NEURAL NETWORK PROCESSING OF IMPACT ECHO NDT DATA

Rushna Begum (BEng.)

This thesis is submitted for the Degree of Doctor of Philosophy

City University

Department of Civil Engineering

February 2000

To

Tasneem and Ibraheem

CONTENTS

Chapter 6: The Back Propagation Network

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

Denis Chamberlain	City University	Supervising my research, proof reading and encouragement.
EPSRC		Funding and grant.
Larry Olson	Olson Engineering	Providing impact echo data for neural network analysis.
John Hill	City University	Advice on Fast Fourier Transform.
Mark Pickering	Strucom	Help with ANSYS.
Julian Chua	City University	Encouragement and friendship.
Shereen Akrawi	City University	Encouragement and friendship.
Ross Rashkoff	City University	Help with ANSYS.
Maruk Miah	LogiGates Ltd	Providing me with computers, and believing in me.
Omar Chuto Miah	eRegeneration Ltd	Encouragement and believing in me.
Family		For looking after my children and putting with me.

I especially would like to thank **my husband**, who has constantly encouraged me and made it possible for me to complete this research.

DECLARATION

"I grant powers of discretion to the University Librarian to allow this thesis to be copied in whole or in part without further reference to me. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgment".

ABSTRACT

In the developed world, acid rain attack, chloride contamination, and inadequate design of quality control during the construction stage, has contributed to the damage of reinforced concrete buildings and structures. The importance of non-destructive testing (NDT) of concrete is substantial, so that consistent and effective maintenance of the structures is implemented correctly. This requires assessment methods that can identify the initiation of defects so that appropriate actions may be taken to prevent large scale deterioration.

The impact echo test method is one of the most recent of the NDT used on concrete for the detection of damage, and has been applied in this research. However, the interpretation and analysis of the output requires subjective judgement. This research proposes to use an artificial neural network in impact echo data output analysis and interpretation. Neural networks comprise of numerical processing elements which are linked in a way that the network generally can learn by examples and store such experience for later use. They are trained using past data records from the output, so that an appropriate trained network is able to generalise when presented with inputs not appearing in the training data. The major feature of neural networks for this application is that it does not involve subjective judgement, and provides a fast and efficient method for analysis of large quantity of data.

In the research, the finite element method is used to build a simulation of the NDT method. A model of a wall with voids, for example, is built where a force or pressure is used to represent the impact, the response measured at a nearby point. As well as providing theoretical analysis, this numerical method allows creation of reliable data for use in neural networks training. However, due to the time limit, the author was not able to use finite element data in the neural network analysis.

Chapter one

Introduction

1.1. Introduction

This chapter gives the background and scope of the research, which is the need for data interpretation in impact echo testing. It explains what is done to the data once it has been collected, an artificial neural network used to automate interpretation. Simulation of the impact echo system is also evaluated using the finite element method. The research objectives are presented with the scope of this investigation.

The two hypotheses for this research are:

- Hypothesis 1: The interpretation of impact echo data can be reliably implemented through the use of a neural network.
- Hypothesis 2: Finite element models, particularly incorporating contact elements, can provide a reliable means of representing impact echo systems for such purposes as generating neural network training data.

1.2. Background

The decay of reinforced concrete in buildings and structures has become a very serious problem in the developed world. Acid rain attack, chloride contamination, poor design and lack of quality control during the construction phase are principle agencies in this. Considering reinforced concrete in the UK alone, there are estimated to be 4,500 tall buildings, 90,000 road bridges and 5,000 storage vessels, all needing routine inspection for maintenance. Because of the substantial shortfall in funds available for essential repair of these, it is imperative that dependable priorities can be set. The provision of productive, reliable and economic non-destructive testing (NDT) methods are crucial in this.

Achieving effective inspection capability, through integration of robots, with state-of-the-art access systems, is a 'construction automation' objective adopted by the Construction Robotics Centre. The 'CURIO 1' (City University Remote Inspection

Operations) robot is a product of this approach (Chamberlain & Bleakley, 1994). This is particularly applicable to tall buildings, chimneys and structures, where suspended access systems are effective. In figure 1.1, it is shown undertaking NDT work on a reinforced concrete wall. Unlike typical factory robots, CURIO has a high payload to self-weight ratio. The tool manipulating capacity is currently rated at 20 kg, sufficient for most commonly employed NDT equipment (Chamberlain, 1994). However, with structural modifications and increased motor power, coring and similar heavy-duty operations could be performed. Furthermore, it is also interesting to note that the overall payload is sufficient to accommodate a tool/probe changing facility. A NDT method that has been adopted for delivery by the robot, sub-surface defect determination by impact echo response. Further to this, the use of ground penetrating radar systems is an attractive prospect. However, the combination of high costs, interpretation skill requirements and limited resolution remain barriers to wide usage.

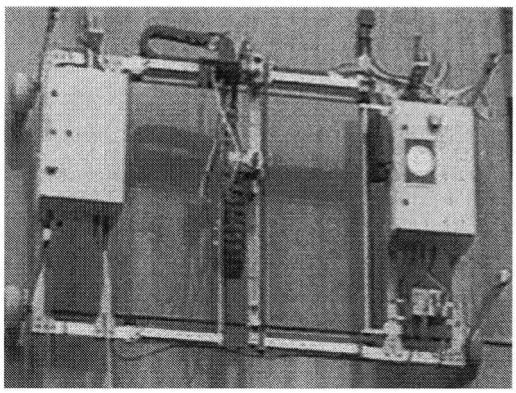


Figure 1.1 CURIO 1 carrying out NDT inspection

1.3. Objectives

In the development and application of NDT methods by robot, it is important to recognise the advantages of precise position and motion control. Considering re-bar detection, for example, the waveform (position and value) derived by scanning a surface can be more informative than a set of isolated readings (Chamberlain, 1992). A further example is radar, where accuracy is improved through position and motion control of the antenna. In fact, to take full advantage of the robotic application, it is advantageous not to think of the robot as simply a means of replicating manual survey activity. Here, the ability to simulate NDT experiments is useful in designing and evaluating methods and procedures suitable for robotic application. Advancing impact echo system technology represents a contribution to this, through added intelligence.

The objectives of this research are thus to:

1. conduct impact echo experiments to provide valid data for analysis.
2. establish a suitable artificial neural network for data analysis giving defect prediction.
3. use the impact echo data to train and verify the neural network approach.
4. optimise the neural network performance.
5. build a simulation of the impact echo system using finite element modelling as a means of achieving valid data sets.
6. Use a contact element to model a defect interface and understand its possible contribution towards model accuracy.

1.3.1. Impact echo method

Impact echo equipment comprises an impact device, a surface displacement transducer and an echo signal analyser. The transducer is located at close proximity to the impact device, as shown in figure 1.2. Chapter 2 discusses the typical arrangement of this equipment. With early forms of the equipment, the interpretation was based on the time of flight (tof) of the reflected P- stress wave (Lin & Sansalone, 1992).

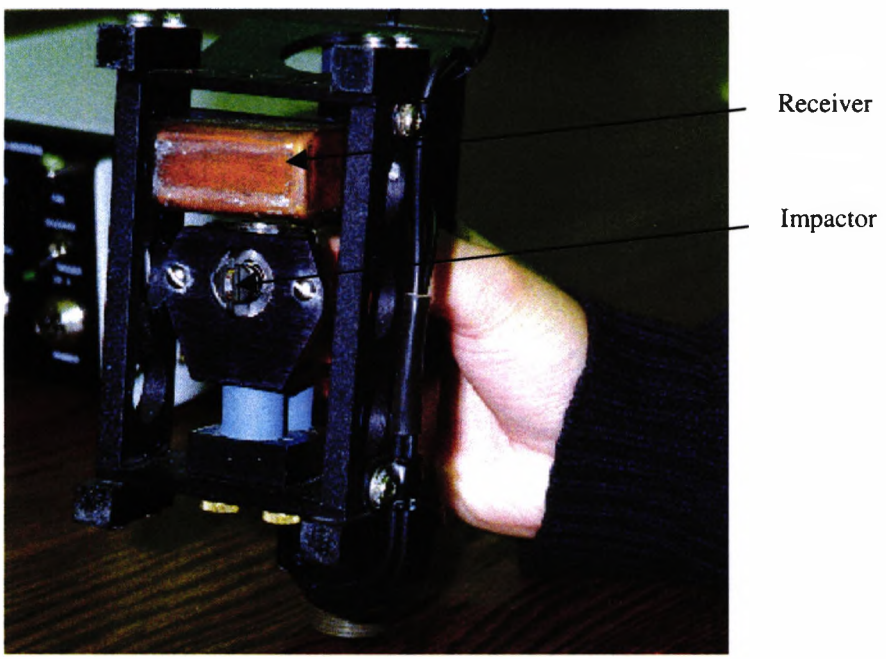


Figure 1.2: Impact echo device

However, it is now invariably based on frequency analysis of the surface displacement wave form. Fortunately, for this, high frequency analogue to digital conversion can now be achieved using low cost personal computer compatible hardware. Of the NDT

methods adopted for robotic delivery, it is the least established in general survey work, being often associated with insitu pile testing (Stain, 1982) and similar integrity investigations where low frequencies are more relevant. The equipment developed for these tasks is unsuitable for general survey, because the upper limit of the transducer frequency band is only 20 kHz-25 kHz. For the detection of shallow interface features, equipment offering significantly higher receiver frequencies needs to be used.

This is easily understood by considering a stress pulse which is propagated into a concrete component, reflected at an interface feature and then received by a nearby transducer. Figure 1.3 shows the problem investigated in this research. The corrosion of the reinforcing bars causes delamination. The depth of the defected area around the vicinity of the reinforcing bars are about 30 to 50 mm.

Assuming the stress wave speed ($C_{p)}$ to be 4000 m/s and the depth of the reflecting interface (d) to be 40 mm (typical reinforcement cover delamination), the corresponding tof is 20μ secs. In this case, the surface wave form would include 50 kHz components. For accurate analysis, a sampling rate of more than 250 kHz-500 kHz is clearly indicated.

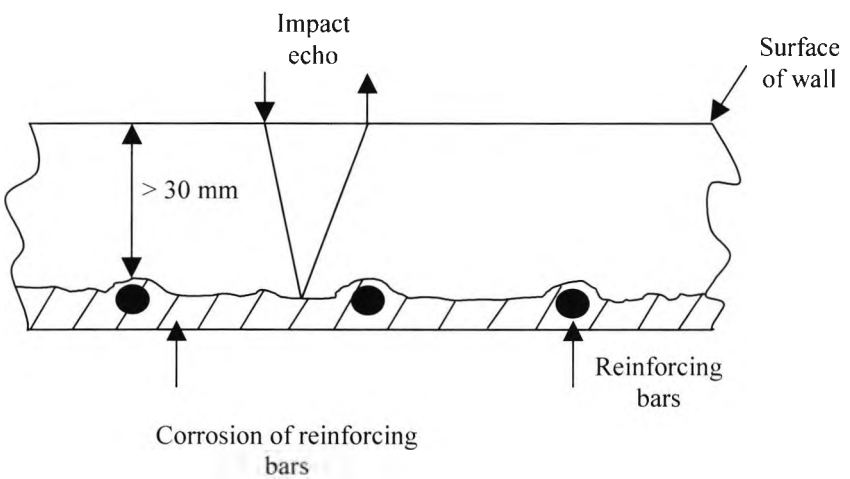


Figure 1.3: Illustration of a defect in reinforced concrete

The force-time history of the contacted impact device approximates to a half sine curve, the duration of which directly influences the frequency spectrum of the input pulse. A short pulse favours the detection of shallow interface features rather than deep ones, however, the amplitude of each component frequency is lower.

The impact echo device was not readily available during the early days of the study to detect this type of frequency. In despite of this, the author carried out some preliminary investigations using a hammer and a digital audio tape(DAT) recorder. The hammer was used as an impactor and the DAT recorder used with a microphone to record the sound generated from the hit on the concrete surface.

The experiment was carried out in a laboratory of the civil engineering department. An area of visible cracked surfaces, hollow and sound concrete were used. Figure 1.4 shows an example of a surface tested. ' Tap10 ' is the name given for 10th hit on the concrete surface.

The sampling rate for DAT was 44 kHz but the actual sampling rate used was 20 kHz since the maximum frequency of the microphone was 20 kHz. The raw and FFT data are shown in figures 1.5 and 1.6. It can be seen from figure 1.6 that the maximum frequency response seems to less than 10 kHz. As discussed previously, the sampling frequency needs to be about 250 kHz or a minimum of 100 kHz.

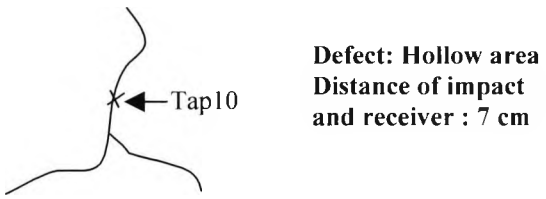


Figure 1.4: An example test on concrete

Other methods were considered, radar and ultrasonic whose sampling frequencies are in the M Hz range, however, these were not possible because they were too expensive to purchase. In fact, a sensitive transducer with a wide range of frequencies was not produced in the U.K. at the time, so to replicate the impact echo equipment was not an option, indeed the preliminary experiment described, proved this.

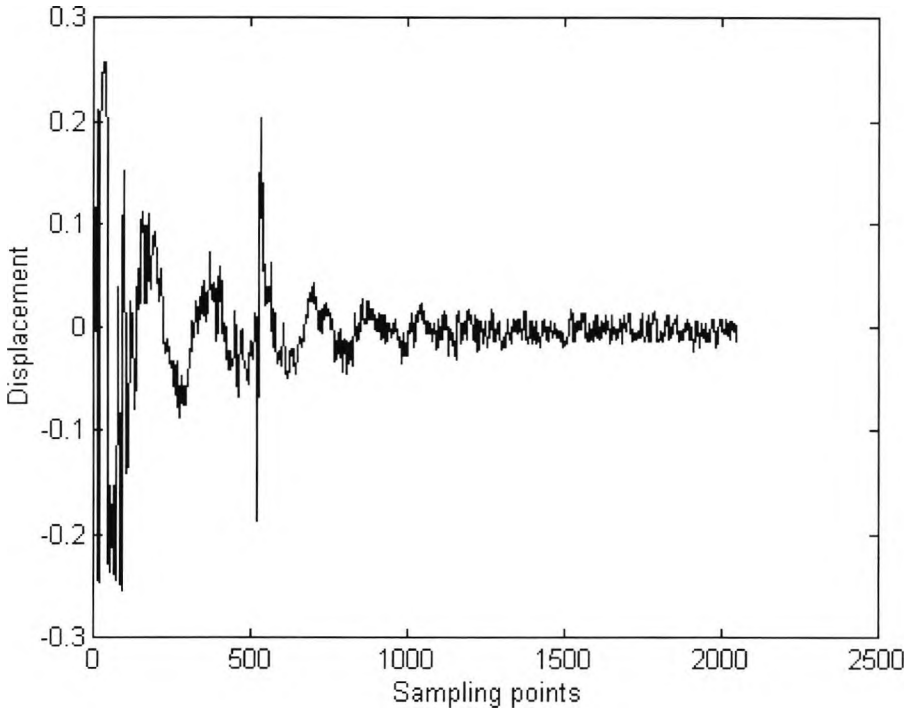


Figure 1.5: Waveform of the response of hammer on concrete in time domain

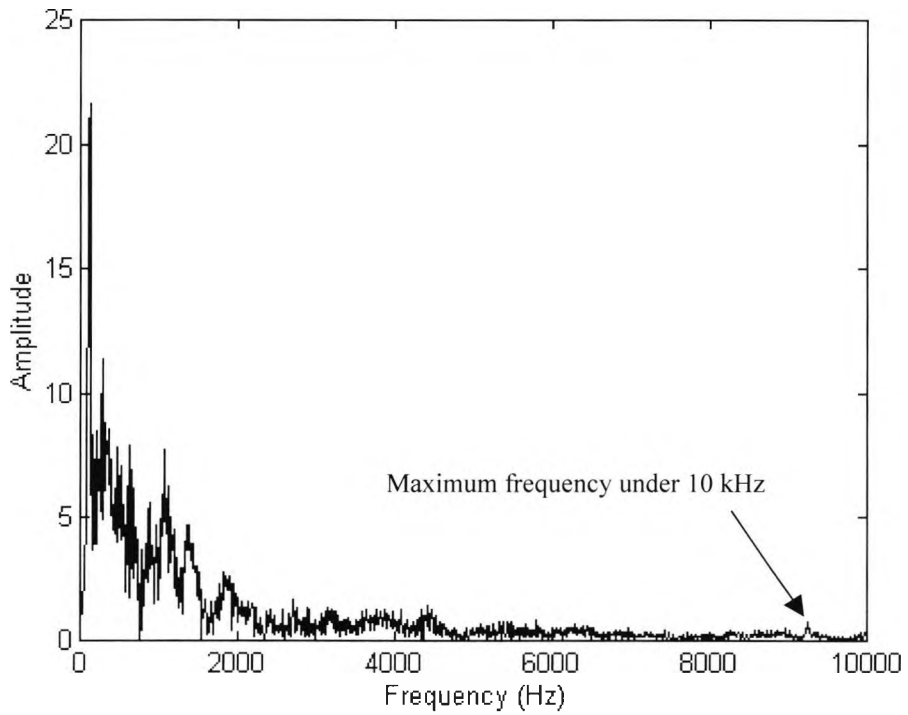


Figure 1.6: Amplitude spectrum of the response of hammer on concrete in frequency domain

Also, a 'Voidscan', which consisted of an instrumented delaminated device meter and a multi-head impact hammer, was considered, which was capable of detecting voids, delaminations, cracks, honeycombing and duct systems (Hammond, 1992). Unfortunately, this equipment is no longer available in the U.K. Finally, the author travelled to the U.S.A to obtain a realistic set of impact echo results, using walls with manufactured defects, not specifically designed for impact echo investigations.

Once, the impact echo results were obtained, a suitable network had to be devised. This was done by examining a variety of neural networks, described in chapter 5. To achieve additional data for neural network training, it was decided to use finite element modelling. The model of the impact echo was not so simple as initially thought. Although this method was excellent for producing data at low cost, processing times were extremely long, up to seven days. The finite element software also consumes large amounts of hard disk space. This was easily solved by increasing the hard disk space, however, it still did not reduce the amount of time to solve a particular problem. A 500 element model of 3D would take about 3 days to complete for example. In fact even when a 1/4 of a model was built, this had no effect on the time to solve. The author has looked into Supercomputers, but this meant having to use UNIX on the University computer, which meant competing with other students for computer use. Also, obtaining the results took a considerable amount of time after the solution was complete. So, greater efforts were made on simulating 2D models rather than 3D, because the 3D model requires longer time to process.

1.3.2. Echo analysis

The periodic nature of the reflected P-waves is exploited in the analysis of the echo signal. Fast Fourier transformation (FFT) is used to investigate its content in terms of frequencies and their corresponding amplitudes.

1.3.3. Finite element modelling

To support prediction and additional data requirements for a range of defect cases, a modelling tool is required. The finite element method has previously been used to model impact echo systems (Cheng & Sansalone, 1991). In the work undertaken by the author, 3 types of elements are used:

- 2D, 8 node quadrilateral
- 2D, 2 node contact element
- 3D, 20 node

These are described in chapters 8, 9 and 10. The 2D quadrilateral and 3D elements are used to define the form, material composition and interface content of the material structure. Use of the interface elements allows a 'no-tension' regime, with contact forces developing on crack closure. In this research, the contact element is used to accurately represent a crack; the importance of this is described in chapter 9. However, due to the time limit, finite element data was not used in the neural network analysis.

1.3.4. Neural network interpretation

Use of a neural network was most suited to this application due to the use of large amounts of data. Its primary role is automating data that is complex and not amenable to conventional analysis. Considerable experience is required in interpreting impact echo data, unlike commonly employed NDT methods. The neural network is used as a tool in assisting with the understanding of the response obtained from such a system, ideally giving reliable information such as defect depth.

An artificial neural network is a system of inter-linked, basic data processors. These are not programmed in the usual serial way of relating data to decisions. Rather, using cases of inputs and outputs, the network is assigned the computational task of building an appropriate decision model. Of the many different types of networks, the back-propagation network (Caudill, 1990) is the most commonly used. Furthermore, it is ideally suited to classification problems like the problem in-hand. Chapter 7 shows the structure of the network, which comprises input nodes, hidden nodes and output nodes. The input nodes are the amplitudes at equidistant sampling points on the amplitude-frequency spectrum. On the output side, there is a single node holding the theoretical frequency of the defect interface. It is noted that equal spacing on the input frequencies (amplitudes) do not give equal spacing for the inverse of these, which relies more significantly to depth.

To train the network, it is necessary to present a sufficient number and range of input and output data sets to the network. These progressively tune the weights applied to the interconnecting nodes to give the solution algorithm. Convergence error indicators are used to follow the progress in this. Once trained, the network can be used to classify new input patterns with great efficiency.

The back propagation network was concentrated on to analyse the impact echo data. Time did not allow further investigations of other neural networks, learning rules and transfer functions, which are described in chapter 5.

1.4. Structure of thesis

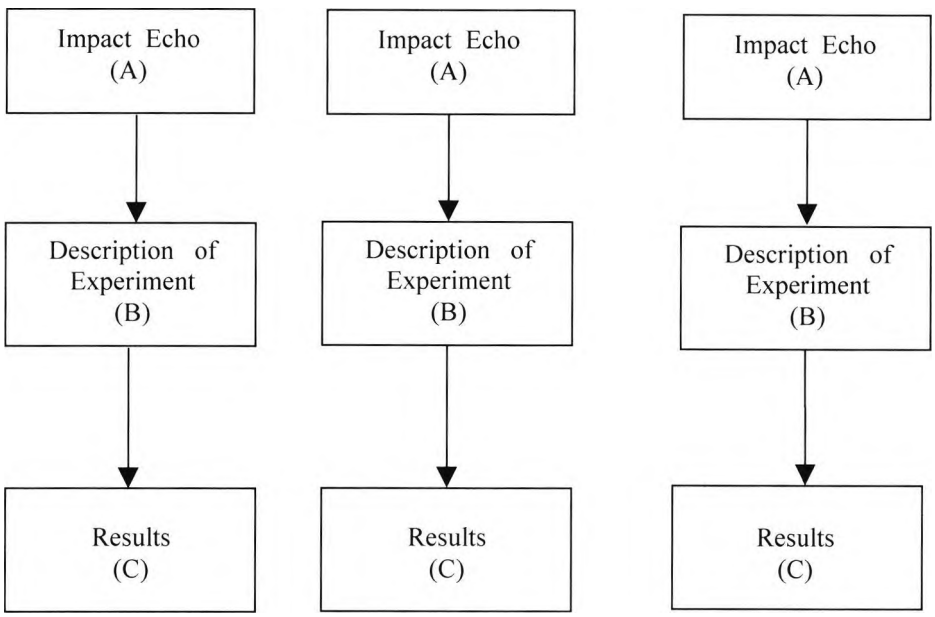


Figure 1.7: Structure of research

The thesis has three parts, which cover the following:

- Impact echo theory and application
- Neural network modelling studies
- Finite element simulation studies

Each part begins with the corresponding theory, followed by discussion of the application, description of experiment, and concluding with discussion of results. This is illustrated in figure 1.7, A, B and C indicating the order in which they are expressed.

Two parts of the thesis are related to each other in the following manner (i) the impact echo technique is modelled using neural networks, and (ii) the finite element method is applied to simulating impact echo cases. This relationship is shown in figure 1.8.

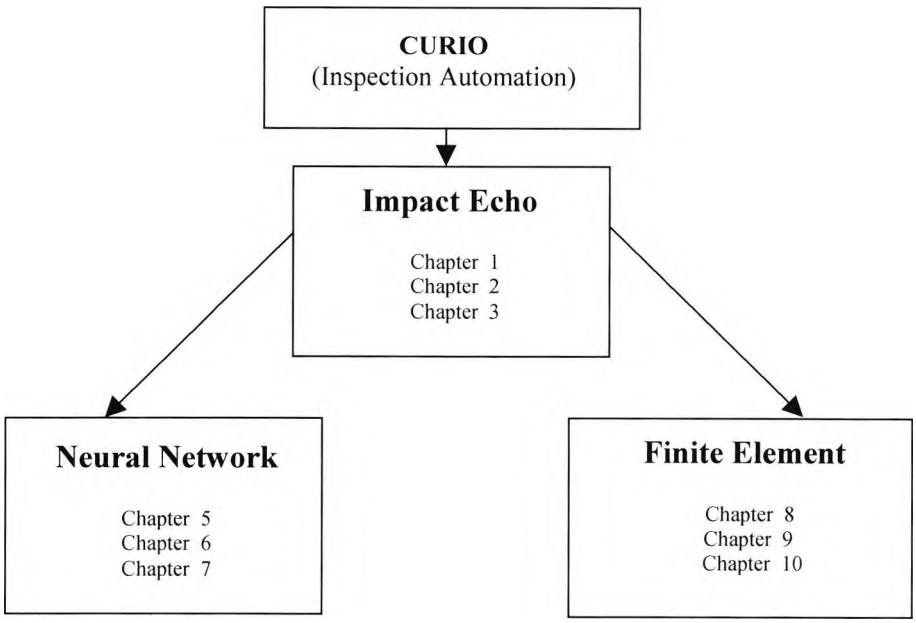


Figure 1.8: Relationship between chapters

1.5. Overview of state-of-the-art

There have been numerous papers presented on the impact echo method used on concrete. For example, the impact echo equipment was applied in concrete slabs with and without overlays (Sansalone & Carino, 1989) to detect delaminations. It has been used to find voids in grouted ducts (Carino & Sansalone, 1992), the results found to be encouraging.

Simulation of impact echo data, using the finite element method has been presented, by researchers, where a 2D model (Sansalone, Carino, & Nelson, 1987; Sansalone & Carino, 1990; Lin, Sansalone & Carino 1990) with a 8 node plane strain element was built and tested. A 3D model has been created, this containing an 8 node solid element (Cheng & Sansalone, 1991). Also, a finite element investigation into the use of impact echo methods for the investigation of ducted post-tensioned bridge beams has been shown (Martin, Hardy, Usmani & Forde, 1995). The results from all these models were fairly accurate, when compared to practical results of the impact echo method. However, these numerical methods have not used the interface element to model a void. This is investigated in chapter 9 to see if contact elements can make a contribution to achieving better simulated data for impact echo and neural network analysis.

There has been a research on t he use of neural networks, using the back propagation network on impact echo data (Pratt & Sansalone, 1991). However, these researchers

have used 150 input nodes with 200 training patterns. This means that the network has not been trained adequately to deliver significant results (Masters (1993:246-251)). This is discussed in chapter 7, under Neural Network Results. The neural network method has been applied in ground penetrating radar applications in pavement evaluation and assessment (Attoh-Okine, 1995). Also, the use of neural computing on impact damage detection in carbon fibre composites using acoustic emission has shown to be more effective (NCAF, 1999).

Following the structure of the thesis, detailed literature reviews are reported in table 1.1

Chapter	Topic
2	Impact echo theory
5	Neural network theory and applications
6	Back propagation theory
8	Finite element theory
9	Contact element theory

Table 1.1: Literature reviews

1.6. Summary

The purpose of this research are:

1. To carry out impact echo tests to get a variety of data for neural network analysis.
2. Real specimens are expensive and a large number, and variety are necessary to give sufficient basis for neural network training and verification. Because of this, it was decided to investigate means of obtaining simulated data i.e. use of finite element modelling.
3. Pure, noise free signals, compared with real signals are achieved by finite element modelling. The predicted idea is that real signals show the influence of the contact transfer with crack closure. For this reason, contact element incorporation was investigated to determine if more realistic signals could be achieved. This represents a possible improvement over finite element models previously produced (Cheng & Sansalone, 1991).

Chapter two

Non-destructive testing and impact echo

2.1. Introduction

As introduced in chapter one, the importance of non-destructive testing of concrete is considerable in order to achieve effective maintenance of the concrete structures. This requires assessment methods that can identify the initiation of defects so that corrective actions may be taken to prevent large-scale deterioration. This chapter reviews a variety of NDT methods, followed by a summary of these. It then focuses on the impact echo technique, for which the theory and application is presented. Finally, the two types of impact echo devices available are compared. This provides the backdrop for the following chapters, which take up the impact echo investigation.

2.2. Non-destructive techniques

In the following sections, the commonly employed NDT methods are briefly reviewed. A summary of this is given in table 2.1. The purpose is to convey a sense of the range of NDT equipment appropriate for broad defect detection. Delaminations, commonly associated with the expansive action of near to surface steel reinforcement, are the main target.

2.2.1. Pulse echo

This technique involves a hammer or heavy chain, which strikes the concrete surface. If a delaminated area is struck, a hollow sound is heard, which is different from the ringing sound produced by a solid area. Although this method has been widely used, it has been found to be time-consuming, and relies on subjective judgement of an operator to interpret the noise signal. Moreover, while this mechanical device gives a rapid test, soundings becomes less reliable as the depth of delamination increases or asphalt concrete overlays are present (Manning, 1985).

2.2.2. Infrared thermography

Infrared thermography detects delaminations by observing the variations in surface temperature with an infrared camera. A delamination introduces an air gap into a slab.

This air gap acts as an insulator that restricts heat flow into and out of the slab. So, during conditions of heat flow, the surface temp is affected and local hot or cold spots are created above the delamination. For passive excitation, however, this depends on having a reliable heating cycle. In all cases, data interpretation can be complicated by local variations in surface properties (emissivity), which can be interpreted incorrectly as temperature differences (Kunz & Eales, 1985). Active excitation, using heat sources, would require considerable power and long heating cycles.

2.2.3. Radar

This method uses an antenna to introduce pulses of high frequency electromagnetic waves into the concrete. As a result, this pulse is reflected by interfaces between materials of different dielectric properties, the reflections measured by a receiving antenna. The received signals are then displayed as a function of time.

It is possible to detect the presence of interfaces as well as depth in concrete. However, because of reflections from many interfaces, the received signal can be complex and considerable experience required interpreting the radar data (Clemena, 1983). Calibration or knowledge of the dielectric properties is necessary, a problem encountered frequently for the application.

2.2.4. Ultrasonic pulse velocity

This is the measurement of the transit time of an ultrasonic pulse between a transmitter and receiver. The velocity of the pulse can be evaluated if the distance between the transmitter and the receiver is known.

When testing concrete, a coupling agent, petroleum jelly for example, is applied to achieve this. The velocity of sound waves through the concrete is minimised by the presence of voids or cracks (NCHRP, 1985). By utilising relationships between speed of sound and the strength of concrete, the time of flight data can be interpreted in terms of strength. Dimensions must be known, however, a problem in many situations.

2.2.5. Remote viewing inside a structure

Remote viewing (endoscope) can often be the only option to monitor areas of a structure where access is limited. The hardware used is fibre optics, video cameras and periscopes. The fibre optics technique uses a bundle of glass fibres that transmit light on the defect being viewed. The images are then transmitted back to a lens for viewing

by eye or camera (Emmons, 1994). However, defect size must be of the order of several mm before insertion is possible.

2.2.6. Radiography

A beam of gamma rays is directed through the concrete towards a film held against the opposite face of the test member. This provides a photograph of the interior of a concrete member, showing variations in density, which relates to strength and durability. Voids can be located with this method (Bungey, 1992)). Equipment is expensive, heavy and must be operated by a qualified radiographer. Interpretation of images requires considerable skill.

Method	Principal Applications
Pulse echo	Thickness assessment and location of delamination or major voids in slabs. Pile integrity testing
Infrared Thermography	Location of delamination, moisture, major ducts or voids in slabs or walls
Radar	Location of voids, cracks, moisture, reinforcing bars within concrete elements and voids below ground slabs. Estimation of slab thickness
Ultrasonic pulse velocity	Uniformity of concrete quality and strength, location of internal defects
Remote viewing inside a structure	Detection of voids, delaminations and location and conditions of embedded metals.
Radiography	Location of internal poor compaction, voids and reinforcement
Impact echo	Location of voids, cracks, delamination, reinforcing bars within concrete structure. Determination of slab thickness

Table 2.1: Summary NDT methods and their principle applications

2.3. Impact echo method

The impact echo method is a non-destructive measuring method using stress waves. It was created by Cornell University and the National Institute of Standards and Technology in the United States (Sansalone & Carino, 1986). It can be used to detect hidden damage and determine the extent of this within a concrete section, delaminations for example.

Stress wave propagation has been applied in non-destructive testing of concrete. However, these attempts have not shown much success. Nonetheless, some progress

has been made in measuring the thickness of plate elements and in integrity testing of rod like structures, piles for example (Stain, 1982). Progress has been hindered because of the heterogeneous nature of concrete, which strongly attenuates high frequency waves. So, traditional wave propagation methods employed for flaw detection in metals cannot be used for evaluation of concrete (Sansalone, Carino, & Nelson, 1987). Emmons (1994) has stated that detecting flaws under the surface of concrete has always been complicated, and the impact echo equipment has provided a reliable method for finding defects. Indeed, one of the advantages of the impact echo method is that it only requires access to one side of the test member, whereas, with the ultrasonic pulse velocity method, for example, access to both sides are needed.

Although the impact echo is a straightforward method to apply, successful interpretation of the displacement waveform needs understanding of the interaction of transient stress waves with internal defects, providing advances can be achieved in the interpretation intelligence, the purpose of this research. The method has clear advantage over other NDT methods. In the next section the theory of this impact echo technique is presented.

2.3.1. Comparison of two impact echo devices

The two impact echo devices commercially available, which are used most widely, are OLSON and DOCter (Defect Orientation Confirmation Tester). These devices are similar to one another and are capable of locating defects for a typical reinforced cover delamination, discussed in chapter 1. As shown in figure 2.1 and figure 2.2 both devices are easy to use, but the OLSON device is more comfortable to handle. It can be applied horizontally and vertically, whereas, the DOCter device has been designed to be held horizontally, indeed to test a vertical wall. A special case had to be made to adapt to the test structure (Carino & Sansalone, 1992), figure 2.3 shows the DOCter in use. However, the DOCter has been commercially available longer than the OLSON. A comparison between these apparatus is summarised on table 2.2.

Figure 2.1: OLSON impact echo device

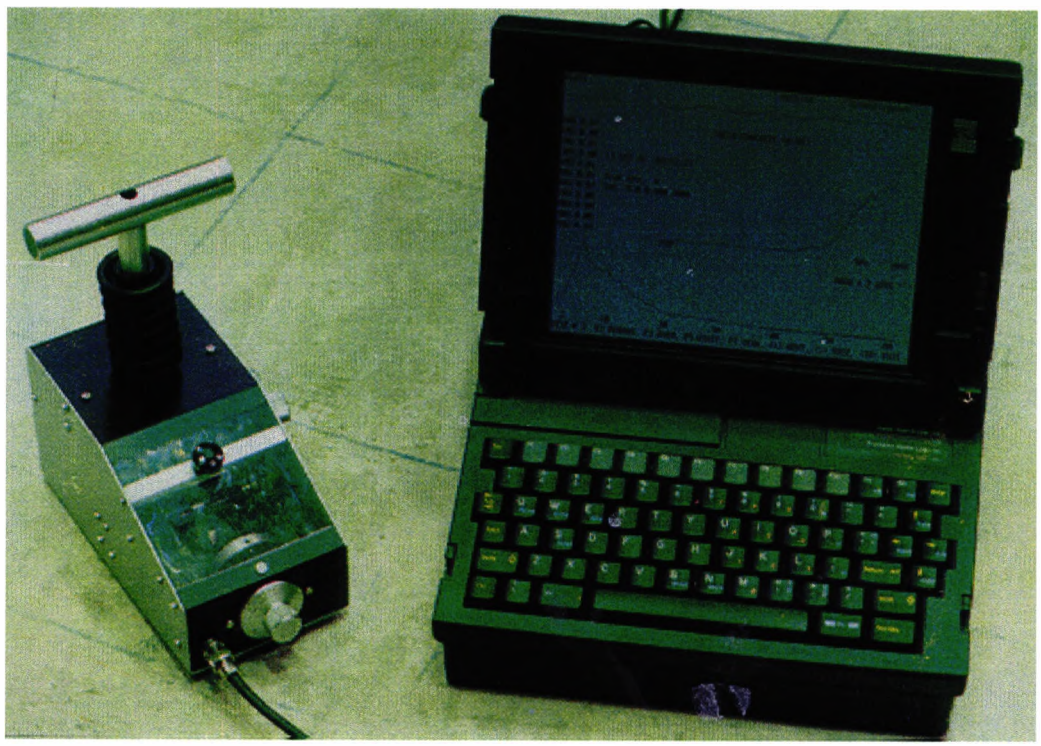


Figure 2.2: DOCter impact echo device

Impact echo device	Impact echo device
OLSON system	DOCter system
Neural network software not used in the impact echo device.	Neural Networks software is integrated in the impact echo device
Rugged System Resist dirt, abuse etc.	Rugged System Resist dirt, abuse etc
Portable	Portable
Very easy to use	Very easy to use
Integrated Impactor/Receiver Unit	Integrated Impactor/Receiver Unit
Cost: 22,000 U.S.A Dollars	Cost: 150.300 DKK
Sampling Frequency used in experiments: 1MHz	Sampling Frequency used in experiments:1M Hz
Source of impact: Instrumented hammer or impactor using solenoid of various size to generate particular frequency range	Source of impact: Impactor using ball bearings of various size to generate particular frequency range
Receiver : Displacement Transducer used	Receiver : Displacement Transducer used

Table 2.2: Comparison of DOCter and OLSON impact echo devices

Figure 2.3: DOCter system in use

2.3.2. Theory

In this method, a stress pulse is introduced into an object by mechanical impact on its surface. This pulse travels into the solid as spherical compression (P) and shear (S) waves. Also, a Rayleigh (R) wave is generated, which propagates along the surface of the object. These wave characteristics are summarised in table 2.3. The P- and S- waves are reflected by the solid/air interface and by internal defects of sufficient size. For a plate, for example, multiple reflections (echoes) occur as the waves travel back and forth between the top and bottom of the object surfaces. This is indicated in figure 2.4(A).

When a P- or S- wave hits a boundary at oblique incidence, a reflected P- and S- wave are created by mode conversion (Kolsky, 1963). Arrival of reflected waves at a surface where an impact was generated, produces displacement measured by a transducer placed in the vicinity. If the receiver (transducer) is located close to the impact point, the displacement caused by P-wave arrivals are detectable. This is because the normal displacement caused by the P-wave is larger than the displacement caused by the S-waves.

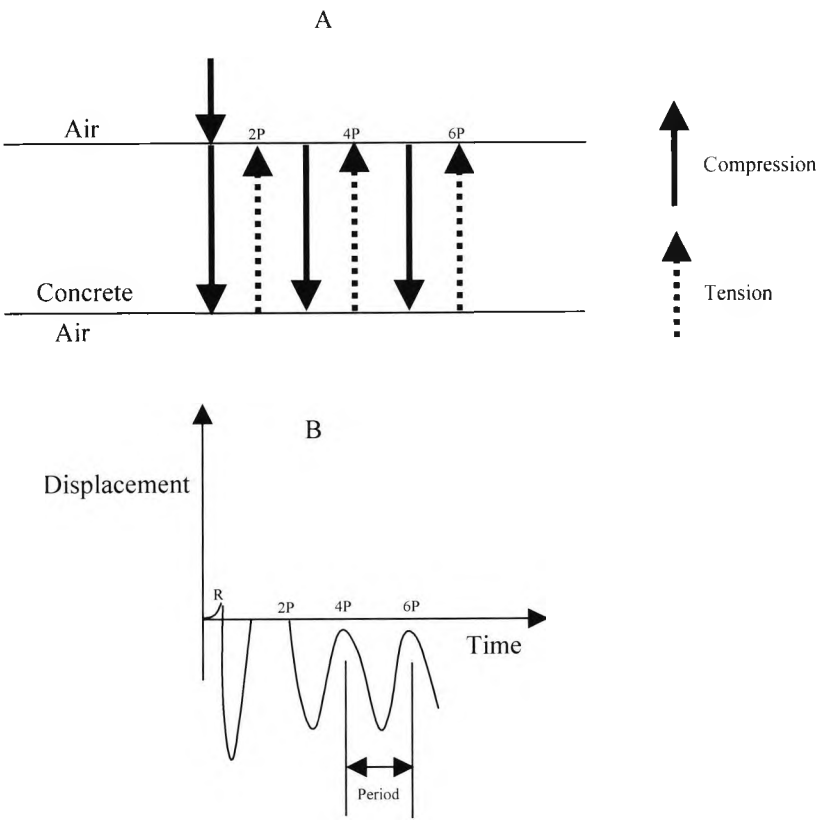


Figure 2.4: Reflection of impact echo waves (Carino & Sansalone, 1986)

Figure 2.4 (B) shows the normal displacement at a point on the top surface of an infinite plate, caused by an impact at a nearby point on the same surface. This response includes displacements caused by the arrival of the R-wave moving along the plate surface with the arrival of multiply reflected P- and S- Waves and mode converted waves. For simplicity S-wave and mode converted waves are not shown (PS, 3PS,....).

The R-wave response is denoted by a R. and P-wave arrivals are denoted by 2P, 4P and 6P. 2P refers to a P-wave that has undergone a single reflection from the bottom of the plate, so it has travelled about the twice the plate thickness. The 4P denotes a P-wave that gone through two reflections from the bottom surface of the plate etc. (Carino, Sansalone & Hsu, 1986). If the wave speed C_p in the concrete is known, the distance T, to reflecting interfaces can be determined as (Sansalone & Street, 1998):

$$T = \frac{C_p}{2 f_p} \qquad \text{Equation 2.1}$$

where f_p is the detected corresponding frequency.

Rearranging equation 2.1 in terms of frequency gives:

$$f_p = \frac{C_p}{2 T} \qquad \text{Equation 2.2}$$

P-waves	S-waves	R-waves
Body wave	Body wave	Surface wave
Dilatational, this is a compression wave	Distortional, this is a shear wave	This is a Rayleigh wave
Particle motion is parallel to the direction of propagation	Its motion is perpendicular to the direction of propagation.	Wave travels along solid surfaces
P-wave spherical when stress waves are generated by a point source applied normal to the top surface of a plate	S-wave spherical when stress waves are generated by a point source applied normal to the top surface of a plate	R-wave cylindrical when stress waves are generated by a point source applied normal to the top surface of a plate
P-waves can propagate in all types of media	S-waves can propagate only in media with shear stiffness i.e. solids	In a solid/liquid interface R-waves can propagate along the interface.

Table 2.3: Comparison of P-, S- and R- waves

2.3.3. Method of application

The impact echo equipment consist of three components:

- Impact source
- Displacement transducer
- Waveform analyser

A typical layout of the apparatus is shown in figure 2.5.

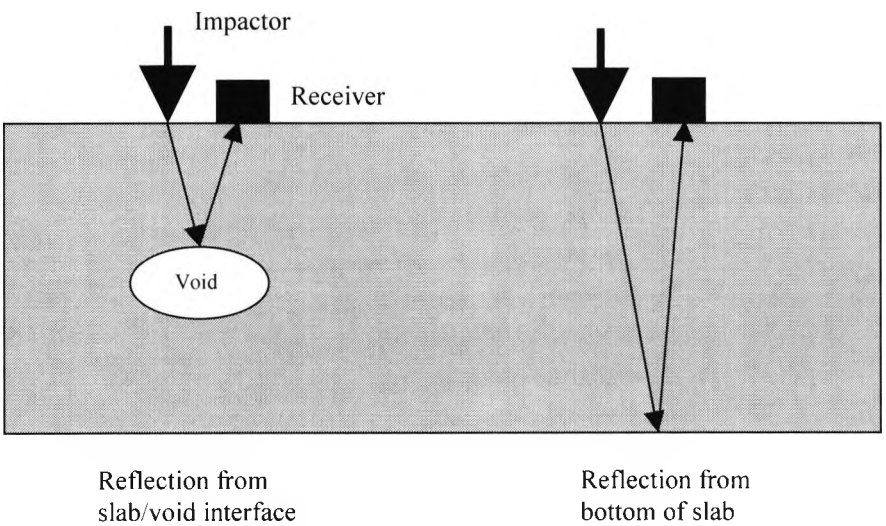


Figure 2.5: Impact echo test configuration

2.3.3.1. Impact source

The impactor provides a known or controlled input force to the structure. The duration of impact depends on what range of frequency is required for detection of defects. The force-time history of the contacted impact device approximates to a half sine curve. The duration of the impact is the contact time, which directly influences the frequency spectrum of the input pulse, shown in figure 2.6. A short pulse favours the detection of shallow interface features rather than deep ones. Analysis of the input spectrum with hardened steel ball impacts shows that contact times as short as 30μ sec is achievable on dense concrete (Carino & Sansalone, 1992). From this, it is confirmed that a transducer with a frequency response of up to 50 kHz is appropriate.

The contact time of the impact depends on (Sansalone & Carino, 1989) the size and mass of the impactor as well as depending on the condition of the concrete surface. The contact time will be shorter on a smooth surface than on a rough surface for example.

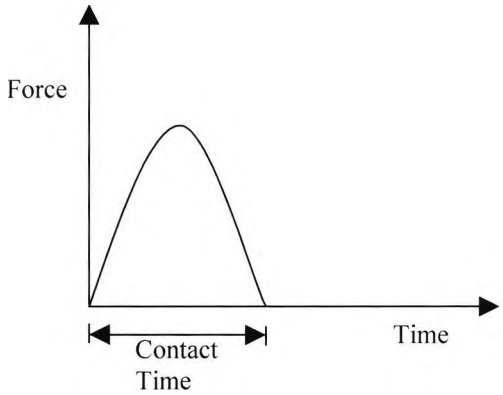


Figure 2.6: The force-time history of the impact pulse

3.3.3.2. Displacement transducer

The receiving transducer converts the mechanical motion of the structure into an electrical signal. This is capable of accurately measuring surface displacement, is very sensitive and able to respond to a wide range of frequencies.

3.3.3.3. Waveform analyser

A waveform analyser is used to capture the transient output of the displacement transducer. It stores the digitised waveform and performs signal analysis.

2.3.4. Signal analysis

The interpretation of the measured waveforms was performed in the time domain initially. This was done by calculating the time between the start of the impact and the arrival of the first P-wave response. Whilst it was adequate, it was extremely time consuming, and was overcome by interpreting the data in the frequency domain using Fast Fourier Transform (FFT) methods (Sansalone & Carino, 1986). This transformation breaks the waveform down into a series of basic sine functions, resulting in an amplitude spectrum, which is a graph of amplitude against frequency. Transient waves generated by a point source are composed of a range of frequencies, which have a number of different wavelengths. The peaks in the amplitude spectrum show the transient resonance caused by multiple reflections of waves between surfaces and internal defects.

2.3.5. Interpreting the impact response

Although the impact echo is a simple method, successful interpretation of the displacement waveform needs a basic understanding of the interaction of transient stress waves with internal defects. Figure 2.7 shows the typical transient response of

sound concrete. It is important to remember that reflection from the bottom occurs at a lower frequency than that of a shallower slab or void.

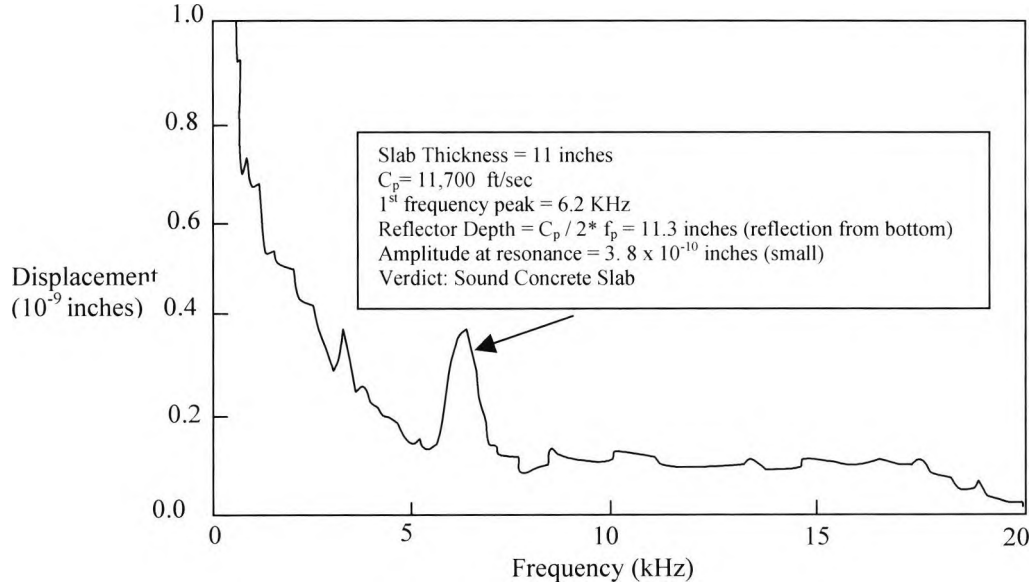


Figure 2.7: Impact echo results on a sound concrete slab (Olson, 1995)

The dominant peak indicated in figure 2.7 has a low frequency of 6.2 kHz. Using equation 2.1, this corresponding to 11.3 inches. The slab thickness is 11 inches, thus, this confirms that the impact response has been accurately interpreted.

Similarly, figure 2.8 illustrate the result of a defect. In this example, a higher frequency is expected and there is a dominant frequency at about 11.2 kHz. This corresponds to a depth of 6.3 inches, which is the reflection from the crack surface.

