

T-4084

THE ROLE OF TRACE ATTRIBUTES IN THE IMPLEMENTATION
OF NEURAL NETWORKS FOR SEISMIC
FIRST ARRIVAL EVALUATION

by

Elizabeth A. Elkington

ARTHUR LAKES LIBRARY
COLORADO SCHOOL OF MINES
GOLDEN, CO 80601

ProQuest Number: 10783742

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



ProQuest 10783742

Published by ProQuest LLC (2018). Copyright of the Dissertation is held by the Author.

All rights reserved.

This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.


ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

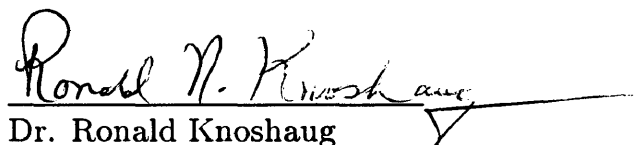
T-4084

A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Master of Science (Geophysical Engineering).

Golden, Colorado


Date 11/19/91

Signed: 
Elizabeth A. Elkington

Approved: 
Dr. Ronald Knoshaug
Thesis Advisor

Golden, Colorado

Date 19 November 1991


Dr. Phillip R. Romig
Professor and Head,
Geophysics Department

Abstract

Neural networks, with their pattern recognition capabilities, lend themselves to the problem of seismic first arrival evaluation which has a clear pattern associated with it. When these seismic first arrivals are obscured by noise or variations in amplitude and character, one must infer where to pick the first arrival. Amplitude data and analytic trace attributes are input to a backpropagation neural network and evaluated for their contribution to picking seismic first arrivals.

A sliding window procedure was used to present the data to the neural network. The performance of the network was improved via manipulation of various network parameters. Using amplitude data as input was effective in picking tomographic data which exhibited consistent waveform and little noise. The network outputs for this data were above or near the designated threshold value. These first breaks were picked within one sample of the desired picks. For refraction data, this effectiveness breaks down near the critical angle where the waveform and amplitude vary and the pattern the neural network is trying to recognize is less well defined. The introduction of multiple attributes provides considerable improvement over amplitude information

alone. In the case of refraction data, a combination of amplitude data and the three analytic trace attributes, envelope, envelope slope, and instantaneous phase, resulted in the strongest outputs. With one or two attributes used as input, the network output was scattered and did not follow the trend of the first break across the record. When more attributes were input, the network outputs followed the first break trend more closely, although output values were not as concise as in the tomographic case.

Attributes were also obtained by condensing a window of data into a single value. The values are tied to the peak value of the window, and the peak is seldom the location of the first break. These attributes provide trend evaluation but do not retain the specific local significance as does the time-sampled data. These window-sampled attributes could be used as preliminary evaluations of the data to identify the trend of the first break. The window-sampled attributes proved most effective using a multiple trace, multiple attribute input approach.

This research indicates promise for the use of neural networks to identify patterns in seismic traces and specifically for first arrival evaluation. The backpropagation neural network tested in this research picked tomographic first arrivals well but provided only marginal results in refraction data cases for offsets near critical. It is suspected that these methods would work much better well beyond the critical distance.

Table of Contents

Abstract		iii
List of Figures		vii
List of Tables		xi
Acknowledgements		xii
1 Introduction		1
2 Trace Attributes		4
2.1	Defining First Arrivals	4
2.2	The Analytic Trace and Its Attributes	7
2.3	Attributes Analysis	10
2.3.1	Time-Sampled Attributes	10
2.3.2	Window-Sampled Attributes	17
3 Neural Network Parameters		25
3.1	General Concerns	25
3.2	Parameters to Evaluate	26
3.2.1	Network Size	28
3.2.2	Learning Rate	28
3.2.3	Activation Function	29
3.2.4	Initial Weights	29
3.2.5	Output Threshold	30
3.2.6	Output Range	30
4 Results		32
4.1	Input Data	32
4.2	Paradigm	33
4.2.1	Simulated Annealing	38

4.3	Programs	39
4.3.1	Training Set Preparation	40
4.3.2	Neural Network Training	40
4.3.3	Picking Data Using the Neural Network	41
4.4	Network Size	43
4.5	Learning Rate	45
4.6	Activation Function	46
4.7	Initial Weights	47
4.8	Output Threshold	49
4.9	Output Range	52
4.10	Trace by Trace Inputs	53
4.10.1	Tomographic Data	53
4.10.2	Refraction Data	55
4.10.3	Synthetic Refraction Data - Noise Effects	57
4.10.4	Blocked Data	59
4.11	Five Trace Inputs	62
4.12	Coherent versus Noncoherent Targets	65
5	Conclusions	66
	References Cited	69
A	Neural Network Background	71
A.1	History	71
A.2	Biological Basis	72
A.3	Network Construction	73
A.4	Backpropagation Algorithm	75
A.4.1	General Overview	75
A.4.2	Training Procedure	78
A.4.3	Derivation of Weight Changes	78
B	Processing Flow	84
C	Results Listings	86
C.1	Tomographic Data	86
C.2	Refraction Data	86
C.3	Synthetic Refraction Data	95
C.4	Window-Sampled Data	96
C.5	Five Trace Input	96

List of Figures

2.1	An example of the amplitude information from field data providing a clear pattern for recognition by a neural network.	12
2.2	Amplitude Data – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.	14
2.3	Envelope – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.	15
2.4	Envelope Slope – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.	16
2.5	Instantaneous Phase Data – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.	18
2.6	Instantaneous Frequency – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.	19
2.7	Peak amplitude from synthetic refraction data contoured to illustrate the trend of this attribute.	21
2.8	Mean power level from synthetic refraction data contoured to illustrate the trend of this attribute.	23
2.9	Power ratio from synthetic refraction data contoured to illustrate the trend of this attribute.	24
3.1	The sigmoid function with output ranges from -.5 to .5.	31
4.1	Original data set of grossly simplified trace-like data to test neural networks ability to determine the first break for various rise times.	34
4.2	Tomographic data collected by undergraduate students during the 1989 geophysical field camp used for testing the neural network.	35

4.3	Refraction data collected by undergraduate students during the 1990 geophysical field camp used for testing the neural network. . . .	36
4.4	Synthetic refraction data, generated by the Cshot modelling program written by Dr. P. Docherty, used for testing the neural network.	37
4.5	Sliding window input, used for inputting data into the network, targets the center of the window for locating the first break position.	41
4.6	Example of a network with 21 nodes in the input layer, 10 nodes in the hidden layer, and 1 node in the output layer.	42
4.7	Training sets are made up of the cumulative traces in the data set selected for training.	42
4.8	Configuration of input values when blocked attributes are used. . . .	44
4.9	Each trace receives equal representation when the output threshold is .1.	50
4.10	A comparison of the neural network picks to my picks using the tomographic data set, Data Set 1. The neural network parameters output threshold, O.T., and learning rate, L.R., were varied in the first two examples. The number of presentations of the first break picks were increased in the last example.	54
4.11	Use of multiple attributes to improve the picking capabilities of the neural network compared to my picks. The data set used for training and picking is the refraction data set, Shot Point 24. . . .	56
4.12	Comparison of outputs using a single first break presentation per trace and repeated first break presentations per trace for training with the refraction data.	58
4.13	Comparison of outputs using synthetic refraction data with varying degrees of noise.	60
4.14	Comparison of outputs using blocked attribute data from Shot Point 24.	61
4.15	Input/Output scheme for a sliding window with five trace input and five outputs, one for each trace.	62
4.16	Comparison of outputs using five input traces at a time. The blocked attribute data from Shot Point 24 was used as the input data. Notice that the neural network picks, N.N. Picks, display only five trace outputs per box which differs from the My Picks box. .	64
A.1	The biological neuron after which the neurons in an artificial neural network were patterned.	74
A.2	Schematic of the forward pass through a neural network.	76
A.3	The sigmoid activation function, sometimes known as the logistic or squashing function.	77

A.4 Physical diagram of a neural network in a general case. 79

B.1 Flow chart of the processing scheme for training the neural network and picking first breaks. 85

C.1 Data Set 2 is picked with a neural network trained with Data Set 1. My picks for Data Set 2 are shown for comparison to the neural network's picks. 87

C.2 Data Set 3 is picked with a neural network trained with Data Set 1. My picks for Data Set 3 are shown for comparison to the neural network's picks. 88

C.3 Data Set 4 is picked with a neural network trained with Data Set 1. My picks for Data Set 4 are shown for comparison to the neural network's picks. 89

C.4 Data Set 5 is picked with a neural network trained with Data Set 1. My picks for Data Set 5 are shown for comparison to the neural network's picks. 90

C.5 Shot Point 11 is picked with a neural network trained with data from Shot Point 24. My picks for Shot Point 11 are shown for comparison to the neural network's picks. 91

C.6 Shot Point 23 is picked with a neural network trained with data from Shot Point 24. My picks for Shot Point 23 are shown for comparison to the neural network's picks. 92

C.7 Shot Point 35 is picked with a neural network trained with data from Shot Point 24. My picks for Shot Point 35 are shown for comparison to the neural network's picks. 93

C.8 Shot Point 47 is picked with a neural network trained with data from Shot Point 24. My picks for Shot Point 47 are shown for comparison to the neural network's picks. 94

C.9 Synthetic refraction data containing no noise is picked using a neural network trained with data having a S/N ratio of 2. My picks are included for comparison. 95

C.10 Window-sampled attributes from the synthetic refraction data set are picked using a network trained with the first ten traces of the same synthetic refraction data set. My picks for the data are given for comparison. 96

C.11 Panels 1 to 12 show the progression of picks output from a neural network implementing the five trace input method. Note the redundancy of picks as the panels step forward one trace per panel. 98

C.12 Panels 13 to 20 continue the progression of picks output from a neural network implementing the five trace input method. A total of twenty panels are output for a data set containing twenty-four traces. 99

List of Tables

4.1	Comparison of output threshold of .9 versus output threshold of .1 using tomographic Data Set 1 for training. The dashes indicate that the network did not iterate back over the data in the training set specified.	51
4.2	Comparison of output threshold of .4 versus output threshold of .1 using refraction data set, Shot Point 24, for training.	51

Acknowledgements

I would like to thank my advisor, Dr. Ronald Knoshaug, for the many hours he spent helping bring this research project to fruition. His advise and encouragement were greatly appreciated. My thanks also goes out to my committee members, Dr. Frank Hadsell and Dr. Phillip Romig, for the assistance and motivation they provided from the every beginning of this research project.

A special thanks goes out to Mr. F. David Lane, fellow graduate student and Ph.D. candidate. His initial work on analysis of seismic data using neural networks provided the springboard for this research. In addition, he offered considerable advise and guidance throughout this research effort.

I would also like to recognize the Knowledge Engineering in Exploration consortium, KEX, for its financial support, and the geophysics department for assisting me via teaching assistantships.

Chapter 1

Introduction

The interpretive aspects of geophysics frequently involve various types of pattern recognition or pattern matching. Trace editing and first break picking are a few of the fundamental processing steps which clearly involve pattern recognition. Many areas of interpretation, from well logs to seismic data, involve pattern recognition as well. The focus of this research is on evaluating key factors in the implementation of a pattern recognition technique for the evaluation of first arrivals in seismic data. The pattern recognition technique involves a backpropagation neural network.

Neural networks learn patterns in characteristic data and then are able to identify those patterns in subsequent data. One might deduce that first arrival identification offers a clear pattern which geophysicists use to pick first breaks. Thus, this sort of problem would lend itself to pattern recognition via a neural network.

The main benefits of automating the first break picking process are a savings of money and time while allowing the person performing the first break picking to do less mundane tasks. Using an artificial intelligence technique, such as neural networks, to

automate the process might yield results which better emulate how a human would pick the first breaks. This may improve acceptance of the automated method by those making the first breaks picks. Neural networks have been known to identify patterns despite considerable noise for other applications. Neural networks are also known for their consistency. Both are critical to the success of an automated first break picker. Yet many issues surround the successful implementation of a neural network. Considerable effort is involved in determining what data will best portray the pattern we are trying to find. In cases where data is simple and consistent in form, a neural network may require little more than amplitude information. When data is more complex as in refraction data, the neural network may require other attributes to define this event.

Many people have already investigated neural networks for the evaluation of seismic first breaks. However for each effort, different types of input data were used. McCormack from ARCO Research utilizes bitmaps of the data. This image data is fed into a sizable network which begins at a point in the trace and moves up until the first arrival peak is encountered [1]. Taner suggests that the analytic trace provides potential attributes for supervised learning techniques such as neural networks [2]. Veezhinathan from Amoco developed a neural network to pick first breaks by incorporating grouped attributes derived from the analytic trace for input [3].

Numerous different methods have been developed to automatically pick first

breaks. Most implement a correlation technique. Some test for consistent wave shape, continuity of the signal, and may use some predictive techniques based on previous values [4, 5]. Neural networks appear to be gaining acceptance more so than other algorithms used for the determination of first breaks because the person responsible for obtaining accurate picks selects the first break examples for training. Thus the network will learn to pick values the way the person does.

Neural networks provide a tool which could be useful in many areas of geophysics. However, the issues of what inputs to use and how to design the neural network to best identify a particular pattern are not always straightforward. This research investigates the merits of various inputs and network parameters for the implementation of a backpropagation neural network to pick first breaks in seismic data. It is hoped that information contained in this document can enhance future developments involving the use of neural networks for not only first break analysis but for various forms of seismic data analysis.

Chapter 2

Trace Attributes

Determining the most effective representation of data to present to a neural network is one of the most critical problems in the successful implementation of a neural network. Different representations work with varying degrees of success. The type of data and the way it is presented dictate the effectiveness of the neural network and its tolerance for variation. A thorough analysis of trace attributes is a critical step in determining which attributes provide us with the most definitive information about the first break.

2.1 Defining First Arrivals

When seismic first breaks are mentioned a number of positions on the seismic record come to mind. The first motion or first break is often associated with the first emergence of energy on the seismic trace. The first arrival can be considered as this location or the first occurrence of the seismic wavelet packet. Clearly ambiguity exists in these definitions. The first break pick location, which is the aim of this

research, is a point on the trace which marks the location of the first arrival. This location may be the first motion or may be a peak on the first arrival wavelet packet. It is most probable that the actual time when the first seismic energy arrives is not going to be the same as the first break pick. Thus the most important factor in picking first breaks is to remain consistent in how one chooses to pick the first break. In other words, consistently pick the same feature throughout the record. In this document when the word pick is used in conjunction with first break or first arrival, we are referring to the arbitrary location on the trace selected by a person to be the first break. If you are picking direct arrivals or refracted arrivals, these arrival types dictate to a certain extent the position and character of the pick. In addition, the source signature wavelet will dictate where on that wavelet the pick is located. Explosive sources, which give signatures resembling minimum-phase wavelets, would likely be picked at the first emergence of energy or near the first zero crossing on the data. However, Vibroseis data, which is processed to yield zero-phase wavelets, would be picked at wavelet peaks coinciding with the first break. Depending on how the first break pick is defined, the neural network must be designed to accommodate the attributes used to define this position. Perhaps a single attribute is adequate in one case and multiple attributes are required in another case to discriminate between the first arrival and other undesired events on the trace.

First arrivals are generally straightforward to pick. The first arrival event

yields a linear feature of strong amplitude and consistent slope which geophysicists are quick to recognize. However, the amplitude may not always be strong, and noise may mask a clear first arrival in any single trace. Additional attributes may be inferred at this point in an attempt to visually correlate the data across the unclear zone. Despite this loss of a clear first arrival in the data, the trained eye can in most cases identify the first arrival trend and make a reasonable estimate of a questionable first arrival.

Preference is another variable which enters into the specific first break location selected by a given person. Some people prefer to pick peaks and some prefer to pick the first zero crossing. Although the exact point at which a person chooses to pick the first break may vary from person to person, the picker will try to consistently pick the same feature on successive traces following the sloping trend of the first arrival. A study by Dr. R. Knoshaug indicates these differences in picking methodologies.

Ultimately, the most descriptive attributes need to be found. This is done by first determining what people use to analyze first breaks. Other attributes, although they are not typically used by people for this task, may be helpful in delineating the first arrival. Amplitude is often times a strong indicator and may prove to be the only attribute required in cases of high signal to noise ratio, particularly in cases of direct arrivals and tomographic data. The effectiveness of the amplitude diminishes when noise is introduced and in the case of refracted arrivals where the first arrival is

no longer highest in amplitude. In these cases a geophysicist may resort to matching waveform shape which implies the use of frequency and phase along with amplitude. Thus additional attributes may need to be incorporated to effectively separate out the first arrival.

2.2 The Analytic Trace and Its Attributes

The analytic trace, defined in Bracewell [6] and used by Taner and Sheriff [7] as a tool for interpretation, offers a number of potential attributes to help locate the first arrival. It has been selected as a source of useful attributes because it is a transformation technique which retains local significance [7]. This is important because selecting the first break as a specific sample or location on the trace requires attributes which describe local trace characteristics and not attributes which describe the trace as a whole.

The analytic trace is a complex trace resulting from the Hilbert transform of a real trace. The Hilbert transform is defined by the following equation.

$$F_{Hi}(t) = \text{P.V.} \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{f(t') dt'}{t' - t}$$

By taking the principal value of the integral, the divergence at $t = t'$ is handled.

$F_{Hi}(t)$ is then a linear functional of $f(t)$ and can be written as the convolution,

$$F_{Hi}(t) = \frac{-1}{\pi t} * f(t) .$$

The complex trace is associated with a real trace $f(t)$ in the following manner,

$$\hat{f}(t) = f(t) - iF_{Hi}(t) . \quad (2.1)$$

The Hilbert transform is the quadrature function of $f(t)$. The analytic trace represents a helix that dilates and contracts. $f(t)$ is the projection of this complex function on the real axis, and $F_{Hi}(t)$ is the projection of the analytic trace on the imaginary axis. The analytic trace contains no negative frequency components, thus the analytic trace can be obtained from $f(t)$ by suppressing the negative frequencies [6]. This suppression is achieved by multiplying the Fourier transform of $f(t)$, by the Heaviside function, $H(f)$, in the frequency domain. Through manipulation in the time domain, the analytic trace as first shown in equation (2.1) is the result. The Hilbert transform yields a function whose phase is shifted by $\pi/2$. This shift can be observed in plots of instantaneous phase shown later in Figure 2.5.

A number of attributes are derived from the analytic trace including: instantaneous phase, instantaneous frequency, and from its envelope, mean power, power ratio and envelope slope [2, 6, 7]. Instantaneous phase is a promising attribute which

becomes coherent with the onset of the seismic source signal. This is calculated in the following manner,

$$\theta(t) = \tan^{-1} \frac{F_{Hi}(t)}{f(t)} .$$

Instantaneous frequency, which is the rate of change of instantaneous phase, is expressed as[8],

$$\omega(t) = \frac{d\theta(t)}{dt}$$

$$\omega(t) = \frac{2}{\Delta t} \text{Im} \left[\frac{\hat{f}(t) - \hat{f}(t - \Delta t)}{\hat{f}(t) + \hat{f}(t - \Delta t)} \right] .$$

A number of attributes are derived from the envelope of the analytic trace. The envelope is calculated in the following manner,

$$E(t) = \sqrt{f^2(t) + F_{Hi}^2(t)} .$$

Envelope slope, $ES(t)$, is the rate of change in the envelope of the analytic trace and is expressed as,

$$ES(t) = \frac{\Delta [E(t)]}{\Delta t} .$$

Mean power level, $MPL(t)$ in equation (2.2), and power ratio, $PR(t)$ in equation (2.3), are attributes which summarize the data in groups. This is done by analyzing five sample windows centered around the maximum value of a wavelet peak. These attributes were illustrated in work conducted by Amoco where they

grouped data by peaks. In their work, researchers from Amoco also used envelope slope and peak amplitude as group attributes [3]. The peak amplitude was simply the maximum value of the wavelet peak.

$$MPL(t) = \frac{1}{5} \sum_{t-2}^{t+2} E^2(t) \Delta t \quad (2.2)$$

$$PR(t) = \frac{MPL(t+2)}{MPL(t-2)} \quad (2.3)$$

2.3 Attributes Analysis

The attributes mentioned so far can be divided into two groups based on how they are input into the neural network. Attributes in the first group relate to individual time samples. Second group attributes look at a collection of time samples and attach a single value to that collection. The idea behind looking at these groupings is to see if condensing the information helps the neural network recognize specific features faster.

2.3.1 Time-Sampled Attributes

The first group includes the following: amplitude data and analytic trace attributes such as envelope, instantaneous phase, frequency, and envelope slope. The hope is that a neural network can effectively interpret features with a minimum of preprocessing. Amplitude data would ultimately be the simplest means of avoiding

upfront processing time of calculating transforms or special groupings. Amplitude data appears to be effective in low noise situations and particular cases such as tomographic data where the first arrival is the highest amplitude event and is not contaminated by other wave types. In the case of refraction data where data gets more complicated, amplitude data does not contain enough information to yield a generalized solution. An evaluation of each of the attributes in this group is contained in this section.

