));
A(2) = exp(0.23*DI_B*(165 - TL_B));
A(3) = exp(0.23*DI_C*(169 - TL_C));
A(4) = exp(0.23*DI_D1*(222 - 13.02*log(TONE_D1) - TL_D1));
A(5) = exp(0.23*DI_D2*(222 - 13.02*log(TONE_D2) - TL_D2));
A(6) = exp(0.23*DI_D3*(222 - 13.02*log(TONE_D3) - TL_D3));
A(7) = exp(0.23*DI_D4*(222 - 13.02*log(TONE_D4) - TL_D4));
A(8) = exp(0.23*DI_E*(158 - TL_E));

STEP_09 = 'COMBINE TONALS'

PA = 2*pi*rand(1);
PB = 2*pi*rand(1);
PC = 2*pi*rand(1);
PE = 2*pi*rand(1);

PHI = 2*pi/fs;

for t = 1:fs
    w = PHI*t;

    TONALS_01(t) = A(1)*cos(w*TONE_A+PA)+A(2)*cos(w*TONE_BI+PB)+...
        A(2)*cos(w*TONE_B2+PB)+A(3)*cos(w*TONE_C+PC);
    TONALS_02(t) = A(4)*cos(w*TONE_DI)+A(5)*cos(w*TONE_D2)+...
        A(6)*cos(w*TONE_D3)+A(7)*cos(w*TONE_D4);
    TONALS_03(t) = A(8)*cos(w*TONE_E+PE);
end

TONALS = TONALS_01 + TONALS_02 + TONALS_03;

% Find the largest amplitude

ADJ = 0;
for n = 1:8,
    if A(n) > ADJ
        ADJ = A(n);
    end
end
end

```



```

% Take the fft of the combined tonal signal and normalize it.
% Next, adjust signal to maximum amplitude.

TONFFT = fft(TONALS);
TONENORM = normal(TONFFT);
TONALS = real(ifft(ADJ*TONENORM));

% Combine the combined tonal signal with the total source level signal.
TARGET_SIG = TARGET_SIG + TONALS;

% DETERMINE THE AMBIENT NOISE LEVEL

% (1) CALCULATE THE SHIPPING DENSITY NOISE
STEP_10 = 'CALCULATE SHIPPING DENSITY'

% SD = SHIPPING DENSITY
NOISE_SIG = zeros(1:fs);

for f = 1:FREQ
    if f < 44
        SHIP_NOISE(f) = 6.51*(log(f) + 48 + SD*3 + rand(1));
    else
        SHIP_NOISE(f) = 139 + SD*3 + rand(1) - 17.37*log(f);
    end

    PHASE = 2*pi*rand(1);
    AMP = exp(0.23*SHIP_NOISE(f));
    w = 2*pi*f;

    for k = 1:fs
        NOISE_SIG(k) = NOISE_SIG(k) + AMP*cos(w*k + PHASE);
    end
end

NOISENORM = normal(fft(NOISE_SIG));
NOISE_SIG = real(ifft(exp(0.23*(74.26 + 3*SD))*NOISENORM));

```

```

% (2) CALCULATE THE SEA STATE NOISE

STEP_11 = 'CALCULATE SEA STATE NOISE'

% SS = SEA STATE
AMBIENT_SIG = zeros(1:fs);

    for f = 1:FREQ,
        if f <= 10,
            AMBIENT(f) = 78 + SS*3 + rand(1) - 6.51*log(f);
        end

        if f > 10,
            if f <= 80,
                AMBIENT(f) = 4.78*log(f) + 50 + SS*3 + rand(1);
            end
        end

        if f > 80,
            AMBIENT(f) = 109 + SS*3 + rand(1) - 8.69*log(f);
        end

        PHASE = 2*pi*rand(1);
        AMP = exp(0.23*AMBIENT(f));
        w = 2*pi*f;

        for k = 1:fs,
            AMBIENT_SIG(k) = AMBIENT_SIG(k) + AMP*cos(w*k + PHASE);
        end
    end

    AMBNORM = normal(fft(AMBIENT_SIG));
    AMBIENT_SIG = real(ifft(exp(0.23*(79 + 3*SS))*AMBNORM));

% Combine shipping noise and ambient noise

    TOTAL_NOISE = AMBIENT_SIG + NOISE_SIG;

% Combine total noise with the target source signal

    TARGET_SIG = TOTAL_NOISE + TARGET_SIG;

% Display acoustic waveform

    plot(TARGET_SIG(1:500)), title(' ACTUAL SIGNAL ')
    xlabel(' Time Samples '), ylabel(' AMPLITUDE ')

    TEMPFFT = fft(TARGET_SIG);
    SPECTRUM = db(abs(TEMPFFT(512:-1:1)));
    plot(SPECTRUM(1:250)), title(' ACTUAL SPECTRUM ')
    xlabel('FREQUENCY (Hz) '), ylabel('MAGNITUDE (dB)')
    pause

    SIGNAL = TARGET_SIG
end

```

```

function signal = cavit(FREQ,CFREQ,RANGE)
% signal = cavit(FREQ,CFREQ)
signal = zeros(1:2*FREQ);
clc
N = 2*FREQ;
theta = 6.28/N;
SP_LOSS = 20*log(RANGE)/log(10);
for f = 1:CFREQ,
    home
    STEP_02 = 'DETERMINE CAVITATION NOISE'
    percent_done = 100*f/FREQ
    p = rand(1)*6.28;
    phi = theta*f;
    M = 8.6859*log(f);
    M = .98*M + .02*M*rand(1) - SP_LOSS - atten(f)*RANGE*1E-3;
    A = exp(M*.23);

    for m = 1:N,

        frequency(m) = A*cos(phi*m+p);

    end

    signal = signal + frequency;
end

MAX = 17.3913*log(CFREQ);
for f = (CFREQ+1):FREQ,
    home
    STEP_01 = 'DETERMINE CAVITATION NOISE'
    percent_done = 100*f/FREQ
    p = rand(1)*6.28;
    phi = theta*f;
    M = MAX - 8.6859*log(f);
    M = .98*M + .02*M*rand(1) - SP_LOSS - atten(f)*RANGE*1E-3;
    A = exp(M*.23);

    for m = 1:N,

        frequency(m) = A*cos(phi*m+p);

    end

    signal = signal + frequency;
end
end

```

```

function r = DI(L,f,p,aspect)
% r = DI(L,f,p,aspect)
r = (sin(L*3.14*f/4875*sin(aspect+p))/(3.14*L*f/4875*sin(aspect+p)))^2;
end

```

```

function m = DB(sigfft);
% m = DB(sigfft)  sigfft = absolute value of your signals fft.
N = size(sigfft');
for n = 1:N/2
    m(n) = 4.34294*log(sigfft(n));
end
end

```

```

function X = normal(T)

s = size(T');
scaler = abs(T(1));
for n = 1:s
    if scaler < abs(T(n))
        scaler = abs(T(n));
    end
end
X = T/scaler;
end

```

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2. Library, Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
3. Department Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4. Professor M. Tummala, Code EC/Tu Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	4
5. Professor C. Yang, Code EC/Ya Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
6. Dr. R.N. Madan Code 1114 Office of Naval Research 800 N. Quincy Street Arlington, Virginia 22217-5000	1
7. Commanding Officer Attn: LT David F. Moore Mare Island Naval Shipyard Vallejo, California 94592-5000	1