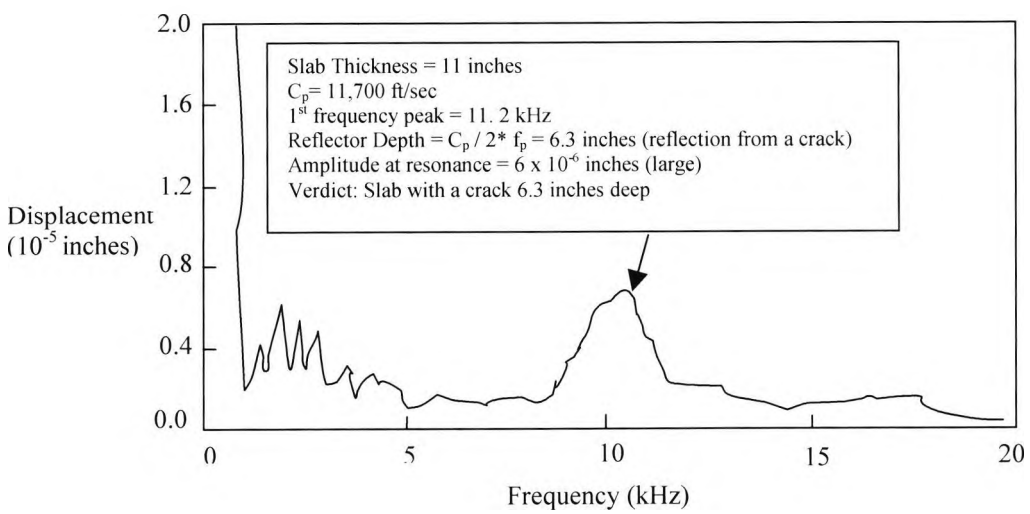


Figure 2.8: Impact echo results on a cracked slab (Olson, 1995)

2.3.6. Detectable defects

The impact echo test is carried out to monitor the conditions of slabs, beams, columns, walls, pavements, runways, tunnels and dams (Emmons, 1994). It is employed to detect voids, honeycomb, cracks, delamination and other damage in concrete, wood, stone and masonry materials. In this research, it is the detection of planner defects in concrete that is of interest.

2.4. Summary

A summary review of NDT methods has been given, which indicates the potential significance of the seldom-employed impact echo method. The governing theory of the impact echo and its application has been shown. Also, a typical response of an impact echo device on sound and delaminated concrete has been illustrated. A comparison of the two types of impact echo device has been discussed. In chapter 3, the experiments conducted with this apparatus are presented.

Chapter three

Description of experiments

3.1. Introduction

This chapter covers the application of the impact echo equipment to a series of concrete walls. The purpose of this is to obtain varied data for the subsequent investigation into neural network interpretation. Drawings are provided, illustrating the manufactured defects and the survey lines adopted for impact echo sampling. The experiment was conducted in the U.S.A, at the premises of Olson Engineering Inc. It includes photographs of the impact echo equipment discussed in chapter 2.

3.2. Description of concrete specimens

The experiment was carried out in a barn. There were four concrete wall specimens which are labelled A, B, C, D, all having a variety of built in defects. Styrofoam was used in building the voids present in walls A and D (Jalinoos, Olson & Aouad, 1993). The simulated defects are voids, honeycombs and partial delamination, involving use of a sun-screen material.

Walls A, D and C were tested. Wall A contained voids, wall D contained voids and honeycombs, and wall C contained sunscreen material.

The impact echo method was employed for testing the walls for the first time. Originally, the four walls were built for other NDT purposes (Olson, 1995). The nominal dimensions of the walls were 48x48x12 inches (122x122x30.5 cm).

The experimental equipment was set up as shown in figure 3.1. For each impact the result was displayed on the computer. The impact echo device was held normal to the wall as shown in figure 3.2. Wall A was tested on both sides (east and west), for wall D, the east side was tested, and for wall C, the west side was tested. The defected area in the concrete was clearly marked (represented by a cross), indicating where the impact echo equipment was applied. Two specimen walls are shown in figure 3.3. Figure 3.4 illustrates the direction of the impact applied on test locations.

Figure 3.1: Impact echo system

There were 15 hits on each marked cross, giving 15 sets of data. Each set had 1024 data points, which illustrate a result for a defect or non-defect. The raw data was FFT processed and saved to a file ready for other uses, namely neural network analysis. However, the FFT data could not be immediately converted into a readable form since it was in binary form. It was then decided to use Matlab (examples illustrated in Appendix F) to transform the data from time to frequency domain. In total there were 15 locations which were tested, giving a total of 225 data sets (15 data sets x 15 locations).

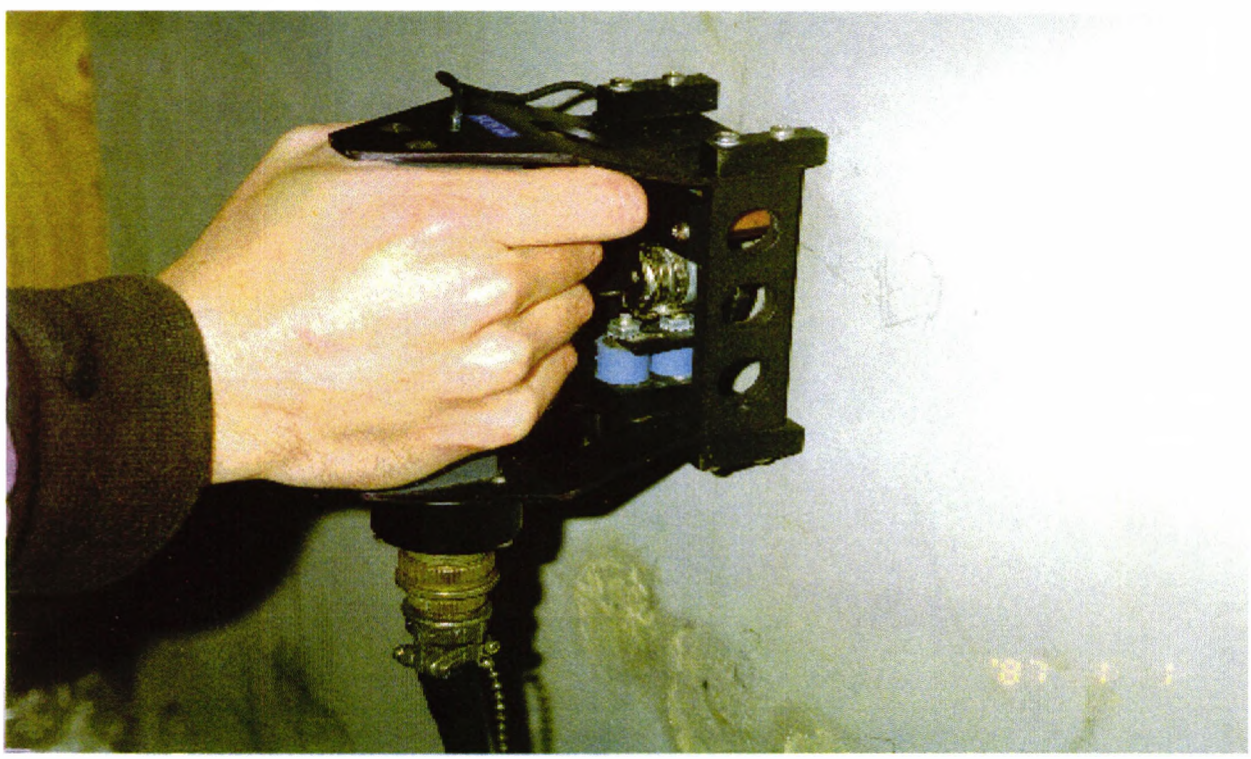


Figure 3.2: Impact echo testing on a specimen wall

Figure 3.3: A photograph of two specimen walls

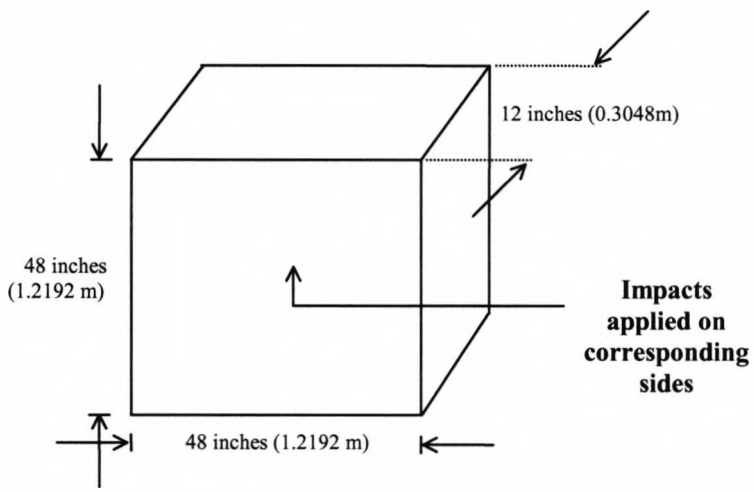


Figure 3.4: Illustration of the direction of impact

Table 3.1 list the test locations on the specimen walls. Figure 3.5 shows a 2D illustration of the tested specimen walls. It also gives the code names for the tested locations. When testing thickness, the impact echo equipment was placed in between two voids i.e. where there were no defects, for example, wacl1 and wacl2 are solid areas. Also, Wacl1 and Wacl1a were tests for the effects of pressing the impactor soft and hard at the same point. There was not any significant discrepancy between the two, so Wacl1 was used in the neural network analysis. The impact areas on the walls are indicated in figure 3.5.

Code name	East or West	Wall	Survey line (lines of the tested locations)
Wav1e	East	A	1
Wav2e	East	A	1
Wav3e	East	A	3
Wav4e	East	A	3
Wav1w	West	A	1
Wav2w	West	A	1
Wav3w	West	A	3
Wav4w	West	A	3
Wacl1	West	A	1
Wacl1a	West	A	1
Wacl2	West	A	3
Wdv5e	East	D	2
Wdv6e	East	D	2
Wdv7e	East	D	4
Wdv8e	East	D	4
Wdcle	East	D	2
Wcsscw	West	C	2

Table 3.1 List of test locations on specimen walls

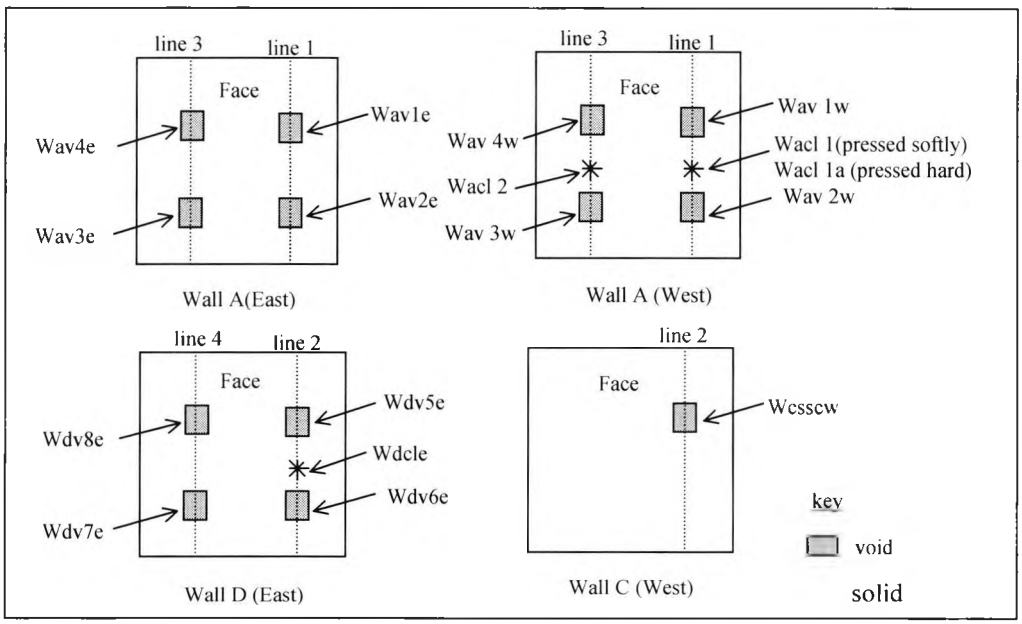


Figure 3.5: Survey lines on walls A, D and C

3.3. Survey lines on wall specimens

The sizes and locations of the defects in walls A and D are illustrated in figure 3.6 through to figure 3.15 respectively. In each figure, isometric, side, cross-section and plan views are indicated. These drawings are not to scale.

3.3.1. Wall A(east and west)

Figure 3.6 shows void1(wav1e and wav1w) with dimensions of 6in x 6in x 1in, which lies on the line 1.

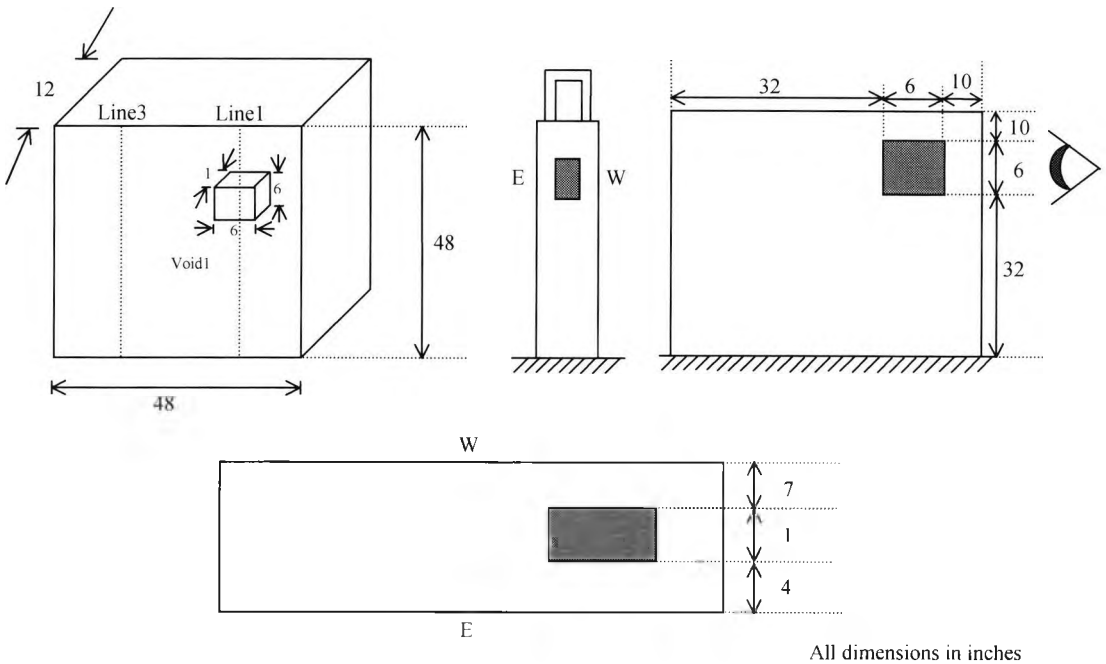


Figure 3.6: Void1

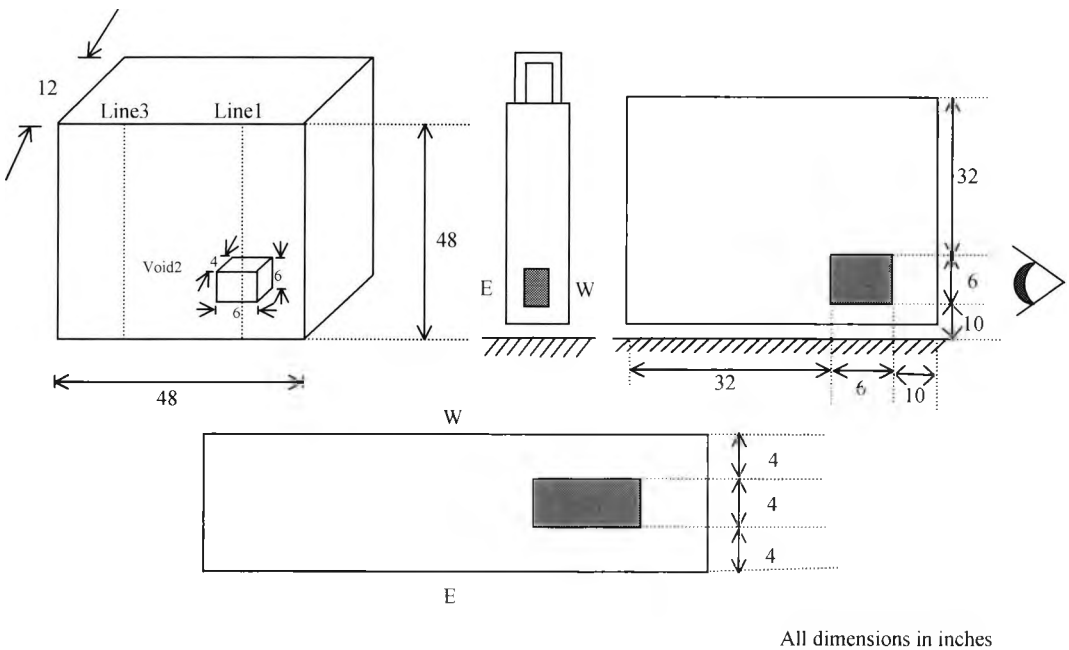


Figure 3.7: Void2

Figures 3.7 shows void2 (Wav2e and Wav2w) with dimensions of 6in x 6in x 4in, which also lies on the line 1. On line 3, are the voids 3 and 4 (Wav3e,Wav3w,Wav4e and Wav4w), having dimensions of 6in x 6in x 3in and 6in x 6in x 2in. These are shown in figures 3.8 and 3.9 respectively.

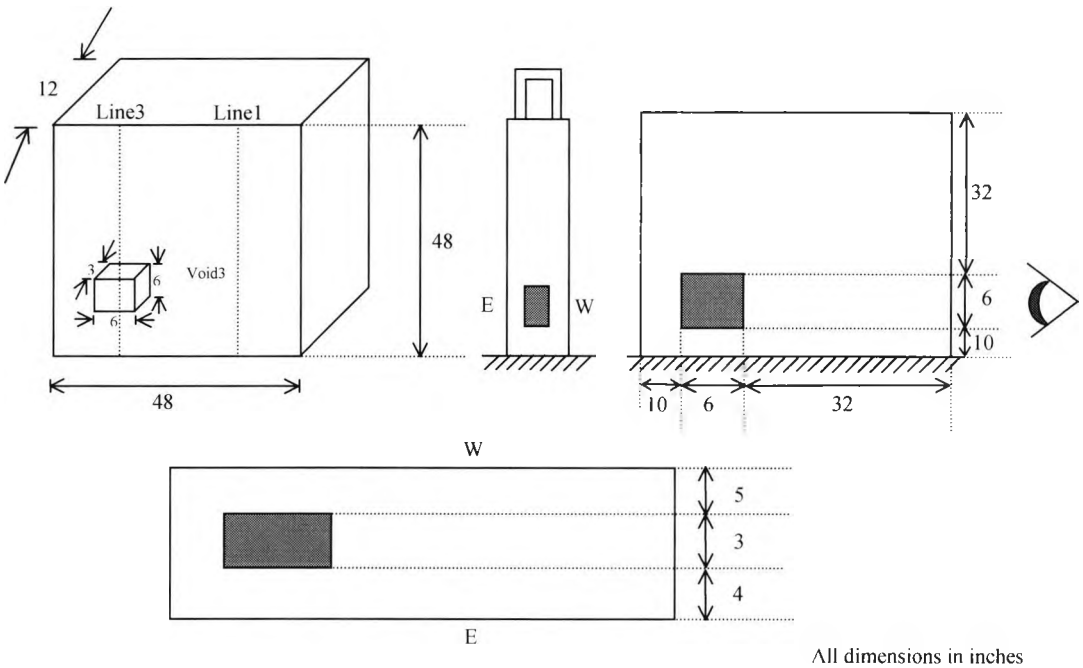


Figure 3.8 : Void3

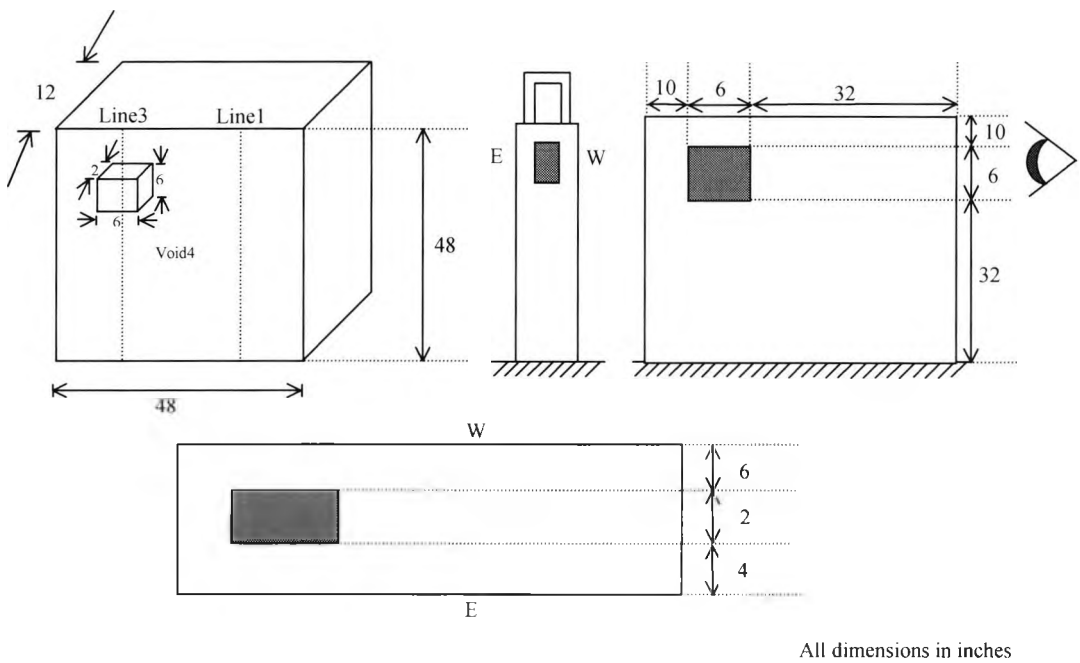


Figure 3.9: Void4

3.3.1.1. Testing of sound areas

The testing locations for sound areas are indicated in figure 3.10, solid1 (Wacl1) and solid2 (Wacl2), the dimensions for them are 48in x 48in x 12in. The west side of the wall A was tested.

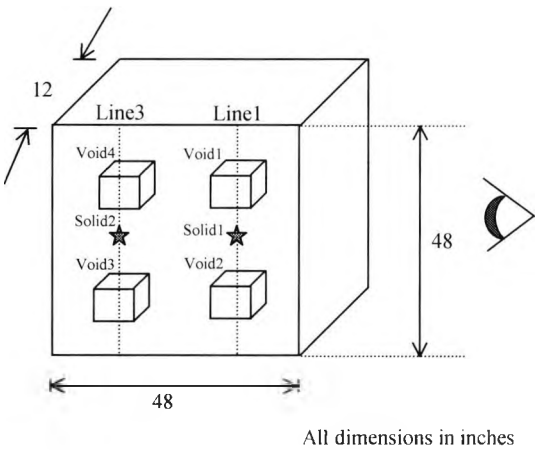


Figure 3.10: Testing of Solid1 and Solid2 (west face)

Also, the east side of wall D was tested for a sound region, solid3 (Wdcle), this shown in figure 3.11. The dimensions are the same as previously indicated figure 3.10.

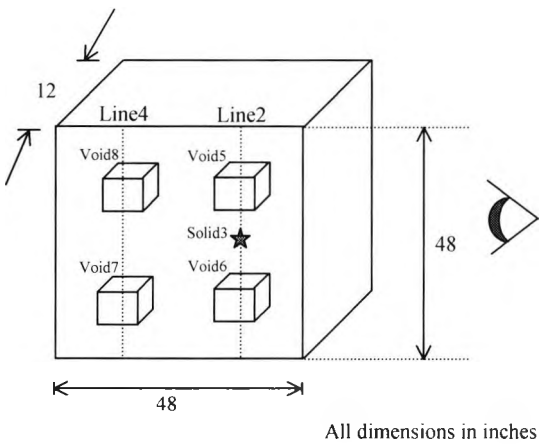


Figure 3.11: Solid 3 (east face)

3.3.2. Wall D (East)

The east side of wall D was tested for a further void, the dimensions are 3in x 6in x 4in, this is shown in figure 3.12.

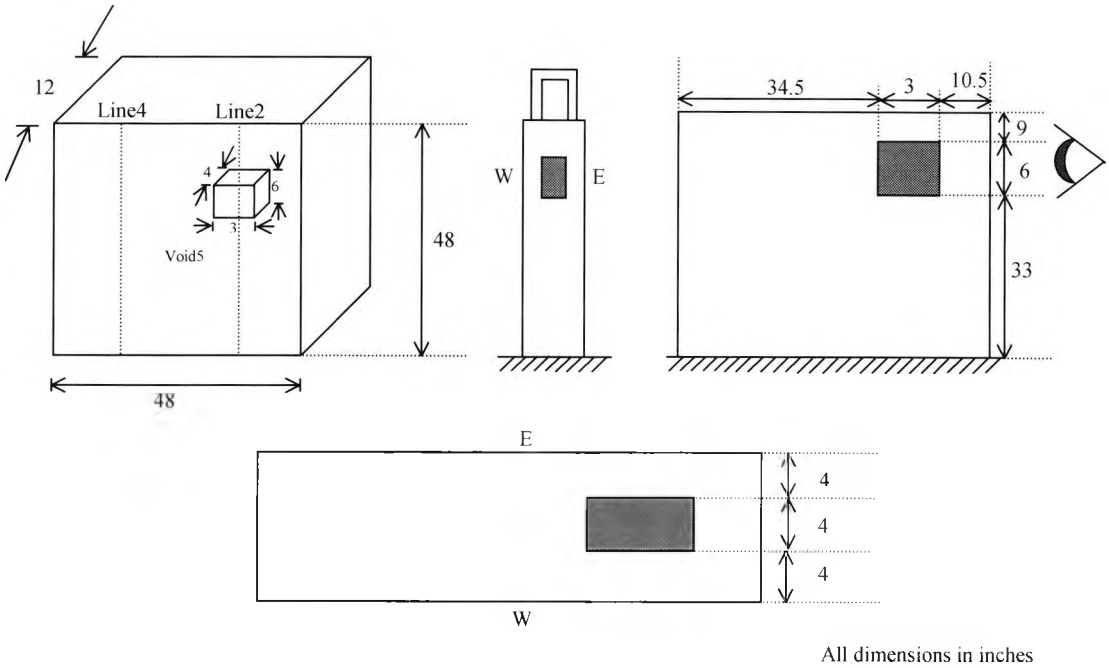


Figure 3.12: Void5

The dimensions for the next three voids are 4in x 4in x 4in, 4in x 4in x 4in and 3in x 6in x 4in, they are illustrated in the figures 3.13, 3.14 and 3.15.

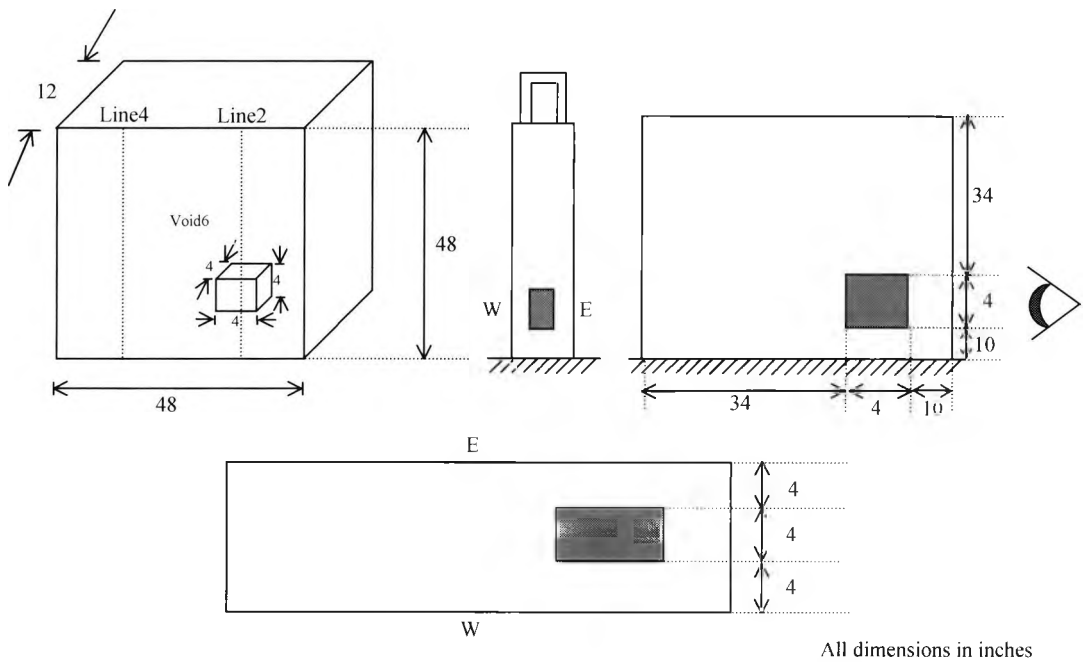


Figure 3.13: Void6

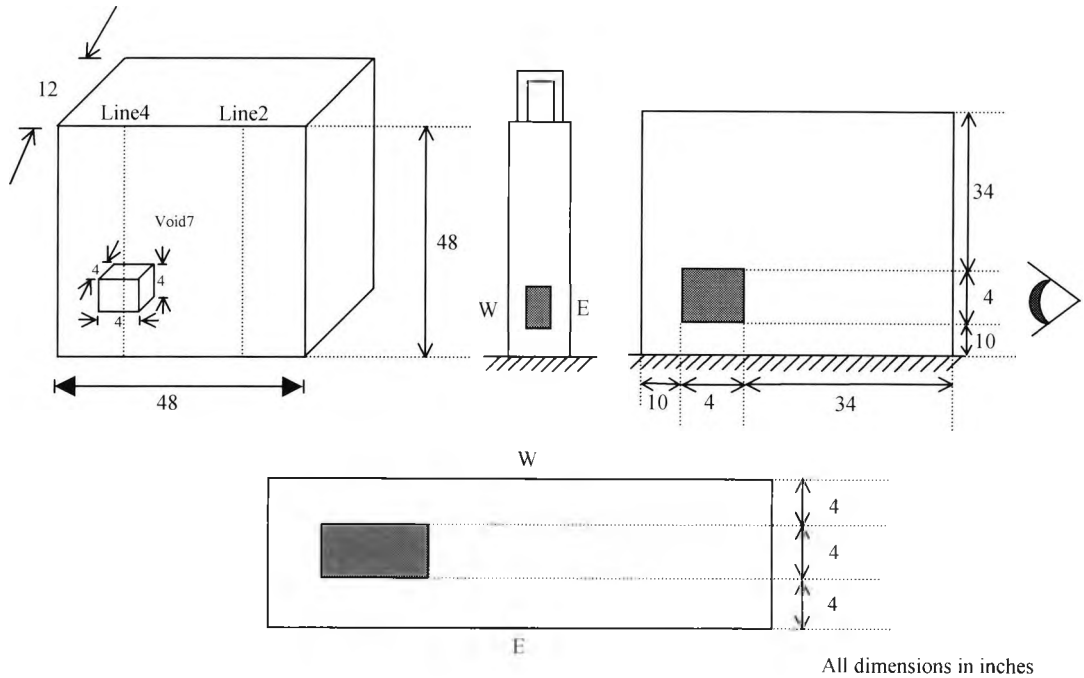


Figure 3.14: Void7(honeycomb)

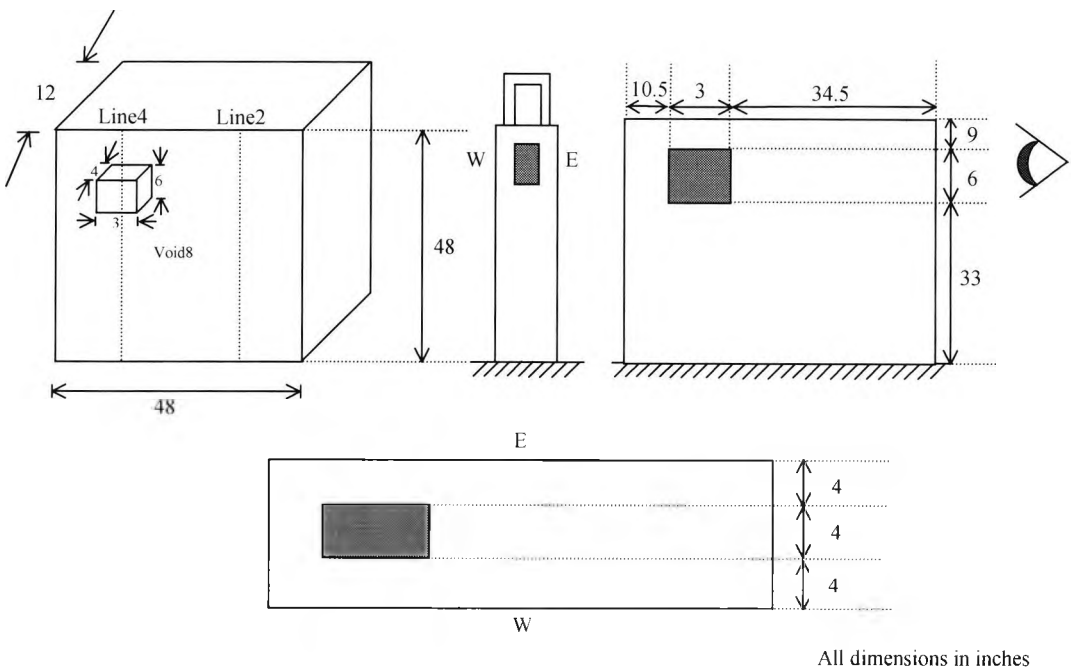


Figure 3.15: Void8 (honeycomb)

3.4. Summary

The impact echo equipment and a series of drawings of the specimen walls have been presented, showing the geometry of the built-in defects. In the next chapter, the results of the impact echo method are discussed.

Chapter four

Impact echo survey

4.1. Introduction

In the last chapter, the use of the impact echo device and the details of the series of test walls were presented. This chapter presents the impact echo results, which are studied analytically to observe the accuracy of the impact echo technique. The method of understanding the amplitude spectrum (FFT graphs) is explained. An example of impact echo data is given in the time and frequency domain to achieve understanding of the data in the amplitude spectrum

4.2. Prediction of the theoretical frequency

In order to interpret the amplitude spectrum graph, the theoretical frequency corresponding to the thickness or depth of defect needs to be predicted. The code names and geometrical drawings of the test cases are illustrated in chapter 3. Using equation 2.2 from chapter 2.

$$f_p = \frac{C_p}{2\,T} \qquad \text{Equation 2.2}$$

Where C_P is the P-wave speed in the test object

f_P is the frequency of P- wave reflections from the interface

T is the depth, the distance to the interface of crack or bottom of test

The depths to the defect are known, which can be deduced from the geometrical drawings, and the velocity in concrete C_p is taken to be 3507.5 m/s, hence, the theoretical frequency f_p can be calculated for all the test points. The f_p values are shown under predicted frequency on table 4.1. In practice, the value of C_p is found by identifying the frequency associated with a known thickness for a sound concrete zone. This can be achieved using a core of the material under laboratory conditions.

4.3. Void defect

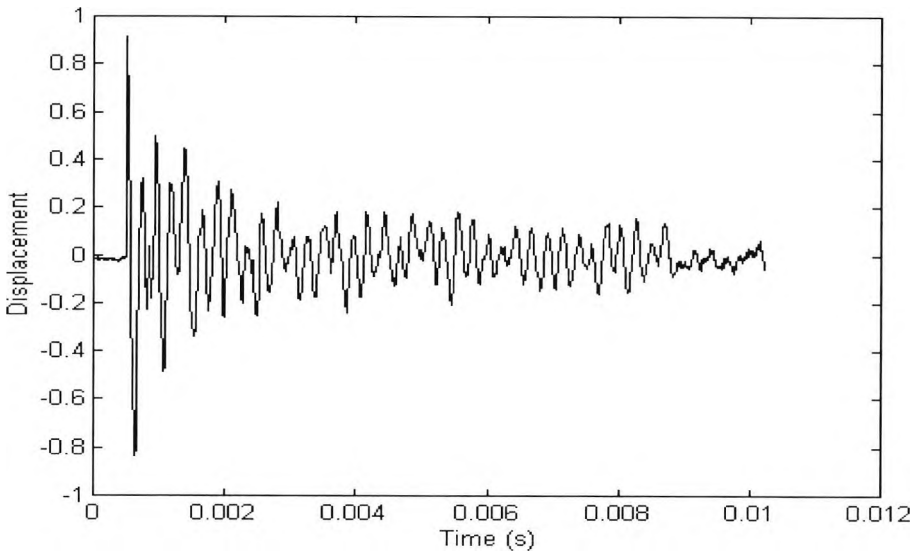


Figure 4.1: Wav1e in time domain

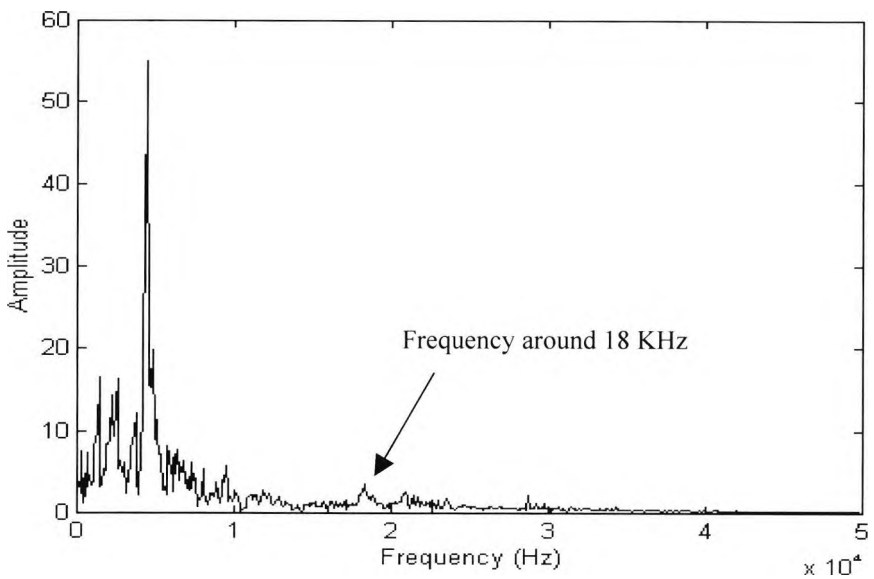


Figure 4.2: Wav1e in frequency domain

Figure 4.1 and 4.2 shows results Wav1e, which are used as an example to illustrate the impact echo test data in time and frequency domain. Although it is possible to understand time domain displacements, shown in figure 4.1, it is somewhat complicated and time consuming, especially when there are thousands of graphs to study. This is explained in chapter 2. The frequency domain shows the peaks in the spectrum which represent the transient resonance caused by multiple reflections of waves between surfaces and internal flaws. In this example, there is a dominant frequency around 18 kHz which corresponds to the void depth. This is predicted to gain an idea of what

frequency to expect, as discussed in section 4.2. In this example, the predicted frequency is deduced as 17.261KHz with a corresponding depth of 0.1016m, hence, the amplitude spectrum is more efficient and easy to comprehend.

4.4. Variation of velocity in concrete

Although the C_P in concrete is taken to be 3705.5 m/s, in reality, the velocity in concrete will never be same from one concrete wall to another or over an individual specimen. There are factors which affect the C_P value, for example, the concrete may be more compacted in some parts than others. An estimated variation of ± 2% was used giving a corresponding range of C_P values of 3900 to 4050 m/s (Yiching & Sansalone, 1992) used to gain an idea of the influence. The f_p for this range has been computed, which is shown on table 4.1. Indeed, the 2% variation of concrete velocity is calculated as:

$$3507.5 \pm 2\% = 3507.5 \pm 70.15 \text{ m/s}$$

Code Name	C_P Theorectical Velocity (m/s)	T Known Depth (m)	f_P Predicted Frequency (KHz)	F_P Results of Impact Echo (KHz)	f_P Predicted frequency (KHz) for Concrete Velocity of 3437.35 m/s	f_P Predicted frequency (KHz) for Concrete Velocity of 3577.65 m/s
Wav1e	3507.5	0.1016	17.261	18.23	16.916	17.607
Wav2e	3507.5	0.1016	17.261	16.68	16.916	17.607
Wav3e	3507.5	0.1016	17.261	8.92	16.916	17.607
Wav4e	3507.5	0.1016	17.261	16.87	16.916	17.607
Wav1w	3507.5	0.1778	9.864	9.99	9.666	10.061
Wav2w	3507.5	0.1016	17.261	15.22	16.916	17.607
Wav3w	3507.5	0.127	13.809	12.60	13.533	14.085
Wav4w	3507.5	0.1524	11.508	12.31	11.277	11.738
Wacl1	3507.5	0.3048	5.754	3.68	5.639	5.869
Wacl2	3507.5	0.3048	5.754	3.68	5.639	5.869
Wdv5e	3507.5	0.1016	17.261	9.21	16.916	17.607
Wdv6e	3507.5	0.1016	17.261	9.21	16.916	17.607
Wdv7e	3507.5	0.1016	17.261	14.45	16.916	17.607
Wdv8e	3507.5	0.1016	17.261	16.29	16.916	17.607
Wdcle	3507.5	0.3048	5.754	5.91	5.639	5.869

Table 4.1: Calculation of the theoretical frequency

Using table 4.1, the predicted frequency is 17.261 kHz for Wav1e, the impact echo results shows a dominant frequency around 18.23 kHz. The depth at this frequency is 0.1016 m. The predicted frequency for Wav1w is 9.864 kHz and impact echo results indicates a dominant frequency about 9.99 K Hz, and the equivalent depth 0.1778 m.

The predicted frequency for Wav3w is 13.809 kHz, table 4.1 shows impact echo results of 12.60 kHz, and the depth for this void is 0.127 m. Wav4w has a predicted frequency of 11.508 kHz, table 4.1, suggests a frequency at about 12.31 kHz, the depth for this void is 0.1524 m. The theoretical frequency for Wdcle is 5.754 kHz, this is sound concrete, and the depth to the end of wall is 0.3048 m. The impact echo results suggest 5.91 kHz.

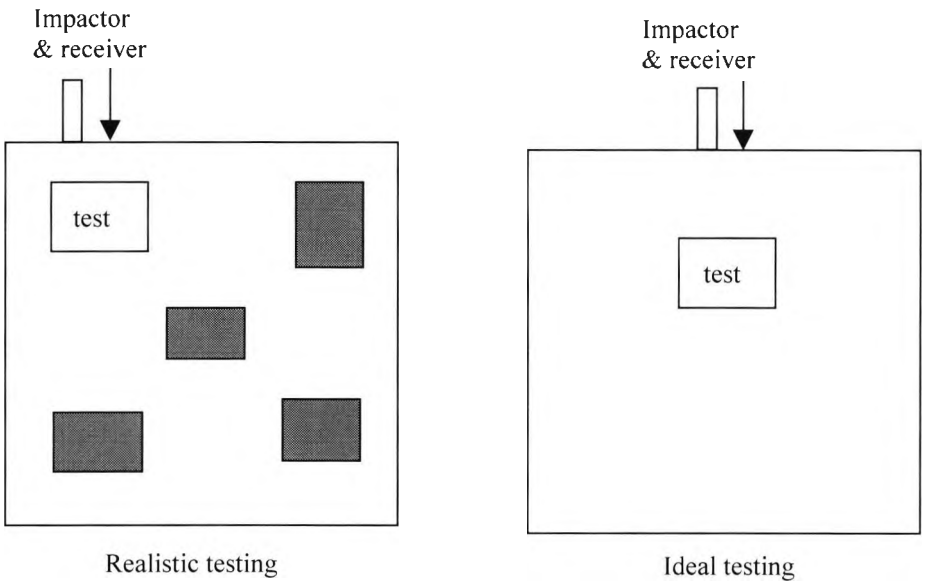


Figure 4.3 : Impact echo testing

So far, these results are fairly accurate and fall within the +/– 2% variation of concrete velocity shown on the last two column of table 4.1, except for the data of Wav1e, Wav4w, and Wdv8e but it is very close to the frequency range. However, the results for Wav3e, Wacl1, Wacl2, Wdv5e to Wdv6e are not so promising. There are several factors that need to be considered. One important factor is that the specimen void or sound concrete was surrounded by boundaries and other built in defects, which can be seen in chapter 3 from the geometrical drawings. Whilst the impact echo response in concrete contains other frequencies including the harmonics of the test object, the response of the voids in the vicinity is added to the signal, where there should be a strong response for the specimen test point. In these circumstances the signal may be weaker than normal. Also, a signal may have to go round bends, which would result in a longer distance producing a lower frequency. This is illustrated in figure 4.3, which shows a realistic and ideal impact echo testing. The ideal testing would give a better signal response than the realistic testing which is embedded with other built in defects.

However, the realistic illustration is true to life, because in a typical wall there would be voids, delaminations and other defects of various dimensions. The ideal case is obviously not practical to achieve.

Another factor to consider is the results for honeycomb defects, which are the Wdv5e to Wdv7e test points. The predicated frequency for Wdv5e is 17.261 kHz shown in table 4.1, impact echo test indicates a frequency 9.21 kHz. This is even lower than the minimum velocity of 3437.35 m/s, which gives a frequency of 16.916 kHz. This is typical for a honeycomb defect, because when the impact generates a signal, the response takes longer to get back at the surface of the test object. It then appears to have a lower frequency. Equation 2.2 shows that if the distance is higher, then the frequency will be lower. Similarly, the results for Wacl1 and Wacl2 are misleading, the frequency of 3.68 kHz, indicating that the wall depth is deeper than the nominal depth of 0.3048 m. If the concrete velocity is smaller, this would suggest that the actual signal would take longer to return which could be true in this case. As illustrated in chapter 3, these sound concrete tests were conducted between test points Wav1w, Wav2w and Wav3w,Wav4w, where the neighboring voids affect the signal.

4.5. Variation of amplitude graphs

Code Name	Standard deviation ($x10^3$)	Mean ($x10^3$)	No. of bad data
Wav1e	1.1716	17.388	1
Wav2e	0.5772449	16.339	1
Wav3e	1.3856	6.8333	0
Wav4e	0.6555874	16.607	1
Wav1w	2.2239	8.2143	1
Wav2w	0.2977728	15.413	1
Wav3w	2.9200	10.677	0
Wav4w	0.8816836	11.523	3
Wacl1	0.7073698	3.7083	0
Wacl2	0.6255578	3.5625	0
Wdv5e	0.8692948	9.5833	0
Wdv6e	0.9309044	9.4167	0
Wdv7e	1.9073	13.281	0
Wdv8e	1.4580	15.740	0
Wdcle	0.2287266	5.625	0

Table 4.2: Error calculation of the impact echo test points

It was found that the records for each test point indicated in chapter 3, were slightly different from each other, so for Wav1e result for example, the 15 amplitude graphs varied, i.e. no graph was exactly the same. In fact, all 225 graphs were studied to observe its repeatability. The divisions on the graphs were calculated as :

$$\text{For one division} = 0.5/16 = 0.031 \text{ cm}$$

The graphs were examined by using a ruler to deduce the frequency of interest, consequently the error for reading with a ruler is estimated as $\pm$ 0.016 cm. It was noted that although the frequencies for a particular defect was similar to the next record, there was a variation on the amplitudes of the important peaks. Also, there were about 8 bad graphs (which showed no dominant peaks in the transient response) out of the 225 graphs this is indicated in table 4.2. For example code name Wav4w in table 4.2 has 3 bad records, so 12 records were used for calculating the standard deviation. Bad data is principally caused by incorrect contact of the device with the specimen surface.

The author conducted FFT processing for all 225 records using Matlab as previously mentioned in chapter 3.

4.6. Summary

The impact echo results have been presented with an example of an impact echo data in raw and FFT forms. The velocity variation of concrete speed was assumed to be $\pm$ 2%, and in most cases, the impact echo results were within this range. The standard deviation for most test cases were less than one, this showed the repeatability of the impact echo device to be about 60% using good data. Also, the maximum standard deviation was about 3, which is still fairly low. In chapter five, neural networks is discussed, which are used to analyse the impact echo results.

Chapter five

Neural networks

5.1. Introduction

This chapter focuses on the application of artificial neural networks to data obtained from real and simulated impact echo testing. The gathering of the former is covered in chapter 7, these involving readings on a group of concrete walls having built-in defects. Simulation of data by finite element modelling is covered in chapter 10. This chapter presents the theory of neural networks. The history of neural network development is also presented. A detailed explanation of significant aspects of network architecture is given and the selection of the back propagation network against other neural networks presented.

Through selection of a suitable network type, its training and verification, it is hoped to achieve advancement in interpretation intelligence for the impact echo method.

5.2. Neural networks

A neural network has been used in this research, because the impact echo method involves large amounts of data that cannot be well described symbolically or mathematically. The rapidly growing number of fielded applications demonstrates the developing superiority of neural networks for many problems. They are significantly better at dealing with more complex or open systems, which may be inadequately perceived and which can not be sufficiently described by a set of rules or equations. Tasks requiring fault tolerance or coping with noise, or involving pattern detection or recognition, diagnosis, abstraction, and generalisation, are ideal for a neural network treatment (Maren, Harston & Pap (1990:220-221)).

Neural networks came into the forefront of computer technology for research and commercial development in the late 1980's. However, this is not a new field, because its history goes beyond that of conventional computing. Its novelty is awareness with justified analysis, and a through understanding of the subject (Aleksander & Morton, 1990).

The range of conventional computer applications is dependent on a human to develop algorithms. This fact distinguishes information processing in conventional computers from human information processing. Namely, humans are capable of developing their behaviour through learning, whereas computers have to wait for some human to feed them the algorithms needed to perform the desired task.

Despite advances in computing, humans are still better at solving non-numeric challenges. The kinds of problems that humans excel at generally involve pattern recognition of either speech or images, or both. Scientists say this is because of the parallel architecture of the brain, which gives humans the ability to solve more easily problems with simultaneous constraints.