Field Data – Amplitude

Clearly the simplest form of data input is the amplitude of field data. There is little if any preprocessing required to prepare it for entry into a neural network. This preprocessing generally involves gaining the data. Amplitude information from the field data and the positioning of these amplitude values along the trace are the information available for input. With simple data cases where noise levels are low, waveforms consistent, and the amplitude of the first arrival high, amplitude data may contain sufficient information for a neural network to recognize the first arrival. Figure 2.1 shows examples of trace data which appear to be adequately represented by amplitude alone. The first arrivals in Figure 2.1 are very uniform. Often the wave shapes change more rapidly. This can be phase shifts induced by encountering inhomogeneities. Also the data undergoes amplitude and phase changes as it approaches the critical angle and head waves appear. In these cases, amplitude does not provide

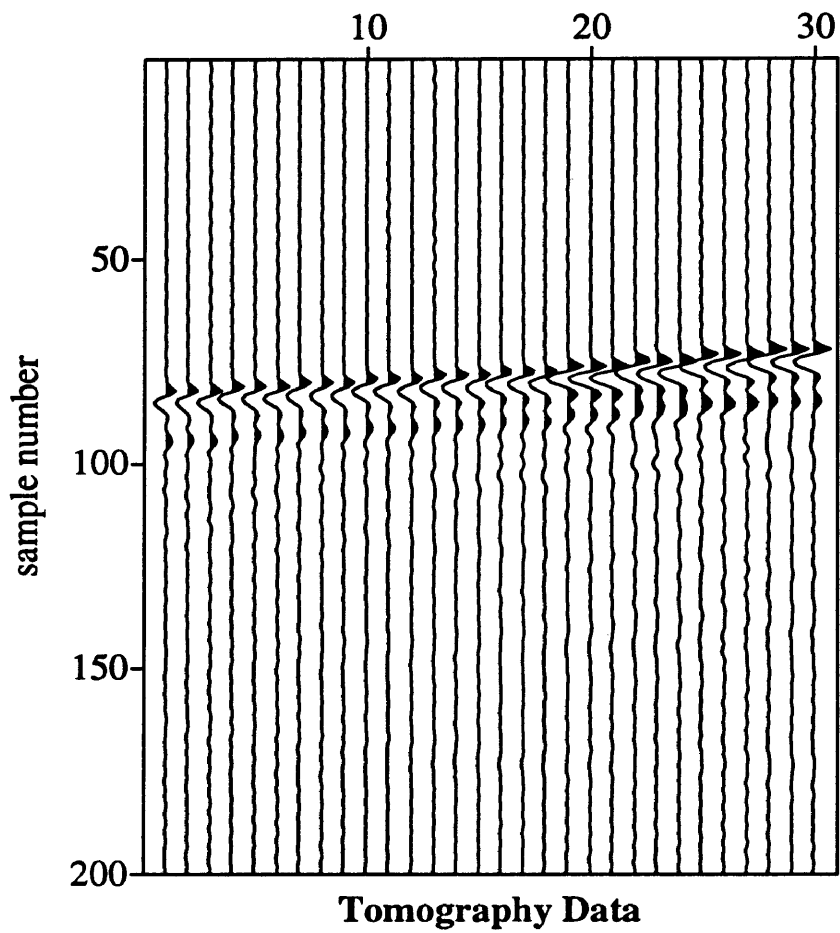


Figure 2.1: An example of the amplitude information from field data providing a clear pattern for recognition by a neural network.

a clear pattern, thus the network may encounter difficulty interpreting this data and data if there are significant levels of noise. Figure 2.2 illustrates these features.

Envelope and Envelope Slope

The envelope and envelope slope of the analytic trace are two separate attributes which both describe the envelope of the data with differing levels of success. Envelope emphasizes the original amplitude data provided in non-ideal cases over a broader range of samples. The envelope slope, derived from the envelope, appears to give a slightly more concise output than the envelope. Looking at these two attributes, envelope appears to retain more of its original character when noise is added. While the envelope slope appears to break down considerably with noise from a visual standpoint. Yet envelope slope converges in fewer iterations than envelope when tested as input to a neural network. Examples of envelope and envelope slope can be seen in the refraction data in Figures 2.3 and 2.4 respectively.

Instantaneous Phase

Instantaneous phase, another analytic trace attribute, provides promising results for cases of lower signal to noise levels and for varying waveform. Although the sample at which the first break occurs might not be clearly evident from this data, the trends provided consistently follow the sloping trend of the first arrival. This consistency is an important feature to look for in attributes describing a specific pattern.

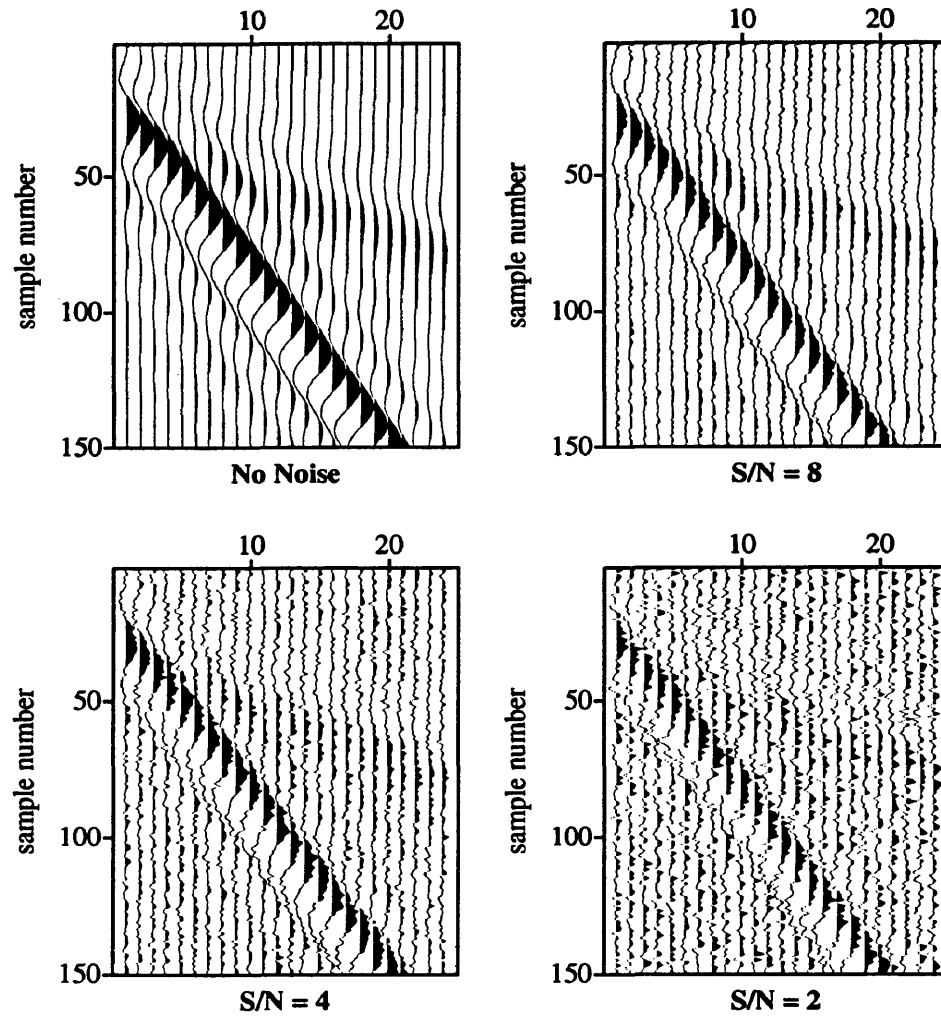


Figure 2.2: Amplitude Data – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.

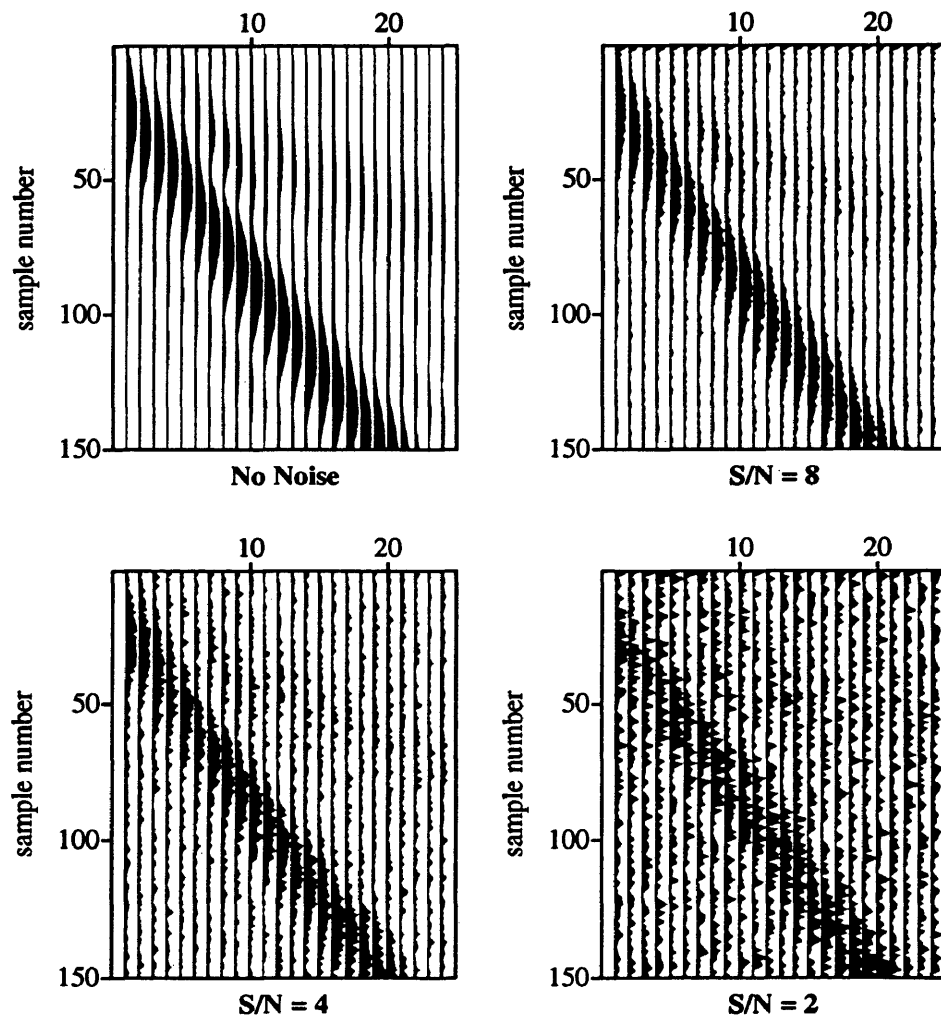


Figure 2.3: Envelope – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.

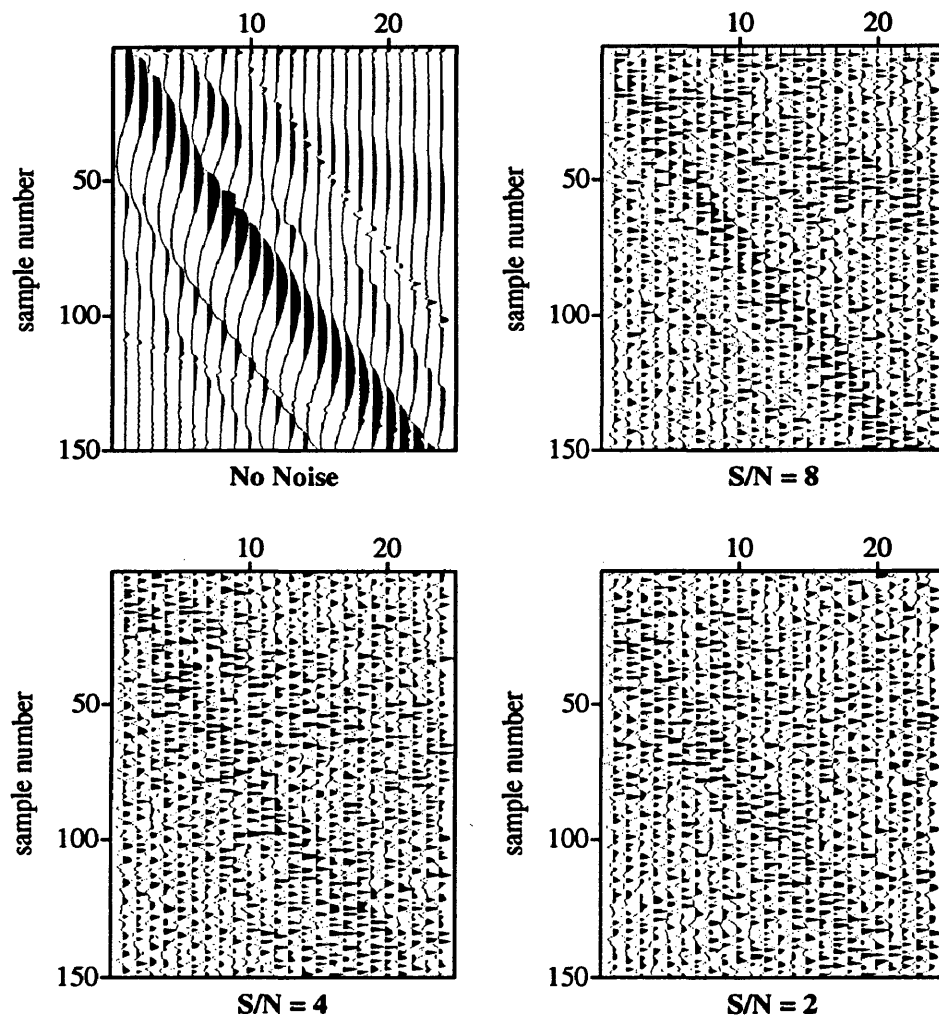


Figure 2.4: Envelope Slope – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.

If features in data are dynamic these will tend to confuse the network rather than help it learn. Illustrations of instantaneous phase and how it appears to maintain its character in non-ideal data cases can be seen in the refraction data examples in Figure 2.5.

Instantaneous Frequency

Instantaneous frequency, the rate of change of the instantaneous phase, is another attribute investigated for use in cases of lower data quality. It appears to contain some features indicating first break locations. However, these features are not consistent in occurrence and shape, thus making it difficult to train a network on this data. Although in some cases it provides promising results, its inconsistency makes its performance highly unpredictable. Examples of this attribute can be viewed in Figure 2.6.

2.3.2 Window-Sampled Attributes

In an attempt to reduce the amounts of data being perused by the neural network, a second group of attributes, blocked attributes, was investigated. This group is organized to extract the important information from a window of trace data and yield one attribute which might contain more information than the individual data points. The hope is that a fraction of the input data could be used to pick the first arrival, thus speeding up the training process. The down side is the increased

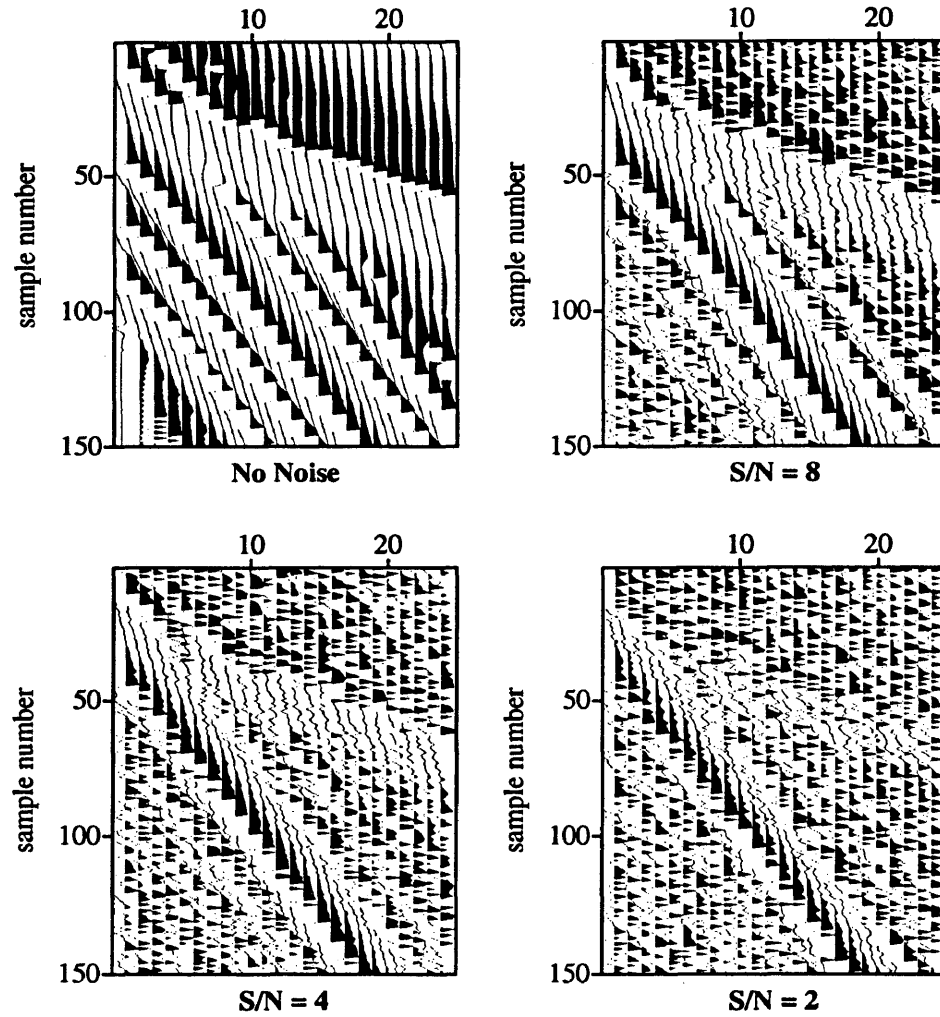


Figure 2.5: Instantaneous Phase Data – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.

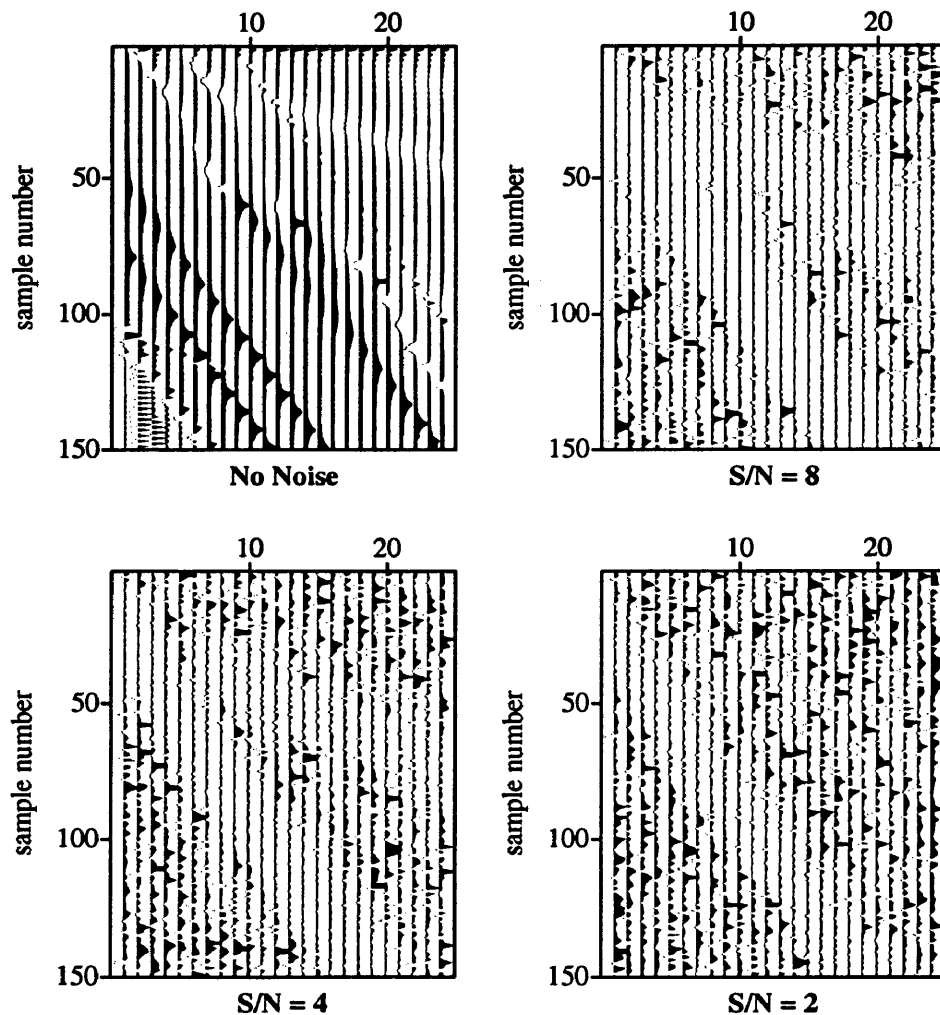


Figure 2.6: Instantaneous Frequency – Synthetic refraction data with varying degrees of noise to show the change of the attribute due to noise.

preprocessing involved in subdividing the traces and assigning these values to the various parts of the trace. Also, since these attributes are based on wavelet peaks, clearly zero-phase data which has not undergone a phase shift would yield the best results from this method. If the actual time of the first arrival is not at the peak and is varying somewhat, this method will at best narrow the investigation to a single window of interest which might then be further investigated using time sampled attributes.

Peak Amplitude

Peak amplitude appears to be the most stable attribute of the blocked attributes that were evaluated. It provided convergence of the network on its own. However, as will be indicated in the results, a combination of blocked attributes appears to be the most effective input. Notice in Figure 2.7 that the most dominant feature is from the direct arrival, trending from window 1 on the left to window 6 on the right, and the refracted arrival, trending from window 1 on the left to window 3 on the right, gives a relatively small response in comparison. Particularly in the first traces, a large amount of interference is taking place between the two wave types.

Mean Power Level and Power Ratio

Mean power level and power ratio, which is a ratio of adjacent mean power levels, provide good convergent characteristics in combination with peak amplitude.