The main areas in which neural networks have been employed are (Shubnikov, 1997):

- Signal processing
- Pattern recognition (Glover, 1988)
- Automated control (Tolat & Widrow, 1988)
- Adaptive associative memory for expert systems
- Speech analysis (Waibel, 1988)
- The solution of high dimension mathematical problems

Human experiences are stored in a network of interconnected brain cells (neurons). The computing that these devices perform is different from the algorithmic, rule-base method. Accordingly, the objective for neural network researchers is to identify how it is that these networks are capable of storing and using experience and replicate them in computer models.

5.3. Development of artificial neural networks

The pioneers of neural networks were Warren McCulloch and Walter Pitts (McCulloch & Pitts, 1943). In 1943 they devised a basic model of variable resistors and summing amplifiers that represented the variable synaptic connections (weights) which connects neurons together and the operation of the neuron body (soma), respectively. This became the basis of most neural networks. Then, in 1949, Donald Hebb (Hebb, 1949) put forward a model of dynamic memory. He stated that a group of neurons could reverberate in different patterns, each of these patterns relating to the recall of different experience.

Frank Rosenblatt designed the Perceptron in 1962 (Rosenblatt, 1958). This is a system, which recognised images using the McCulloch and Pitts model in conjunction with some non-learning feature extractors of an image. Bernard Widrow and Martian Hoff (Widrow & Hoff, 1960) devised one of the early perceptron training algorithms. This is called the Widrow-Hoff algorithm. This was followed by Marvin Minsky and Samuel Papert, who declared that the Perceptron, with its single layered network, was unable to solve anything except problems that were separable linearly (Caudill, 1993). Also, they stated another important difficulty, which was an analysis of the time it takes to train such networks, and how this increases with the number of input neurons. Unfortunately, the scaling of training times as the size of the input space increases, is still unsolved (Taylor, 1997). However, a number of the most persistent scientist have continued these studies. More complex and flexible architectures have been developed in the work of Anderson (Anderson, 1972) and Grossberg (Grossberg, 1987). The theoretical foundation of the field was slowly taking shape; Minsky and Papert's estimates were shown to be solvable. Then in the 1980s, the growth of activity in the neural network field had an explosive reaction.

In 1982, John Hopfield (Hopfield, 1982) observed that stable reverberations were a good way of producing advanced computer memories. However, his findings did not solve the problems of hard learning, which is when in a three layer neural network, the middle layer called the hidden layer, seems to cause problems during training. The training information consisting only of bit patterns of inputs, and their corresponding outputs do not contain a specification of what should be fed to the teaching inputs of these hidden units. There is no hope of achieving the relevant connections between inputs and outputs if they are allowed to act at random. Also, the network could get stuck in the wrong reverberations. This was overcome by Hinton and Sejnowski (Hinton & Sejnowski, 1983) in 1983, who created the Boltzman machine, which was named after Ludwig Boltzman, who studied the effects of heat on the agitation of molecules. The similarity between systems of physical particles and neural network enabled these researchers to use the electronic equivalent of heat (electronic noise) to avoid false reverberations and deal with the hard learning phenomena.

In 1986, Rumelhart, Hinton and Williams (Rumelhart, Hinton & Williams, 1986) concluded that a feed forward network, which did not reverberate, needed researching.

This served to take a pattern at the input of the net and feed it forward to the output, where it is translated into another pattern.

The appearance of multi-layer Perceptrons allowed expansion of the class of solvable problems. Three layer Perceptrons made it possible to mould a structure of functional space of many degree of complexity, limited only by the number of neurons on the network. Training of a multi-layer network is, however, extremely laborious. One such technique is the back propagation (Shubnikov, 1997).

The back propagation algorithm solved the problems of training hidden neurons. This was initially shown by Paul Werbos (Werbos, 1974) and independently discovered in 1985 by Parker (Parker, 1985) and LeCun (LeCun, 1985). It was later publicised by Rumelhart, McClelland and et al (Rumelhart & McClelland, 1986) in 1986.

Another influential contributor to the development of neural computing techniques is Teuvo Kohonen (Kohonen, 1982). He used unsupervised learning, where the network discovers hidden patterns in the input data, which even the designer of the network may be unaware. However, these numerous researchers were applying neural network to simplified problems to examine their ideas instead of solving problems on a realistic scale. In 1984 and 1988 Kohonen's study applied neural network to speech recognition, where phonemes were turned into recognised words (Kohonen, 1988).

The recognition of visual patterns is a major application of neural computing (Fukushima, Miyake & Ito, 1975; Fukushima, 1989). Kunihiko Fukushima in 1988 devised and partially built a machine called the Neocognitron. It is able to identify sections of an image from features, for example, corners and straight lines.

Gorman and Sejnowski have applied the neural network method to the classification of sonar returns from two undersea targets, a metal cylinder and a similarly shaped rock in 1988 (Gorman & Sejnowski, 1988). In 1989, third order neural networks were employed to distinguish between classes of geometric patterns regardless of their translational position, scale, and angular orientation. Researchers claimed that a significant feature of this network is that it is trained on only one view of each pattern, using a simple single-layer perceptron learning rule (Reid, Spirkovska & Ochoa, 1989). Apparently, when a higher-order neural network (HONN) was first created in the 1960s

for performing non-linear discrimination, it was rejected as impractical because of the combinatoric explosion of higher-order terms (Reid, Spirkovska & Ochoa, 1989).

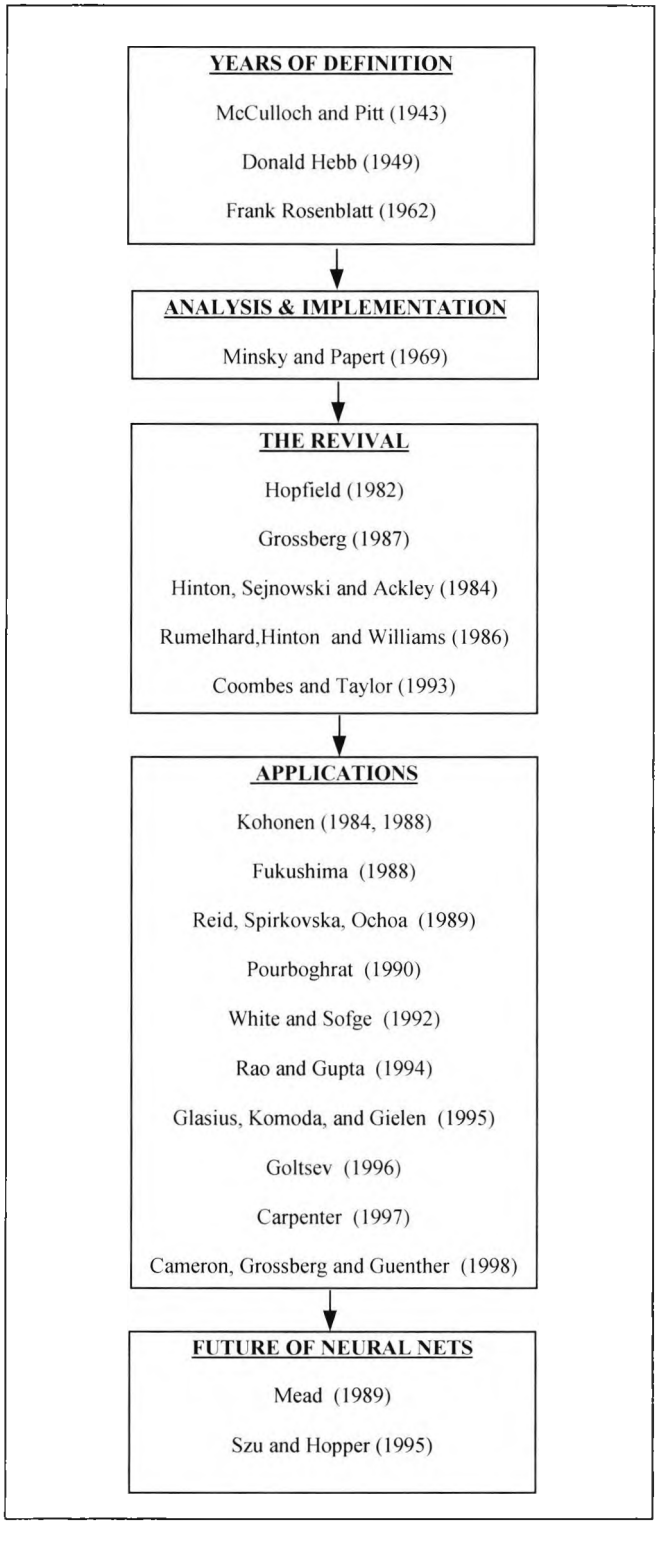


Figure 5.1: Development of Neural Networks

Another important contributor to this field is Carver Mead and his colleagues, who in 1989 designed analogue silicon chips for the processing of sensory information (Mead, 1972). Yoh-Han Pao devised the functional link network in 1989. This became the most

powerful function-mapping model known (Masters, 1993). The contributions of these major pioneers are summarised in figure 5.1.

Since 1989, there have many other researchers who have designed new neural networks and applied them to useful applications. In 1972 Meisel created a statistical algorithm, which Donald Specht revived in 1990, developing the probabilistic neural network (Masters, 1993). This uses a simple hardware processors, which is cheap and fast, allowing economical implementation.

In 1990, a neural network was used to find the optimum collision free path for mobile robots (Pourboghrat, 1990). In 1991, numerous developments occurred in the self-organising network approach (Ritter, Martinetz & Schulten, 1991).

An attempt to understand the nervous system of many species of animals has been investigated. Simplified versions of mechanisms are shown to be of great value in commercial applications, for example, in 1992, models of the eye or ear, and also areas of control, where reinforcement training has led to some very effective control systems (White & Sofge, 1992).

The introduction of an energy function, led to the spin glass approach, with the global ideas on phase transition and temperature entering the field of neural networks for the first time. A spin glass derivation was also given by Amit (Amit, 1989) of the capacity limit of 0.14N as the limit to the number of patterns which can usefully be stored in a network of N neurons.

In 1993, it became clear as to what was the source of the limit on the storage capacity of these networks, and how this might be increased by choosing suitable connectivity to achieve the full capacity N (Coombes & Taylor, 1993), where N is the number of neurons. Also, in that year, researchers used neural networks as part of a hybrid pattern classification system, which can recognise patterns even when they are deformed by transformation of rotation, scaling, and translation or a combination of these (Yuceer & Oflazer, 1993). Similarly, hierarchical artificial neural networks were designed to enhance edge measurement (Lu & Szeto, 1993).

In 1994, researchers used a neural structure called the dynamic neural processor (DNP), to control non-linear multivariable systems (Rao & Gupta, 1994). In 1995, a Hopfied

design neural network has been used and shown to be very effective for path planning and obstacle avoidance. These researchers claim that this deterministic system can rapidly provide a proper path, from any arbitrary start position to any target position, avoiding both static and moving obstacles of arbitrary shape (Glasius, Komoda, & Gielen, 1995).

In 1996, neural network architecture was applied to the problem of texture segmentation in natural scenes (Goltsev, 1996), while in 1997, an efficient back propagation network learning, using dynamically optimal learning rate and momentum factor, was considered (Yu & Chen, 1997). Researchers proposed an unsupervised competitive learning algorithm called Centroid Neural Network (CNN), which estimates optimal centroids of the related cluster groups to each training data (Park, 1997). Also, during that time, an established neural network model, adaptive resonance theory (ART) was introduced as a class of ART models for learning, recognition, and prediction with arbitrarily distributed code representation (Carpenter, 1997).

In 1997 most efforts by neural network researchers were targeting on designing a classifier with a large discrimination power and good generalisation ability which was stated by the IEEE (1997). Indeed in that year, some of the avenues being explored for neural networks were (Taylor, 1997):

- Developments in theoretical understanding - Convergence of training schedules and their speed-up is under investigation.
- Statistical mechanics is leading to significant new insights in problems of storage and response of neural networks.
- In the field of computational neuroscience, attempts have been made to build simple models of the neural systems, which are vital in controlling the response patterns of animals of a given species.

It has been stated that one of the drawbacks of the back propagation algorithm method is that the training patterns must be repeated hundreds of times, and the process of training can take several hours. There are two dangers in training:

- The possibility in which the network ceases to be trained and the process wanders randomly, this is called paralysis.
- The possibility of falling into a local minimum of the energy relief.

Consequently, many heuristics are employed to eliminate paralysis. Because the method of gradient descent is used in training, the error surface has a complex shape; a network can fall into a local minimum and can be incorrectly trained. Statistical methods can be mixed with gradient descent to correct the problem. These are questions of current research (Shubnikov, 1997). Also, an improvement to the back propagation algorithm has been suggested (Liu & Meng, 1997), which is that the gain of the sigmoid should not be fixed to 1 but is made variable so that it reaches convergence faster.

In 1998, neural network researchers used artificial neural networks on the recognition of human head orientation (Rae & Ritter, 1998). This is a method based on three neural networks, which enable a computer to identify the head orientation of a user by learning from examples. One network is used for colour segmentation, a second for localisation of the face and the third for the final recognition of the head orientation.

A new system for real-time detection and classification of arbitrarily scattered surface-land mines from multi-spectral imagery data of minefield has been introduced. This involves a variety of neural network structures for feature extraction, detection, and classification of targets in six different optical bands ranging from near UV to near IR (Miao, Azimi-Sadjadi & et al, 1998).

A self-organising neural network architecture that transforms optic flow and eye position information into representations of heading, scene depth, and moving object locations has been shown to be effective (Cameron, Grossberg & Guenther, 1998). These representations are used to navigate reactively in simulations involving obstacle avoidance and pursuit of a moving target.

It can be concluded that the neural network field is founded on the efforts of a number of researchers who have continued to work on designing, modifying, and implementing different aspects of the artificial neural network. Consequently, due to their efforts, this subject has breadth and depth. It is now better understood by more people, to the extent that there are simplified user friendly software widely available for most applications, where no other solution can be found, or neural network is used part of a system enhancing to deliver an improved performance. Indeed, it is becoming more convenient that artificial neural networks can be used either on their own or in hybrid systems wherever and however is most appropriate.

5.3.1. Summary of neural networks

Table 5.1 below shows a summary of the historical development of the main models of artificial neural networks, each of which reflects definite progress in approaching the principals of this field.

Name of network	Inventors and developers	Years Introduced	Primary Application	Limitations	Comments
Perceptron	Frank Rosenblatt	1957	Typed character recognition	Cannot recognise complex characters e.g. Chinese; sensitive to difference in scale, translation, distortion	The oldest neural network known; was built in hardware rarely used today
Madaline	Bernard Widrow	1960 – 62	Adaptive nulling of radar jammers; adaptive modems; adaptive equalisers(echo cancellers)in telephone lines	Assumes a linear relationship between input and output	Acronym stands for multiple adaptive linear elements; powerful learning law; In commercial use for than 20 years
Avalanche	Stephen Grossberg	1967	Continuous speech recognition; teaching motor commands to robotics arms	Literal playback of motor sequences, no simple way to alter speed or interpolate movements	Class of networks, no single network can do all these tasks
Brain state in a box	James Anderson	1977	Extraction of knowledge from data base	One-shot decision making, no iterative reasoning	Similar to bi-directional associative memory in completing fragmented inputs
Self-organising map	Teuvo Kohonen	1980	Maps one geometrical region e.g. rectangle grid, onto another e.g. aircraft	Requires extensive training	More effective than many algorithm techniques for numeric aerodynamic flow calculation
Cerebellatron	David Mar, James Albus, Andres Pellionez	1969 – 82	Controlling motor action of robotics arms	Requires complicated control input	Similar to avalanche network; can blend several command sequences with different weights to interpolate motions smoothly as required
Hopfield	John Hopfield	1982	Retrieval of complete data or images from fragments	Does not learn, weights must be set in advance	Can be implemented on a large scale
Necognitron	Kunihiko Fukushima	1978 – 84	Hand printed character recognition	Needs unusually large No. of processing elements and connections	Very complicated network; insensitive to differences in scale, translation, rotation; able to identify complex characters e.g. Chinese
Back Propagation	Paul Werbos, David Parker,David Rumel-hart	1974 – 85	Speech synthesis from text; adaptive control of robotics arms; scoring of bank loan application	Supervised training only, correct input/output must be plentiful	The most widely used network, simple to learn

Name of network	Inventors and developers	Years introduced	Primary application	Limitations	Comments
Bi-directional associative memory	Bart Kosko	1985	Content addressable associative memory	Low storage density; data must be properly coded	Easiest network to learn, good educational tool; associates fragmented pairs of objects with complete pairs
Adaptive Resonance Theory	Gail Carpentor & Stephen Grossberg	1978 – 86	Pattern recognition, particularly when pattern is complicated or unfamiliar to humans (radar or sonar readouts, voiceprints)	Sensitive to translation, distortion, changes in scale	Very sophisticated but not applied to many problems
Boltzmann & Cauchy Machines	Jeffrey Hinton, Terry Sejnowsky, Harold Szu	1985 – 6	Pattern recognition for images, sonar, radar	Boltzman machine:long training time. Cauchy machine:generating noise in proper statistic distribution	Simple networks in which noise function is used to find a global minimum
Counter-propagation	Robert Heicht-Nielson	1986	Image compression; statistical analysis;loan scoring	Large No. of processing elements & connections required for high accuracy for any size problem	Functions as a self programming look up table;similar to back propagation only simpler, but less powerful
Functional Link Networks	Yoh-Han Pao	1989	A variable hybrid, Tolerant of chaotic noise contaminating the measured data.	Training time can slow if a large no. of functions is used blindly and if hidden layer is included	Training time can be faster if a good choice of functions allows it to avoid hidden layers.
Propabilistic	Donald Specht	1990	Intrinsically a classifier	Needs a thoroughly representative training set, whole training set must be stored, as well as processed, each time an unknown case is classified, memory needs are large, and execution speed is poor	Requires little or no training time, confidence estimates for network decision is good.

Table 5.1: Famous neural networks

5.4. Neural network architecture

The neurons in a network are connected to each other positively or negatively. The connection weights send different signals to neurons; allowing different and useful patterns of output in response to input stimuli (Maren, Harston & Pap (1990:48-51)).

A typical operation of a neural network is to receive an input pattern, operate on it, and produce an output pattern. Each layer of neurons in a network accesses and processes data simultaneously. Each operation by a layer of neurons is considered as a cycle of operation by the network. For example, a network with three layers would have three cycles of operation, accessing the input in the first layer, processing and passing the input to the second layer and processing and passing this pattern to the third layer. The result pattern in the third layer would be the output of the neural network.

The two primary elements that make up a neural network are:

- Processing elements.
- Interconnections

Processing elements (neurons) are simple devices that receive a number of input signals and, based on those inputs, either produce a single output signal (fire) or not. The output signal of an individual processing element is sent to many other processing elements (and possibly back to itself) as input signals via the interconnections between processing elements. In neural networks, learning is fundamental. Almost every neural network system has some kind of learning capability built in. It has three fundamental descriptors, which define the neural network itself. These include:

- The interconnection architecture between the processing elements.
- The rules that determine whether or not a processing element will fire (the transfer function).
- The rules governing changes in the relative importance of individual interconnections to a processing element's input (training laws).

5.5. The basic neuron

Models of the brains and nerve cells (Dayhoff, 1990) motivated the inspiration for neural network architectures. There is limited knowledge of the brain, but the basic anatomy of an individual nerve cell (neuron) is known.

In figure 5.2, a typical nerve cell of the human brain is illustrated. The output area of the neuron is a long, branching fibre called the axon. An impulse can be triggered by the cell, and sent along the axon branches to the ends of the fibre. The input area of the nerve cell is a set of branching fibre called dendrites. The connecting point between an axon and a dendrite is the synapse. When a series of impulses are received at the dendritic areas of a neuron, the result is usually an increased probability that the target neuron will fire an impulse down its axon.

One important and fundamental aspect of the function of a neuron is that the effect of the synapses is variable. It is this variability that gives the neuron its adaptability (Aleksander & Morton (1990:22-23)). McCulloch and Pitt first designed the basic model. This becomes the basis of neural network calculations in contemporary neural computing.

A McCulloch and Pitt neuron, shown in figure 5.3, is a device that has a number of input signals and a single output signal. The input signals are associated with a relative weight w_i . The result of the input to the neuron is the weighted total input:

Total input = Weight on line 1 x input on 1 +
Weight on line 2 x input on 2 +
Weight on line n x input on n

$$= w_1 x_1 + w_2 x_2 + \ldots . w_n x_n$$

$$= \sum_{i=1}^{n} w_i x_i \text{ for all n input signals} \qquad \text{Equation 5.1}$$

This sum is compared to a threshold value, and if the:
Sum is greater than the threshold, then the value output = 1
Sum is less than the threshold then the value output = 0

Or

The threshold value is subtracted from the weighted sum, and the resulting value compared to zero.
If the result is positive then the output = 1
If the result is negative then the output = 0

So

$$y = f_h \left[\sum_{i=1}^{n} w_i x_i - \theta \right]$$ Equation 5.2

If $f_h(x) = 1 \quad x > 0$

If $f_h(x) = 0 \quad x \leq 0$

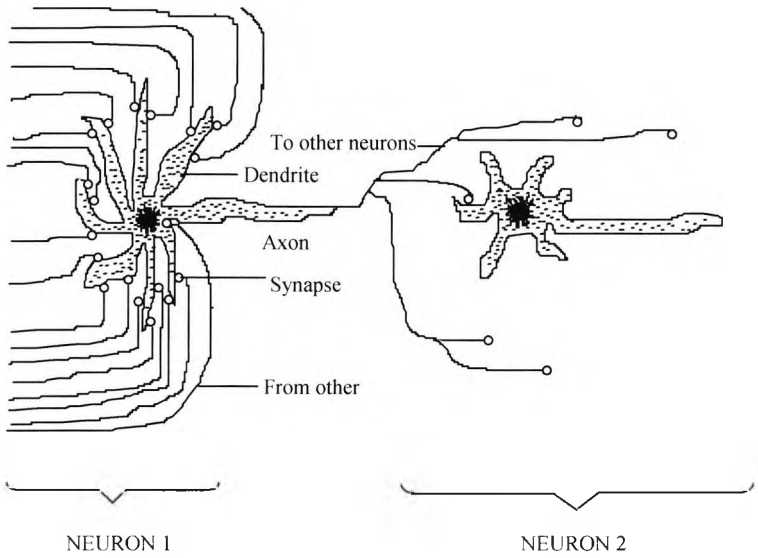


Figure 5.2: The basic feature of a biological neuron (Alexsander & Morton, 1990)

Alternatively, the threshold may be removed out of the body of the neuron and connected to an extra input value that is fixed to be on all the time. So the extra input +1 is multiplied by a weight equal to minus the threshold value, - θ and added like all other inputs. This biases the neuron (Beale & Jackson, 1992) in:

$$y = f_h \left[\sum_{i=0}^{n} w_i x_i \right]$$ Equation 5.3

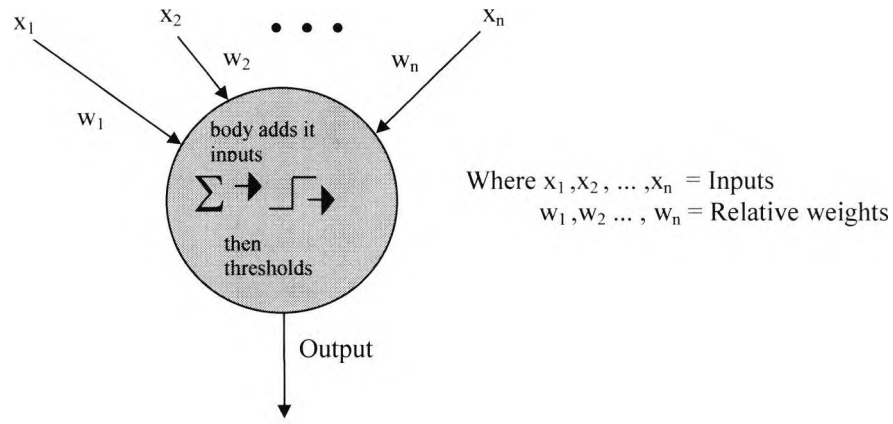


Figure 5.3: A neuron

5.6. Training

A neural network is trained in any of three ways:

- Supervised
- Reinforcement
- Unsupervised

5.6.1. Supervised

A supervised training is the most widely applied, where many samples are collected to serve as exemplars. Each sample in this training set completely specifies all inputs, along with the outputs that are desired when those inputs are presented.

5.6.1.1. Epoch

When a subset of the training set is selected, the samples are presented in that subset to the network sequentially. For each sample, the outputs obtained by the network are compared with the target output. When the whole subset of training samples has been processed, the weights are updated which link the neurons in the network. This updating is carried out in order to minimise a portion of the error in the network's result.

One pass through the subset of training sample with the updating of the network's weights is called an epoch. The number of samples in the subset is called the epoch size.

5.6.2. Reinforcement

This training method uses a reward given to the network by the environment on its response to a given input. This reward may be used to work out modifications to the weights to achieve a maximum reward from the environment.

5.6.3. Unsupervised

An unsupervised training is another principal training method. For this type of training, the network is not provided with outputs for those samples. The assumption is made that each input arises from one of the several classes, and the network's output is an identification of the class to which its input belongs.

The process of training the network consists of allowing it to determine salient features of the training set. These features are grouped into classes, which the network finds distinct.

5.7. Associative memory

A vital characteristic of neural computing is the way information is stored. Memory in neural computing is distributed. The connection weights are the memory units of a neural network. The values of the weights show the current state of knowledge of the network. For example, if an item of knowledge is representing by an input and desired output pair, the knowledge is distributed across many of the memory units in the network. In addition, it shares these memory units with other items of knowledge stored in the network.

Associative memory associates response to a particular input, so that an input is presented and required output is received (Dayhoff, 1990). Figure 5.4 illustrates the different type of associations, which are:

- Auto-associative: this associates a pattern by making the input and response patterns the same, so presentation of an incomplete pattern on the input will result in the recall of the complete pattern. The use of an auto-associative system for pattern completion is indicated in the bottom of figure 5.4.
- Hetero-associative: this associates the input pattern with a different output pattern.

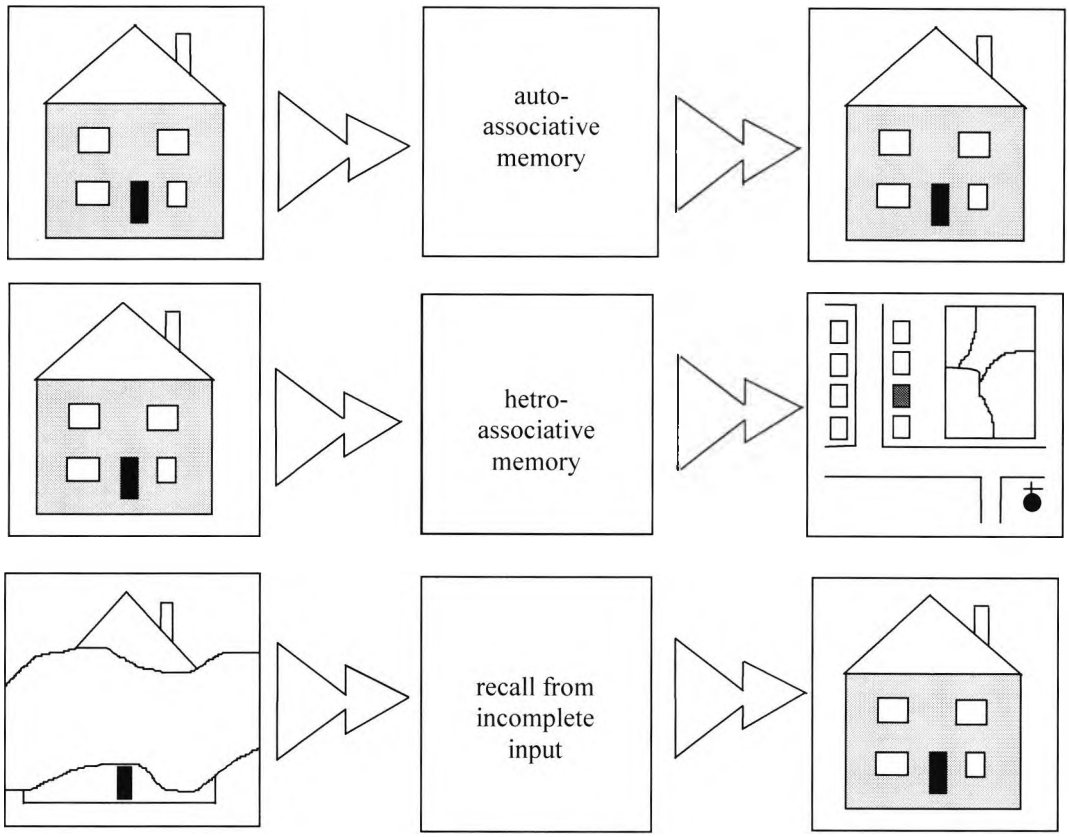


Figure 5.4: The different types of association: auto-association and hetro-association(Beale & Jackson, 1992)

5.8. Hierarchical structure of the neural network

The primary method used to complete different tasks using neural networks is to develop the appropriate structures (Maren, Harston, & Pap(1990: 48-51)). Hence, in order to understand neural network structures, they need to be studied at different levels:

- Level one - This is the lowest level of network structure, which is the neural element itself.
- Level two - Neural network architecture

5.8.1. Transfer function - level one

A typical artificial neuron is a record in an array. The record stores the process of the neuron. This process is ranged between 0 and 1 or -1 and 1. Some neurons have only two possible states, which are the extreme or bounded values. Other neurons activation vary between the bounded limits.

At the first level, neurons are characterised by their transfer function, which operates on the oncoming signals to produce a single output for the neuron. This output is weighted

by the connection strengths to each of the individual connecting neurons. These neurons may each receive a different value, which may be positive, or negative, depending on the sign of the connection weight. Figure 5.5 shows the different types of transfer functions.

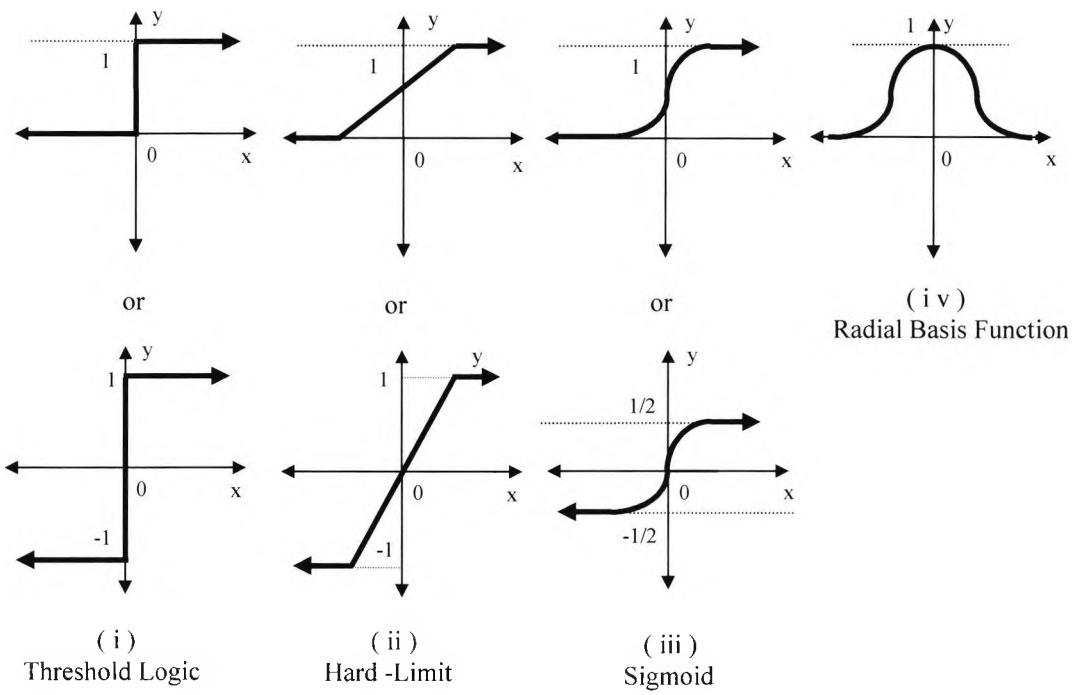


Figure 5.5: Neural network transfer functions (Maren, Harston, & Pap, 1990)

5.8.1.1. Threshold logic nodes

Threshold logic nodes have binary state neurons (0 or 1). If the summed input is greater than or equal to the neuron's threshold, the activation is 1; if less, the activation is 0.

5.8.1.2. Hard limit nodes

This transfer function has an upper and a lower bound, which is set on the summed input from other neurons, plus the threshold, called the total summed input. If this total summed input is less than the lower bound, the activation is defined to be 0 or -1. If the summed input is greater than or equal to the upper bound, the activation is 1. If the summed input is somewhere between the upper and lower bounds, then the activation is defined as a linear function of the summed input (Maren, Harston, & Pap(1990: 53-57)).

5.8.1.3. Sigmoid nodes

The sigmoidal shape of this transfer function means that for most values of the total input, the value given by the transfer function is close to one of the asymptotic values. The higher value asymptote is 1, and the lower one is 0 or -1. This allows the output value of the transfer function to be classified into one of two classes: high or low.

The sigmoid function is the most widely used, and any other function that is smoothly defined over the interval from minus to positive infinity will work just as well, for example the arctangent function is also used.

5.8.1.4. Radial basis function

This function is useful when creating a neural network for continuos function mapping (Maren, Harston, & Pap (1990: 220-221)). The centres and the widths of these functions may be adapted. This makes it a more adaptive function than the sigmoid function. For example, mappings may require two hidden layers of sigmoid function units, whilst a single layer of neurons using radial basis functions can accomplish this.

5.8.2. Network structure-level two

This level concerns the physical organisation and arrangement of the neurons of a neural network. The neurons in a neural network may be organised in a number of different ways. A common neural network may have different layers of neurons, some which accept the input, others which process it and another layer to store the output. Some neural networks combine all these functions into one or two layers. Some networks can pass information forward from one layer to another, but not backward. Others allow backward connections. Some networks allow information to move vertically between layers, but not between neurons in the same layer. Other networks will have one layer with lateral connections between the neurons in that single layer. Some neural networks allow a neuron to send information back to itself. The architectural structure of the network affects its performance and also the applications for which it is stated (Maren, Harston, & Pap, 1990). The different types of neural network architectures are illustrated in figure 5.6 and discussed in the following sections.

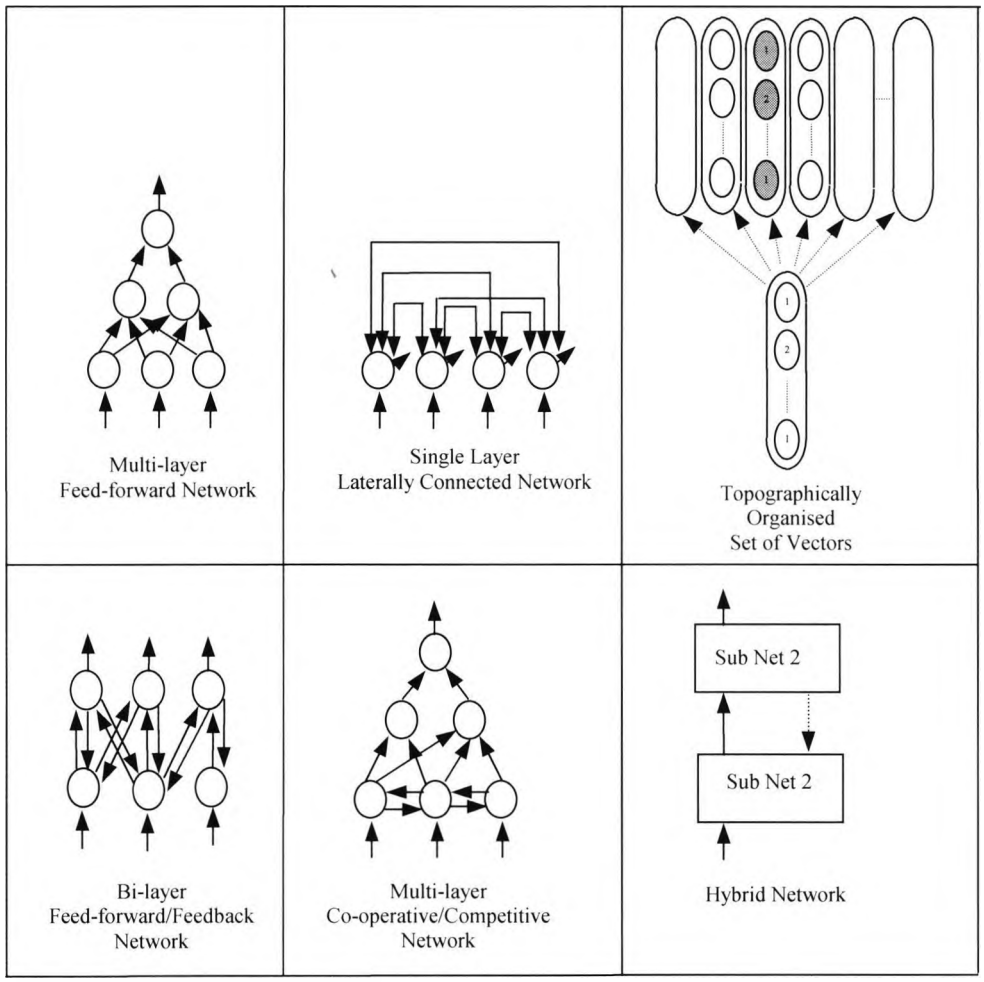


Figure 5.6: The six major types of neural network structures (Maren, Harston, & Pap, 1990)

5.8.2.1. Multi-layered feed-forward neural networks

In this network, all neural signals propagate in a forward direction through the network layers. For example, the back propagation is of this type. The feed-forward networks are suited for pattern classification applications. Some feed-forward networks can learn to generalise about distinguishing characteristics of their input patterns. Their ability to do this depends on the learning algorithm and specific architecture of the network. Feed-forward networks, or networks with more than two layers, are the optimal choice when generalisation as well as pattern recognition is desired (Maren, Harston, & Pap, 1990).

5.8.2.2. Single layer, laterally connected networks

This is the second most widely used type of network architecture. It employs a single layer of laterally connected neurons. It can activate only one pattern at a time due to having one layer. The lateral connections cause different patterns to appear in the single

layer, with each cycle of operation. Laterally connected networks are used for pattern autoassociation. Autoassociative networks can store many patterns. They are good for regenerating clean versions of patterns they have learned when they are given a noisy or incomplete pattern as a starting point. The Hopfield network and Anderson's Brain - State - in - a - Box are examples of single layer, laterally connected autoassociative networks (Anderson, 1972).

5.8.2.3. Single layer, topographically ordered networks

This type of network does not have explicit connections. During learning, a measure of the vector distance between the different vector neurons is used to adjust their relative positions in vector space. An example of this network is the Learning vector Quantization (LVQ) and developed by Teuvo Kohonen (Kohonen, 1982).

5.8.2.4. Bilayered feed-forward and feed-back networks

This two-layer network passes information backward as well as forward during processing. This is done by having feedback connections and the feed-forward connections between the neural elements. This means that these networks have two sets of connection weights, one going from the first layer to the second, and the other connecting the second layer back to the first. This type of network structure is particularly good for associating a pattern in the first layer with another pattern in the second layer; this is called pattern heteroassociation. This network type can also be used for pattern classification.

The most recent networks of this type involve dynamic resonance. The patterns in the first and second layers repeatedly stimulate each other until the pattern in each layer settles to a stable state. These dynamics provide a way to get improved recall of patterns stored in each layer. An example of a feed-forward and feedback network is the Adaptive Resonance Theory and created by Carpenter and Grossberg (Caudill, (1993: 73-80)).

5.8.2.5. Multi-layer co-operative and competitive networks

The lateral connections in this network architecture are specifically designed so that the co-operative (excitatory or positive) connections and the competitive (inhibitory or negative connections) balance each other in a certain way. The co-operative processes used in these networks are designed to explicitly mimic certain biological networks, which have inspired their design. Grossberg et al has developed a Boundary Contour

System (Maren, Harston, & Pap(1990: 53-57)). This is a layered set of nets, which finds the boundary line segmentation in an image.

5.8.2.6. Hybrid networks

A hybrid network is a creation of two or more networks into a single new structure. This method makes it feasible to build systems of interacting networks, which can complete more challenging problems. There are an endless number of ways in which neural networks can be combined with each other and even combined with traditional statistical methods and expert systems.

The reason for designing a strongly coupled hybrid network is to build a network type, which highlights the strengths and minimises the weaknesses of the combined network systems

5.9. Key neural networks

In the following sections, key neural networks are reviewed:

5.9.1. Kohonen network

A Kohonen self-organising network consists of a single layer of nodes as shown in figure 5.7. These neurons are highly interconnected within the Kohonen layer as well as externally. The Kohonen layer neurons receive the entire input pattern into the network and numerous inputs from the other neurons within the layer.

It is important to know that there are two steps, which are vital to the success of this network (Caudill, (1990:25-30)).

- Weight vectors must be initialised
- It must use weight vectors and input vectors that have been normalised to a constant fixed length (usually unity).

If an input pattern is presented to the network, each neuron in the Kohonen layer receives the input pattern and computes the dot product of its weight vector and input vector. It is the neuron with the largest dot product, which is the declared winner in this competition. This is called competitive learning, and it is this neuron which is the only neuron that generates an output signal, others generate 0. In addition, this neuron and that physically closest (neighbour) are the only neurons allowed learning in this pattern

presentation. So, only the winner is allowed to output, only the winner and its neighbours are permitted to adjust their weights.

5.9.1.1. Competition in learning

When each neuron in the layer outputs its activation, which acts as an inhibitory input to each of its fellows, these tend to reduce the activation of the other neurons, and so reduce their corresponding inhibitory inputs to themselves.

In most of these connections, the effect is mainly inhibitory, but in the neuron's immediate neighbours, it is excitatory. This leads every neuron to attempt to fire itself and if not possible help its neighbours to fire by exciting them. If any of the neuron's neighbour wins, it will be allowed to change its weights, even though it cannot output a signal. As the competing outputs circulate within the layer, the neurons settle down so some small portion of the network that has the largest overall excitation produces an output, the remaining part of the layer stays dormant. When the winning neuron is selected, its weights have to be adjusted using the Kohonen learning rule (Caudill, (1990:25-30)):

$$W_{new} = W_{old} + \alpha (X - W_{old})$$

Equation 5.4

where W is the weight vector

X is the input pattern vector

α is the learning constant

Kohonen learning takes the difference between the input vector and the weight vector and adds some fraction of this difference to the original weight vector. The winner gets to adjust its weights and so does its neighbourhood. They change their weights using the same learning rule and apply it to their own weight vectors. The overall effect of this process is that the winning neuron outputs a +1 signal and the winning neuron and its neighbours move their weight vectors slightly closer to the input vector.

During training, the neighbourhood of the winning neuron is slowly a decreasing circle. This means that at the beginning of the training, the neighbourhood of the winner may be large so substantial portions of the network can learn on each pattern. When training starts, the size of the neighbourhood is gradually decreased, so less neurons learn on each iteration. Ultimately, the winner itself will change its weights

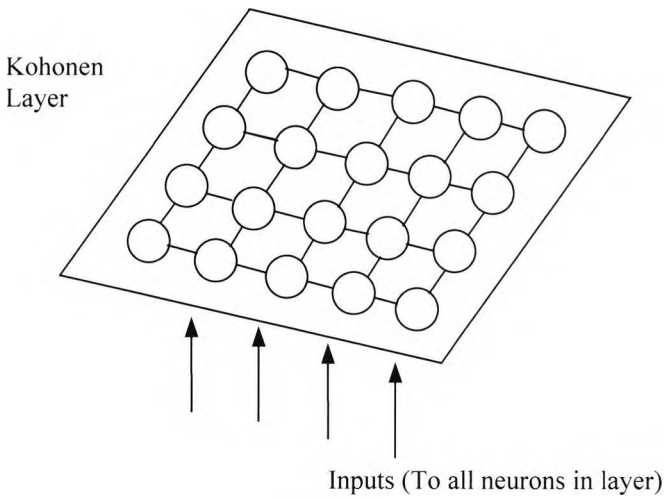


Figure 5.7: A typical Kohonen layer (Caudill, 1990)

The network is ready for the next input pattern. Again, it is presented to every neuron in the layer, and the neurons start to compete again. A winning neuron is selected, it produces an output signal, and this neuron and its neighbour move their weight vectors slightly closer to this input vector. This process is then repeated many times until training is complete.

5.9.2. Hopfield network

The output of each processing element is coupled back to the inputs of every other processing element, except itself. The weights are symmetrical, so the connection weight w_{ij} between processing elements j to i is the same as the connection weight w_{ji} between processing elements i to j.

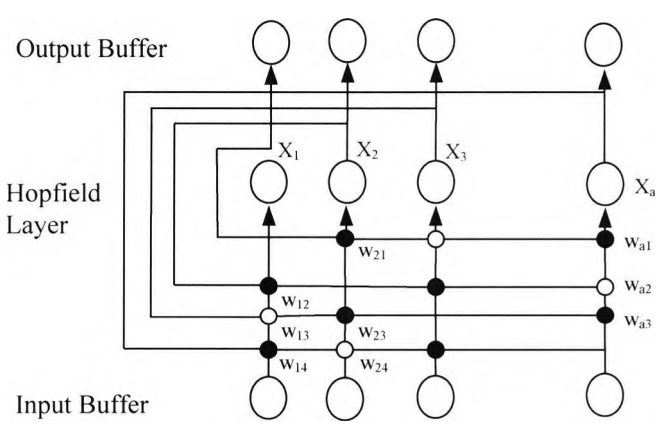


Figure 5.8: Hopfield network architecture (Neural Computing, 1991)

The output of each processing element is connected by weights back to the inputs. Positive weights represent excitatory connections between processing elements, negative weights represent inhibitory connections.

The basic learning rule is:

$$\Delta w_{ij} = (2x_i - 1) (2x_j - 1) \qquad \text{Equation 5.5}$$

Also, symmetry dictates that:

$$\Delta w_{ji} = \Delta w_{ij} \qquad \text{Equation 5.6}$$

Where x_i and x_j have values 0 or 1,

x_i is the output of the current processing element,

x_j is the input to the processing element and

w_{ij} is The connection between the jth processing element and the ith one.

Connections are strengthened when the output of a processing element is the same as the input. Connections are weakened when the input differs from the output of the processing element. The structure of this network is shown in figure 5.8.

5.9.2.1. Energy functions

John Hopfield created a formulation of the dynamics of the network in terms of Ising rings (Neural Computing, 1991). This is the result of seeing the state of the network as an energy surface. This system is:

$$E = - 0.5 \sum_{j \neq i} \sum w_{ij} x_i x_j \qquad \text{Equation 5.7}$$

Where E is the energy of the system

The portion of E affected by the state x_i of a particular processing element is given by:

$$E_i = - 0.5 \sum_{j} (w_{ij} x_i x_j) \qquad \text{Equation 5.8}$$

Where E_i is the energy system affected by x_i

If $x_i = 1$,

$$E_i = - 0.5 \sum_{j \neq i} w_{ij} x_j \qquad \text{Equation 5.9}$$

If $x_i = 0$,then $E_i = 0$, so the change in energy due to one of the processing elements changing states is:

$$\Delta E_i = - 0.5\Delta x_i * \sum_j (w_{ij}x_j) \qquad \text{Equation 5.10}$$

Considering equation 5.10, if x_i is currently 0, it is because the weighted sum of the inputs is negative. For x_i to become 1 ($\Delta x_i = 1$), the weighted sum must become positive. The result is that the energy decreases, ΔE_i in equation 5.10 is negative.

Similarly, if the weighted sum is positive, x_i will be positive. For x_i to become 0 ($\Delta x_i = -1$), the weighted sum must be negative. So, the result is the energy of the system decreases. Thus, an element in the network will only change if and only if it will decrease the overall energy of the network. As a result, the network is guaranteed to converge. However, this logic is valid only if the neural sets change one at a time in some random order. Also, it is approximated by a low neural firing rate.

5.9.3. Counter-propagation

Hecht-Nielsen designed this network (Neural Computing, 1991). Counter-propagation selects from a set of exemplars by allowing them to compete amongst each other. Normalised inputs and competition between exemplars selects the nearest neighbour. This gives a technique of building an adaptive pattern classifier.

This network constructs a mapping from a set of input vectors $\underline{X}$ to a set of output vectors $\underline{Y}$. This acts as a hetero-associative nearest-neighbour classifier. Figure 5.9 below illustrates this form of network architecture.

As shown in figure 5.9, the input layer acts a buffer. The network process requires that every input vector has the same length, these input vectors are normalised.