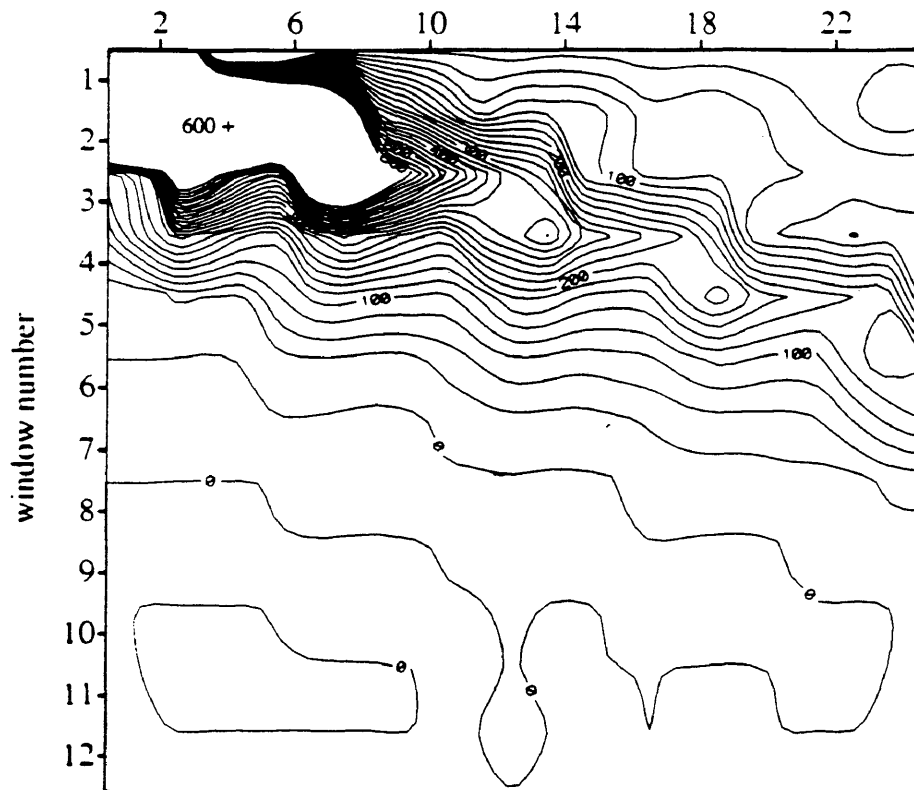


Figure 2.7: Peak amplitude from synthetic refraction data contoured to illustrate the trend of this attribute.

On their own they may or may not converge for a given training set. Figures 2.8 and 2.9 show the trends in mean power level and power ratio respectively.

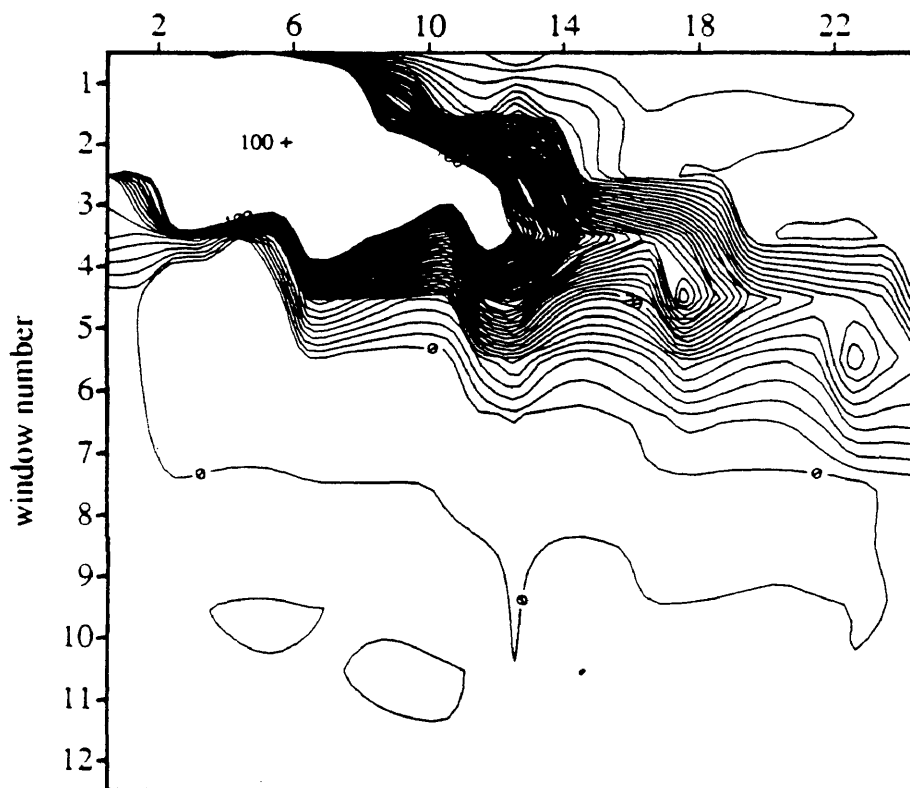


Figure 2.8: Mean power level from synthetic refraction data contoured to illustrate the trend of this attribute.

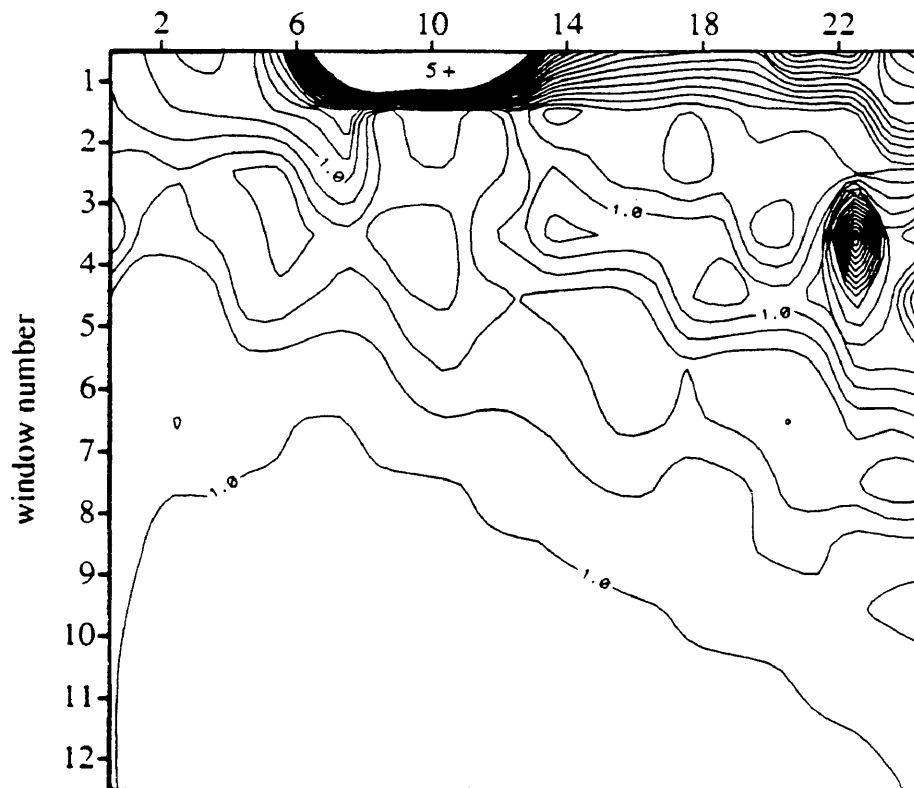


Figure 2.9: Power ratio from synthetic refraction data contoured to illustrate the trend of this attribute.

Chapter 3

Neural Network Parameters

With the understanding that some readers may have a working knowledge of neural networks, the definitions and terms surrounding neural networks and the backpropagation technique have been relegated to Appendix A. If terminology used in this chapter is unfamiliar, it may well be defined in this appendix.

3.1 General Concerns

When implementing a neural network, there are numerous choices to make regarding algorithms and parameters. Once an algorithm is selected, it is important to determine which parameters need to be evaluated, which are unique to the algorithm, and which are inherent to neural networks in general. Network parameters factor into the network's ability to converge as well as its ability to generalize. Some parameters promote fast and stable convergence. Other parameters allow the network to learn the pattern better by helping present the data in a way that the network gains the general essence of the pattern, so subsequent picking yields correct picks. When parameters

are evaluated, it is important to assess them from both of these standpoints. It could be that a network that is slow to converge is better able to generalize and thus will fare better at correctly picking new data introduced to it. Evidence of this is given in the results. There are also instances of this not being the case.

The algorithm being evaluated is the backpropagation algorithm. It is the most widely used network at this time. Because of its ability to handle multiple layers, the number of layers in the network can be expanded to accommodate whatever level of complexity the first break problem poses. The backpropagation technique allows the network structure to be altered freely, thus testing of parameters can take place without rewriting considerable amounts of code. The hope is that important parameters can be implemented in such a way that the network can be flexible enough to handle most variations in seismic data. Simulated annealing was also briefly investigated as a means to minimize the network error. The reasoning behind this was to obtain an algorithm which would avoid local minima if any were visited on the path to minimum error. The results of this study are described in Section 4.2.1 of Chapter 4.

3.2 Parameters to Evaluate

The network parameters which are evaluated in this research can be placed into two categories. Those related to the method and type of data input, which has

in part been discussed in Chapter 2, and those related to the network itself.

The first evaluation made on the input data relates to defining the pattern which is targeted for detection. A chapter has been dedicated to determining how to best see this pattern. However, input of data to the network so the pattern is detectable still poses a problem. The content of the training set needs to be determined. Testing will determine whether all the attributes should be input or just a select group of attributes should be used. The number of training pairs presented and how they are presented to yield a satisfactory result also needs to be evaluated.

With the input layer defined, the parameters affecting the outcome of the network need to be evaluated. Most of these parameters are defined by the backpropagation algorithm. The parameters are then varied to suit the problem at hand. This list of parameters includes,

1. network size
2. learning rate
3. activation function
4. initial weights
5. output threshold
6. output ranges

An explanation of each parameter and how it will be evaluated is contained in the following sections.

3.2.1 Network Size

The network size has an effect on the ability of the network to converge. More complicated problems require more layers and more nodes. Yet, the larger the network the more computations are required. Also, too many nodes in the hidden layer can result in the network memorizing a training set as opposed to generalizing from it. Thus it is desirable to maintain a network size that is large enough to evaluate the problem while keeping it small so it can generalize and keep computational requirements down.

3.2.2 Learning Rate

The learning rate dictates how much of the calculated error of the network should be backpropagated. By allowing too much error to be used in the backpropagation process when a high value for the learning rate is selected, the network may take very large steps causing it to enter a local minimum, or it may saturate the weights making them too large to provide any movement in the output essentially causing network paralysis. Small values cause the network to proceed very slowly. Values can range from .01 to 1. The optimum value is one that allows for convergence while promoting a faster rate of convergence. If the neural network can converge successfully, a value of 1 would be ideal. However in cases where convergence is not a steady descent, smaller values may be necessary for the network to converge.

3.2.3 Activation Function

The activation function modifies the summation at each node to produce an output signal called *OUT*. The sigmoid function is the most commonly used. It has the characteristic of forcing the *NET* value, the summation of the product of weights and inputs at each node, to a range between 0 and 1. This function can be altered slightly in an attempt to fine tune the network. Hyperbolic tangent is another activation function which can be used, however its use is not as common. The main requirement for an activation function is to be differentiable everywhere. If you need multiple layer computing power, the function also needs to be nonlinear.

3.2.4 Initial Weights

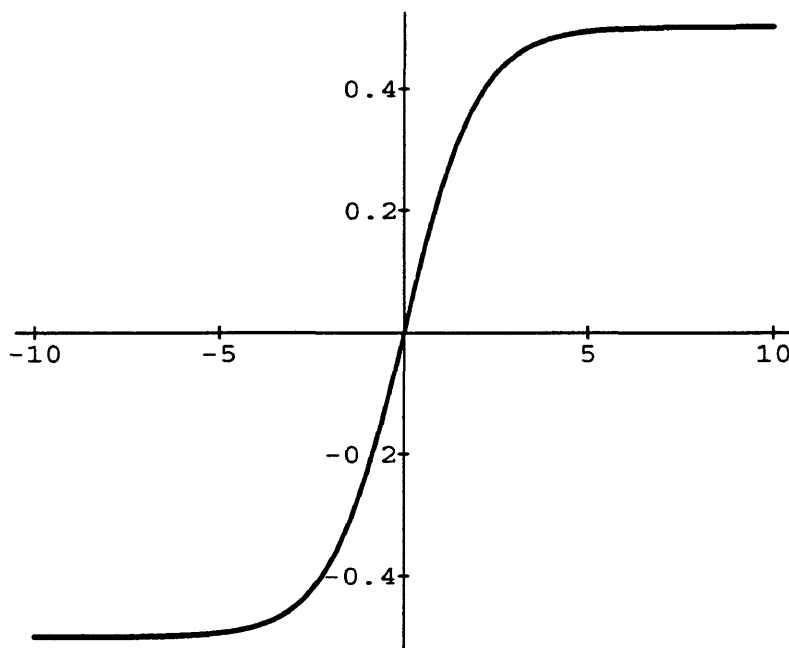
The initial weights can have an effect on the outcome of the network because these weights dictate the starting point on the error surface. Depending on whether local minima are encountered on the descent of the error path, the starting location can impact the network's ability to converge. Thus the initial starting point may be somewhat critical. In most instances, by starting with small random weights, the network's performance should not depend greatly on the weights. Large weights may tend to yield high output values which very little change in the network's output thus causing network paralysis.

3.2.5 Output Threshold

The output threshold is the value the output of the network is required to obtain to be considered trained. Since the maximum output for a sigmoid function is 1, the output threshold may be a value of .8 or .9 for a maximum acceptable error of .2 or .1 respectively. This value can also be used to limit the emphasis the network puts on any one training pair. By keeping the threshold lower at a value of .5 for example, the network learns the training sets half way. The last training set, which contains all the training examples, is allowed to proceed to the threshold of .9. This may prevent undue emphasis being placed on the first traces in the training set. This may assist in providing a more generalized solution and prevent memorizing tendencies particularly on training pairs which are presented more frequently than others.

3.2.6 Output Range

Output ranges can also effect the network's ability to converge by shifting the output range of the sigmoid function down by .5 so that when *NET* equals zero *OUT* also equals zero. The possibility of negative *OUT* values are introduced. This is coupled with a shift in input values to the same range of -.5 to .5. It would be mere speculation to infer what the results of these shifts would be, other than to say that now both the weights and the output values can induce outputs from the negative half of the shifted sigmoid. With the range 0 to 1, only positive values of *OUT* are



$$OUT = 1 / (1 + e^{-NET}) - .5$$

Figure 3.1: The sigmoid function with output ranges from -.5 to .5.

induced. This means the weights dictate the use of the portion of the sigmoid below output values of .5. Figure 3.1 illustrates this shift for the sigmoid function.

Chapter 4

Results

The success or failure of a neural network is not black and white but frequently hits the grey area between the two. Both the type of inputs used in the network and the parameters associated with a given architecture effect the results obtained from a neural network. In some cases the results are very decisive. In others, the results are less decisive, giving evidence of slight trends. Defining the success of the network is therefore difficult to do. The results obtained from this research will be presented in a manner which describes the outcomes but does not specify the result as a success or failure except in cases where failure is clearly evident. Since the reader may hold a different viewpoint on success, the results will be shown and it will be left to the reader to determine if the results qualify for their measure of success.

4.1 Input Data

The problem at hand is to determine whether or not a neural network has the capability to pick first arrivals in seismic data. Initial traces were made of simple 0 to

1 values mimicking the rising slope of the onset of seismic energy as seen in Figure 4.1. Clearly these are unacceptable examples of an actual trace, but it provided a means to determine if such a pattern could be evaluated using neural networks. If so, then the potential existed to evaluate more complicated data sets. Real data was then evaluated. It consisted of tomographic data collected in the field. This data was quite clear and simple in character (see Figure 4.2). Venturing to more complicated data, field data obtained from refraction surveys were evaluated. Now the arrivals are no longer the highest amplitude event on the record and are often subject to phase shifts when going from the direct arrival to the refracted arrival (see Figure 4.3). Synthetic data, Figure 4.4, was then used as a reference for understanding the exact location of the first arrival and how the location related to the various attributes investigated in Chapter 2. This data was obtained from a two-layer model with velocities of 3000 feet per second and 11000 feet per second respectively. The first layer had a thickness of 15 feet and the second layer was considered a halfspace. Direct, head, and reflected waves were modelled.

4.2 Paradigm

Two paradigms were evaluated. The backpropagation algorithm was chosen as the paradigm for thorough evaluation of the first break problem. The simulated annealing algorithm was tested for its capabilities. Simulated annealing was evaluated

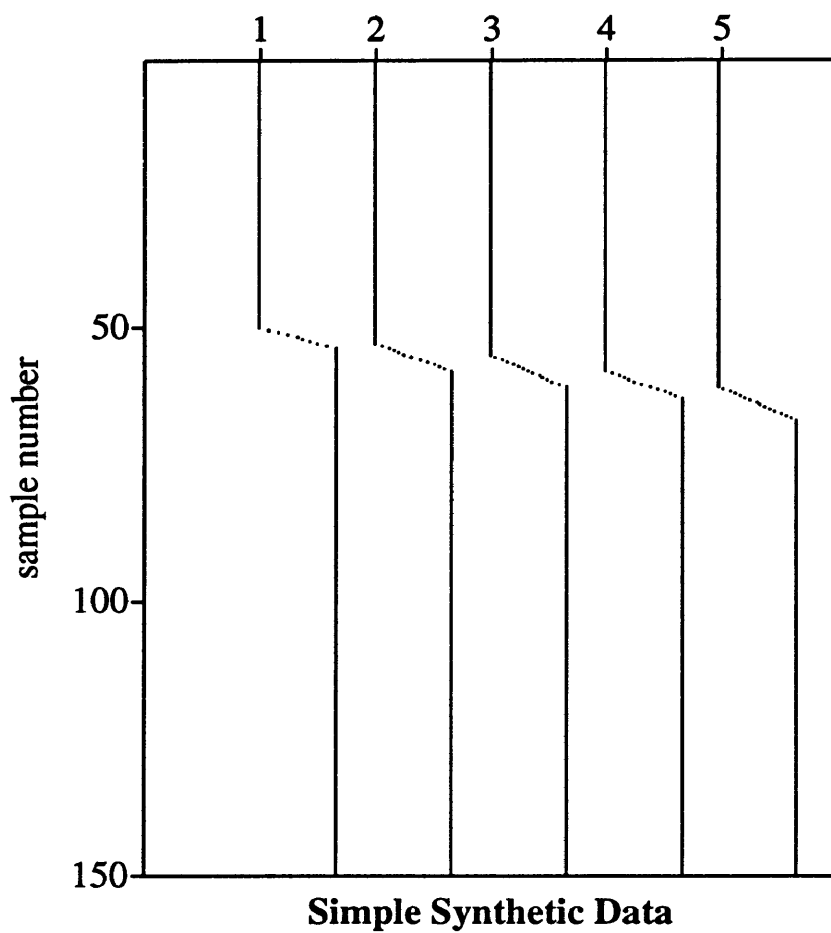


Figure 4.1: Original data set of grossly simplified trace-like data to test neural networks ability to determine the first break for various rise times.

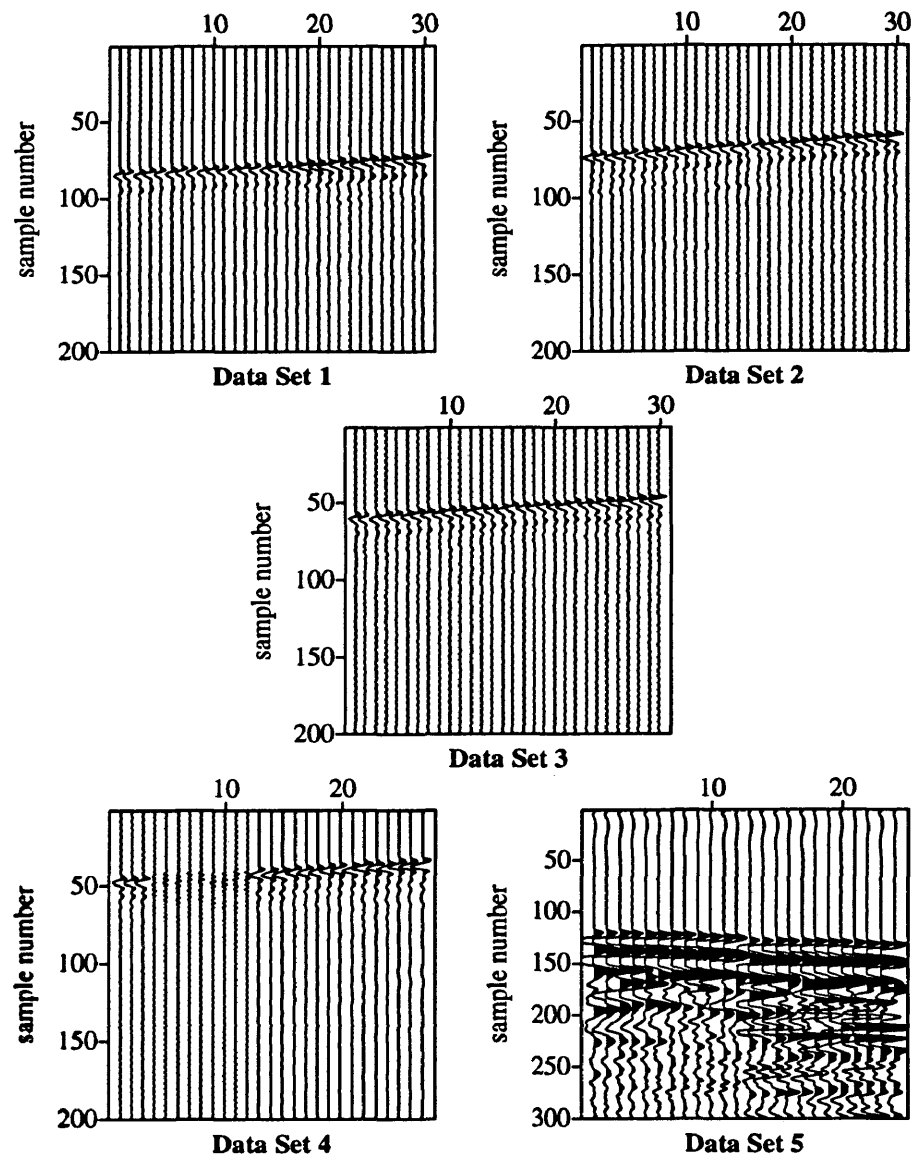


Figure 4.2: Tomographic data collected by undergraduate students during the 1989 geophysical field camp used for testing the neural network.

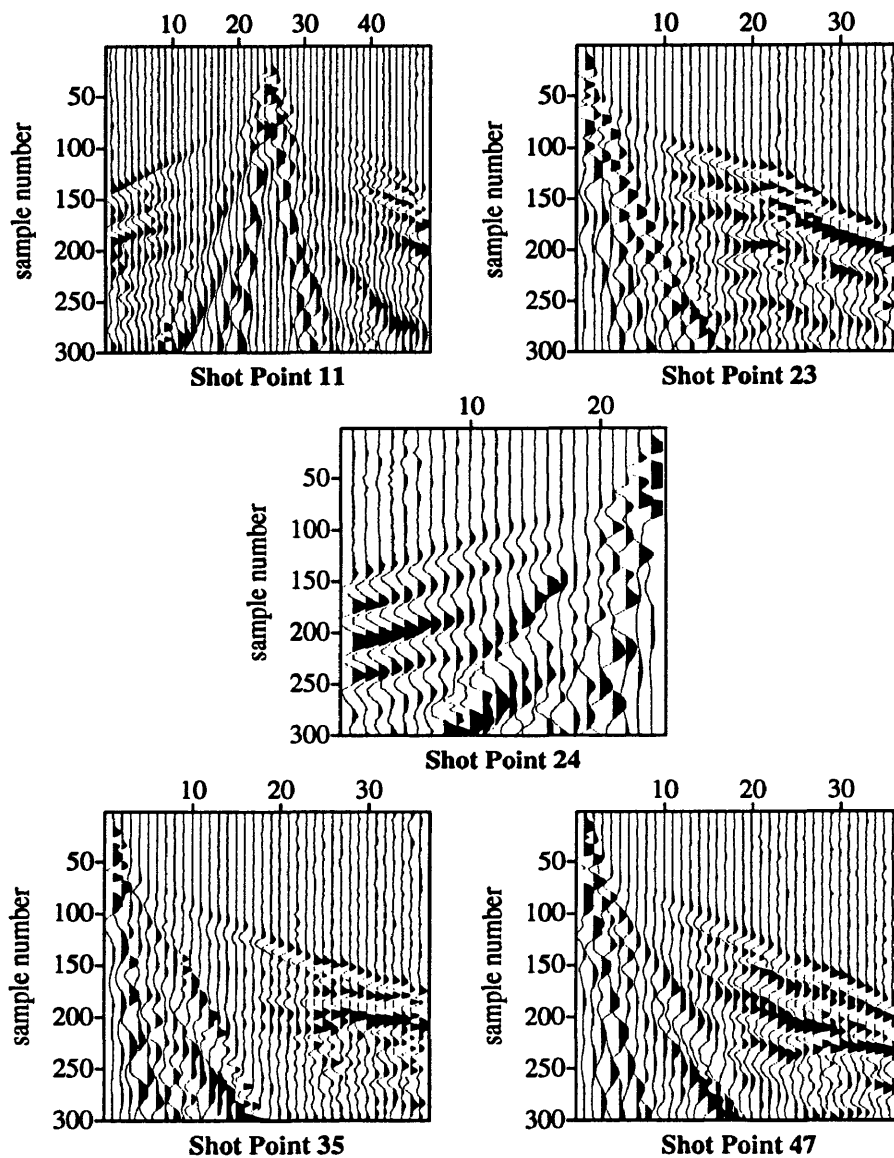


Figure 4.3: Refraction data collected by undergraduate students during the 1990 geophysical field camp used for testing the neural network.

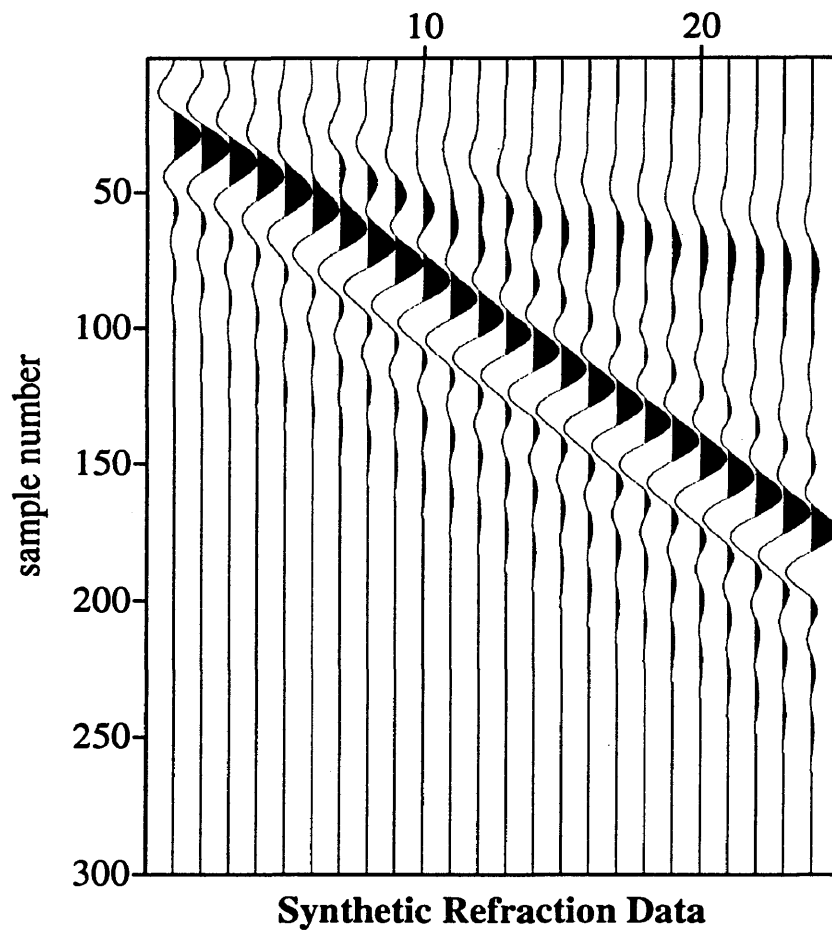


Figure 4.4: Synthetic refraction data, generated by the Cshot modelling program written by Dr. P. Docherty, used for testing the neural network.

as an alternative to backpropagation in the event that local minima problems were found.

4.2.1 Simulated Annealing

Simulated annealing is an optimization technique based on the analogy of the annealing of metals. By heating a metal to a high temperature, perhaps melting, and then cooling it on a controlled schedule, the metal can reach a minimum energy state which minimizes the internal stresses and hence increases the strength of the manufactured object. Thus the idea is that with the proper cooling schedule a global minimum of internal energy is obtained. We also use this concept to minimize the output error of the network. The probability of accepting a weight change to minimize the error is based on the Boltzman distribution. A temperature variable is introduced which is proportional to the reciprocal logarithm of time. More rapid training can be obtained from the simulated annealing method by substituting the Cauchy distribution for the Boltzman distribution. Then the temperature is proportional to the inverse of a linear function of time. This has proven to be much faster. Thus as time steps increase, the temperature decreases. Larger change in the weights is allowed to take place early for high temperatures. The simulated annealing process allows all change which reduces error and in some cases allows increase in error, analogous to jumps to higher energy states in the annealing of metals, to help escape local minima. The amount of weight change and acceptance of higher error states decreases

as the temperature decreases. Each weight is evaluated and changed based on these annealing criteria [9, 10, 11].

Simulated annealing was not selected for further evaluation because backpropagation provides a more straightforward implementation and local minima did not appear to pose severe limitations to the evaluation of first breaks. Simulated annealing required trial and error evaluation of the cooling schedule temperature parameter, and the cpu requirements were high because each weight in the network had to be evaluated for each decrease in temperature. Since local minima did not appear to be a problem for backpropagation, the cpu time and complicated parameter selection of simulated annealing caused me to drop this approach. However, it is worth noting that simulated annealing, when properly parameterized, can be extremely effective for optimization. It could be an effective paradigm if the necessary parameters for implementation were more clearly defined and predictable. For simple data, where an appropriate cooling schedule and acceptance criteria were implemented, the network was able to train. It was a very time consuming process however.

4.3 Programs

All programs, with the exception of the simulated annealing program discussed in the previous section, adopt the backpropagation algorithm for determination of proper weights of the neural network during training. The processing sequence has

been designed such that those parts not directly related to training are kept separate so only the program which performs the training needs to be changed for evaluating the effects on training. The sequence has three parts.

4.3.1 Training Set Preparation

The first program prepares the data for input into the network's training cycle. The data is normalized and traces truncated to assess only the necessary portions of the trace. In addition, the location of the first arrivals on each trace in the training set are specified.

4.3.2 Neural Network Training

The second program accepts data from the previous program. This program contains the training algorithm and will generate the weight set to be used for subsequent picking. Network size, learning rate, and output threshold are all values which can be assigned with each running of the program via prompts. The first layer size is dictated by the size of the window of data being input. In cases of time sampled data the first layer is defined as approximately one period of the dominant wavelet. The output is a single value telling whether or not the window is centered on the desired first break pick (see Figure 4.5). The subsequent network may look like the one shown in Figure 4.6 for data with a dominant wavelet with a length of 21 samples. The window is moved down the entire trace and output error is evaluated and back-

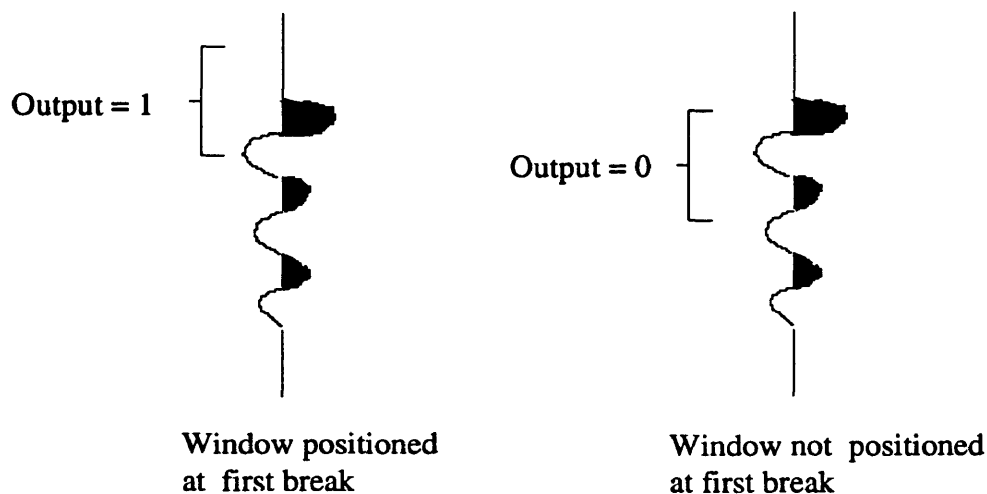


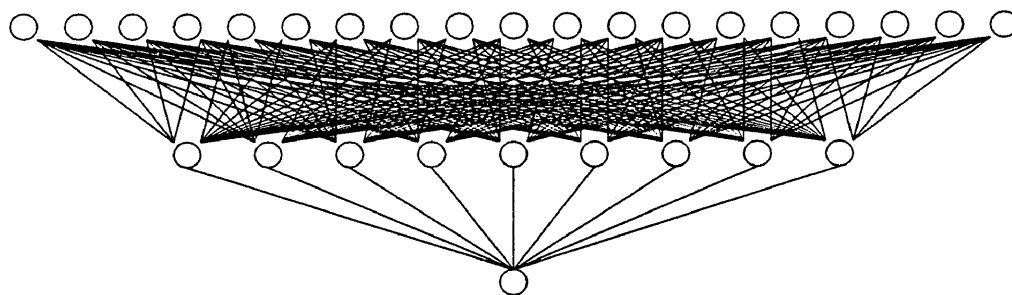
Figure 4.5: Sliding window input, used for inputting data into the network, targets the center of the window for locating the first break position.

propagated at each position. This is done for all traces in the training set. Training is done by evaluating the first trace, then traces one and two, and then one, two, and three and so on (see Figure 4.7). In the section on output threshold, this value can be changed to allow each trace to receive equal representation as well.

Once a minimum error has been obtained for the entire training set, the network is considered trained, and the weight set established in the training session is used to pick subsequent data of like character in a third program.

4.3.3 Picking Data Using the Neural Network

The third program conducts a feedforward pass through the network using new data. The resultant from the network is the first break picks. A diagram of the



Structure of a 21,10,1 Neural Network

Figure 4.6: Example of a network with 21 nodes in the input layer, 10 nodes in the hidden layer, and 1 node in the output layer.

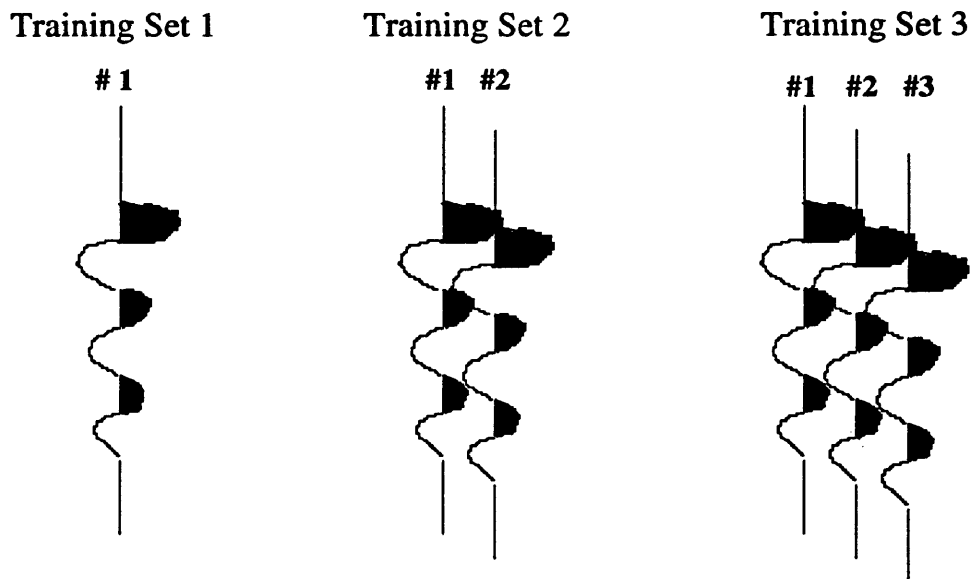


Figure 4.7: Training sets are made up of the cumulative traces in the data set selected for training.

processing steps can be found in Appendix B.

Initially the training phase accepted only one type of time sampled data for a given session. Variations to the training program include allowing for multiple attributes , multiple traces, analysis of coherent versus noncoherent data, and shift of the output range to -0.5 to 0.5 . From Chapter 3 a number of network parameters were specified for evaluation. These parameters have been incorporated via prompts or recoding into some or all of the above programs and are evaluated in the following sections.

4.4 Network Size

The size of the network, which relates to the number of nodes and layers, does impact the convergence of the network. The number of layers required is dictated by the complexity of the data. It is desirable to run the network with the minimum number of layers possible for convergence because the number of computations increase dramatically with the introduction of additional layers. Data such as the tomographic data is simple enough, the network requires two layers in most cases. However, three layers assure convergence for all the data sets in the tomographic group. Three layers are adequate for most of the refraction data cases with the exception of some of the individual instantaneous attributes. Four layers were needed for the network to converge when using envelope, envelope slope, and instantaneous frequency as single

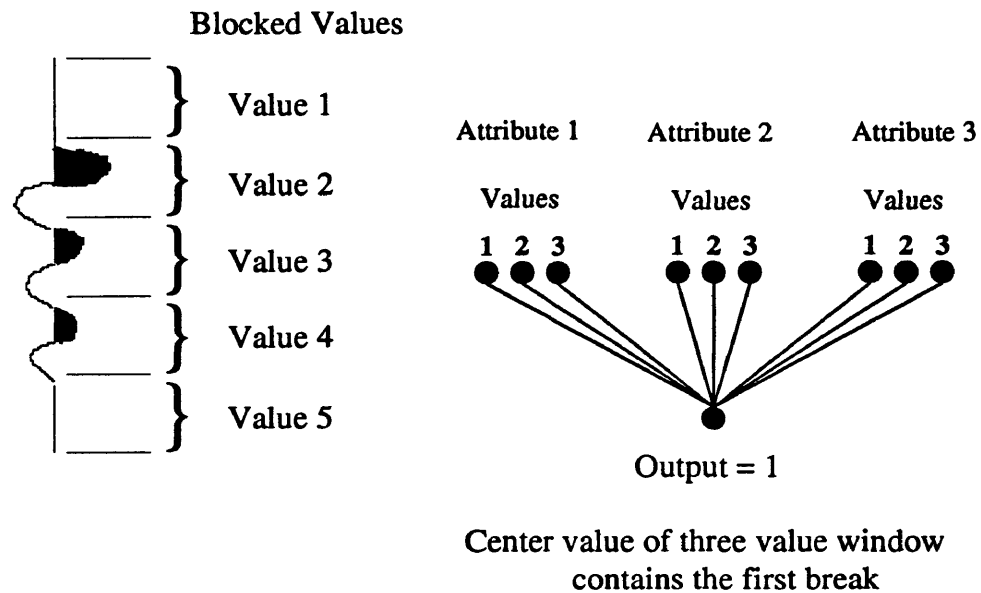


Figure 4.8: Configuration of input values when blocked attributes are used.

attribute inputs. However when some of these same attributes were used in concert such as phase and envelope slope or phase, envelope slope, and raw data, a three layer network was able to converge. The number of nodes in the input layer is primarily dictated by the form of the input data. In one case, a moving window of data is used that is approximately the length of the dominant wavelet or slightly larger. Thus, if the period is defined by 33 samples, the first layer has 33 nodes. For blocked data, a single value defines the attribute for a given window. A single input value did not define these attributes adequately. Thus three values were used as inputs and the middle value of the three would target the window containing the first arrival (see

Figure 4.8).

Convergence dictates the minimum number of nodes in the hidden layers. By testing, one finds that too few nodes cause excessive iterations or nonconvergence. Too many nodes either offer little change in convergence rate or may increase the number of iterations, and training becomes more memorization of the training set than generalization from it. In addition, the larger the number of nodes, the more computations are required. By trial and error it is possible to keep the number of nodes to a minimum. However, the combination of weights and size of hidden layer can cause significant fluctuations in the number of iterations needed. Thus when first testing size parameters, more than one hidden layer size should be evaluated before conclusions on the network's ability to converge are drawn. The output layer is dictated by the input data format. If a single trace is input there will be a single node in the output layer. If a five trace sequence is evaluated then there will be five outputs corresponding to the five traces. The input and output layers are dictated by the problem specifically and the hidden layers provide for most of the variations in network size.

4.5 Learning Rate

The learning rate affects the rate of convergence and can affect the stability of the network in some cases. For data with a consistent descent in error while training,

the rate of convergence varies directly with the learning rate. A learning rate of 1.0 can be used and the network will converge at the fastest rate possible. In some instances however, convergence is not a steady descent, but displays a somewhat erratic progress toward the solution. By decreasing the learning rate, this behavior seems to become less erratic. However, this situation has proven difficult to repeat, thus to avoid unstable situations in new training environments a learning rate of .5 to .6 was adopted. This generally will overcome the irregularities and still provide a relatively high speed of convergence.

Variable learning rates were also evaluated. Learning rates were allowed to change using values equal to the maximum error encountered at each training set presentation. Thus if the maximum error was .88, the learning rate was .88. The learning rate was also varied as 1 minus the maximum error. Here for an error of .88 the learning rate would be .12. These variable learning rates generally induced adverse effects. The convergence frequently became unstable and could jump from an error of .1 to .8 in a single iteration. Because this behavior appears to be induced by the varying learning rate, constant values for the learning rate are recommended.

4.6 Activation Function

The sigmoid and hyperbolic tangent functions were investigated as activation functions for backpropagation. Using the sigmoid function, the network converged.

When the hyperbolic tangent function was used, the network did not converge in any of the test cases. The behavior with the hyperbolic tangent was similar to that of the range shifted sigmoid. During the training processes, the error decreased steadily from high values of .9 to lower values as though it were beginning to converge; then, the error would increase until it reached 1 or very close to 1 with no evidence of decreasing.

4.7 Initial Weights

Initial weights are set to small random values. Frequently in neural network literature small random values are recommended as a starting point. Yet it is not always clear what range of values is small enough to allow the network to converge quickly and effectively without becoming too small. The impact of the starting value of the random number generator which dictates the starting point on the error surface was also looked at. The starting value appears in most cases to have limited effect. It can induce differences in the number of iterations required for convergence, however in the test cases run it did not appear to cause a given data set to converge or not converge. Depending on the error surface encountered, it is foreseeable that local minima could be encountered for one weight set and not for another but that did not appear to be the case in this research.