So, to normalise to a value of 1:

$$||\underline{X}|| = 1 \qquad \text{Equation 5.11}$$

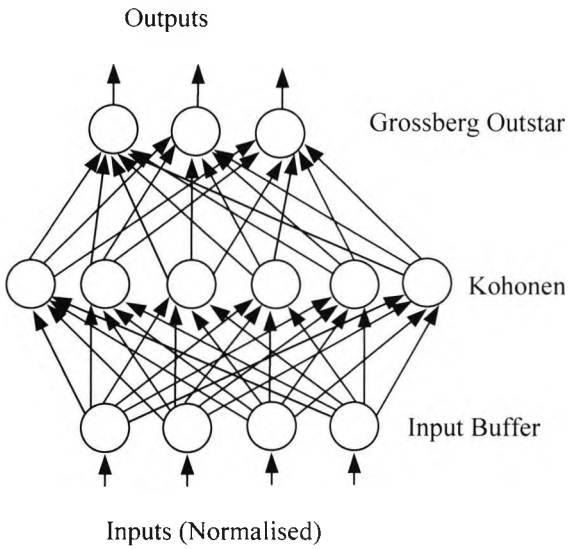


Figure 5.9: Uni-flow counter-propagation network (Neural Computing, 1991)

When this requirement is met, the entire input vector $\underline{X}$ lies on a unit sphere, which is indicated in figure 5.10. The Kohonen layer acts as a nearest neighbour classifier. The processing elements in this layer compete. The one with the maximum output wins. Recalling the Kohonen learning rule:

$$W_{new} = W_{old} + \alpha (X - W_{old}) \qquad \text{Equation 5.4}$$

Which in vector form is:

$$\underline{W}_{new} = \underline{W}_{old} + \alpha (\underline{X} - \underline{W}_{old}) \qquad \text{Equation 5.12}$$

Considering the figure 5.10, the vectors $\underline{W}$ will tend to rotate around the unit sphere until they are at the centre of those input vectors $\underline{X}$ that create a maximum output. In vector form, the output I of the processing element (before competition) is:

$$I = \underline{W} \cdot \underline{X} \qquad \text{Equation 5.13}$$

This inner product is:

$$I = \|\underline{W}\| * \|\underline{X}\| * \cos (t) \qquad \text{Equation 5.14}$$

Since the length of both the weight vector $\underline{W}$ and the input vector $\underline{X}$ are 1, then equation 5.14 reduces to:

$$I = \cos (t) \qquad \text{Equation 5.15}$$

Where t is the smallest angle of the processing element with the closest weight vector

The Kohonen layer measures the cosine of the angle between the input vector X and the weight vector W. The processing element with the closest weight vector (smallest angle t) has the highest output and wins. So, during learning, the processing elements in the competitive layer adjust their weight vectors to divide up the input space in approximate correspondence to the frequency with which the input occurs. The output layer uses a Grossberg Outstar (Neural Computing, 1991). This is a processing element, which learns to give a specific output when certain input is applied. Because the Kohonen layer delivers a signal output by competition, this layer provides a way of decoding that input to an output class.

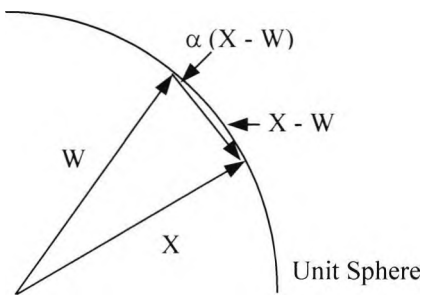


Figure 5.10: Kohonen learning in vector form (Neural Computing, 1991)

5.9.4. Adaline and Madaline

Widrow (Neural Computing, 1991) developed the Adaline (ADAptive LInear Neuron). It is like the Perceptron, which is capable of classifying linearly separable patterns. However, specific multi-layer extensions to the adaline concept enhance a better method for separating the input space.

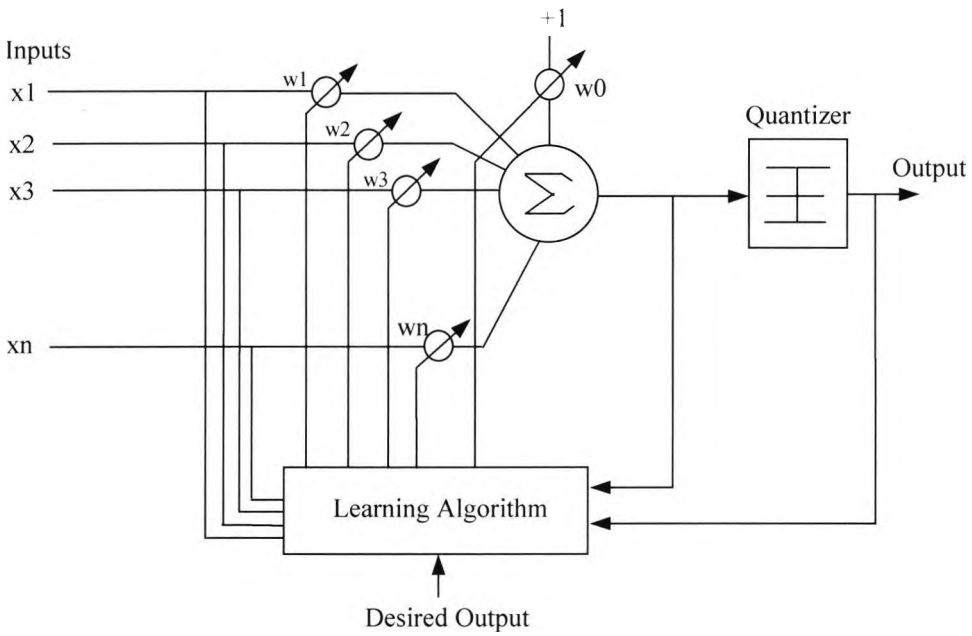


Figure 5.11: Adaline processing element (Neural Computing, 1991)

Figure 5.11 illustrates the basic adaline processing element, which contains a series of trainable weights. A bias weight is connected to a fixed input of +1. The inputs are bi-state (-1 and +1), where weights are signed both positive and negative. The weighted sum of the inputs and the bias are applied to a quantizer, which converts the output to -1 or +1. A learning control mechanism samples the inputs, the output, and the desired output and uses these to adjust the weights. The weights must be set to random values initially.

Several types of the adaline learning algorithm exist. Pattern-Recognising Control System is one of them (Widrow & Smith, 1963). The systematic approach for this follows the sequence:

- Apply the selected input to the processing element and the desired output to the desired output.
- Compute the error - The difference between the weighted sum prior to the quantizer, the desired output.
- Adjust each of the weights so that they reduce the error 1/n (n = number of weights) of the error at the input to the quantizer. Because all of the inputs are either + or – 1, this adds or subtracts a fixed amount from each input, depending on its sign.

The Widrow – Hoff Learning is the calculation of the error signal for each iteration and changing the weights to get rid of the error.

5.9.5. The back propagation network

A back propagating network is commonly used as a three layer network, this shown in figure 5.12. In this type of network, all information flows forward from the input neurons to the neurons in the hidden layer, and from there to the output neurons. No information is passed backward during actual running of the network; the back propagation refers simply to the learning stage.

This algorithm is a generalisation of the method of least squares (Widrow-Hoff) for training a basic Perceptron, as applied to multi-layer Perceptrons.

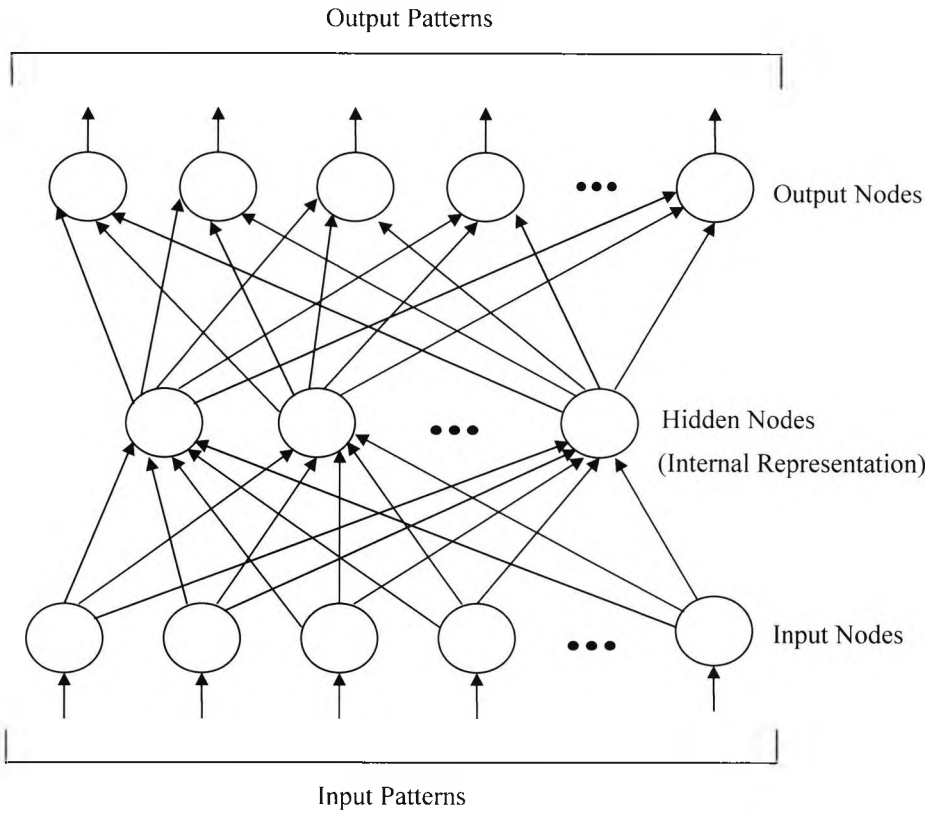


Figure 5.12: The architecture of the back propagation network

5.9.5.1. Network processes

The network is presented with pairs of patterns; an input pattern paired with target output. For each presentation, weights are adjusted to decrease the difference between the network's output and the target output. A training set, which is a set of input and target pattern pairs, is used for training, presented to the network many times. After training is stopped, the performance of the network is tested.

The algorithm uses two steps; a forward propagating step followed by a back propagating step. Both the forward and back propagation steps are completed for each pattern presented during training.

5.9.5.1.1. Forward step

This step begins with the presentation of an input pattern to the input layer of the network, and continues as activation level calculations propagate forward through the hidden layers. In each successive layer, every processing unit sums its inputs and then applies a sigmoid or a similar transfer function to compute its output. The output layer of units then produces the output of the network.

5.9.5.1.2. Backward step

The backward step begins with the comparisons of the network's output pattern to the desired output, where the difference (error), is calculated. The backward propagation step then calculates error values for hidden units and changes for their incoming weights, starting with the output layer and moving backward through the hidden layer. In this back propagating step, the network corrects its weights in order to minimise the overall error.

A more detailed discussion of this particular network is covered in chapter 6.

5.10. Application of neural networks

Once there has been a decision to use a neural network approach, there are a variety of different types of neural networks to choose from. Every neural network serves a different type of purpose. Feed-forward networks are used for pattern classification, for example. Single layer laterally connected networks are suited for autoassociation. Feed-forward feedback networks are good for heteroassociation. Multi-layer, co-operative and competitive networks are useful for sophisticated processes in which the interrelation of different types of information helps in making some sort of decision. Also, sometimes, no single network can solve a difficult problem, and a system of interacting networks must be created in order to resolve the task.

It is vital to consider a variety of neural network architecture for a specific task. Indeed, the choice of a neural network depends on the nature of the task application (Caudill, 1992), such as:

- Mapping - This is a problem where an input pattern is linked with a particular output pattern. For example, the input pattern that contains a pixel image of a character could be mapped to the output pattern of that character's ASCII code. Back propagation and counter propagation networks are very good at these problems
- Associate memory - This stores information by associating it with other information. Recall is completed by giving the association and having the network deliver the stored information. So, if the network is required to produce one of the output patterns it has been trained with, to recall it, this would be an associate memory problem. These problems can be accomplished by a variety of networks, which are back propagation, counter propagation and Kohonen .

- Categorisation - This is when the inputs are to be clustered into categories. The network is given with an input pattern and it responds with the category to which the pattern belongs. Self-organising networks is the relevant choice, such as Kohonen and adaptive resonance networks.
- Temporal Mapping - The input data has a consideration of time, for example the input may be a time sequence of sensor readings that must map to an output pattern. Many process control patterns are temporal mapping problems. The most suited networks are avalances, back propagation and recurrent network.
- Image Processing - A high resolution image can contain typically 1,024 x 1,024 pixels, or more than one million elements in the input vector, so because of the limited size of current neural networks, image processing problems need special pre-processing techniques and the size of input data must be reduced. A categorising or mapping network is suitable to use.

5.11. Selection of a model-the back propagation network

The most commonly used network for pattern recognition is a multi-layered feed-forward network, namely the back propagation network (Maren, Harston & Pap, (1990:220-221)). As discussed, the network of this structure is most suited to the application of the impact echo data. Essentially, this application is a mapping problem, which makes it more, suited to this type of network.

The back propagation network is good at generalisation (Freeman, 1992), this means that given several different input vectors, all belonging to the same class, a back propagation network will learn to key off significant similarities in the input vectors. Irrelevant data will be ignored. This is most appropriate in the impact echo data, where there are two dominant frequencies (Cheng & Sansalone, 1991). Here, the transient waveform has two characteristics:

- A dominant frequency relating to reflection off the bottom surface of the test object.
- A dominant frequency corresponding to reflection off the defect surface within the test object.

The remaining data in the transient waveform is complex, having other frequencies such as rod and the test object's natural frequencies. Noise is another factor in the transient waveform.

It has been suggested in the Proff/II plus neural network software (Neural Computing, 1991) that the most frequent cited classification networks are:

- Categorical Learning
- Counter-propagation
- Learning Vector Quatization (LVQ)
- Probabilistic Neural Networks
- Self-Organising-Map into Categorisation

The probabilistic network is a specialised classifier, which might be appropriate for use; other than it does not have the wide applicability of the back propagation network. Also, although few training samples are needed, as summarised in table 5.1, a training set needs to be in great detail. This is good practice for any neural network model, which benefits from having a comprehensive training set; though this network is more selective in its approach to training. Again, with the back propagation network, if only a few samples are available, with few hidden neurons, network can be trained, and it will probably do a good job at generalising far from known points. The unsupervised network types, categorical learning, counter-propagation, LVQ, self-organising-map into categorisation are not suited to this problem. These forms of the Kohonen network which is a weak classifier (Masters, 1993). It has no hidden layers and is mainly linear in its response. Also, most Kohonen networks have a very large number of output neurons, whereas, in this research there a few output nodes and many input nodes. It is concluded that a back propagation network is the best solution for this application.

5.12. Summary

The history of neural network including the major contributors has been presented. The basic theory of the neuron and a discussion of a variety of networks have been presented to enable the most appropriate neural network architecture to be selected for this research. The back propagation network has been selected for modelling the impact echo problem, which will be discussed in chapter 6. The principal reasons for its selection are:

- Good at generalisation
- Network architecture is most suited to the impact echo application
- Most widely used network
- Immediate availability to author

The next chapter develops further the operation of the back propagation network.

Chapter six

The back propagation network

6.1. Introduction

In the previous chapter, the general types of neural networks was reviewed, culminating in the selection of back propagation network. The back propagation network is thus the focal point in this chapter. In this chapter the significant aspects of this network are discussed in more detail. The back propagation network, which uses a sigmoid transfer function and the generalised delta rule as its learning law, is covered in both mathematical and non mathematical terms. The importance of the sigmoid transfer function and local minima is also presented.

6.2. Back propagation network

The back propagation network is an outcome of earlier work on Perceptrons, with the addition of a hidden layer and use of the generalised delta rule for learning. Indeed, it is a Perceptron with a different transfer function in the artificial neuron and a more robust and capable learning rule.

Networks of these type use supervised training. As can be seen from figure 6.1, the network has no reverse or lateral connections. The name of the network is associated with the search for various heuristics that allow it to find an approximate solution for the training problem in a limited time.

The back propagation error method is one of the best algorithms for training multi-layer Perceptrons (Shubnikov, 1997), and among the currently most widely used types of neural network systems. The algorithms represent a generalisation of the Widrow-Hoff rule (Widrow & Hoff, 1960) for training a simple Perceptron, as applied to multi-layer Perceptrons. The root mean square (rms) error between the actual and expected outputs is minimised, and the initial weights of the connections and the thresholds are random.

The back propagation network has three layers as illustrated in figure 6.1. It comprises the following:

- Input Layer - this is where the input data is entered and is partially labelled
- Hidden Layer - this layer forms the internal representation of the training set which is not provided by the programmer
- Output Layer - at this level a final decision is made on the data

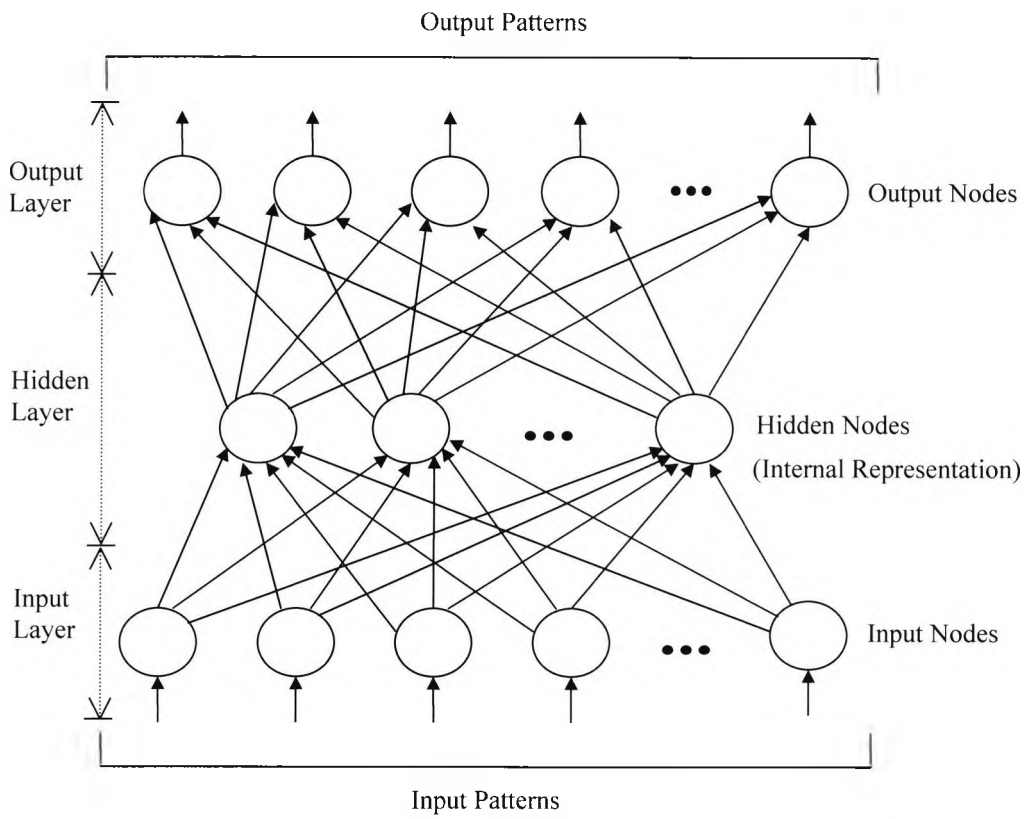


Figure 6.1: The architecture of the back propagation network

6.3. Characteristics of the back propagation network

The primary characteristic of the back propagation network is that it forms a mapping from a set of input nodes to a set of output nodes, using features extracted from the input pattern. This network can be designed and trained to deliver a whole range of mappings. It is possible to do this because the nodes in the hidden layer(s) of the network learn to respond to features found in the input. The advantages and disadvantages of using hidden nodes are (5th Annual Neural Networks Summer School, 1995):

- Advantage: the hidden nodes can be used to extract more complex features from the input layer. permitting a more complicated task to be learned
- Disadvantage: learning becomes time consuming with the addition of hidden nodes

This network typically requires long periods of training in order to learn pattern classes. The training set for this network has to be presented many times, about 100 – 10,000 times or even more in certain circumstances, in order for the interconnection weights between the neurons to settle into a stable pattern that correctly classifies the input patterns (Maren, Harston,Pap (1990: 92-103)).

Whilst the back propagation delivers high performance when given patterns similar to those it has learned, it does not have the ability to recognise new categories of patterns, which is typical with supervised learning. If the network is to recognise and classify a new pattern, it needs to be given examples of the new pattern type along with the corresponding category identification. Furthermore, since training a new pattern involves changing already stabilised inter-neuron connection weights; the network must also be retrained on previously known categories. This enables the new connection weights to give correct decisions for all categories. The need for long training and retraining new categories is one of the major limitations of this type of network.

6.4. Activation functions

It is vital for a network to be trained to produce realistic activation levels, rather than extreme values that are difficult or impossible to achieve. For example, the majority of current models use a sigmoid (S-shaped) logistic activation function:

$$f(x) = \frac{1}{1 + e^{-x}} \qquad \text{Equation 6.1}$$

The sigmoidal shape of the transfer function implies that, for most values of x which is the sum of inputs, the value given by the transfer function is close to one at the asymptotic values. Generally, the higher valued asymptote is 1, and the lower one is 0, but these values can never be attained. The network is usually trained on moderate values, such as 0.1 for one decision and 0.9 for another decision.

The importance of this transfer function, or any similarly shaped function, is that its derivative is always positive, and close to 0 for either large positive or large negative values of x. When the derivative is at its maximum, x is 0. This is essential in order for the back propagation algorithm to work well. As can be seen below in the

mathematical analysis of the algorithm and in figure 6.2, the changes made to the weights are proportional to the derivative of the activation. This means:

- If the derivative is near 0, then the changes are small. This is satisfactory, because the derivative is near 0 when the activation value is near 0 or 1,one of the two stable states.
- When the activation of the neuron is in the middle range, the neuron's output requires to be changed greatly, forcing it to produce a value near one of the stable states.
- In the middle range, the derivative is largest. Here the change in the weights, which is proportional to the derivative, is large.

Hence, the sigmoid transfer function provides smooth and differentiable behaviour, allowing control of the learning law.

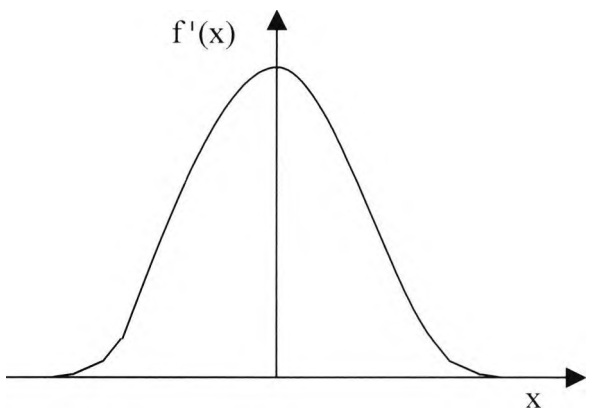


Figure 6.2 : Derivative of a sigmoidal curve

6.5. Local minima

Figure 6.3 shows the idea of converging to a local minimum in weight space, which is the Euclidean space of dimension equal to the total number of weights. Each point in this space represents a vector of weight values. Also, as shown in equation 6.7, the network computes an error or energy function, which represents the amount by which the output of the network differs from the required output. Because the output of the network is related to the weights between the layers and the input applied, the energy is a function of the weights and inputs to the network (Beale & Jackson, 1992). Figure 6.3 is a graph illustrating the energy function showing how varying the weights affects the energy, for a fixed input pattern p. It is a multi dimensioned space in which each axis represents one specific weight, so one point in the space would represent one unique

combination of possible weight values that the network could have. This is the weight space.

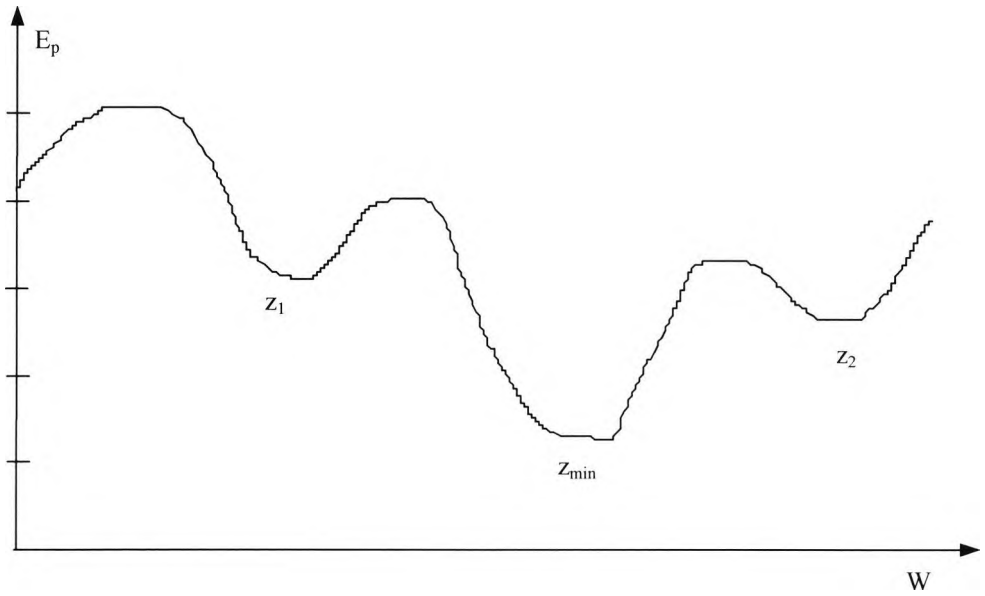


Figure 6.3: Cross-section of a hypothetical error surface in weight space (Freeman, 1992), Z_{min} is the global minimum, and Z_1, Z_2 are other minima's.

The generalised delta rule aims to minimise the error function E_p by adjusting the weights in the network so that they correspond to those at which the energy surface is lowest. When the network stabilises on a minimum, whether it is local or global, learning comes to a halt. If a local minimum is achieved, the error at the network outputs may still be unacceptably high (Freeman(1992:55-63)). However, in reality this problem does not cause much difficulty. If a network stops learning before reaching an unacceptable target, the number of hidden nodes or learning parameters can be adjusted to combat this. Alternatively, learning can be started all over again with a different set of initial weights. In fact, there is no guarantee that the network has reached the global minimum rather than a local one when the network has reached an acceptable solution.

6.6. Training steps

As discussed in the chapter 5, there are two training steps:

- First step - This is the forward stage, where the input is applied and allowed to propagate to the output. The error values of the output units are calculated and compared to the known target output.
- Second step - This is the backward phase, where the errors are propagated backwards and weight changes adjusted. This means when the output error has been

calculated, the weight changes in the output layer can be made using equations 6.5 and 6.4. These fixes the error values for the previous layer using equation 6.6, and the weight changes made using equation 6.4 in the first inner layer. This technique may be propagated backwards until the weights in the input layer are adjusted. Hence, these two training steps are repeated again and again. This procedure is summarised in figure 6.4:

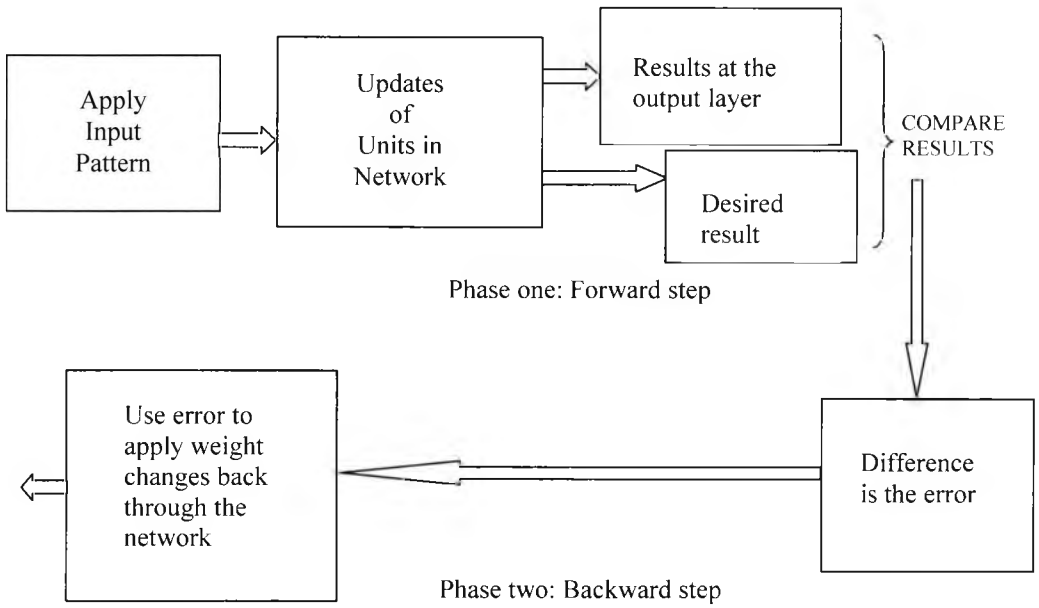


Figure 6.4: Stages of the forward and backward steps for training the back propagation network (Dayhoff, 1990)

6.7. The generalised delta rule

This is learning algorithm for the multi-layer feed-forward network, which controls the training of the hidden unit by a process of propagation of measured errors from the output layer.

If the activation of the jth unit for the pth training pattern is:

$$a_{pj} = \sum_{(\text{for all } i)} w_{ji}\, o_{pi} + u_j \qquad \text{Equation 6.2}$$

where a_{pj} is the activation of the node j of pattern p

i is the sum of all inputs at the ith unit

w_{ji} is weight from node i to j

u_j is threshold at node j

o_{pi} is the actual output on node i

To establish an output function expressed globally, so that the weights for the neural nodes can be altered:

$$o_{pj} = f_j (a_{pj})$$

Equation 6.3

where o_{pj} is the actual output on node j.

f_j is the threshold function acting on the weighted sum, this is the output from each unit j.

This deals with a whole class of neuron models where the output can be specified in a general way. So, for a feed-forward net with input, output and hidden layers (Aleksander & Morton, 1990):

- For the pth presentation of an input/output pair for training, the change for the weight which joins the jth unit to its ith incoming connection ($\Delta_p w_{ji}$) is proportional to some computed error (δ_{pj}) for this jth unit, therefore:

$$\Delta_P w_{ji} = \alpha \delta_{pj} o_{pi}$$

Equation 6.4

where

δ_{pj} is the computed error for the jth node

α is the constant which determines the rate of learning

$\Delta_P w_{ji}$ is the change in weight

- For output units, the error is calculated on a knowledge of a desired target output for the jth unit, t_{pj}, the error is expressed as :

$$\delta_{pj} = (t_{pj} - o_{pj}) f'_j (a_{pj})$$

Equation 6.5

where

t_{pj} is the target output for pattern p on node j.

$f'_j (a_{pj})$ is the rate of change of o_{pj} with respect to $f_j(a_{pj})$

Also, the term $(t_{pj} - o_{pj})$ indicates that the error is proportional to the difference between the actual output o_{pj} and the target output t_{pj}.

- For a hidden unit, and its output connected to k units, its error is defined as being proportional to the sum of the errors of all these k units as modified by the weights connecting these units:

$$\delta_{pj} = \left[\sum_{(\text{for all } k)} \delta_{pk} w_{kj} \right] f'_j (a_{pj}) \qquad \text{Equation 6.6}$$

where δ_{pk} is the computed error for the kth node

w_{kj} is the weight from node j to k

The use of these equations have been discussed in section 6.6. Which can be used repeatedly until an acceptable level is achieved. Rumelhart et al (Rumelhart, Hinton & Williams, 1986) showed that overall output error can be expressed as half the sum of the squares of all the output errors:

$$E_p = 0.5 \sum_{(\text{for all } j)} (t_{pj} - o_{pi})^2 \qquad \text{Equation 6.7}$$

where E_p is the error function for pattern p

They suggested that an S shaped function illustrated in figure 6.5 is beneficial, which in this case becomes:

$$o_{pj} = 1/(1 + e^{-apj}) \qquad \text{Equation 6.8}$$

where:

$$a_{pj} = \sum_{(\text{for all } i)} w_{ji} o_{pj} + u_j$$

As discussed in section 6.4, the benefit of this function is that its derivative is easily found. Other sigmoid functions, such as hyperbolictangent and arctangent can also be used.

Since:

$$f'_j (a_{pj}) = \delta o_{pj} / \delta a_{pj}$$

$$f'_j(a_{pj}) = o_{pj}(1 - o_{pj}) \qquad \text{Equation 6.10}$$

for output nodes, equation 6.5 becomes:

$$\delta_{pj} = (t_{pj} - o_{pj})\, o_{pj}(1 - o_{pj}) \qquad \text{Equation 6.11}$$

for hidden nodes equation 6.6 becomes:

$$\delta_{pj} = \left[\sum_{\text{(for all k)}} \delta_{pk} w_{kj}\right] o_{pj}(1 - o_{pj}) \qquad \text{Equation 6.12}$$

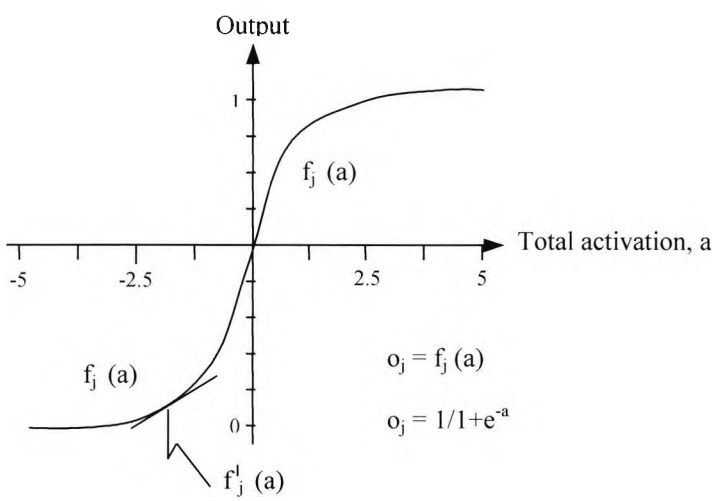


Figure 6.5: Characteristics s-shape of the sigmoid function (Aleksander & Morton, 1990)

6.8. Summary

The factors concerning the back propagation network have been discussed, including the sigmoid transfer function, which has been shown to be important in order to control the learning law. Mathematical and non-mathematical approaches have been defined to illustrate the generalised delta rule, and the important equations of this learning law have been presented. In the next chapter, the implementation and results of the back propagation are presented for the impact echo problem.

Chapter seven

Neural network implementation

7.1. Introduction

This chapter deals with the back propagation network application, for which the theory was presented, in chapter 6. The factors which affect a network performance are explained. Modelling of the impact echo for neural network usage is covered. Then, the method of optimising the number of hidden nodes, learning rate, momentum term is dealt with. The results of an investigation into how increasing the number of data sets can make a difference to the back propagation network is then presented.

7.2. Types of data

For all cases of neural network testing, there were 225 data sets from the 15 tested locations (see page 4, chapter 3) which were amplitude spectrum graphs defining a pattern for each defected case. This was taken from walls A and D (see chapter 3 for more details). There were three defect cases of investigation:

- Sound concrete
- Honeycomb concrete
- Voided concrete

7.3. Training a network

There are no specific rules for how best to train a network for an application. With many aspects of neural network systems, it seems that experience is the best teacher (Freeman, 1992). By using as many neural networks as possible, an appreciation for how to select and prepare training sets is achieved. Due to the time limit, this has proved to be not possible. However, selection and preparation of data has been given substantial consideration.

The back propagation network does not extrapolate well, so, if it is inadequately or insufficiently trained on a particular class of input vectors, then subsequent identification of members of that class may be unreliable. Thus, it has been stated that (Freeman, 1992) it is essential to:

- Ensure that the training data covers the whole-expected input space.
- Choose training vector pairs randomly from the available set.

7.4. Validation

The competence of a neural network must be analysed before the trained network is placed into service. This process is called validation, and is achieved by separating data cases into two parts:

- Training set
- Validation set

Usually, a neural network software allows the user to validate the data before the main analysis, for example. This procedure operates by an adaptive method, where the program tries to find the appropriate set of the learning rate and momentum term for a particular application to achieve maximum results.

7.5. Momentum term

In neural network training, momentum is expressed as:

$$\delta_p w_{ji}(t+1) = w_{ji}(t) + \alpha\delta_{pj} o_{pi} + \beta(w_{ji}(t) - w_{ji}(t-1)) \qquad \text{Equation 7.1}$$

where $w_{ji}(t)$ is the weight from node i to node j at time t (present time)

$w_{ji}(t+1)$ is the weight from node i to node j at time t+1 (future time)

$w_{ji}(t-1)$ is the weight from node I to node j at time t-1 (previous time)

β is the momentum factor $0 < \beta < 1$

δ_p = error term for pattern p

Momentum is a method for increasing the speed of convergence. When calculating the weight change value, a fraction of the previous weight value is added. This keeps the weight changes going in the same direction (momentum). This additional term is applied to the weight change equation on the output layer (see chapter5) and on the hidden layer. The momentum parameter is commonly set to a positive value less than one (Freeman, 1992). Use of the momentum term is optional.

7.6. Learning rate

The back propagation network is a hill descending procedure, which can become stuck in local minima, as was discussed in the chapter 6. This problem can be solved if the

network is restarted for different initial weight values, so that the correct solution will appear as being dominant. As illustrated in figure 6.3 and mathematically expressed in equation 6.4, the learning rate α is given by:

$$\Delta_P w_{ji} = \alpha \delta_{pj} o_{pi} \qquad \text{Equation 6.4}$$

recalling that $\Delta_P w_{ji}$ is the weight changes for the pattern p

δ_{pj} is an error term for pattern p on node j

o_{pi} is the actual output at node i for pattern p

The figure 6.3 in chapter 6, has sloping ravines, and has an optimal point that is, a lowest point, which represents the best solution. The size of the steps to get to that point is referred to as the learning rate. The learning rate parameter α affects the network performance.

If the rate at which the weights are changed is steadily decreased, the generalised delta rule is able to obtain a better solution. If the learning rate α is made large initially, large steps are taken across the weight and energy space towards the solution. As the learning rate is decreased, the network weights settle into a minimum energy configuration without overshooting the stable position, as the gradient descent takes smaller downhill steps. This technique allows the network to bypass local minima at first, then locate, and settle into deeper minima without oscillating widely. However, the reduction in the learning term means the network will take longer to converge.

This term must be a small number of the order of 0.05 to 0.25, to ensure that the network will settle to a solution (Freeman, 1992). Hence, it is vital to select a learning parameter that delivers maximum rate learning, without the values oscillating wildly (Carling, 1992). The importance of the number of hidden nodes in a network, the learning rate, and time taken to train is illustrated in table 7.1.

Number of hidden nodes used	**Learning Rate**	**Number of iteration**
2	0.25	245
32	0.25	120
8	0.1	450
8	0.75	68

Table 7.1: A table showing the relationship between the number of hidden nodes, the learning rate and number of iterations (Carling, 1992)

Considering table 7.1, it has been stated that adding more hidden nodes does not necessarily greatly reduce the time to learn, and, if the learning rate was greater than about 0.75, it is found to give unstable behaviour.

7.7. Weights

As illustrated in figure 6.1 of chapter 6, the interconnection between the nodes has an associated connection strength, the weights. These weights represent the influence of its inputs over its output value. Depending on the sign of the weight, an input may exite or inhibit the neuron.

It has been suggested that weights should be initialised to small, random values in the range +0.5 or –0.5 (Freeman, 1992).

7.8. Number of iterations

The number of iterations refers to the number of training examples presented to the network when it is learning. For example 50,000 iterations have been chosen in the training of the back propagation network. There is an optimal amount of training for a network; achieved by training a network until there is no further significant improvement in its performance, indeed this has been noticed with the 50,000 iterations, which was the optimum.

7.9. Hidden layers and nodes

It has been suggested to use as few hidden nodes as possible, because each unit adds to the load on the central processing unit. For networks of hundreds or thousands of inputs, the hidden layer needs to be a small fraction of the size of the input layer (Freeman, 1992). If the network fails to converge to a solution, it may be that more hidden nodes are required. If it does converge, the number of hidden nodes may be reduced so that the network can settle on a size on the basis of overall system performance.

Also, it is possible to remove hidden units that are superfluous. If the weights values on the hidden nodes are examined periodically as the network trains, it may be possible to see that weights on certain nodes change very little from their starting values. These nodes may not be participating in the learning process, and fewer hidden nodes will then suffice.

7.9.1. Calculation of the number of hidden layers

The pyramid shape used by Masters (1993) has been used as a guide to calculate the number of hidden layers and nodes in this research; this is called the geometric pyramid rule, which is shown in figure 7.1. This author has shown not to use more than 2 hidden layers. Indeed, for many practical problems it has been stated not to use more than one hidden layer.

7.9.2. Calculation of the number of hidden nodes

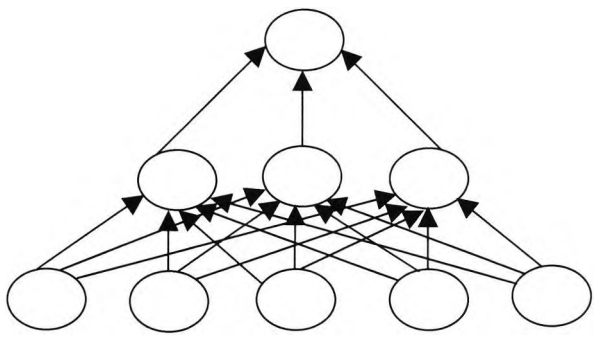


Figure 7.1: A typical three-layer network

The number of hidden nodes was calculated by using the equation below:

$$\text{Hidden Nodes} = (MN)^{1/2} \qquad \text{Equation 7.1}$$

Where M is the number of output nodes and N is the number of input nodes. Also, it has been stated that the number of hidden nodes in a back propagation network must be very small compared to the number of training cases (Neural Computing, 1993). The training set size and hidden layer sizes are thus linked together. When they are unbalanced in one direction, the network is unable to learn as well as it should. Similarly, when unbalanced in the other way, the network learns too much and learns spurious relationships. It will learn irrelevant aspects of the training set, hence generalises poorly, so there must be a balance.

It is essential to use the bare minimum number of hidden neurones required to obtain an acceptable performance. Increasing the number beyond the minimum will cause deterioration in generalisation ability.

The factors discussed in section 7.3 to 7.9 have all been taken in consideration in producing the neural network model and results.

7.10. Modelling impact echo by neural network

Data was trained using the back propagation network. This was collected using an impact echo device, as described in chapter 3. The method of application and wall specimen geometry are given in chapters 2 and 3.

The input nodes of the network take amplitude values, which were achieved, from FFT processing of the signal data. There is one output node, which contains the theoretical frequency for a given defect, which is predicted using the following equations reproduced here (presented in chapter two):

$$T = C_p * \Delta t/2 \quad \text{if } \Delta t = 1/f, \text{ then} \qquad \text{Equation 2.1}$$

$$f_p = \frac{C_p}{2\,T} \qquad \text{Equation 2.2}$$

Where C_p is the wave speed in concrete.

T is the distance to reflecting interface of the defect

Δt is the round trip travel from the start of the pulse to the arrival of the first P-wave. (See chapter 2 on impact echo theory)

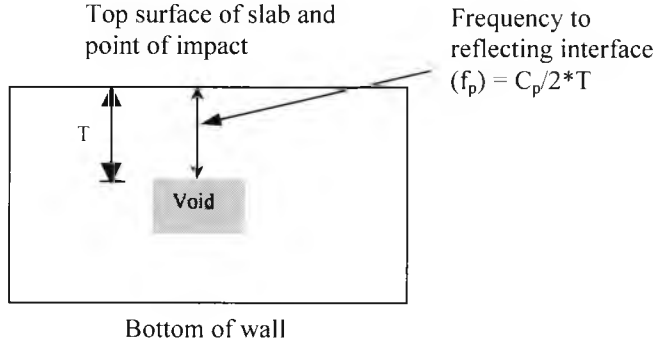


Figure 7.2: Impact echo arrangement (not drawn to scale)

This arrangement can be seen in figure 7.2. For example, when the theoretical velocity is 3507.5 m/s and known depth is 0.1016 m, then using the equation 2.2, the theoretical frequency is 17.261 kHz.

Figure 7.3 is given to assist understanding the FFT graph for impact echo. In figure 7.3, there is a dominant peak at about 18 kHz. In this case, this would be the main feature of

the graph. The design of the input file for the neural network is illustrated in the figure 7.4.

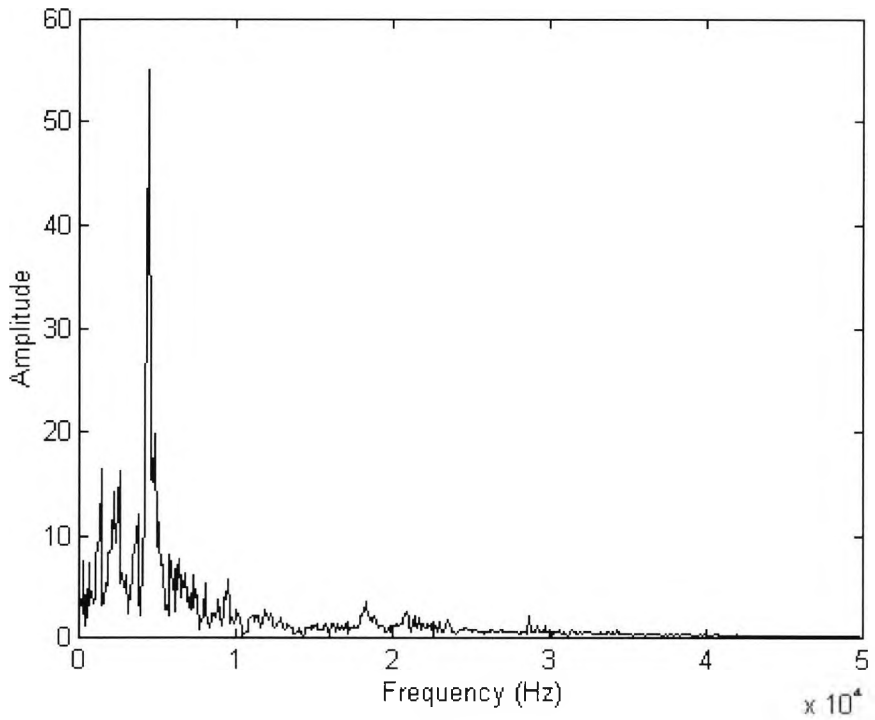


Figure 7. 3: An example of a FFT graph

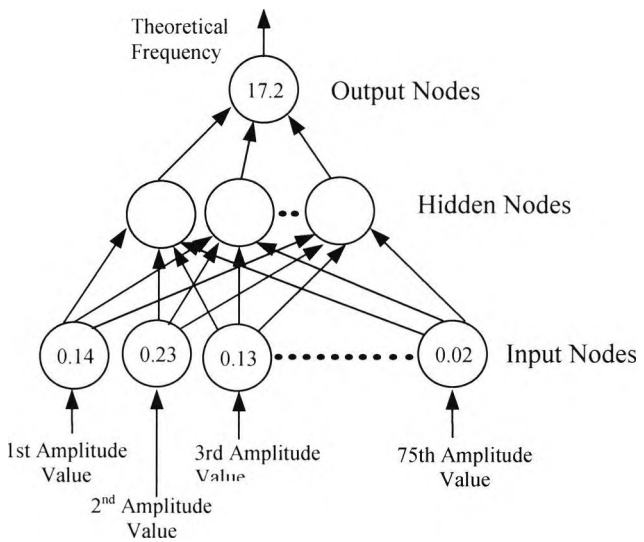


Figure 7.4: The design of the back propagation network used for the analysis of impact echo data

Considering figure 7.4, the input node represent the amplitude values obtained from pre-processing, which includes FFT processing, interpolation and filtering. If there are 75 pre-processed values then there will 75 input nodes. As discussed, the calculated theoretical frequency will be the output node. Examples of a training file and testing file

are given in appendix A. This method of interpretation of an FFT graph has been applied to the testing of data correlation, optimisation, and the effect of increasing the number of data sets in testing.

7.10.1. Method of data correlation and network optimisation

Optimisation is an important aspect of the neural network testing, experimentation with various network designs has been studied in order to obtain optimum performance. A parametric study for optimisation is undertaken, one factor changed to see the effect of the results it had on the results, while the other factors were kept constant. The objective was to examine the number of iteration and the effect of increasing or decreasing hidden nodes.

For neural network analysis, the raw data obtained from the impact echo testing had to be pre-processed. This was completed to improve the learning of the network, hence producing a better result. The process of this procedure is shown in figure 7.5.

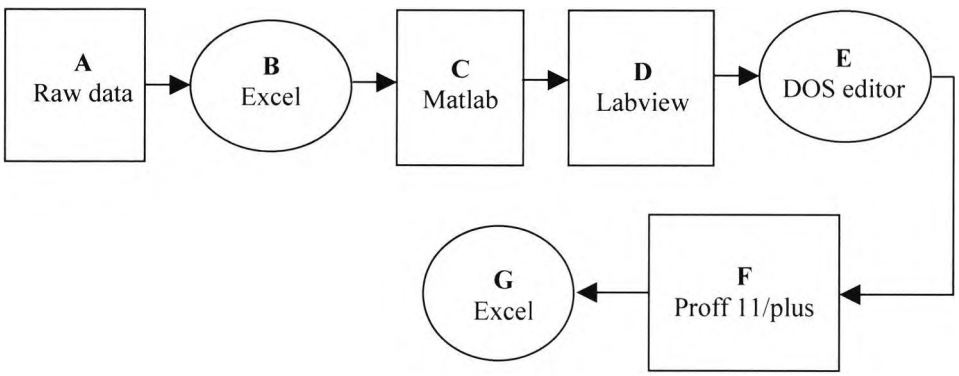


Figure 7.5: An overall procedure of pre-processing, testing and post-processing for neural network analysis

At stage A: For each of the sample points indicated in figure 3.5 of chapter 3, the impact hammer was applied about ten times. For each hit, the output was a transient wave.

Stage B: Using Excel, the data consisted of a column of 15 sets of data, each data set had 1024 data points, which were 15 responses for a given test point, produced by the impact echo device. The data was then split into files with the 1024 points.

Stage C: Using Matlab, these text files were converted into Matlab file format as .m files. Then, for each file, FFT was performed and results plotted and saved to a new file.

Stage D: Using Labview, the data is now reduced to 512, since the remaining 512 are only a reflection (FFT theory). It is further reduced to 256. This was done by a program in Labview, which takes the first two data points in an array and finds the average, and then moves on to the next two data points from the one dimensional array, and so on. This procedure was repeated 225 times for the 256 data points, each time the Labview program accumulated the data to file. So, it finished with 225 data sets, each set having 256 data points.

Stage E: The file created at stage C is added with the calculated theoretical frequency using equation 2.2, and then split into two files, one for training the neural network and the other for testing (see appendix A for the structure of an input file). The training file contains more data examples than the testing file.

Stage F: The learning and testing file is included in the neural network learning. The network is tested and a result file produced.

Stage G: The output of the neural network is observed and the equivalent depths, error, error percentage and other information processed and saved as Excel files.