The weight size displayed an interesting result. The size of the initial weights

were varied by orders of magnitude from the input values. The first set of weight values were roughly one order of magnitude smaller than the inputs. These initial values were then lowered an order of magnitude at a time and the behavior of the training process of the neural network was observed. One could disallow larger initial weight values because values larger than the output range of the activation function would tend to send network output values to the extreme values of the function where the flatness of the function would inhibit change. The input data is normalized between -1 and 1. Thus one would assume the random values should range over something smaller than this. The weight range -.5 to .5 was investigated first and provided satisfactory results when used in conjunction with other appropriate parameters. However, when weights were reduced by 1 order of magnitude, -.05 to .05, the network using refraction data sets for training appeared to plateau in the training process just below a threshold of .5. When the magnitude was reduced by 2 orders, -.005 to .005, the network using tomographic data for input plateaued out just below a threshold of .5 when a single trace was presented. The process was halted at over 200K iterations because this length of training seemed excessive. When all the traces were presented at once, the network was able to converge in 10228 iterations. In the previous case where the weights were reduced by just one order of magnitude, the number of iterations necessary was just 176, and the initial weight set, range -.5 to .5, took only 81 iterations to train. Thus values within about 1 order of magnitude

of the inputs seem reasonable for the weight size.

4.8 Output Threshold

In an effort to improve generalization and aid in convergence, a modifiable output threshold was introduced. The threshold value used with the sigmoid function is typically .8 or .9 which are values considered close to 1. By reducing the threshold, the network is learning the training set partially before being presented the next case. Once all training cases are presented at the lower threshold, the threshold is changed to .9 for the completion of training. When using the methodology of the first training set being the first trace and the second set being traces one and two and so on as described in Section 4.3, the tendency may be for the network to give the first traces undue credit in the training process by conforming mostly to the forms given in the first traces rather than generalizing from the full data set. By introducing a threshold value of .1 the program bypasses the cumulative training process and essentially jumps to the last training set which gives equal representation to all the traces (see Figure 4.9). The result here was quite dramatic. Convergence rates were significantly reduced. Some data types which did not previously converge were able to converge, and some traces which induced significant alteration of weights, indicated by excessive numbers of iterations for a given training set, now converged in numbers of iterations which were fractions of that experienced using higher output thresholds.

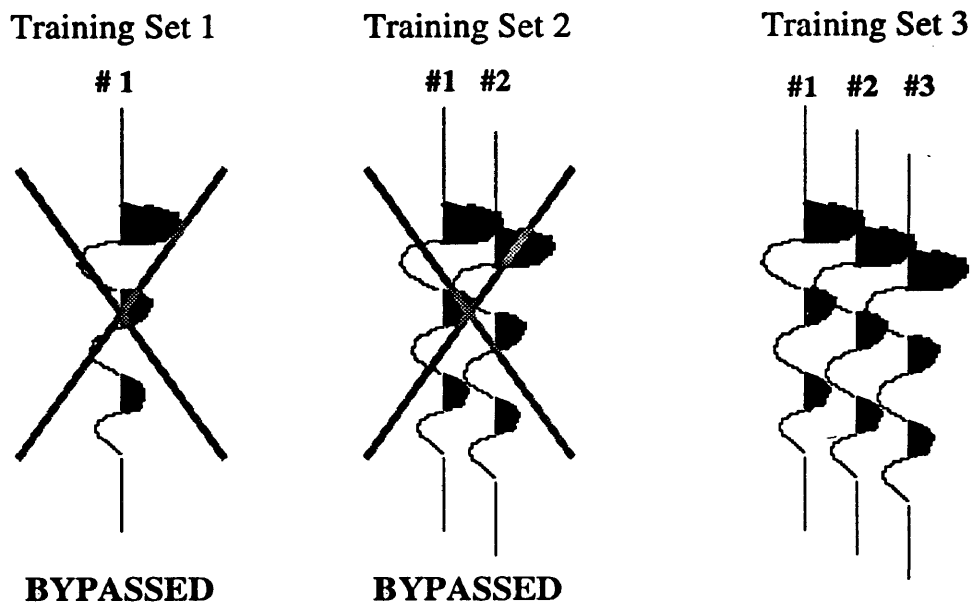


Figure 4.9: Each trace receives equal representation when the output threshold is .1.

These results can be seen in Tables 4.1 and 4.2. It is worth noting in these two examples that the robustness, the ability to generalize, is not significantly different despite the dramatic differences in iterations.

Tests done using the high output threshold values throughout the training process gave evidence that the training process was spending many iterations trying to define detail rather than getting the general picture from the data. The first traces were setting the weight values. Thus, subsequent traces had to rework the weights if new characteristics were encountered. This task proved arduous and sometimes impossible given the right conditions.

Table 4.1: Comparison of output threshold of .9 versus output threshold of .1 using tomographic Data Set 1 for training. The dashes indicate that the network did not iterate back over the data in the training set specified.

Training Set	Iterations	
	Output Threshold .9	Output Threshold .1
1	226	—
2	37	—
3	—	—
4	—	—
5	11	—
6	—	—
7	24	—
8	9	—
9	8	—
10	40	81

Table 4.2: Comparison of output threshold of .4 versus output threshold of .1 using refraction data set, Shot Point 24, for training.

Training Set	Iterations	
	Output Threshold .4	Output Threshold .1
1	692	—
2	120	—
3	110	—
4	2247	—
5	5952	—
6	1033	—
7	29303	—
8	38	—
9	178	—
10	557	1644

4.9 Output Range

The output range of the sigmoid function, which is typically 0 to 1, is shifted to -.5 to .5. This shift was suggested for use with binary data by Stornetta and Hubermann as referenced by Wasserman [9]. The backpropagation technique makes weight changes proportional to the output value of the previous layer. For binary data with approximately half of the input values being zero, the data would induce no change in weights tied to the zero input value. In an effort to change this difficulty with the zero values, inputs were shifted to -.5 and .5 along with the output range. This yielded a 30-50 percent decrease in convergence time.

This concept was investigated for non-binary data such as trace data. The input data, previously normalized from 0 to 1, was changed to -1 to 1 and the training time improved markedly. This is likely due to promoting the use of values of the sigmoid for input values equal to or less than 0, which is the lower half of the sigmoid function, as well as avoiding encounters with zero-valued inputs. This change was kept as standard procedure. The shift of the output range to -.5 to .5 proved ineffective in most cases. It would induce instability and eventual divergence of the network. In earlier studies, when input values ranged from 0 to 1 and the simplest data forms were used, the network might see the convergence speedups suggested by Stornetta and Huberman, but the instability existed such that slight shifts in parameters and different data sets might just as likely diverge. The output range of 0 to 1 for the

sigmoid activation function appears to be the most advantageous with regard to the trace data utilized in this research.

4.10 Trace by Trace Inputs

The most extensively used program for the first break study was the trace by trace program. It was the test case for parameter evaluation as well as attribute evaluation. For simple data, tomographic data, and refraction data, the trace by trace program was able to train with one or more attributes per trace. Substantial noise could be added to the synthetic refraction data set and convergence was still obtained.

4.10.1 Tomographic Data

The robustness of the trained weight set was quite high for the tomographic data set. Figure 4.10 gives comparisons of the network's picks to my picks. The first ten traces were used for training purposes, thus the pick values are a one for one match to my picks. The marks are black for values above the threshold value of .9 and any outputs less than .1 are not included. The shades of gray indicate values between .1 and .9. In all cases the network's picks were in agreement with each other. These picks differed from mine on traces 22 and 23, where the network selected one sample later than I did. Notice that the variation in learning rate, L.R., and output threshold, O.T., had little if any effect on the network's outputs. As seen

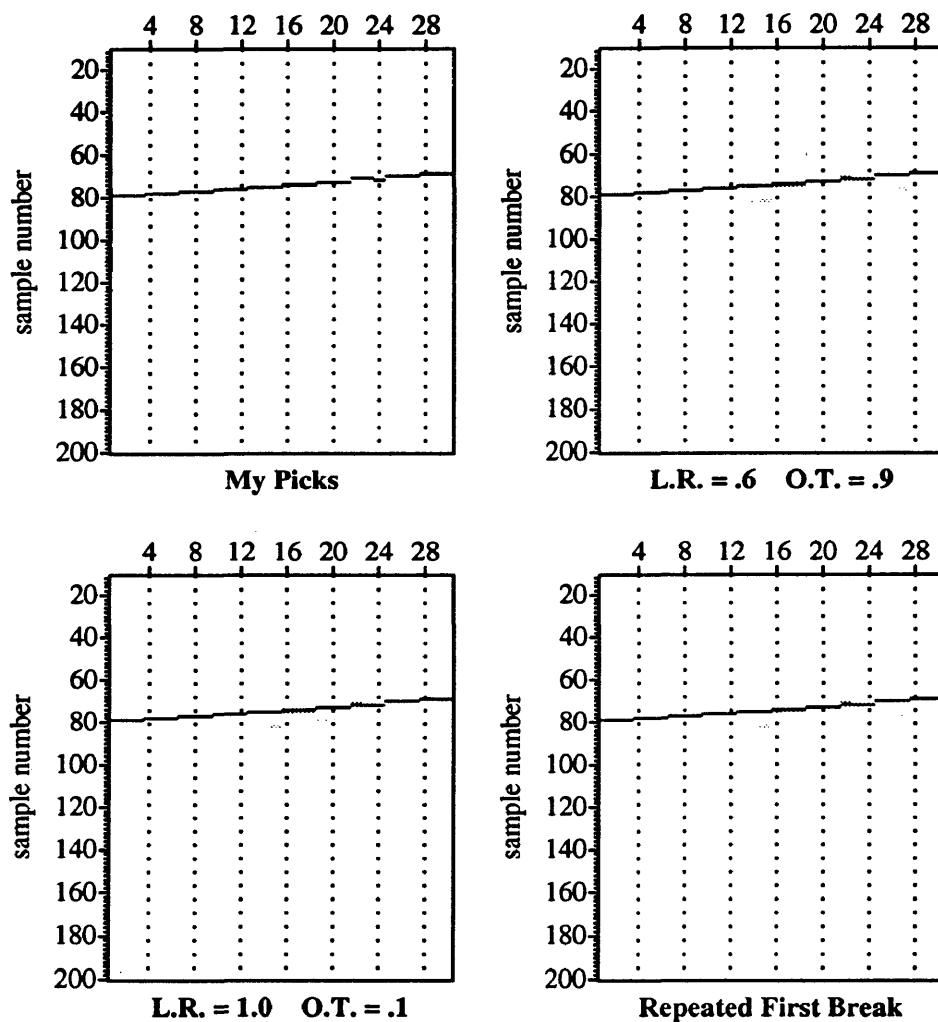


Figure 4.10: A comparison of the neural network picks to my picks using the tomographic data set, Data Set 1. The neural network parameters output threshold, O.T., and learning rate, L.R., were varied in the first two examples. The number of presentations of the first break picks were increased in the last example.

in Section 4.8, the number of iterations required for training differed considerably. In the fourth example in the figure, the presentation of the first break was repeated one for one with all other positions on each trace. This increased the number of first breaks seen by the network to a number equal to the number of windows visited which were not first breaks. Notice a slight reduction in extraneous responses below the first break picks in traces 25 and 27. However, the substantial increases in the amount of data presented appears to have a minimal effect on the network's first break choices.

Although not pictured, increasing the number of traces used in training the network did yield higher output values in subsequent picking. This suggests that providing the network with more examples makes the network outputs more robust. Additional examples of the tomographic data results can be found in Appendix C Section C.1.

4.10.2 Refraction Data

The refraction data results were less impressive. This decrease in capability is likely due to the decreased consistency of waveform and character within the data set. Figure 4.11 gives evidence that the introduction of additional attributes to the network improves the network's robustness. Although results are still not particularly concise, they exhibit improving clarity. The two attributes used were instantaneous phase and envelope slope. The three attribute case included raw data, and the four attribute case included envelope. This result offers support to the idea that the form

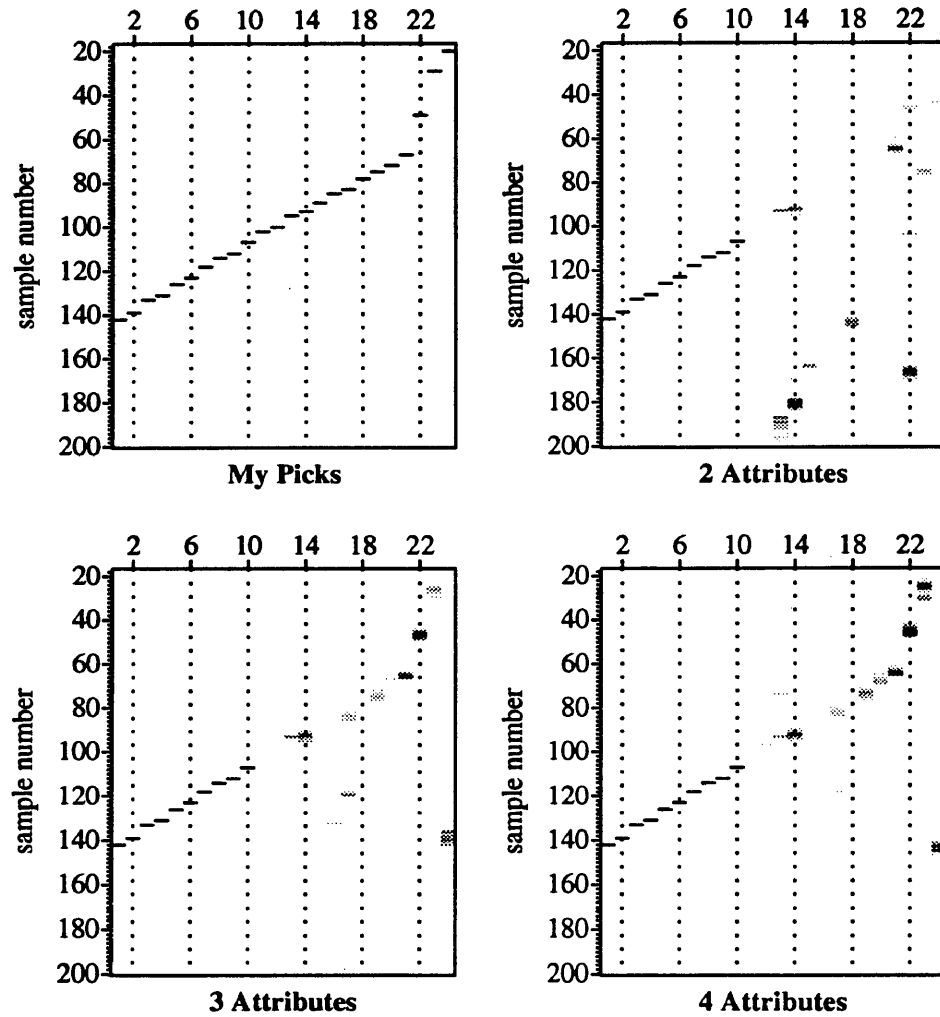


Figure 4.11: Use of multiple attributes to improve the picking capabilities of the neural network compared to my picks. The data set used for training and picking is the refraction data set, Shot Point 24.

in which data is input to the network is important, and the number of attributes needed depends upon the data type being used. Figure 4.12 shows that increasing the number of times the desired first break pick is presented to the network in the refraction data case is detrimental to the outputs of the network. This is likely due to the decreased consistency of the waveform. The first ten traces of the data set Shot Point 24 look somewhat different from the later traces. Making the network see repeated examples of the first traces is taking away the network's generalizing capabilities.

Although output values were no longer signalling picks near the desired threshold of .9 as was the case with the tomographic data, they were triggering lesser responses along an appropriate trend. Clearly this is not the absolute precision one would hope for, however these outputs could potentially be used for statistical inputs where multiple shots are evaluated. The data's fold could provide the repetition necessary to perhaps narrow the picks to a specific value or to within a sample. Additional results using refraction data can be found in Appendix C Section C.2.

4.10.3 Synthetic Refraction Data - Noise Effects

The synthetic refraction data was evaluated primarily to observe noise effects on the results in an environment where the noise levels could be controlled. The synthetic refraction data gives evidence that the neural network has difficulty picking refraction data in general. When noise is added the neural network clearly breaks

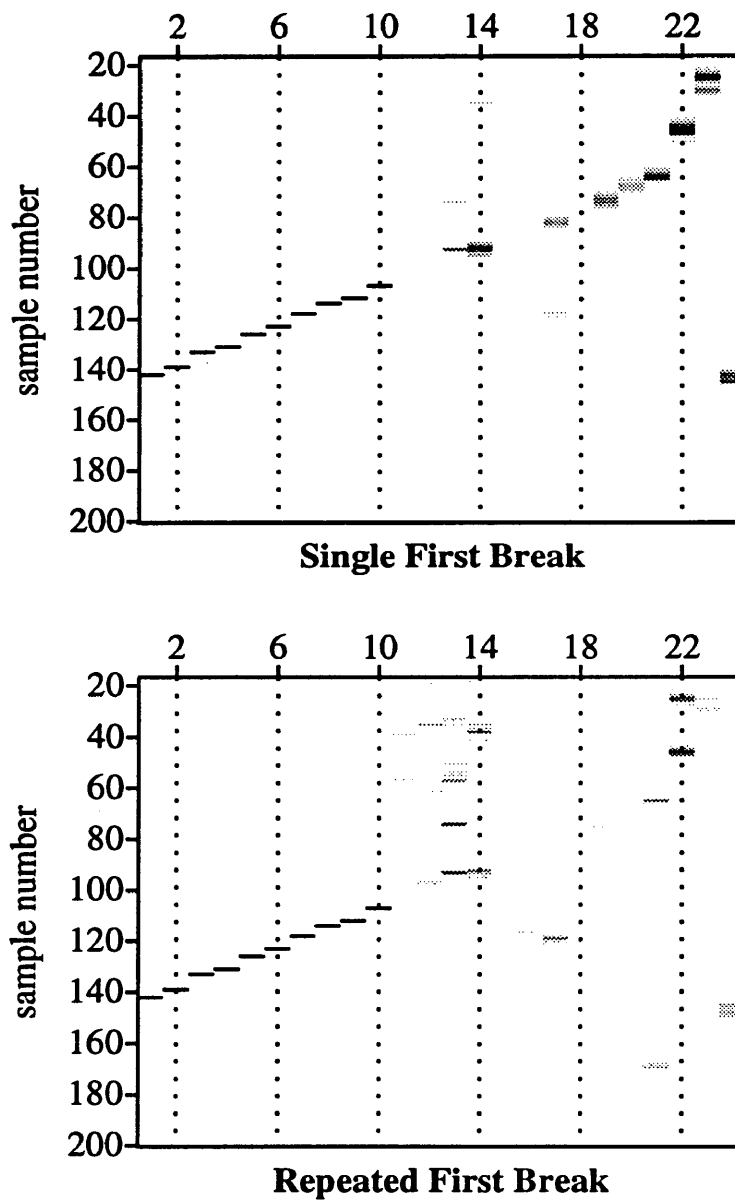


Figure 4.12: Comparison of outputs using a single first break presentation per trace and repeated first break presentations per trace for training with the refraction data.

down in accuracy (see Figure 4.13). An interesting note is that the more noise the data contained the faster the network trained. This could be considered an example of shorter convergence yielding a less robust network than the longer converging less noisy cases which clearly are more robust. Another example worth noting is that networks trained with noisy data appear to pick data containing less noise better than data containing noise levels comparable to the training set. This may suggest that the network is generalizing from the features of the first break despite the high noise contamination. Additional results from synthetic refraction data can be found in Appendix C Section C.3.

4.10.4 Blocked Data

The blocked attribute data allowed lengthy trace data to be condensed into much smaller groups of data. This made for much smaller less computer intensive networks. In the example shown in the text three blocked attributes were used as input. Again the neural network displayed difficulty picking the refraction data set. The picks break down particularly in the zone where the refraction data's amplitude is very low relative to other events on the trace (see Figure 4.14). The blocked attributes at best narrow the evaluation zone to a few windows of interest on the trace. Reevaluation of selected windows with time-sampled input data might then provide a specific first break pick. The main draw back here is the increased preprocessing time required to put the data in blocked format and evaluate which window corresponded to

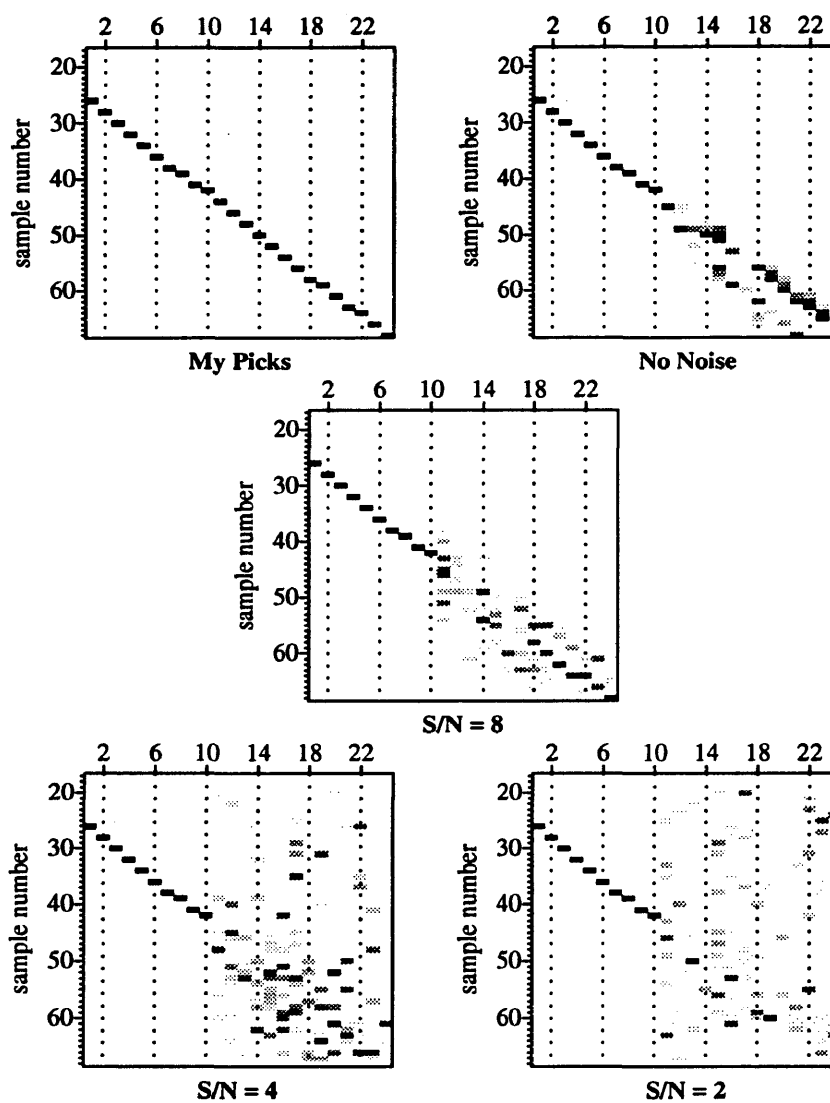


Figure 4.13: Comparison of outputs using synthetic refraction data with varying degrees of noise.

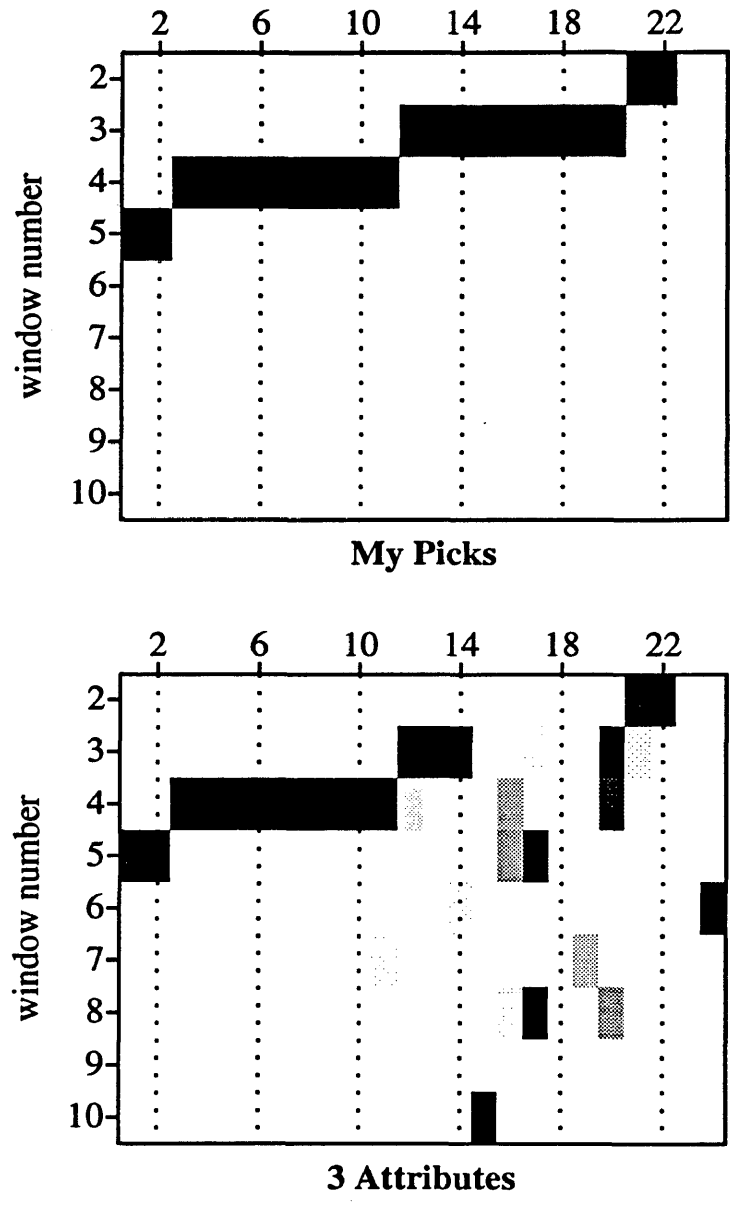


Figure 4.14: Comparison of outputs using blocked attribute data from Shot Point 24.

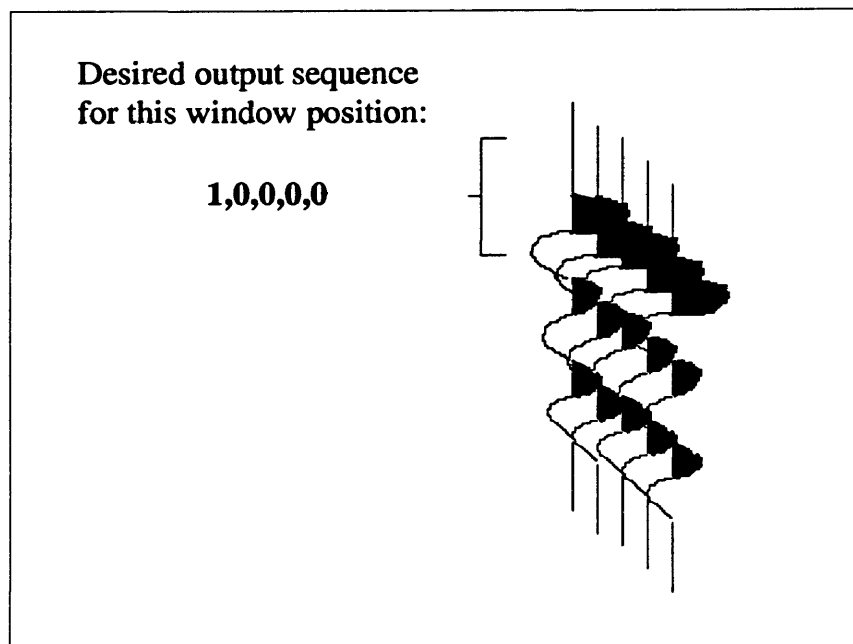


Figure 4.15: Input/Output scheme for a sliding window with five trace input and five outputs, one for each trace.

which peak. To see more results using various data sets, see Appendix C Section C.4.

4.11 Five Trace Inputs

The multiple trace implementation was investigated to capitalize on the sloping trends so evident in first break moveout of the direct and refracted arrival. Commonly a geophysicist will utilize this trend to interpolate picks on traces which are noise contaminated or corrupted in some other way. An example of how data is input and output to and from the neural network is shown in Figure 4.15.

The difficulties faced by making the neural network analyze multiple trace

input are that frequently these slopes change because of subsurface velocity changes within a set of data which is being evaluated. Not only does the waveform and trace character need to be consistent from the training set to the data set, the moveout slope also needs to be consistent. A neural network tends to make the data it picks conform to a learned slope which may suppress the detail that one wishes to retain when making first break picks. This method was most logically used with the blocked data attributes because locating the trend of the data was the primary capability of this attribute. Figure 4.16 shows the results for using the five trace method with these blocked attributes as input. The outputs of this method appear more robust than previous blocked methods. Although not displayed here, this method provides repetitive picks. Traces 5-20 are picked five different times as the five trace input is moved across the data. Thus, uncertain picks in one five trace grouping might be resolved in another five trace grouping containing the trace with the uncertain pick. Another possibility is that an average of the outputs could be taken for each trace. An example of the entire set of picks can be found in Appendix C Section C.5.

Perhaps this sloping data characteristic could be better implemented via an expert system which imposes constraints on the window where a neural network is allowed to locate the first break. Thus true variations in the subsurface are not forced to conform to the understanding of the network.

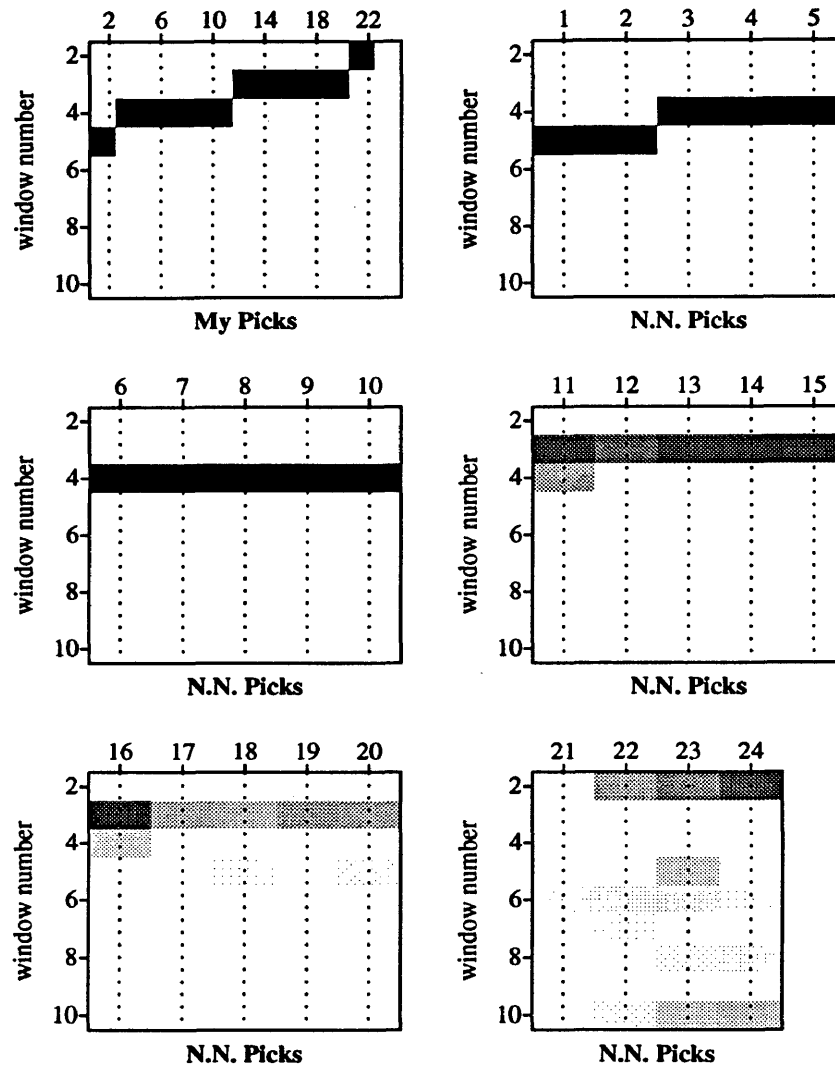


Figure 4.16: Comparison of outputs using five input traces at a time. The blocked attribute data from Shot Point 24 was used as the input data. Notice that the neural network picks, N.N. Picks, display only five trace outputs per box which differs from the My Picks box.

4.12 Coherent versus Noncoherent Targets

Looking at the data from a different aspect, the target for evaluating the first break was changed. Rather than isolating a single sample value at which the first break occurs, the data is separated into categories being either noncoherent or coherent. Noncoherent data is everything before the first break, and coherent data is everything after and including the first break. Noncoherent data has a desired output of 0 and coherent has a desired value of 1. This method displayed difficulty in training and in most cases it would not converge. Because many of the trace attributes are not constant values but exhibit changing values in the zone of coherent data, the sliding window technique implemented in this research would encounter varying patterns in the data depending on its position. A single weight set is looking for a single characteristic pattern. With the sliding window method multiple patterns requiring a threshold value of 1 would be encountered. This is most likely the cause of the difficulties in training.

Chapter 5

Conclusions

Trace attributes clearly play a role in implementing neural networks for seismic first arrival evaluation. When data becomes more complex, multiple trace attributes are needed to clarify the first break pick location. This is an important result because often it is argued that data should remain in its original state for evaluation using a neural network. These results suggest otherwise. The attributes which contributed the most were the amplitude coupled with instantaneous phase, envelope, and envelope slope. These additional attributes were particularly helpful when evaluating refraction data. The refraction data was modelled near the critical angle, thus numerous complexities were encountered simultaneously. The multiple attributes improved the network's output for this complicated situation. Outputs, which exhibited significant scatter when only one or two attributes were input to the network, trended much closer to the desired first break picks as more attributes were input.

This brings us to another issue, the types of data being evaluated. The tomographic data, which exhibited consistent waveforms across the record, yielded high

output values above or near the .9 threshold value indicating a first break pick. The picks were all within one sample of the picks I had selected. The picking of refraction data, however, was difficult because it was picked across a transition zone from one wave type to another. Even with the improvement sustained from the additional input attributes, the picks almost always varied from my picks, and the output values were often below the threshold value of .9, suggesting uncertainty in the pick. One could infer that refraction data could be picked with greater success if the data were evaluated separately with one network for direct arrivals and one for far offset refracted arrivals. Then the data would likely be more consistent in form from trace to trace and yield outputs like that of the tomographic data.

Condensing traces into window-sampled attributes for input to the network provides a tool for trend evaluation in the data. The output of this trend evaluation could be used as a first pass look at the trace data, focussing in of the zone of interest for evaluation by another network which would output more specific results. The window-sampled input method can pick the window where the first break resides to the nearest window. If picks to the nearest sample are desired, this method cannot be depended on for such results. The time-sampled input approach does provide the means to pick to the sample level.

There are some upfront costs involved in implementing a neural network which include input data evaluation, network parameter evaluation, and training the net-

work. However, once trained, the neural network provides consistency and speed, strong characteristics looked for in first break pickers.

References Cited

- [1] McCormack, M. D. 1990. Seismic trace editing and first break picking using neural networks. Expanded Abstracts. Proc. of SEG 60th Annual International Meeting, pages 321–324, San Francisco, CA.
- [2] Taner, M. T. 1988. The use of supervised learning in first break picking. In Proc. of the 1988 Symposium of Geophysical Society of Tulsa, ed. E. Bielanski, Tulsa, Okla.
- [3] Veezhinathan, J., D. Wagner, and J. Ehlers. 1990. First break picking using a neural network. (to be published).
- [4] Gelchinsky, B., V. Shtivelman. 1983. Automatic picking of first arrivals and parameterization of traveltimes curves. Geophysical Prospecting. 31: 915–928.
- [5] Hatherly, P. J. 1982. A computer method for determining seismic first arrival times. Geophysics. 47 (10): 1431–1436.
- [6] Bracewell, R. N. 1986. The fourier transform and its applications. 2d ed. New York: McGraw-Hill Book Company.
- [7] Taner, M. T., F. Koehler, and R. E. Sheriff. 1979. Complex seismic trace analysis. Geophysics 44 (6): 1041–1063.
- [8] Yilmaz, O. 1987. Seismic data processing. In Investigation in Geophysics, ed. S. M. Doherty, Vol. 2, p. 521. Tulsa, Okla: Society of Exploration Geophysicists.
- [9] Wasserman, P. D. 1989. Neural computing theory and practice. New York: Van Nostrand Reinhold.
- [10] Corana, A., et al. 1987. Minimizing a multi-modal function of continuous variables with the “simulated annealing” algorithm. ACM Transactions on Mathematical Software 13 (3): 522–580.
- [11] Kirkpatrick, S., C. D. Gelatt Jr., and M. P. Vecchi. 1983. Optimization by simulated annealing. Science 220 (4598): 671–680.

- [12] Rumelhart, D. E., G. E. Hinton, and R. J. Williams. 1986. Learning internal representation by error propagation. In Parallel Distributed Processing: Exploration in the Microstructures of Cognition, Vol. 1, Pp. 318–362, MIT Press, Cambridge, Mass.

Appendix A

Neural Network Background

The majority of this appendix is referenced from Wasserman [9]. In instances where information is gathered from sources other than Wasserman, those sources will be cited.

A.1 History

Man has long been fascinated with how the human brain functions. This fascination has kindled efforts to mimic the brain's ability to learn. The fields of neuroanatomy, neurophysiology, and psychology have been developing models of the brain for many years. The model proposed by D. O. Hebb in 1949 tied the structure of the brain to a learning law. This model acted as the basis for the development of the artificial neural network.

Through the 1950's and 1960's the biological and psychological aspects of the neural system were combined to produce artificial neural networks. The first systems were made using electronic circuits and progressed to computer simulation,

the form of neural network used in this research. The primary network studied during these years was the perceptron, single-layered network. It was successful for many problems, however for certain classes of problems it appeared to be unable to work. This inability was published by Minsky and Papert in 1969 where they proved the perceptron incapable of solving the XOR, exclusive-or problem. Because it appeared that neural networks would never be able to solve problems which were not linearly separable, the study and use of neural networks entered a hiatus during the 1970's.

A few research efforts continued during this hiatus by Kohonen, Grossberg, and Anderson. It was not until the 1980's when the backpropagation network, invented in three separate instances by Werbos in 1974, Parker in 1982, and Rumelhart, Hinton, Williams in 1986, provided the step needed to cross the barrier imposed by linearly inseparable problems by allowing multiple layers to be incorporated into neural networks. Although not trouble free, the backpropagation algorithm has become the most successful and widely used neural network algorithm of today.

A.2 Biological Basis

Neural networks of the computerized variety were clearly inspired by the physical makeup of the human brain. The human nervous system is comprised of 10^{11} neurons linked together by 10^{15} connections. Neurons are capable of receiving, processing, and transmitting electrochemical signals over neural pathways. Taking a

biological view of a neuron, the neuron has many extensions called dendrites. The connection points between the dendrites are known as synapses. The receiving side of a synapse conducts inputs to the cell body where inputs are summed, some inputs exciting and others inhibiting the cell's ability to fire. When total excitation exceeds a threshold, the cell fires sending a signal down the axon to other neurons (see Figure A.1). This biological process is the foundation on which computerized neural networks are based.

A.3 Network Construction

Beginning with the constructs of the fundamental processing element, the neuron, an artificial neuron is defined. Inputs to the neuron, X , are first multiplied by a weight corresponding to the synaptic strength. All the weighted inputs are then summed to determine the activation level of the neuron.

$$NET = \sum XW$$

An activation function further processes the value of NET to yield the corresponding output of the neuron. This function can be linear or, to allow for a more general network, a nonlinear function may be desired.

$$OUT = f(NET)$$

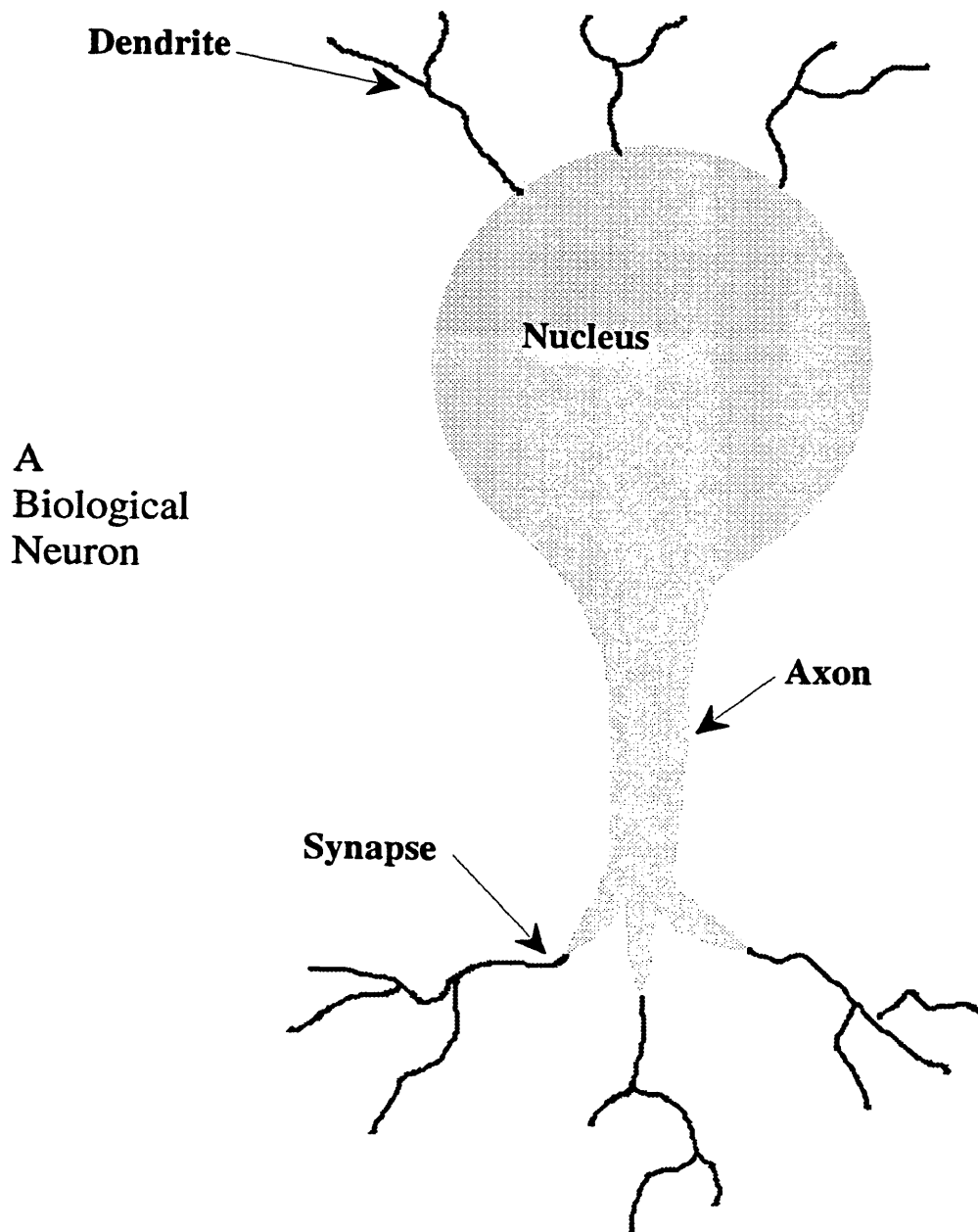


Figure A.1: The biological neuron after which the neurons in an artificial neural network were patterned.