7.10.1.1. Method of data evaluation

Actual Frequency (Hz)	Detected Frequency (Hz)	Actual Depth (mm)	Detected Apparent Depth(mm)	Error (mm)	Error (%)
17261	16982.9	101.6	103.3	-1.7	-1.6
17261	17788.0	101.6	98.6	3.0	3.0
17261	17418.9	101.6	100.7	0.9	0.9
9864	11628.2	177.8	150.8	27.0	15.2
17261	17217.6	101.6	101.9	-0.3	-0.3
17261	16069.2	101.6	109.1	-7.5	-7.4
13809	14325.8	127.0	122.4	4.6	3.6
11508	14809.5	152.4	118.4	34.0	22.3
17261	16692.4	101.6	105.0	-3.5	-3.4
17261	13638.7	101.6	128.6	-27.0	-26.6
17261	17115.9	101.6	102.5	-0.9	-0.8
17261	9446.9	101.6	185.6	-84.0	-82.7
5754	6928.4	304.8	253.1	51.7	17.0
5754	8574.2	304.8	204.5	100.3	32.9
5754	10878.9	304.8	161.2	143.6	47.1

Table 7.2: Results of 50,000 iterations, 3 hidden nodes

In order to simplify the impact echo data, a data correlation method was used. This was done by studying plots of the 256 points. It was discovered that some graphs were

similar to one another, so if there were 4 graphs, which looked identical, then one graph from that class would be used to represent that group of defect or non-defect. About 50 training examples were presented to the network, and 15 examples were used to test the network.

Actual Frequency (Hz)	Detected Frequency (Hz)	Actual Depth (mm)	Detected Apparent Depth(mm)	Error (mm)	Error (%)
17261	15549.0	101.6	112.8	-11.2	-11.0
17261	17486.8	101.6	100.3	1.3	1.3
17261	16731.1	101.6	104.8	-3.2	-3.2
9864	11767.8	177.8	149.0	28.8	16.2
17261	16534.1	101.6	106.1	-4.7	-4.4
17261	15675.7	101.6	111.9	-10.3	-10.1
13809	11074.4	127.0	158.4	-31.4	-24.7
11508	14844.3	152.4	118.1	34.3	22.5
17261	16300.8	101.6	107.6	-6.0	-5.9
17261	11687.6	101.6	150.1	-48.6	-47.7
17261	17244.9	101.6	101.7	-0.1	-0.1
17261	13114.1	101.6	133.7	-32.1	-31.6
5754	9640.0	304.8	181.9	122.7	40.3
5754	11895.0	304.8	147.4	157.4	51.6
5754	11086.0	304.8	158.2	146.6	48.1

Table 7.3: Results of 1,000 iterations, 3 hidden nodes

Actual Frequency (Hz)	Detected Frequency (Hz)	Actual Depth (mm)	Detected Apparent Depth(mm)	Error (mm)	Error(%)
17261	17069.0	101.6	102.7	-1.1	-1.1
17261	17817.9	101.6	98.4	3.2	3.1
17261	17451.4	101.6	100.5	1.1	1.1
9864	11621.2	177.8	150.9	26.9	15.1
17261	17230.4	101.6	101.8	-0.2	-0.2
17261	16073.0	101.6	109.1	-7.5	-7.4
13809	14395.2	127.0	121.8	5.2	4.1
11508	14887.5	152.4	117.8	34.6	22.7
17261	16710.1	101.6	105.0	-3.3	-3.3
17261	13665.8	101.6	128.3	-26.7	-26.3
17261	17113.5	101.6	102.5	-0.9	-0.9
17261	9719.1	101.6	180.4	-78.8	-77.6
5754	6974.7	304.8	251.4	53.3	17.5
5754	8621.5	304.8	203.4	101.3	33.3
5754	11269.6	304.8	155.6	149.2	48.9

Table 7.4: Results of 100,000 iterations, 3 hidden nodes

To find the optimum number of investigation, it was noted that after 50,000 the result did not change considerably indicated in table 7.4. Indeed, table 7.2 and 7.3 shows the improvement from 1,000 iteration to 50,000 to be about 70%.

Actual Frequency (Hz)	Detected Frequency (Hz)	Actual Depth (mm)	Detected Apparent Depth(mm)	Error (mm)	Error(%)
17261	17400.0	101.6	100.8	0.8	0.8
17261	17994	101.6	97.5	4.1	4.1
17261	17810.4	101.6	98.5	3.1	3.1
9864	12039.8	177.8	145.7	32.1	18.1
17261	17640.1	101.6	99.4	2.18	2.2
17261	16871	101.6	104.0	-2.3	-2.3
13809	14220.3	127.0	123.3	3.7	2.9
11508	15743.6	152.4	111.4	41.0	26.9
17261	17005.0	101.6	103.1	-1.5	-1.5
17261	13838.9	101.6	126.7	-25.1	-24.7
17261	17748.2	101.6	98.8	2.8	2.7
17261	10667.1	101.6	164.4	-62.8	-61.8
5754	8286.1	304.8	211.6	93.1	30.6
5754	9416.4	304.8	186.2	118.5	38.9
5754	10334.2	304.8	169.7	135.1	44.3

Table 7.5: Results of 8,000 iterations, 4 hidden nodes

Actual Frequency (Hz)	Detected Frequency (Hz)	Actual Depth (mm)	Detected Apparent Depth(mm)	Error (mm)	Error(%)
17261	17050.4	101.6	102.9	-1.3	-1.2
17261	17682.9	101.6	99.2	2.4	2.4
17261	17526.2	101.6	100.1	1.5	1.5
9864	11216.6	177.8	156.4	21.4	12.1
17261	17488.8	101.6	100.3	1.3	1.3
17261	16409.9	101.6	106.9	-5.3	-5.2
13809	14416.9	127.0	121.6	5.4	4.2
11508	14813.6	152.4	118.4	34.0	22.3
17261	17050.7	101.6	102.9	-1.2	-1.2
17261	13615.1	101.6	128.8	-27.2	-26.8
17261	17375.6	101.6	100.9	0.7	0.7
17261	10259.7	101.6	170.9	-69.3	-68.2
5754	8274.9	304.8	211.9	92.9	30.5
5754	9362.6	304.8	187.3	117.5	38.5
5754	11160.8	304.8	157.1	147.7	48.4

Figure 7.6: Results 8,000 iterations, 10 hidden nodes

The results of table 7.5 and table 7.6 show a overall improvement of about 60% from increasing the nodes from 4 to 10, however, observing each result from table 7.5 and 7.6, it is noted that there is slight increase which is not substantial. Indeed, it was decided at this stage to consider other ways to have a significant effect on the optimisation of impact echo data. Also, the approach taken to deliver these results were in doubt. The next section describes the improvement on the impact echo data.

7.10.2. Method Increasing the number of data sets

It has been stated that due to their large number of parameters, neural networks are very sensitive to overfitting when compared to statistical methods. For example, if a three layer network had 25 inputs and 10 hidden nodes, there will be 260 free parameters, this is calculated by using (Masters, 1993):

$$\text{Weights} = (n + 1) * m \qquad \text{Equation 7.2}$$

Where n is the number of inputs and m is the number of hidden nodes

Previously, in neural network testing, 256 inputs have been used. Current impact data has now been reduced to 75 inputs, the minimum used so far, yet it is still a large amount of input neurons according to Masters (1993). If there are only two input inputs, about six training samples will mostly represent every possible error pattern. With the 75 input patterns, the chance of some pattern being repeated many times in the training set is much higher. The only way to prevent the network from learning unique characteristics of the training set, to the detriment of learning universal characteristics, is to provide it with plenty of examples, so that it cannot possibly learn all of their idiosyncrasies. As a guideline, it has been suggested:

- Compute the number of weights in the network using equation 7.2.
- Double it to get the minimum number of training samples required.
- Double it again to get a better sample size

It has been worked out that for this particular application, using 75 inputs and then using equation 7.2 gives 1824 free parameters. If this is doubled, then to obtain a minimum number of training samples is about 3600.

It was then necessary to increase the data of 225, to accomplish this task, and to verify this theory. However, it was impossible to repeat the experiment since it was conducted in the U.S.A. The best option was to add noise to create a new set. The following sections describe the procedures taken to produce the extra data and other processes to prepare data for the neural network analysis.

7.10.2.1. Data preparation

The preparation to increase the data for the above section is similar to the method described in section 7.10.1. Again, all the 225 data was used, in the following order:

- Editing the raw data
- Adding noise
- FFT processing
- Reduction of data (Interpolation)
- Filtering
- Preparation for neural network analysis

Once this was completed for the first set of 225 data, the above method was repeated with a different set of noise added to the original data. The whole technique was repeated until data set of 3600 were created. These procedures are discussed further in the following section.

7.10.2.1.1. Formatting raw data

The raw data collected from the impact echo came in one column which consisted of a set of 15, each set having 1024 data points, which represented one impact result (also discussed in chapter 3). This had to be separated into separate files e.g. 15 files, which means that for one test point, there was 15 different transients waveform. This had been completed for other neural network analysis including data correlation testing.

7.10.2.1.2. Data created by adding noise

A requirement in creating more data, by adding noise, was that the noise should not alter the original data significantly, whilst providing the necessary additional data. In order to accomplish this, a variety of standard deviation values of the noise were experimented with. The values tested were 0.5, 0.03, 0.025, 0.02, 0.015, 0.01, and 0.00.

The standard deviation controls the noise addition so for each level of standard deviation, noise was added to the original data and amplitude spectrum performed. The FFT processing made it easier to see any difference as can be seen from figures 7.6 to 7.12 and so it was used to compare with graphs having no noise added. The conclusion was that, the standard deviation of 0.5 was too much as it changed the original data completely.

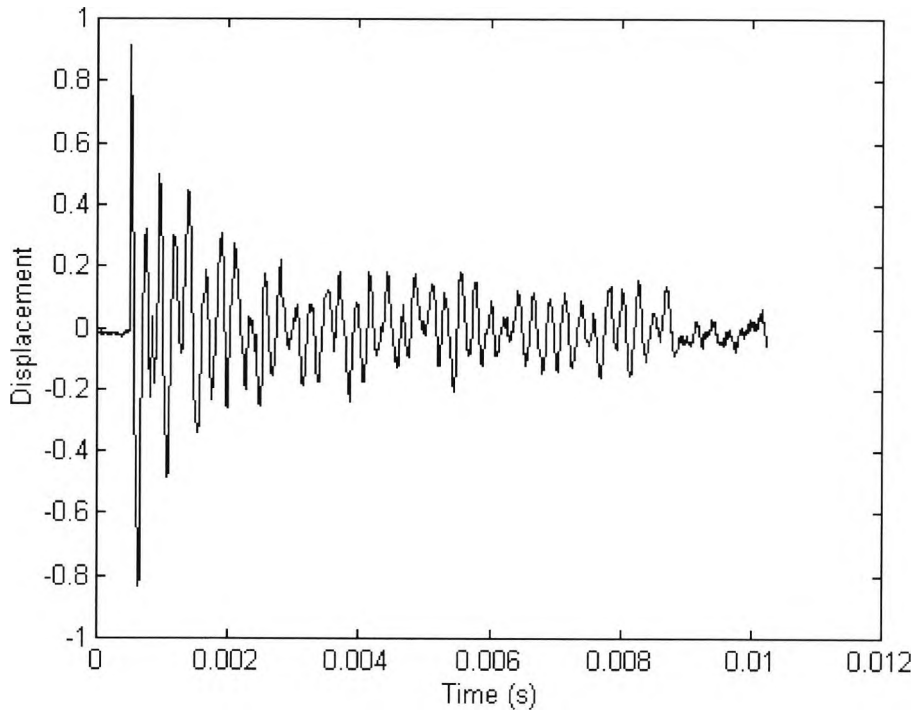


Figure 7.6: Original data without noise

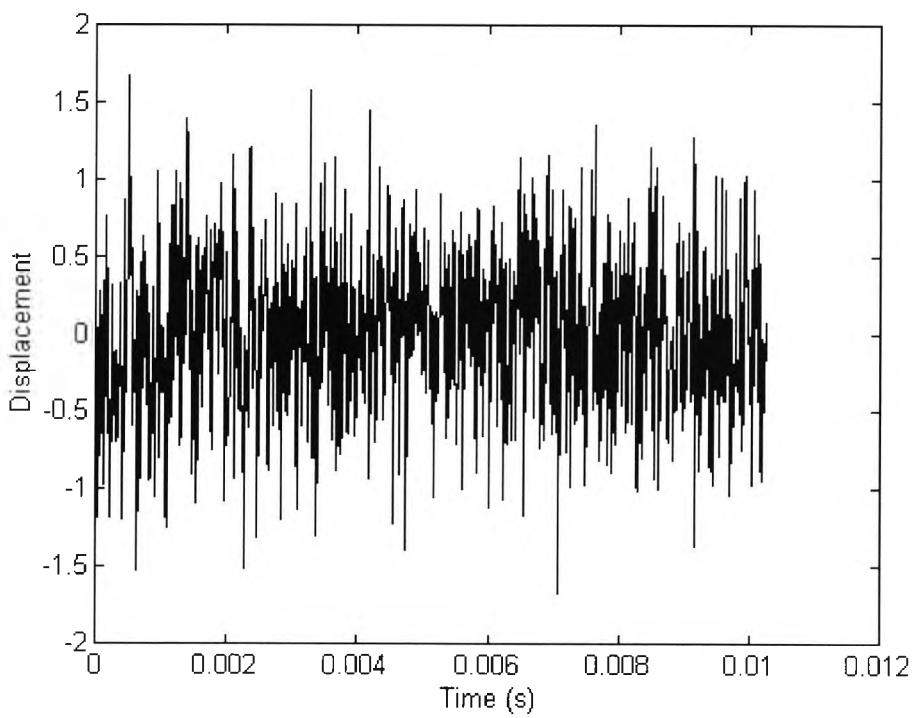


Figure 7.7: Addition of noise, standard deviation = 0.5

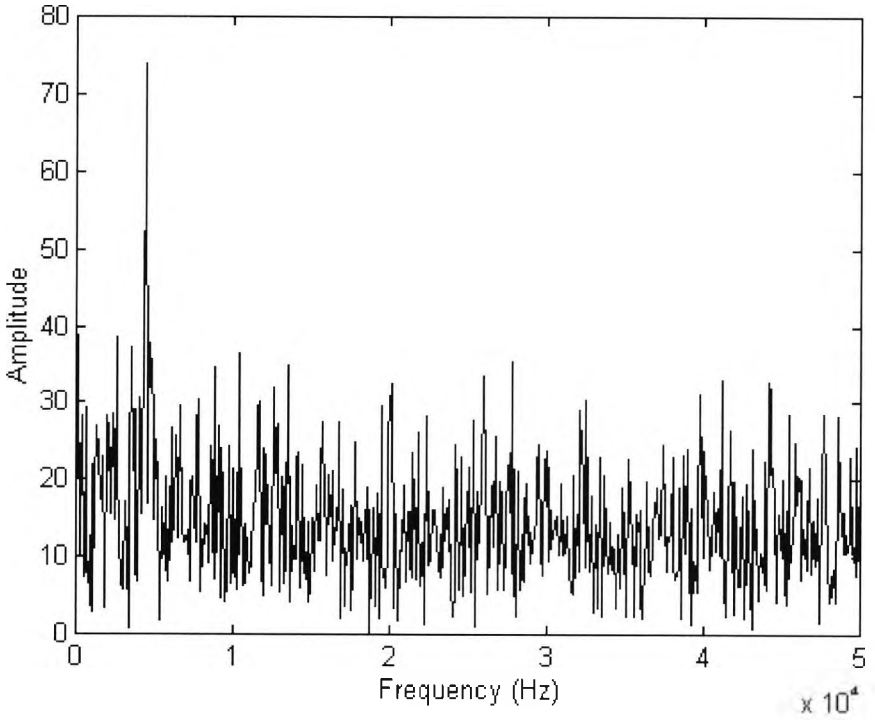


Figure 7.8: Amplitude spectrum, standard deviation = 0.5

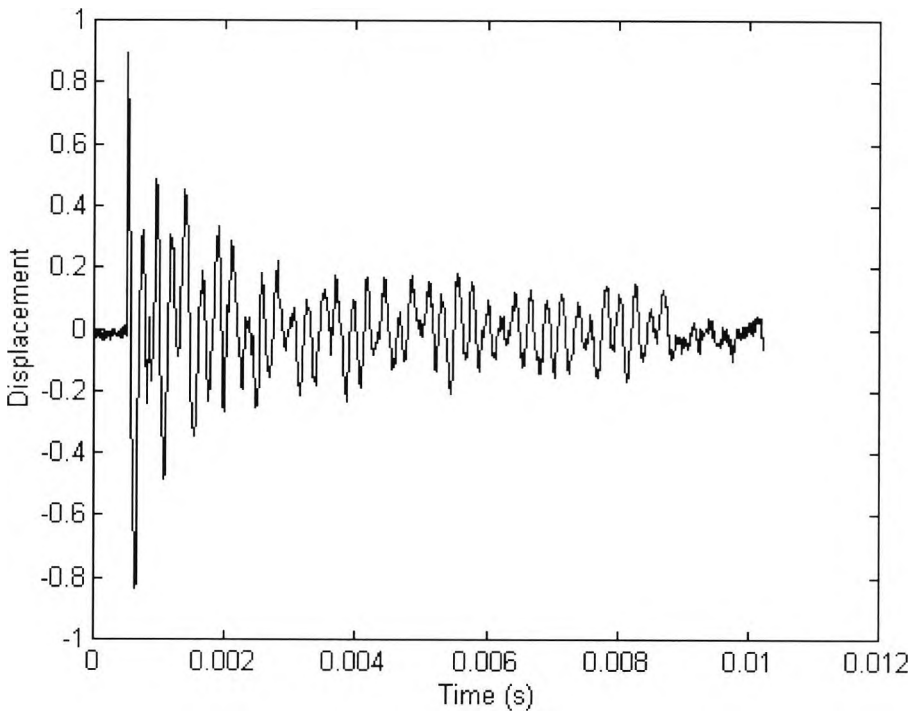


Figure 7.9: Addition of noise, standard deviation = 0.01

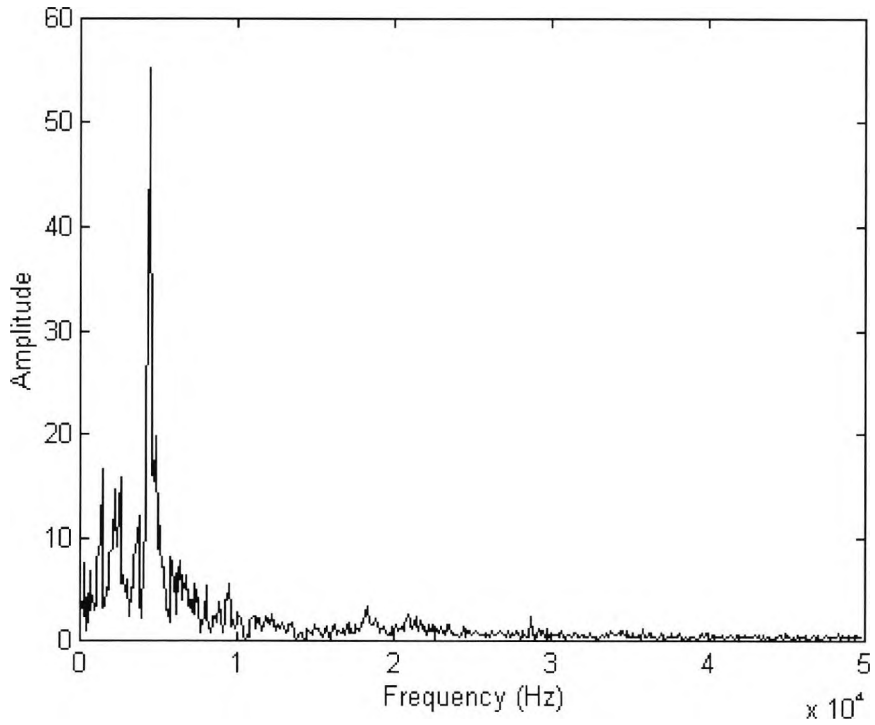


Figure 7.10: Amplitude spectrum, standard deviation = 0.01

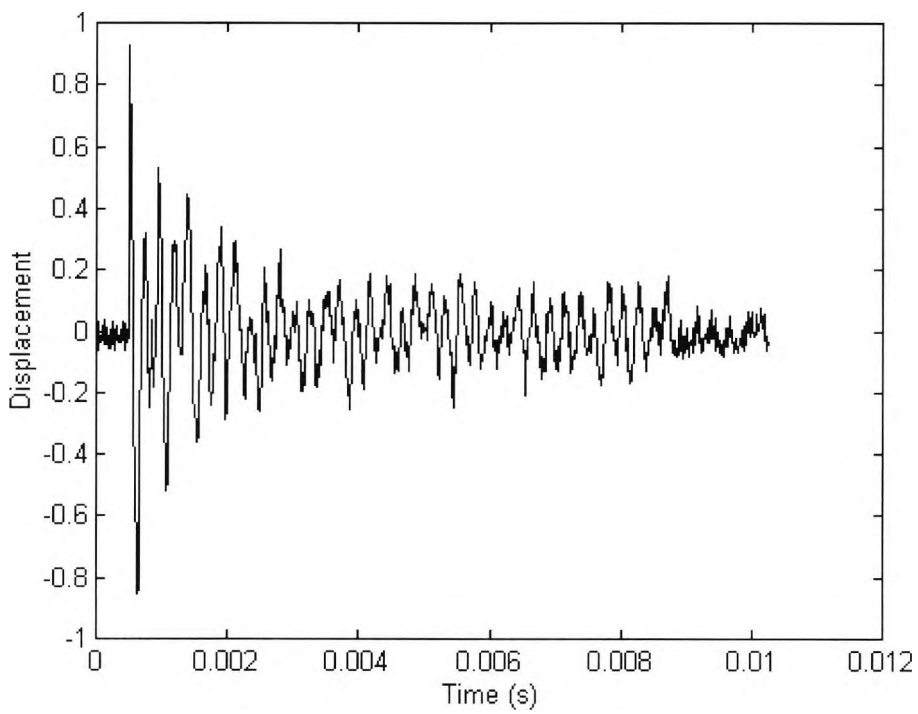


Figure 7.11:Addition of noise, standard deviation = 0.025

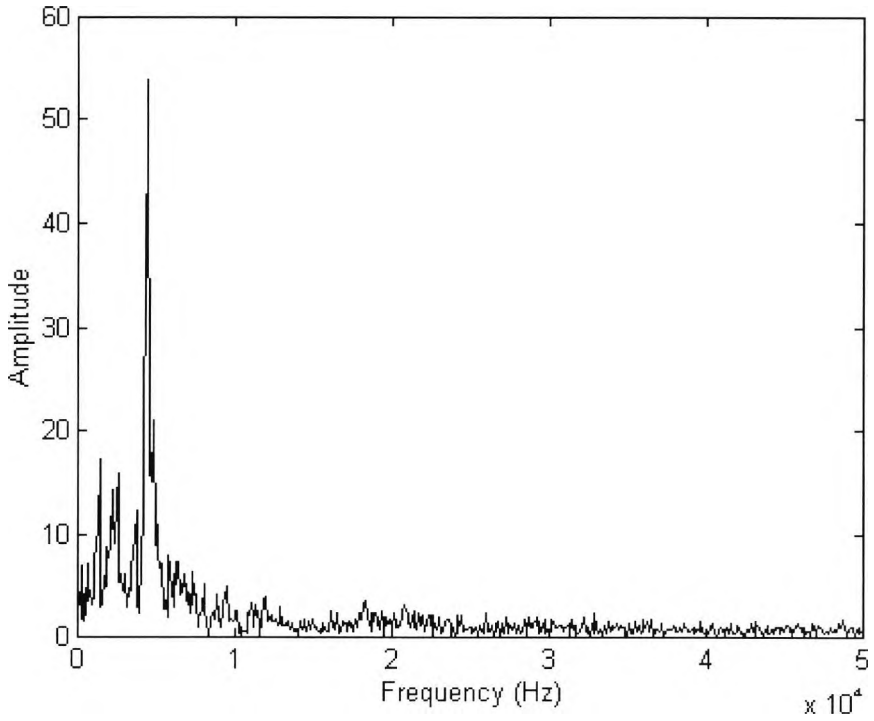


Figure 7.12: Amplitude spectrum, standard deviation = 0.025

However, the standard deviation of 0.01 was too small and made no difference to real data. It was found that the standard deviation of 0.02 was ideal. However, as the data set was increasing, when the number of data set was 795, the neural network result showed no substantial performance, so it was decided to increase the standard deviation slightly, hence the standard deviation of 0.025 was used.

Labview was used to generate the noise, the program and examples are included in Appendix F. The program which produces the noise had to be run 225 times, for a new set of 225 data to be obtained. The calculation for the number of data sets for neural network analysis shown in section 7.10.2 indicates that a data set of about 3600 is required. This means the Labview program would have to be repeated 16 times (3600/225) for each set of 225 data.

7.10.2.1.3. FFT Processing

Once the noise was added, each set of 225 data needed FFT processing. This had two advantages; one was that it was much clearer and easier to understand the data in frequency terms rather than in time domain, the other was that it reduced the data from 1024 points to 512 (FFT theory). This was a help because the reduction of data is the primary objective in this type of neural network analysis. This is considered further in the next section. The FFT method using Matlab is shown in the Appendix C.

7.10.2.1.4. Interpolating data

The 512 data need further reduction. The function of the interpolation was that, whilst it minimised the data, it also had the additional benefits of smoothing the graph, so more dominant peaks showed, and smaller peaks disappeared. But at this stage, it was not known how many points to settle on. This needed investigation. A range of frequency increment used is given in table 7.7.

Frequency Increment (Hz)	The Equivalent Points
97.65625	512
195.3125	256
292.96875	170
390.625	128
488.28125	102
585.9375	85
250	200

Table 7.7: Illustration of the frequency increment

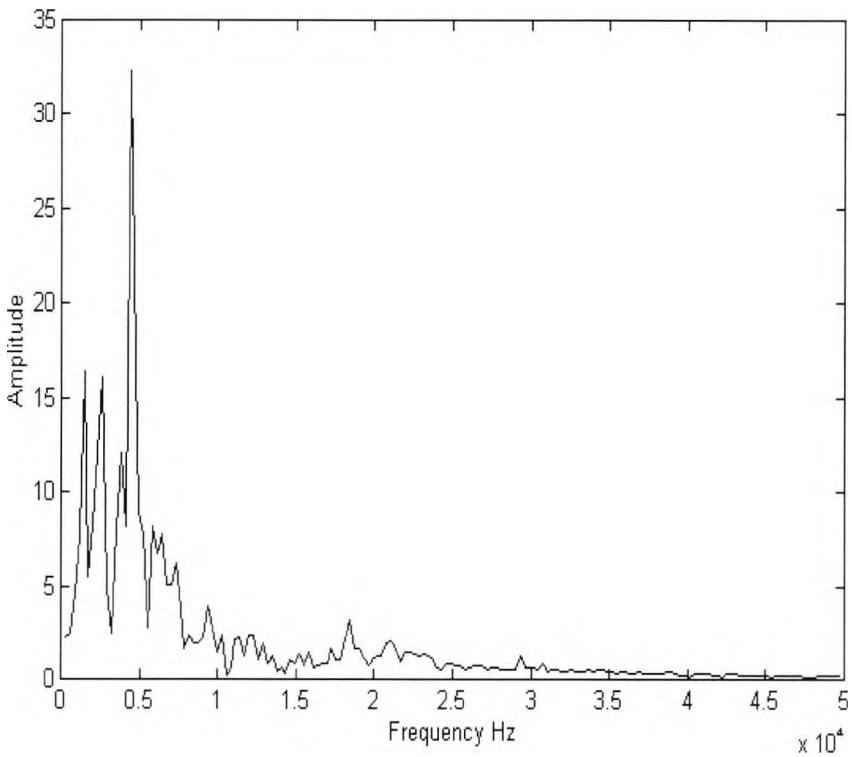


Figure 7.13: Amplitude spectrum with 170 points

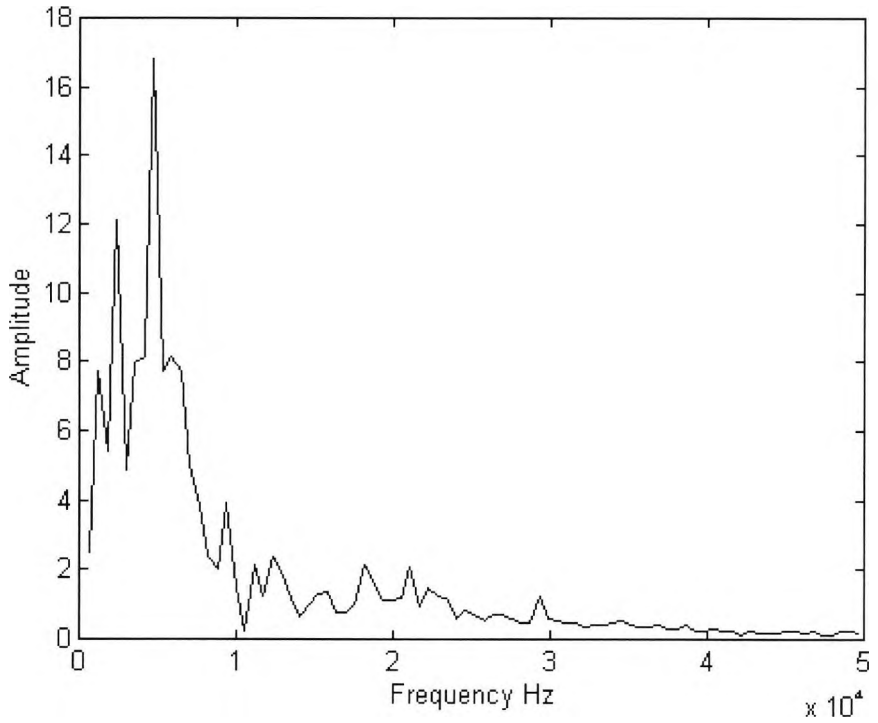


Figure 7.14: Amplitude spectrum with 85 points

The frequency increment was calculated using the following:

$$\text{Record Length} = N \times S_p \qquad \text{Equation 7.3}$$

Where N is the number of sampling points

S_p is the Sampling frequency

So, since the number of sampling points was 1024 points (see appendix C for further FFT theory and application), and the sampling period was 10 μs (period = 1/frequency = 1/ S_p), then:

$$\text{Record Length} = 1024 \times 10 \times 10^{-6} = 0.01024 \text{ s}$$

The frequency increment for this record length is:

$$\text{Frequency increment} = 1/\ 0.01024 = 97.65625 \text{ Hz}$$

The frequency increment of 97.65625Hz was then multiplied by two, giving 195.3125 Hz. This new figure was used to calculate the number of sampling points, using equation 7.3. This delivered 256 points. N.B, the record length used was 50 kHz because with FFT theory the other half of the signal can be ignored. Also it is important

to point out that the frequency increment is used to interpolate the equivalent amplitude value. This procedure was repeated by multiples of 3, 4 and 5.

Using Matlab, the interpolation was completed. The graphs with this variety of frequency increments were plotted and studied to see which gave better results. The aim was to find a graph, which gave the most useful information about a particular defect or non-defect, and also used the minimum number of points. As shown in figures 7.13 to 7.14, it was found that the frequency increment of 292.96875 Hz (170 points) met this requirement. However, the frequency increment of 250 Hz was used to verify if it would give better results than this. It did not, so the 292.96875 Hz increment were used. The Matlab program for interpolation and graphs of these increments can be found in Appendix F.

7.10.2.1.5. Data filtering

Once the impact echo data was reduced to 170 points, there were still parts of the graph, which could be discarded. For example the depth of the wall specimen is of 12 inches (304.8 mm) gives an equivalent frequency of 5.74 kHz. This means any frequency below this can be discarded, because it has no use and also, it reduces the data, which is vital for neural network, making it easier to train, the ultimate goal.

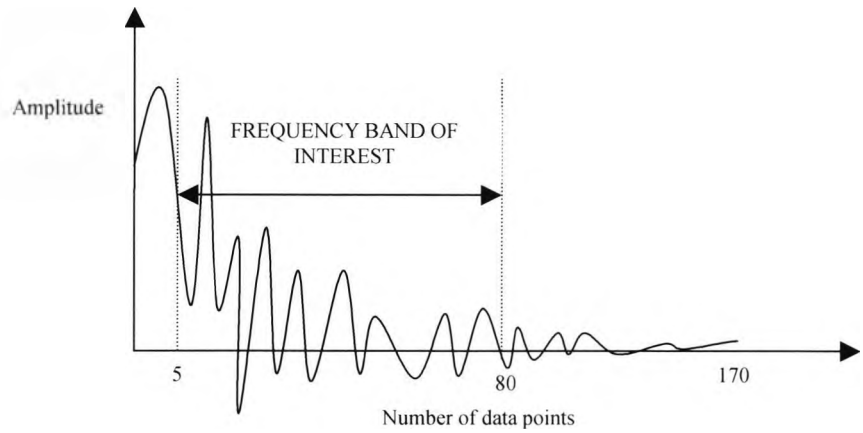


Figure 7.15: Filtering unwanted data

Also, higher frequency was of no interest because these correspond to very shallow defects, which were not present. The maximum void frequency was 17.261Khz (theoretical). In practice, the graphs gave about 19kHz, so anything above 20Khz could be eliminated. This observation is indicated in Figure 7.15. However, the x-axis shows the number of data not frequency, but idea is the same.

When the data was filtered, it was appended to a file. Again, this would be repeated 225 times, so each file would have 225 data sets. Now, the file needed editing for the neural network format. This procedure was completed by using Labview. The program is given in Appendix F.

7.10.2.1.6. Accumulating the data

Each time a set of 225 was created for neural network use it needed to be in a format that the network can use to train and test. Proff 11/Plus was the neural network software used in this research. Method and examples of this technique can be found in Appendix A.

The 225 data examples were trained and tested, and results noted. Then the next set of 225 data were added with noise, FFT processed, interpolated, filtered, edited, and added to the original file for neural network training. This was repeated until the data set increased to about 3833. This data is given in Appendix D. Table 7.8 shows all the data sets used in the neural network training with the number of hidden nodes for each data set. As shown, the minimum data set used was 194 training data, 12 test data, the maximum used is 3833 training data, with 163 test data. It is important to note however, that due to some error not all of the 225 data records were used in each processing instead 223 data records were used in most cases.

Data Sets	Test sets	Hidden nodes
194	12	14
396	34	20
577	77	24
795	81	28
1015	86	32
1232	91	35
1453	94	38
1671	100	41
1892	104	43
2111	110	46
2330	116	48
2548	121	50
2765	127	53
2983	134	55
3201	141	57
3420	147	58
3615	156	60
3833	163	62

Table 7.8: Calculation of hidden nodes for each set of training and testing file

7.11. Further improvements

Originally, this application was considered to be a function approximation problem shown in figure 7.16, that is, a graph represented a pattern and a function had to be fitted in order to find a solution. The back propagation was thought to be the best option. The results for the increase of the number data sets shows a slight improvement. Figures 7. 19 and 7.20 show the original data with 194 data sets which has no noise added and the addition of noise with 2765 data sets. In these figures, 'actual' represents the desired results set by the author, and the 'neural network' is the results the network has managed to achieve.

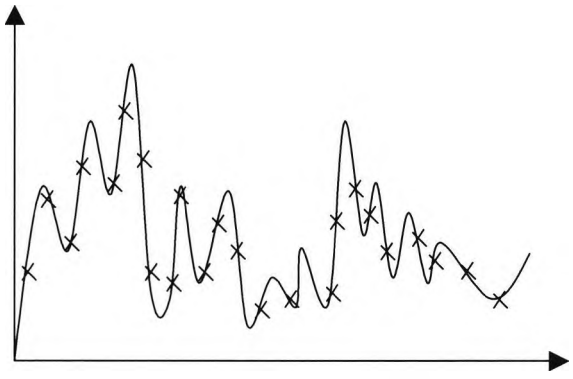


Figure 7.16: Function approximation problem

Learning Parameter (hidden) (α)	Learning Parameter (output) (α)	Momentum (β)	Hidden Nodes	Transfer function (input)	Transfer function (hidden)	Transfer function (output)
0.3	0.15	0.4	61	Linear	Sigmoid	Sigmoid
0.3	0.15	0.5	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.6	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.7	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.8	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.9	61	Linear	Sigmoid	Sigmoid
0.25	0.15	1.0	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.85	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.86	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.87	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.88	61	Linear	Sigmoid	Sigmoid
0.25	0.15	0.89	61	Linear	Sigmoid	Sigmoid
0.25	0.05	0.89	61	Linear	Sigmoid	Sigmoid
0.25	0.1	0.89	61	Linear	Sigmoid	Sigmoid
0.25	0.11	0.89	61	Linear	Sigmoid	Sigmoid
0.25	0.2	0.89	61	Linear	Sigmoid	Sigmoid
0.3	0.15	0.89	61	Linear	Sigmoid	Sigmoid
0.35	0.15	0.89	61	Linear	Sigmoid	Sigmoid
0.4	0.15	0.89	61	Linear	Sigmoid	Sigmoid
0.5	0.15	0.89	61	Linear	Sigmoid	Sigmoid

Learning Parameter (hidden) (α)	Learning Parameter (output) (α)	Momentum (β)	Hidden Nodes	Transfer function (input)	Transfer function (hidden)	Transfer function (output)
0.4	0.15	0.89	62	Linear	Sigmoid	Sigmoid
0.4	0.15	0.89	60	Linear	Sigmoid	Sigmoid
0.4	0.15	0.89	59	Linear	Sigmoid	Sigmoid
0.4	0.15	0.89	63	Linear	Sigmoid	Sigmoid
0.4	0.15	0.89	61	Sigmoid	Sigmoid	Linear
0.4	0.15	0.89	61	Linear	Sigmoid	TanH
0.4	0.15	0.89	61	Linear	Sigmoid	Linear
0.25	0.15	0.89	61	Linear	Sigmoid	Linear
0.25	0.15	0.89	62	Linear	Sigmoid	Linear
0.25	0.15	0.89	59	Linear	Sigmoid	Linear

Table 7.9: Cases of learning rate, momentum and hidden nodes used to find the optimum parameters

There was still room for improvement. So far, the learning parameter and momentum term were used as the default values provided by the neural network software. Adjusting these values would make some improvement, so the next objective is to find the optimum pair that would give the best result. Indeed, the parameters investigated were:

- Momentum β
- Learning rate α for hidden and output layer
- Hidden nodes
- Transfer function

7.11.1. Re-organising the data

The first stage in improving the results was recognising that the 3833 data sets contained some bad data which could be misleading for the network. These data sets were eliminated. This procedure was repeated with the testing file as well. The learning and testing file was reduced to 3698 and 160 data sets. A selection of data from this new learning file was chosen to create a smaller file, because it was difficult to see how well the network was trained with the 3698 data sets file because it was too large. Usually, about 2 to 3 data patterns was taken from each of the data sets shown on table 7.8.

7.11.2. Testing neural network on trained data

The aim was to test the network on the same training data it had learned from, because this was a very good starting point to see how well the network is trained before testing

the network on data that it has not seen before. The results are shown on figure 7.17. It can be seen that there are large discrepancies. This suggests that the network has not learnt the data well, ideally there should be negligible differences between the actual and neural network data. To minimise this difference, as discussed in section 7.11, the four parameters need to be adjusted to find the optimum results.

7.11.3. Investigation of optimum parameters

The next step was to find the optimum pair of learning rate, α and momentum, β, a selection of these values were taken. The data for different values of these two parameters are included in the Appendix D. Figure 7.17 and 7.18 show the default and the optimum parameters for α an β.

Once the optimum α and β was found, the next task was to find the optimum number of hidden nodes. Indeed, it was decided to have a higher learning rate in the hidden layer than the output layer, because having a larger learning coefficient at the hidden layer than for the output layer allows the hidden layer to form feature detectors during the early stages of training, so that these feature detectors can then be combined to form more complex detectors at the output layer (Using Neural Networks, 1993). Moreover, having different learning parameters for the subsequent layers reduces the learning time. The range of hidden nodes was investigated; the data is enclosed in the Appendix D.

By altering the transfer function for each layer in the network could further improve the results, so this was examined. The transfer function for the output layer was changed from sigmoid to Linear and TanH and network was trained again to observe any improvements. The Linear transfer function should slight improvement and α and β were changed to 0.25 and 0.15 to observe any improvement, also, the hidden nodes were altered to 62 and 59. The data for all these are included in the Appendix D.

When the optimum parameters were found which is discussed in the next section, the network was finally trained again and tested on unknown data. The results shown in figures 7.21 and 7.22 indicate about 10% improvement when compared to the original data with 3833 data sets.

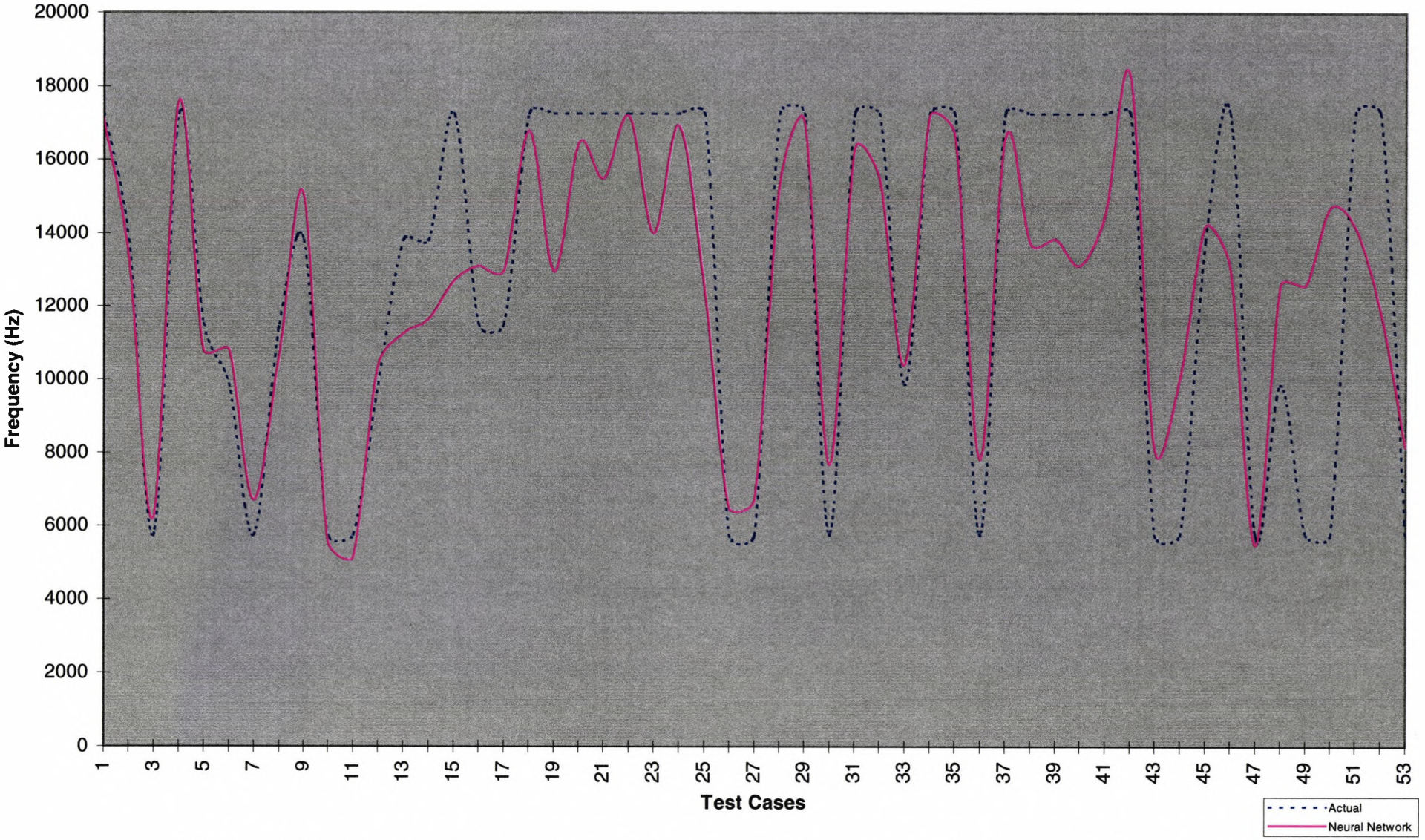


Figure 7.17: Momentum = 0.4, learning rate (hidden) = 0.3, learning rate (output) = 0.15 (all default values)

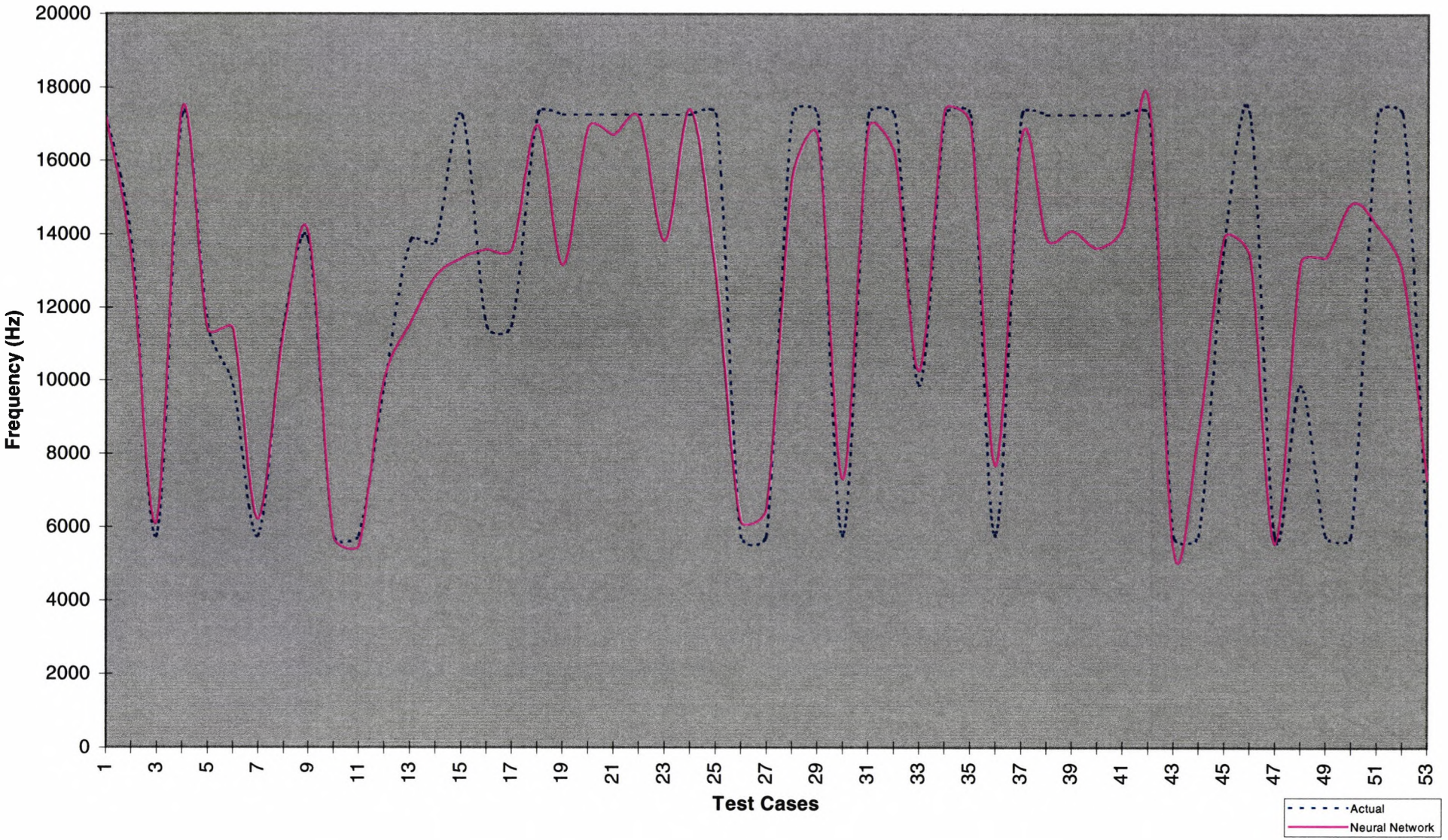


Figure 7.18: Momentum = 0.89, learning rate (hidden) = 0.4, learning rate (output) = 0.15