Although crude in comparison to the biological neuron, the artificial neuron shows strong similarities to the biological systems.

To increase the computational power of a network, a multilayer network with a nonlinear activation can be used. The nonlinear activation function is critical because a linear activation function would provide no additional computational power over a single-layered network. It has also been determined by research conducted by Kolmogorov that the largest number of layers required to describe a problem is three. Note that in this research the input layer is included as a layer in the network. Thus the two cases stated above would be two and four layer networks in the context of this research.

A.4 Backpropagation Algorithm

The backpropagation technique for training neural networks was brought to the forefront in 1986 by Rumelhart, Hinton, and Williams. Others were found to have described this methodology at earlier times, Parker in 1982 and Werbos in 1974. However, the use of this method was driven by the later work.

A.4.1 General Overview

The benefit of this technique is that it can be applied to a network with any number of layers. This allows for the training of simple to complicated patterns requiring two layers to four layers in the most complicated cases. The training algo-

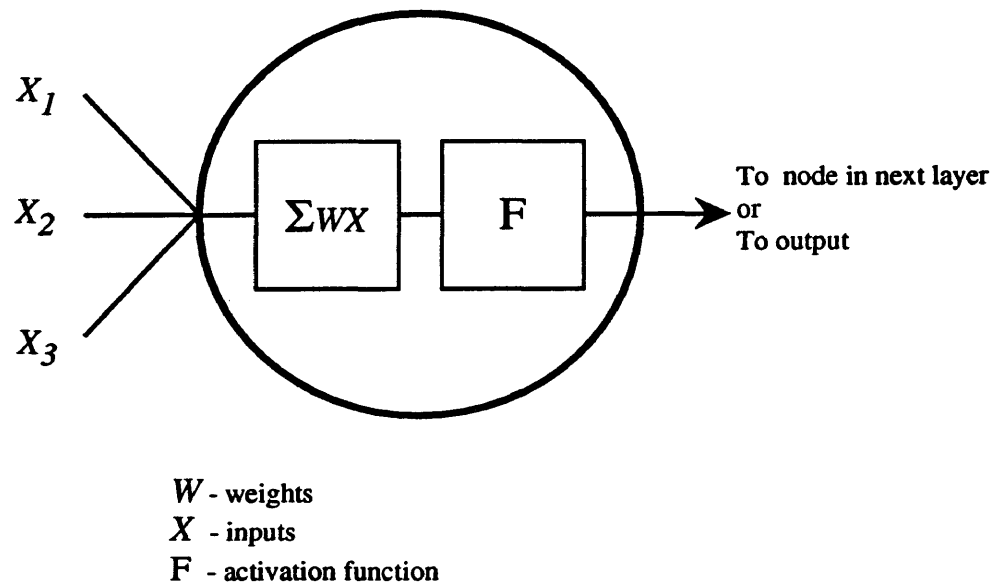


Figure A.2: Schematic of the forward pass through a neural network.

rithm, based on the generalized delta rule shown in equation (A.2), utilizes gradient descent error minimization. Because gradient descent depends on calculating derivatives of the outputs of the network, the network must be based on a function that is differentiable everywhere. This function is known as the activation function. A set of inputs is multiplied by corresponding weights and the products are summed for each node in the subsequent layer. This sum, which will be referred to as *NET*, is calculated for all neurons in the net. The activation function is then applied to the value of *NET* to yield a value called *OUT* (see Figure A.2). The most commonly used function for this purpose is the sigmoid function. It is sometimes referred to as

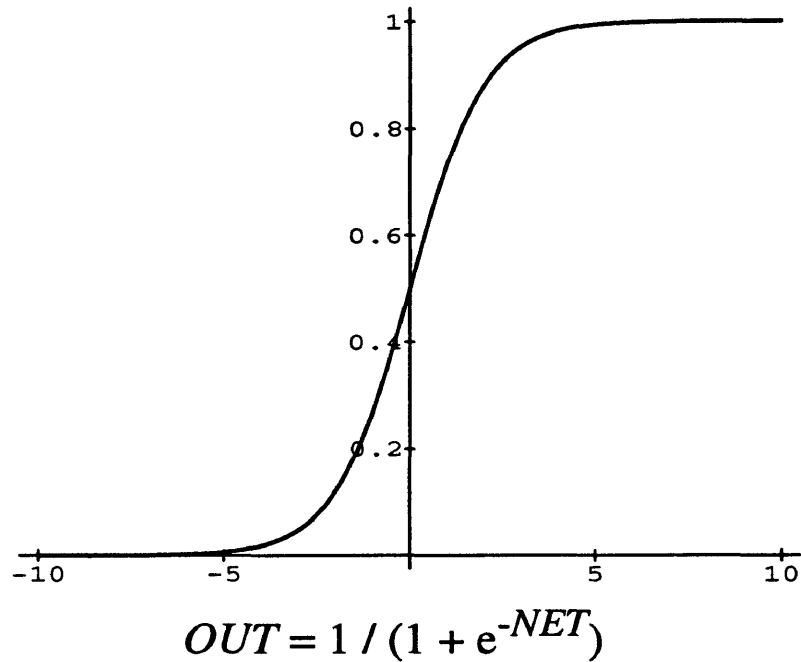


Figure A.3: The sigmoid activation function, sometimes known as the logistic or squashing function.

the logistic or squashing function because it forces the values of NET to an OUT value between 0 and 1 (see Figure A.3).

$$OUT = \frac{1}{1 + e^{-NET}}$$

This function is differentiable everywhere and its derivative is very simple to

calculate.

$$f'(NET) = \frac{\partial OUT}{\partial NET} = OUT(1 - OUT) \quad (A.1)$$

Because the sigmoid function is nonlinear, it allows the multi-dimensional networks to have a greater ability to represent complicated features than their simple two-layer counterparts.

A.4.2 Training Procedure

The steps of training a network are as follows.

1. Initialize network weights.
2. Input a training set into the network.
3. Calculate the output of the network.
4. Check the desired value against the output.
5. Adjust weights to minimize the error.
6. Repeat process, starting with 2, until error reaches an acceptable minimum.

A.4.3 Derivation of Weight Changes

Understanding the reasons behind the weight changes in a backpropagation neural network is important, yet it can get confusing. Figure A.4 should help clarify the large number of indices to be encountered in this section. The backpropagation technique stems from the delta rule. An example of this can be seen for an

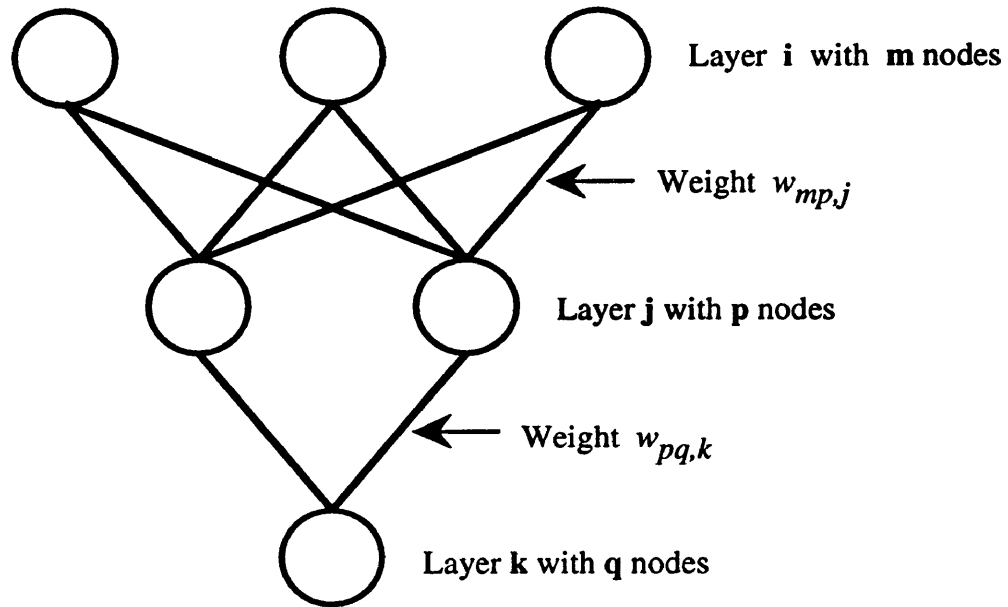


Figure A.4: Physical diagram of a neural network in a general case.

input/output pair in the following equation.

$$\Delta w = \eta (Target - OUT)X = \eta \delta X \quad (A.2)$$

The value, η , is known as the learning rate. It can be constant or variable, and it dictates the amount or percentage of change to take place with values which range between .01 and 1. From this simple relationship, the more complicated case for networks with more than two layers is investigated.

The process of adjusting the weights has the goal of minimizing the sum-squared error between the output of the network and a target value. The sum-squared

error for a given input/output training pair can be given as,

$$E = \frac{1}{2} \sum (Target_q - OUT_q)^2 .$$

Because the network is changed via the weights, the relationship of the change in weights needs to be related to the error.

$$\Delta w_{pq,k} \propto - \frac{\partial E}{\partial w_{pq,k}}$$

Through mathematical manipulation it can be said that,

$$\frac{\partial E}{\partial w_{pq,k}} = \frac{\partial E}{\partial NET_q} \frac{\partial NET_q}{\partial w_{pq,k}} . \quad (A.3)$$

By definition, the following can be found of NET ,

$$\frac{\partial NET_q}{\partial w_{pq,k}} = \frac{\partial}{\partial w_{pq,k}} \sum_p w_{mp,j} OUT_p = OUT_p .$$

To resolve the other part of equation (A.3), δ is given as,

$$\delta_q = - \frac{\partial E}{\partial NET_q} .$$

So an equivalent statement to equation (A.3) is,

$$\frac{\partial E}{\partial w_{pq,k}} = \delta_q OUT_p$$

or,

$$\Delta w_{pq,k} = \eta \delta_q OUT_p .$$

To determine the value of δ , the chain rule can be applied to write a partial derivative as the product of two partials.

$$\delta_q = -\frac{\partial E}{\partial NET_q} = -\frac{\partial E}{\partial OUT_q} \frac{\partial OUT_q}{\partial NET_q}$$

Further analysis of the above relationship shows,

$$\frac{\partial OUT_q}{\partial NET_q} = f'(NET_q)$$

and,

$$\frac{\partial E}{\partial OUT_q} = -(Target_q - OUT_q) .$$

The resulting value of δ can then be written as tangible values related to the network.

$$\delta_q = (Target_q - OUT_q) f'(NET_q)$$

The value of δ in the previous equation applies to the output layer only. For the hidden layers, we again use the chain rule to yield the following result.

$$\begin{aligned} \sum_q \frac{\partial E}{\partial NET_q} \frac{\partial NET_q}{\partial OUT_p} &= \sum_q \frac{\partial E}{\partial NET_q} \frac{\partial}{\partial OUT_p} \sum_p w_{pq,k} OUT_p \\ &= \sum_q \frac{\partial E}{\partial NET_q} w_{pq,k} \\ &= - \sum_q \delta_q w_{pq,k} \end{aligned}$$

So for hidden layers, δ can be calculated simply as,

$$\delta_p = f'(NET_p) \sum_q \delta_q w_{pq,k} .$$

Now that both δ 's can be calculated, the equations for generating the weight changes can be written.

$$\Delta w = \eta \delta OUT$$

Using the sigmoid as the activation function and from equation (A.1), the weight change for the output layer is as follows,

$$\delta_q = OUT_q(1 - OUT_q)(Target_q - OUT_q)$$

$$\Delta w_{pq,k} = \eta \delta_q OUT_p$$

$$w_{pq,k}(n+1) = w_{pq,k}(n) + \Delta w_{pq,k} ,$$

and for the hidden layer case,

$$\delta_p = OUT_p(1 - OUT_p) \sum_q \delta_q w_{pq,k}$$

$$\Delta w_{mp,j} = \eta \delta_p X$$

$$w_{mp,j}(n+1) = w_{mp,j}(n) + \Delta w_{mp,j} .$$

The majority of this section was referenced from the work of Rumelhart, Hinton, and Williams [12] with some changes in indices and variables to suit this manuscript.

Appendix B

Processing Flow

ARTHUR LAKES LIBRARY
COLORADO SCHOOL OF MINES
GOLDEN, CO 80401

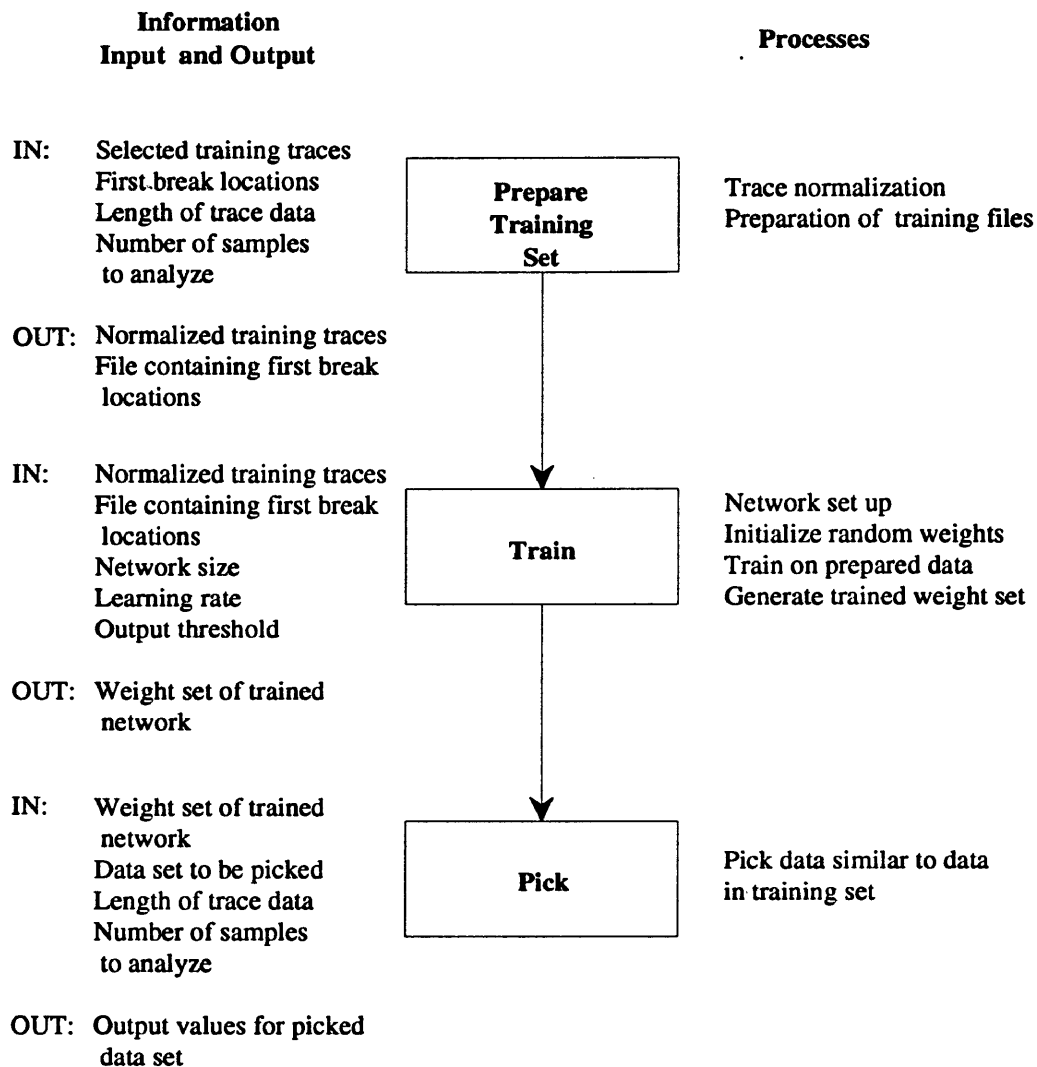


Figure B.1: Flow chart of the processing scheme for training the neural network and picking first breaks.

Appendix C

Results Listings

C.1 Tomographic Data

This section contains additional results obtained using the tomographic data sets. The following examples show picks made on subsequent data sets by the network trained with the first ten traces from Data Set 1 (see Figures C.1 to C.4). Notice that the results for Data Set 5 show no consistency at all. This can be attributed to the differences in waveform between Data Set 1 (the training set) and Data Set 5. Pictures of this data are located in Chapter 2.

C.2 Refraction Data

This section contains additional results obtained using the refraction data sets. The following examples show picks made on subsequent data sets by the network trained with the first ten traces from Shot Point 24 (see Figures C.5 to C.8).

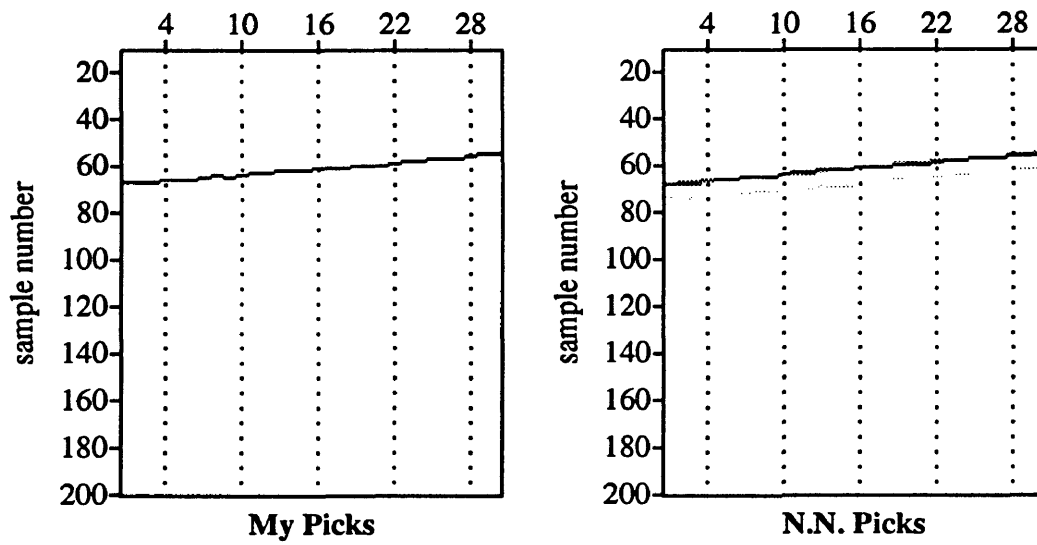


Figure C.1: Data Set 2 is picked with a neural network trained with Data Set 1. My picks for Data Set 2 are shown for comparison to the neural network's picks.

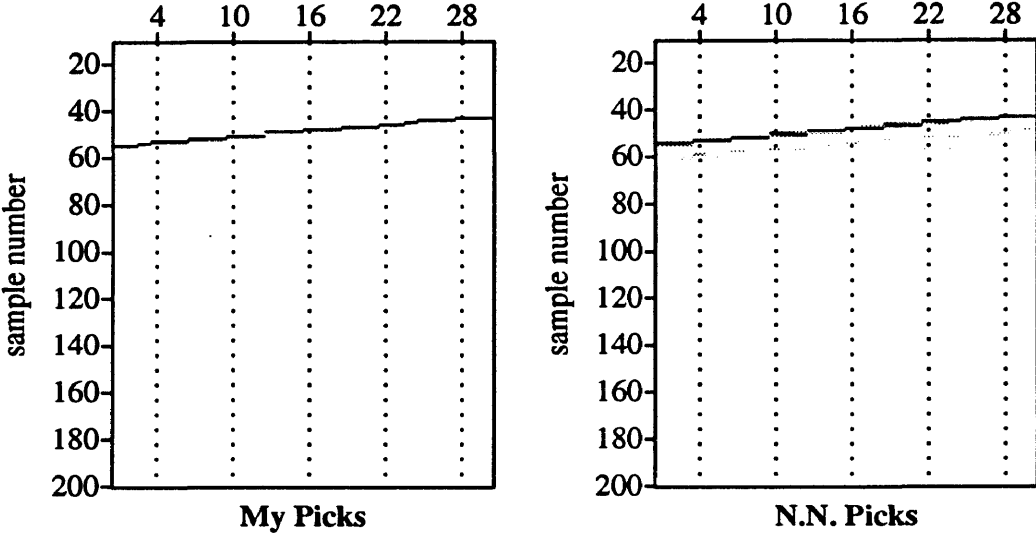


Figure C.2: Data Set 3 is picked with a neural network trained with Data Set 1. My picks for Data Set 3 are shown for comparison to the neural network's picks.