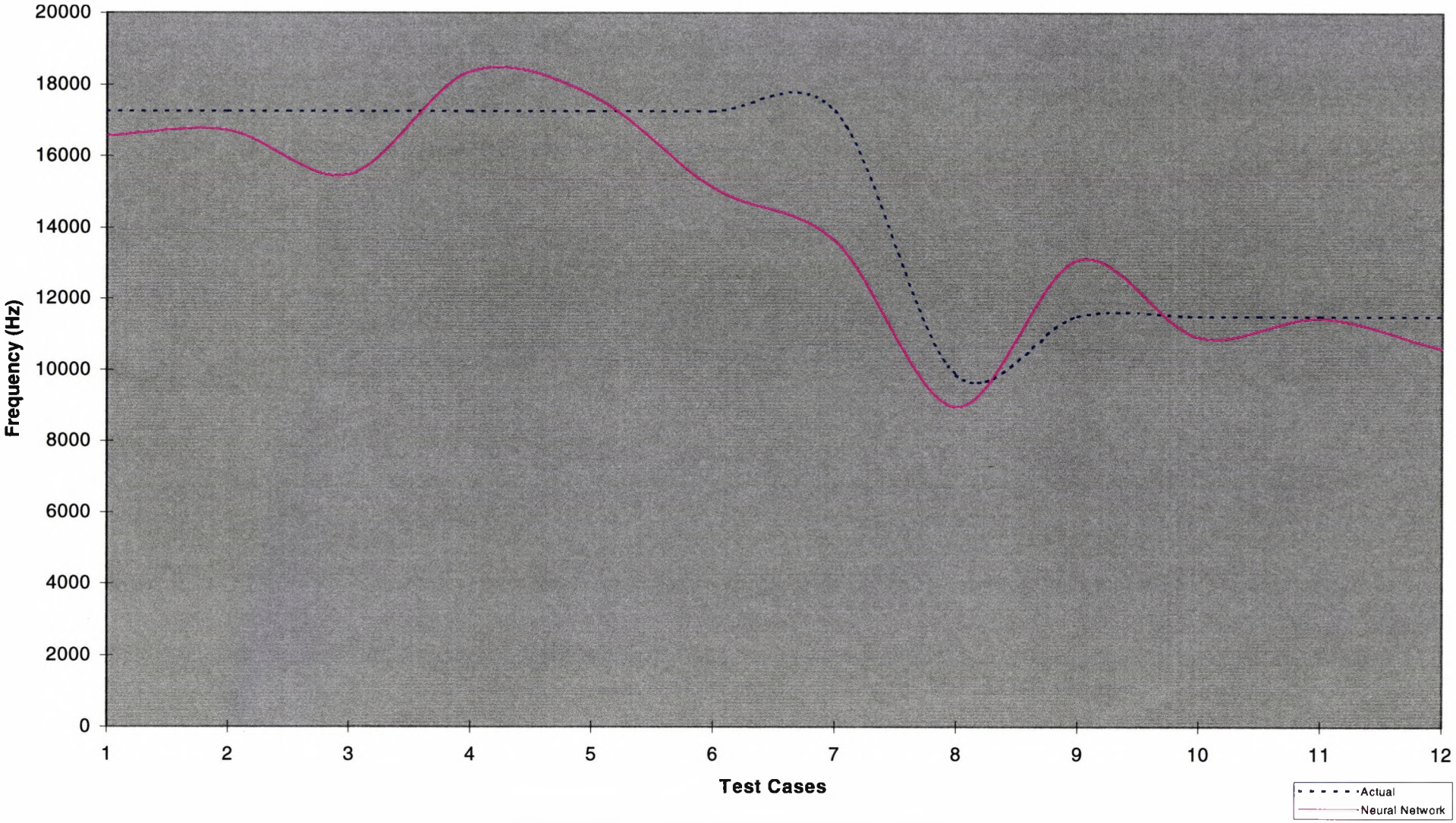


Figure 7.19: Original data without any noise

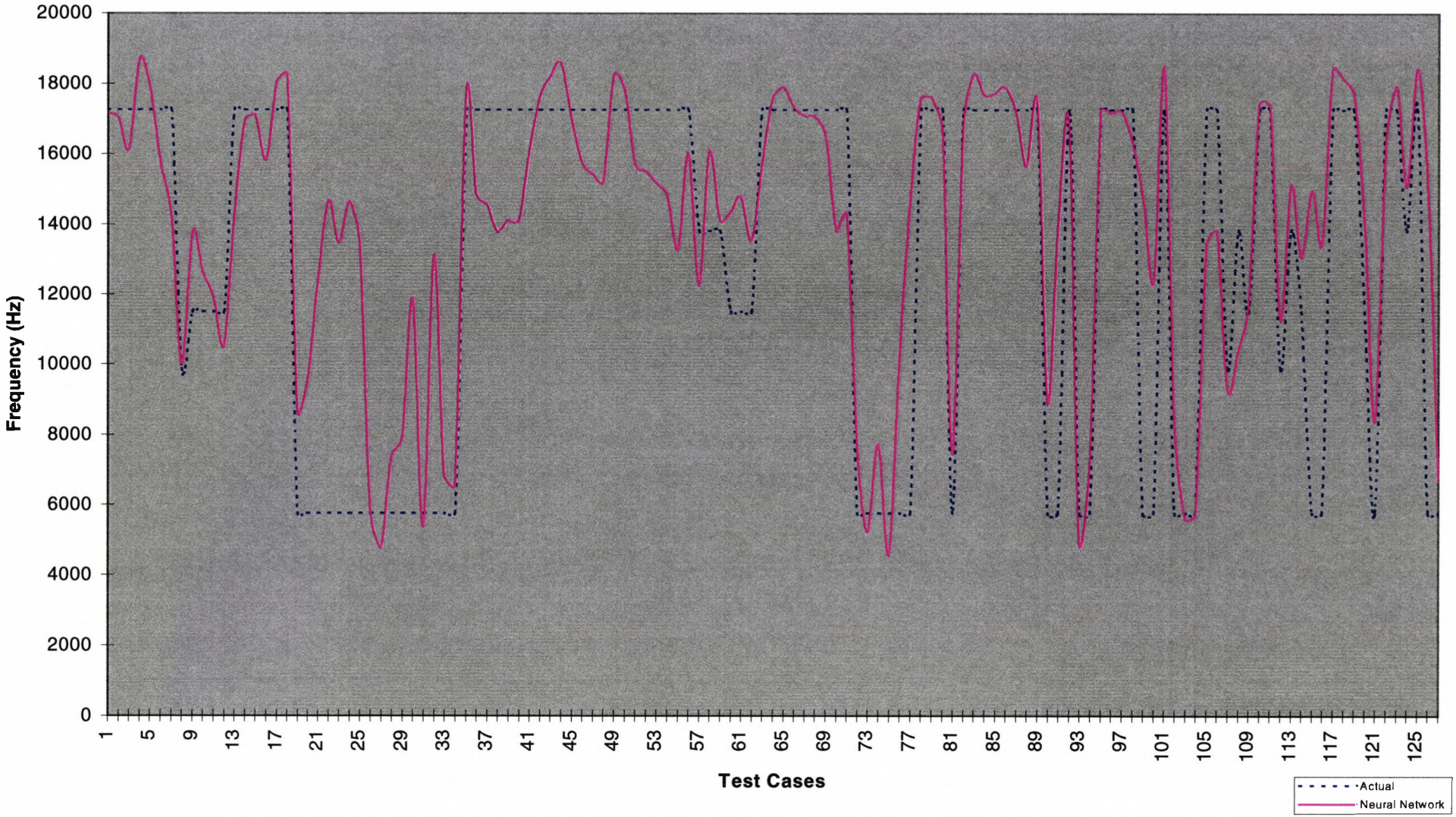


Figure 7.20: Data with addition of noise (2765 training data, 127 test data and 53 hidden nodes used)

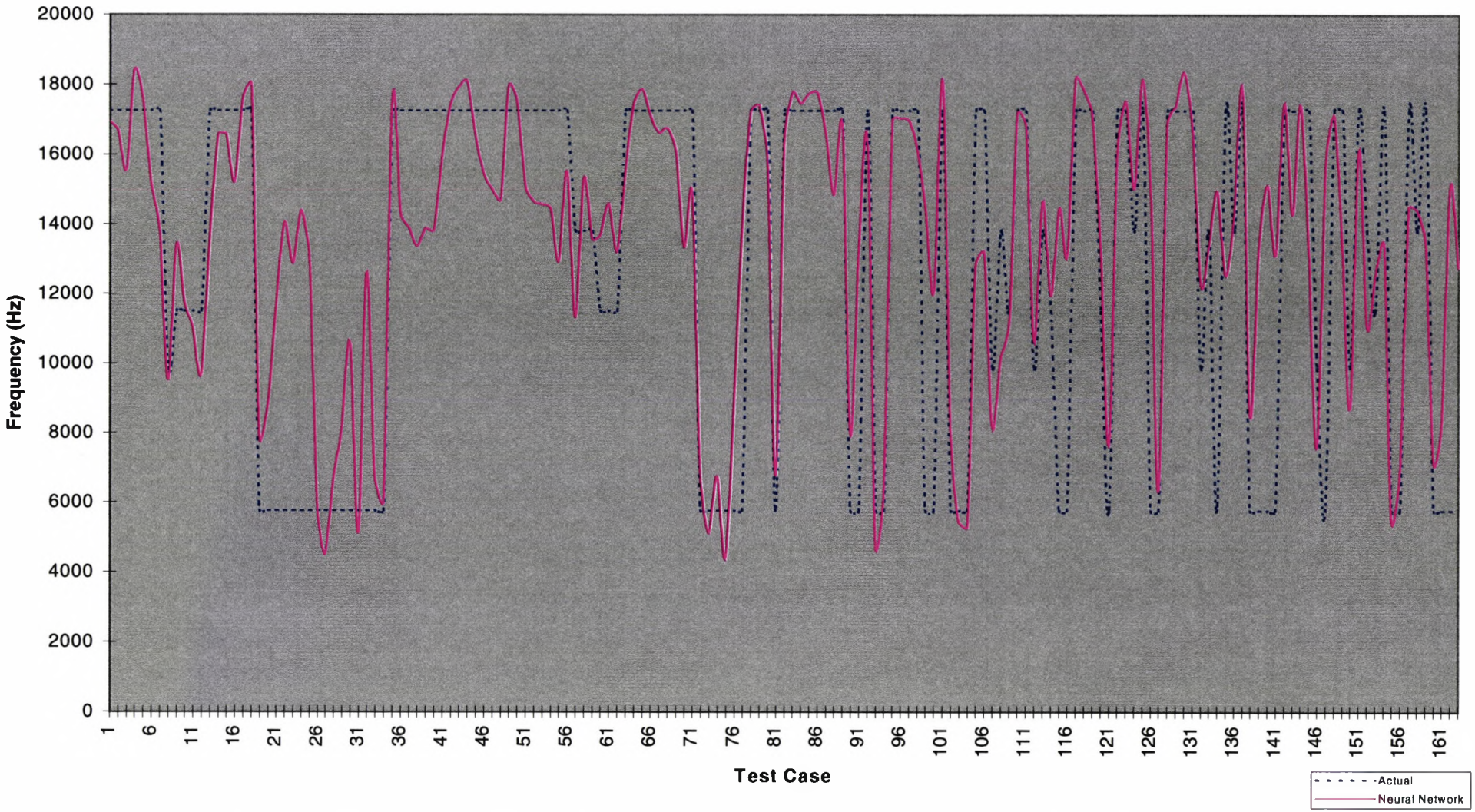


Figure 7.21: Results with default parameters (3833 training data, 163 test data, 62 hidden nodes used)

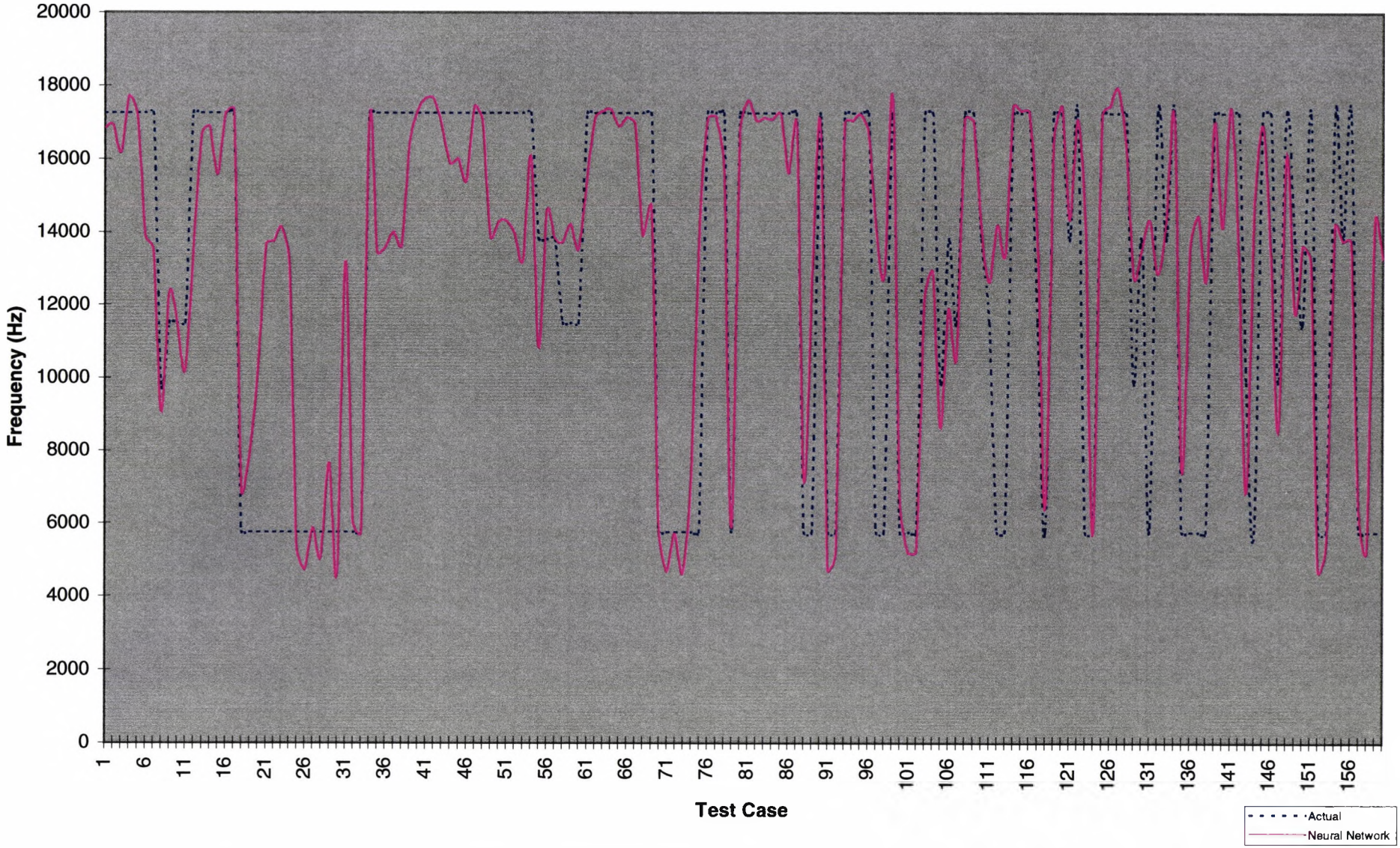


Figure 7.22: Results with optimum parameters (3698 training data, 160 test data, 61 hidden nodes used)

7.12. Neural network results

During the investigation of trying to find the optimum parameters, the neural network results started to show improvement when the momentum value was between 0.85 to 0.9. An increment of 0.01 was added to the momentum value of 0.85, the results were observed. This was repeated until momentum was 1. As mentioned in section 7.5, the momentum factor should be in the order of 0 to 1. When the network was trained with a momentum value of 1, the results were the worst results the author had seen in the neural network analysis. The error was very large. Hence, momentum value of 0.89 was selected. Also, Section 7.6 stated to use a range of learning rate values of 0.05 to 0.25. The maximum value of 0.25 did deliver good results, but was found to do better at 0.4.

This proved that the apparent optimum results is achieved with:

- $\alpha = 0.4$ (hidden)
- $\alpha = 0.15$ (output)
- $\beta = 0.89$
- Hidden nodes = 61
- Transfer function (input) = Linear
- Transfer function (hidden) = Sigmoid
- Transfer function (output) = Sigmoid
- Iteration = 50,000

Table 7.10 shows results of neural network with the predicated frequency and impact echo. The neural network was produced from data relating to each location, which was illustrated in chapter 3. The data for each location was put together and the average was computed. As can be seen from table 7.10, the neural network results for defects are generally 80% accurate. However, the results for sound areas are not so encouraging, as discussed in chapter 4, the sound areas tested were surrounded by other defects which consequently would weaken the signal for a solid. This effect is obvious in this case.

However, the results could still be improved, namely, this application may be purely a classification problem, although it was assumed to be a mapping, classification and function approximation problem. It seems that certain areas of an amplitude spectrum

can be 'classified' as defect or non-defect, in which case a LVQ or Radial Basis Network is most suited. Also, when the network was tested on the same data, as shown in figure 7.18, there is still a considerable amount of difference, realistically, the actual and neural network response should be an exact fit. The impact echo application does not have precise data, as discussed the data varies from one record to another record. The 15 records for one test point had some data which were fairly lower than the predicted frequency. For example, for a predicted frequency of 17.261 kHz, one record may suggest about 16 kHz, another may indicated about 10 kHz.

Code Name	f_P Predicted Frequency (kHz)	f_P Impact Echo (kHz)	Neural Network
Wav1e	17.261	18.23	17.11
Wav2e	17.261	16.68	14.55
Wav3e	17.261	8.92	15.81
Wav4e	17.261	16.87	15.78
Wav1w	9.864	9.99	9.20
Wav2w	17.261	15.22	16.64
Wav3w	13.809	12.60	13.30
Wav4w	11.508	12.31	11.47
Wacl1	5.754	3.68	11.45
Wacl2	5.754	3.68	7.24
Wdv5e	17.261	9.21	17.00
Wdv6e	17.261	9.21	17.10
Wdv7e	17.261	14.45	15.64
Wdv8e	17.261	16.29	14.99
Wdcle	5.754	5.91	10.81

Table 7.10: Comparison of results

7.13. Summary

The important factors concerning a neural network was discussed which are, validation, momentum, learning rate, weights, iteration, and hidden nodes. The method for preparing data for training a neural network was presented, with an explanation of how the impact echo was modelled using the neural network technique. It was shown how neural network data was improved by adjusting the parameters discussed earlier in the chapter. It can be concluded that to achieve the optimum results with the experiment conducted with the impact echo device, the parameters setting given in section 7.12 must be adopted.

Chapter eight

Finite element theory

8.1. Introduction

This chapter covers the theory of the finite element method which has been applied to the simulation of impact echo system in order to achieve more data sets.

The impact echo system is a typical practical, physical problem whose solution by conventional analytical methods is either too difficult or impossible. In order to evaluate the precise response of the body by the impact force, a 'closed form' solution of the equations which govern its deformation need to be determined. However, because practical problems tend to have complex geometrical configurations, it is exceedingly difficult to obtain such solutions. For the analyst, such problems require a numerical technique. The finite element has been adopted in this research as a method for obtaining a wide range of impact echo data sets.

8.2. Theory of finite elements

The finite element method is a general numerical technique for obtaining approximate solutions to the partial differential equations that arise from boundary value problems. The method approximates a structure in two distinct ways:

- The division of smaller parts is called finite elements. The process of dividing up a structure is called discretisation. Each element is usually a bar, beam, plate or solid, which has an equation of equilibrium that can be solved or approximated. Element have nodes that connect it to the adjacent elements, and these points on the structure represent the equilibrium of the entire structure. The nodes are also used to capture the global motion of the structure as it vibrates. A finite element mesh is the collection of nodes and finite elements.
- For each individual finite element, the equation of dynamic equilibrium is calculated and solved. The solutions of the element equations are approximated by a linear combination of low order polynomials. The solutions for each polynomial is made compatible with adjacent solution, which are the continuity conditions at nodes common to two elements. The solutions are accumulated, resulting in global mass and stiffness matrices, that describe the dynamics of the structure as a whole.

8.2.1. Formulation of the stiffness matrix

In this research, as previously discussed in chapter 1, three types of element were employed. The first was an eight node 2D element, the second was a contact element, (two different types of contact element are discussed further in chapters 9 and 10). The third was a twenty node 3D element. This section concentrates on the theory of the 2D element. It presents the formulation of the stiffness matrix. Indeed, in each finite element software, this is processed by using an appropriate algorithm.

The procedure begins by converting the displacements of the element into the form of nodal displacement using suitable polynomial equations. Then, strains are evaluated from these nodal displacements, which can be used to deduce stresses by the use of Hook's law. Finally, the unknown parameters in these polynomial equations are solved by applying the principle of minimum potential energy.

8.2.1.1. Nodal displacements

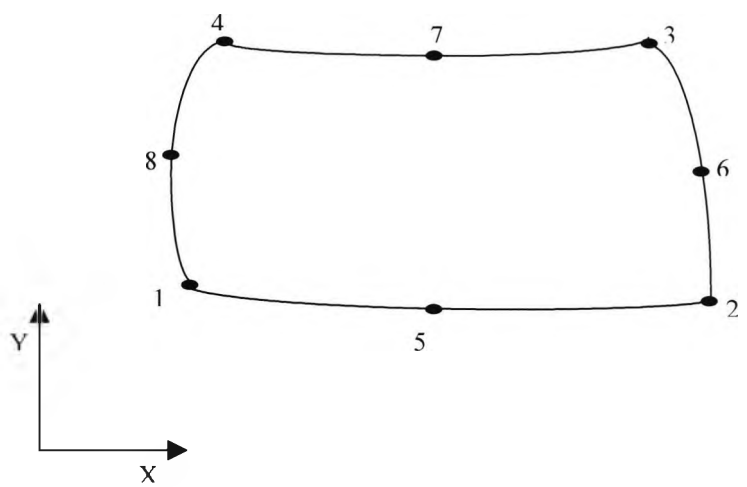


Figure 8.1: Eight-node rectangular element

The 2D element with eight nodes with two degrees of freedom at each node, is shown in figure 8.1. These are displacement in the x and y co-ordinate axis which can be referred to as displacements u and v (Zienkiewicz, 1971).

The element shown in figure 8.1 has a total of 8 nodes, so there are a total of 16 degrees of freedom (DOF) for each element. The displacement function at each node is given by:

$$U = \alpha_1 + \alpha_2 x + \alpha_3 y + \alpha_4 x^2 + \alpha_5 xy + \alpha_6 y^2 + \alpha_7 x^2 y + \alpha_8 xy^2 \qquad \text{Equation 8.1}$$

$$V = \alpha_1 + \alpha_2 x + \alpha_3 y + \alpha_4 x^2 + \alpha_5 xy + \alpha_6 y^2 + \alpha_7 x^2 y + \alpha_8 xy^2 \qquad \text{Equation 8.2}$$

where α_i i = 1 to 8 are constants to be evaluated.

By substituting in the appropriate x and y co-ordinates for each of the eight nodes, a set of eight simultaneous equations are created. These equations can then be solved to find the unknown coefficients. These can be substituted back into equation 8.1. giving:

$$\begin{aligned} U = \frac{1}{2ab} \; [& -(x-a)(y-b)(y+b/a\,x+b)/2b\,(u_1) \\ & (x+a)(y-b)(y-b/a\,x+b)/2b\,(u_2) \\ & (x+a)(y+b)(y+b/a\,x-b)/2b\,(u_3) \\ & -(x-a)(y+b)(y-b/a\,x-b)/2b\,(u_4) \\ & (x+a)(x-a)(y-b)/a\,(u_5) \\ & .(x+a)(y+b)(y-b)/b\,(u_6) \\ & -(x+a)(x-a)(y+b)/a\,(u_7) \\ & (x-a)(y+b)(y-b)/b\,(u_8)\,] \end{aligned} \qquad \text{Equation 8.3}$$

This can be expressed as:

$$U = [\; N_1 \; N_2 \; N_3 \; N_4 \; N_5 \; N_6 \; N_7 \; N_8 \;] \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \end{pmatrix} \qquad \text{Equation 8.4}$$

Where N_1 to N_8 are the shape functions. This process is repeated for v, resulting in:

$$U(x,y) = \sum_{i=1}^{n} N_i(x,y)\, u_i \qquad \text{Equation 8.5}$$

$$V(x,y) = \sum_{i=1}^{n} N_i(x,y)\, v_i \qquad \text{Equation 8.6}$$

If a local co-ordinate system (ξ ,η) is chosen for convenience as illustrated in figure 8.2.

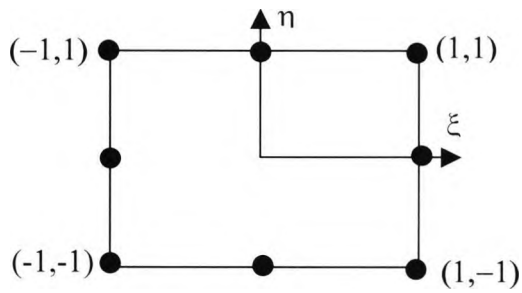


Figure 8.2: A local co-ordinate system for the 2D element

8.2.1.2. Equations of strain

There are three strain components in small deflection two dimensional stress analysis:

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{Bmatrix} = \begin{Bmatrix} \frac{\delta u}{\delta x} \\ \frac{\delta v}{\delta y} \\ \frac{\delta u}{\delta x} + \frac{\delta v}{\delta y} \end{Bmatrix}$$

Equation 8.7

Equation 8.8

Equation 8.9

In terms of nodal displacement and shape functions, the strains can be written in matrix form:

$$\begin{pmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{pmatrix} = \begin{pmatrix} \frac{\delta N}{\delta x} & 0 \\ 0 & \frac{\delta N}{\delta y} \\ \frac{\delta N}{\delta y} & \frac{\delta N}{\delta x} \end{pmatrix} \begin{pmatrix} U \\ V \end{pmatrix}$$

Equation 8.10

In summary form, equation 8.10 becomes:

$$\{\varepsilon\} = [B]\{\delta\}$$

Equation 8.11

To change to a local co-ordinate system, as illustrated in figure 8.2, and using the rules of partial differentiation:

$$\frac{\delta N}{\delta \xi} = \frac{\delta N}{\delta x}\frac{\delta x}{\delta \xi} + \frac{\delta N}{\delta y}\frac{\delta y}{\delta \xi} \qquad \text{Equation 8.12}$$

and

$$\frac{\delta N}{\delta \eta} = \frac{\delta N}{\delta x}\frac{\delta x}{\delta \eta} + \frac{\delta N}{\delta y}\frac{\delta y}{\delta \eta} \qquad \text{Equation 8.13}$$

In matrix form becomes:

$$\begin{pmatrix} \frac{\delta N}{\delta \xi} \\ \frac{\delta N}{\delta \eta} \end{pmatrix} = \begin{pmatrix} \frac{\delta x}{\delta \xi} & \frac{\delta y}{\delta \xi} \\ \frac{\delta x}{\delta \eta} & \frac{\delta y}{\delta \eta} \end{pmatrix} \begin{pmatrix} \frac{\delta N}{\delta x} \\ \frac{\delta N}{\delta y} \end{pmatrix} \qquad \text{Equation 8.14}$$

The inverse of this can be found by using the Jacobian matrix [J]:

$$\begin{pmatrix} \frac{\delta N}{\delta x} \\ \frac{\delta N}{\delta y} \end{pmatrix} = [J]^{-1} \begin{pmatrix} \frac{\delta N}{\delta \xi} \\ \frac{\delta N}{\delta \eta} \end{pmatrix} \qquad \text{Equation 8.15}$$

The derivatives thus have a mapping from local to x, y system.

8.2.1.3. Stress-strain equations

By using Hook's law, the strains can expressed in terms of stress:

$$\varepsilon_x = \frac{1}{E}(\sigma_x - \upsilon\sigma_y) \qquad \text{Equation 8.16}$$

$$\varepsilon_y = \frac{1}{E}(\sigma_y - \upsilon\sigma_x) \qquad \text{Equation 8.17}$$

$$\gamma_{xy} = \frac{2(1+\upsilon)}{E}\tau_{xy} \qquad \text{Equation 8.18}$$

Thus, the complete stress-strain relationship is:

$$\begin{bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{bmatrix} = \left[\frac{E}{1 - \upsilon^2} \right] \begin{bmatrix} 1 & \upsilon & 0 \\ \upsilon & 1 & 0 \\ 0 & 0 & (1-\upsilon)/2 \end{bmatrix} \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{bmatrix} \qquad \text{Equation 8.19}$$

This is written in the summary form:

$$\{\sigma\} = [D] \{\varepsilon\} \qquad \text{Equation 8.20}$$

8.2.1.4. Minimum potential energy formulation

The stiffness matrix of the structure can be derived by equating the external and internal work.

The work done by the nodal forces is the external work which is

$$\text{External work} = 1/2 \{ \delta^e \}^T \{ P^e \} \qquad \text{Equation 8.21}$$

The strain energy in the structure is the internal energy which is:

$$\text{Internal work} = 1/2 \int \{ \varepsilon \}^T \{ P^e \} \qquad \text{Equation 8.22}$$

Substituting equations 8.11 and 8.20 into equation 8.22 the internal work becomes:

$$\text{Internal work} = 1/2 \{\delta^e\} [\int [B]^T [D] [B] d (\text{volume})] \{\delta^e\} \qquad \text{Equation 8.23}$$

Equating equations 8.21 and 8.23 gives:

$$\{ P^e \} = \int [B]^T [D] [B] d (\text{volume}) \{\delta^e\} \qquad \text{Equation 8.24}$$

In the local co-ordinate system, the limits of integration needs to be changed:

$$\{ P^e \} = \int_{-1}^{1} \int_{-1}^{1} [B]^T [D] [B] [J] \{\delta^e\} d\xi \, d\eta \qquad \text{Equation 8.25}$$

Since K =Pδ, the stiffness matrix is:

$$\{ K \} = \int_{-1}^{1} \int_{-1}^{1} [B]^T [D] [B] [J] \, d\xi \, d\eta \qquad \text{Equation 8.26}$$

8.2.2. Finite element solution

Figure 8.3 summarises the significant steps in the finite element solution (Rocky et al, 1995). Each stage is discussed in the following sub-sections.

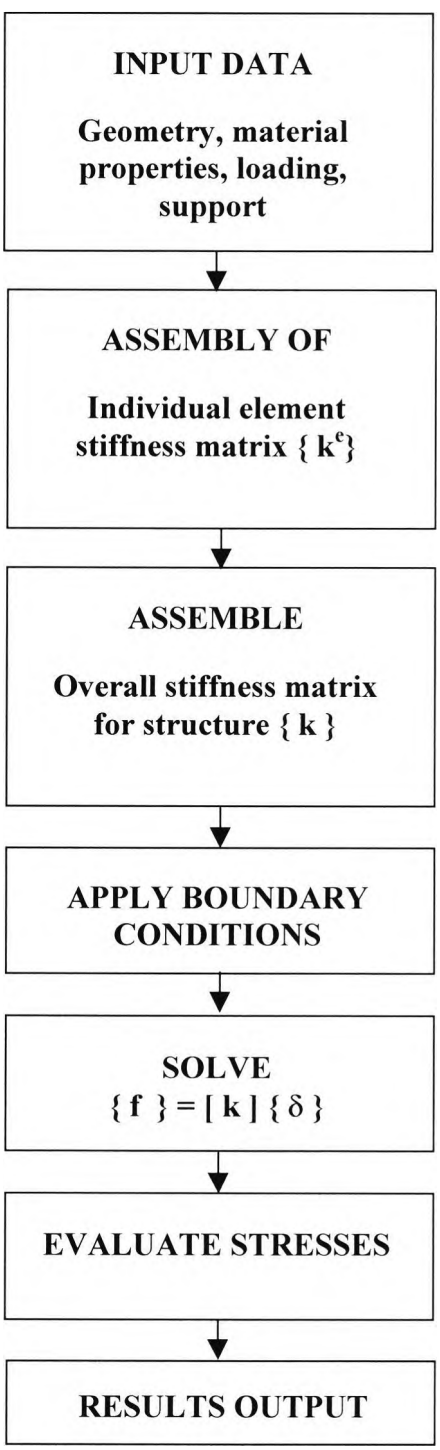


Figure 8.3: Finite element procedures in computer programming

8.2.2.1. Input data

The first stage of any F.E program is to read and check the data. This data file will contain the information required to define the model. Indeed, the data required to define the model for an analysis is:

- Nodes - The nodes are defined by their co-ordinates and each node is allocated a number.
- Elements - The elements are each allocated a number and are defined by the list of nodes which make up the element. The order of the node numbers is very important in the element definition, Appendix B shows an example element with the order in which the nodes should be specified. Each element also requires a physical and material property. These tables are listed in separate tables. The element data, therefore, also contains two pointers to the relevant entries in physical and material property tables (Gordon, 1993).
- Physical Properties - The physical properties of the element will vary for each element type. These properties may vary from element to element in which case a table is required. Each set of properties in the table are allocated a number. This number will then be specified in the element data.
- Material Properties - The material properties are listed in a similar manner to the physical properties. The properties are generally obtained from tables. An example is illustrated on table 8.1.
- Restraints (boundary conditions) - The boundary conditions are specified by a node number, a degree of freedom and a value. Generally, the node will be restrained and the value entered will be for a zero displacement.
- Loading - The load data is entered in a similar manner to the restraint data. The data may consist of a node number, a direction number and a value. The value entered will be the magnitude of the load and the sign will indicate the sense of the load. Alternatively, if a distributed load is applied, the data will be an element number, and edge number and the value of the pressure (The impact of a hammer is simulated by applying pressure on one or two elements).
- Specification of Results Required -The user generally specifies the result to be calculated and displayed in the output file. Some of the results obtained are the displacements, reactions, strains, stresses, etc. To obtain results from the completed solution, node number, type of results e.g. displacement, axis direction in which the results are to be obtained need to defined.

Young's Modulus (Pa)	Density (kg/m3)	Poisson's Ratio
33100 x 10^6	2300	0.2

Table 8.1: Material properties

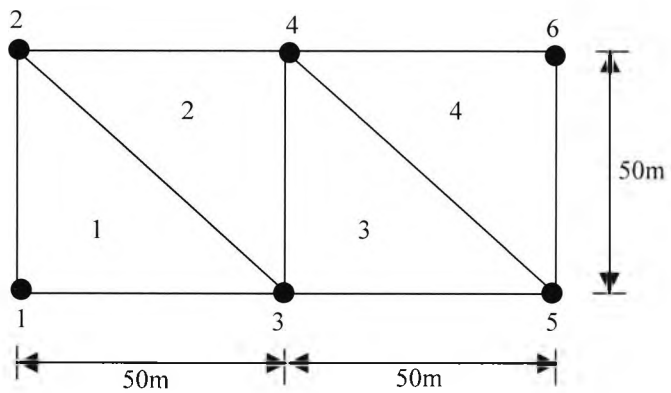


Figure 8.4: Example of 2D model using triangular elements

Considering the 2D model example in figure 8.4, there 6 nodes and 4 elements. At each node there are 2 DOF(degrees of freedom) which are displacements, V_n is vertical and U_n is horizontal, n refers to the number of each node.

For the example, the nodes are stated as:

N,1,0,0
N,2,0,50
N,3.50,0
N,4.50,50
N,5,100,0
N,6,100,50

The elements become:

EN,1,1,3,2
EN,2,2,3,4
EN,3,3,5,4
EN,4,4,5,6

8.2.2.2. Assembly of the overall stiffness matrix

When the calculation of the stiffness matrix for each element is completed. As described in section 8.2.1, the overall stiffness matrix has to be assembled. Using the 2D example in figure 8.4, the example has 2 DOF at each nodes, giving a 12 x 12 overall stiffness matrix. For each element, the stiffness matrix is 6 x 6 because there are

3 nodes defining an element. For element 2 the 6 x 6 element stiffness matrix has included into the global matrix. This is shown below in figure 8.5. Once all the stiffness matrixes for all 4 elements have been included into the global matrix, then it is complete.

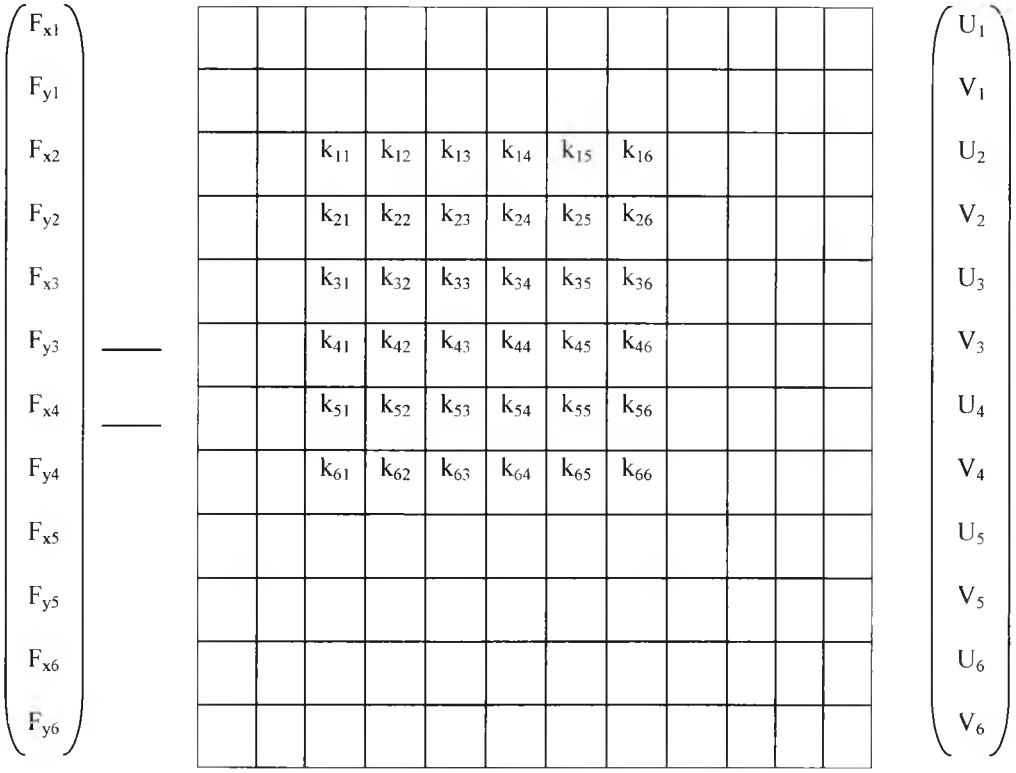


Figure 8. 5: Stiffness matrix of element 2 in the global matrix

8.2.2.3. Application of boundary conditions

In most cases, the finite element model will be constrained at certain nodes, in order to prevent the model accelerating under the action of applied forces. This allows the appropriate rows and columns to be eliminated. This is done by several methods employed to carry out this inspection. One method is to multiply the diagonal of the overall stiffness matrix by a large number. This is demonstrated in equations 8.27, 8.28 and 8.29:

$$F_{xi} = K_{i1}U_1 + K_{i2}V1 + K_{i3}U_2 + \ldots + K_{ii}U_i \ldots + K_{in}V_n \qquad \text{Equation 8.27}$$

$$F_{xi} = K_{i1}U_1 + K_{i2}V1 + K_{i3}U_2 + \ldots + (10^{25})\,K_{ii}U_i \ldots + K_{in}V_n \qquad \text{Equation 8.28}$$

$$U_i = \frac{F_{xi} - (K_{i1}U1 + K_{i2}V1 + K_{i3}U_2 \ \ldots + K_{in}V_n)}{(10^{25})\,K_{ii}U_i} \qquad \text{Equation 8.29}$$

8.2.2.4. Solution for displacements

Many algorithms are available to solve the stiffness equations. Gaussian elimination is one such method, but the disadvantage is that it requires long periods to solve a particular problem. Hence, it is vital to minimize the matrix by labeling the nodes and elements appropriately. A review of the many efficient alternatives is beyond the scope of this research.

8.2.3. Loading

When forces acting on an elastic body varies with time, it is essentially a dynamic problem. The stress waves initiated by the exciting forces, propagate within the body which deforms as a result in a time dependant fashion. So, the analysis of these problems involves a study of the manifestations of wave motion within the body.

The finite element can be used to analyse dynamic problems. This requires solution of three matrices:

- Stiffness matrix-This is the relation of an element's extremity forces to its extremity displacement, global stiffness is the assembled stiffness matrix of all the elements.
- Mass matrix-the inertia of body represents its internal resistance to vibration, in the discretized form, the total inertia of the body is represented by global mass matrix.
- Damping matrix-In damped vibration, the resistance is also provided by the internal friction of the body which results in energy dissipation. The total damping is represented by its global damping matrix.

The resulting system of equation is:

$$F(t) = [M] \{d^2u/dt^2\} + [C]\{du/dt\} + [K] \{u\} \qquad \text{Equation 8.30}$$

F(t) is the forcing function (½ sine wave) – the transient dynamic analyses are used to solve for the response of a structure to a time-history forcing function i.e. the ½ sine wave which is described in chapter 10. When the ½ sine wave impact is applied, the forcing function is then set to zero, allowing results to be produced for specified times after the impulse.

[M] is the structural mass matrix- this is defined by :

$$[M] = \sum_{i=1}^{NE} [M^e_i] \qquad \text{Equation 8.31}$$

where $[M^e_i]$ is the individual element matrices

[C] is the structural damping matrix- this is represented by:

$$[C] = \alpha [M] + \beta[K] + \sum_{j=1}^{NMAT} \beta_j[k_j] + [C_\xi] + \sum_{k=1}^{NEL} [C_k] \qquad \text{Equation 8.32}$$

where $\alpha [M]$ is the mass damping,

$\beta[K] + \sum_{j=1}^{NMAT} \beta_j[k_j]$ is the structural damping, $[k_j]$ is the portion of the stiffness matrix based on material j, NMAT is number of materials in the model,

$[C_\xi]$ is the constant damping,

$\sum_{k=1}^{NEL} [C_k]$ is the element damping, $[C_k]$ is the individual element damping matrix, NEL is the number of damping elements

[K]is the overall structural stiffness matrix-this is defined by:

$$[K] = \sum_{i=1}^{NE} [K^e_i] \qquad \text{Equation 8.33}$$

where NE is the number of elements,

$[K^e_i]$ is the individual element stiffness matrices

{u} = Nodal displacement vector

{du/dt} = Nodal velocity vector

$\{d^2u/dt^2\}$ = Nodal acceleration vector

The dynamic analysis is applicable to any and all non-linearities, and uses a direct time integration scheme to solve for the unknown displacements at a given time point for

transient analysis (ANSYS). The Newmark integration method is employed in the finite element method , which consider the following :

1. The displacement vector (u_n) is a function of previous and current displacements, velocities and accelerations.
2. The solution of a linear system will never diverge, no matter how large the integration time step (Δt) is. Δt is defined as the difference between successive time-points, i.e. ,

$$\Delta t = t_n - t_{n-1}$$

Using the equation of motion from equation 8.30 :

$$F\{t\} = [M]\{d^2u/dt^2\} + [C]\{du/dt\} + [K]\{u\}$$

For non-linearities, the Newmark equation can be solved iteratively at a single time point, so to provide any number of equilibrum iterations:

$$(a_0[M] + a_1[C] + [K_t^i]\{\Delta u_t^{i+1}\} = \{F_t\} - \{F_t^{nr,i}\}$$

$$+ [M](a_0(\{u_{t-\Delta t}\} - \{u_t^i\}) + a_2\{du/dt_{t-\Delta t}\} + a_3\{du^2/dt_{t-\Delta t}\}) + [C](a_1(\{u_{t-\Delta t}\} - \{u_t^i\})$$

$$+ a_4\{du/dt_{t-\Delta t}\} + a_5\{du^2/dt_{t-\Delta t}\})$$

Where a_0 and a_5 are functions of the time step size and Newmark integration parameters.

8.3. Summary

The finite element theory was presented, using a 2D element to describe the stages needed to deduce the stiffness matrix by a general approach. A simple diagram was used to illustrate the computer programming of the finite element method. In the next chapter, the theory of the interface element is covered.

Chapter nine

Theory of contact elements

9.1. Introduction

In the previous chapter, the theory of finite element was presented. This chapter discusses, the use of the contact element in finite element modelling of interfaces of the type arising with crack type defects. In the research, use of contact elements is investigated as a means of achieving more realistic simulations for cases with planner defects. The theory that governs this element is presented and its application. Two types of contact elements are reported.

9.2. Theory

Use of the interface elements allows a 'no-tension' regime with compression-shear contact forces developing on crack closure. The contact element shown in figure 9.1 is used as an example to show this characteristic. Figure 9.2 illustrates the contact force components f_n, the normal force across the gap and f_s the sticking force across the gap. In the force-displacement graphs, k_n and k_s are the normal and sticking stiffness, respectively. In the analysis, the stiffness contribution of the interface element in the closed-friction state is given by equation 9.1. The numbers 1 to 6 in equation 9.1 and 9.2, refer to the degrees of freedom illustrated in figure 9.3.

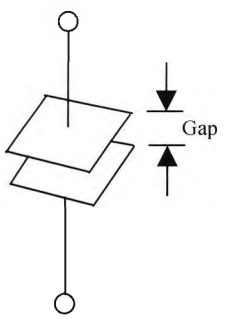


Figure 9.1: Contact element

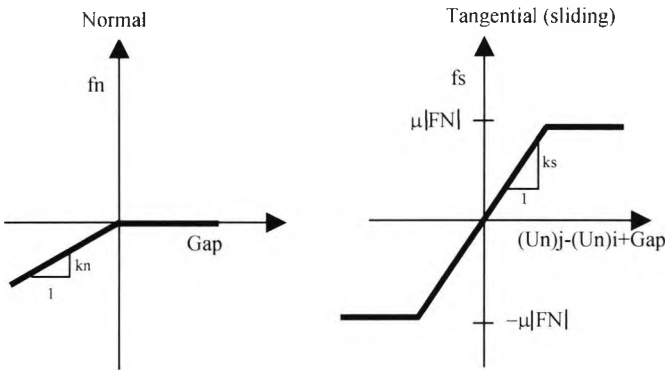


Figure 9.2: Force-deflection relationship for a contact element

$$K_L = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{pmatrix} k_n & 0 & 0 & -k_n & 0 & 0 \\ 0 & k_s & 0 & 0 & -k_s & 0 \\ 0 & 0 & k_s & 0 & 0 & -k_s \\ -k_n & 0 & 0 & k_n & 0 & 0 \\ 0 & -k_s & 0 & 0 & k_s & 0 \\ 0 & 0 & -k_s & 0 & 0 & k_s \end{pmatrix} \end{array} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix}$$

Equation 9.1

For the closed-sliding state, however, its contribution is reduced to:

$$K_L = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{pmatrix} k_n & 0 & 0 & -k_n & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -k_n & 0 & 0 & k_n & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix}$$

Equation 9.2

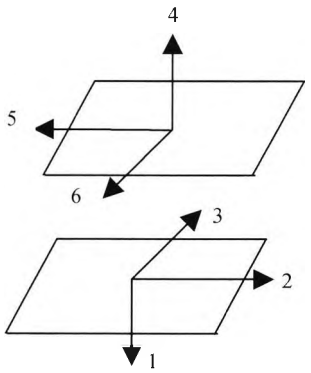


Figure 9.3: Degrees of freedom for the interface element

9.3. Application of contact elements

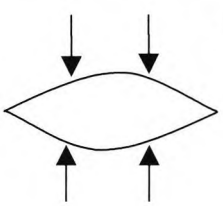
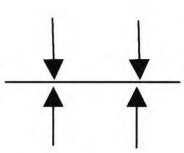
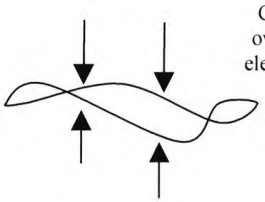
Gap Status	Without Contact Element	With Contact Element
 Gap Open	No forces transferred	No forces transferred
 Gap Closed	No forces transferred	Compression forces transferred (shear also possible)
 Cross over of elements	Possible	Not possible

Figure 9.4: Gap behaviour with and without use of contact element

Where planer cracks and delaminations occur in concrete, these give rise to a hollow sound, which is caused by the two surfaces hitting together. In finite element modelling, this situation needs to be modelled effectively. The contact element can be used to achieve this effect. It does this by acting as a secondary vibration, the interface element rebounding the stress waves. Without the use of this element in the analysis, the two surfaces can pass through each other, not a transmitting stress wave.

The modeling of impact echo by finite element has been successfully implemented (Sansalone, Carino & Nelson, 1987; Sansalone & Carino, 1990; Lin, Sansalone & Carino, 1990; Cheng & Sansalone, 1991), however, these researchers have not considered the effect of a contact element. In reality, a crack can be a fraction of a mm in width. Use of the contact element is advantageous in this case, because such dimension can be modeled. More importantly, additional frequency features should be apparent for simulated defects in the finite element model of the impact echo system.

The contact element was employed to allow simulation of the two surface of a crack colliding. This is shown in figure 9.4. The interface shows two surfaces which can break physical contact. As discussed in the following sections, the contact element gives the possibility for compression in the direction normal to the surfaces, this makes it possible to model a delamination.

9.3.1. Calculation of stiffness

The calculation of the stiffness of the contact element is based on the stiffness of the surrounding solid material, which has been removed and replaced by the contact element. The stiffness is deduced by:

$$\text{Stiffness} = AE/L \qquad \text{Equation 9.3}$$

Where A is the surface area associated with a contact node,
E is the Young's modulus for the test solid,
L is the length of the contact element

When the contact element gap is closed, the element behaves like a solid material and, when the gap opens, the stiffness is reduced to represent the loss of stiffness.

The next sections describe two type of contact elements, which have been studied to observe the performances of their capabilities and any differences between them.

9.4. Contac12 element

Contac12 is a 2D point to point contact element. It represents two surfaces which may break physical contact and may slide relative to each other. The element can support only compression in the direction normal to the surfaces and shear in the tangential direction. It has two degrees of freedom at each node which is translation in the x and y directions.

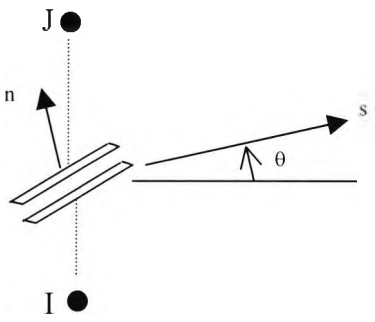


Figure 9.5 : Contac12

As shown in figure 9.5, the element is presented by two nodes, an angle to define the interface, two stiffness (k_n and k_s), an initial displacement interference (gap).With the use of this interface element, the impact point is known which can be seen in figure 9.5.

9.5. Contac48 element

This element can be used to represent contact and sliding between two surfaces in two directions. It has two degrees of freedom at each node which are translations in x and y directions. Contact is apparent when the contact node penetrates the target line which acts like a billiard ball where the impact points is not know. This is the major difference between Contact 48 and Contact12.

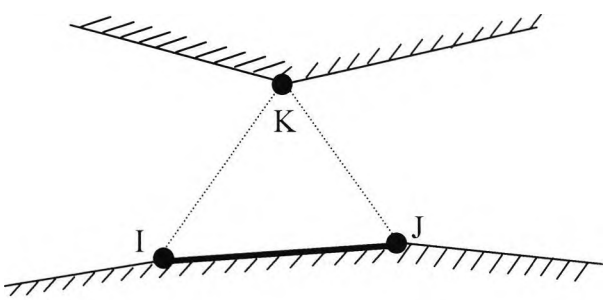


Figure 9.6: Contac48

Figure 9.6 shows the geometry and node locations. The element geometry is a triangle with the base as a line connected by two nodes on one of the surfaces (target line). On the opposite surface, lies a node (contact surface). The line on the target surface is a target line, and the nodes at the ends of the target line are target nodes. The node on the contact surface is a contact node.

It was decided to use Contac12 because it was quicker to run. Modeling of defects in the shear direction was not of interest to the author, which is the case with the Contac48 element.

9.6. Summary

The theory and application of the contact element has been presented. They enable interface problems to be more correctly represented in impact echo systems having related defects. The two contact elements, Contac12 and Contac48 have been presented, impact is pre-determined with the first element, whereas impact is not pre-determined with the other.

In the next chapter, the results of finite element cases, including the contribution of Contac12, is presented.

Chapter ten

Modeling of impact echo systems

10.1. Introduction

Chapters 8 and 9 give the theory of finite elements and the interface element respectively. This chapter illustrates the solution method for the transient impact response of bounded structures containing defects, and discusses the way this numerical method has been used to simulate the impact echo system. The modelling of the impact is shown, and the calculation of the size of element and Integration Time Step (ITS) explained. Finally, the results of the test cases are presented.

10.2. Modelling of impact echo systems

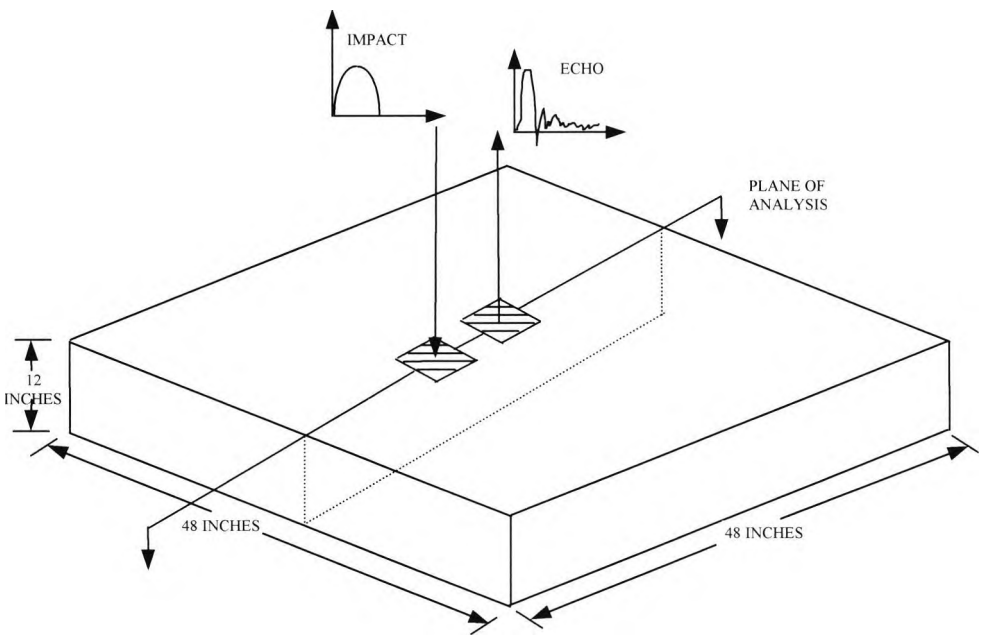


Figure 10.1: Simulation of the impact echo system using finite elements

Figure 10.1 shows the basic model of the impact echo system. The impact is generated by applying a pressure loading with a force-time history of a half-cycle sine curve over an element. Here, force can be used instead of pressure.

The waveform produced by the impact is a simulation of the impact echo response measured by a surface displacement transducer.

10.2.1. Impact pulse

As discussed in chapter 2, the contact time or the duration of the impact in concrete is an important variable in impact echo testing (Pratt & Sansalone, 1991). This is because the contact time influences the frequency content of the stress pulse. Impacts with short contact time contain higher frequency (shorter wavelength) components which are needed when testing thin objects and detecting shallow flaws i.e. demanding high resolution.

As discussed, the impact of a spherical mass was simulated by applying a uniform pressure over an elements at the centre of the plate or a force at a specified node. The time history of the applied pressure was half a sin curve, made up of small individual pulses, as illustrated in figure 10.2. In fact, the more points used to define the ½ sine wave, the smoother the output response. Hence, an accurate result is obtained.

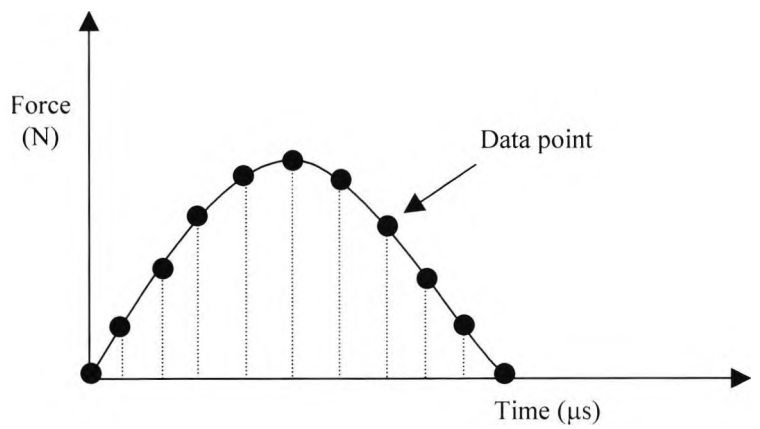


Figure 10.2: Simulating the impact time

The impact is represented by 30-25 points for the 1/2 sine wave. To enable the response to be examined after the end of the pulse, it is necessary to run the pulse on with zero force for a longer period of time. For example, if the total duration is 10240 μs, 10215 μs is allowed to respond to this impact force, this is shown in figure 10.3. As output is provided at each load step, the number of the load steps corresponding to the 'number of sampling points' from the force against time graph in figure 10.2. Equal time steps are used with the output, fed directly to the FFT analyser as discussed in chapters 3 and 4.

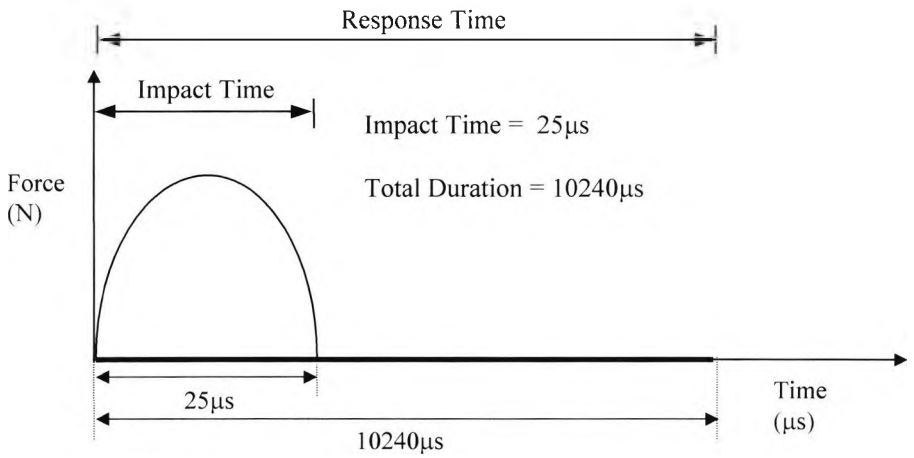


Figure 10.3: Total impact of the finite element simulation

10.2.2. Determining element size

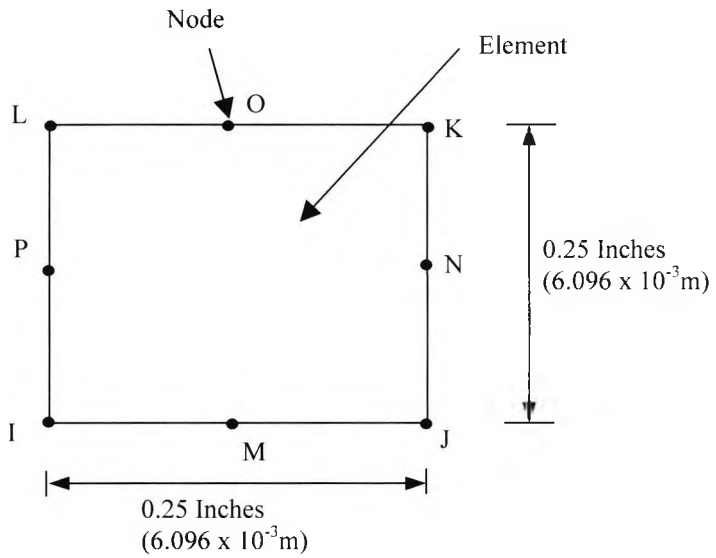


Figure 10.4: Size of element

For 0.25m to 0.5m thick plates subjected to a force-time function (simulated impact by steel sphere and contact time = 25-31 microseconds), rectangular elements with dimensions of the order of 0.02 times the plate thickness, give best results (Sansalone, Carino, Nelson, 1987). Thus:

$$E_s = 0.02 \times d \qquad \text{Equation 10.1}$$

Where:

E_s = Element size

d = Plate thickness

To calculate element size for the model shown in figure 10.1 :

If d = Plate thickness = 0.3048m

Then $E_s = 0.02 \times d = 0.02 \times 0.3048 = 6.096 \times 10^{-3}$ m

The element size indicated above is the ideal dimension to use, which is shown in figure 10.4. However, to achieve an accurate result, requires thousands of elements. This is difficult to achieve with the computer resources available. As explained in chapter 1, processing time is subsequently very slow.

10.2.3. Solution time step solution

The accuracy of the solution is dependant on the time interval for the cycles of solution. This is referred to as the ITS, which is a function of problem dimension, element size, mass density and wave propagation characteristics (ANSYS, 1993). This is used in the finite element model, so that a smooth transient response representation of the impact is achieved. Hence, the ITS governs the accuracy of a transient solution. The smaller the ITS, the better the accuracy, but the larger the ITS, the faster the computer will finish the solution. It is vital to select an optimum ITS.

Methods have been proposed for estimating suitable ITS values (Sansalone, Carino, & Nelson, 1987; ANSYS, 1993). Calculation by two different methods is given in the following sections.

10.2.3.1. Method1(ITS_{m1})

Where the effects of the elastic wave at the micro level are to be analysed, the ITS must be small enough to resolve this wave. More elements along the direction of the wave will be necessary for the wave shape to be adequately represented. The ITS is estimated as (ANSYS,1993):

$$ITS_{m1} \leq \Delta x/3C \qquad \text{Equation 10.2}$$

Where :

- Δx is the element size less than or equal to L/20
- L is the structure dimension along direction of wave or wavelength of the excitation which ever is less
- C is the elastic wave speed which is $(E/\rho)^{1/2}$
- E is the elastic modulus
- ρ is the mass density

For the wall specimen in figure 10.1:

$L = 12 \times 25.4 \times 10^{-3}$ m (To convert from inches to metres)

Therefore : $\Delta x = L/20 = (12 \times 25.4 \times 10^{-3}) / 20 = 0.01524$m

$E = 33100 \times 10^{6} N/m^{2}$
$\rho = 2300$ kg/m^{3}

Therefore : $(E/\rho)^{1/2} = (33100 \times 10^{6}/2300)^{1/2} = 3793.587266$ ms^{-1}

Therefore : $ITS_{m1} = (0.01524/(3 \times 3793.587266)) = 1.339 \times 10^{-6}$ s

10.2.3.2. Method2(ITS_{m2})

The time step required for numerical stability, wave propagation applications require the use of very small time steps to obtain an accurate solution. Numerical stability requires that the time step ITS, meet the following criterion (Sansalone, Carino, & Nelson, 1987):

$$ITS_{m2} \leq ITS_{max} = L/C_p \qquad \text{Equation 10.3}$$

Where:

L is the Shortest dimension of element (m),

C_p is the P wave speed in material

So it was advisable to use:

$$ITS_{m2} = 0.67 ITS_{max} \quad \text{i.e } ITS_{m2} \leq 0.67\, L/C_p \qquad \text{Equation 10.4}$$

To check the ITS_{m1} value with this method:

So far in the analysis the dimensions of an element was 0.25 by 0.25 inches , so

if L = Shortest dimension of element (m) = $0.25 \times 25.4 \times 10^{-3} = 6.35 \times 10^{-3}$m

C_p =P wave speed in material (m/s) = 11500ft/s = 11500x0.305=3507.5m/s

therefore

$ITS_{max} = L/C_p = 6.35 \times 10^{-3} / 3507.5 = 1.810 \times 10^{-6}$s

$ITS_{m2} = 0.67 ITS_{max} = 0.67 \times 1.810 \times 10^{-6} = 1.213 \times 10^{-6}$s

Comparing this value with the calculated ITS_{m1} value calculated in 10.2.3.1:

$$ITS_{m2} = 0.67ITS_{max} = \underline{1.213 \times 10^{-6}s}$$

and

$$ITS_{m1} = \underline{1.339 \times 10^{-6}\ s} \text{ from previous section}$$

hence these values satisfy both recommendations embodied in equations 10.2 and 10.4.

The ITS used in the finite element modelling was 1.34×10^{-6} s because of time limitations, smaller the ITS, the longer the model took to run.

10.2.4. Finite element mesh

When nodes are numbered and defined by their co-ordinates, elements need to be generated, this is called meshing, an example of the problem model based on figure 10.1, is shown in figure 10.5. It shows the number of elements needed if the size of an element was 6.096×10^{-3}m, which was calculated from equation 10.1, this gives a total of 10,000 elements.

There are two types of modelling, one is done manually, this is shown in figure 10.5. The other is automatic, this shown in figures 10.9 and 10.15. These methods are discussed further in Appendix B.

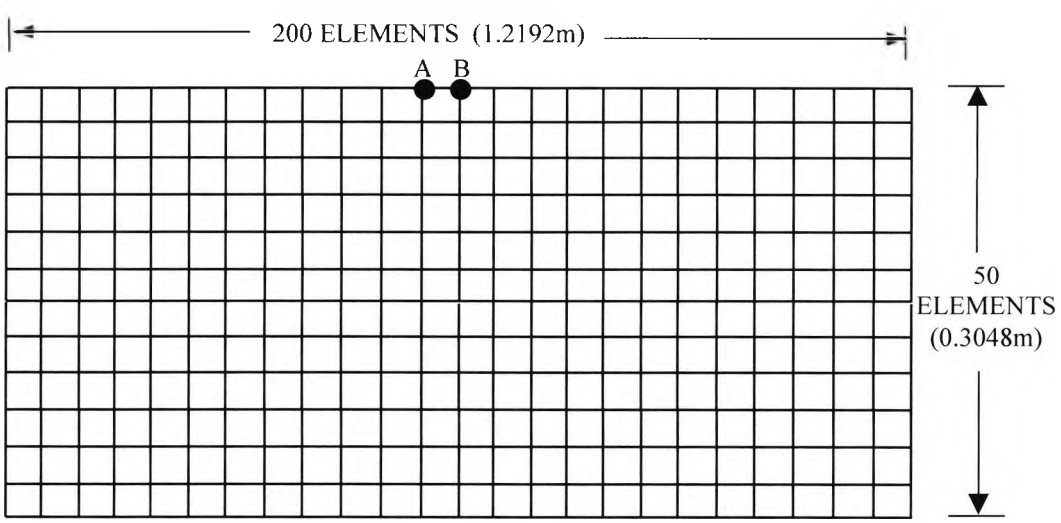


Fig 10.5 : Illustration of problem model (not drawn to Scale), point A is the impact point and point B is the receiver point

10.3. Test cases

The types of defects/non defects in concrete modelled were:

- 2D solid (Verifying Lin, Sansalone & Carino,1990)
- Voids using contact elements in 2D
- Voids without using contact element in 2D
- Multiple voids in 3D (a simulation of wall A of the Olson specimen wall, discussed in chapter 3 and 4).

The following sections describe the method of modelling each test case and presents the results, accordingly.

10.3.1. Axisymmetric model of impact echo system

The objective of Lin, Sansalone & Carino (1990) was to investigate the possibility of using the impact echo method for detecting voids in plates containing thin layers. These authors have modelled several finite element cases of plastics ducts with and without voids.

The author developed a finite element model to verify Lin et al results. Also, this served as a starting point to actually model a real life situation, hence gaining some useful experience. The plate is 0.3 m thick and 1.8 m in diameter, in the finite element model half of the plate diameter was used. A representation of a solid, defect free, model is shown in figure 10.6.

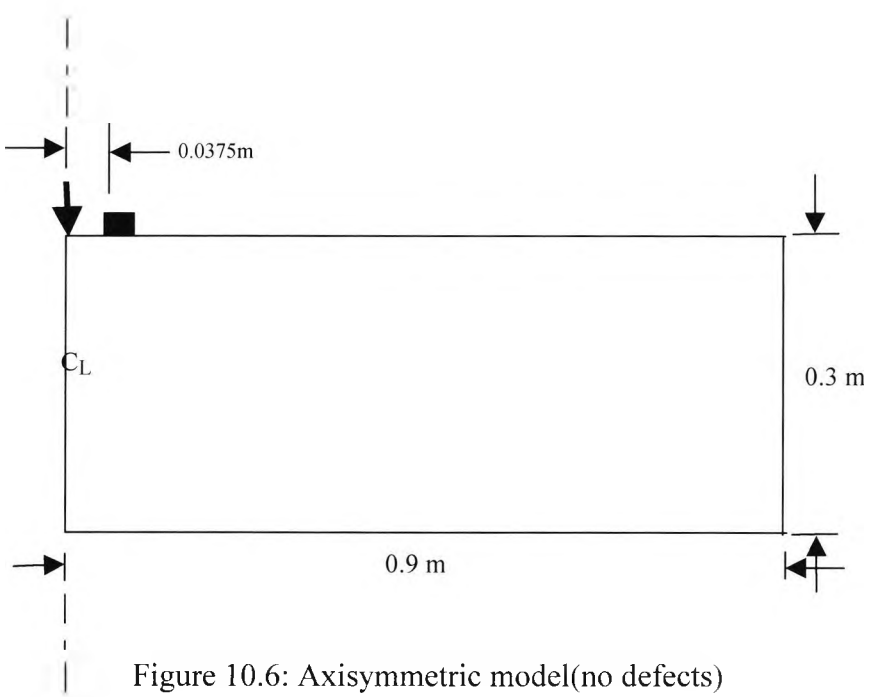


Figure 10.6: Axisymmetric model(no defects)

In the analyses, a 2D, axisymmetric element was used. The impact of a spherical mass was simulated by applying a uniform pressure over one element (including the mid-nodes of Plane82), Lin et al have used two elements. The response was measured five elements from the impact point. The advantage of using the element Plane82 with its 8 nodes, means for every 1 element used by the researchers, Plane82 uses 4 elements, the other differences are shown on table 10.2. The impact was applied at the centre of the plate. The typical dimension of the quadrilateral elements used in the investigation was 0.0075 m in both directions. This confirms equation 10.4 (size of element =0.02 x 0.3=0.006), which can be used as an estimation of the dimension to use.

The generated strains are very low in impact echo testing (Pratt & Sansalone, 1991), the elastic properties shown in table 10.1 represent these values at low strain levels.

Material	Density (kg/m^3)	Youngs Modulus (Pa)	Poisson's Ratio
Concrete	2300	33100	0.2

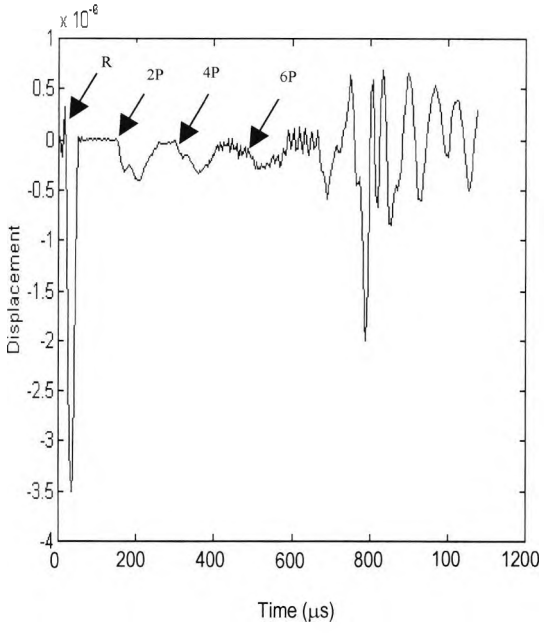
Table 10.1: Material properties for the axisymmetric model

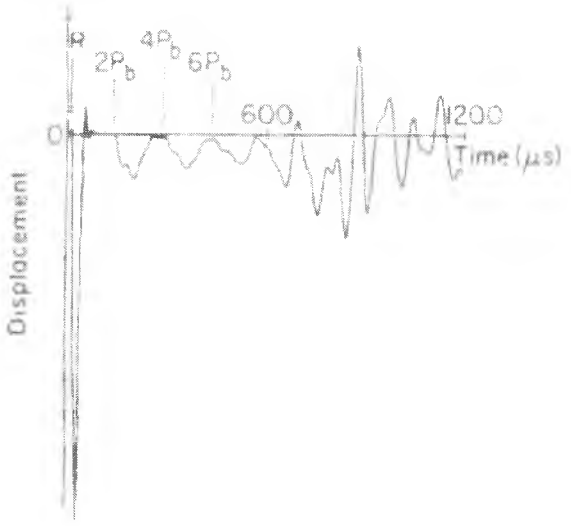
Lin et al	Authors
No constraints used	Constraints used on the centre line
40 elements through the plate thickness, 120 elements through half of the plate diameter used, total elements = 4800	21 elements through the plate thickness, 60 elements through half of the plate diameter used, total elements = 1260

Table 10.2: Comparing results

10.3.1.1. Results of axisymmetric model

The results for the defect free solid, model is illustrated below. It shows the authors results against Lin et al. The waveform is shown in time and frequency domain.

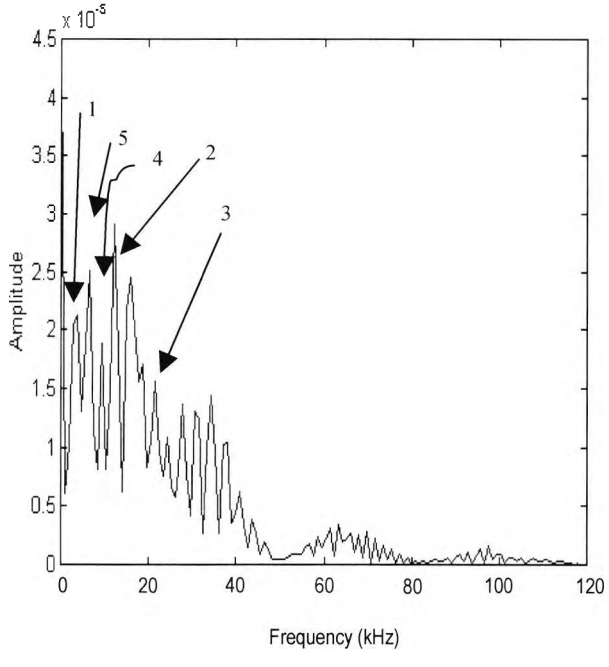


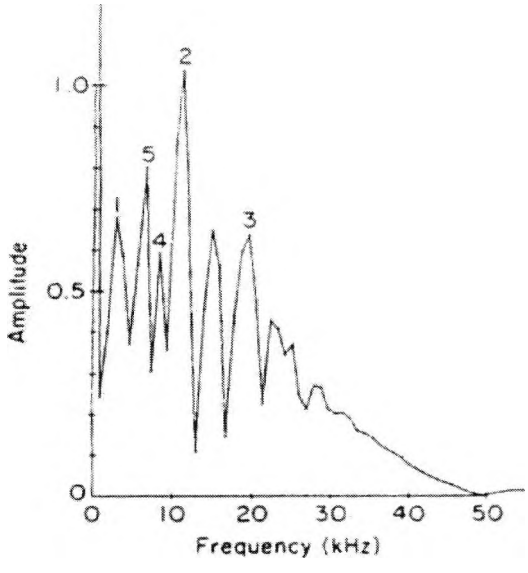


Author

Lin, Sansalone, & Carino, 1990

Figure 10.7: Waveform of defect free, solid in time domain





Author

Lin, Sansalone, & Carino, 1990

Figures 10.8: Amplitude spectrum of defect free, solid in frequency domain

Figure 10.7 shows the surface displacement waveform at a point located 0.0375 m from the impact point. As discussed in chapter 2, the waveform contains an initial large downward displacement produced by the arrival of the R-wave and displacement caused by the arrival of P waves which multiply between the top and bottom surface of the plate, as well as from the perimeter of the plate. Figure 10.8 illustrates the amplitude spectrum. It shows the three flexural modes labelled 1,2 and 3. The frequency peaks labelled 4 and 5 are the rod mode and P-wave thickness mode. These frequencies are compared against the corresponding depths for the author's work and that of Lin et al.

The results of the author's verification of Lin et al result is presented in table 10.3. The results are over 90% in agreement, satisfactory, particularly considering that the author did not use as many elements as the Lin et al group did. Use of less elements would tend to make the author's model stiffer, thus achieving slightly high frequencies, as found.

	Lin et al Frequency	**Lin et al Depth**	**Author Frequency**	**Author Depth**
Solid plate	1= 2.8kHz 2=11.2kHz 3=19.5kHz 4=8.4kHz 5=6.7kHz	1=0.71m 2=0.18m 3=0.10m 4=0.24m 5=0.30m	1=3.1kHz 2=11.5kHz 3=21.5kHz 4=9.2kHz 5=6.2kHz	1=0.65m 2=0.17m 3=0.09m 4=0.22m 5=0.32m

Table 10.3: Comparison of Lin et al and author's result for the defect free, solid

10.3.2. 2D defect model with contact element

Figure 10.9 shows the automatic mesh of the defect model which incorporates contact elements. Figure 10.10 shows a schematic illustration of the model developed to study the effect of using contact elements. As described in chapter 9, the defect modelled by the contact element is planer, with an initial gap of 0.1 mm. It has two nodes with stiffness', in the normal and shear direction (k_n and k_s) relative to the gap direction. The area where the contact element was implemented is shown in figure 10.10. The material properties used for both elements, are given in tables 10.4 and 10.5. The details of the model are:

- Total number of elements = 250
- Total number of contact elements = 8
- Element type = Plane 82
- Interface element used = Contac12

- Force used to generate the impact
- Impact time = 30 μs
- Total duration = 10240 μs
- Distance of impact and receiver = 0.01852 m

The model was run with the contact element, with results presented in time-frequency domain, as shown in figures 10.11 and 10.12. It was run again, without the contact element, to the same dimensions described in figure 10.10. The results are presented in the time and frequency domain, as shown in figures 10.13 and 10.14.

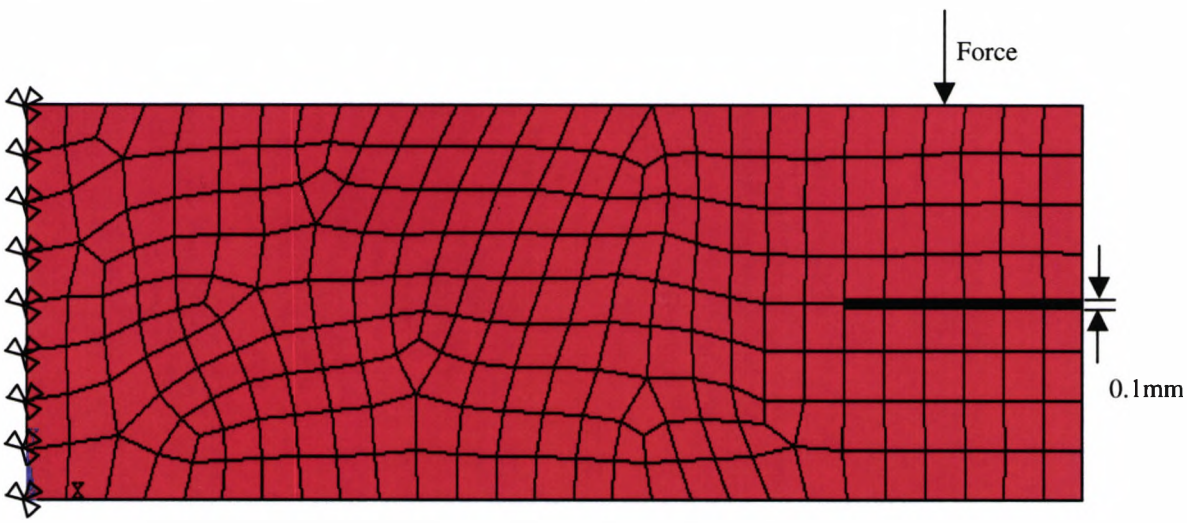


Figure 10.9: Mesh of the model with contact element

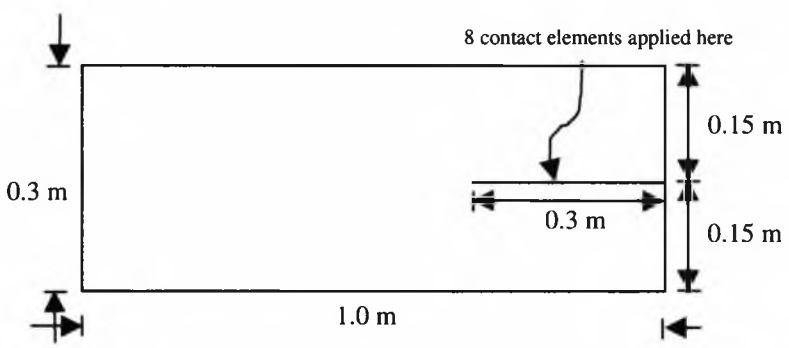


Figure 10.10: Modelling with the contact element

Material	Density (t/m^3)	Young's Modulus kN/m^2	Poisson's Ratio
Concrete	2.3	$3.31x10^7$	0.2

Table 10.4: Material properties for Plane 82

K_n (kN/m)	K_s (kN/m)	g (m)	t (initial status open crack)	Mu (coulomb friction)
8.275×10^6	8.275×10^5	0.0001	3	0.1

Table 10.5: Material properties for Contac12

10.3.2.1. Results of 2D model with & without contact element

The results of with and without contact element is presented below.

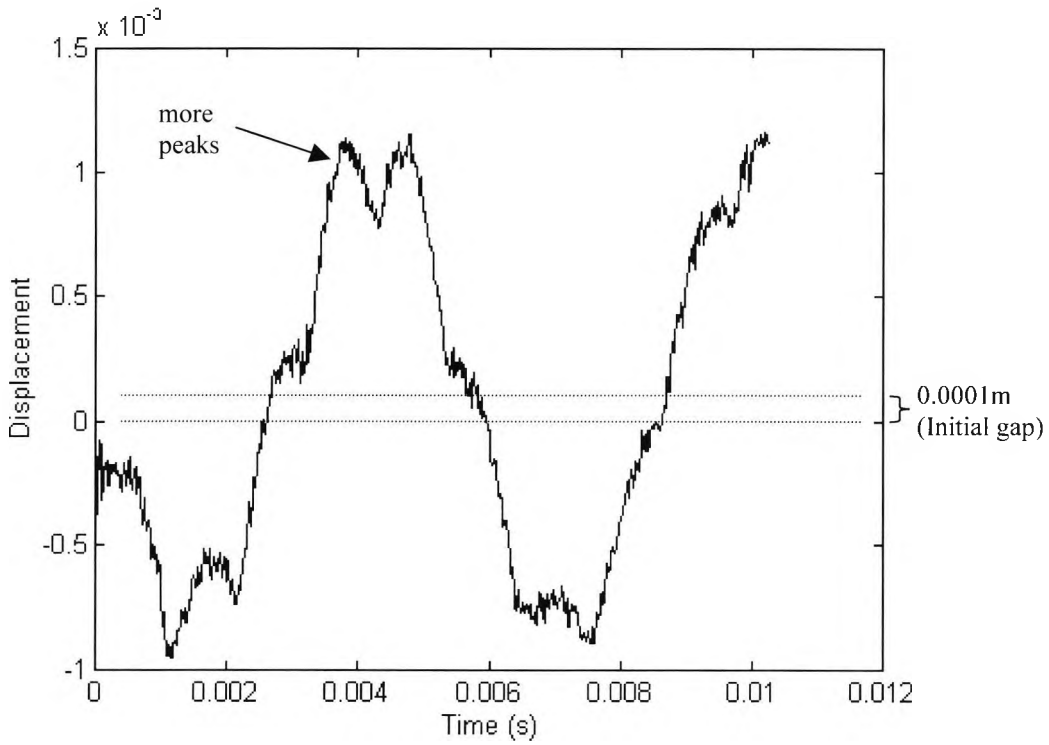


Figure 10.11: Waveform of model with contact element in time domain

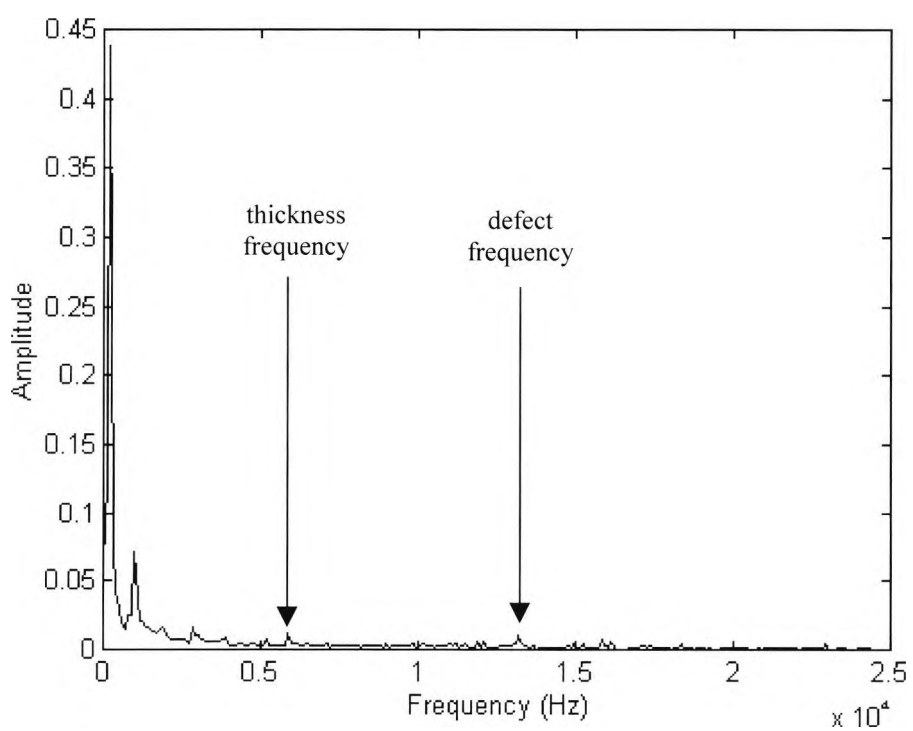


Figure 10.12: Amplitude spectrum of model with contact element in frequency domain

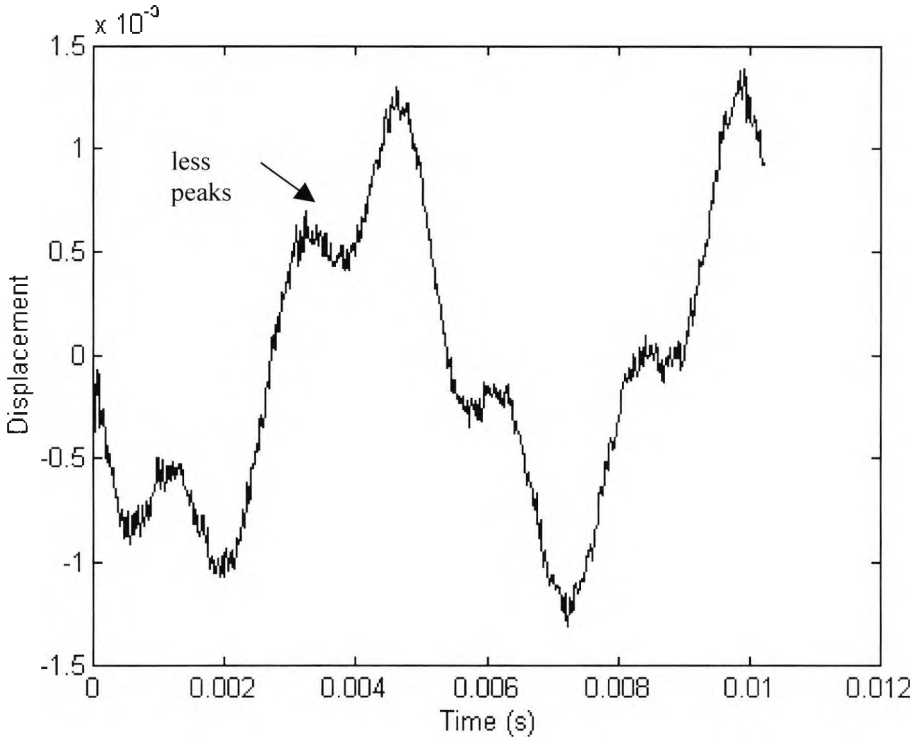


Figure 10.13: Waveform of model without contact element in time domain

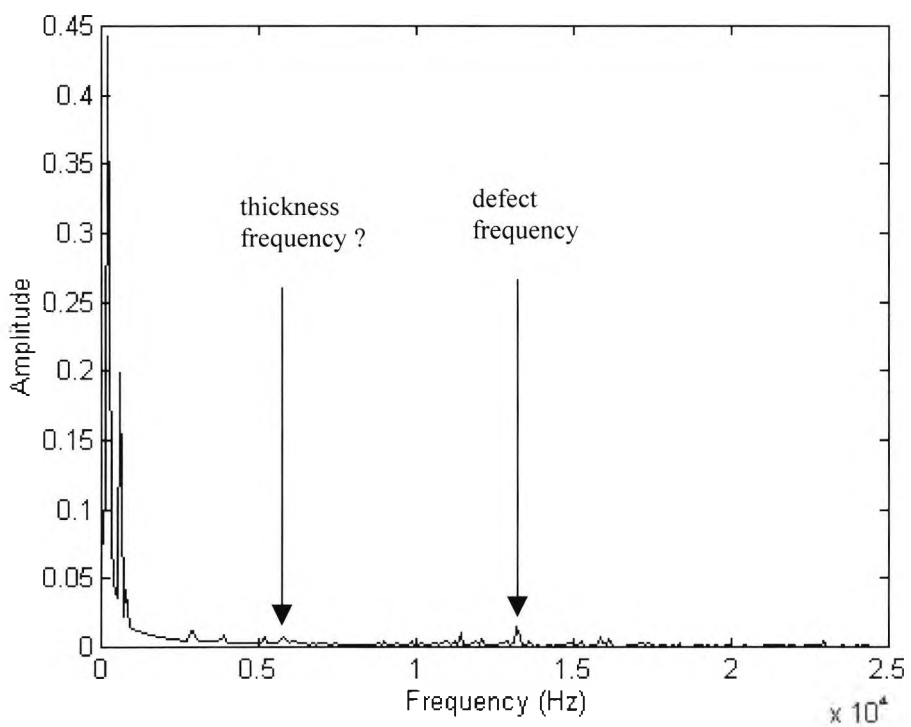


Figure 10.14: Amplitude spectrum of model without contact element in frequency domain

The results for with and without contact element, shown in figures 10.11 to 10.14 are similar, but the difference is that results of figure 10.11 and 10.12 show two cantilevers hitting each other occasionally, while the result of figure 10.13 and 10.14 show two cantilevers passing through each other.

The defect with 0.15m depth gives a predicted frequency of 13.33 kHz. This is indicated in both graphs where it is 13.23 kHz. However, for a depth of 0.3m, the predicted thickness frequency is 6.67 kHz, the amplitude graph with the contact element shows a dominate peak of about 5.89 kHz. This suggest, that the signal has actually passed through the contact element to reach the bottom of the plate, and return back to the top surface. So the two surfaces of the defect were in contact during the compression force. The amplitude graph without the contact element does not clearly show a peak in that region, this suggest the opposite i.e. the two surfaces of the defect were not in contact during compression force, confirming the effect achieved without contact elements discussed in section 9.3 of Chapter 9.

In fact, the model with the interface element is more realistic as expected. Indeed, the response with the contact element is more complex with additional peaks compared with the model without contact element. These are shown in figures 10.11 and 10.13. It can be seen that after the 'gap' of 0.0001 m, there are peaks. These additional peaks could be due to the effects of the surfaces of the defect colliding, the peaks between ± 0.001m maybe the displacements of each surface of the defect after collision. Also, it is nosier than the model without the contact element.

The amplitude graph of figure 10.12 show pronounced peaks before the 5 kHz, which suggest perhaps a beat phenomena where the collision of the two sides of the defect have a pronounced lower frequency feature. However, the amplitude spectrum of the model without contact elements does not show many peaks before the 5 kHz.

10.3.3. 3D model of Wall A (Olson specimen wall)

The 3D model is the simulation of wall A of the Olson specimen wall, which has been described in chapter 3. A finite element mesh of the wall and a systematic drawing is shown in figures 10.15 and 10.16.

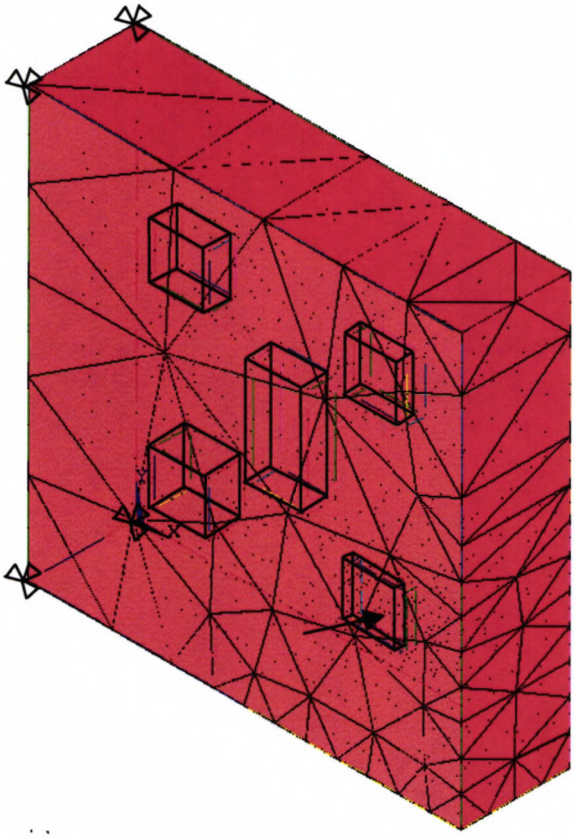


Figure 10.15: Mesh of 3D model

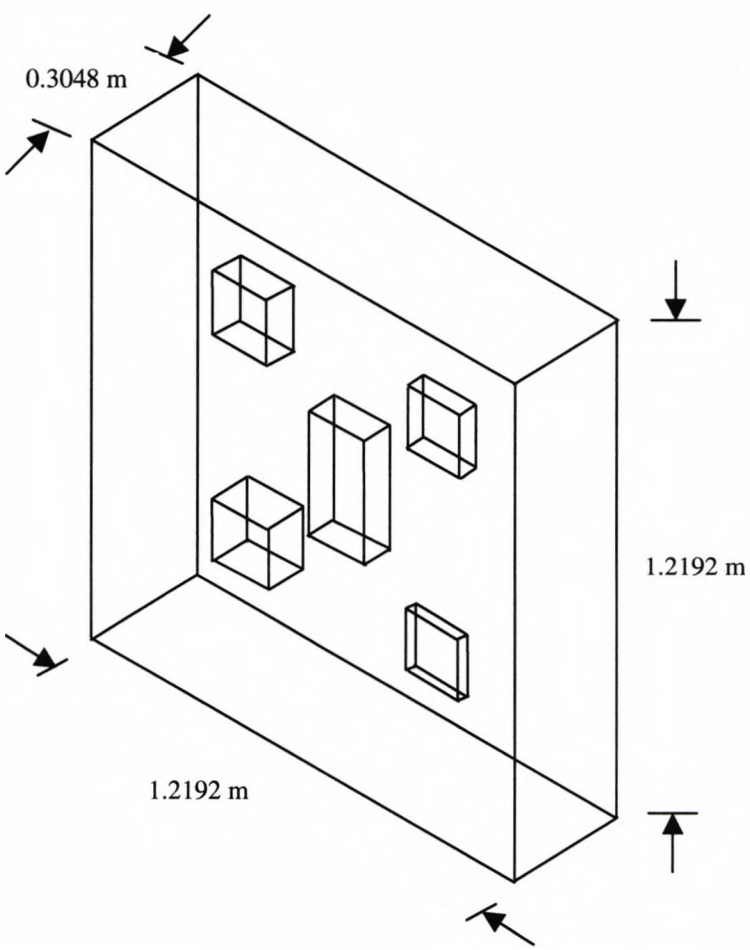


Figure 10.16: 3D model with nominal dimension (not drawn to scale)

The material properties used in the modelling are presented on table 10.6. The details for this model are:

- Number of elements = 610
- Impact time = 25 μs
- Total impact time = 10240 μs
- Force use as impact
- Distance of impact and receiver = 0.0875 m

Material	Poisson's Ratio	Density (kN/m^3)	Young's Modulus (kN/m^2)
Concrete	0.18000	2.3000	0.26000×10^8

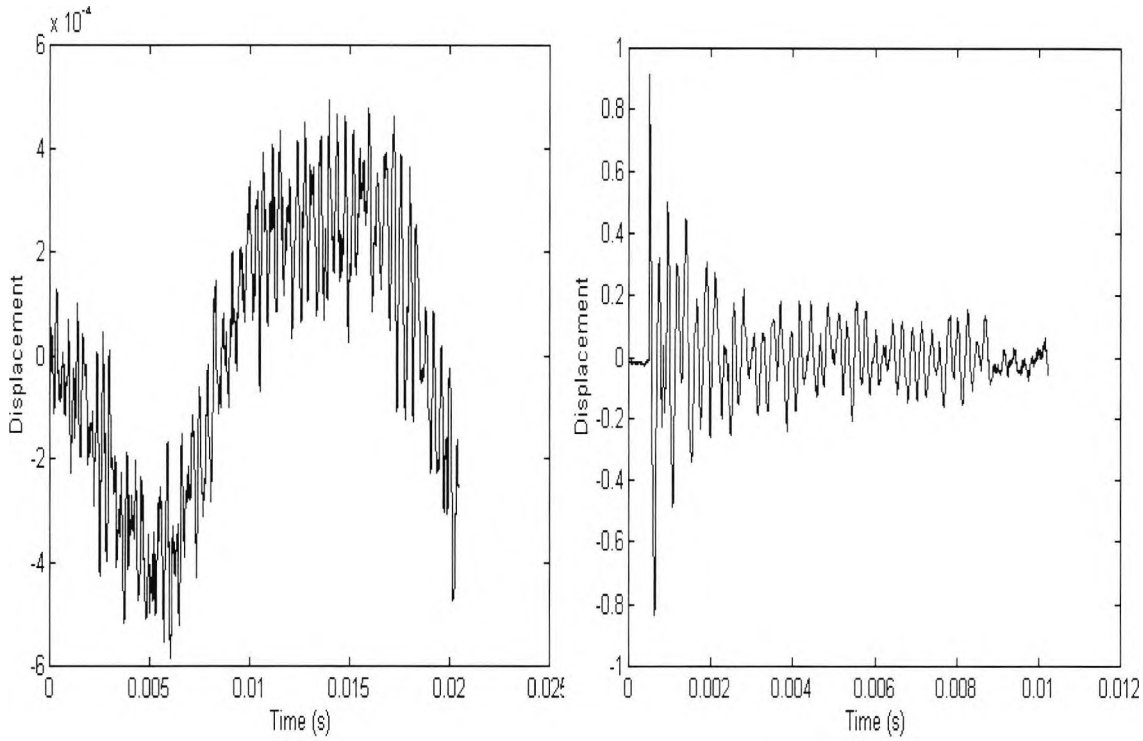
Table 10.6: Material properties 3D model

10.3.3.1. Results of 3D model of Wall A

The results for the 3D model of wall A is shown in figures 10.17 and 10.18 along with the practical results. They are presented in time and frequency domain.

The main difference between the finite element and physical results is noise. This, and effects of other voids in the vicinity, has made the impact echo response of the physical test more complicated. The finite element results show lower frequencies and a maximum frequency of about 15 kHz. This means the mesh of the model needs to be even finer, but as discussed in chapter one, the processing time is extremely slow. Also, with the 3D model, the processing time will be even longer than a week to run and obtain a solution. As explained in sections 10.2.3 and 10.2.4, the size of the element is significant, the more elements used the greater the accuracy.