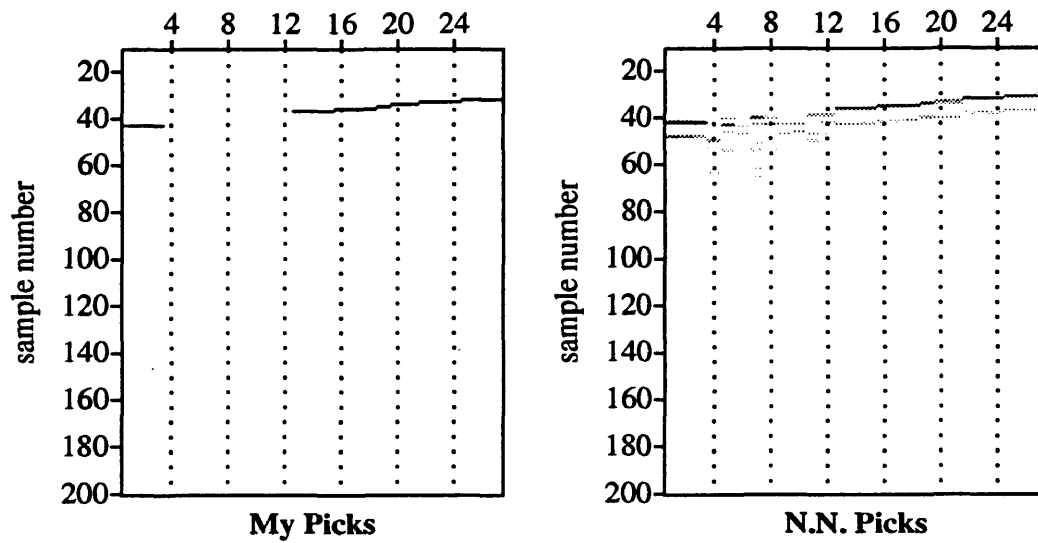


Figure C.3: Data Set 4 is picked with a neural network trained with Data Set 1. My picks for Data Set 4 are shown for comparison to the neural network's picks.

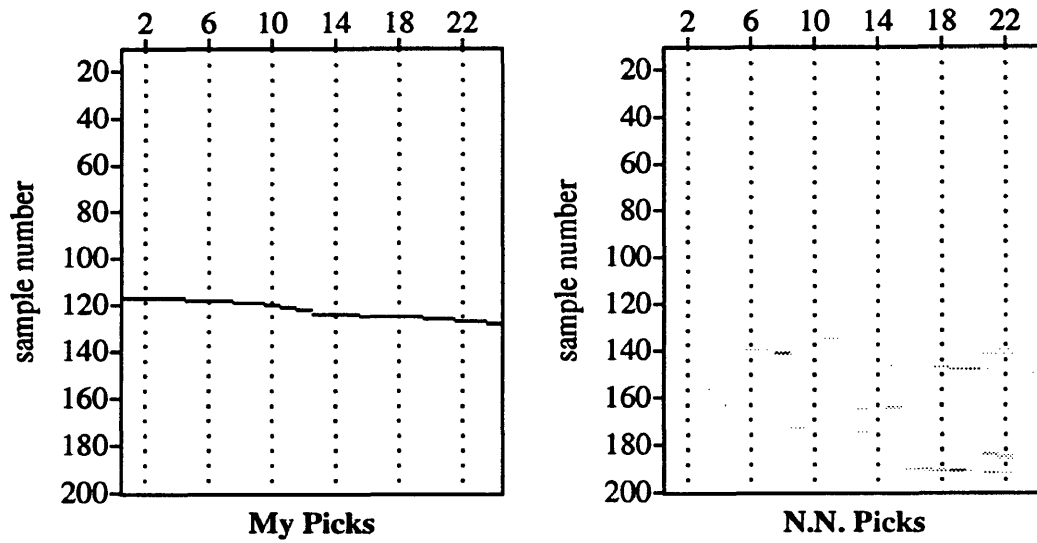


Figure C.4: Data Set 5 is picked with a neural network trained with Data Set 1. My picks for Data Set 5 are shown for comparison to the neural network's picks.

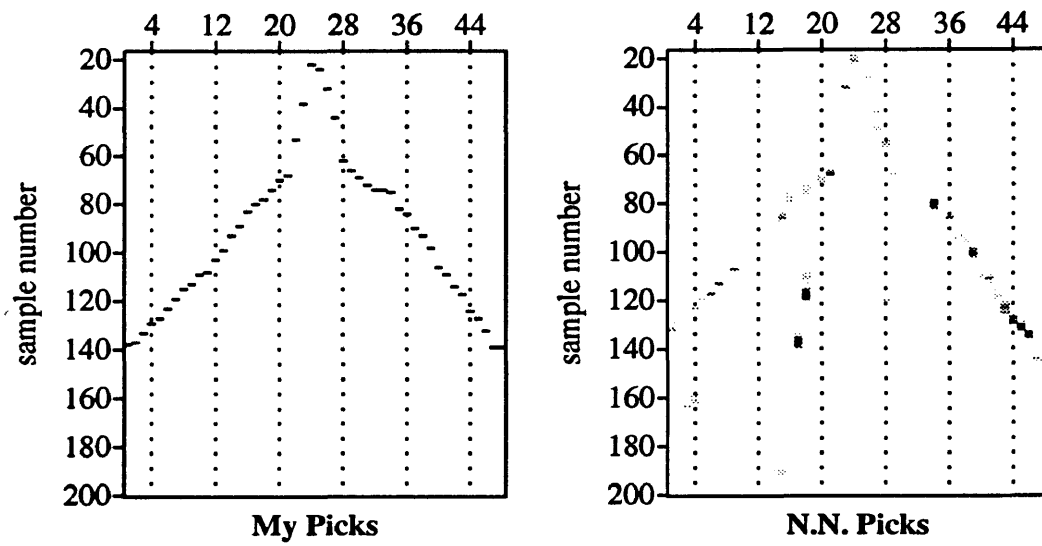


Figure C.5: Shot Point 11 is picked with a neural network trained with data from Shot Point 24. My picks for Shot Point 11 are shown for comparison to the neural network's picks.

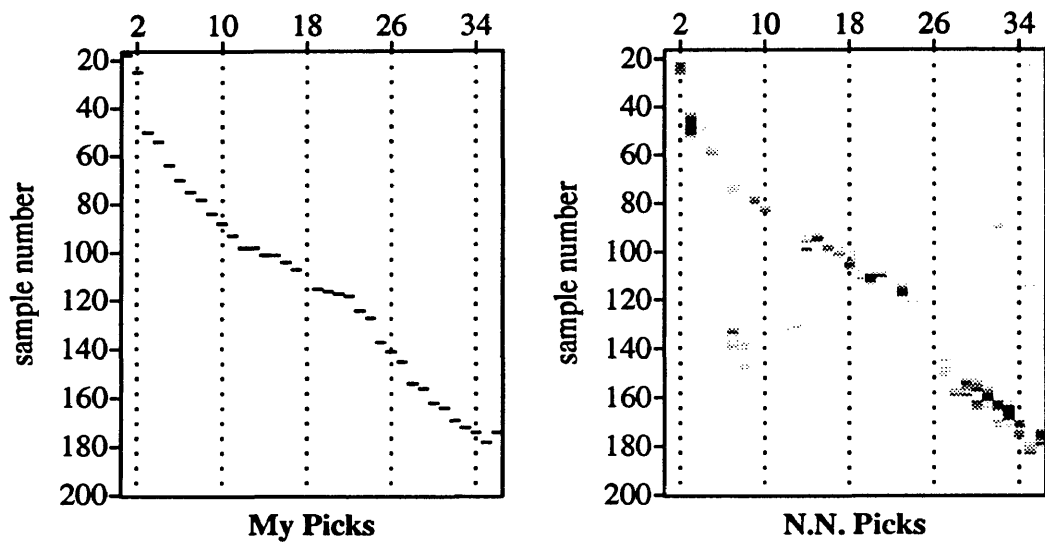


Figure C.6: Shot Point 23 is picked with a neural network trained with data from Shot Point 24. My picks for Shot Point 23 are shown for comparison to the neural network's picks.

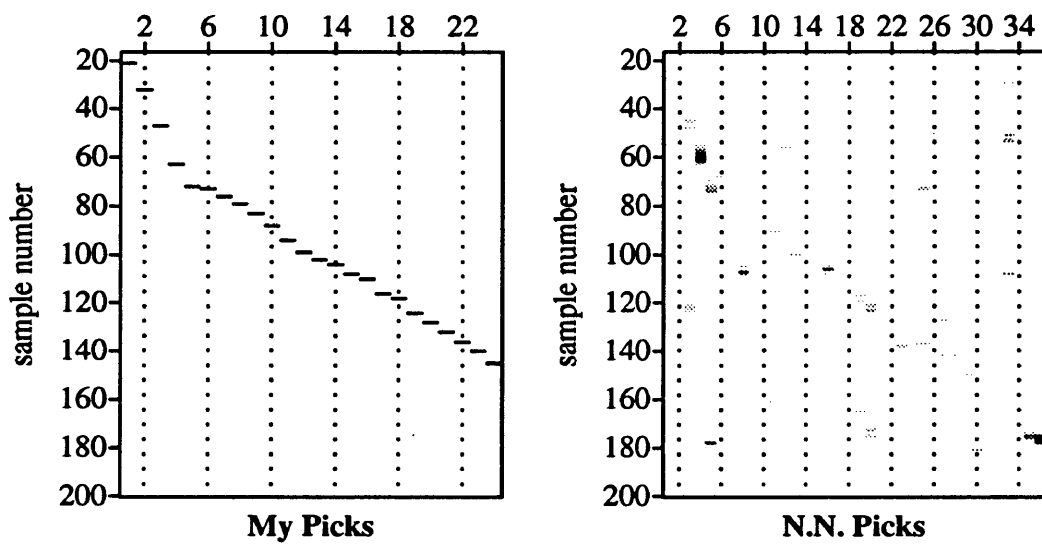


Figure C.7: Shot Point 35 is picked with a neural network trained with data from Shot Point 24. My picks for Shot Point 35 are shown for comparison to the neural network's picks.

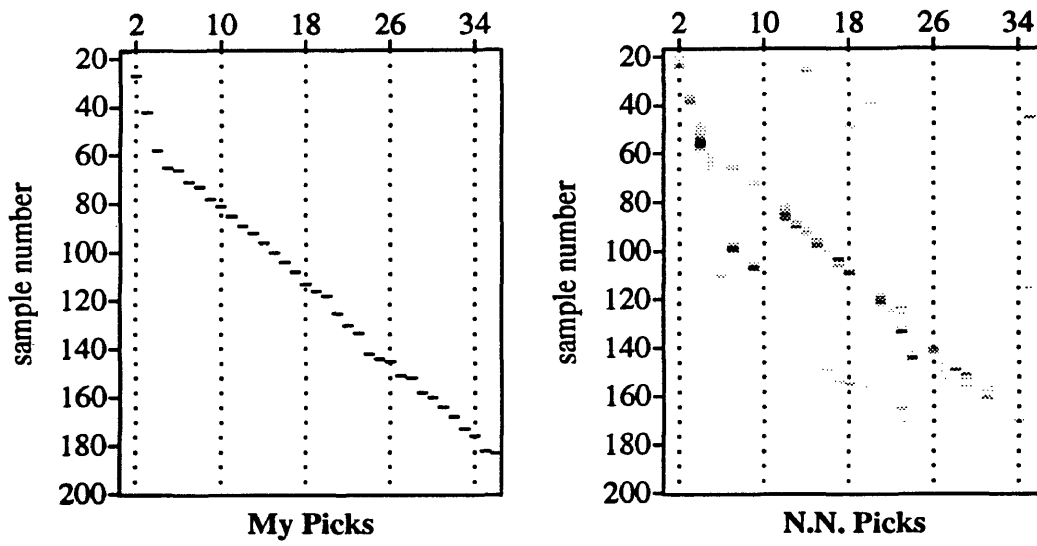


Figure C.8: Shot Point 47 is picked with a neural network trained with data from Shot Point 24. My picks for Shot Point 47 are shown for comparison to the neural network's picks.

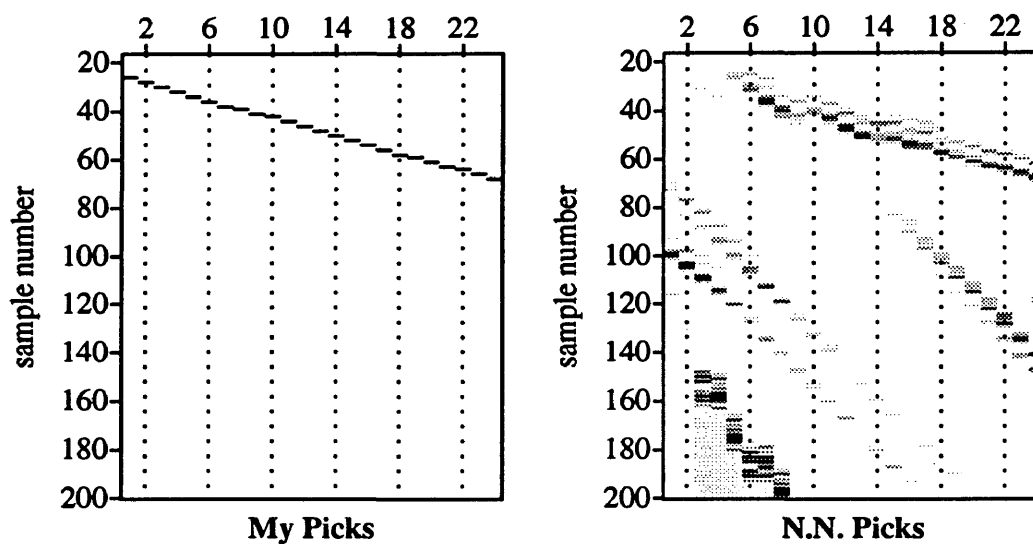


Figure C.9: Synthetic refraction data containing no noise is picked using a neural network trained with data having a S/N ratio of 2. My picks are included for comparison.

C.3 Synthetic Refraction Data

This section contains shows an example of a network trained with very noisy data picking data which contains no noise (see Figure C.9). There is evidence that the noisy training set is better able to pick less noisy data where a noise free training set does a poor job of picking noisy data.

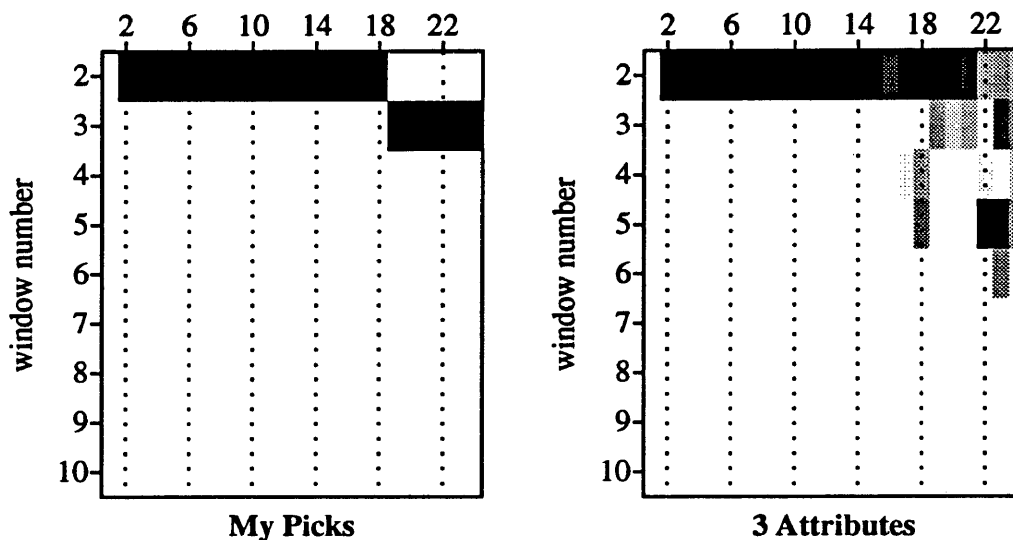


Figure C.10: Window-sampled attributes from the synthetic refraction data set are picked using a network trained with the first ten traces of the same synthetic refraction data set. My picks for the data are given for comparison.

C.4 Window-Sampled Data

This section gives the blocked attribute results of picking the synthetic refraction data (see Figure C.10). It appears that the results were less favorable here than those obtained from the blocked attributes of real data, Shot Point 24.

C.5 Five Trace Input

This section contains the full set of outputs for the five trace example give in the results chapter. Each panel in the two figures containing this data contains five

picked traces. The first panel starts with traces 1 to 5 and the next contains traces 2 to 6 and so on (see Figures C.11 and C.12). Notice the variations from one panel to the next. In some panels the values are the same as my picks and in others the picks are very different for the same trace. This suggests that the redundancy of this approach could prove valuable in improving the quality of the picks overall.

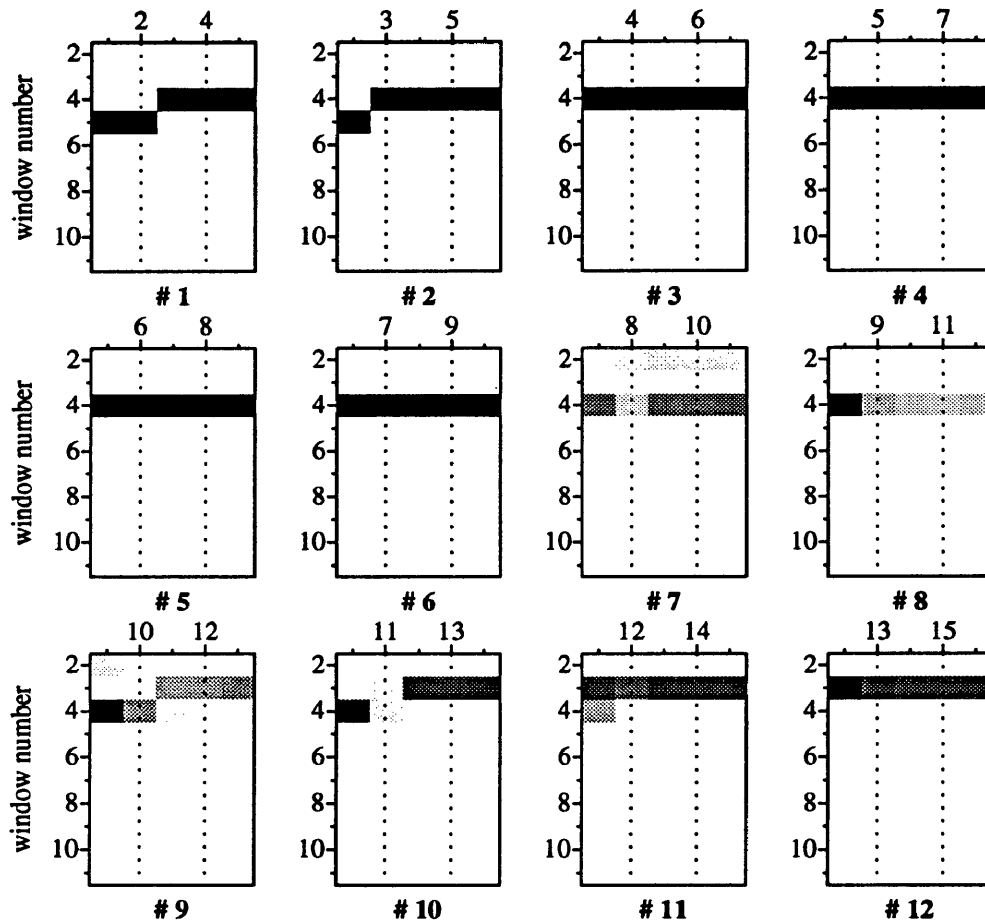


Figure C.11: Panels 1 to 12 show the progression of picks output from a neural network implementing the five trace input method. Note the redundancy of picks as the panels step forward one trace per panel.

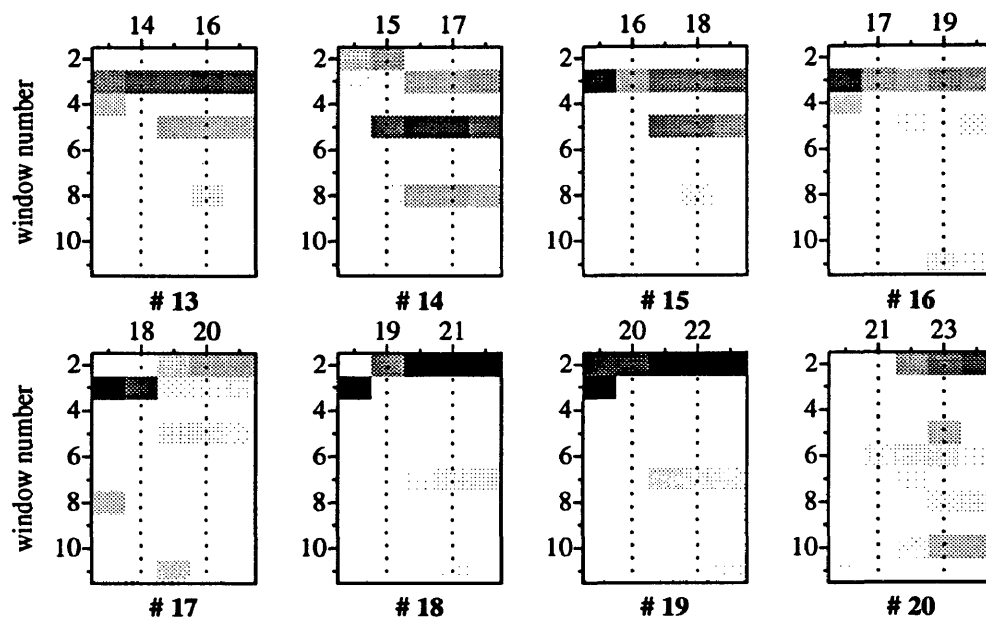


Figure C.12: Panels 13 to 20 continue the progression of picks output from a neural network implementing the five trace input method. A total of twenty panels are output for a data set containing twenty-four traces.