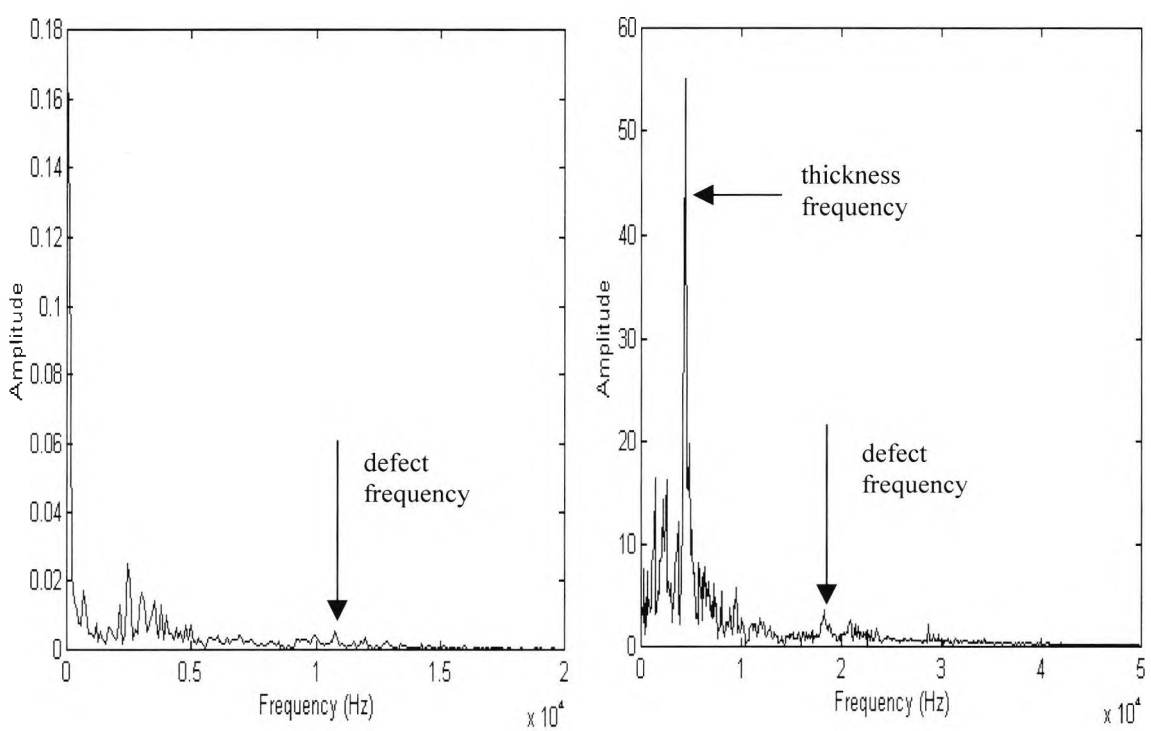
For a defect of 0.1016m depth, the frequency of interest is 17.261 kHz and for a depth plane of 0.3048 m, the thickness frequency is 5.754 kHz. In figure 10.18, the finite element result shows a peak about 11 kHz but does not show a dominant peak for the thickness frequency. In this finite element modelling, contact elements have not been used. Use of the contact elements demonstrated in section 10.3.2.1, would improve the accuracy of the thickness and defect frequency of the finite element model to the experimental results. Considering time and resource limitations, the detailed modelling of 3D impact echo systems is beyond the scope of this research. For this reason, 2D models have formed the main basis of the author's work as stated in section 1.3.1.



Finite element results

Experimental results

Figure 10.17: Waveform of 3D model in time domain



Finite element results

Experimental results

Figure 10.18: Amplitude spectrum of 3D model in frequency domain

10.4. Summary

The significance of the ITS, impact time, size of element and meshing have been dealt with.

Finite element simulation to verify Lin et al results, has been presented and shown to be in agreement. Modelling with and without contact element has been shown. Although the results appeared to be similar, the result with contact element was shown to be more realistic. The modelling of wall A of Olson specimen wall has been illustrated, the results are most encouraging, however, the resources needed for worthwhile 3D modelling are beyond the scope of this research.

Whilst the finite element approach leads inevitably to idealisation of material components and their defects, its is clearly a rapid and low cost way of obtaining a broad range of training data. The absence of noise is the main difference between this theoretical data and that derived from physical test on concrete. Use of the contact element causes such noise, the two sides of a defect then able to interact, rather than unrealistically pass through each other.

Chapter eleven

Conclusions

11.1. Introduction

In this chapter, achievement against the objectives stated in chapter 1 is discussed with recommendations for future work. For convenience, the individual objectives are repeated here, prefacing the corresponding conclusions.

11.2. Discussion

Conduct impact echo experiments to provide valid data for analysis.

This objective has been successfully achieved. The author was able to use the impact echo device, tested on 15 locations of experimental walls, and obtain valid data for the neural network study. For each location there were 15 sets of raw data, which were gathered in the time domain. Although it was possible to interpret the response of the impact in this form, it was convenient to understand the graph in the frequency domain, as discussed in section 2.3.4 of chapter 2. Hence, the FFT processing was carried out on all the raw data to achieve this, giving a total of 225 FFT graphs.

To understand the graphs, the predicted frequency was deduced for each defect depth, and depth of wall, using $T = C_p / 2 f_p$, where T is the depth, C_p is the stress wave velocity in concrete, and f_p is the frequency corresponding to the stress wave velocity. The theory is discussed in section 2.3.2 of Chapter 2. Defect depth information was provided by Olson Engineering, who produced the test walls. This was used to predict the significant frequency for each FFT graph. The repeatability of the impact echo device was examined and discussed in 4.5 of chapter 4. The FFT graphs for each location was found to be different from one graph to the next. The variation in these graphs was investigated, and the standard deviation was calculated for each location. It was found that generally the standard deviation was less than one, this is shown in table 4.2 of Chapter 4. Another factor considered was the stress wave velocity variation. Because the stress wave velocity in concrete will tend to vary over an individual wall. It was assumed that there would be $\pm 2\%$ fluctuation of this velocity. However, it was found the range of frequencies interpreted from the FFT graphs were close to

frequencies with the ± 2% stress wave velocity variation. So this variation did not affect the impact echo data. The frequencies interpreted from the impact echo experiment is shown, against the frequencies with the variation of stress wave velocity on table 4.1 of Chapter 4. The main findings is that the impact echo data is deemed to be sufficiently reliable for the purposes on training a neural network.

Establish a suitable artificial neural network for data analysis giving defect prediction.
A suitable neural network for the data analysis was established, following consideration of the neural networks, algorithms and transfer functions. The author studied the main neural networks and concluded that the back propagation network was appropriate to use. This is the most widely used and well established neural network. As discussed in chapter 5, the back propagation network architecture is suitable for the impact echo application and is good at generalisation. So, if several different input vectors are presented to the network, which corresponds to the same class, the back propagation network will learn to key off significant similarities in the input pattern, unwanted data will be ignored. This is most appropriate in the impact echo application, where there are two important frequencies present in the transient waveform of the impact echo data. These significant features are:

- A dominant frequency relating to the reflection off the bottom surface of the test object (thickness frequency).
- A dominant frequency corresponding to the reflection off the defect surface within the test object.

A typical FFT graph shown in Chapter 4, shows other frequencies which are of no interest, by using the back propagation network, these unwanted frequencies are neglected and the two significant features are known to the neural network.

Use the impact echo data to train and verify the neural network approach.
This objective was accomplished, with impact echo data used to train the back propagation network, but not so straight forward as expected. It was found that the available total data of 225 sets was in fact not enough to train the network. With 75 inputs to the neural network, an estimated 3600 data sets was needed in order to avoid over-fitting. Indeed, the network is more likely to repeat patterns that it has learnt if adequate data sets is not presented. This meant that more data had to be created to supplement the experimental data obtained from Olson Engineering in the U.S.A.

It was decided to create additional data from the existing data, by adding noise to it. This had been tried by other authors whom were successful in their objectives. There were two possibilities for achieving this, to write a simple C program to generate data, or review available software. Labview, which provides icon based programming, was used.

A study was made to decide on a level of standard deviation with added noise, so as not to alter the actual data too much, whilst still serving its purpose. This study is illustrated in section 7.10.2.1.2 of Chapter 7. Creating this data for use with the neural network was a long process, in which involved first adding noise to each set of data. Then the FFT processing was done.

The remaining data was interpolated, which consequently reduced the data. The function of the interpolation was that dominant peaks showed and smaller peaks diminished. However, to find the optimum number of data points to use, needed investigation. This is discussed in section 7.10.2.1.4 of Chapter 7.

It was observed, that there were some unwanted frequencies which could be eliminated, this would also reduce the data again, so these unwanted frequencies were filtered out.

Finally, the remaining data was accumulated. For each stage, processing was repeated 225 times. The accumulated data for neural network had to be edited into a format that the neural network could understand, this is discussed in Appendix A. Each data set was labelled so that the author knew the origin of its test case. The equivalent predicted frequency related to its depth of interest was taken as the output, the accumulated data was taken as the input, and the number of hidden nodes to use for the neural network analysis, was calculated based on the input and output nodes, this is discussed in Chapter 7. This processing had to be repeated many times until the author was satisfied with the number of data sets for the training of the neural network. The statistics of training and testing data are shown on table 11.1.

Table 11.1 shows the original statistics data set for the training and testing(out of the 225 data sets, 19 sets were lost due to author's error). The statistics of the total number of data sets after adding noise is shown in the next column. The were some 'bad' data in the sets which were of no use. This was pointed out by Olson Engineering when the

responses of the impact echo device was examined. This is where the FFT graphs show no significant peaks, so these were taken out.

	Original Data set	**Total number of data sets after addition of noise**	**Total number of data sets after eliminating 'bad' data**
Training data set	194	3833	3698
Testing data set	12	163	160

Table 11.1: Statistics of neural network training and testing data sets

Optimise the neural network performance.

This objective to optimise the neural network performance was met. The results of the neural network, after addition of more data was observed. It showed that in some cases, particularly for void defects, the result delivered by the neural network after testing were better than 90% accurate, but in other cases, the neural network produced about 50% accuracy. Figure 7.21 of Chapter 7 shows the neural network results. It was then decided to investigate whether it was possible to improve the results.

A selection of data from the original training data was taken, because the original training data file was too large to use. The neural network was trained and tested on the training data. The author expected to observe about 1% error difference between the predicted frequency given to the neural network to learn from and the frequency given by the neural network after testing. This would mean that the network had learnt well. Unfortunately, this was not the case, and there was a considerable amount of difference, this is shown in figure 7.19 of Chapter 7. The most apparent factors to change, were the learning parameters(α), momentum(β), transfer function type, and the number of hidden nodes. An investigation to find the optimum parameters was carried out. It can be concluded that to achieve the optimum results with the impact echo experimental data, the parameters presented in table 11.2 must be adopted for neural network analysis. The results with these parameters are shown in figure 7.22 of Chapter 7.

Learning Parameter for hidden nodes (α)	Learning parameter for output nodes (α)	Momentum β	Number of hidden nodes	Transfer function type for input nodes	Transfer function type for hidden nodes	Transfer function type for output nodes	Number of Iteration
0.4	0.15	0.89	61	Linear	Sigmoid	Sigmoid	50,000

Table 11.2: Optimum parameters for neural network training

The importance of the learning rate, momentum, and the number of hidden nodes is discussed in chapter 7. When the optimum of these values were found, as shown in table 11.2, the transfer function for the hidden, and output layer was changed from the default values set by the neural network software to various transfer functions. This was done to see if, further improvements could be made. It was vital to address this, because when a summation value(the sum of the inputs x the weights) coming into a processing element is beyond the processing element's function range, then that processing element becomes 'saturated'. When this happens, the saturated processing element produces an error 0 i.e. no learning can occur. The different transfer function, with its function range, is discussed in chapter 5. As explained in chapter 7, the transfer functions TanH, Sigmoid and Linear were investigated to see which would deliver an improvement, the optimum values presented in table 11.2. Results with these transfer functions are presented in Appendix D.

Another important factor investigated is the number of iterations employed during neural network training. It was found that the optimum number of iteration was 50,000 for the impact echo application. This was indicated by the root mean error (RMS), this shows a measure of the performance of a neural network training. It showed that after this number of iteration, the RMS stayed constant. The method of neural network training is explained in Appendix A.

Build a simulation of the impact echo system using finite element modelling as a means of achieving valid data sets.

Simulation of the impact echo system, using finite element modelling, was used successfully as a means of achieving additional valid data sets. A 2D, defect free, finite element model of the impact echo system was built and tested to verify work of Lin, Sansalone & Carino, 1990. The results are presented in table 10.3 of Chapter 10. This was over 90% in agreement.

A 3D finite element model based on wall A of the Olson specimens, was created and results obtained. The FFT graph from the impact echo experiment was about 18 kHz, which is close to the predicted frequency of 17.261 kHz. Results of the finite element model gave a frequency of 11 kHz. The reason for this difference is that the model needs to be meshed with more elements. As discussed in chapter 10, the calculated size of an element, based on wall A's dimension, should be 6.096×10^{-3}m by 6.096×10^{-3}m,

leading to a mesh with 10,000 elements. However, to run a model with this number of elements would be very slow. The size of element used to verify Lin, Sansalone, & Carino, 1990 study of the 2D model was 7.5 x 10^{-3}m by 7.5 x 10^{-3}m, which gave a mesh with 4800 elements(half of the plate was modelled). For this model, the author used an 8 noded element, which served as four elements to a one element from the 4 noded element, giving a total of 1260 elements. This took about 2 ½ days to run and obtain a solution. The 3D model contained 610 elements, this took 3 ½ days to process.

Also, The distance between the impactor and receiver was large for the 3D model. A typical distance of the impactor, and receiver of the impact echo equipment is in the range of 50.8 to 76.2 x 10^{-3}m (2 to 3 inches). The distance for the verification of Lin, Sansalone, & Carino, 1990, was 37.5 x 10^{-3}m, The distance between the impact and receiver for the 3D model was 87.5 x 10^{-3}m. This is a considerable difference. However, to decrease this distance means creating more elements, as discussed, using more elements would take longer to process. Another reason for the weak signal produced by the finite element model, is that the impact was applied near the boundary of the wall, giving the risk of unwanted boundary reflection. This is illustrated in Chapter 10.

Achieving this objective meant trying to find the right model which delivered a satisfactory result. Initially, the author built small models in order to understand the process. Subsequently, the models were built bigger, taking longer to run. For detailed 3D models, the time implications were unacceptable for this study. Also, due to the time limit, finite element data was not used in the neural network training.

Use a contact element to model a defect interface and understand its possible contribution towards model accuracy.

This objective was achieved, using a contact element to model a planner separation of 0.0001 m. A study was undertaken with and without this contact element in order to understand its contribution towards model accuracy. A model of 250 elements was designed and built with 8 contact elements. The results were noted, and the same model run again without the contact elements. The results of both of these models was compared, and it was observed that they were similar. However, the model with the contact element produced a nosier response, showing more peaks in the raw signal time domain than the model without contact elements. Figure 10.11 of chapter 10, shows

that after the initial gap of 0.0001m there were peaks. This is the evidence that the two surfaces of the defect were in occasional contact, hitting and rebounding.

The displacement of this finite element case with the contact element, are in the range of ± 0.001m, which could be the effect of the collision of each surface of the defect. In the frequency domain, frequencies under 5 kHz indicate a beat phenomena where the collision of the two sides of defect have produced a lower frequency, this is shown figure 10.12 in Chapter 10. The thickness frequency is indicated by a dominant peak, proving that the defect interfaces have made contact, in order for the signal produced by the impact force to pass through it to reach the bottom and travel back to the top surface. However, this thickness frequency is not indicated in the finite element model without the contact element shown in figure 10.14. For realistic finite element models of impact echo system, contact elements should be used where adjacent surfaces are likely to meet during the impact echo process.

11.3. Hypothesis

Hypothesis 1: The interpretation of impact echo data can be reliably implemented through the use of a neural network.

Considering hypothesis 1: The impact echo method is a tool for helping the engineer in deciding to make appropriate decisions for an optimum repair strategy, as discussed in chapters 1 and 2. Henriksen, 1995, states that limitations in both the method of impact echo and user's experience needs to considered for reliability. The application of the neural network to assist in interpretation is the intention of this research, and to provided analysis of large amounts of data in a fast and efficient way. By using neural network, the subjective judgement is eliminated, an important contribution. Table 11.3 show the results of neural network analysis for the impact echo data.

Using the 3698 data set, stated in table 11.1, all data related for each test case(discussed further in chapter 3) were put together and the averages of the frequencies was computed. This is shown in table 11.3 under the neural network frequency column.

Considering these results, the neural network performance is generally over 80% accurate in most cases, when compared to the predicted frequency and depth under column 2 and 3. The test cases Wacl1 and Wdcle has the maximum error of 5.696 and

5.056. This is because the impact echo test was conducted between two voids, which could have resulted in weak response. Hence hypothesis 1 has been successfully fulfilled.

(1) Cases	(2) Predicted Frequency (kHz) (b)	(3) Depth of wall (m)	(4) f_P Impact Echo Frequency (kHz)	(5) Depth correspon-ding to f_P (m)	(6) Neural Network Frequency (kHz)	(7) Depth correspon-ding to Neural Network frequency (m)	(8) Difference between predicted and neural network frequency (kHz)
Wav1e	17.261	0.1016	18.23	0.0962	17.11	0.1025	0.151
Wav2e	17.261	0.1016	16.68	0.1051	14.55	0.1205	2.711
Wav3e	17.261	0.1016	8.92	0.1966	15.81	0.1109	1.451
Wav4e	17.261	0.1016	16.87	0.1040	15.78	0.1111	1.481
Wav1w	9.864	0.1778	9.99	0.1756	9.20	0.1906	0.664
Wav2w	17.261	0.1016	15.22	0.1152	16.64	0.1054	0.621
Wav3w	13.809	0.127	12.60	0.1392	13.30	0.1319	0.509
Wav4w	11.508	0.1524	12.31	0.1425	11.47	0.1529	0.038
Wacl1	5.754	0.3048	3.68	0.4766	11.45	0.1532	5.696
Wacl2	5.754	0.3048	3.68	0.4766	7.24	0.2422	1.486
Wdv5e	17.261	0.1016	9.21	0.1904	17.00	0.1032	0.261
Wdv6e	17.261	0.1016	9.21	0.1904	17.10	0.1026	0.161
Wdv7e	17.261	0.1016	14.45	0.1214	15.64	0.1121	1.621
Wdv8e	17.261	0.1016	16.29	0.1077	14.99	0.1170	2.271
Wdcle	5.754	0.3048	5.91	0.2967	10.81	0.1622	5.056

Table 11.3: Averages of neural network analysis on impact echo data

The columns labelled 1 to 8 in Table 11.3 are:

1. test cases investigated.
2. predicted frequencies.
3. the actual depth of defects and non-defects corresponding to the predicted frequencies.
4. frequencies obtained with the impact echo experiment for defects and non-defects.
5. the corresponding depths to frequencies with the impact echo experiment.
6. neural network frequencies after training and testing the network
7. the corresponding depths to the neural network frequencies.
8. difference between the predicted frequency and neural network frequency.

Use of neural network analysis is appropriate for automatic interpretation of impact echo data, most usefully as user support, rather than a decision system.

Hypothesis 2: Finite element models, particularly incorporating contact elements, can provide a reliable means of representing impact echo systems for such purposes as neural network training.

It has been shown in chapter 10, that, by implementing a contact element, a realistic representation of the impact echo system is possible. The results with the contact

element, indicate that the two surfaces of the defect interface are colliding with each other during an impact force, which is the true characteristic of a real narrow crack type defect. The calculated predicted frequency based on the depth of the defect, was 13.33 kHz, and the amplitude graph of the finite element model with the contact element has shown a frequency of 13.23 kHz. Also, the predicted thickness frequency was 6.67 kHz, the finite element case with the contact element gave a frequency of 5.89 kHz. These results are over 85 % accurate.

The 3D finite element model representing a specimen wall tested by the impact echo equipment, discussed in Chapter 10 is about 60% accurate considering that the model had only 610 elements. As discussed previously, to improve this accuracy, more elements needs to created. Also, by implementing contact elements in the 3D finite element model will further enhance its accuracy. However, the time cost of such models, given current computing power, is likely to be prohibitive. Results of the finite element model with contact element are similar to the predicted results. The 3D finite element model(did not have contact elements) results has proved to be encouraging, so to conclude, hypothesis 2 has been proven.

11.4. Recommendations

Future improvements recommended:

1. *Undertake experiment with other neural networks to improve results, for example, Kohonen or Radial Basis Function network.*

   The impact echo application so far has been considered to be a function approximation problem, so a back propagation was selected to train and test the network. Further improvements need to be investigated, that is, re-consider the application and test Kohonen and Radial Basis Function neural networks.

2. *Investigate use of the Struttgart Neural Network software. This has the validation process incorporated, allowing determination of the correct combination of network parameters to obtain optimum results.*

   This neural network software was designed by researchers at Struttgart University. Efforts were made to use this neural network software, the main obstacle was that it only ran under UNIX, which was not convenient for the author. Also, the Fifth Annual Neural Networks Summer School, 1995 has recommended to use the neural network software by Struttgart University.

3. *Use super computers or a well equipped computer to run finite element models with fine element meshes to improve accuracy, and create more data for neural network training.*

   As shown, the ideal size of element would be 6.096 x 10^{-3} m by 6.096 x 10^{-3}m for wall A. This requires 10,000 elements to build a finite element model. If this number of elements is difficult to achieve, then a ¼ of the wall is suggested to build and test. This would take less longer to process, but attention to the size of element and distance between the impact and receiver must be considered.

4. *Investigate a range of defect model with the impact echo system to improve the neural network performance*

   To take advantage of the reliability of the neural network analysis, a variety of impact echo data would deliver a comprehensive training data set. Indeed, the more varied data presented to the network, the better the training and performance. This is the only way to prevent the network from learning unique characteristics of the training set, by providing it with plenty of different examples, so it cannot possibly learn all their idiosyncrasies.

5. *Adopt larger physical test specimens with single defects for impact echo investigations*

   In the impact echo testing, the response of a defect has shown to be weak, with other voids and boundaries affecting the signal returning to the surface. A recommendation to build larger specimens with single defects and thus achieve more distinct signals is suggested.

6. *Adopt irregular void models that represent more realistically real defects*

   Planner voids have been modelled in this research. A realistic void, in terms of dimension and angle needs to be considered. In reality, no void will be a perfectly square or circular, for example. It will be irregular and positioned in a complicated way. A similar approach should be taken when building specimens for experimental use with the impact echo equipment.

REFERENCES

Aleksander, I and Morton, H. (1990). An Introduction to Neural Computing. (London: Chapman and Hall).

Amit, D. (1989). Models of Brain Function (Cambridge University Press).

Anderson, J.A. (1972) A Simple Neural Network Generating an Interactive Memory. Mathematical Biosciences:14:197-220.

ANSYS (1993). Dynamics Manual: User's guide for revision 5.0

ANSYS (1992). User's Manual: Volume III, Elements, revision 5.0.

ANSYS (1992). Getting started with the ANSYS program, revision 5.0.

ANSYS (1992). Dynamics User's Guide for Revision 5.0, pp 1-12 - 1-26

ANSYS (1992). Theory Volume IV Manuel, 17.0 Analysis Procedures, 17.2 Transient Analysis, pp 17-5 - 17-17.

Attoh-Okine, N. 1995. Use of artificial neural network in ground penetrating radar applications in pavement evaluation and assessment. International Symposium Non-Destructive Testing in Civil Engineering (NDT-CE).1:93 – 100.

Beale R and Jackson T. (1992). Neural Computing - An Introduction. (London:IOP).

Bungey J.H. (1992). Testing concrete in structures – A guide to equipment for testing concrete in structures. (London:CIRIA).

Cameron S, Grossberg S and Guenther F.H. (1998). A self-organizing neural network architecture for navigation using optic flow, Neural Computation.10:313 – 352.

Carino N.J, Sansalone M and Hsu N.N. (1986). Flaw detection in concrete by frequency spectrum analyses of impact echo waveforms. In International advances in Non-destructive Testing, edited by W.J. McGonnagle. (New York:Gordon and Breach Science Publishers).

Carino N.J and Sansalone M. (1992) Detection voids in metal tendon ducts using impact - echo method. Materials Journal of the American Concrete Institute 89(3): 296-303.

Carling A. (1992). Introducing Neural Networks (Wilmslow:Sigma).

Carpenter G.A. (1997). Distributed learning, recognition, and prediction by ART and ARTMAP neural networks. Neural Networks. 10(8):1473 - 1494.

Caudill M. (1990). Introduction to Neural Networks. In: Neural Network Primer, edited by M. Caudill. (San Francisco: Miller Freeman).

Caudill M. (1990). Part IV: Kohonen learning. In: Neural Networks Primer, edited by M. Caudill. (San Francisco:Miller Freeman).

Caudill M. (1993). Part I: What is a Neural Network? In: Neural Networks Primer, edited by M. Caudill. (San Francisco:Miller Freeman).

Caudill M. (1992). Avoiding the great back propagation trap. In: Neural Network Special Report. (San Francisco:Miller Freeman)

Caudill M. (1993). Part 3: Back propagation networks. In: Neural Networks Primer, edited by M. Caudill. (San Francisco: Miller Freeman).

Caudill M. (1993). Part 8: Adaptive resonance network. In: Neural Networks Primer, edited by M. Caudill. (San Francisco:Miller Freeman).

Chamberlain D.A and Bleakley G.J. (1994). CURIO NDT robot for the construction industry. In: Industrial Robot. (MCB University Press).

Chamberlain D.A. (1994). A remotely operated building inspection cell, Advanced Robotics, The International Journal of the Robotics Society of Japan. 6(5).

Chamberlain D.A. (1992). A rebar detection robot. In Proceedings of 9th International Symp. On Automation and Robotics in Construction, Tokyo 1992.

Cheng C and Sansalone M (1991). The impact-echo response of concrete plates containing delaminations: numerical, experimental and field studies, Materials and Structures 26 : 274-285.

Clemena G.G. (1983). Non-destructive inspection of overlaid bridge decks with ground penetrating radar. In Transportation Research Record No. 899, Transportation Research Board, pp. 21-32.

Cooley J.W and Turkey J.W. (1965). An algorithm for the machine computation of complex fourier series. Mathematics of Computations. 19:297-301.

Coombes S and Taylor J.G. (1993). Using generalised principal component analysis to achieve associative memory in a Hopfield net, Network:5:75 - 88.

Dayhoff J. (1990). Neural networks architectures - an introduction. (U.S.A: Van Nostrand Reinhold).

Emmons P.H. (1994). Concrete Repair and Maintenance Illustrated (R.S. Means Company, inc.)

Fifth Annual Neural Networks Summer School. (1995) University of Cambridge Programme For Industry. Vol1,pp1.10,24-27th July 1995, Emmanuel College, Cambridge.

Freeman J.A. (1992). Back propagation in a neural network. In Neural Network Special Report. (San Francisco: Miller Freeman)

Fukushima K, Miyake S and Ito T. (1975). Neocognitron: a neural network model for a mechanism of visual pattern recognition, IEEE Transaction on Systems, Man, and Cybernetics. SMC – 13:826 - 834.

Fukushima K. (1989). Analysis of the process of visual pattern recognition by the Neocognitron, Neural networks. 2:413 - 420.

Glasius R, Komoda A and Gielen S.C.A.M. (1995). Neural network dynamics for path planning and obstacle avoidance, Neural Networks. 8(1):125 - 133.

Glover D.E. (1988). A hybrid of optical fourier/electronic neurocomputer machine vision inspection system. Proc.Vision 1988 Conference, sponsored by SME/MVA.

Goltsev A. (1996). An assembly neural network for texture segmentation, Neural Networks 9(4):643 - 653.

Gordon R.S. (1993). Static 1: Two Dimensional Examples. In ITTI Finite Element Training Project (UMIST).

Gorman R.P and Sejnowski T.J. (1988). Analysis of hidden units in a layered network trained to classify sonar targets, Neural Networks. 1:75 - 89.

Grossberg S. (1987) Science 11, 23, 1987.

Hammond concrete services (1992), literature

Hebb D.O. (1949). The organisation of behaviour (New York: Wiley).

Henriksen C. (1995). Impact-echo testing, Concrete International, 55-58.

Hinton G and Sejnowski T. (1983). Optimal perceptual inference Proc. IEEE Conf. on Computer Vision and Pattern Recognition , 1983 pp 448 – 453 (Washington).

Hopfield J.J. (1982). Neural networks and physical systems with emergent collective computational abilities. In Proc. of the National Academy of Sciences. 1982 79 pp.2554 – 2558 (USA).

IEEE Transactions on Neural Networks. (1997). 8(1):1 –3.

Inman D. J. (1994). Engineering Vibration. (London: Prentice-Hall International).

Jalinoos F, Olson L.D and Aouad M.F. (1993). Ultrasonic Crosshole and Crossmedium Tomography for the Detection of Defects in Structural Concrete. Olson Engineering Literature.

Kohonen T. (1982). Self-organised formulation of topologically correct feature maps, Biological Cybernetics 43:59-69.

Kohonen T. (1988) Computer No. 3, 11.

Kolsky H. (1963). Stress waves in solids. (New York: Dover publications, Inc).

Kunz J.T and Eales J.W. (1985). Remote sensing techniques applied to bridge deck evaluation. In Strength Evaluation of Existing Concrete Bridges, SP-88,American Concrete Institute, Detroit, pp. 237-258.

LeCun Y. (1985). Une procedure d'apprentissage pour reseau a seuil asymetrique Cognitiva 85 pp 599 - 604.

Lin Y, Sansalone M and Carino N.J. (1990). Finite element studies of the impact-echo response of plates containing thin layers and voids. Journal of Non-destructive Evaluation 9(1): 27-47.

Lin Y and Sansalone M. (1992). Detecting flaws in concrete beams and columns using the impact-echo method, Materials Journal of the American Concrete Institute 89(4):394-405.

Liu W and Meng Z. (1997). Study on the model and algorithm of backpropgation network, 2nd International Symposium on Test and Measurement, pp. 291 - 293.

Lockhart G.B, and Cheetham B.M.G. (1989). BASIC Digital Signal Processing (Butterworth).

Lu S and Szeto A. (1993). Hierarchical artificial neural networks for edge enhancement, Pattern recognition. 26(8):1149 - 1163.

Manning D.G. (1985). Detecting defects and deterioration in highway structure. In NCHRP Synthesis No. 118, Transportation Research Board, pp 52.

Maren A, Harston C and Pap R. (1990). Handbook of neural computing applications. (London: Academic Press Ltd).

Martin J, Hardy M.S.A, Usmani A.S, and Forde M.C. (1997). Impact-echo assessment of post-tensioned concrete bridge beams. In Proceedings of the 7th International Conference on Structural Faults and Repair-97, 341-353, Edinburgh, Scotland.

Masters T. (1993). Practical neural network recipes in C++. (London: Academic Press Ltd).

McCulloch W.S. and Pitts W. (1943). A logical calculus of the ideas immanent in nervous activity, Bulletin of Mathematical Biophysics 5:115 - 133.

Mead C. (1989). Analog VLSI and neural systems. (Addison-Wesley Publishing Co)

Miao X, Azimi-Sadjadi M.R, Tian B, Dubey A.C, and Witherspoon N.H. (1998). Detection of mines and minelike targets using principal component and neural network methods, IEEE Transaction on Neural Networks, 9(3): 454 – 463.

Neural Computing Applications Forum (NCAF), August 1999.

Nath B. (1974). Fundamentals of Finite Elements for Engineers. (London:Athlone Press).

National Science Foundation Phase I final Report. (1993). Olson Literature.

NCHRP Report 118, Detecting defects and deterioration in highway structures, TRB. 1985, p.13.

Neural Computing (1991) Neuralworks Professional II/PLUS and NeuralWorks Explorer, NeuralWare, pp NC-235 - NC-240. NeuralWare, Inc.

Neural Computing (1991) Neuralworks Professional II/PLUS and NeuralWorks Explorer, NeuralWare, pp NC-165 - NC-168. NeuralWare, Inc.

Neural Computing (1991) Neuralworks Professional II/PLUS and NeuralWorks Explorer, NeuralWare, pp NC-43 - NC-47. NeuralWare, Inc.

Neural Computing (1991), Neuralworks Professional II/PLUS and NeuralWorks Explorer, p. NC-39. NeuralWare, Inc.

Olson Engineering, Inc. (1995) Literature.

Park D.C. (1997). Development of a neural network algorithm for unsupervised competitive learning, IEEE International Conference on Neural Networks 3:1989 - 1993.

Parker D.B. (1985). Learning logic Technical Report TR-47 Center for Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, MA.

Pourboghrat F. (1990). Neural networks application in autonomous path generation for mobile robots, Society of Photooptical Instrumental Engineers, Vol. 1396, pp. 243 - 251, Applications of Optical Engineering: Proceedings of OE/Midwest 1990.

Pratt D and Sansalone M. (1991). The use of a neural network for automating impact-echo signal interpretation. Review of Progress in Quantitative Non-destructive Evaluation 10A: 667-674.

Rae R and Ritter H.J. (1998). Recognition of human head orientation based on artificial neural networks, IEEE Transaction on Neural Networks, 9(2):257 – 265.

Rao D.H, and Gupta M.M. (1994). Co-ordination and control of non-linear multivariable systems with parametric and structural uncertainties using a dynamic neural network. In Uncertainty Modelling and Analysis: Theory and Applications (Elsevier Science B.V.).

Reid M.B, Spirkovska L and Ochoa E. (1989). Simultaneous position, scale, and rotation invariant pattern classification using third-order neural networks, Neural Networks, 1(3):154 - 159.

Ritter H, Martinetz T and Schulten K. (1991). Neural computation and self-organising maps.

Rockey K.C, Evans H.R, Griffiths D.W, and Nethercot D.A. (1995). The Finite Element Method: A Basic Introduction (Collins).

Rosenblatt F. (1958). The Perceptron: A probabilistic model for information storage and organization in the brain, Psychological. Review, 65:386 - 408.

Rumelhart D.E and McClelland J.L. (1986). Parallel Distributed Processing: Exploration in the Microstructure of Cognition, Vols. 1 and 2 (MIT Press, Cambridge, Mass., 1986).

Rumelhart D.E, Hinton G,E and Williams R.J. (1986). Learning internal representations by error propagation. In Parallel Distributed Processing, edited by Rumelhart, D.E., McClelland, J.L. and the PDP Research Group (Eds.) Chapter 8, 318 - 364, 1986.

Sansalone M and Carino N.J. (1986). Impact echo: a method for flaw detection in concrete using transient stress waves. In NBSIR 86-3452 (NIST PB 87-104444/AS), National Bureau of Standards, Gaithersburg, Maryland, pp222.

Sansalone M and Carino N.J and Nelson H.N. (1987) A finite element study of transient wave propagation in plates, Journal of Research of the National Bureau of Standards 2: 267-277.

Sansalone M, and Carino N.J. (1989). Detecting delaminations in reinforced concrete slabs with and without Asphalt concrete overlays using the impact-echo method. Materials Journal of the American Concrete Institute March-April:175-178.

Sansalone M and Carino N.J. (1990). Finite element studies of the impact-echo response of layered plates containing flaws, International Advances in Non-destructive Testing:313-336 (New York: Gordon & Breach Science).

Sansalone M, and Street W.B. (1998) Impact Echo, Nondestructive Evaluation of Concrete and Masonry. (Bullbrier Press)

Shubnikov E.I. (1997). The main models of neural networks, Journal of Optical Technology, 64:989 - 1003.

Stain R.T. (1982) Integrity Testing. Civil Engineering

Taylor J.G. (1997). The historical background, Handbook of Neural Computation, (IOP Publishing Ltd and Oxford University Press).

The Student Edition of MATLAB: Version 4, User's Guide. (London: Prentice-Hall International).

Tolat V.V. and Widrow G. (1988). An adaptive broom balancer with visual inputs, IEEE International Conference on Neural Networks, 2:641-647.

Using Neural Network (1993) - A Tutorial for Neuralworks, Professional 11/Plus and Neuralworks Explorer, Neuralware, Inc.

Waibel A. (1988). Consonant recognition by modular construction of large phonemic time-delay neural networks, In Neural Information Processing Systems, San Mateo, California, pp. 215-223. (Kaufman Publishers).

Werbos P.J. (1974). Beyond Regression: New Tools for Prediction and Analysis in the Behavioural Sciences, Ph.D. dissertation in statistics, Harvard University.

White D.A and Sofge D.A. (1992). Handbook of intelligent control, (New York: Van Nostrand Reinhold).

Widrow B and Hoff M.E. (1960). Adative Switching Circuits, Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, Part 4, 96 - 104, 1960.

Widrow B and Smith F.W. (1963). Pattern recognizing control systems. In Computer and Informations Sciences Symposium Proceedings,Washington, DC, 1963. (Spartan Books).

Yiching L and Sansalone M. (1992). Detecting flaws in concrete beams and columns using impact echo method, ACI Materials Journal/ July – August 1992, p395.

Yu X.H, and Chen G.A. (1997). Efficient back propagation learning using optimal learning rate and momentum, Neural Networks, 10(3):517 - 527.

Yuceer C and Oflazer K. (1993). A rotation, scaling, and translation invariant pattern classification system, Pattern Recognition, 26(5):687 - 710.

Zienkiewicz O.C. (1971). The Finite Element Method in Engineering Science (McGraw-Hill).

Appendix A

Neural network method

A.1. Introduction

This appendix illustrates the technique of the neural network method on the data obtained by the impact echo (see appendix C). An example of neural network analysis using the back propagation network is presented.

A.2. Neural network professional II/plus

Artificial neural networks have been recognised as powerful and economical tools for solving a large variety of problems in a number of engineering disciplines. There are a number of commercial software and hardware available. The software employed for this research is the DOS version of NeuralWorks Professional II/PLUS created by NeuralWare.

A2.1. Format of input data/output data

The input data can be presented from the following way :

- Keyboard
- An ASCII or binary file
- User-written program

A2.1.1. ASCII file format

For this research, data was presented from an ASCII file format. An ASCII input data files contain data in row and column format. Each row contains the inputs and desired outputs (optional) for one example. So, if there were 4 inputs, and 3 possible outputs, then, there would be 7 numbers for each logical row. Each number would be separated from the others with at least one space or a comma.

A logical row can have many physical rows. The continuation character is the ampersand (&) and must begin in column 1 of the continuation line. Also, ASCII data files can have comments. A comment line starts with an exclamation mark (!). This can

be added at the end of a line as well at the beginning. An example of this is shown below in figure A1 below.

```
! This is an example of a comment line
0.4 0.3 0.5 0.4        !This is a comment for a 4 field record
0.7 0.1 0.9            !This is part of an input with a comment
&0.8                   !and here is the continuation line
```

Figure A1: An example of an ASCII data file

A2.1.2. Files

The important files with the following extensions which are most frequently used are :

- ASCII input data files have an extension .nna
- The .nnd extension is assumed for all network types (this is saved as in binary form as this takes least amount of room, and contain all the information required to recreate a network)
- Result files have an .nnr extension

A.3. Example program

An example of flower classification has been used to demonstrate the use of the neural network software. The objective is to classify iris species into three categories which are:

- Setosa
- Versicolor
- Virginica

These are the three possible desired outputs for the network, which would be represented by 0 or 1. If the iris is a Versicolor, the desired outputs presented to the network would be 0 1 0, for example. The criteria the network uses for classifying the species are based on the length and width of the sepals and petals. These are the four inputs to the network and they will appear as four analogue numbers.

The data for the flower classification back propagation network has been divided into a training data set, which will be used for learning, and a test set which will be used to test the data to see how well it has learnt the data.

The neural network software is started by being in nw2v504 working directory e.g. c:\nw2v504, and then typing nw2x from MS-DOS, or it can be entered from windows.

A.4. Building the back propagation network

The back propagation network is created by selecting :

- InstaNet menu
- The Back Propgation command

This is shown in figure A2

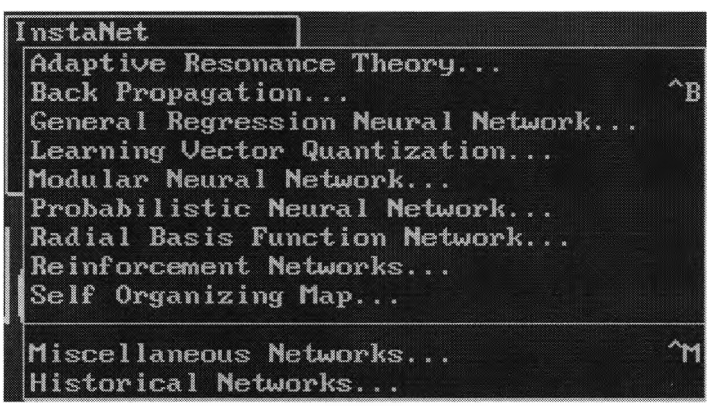


Figure A2: Selection of a network

Next appears the Back Propagation dialogue box shown in figure below, this gives a complete tool kit for building back propagation networks. For the following values for the number of processing elements in each layer in the relevant text entry boxes must be filled for example :

4	Input
3	Hid
0	Hid
0	Hid3
3	Output

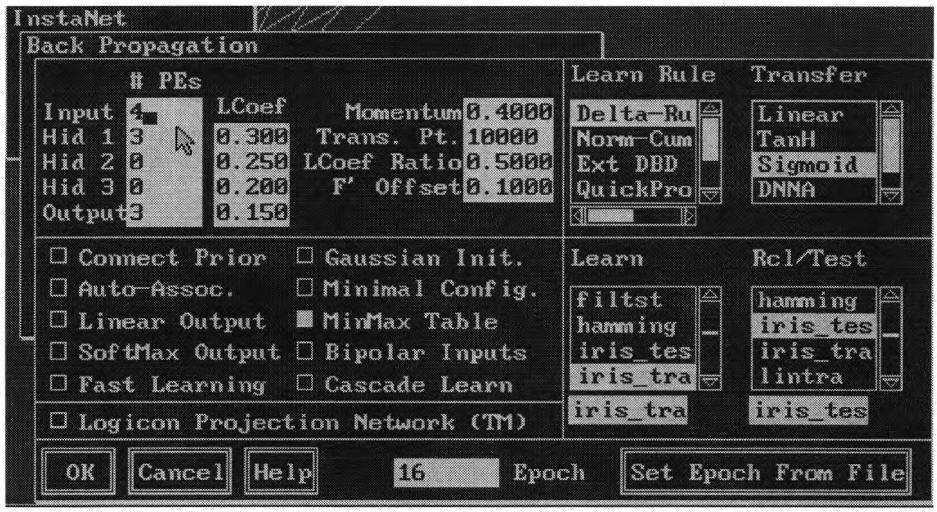


Figure A3: Selection of appropriate parameters from the back propagation dialogue box

Also, select the following :

- Delta rule
- Sigmoid transfer function
- File for learning and testing which is Iris_tra.nna and Iris_tes.nna
- Deselect the Bipolar inputs
- Leave all other values as default
- This is shown in figure A3

Once this has been completed, the instrument menu for the back propagation command will appear as shown in figure A4.

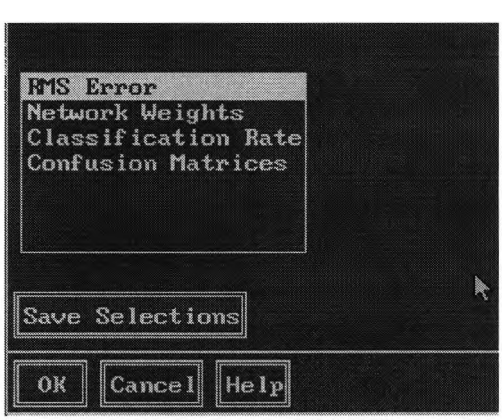


Figure A4: Selection of instruments to monitor progress

This instrument plots the RMS error of the output layer. It is the root mean square error which adds up the squares of the errors for each processing element in the output layer, divides by the number of processing element in the output layer to get an average, the square root of that average is taken.

The RMS is very useful and gives an indication of the performance of a network during training. As learning progresses , the graph for this network usually converges to an error near zero. NeuralWorks builds a back propagation network which will look as below in figure A5. Considering the figure in A5, the input layer is at the bottom of the screen, the output layer is at the top, and the hidden layer is between the input and output layers. The layers named along the right hand side of the network.

Each box in a layer is a processing element. Its colour, size and fill of the processing element show their output values. These can be set through the Utilities/Display Style command. Also, the dotted line and solid lines between processing elements are the connections. The line whether it is solid, dotted, and the colour is an indication of a connection's strength.

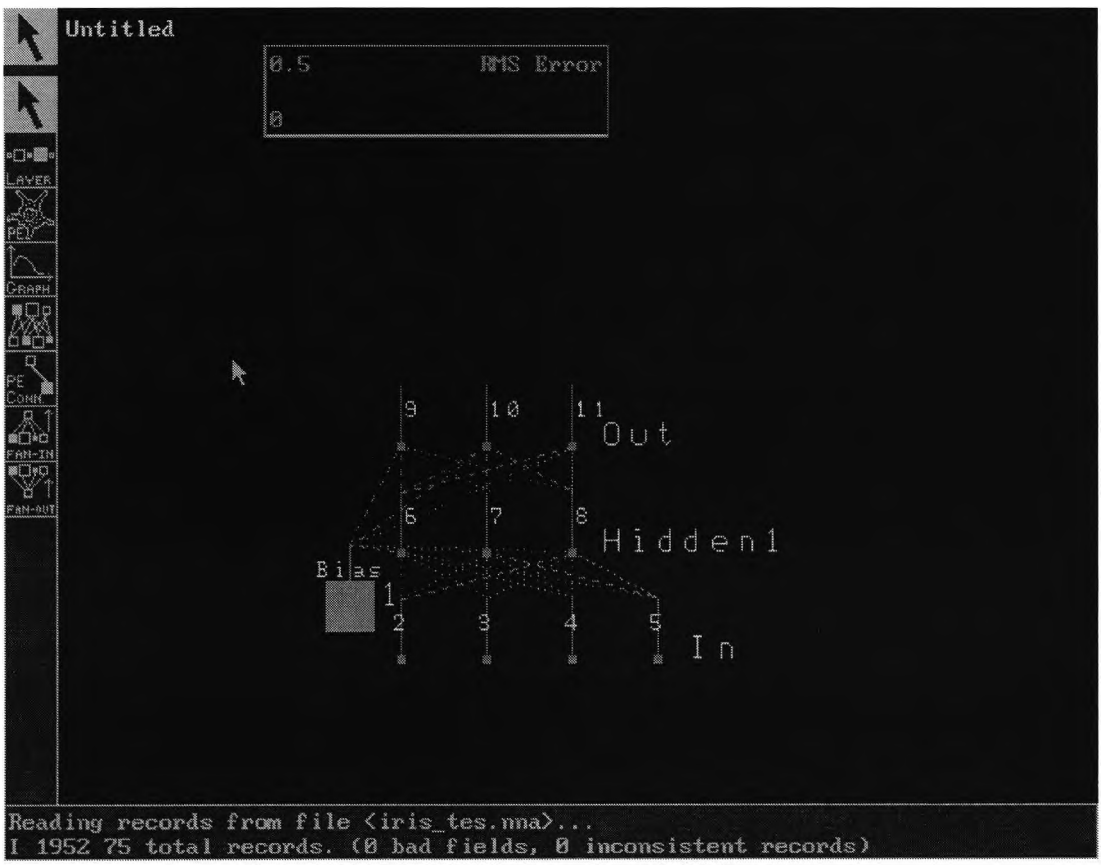


Figure A5: Example network with its instrument

A.5. Input and output parameters

The Parameters dialogue box contains all the parameters that affect the input files. This can be seen in the figure A6, which is found through the I/O menu. The parameters can be viewed by selecting the Parameters command in the I/O menu. Under the Learn Source (labelled Lrn), in the upper left hand corner of the dialogue box, the File Rand. radio button is selected.

NeuralWare claim that by selecting File Rand. option which tells NeuralWorks that the examples will be presented to the network in a random way, random presentation of training data helps the network to prevent local minimum. The binary Load to RAM check box for Lrn must be deselected (this is the option that converts the input file to a binary file). Also, under the Rcl/Test heading, in the upper left of the dialogue box, the File seq. radio button must be selected.

Selection of the training file must appear under the Learn heading on the right side of the dialogue box. The default extension is nna. Also, the selected test file must appear under Rcl/Test heading on the right side of the dialogue box. The default extension is again nna.

The radio buttons Input and Desired O/P under the heading Result File is selected. This ensures that the result file will have the desired values in the input file along with the new values obtained by the network. This makes it easier to compare and see the error differences. Also, when using an ASCII recall file, the network output will be written to an ASCII results file with the same name as the Rcl/test file, but with an extension of nnr.

These selections are recorded by selecting OK on the Record dialogue box.

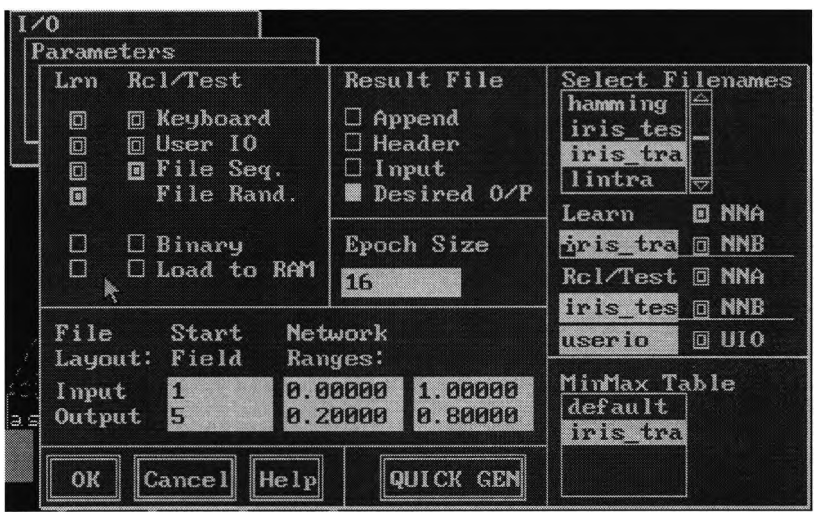


Figure A6: Input and output parameters

A5.1. Min/max tables

Raw values can not be presented to the network, the back propagation network uses sigmoid or hyperbolic transfer functions which respond in a linear fashion to summation between about -2 to +2 . If a user presents a back propagation network with input values such as 10,000, even with small weights in the network, the summation will be huge and the sigmoid will become saturated.

When this happens, the derivative of the sigmoid or hyperbolic tangent is close to zero at large (positive or negative) summation values, and because the derivative is a multiplier in the weight update equation, learning stops for processing elements with big summation values.

Figure A7: Selecting the layer/LRS tool

This problem is overcome by the NeuralWorks software by using a pre-processing facility which computes the lows and highs of each data field for all the input data files to be used with a given network. These lows and highs are stored in a table called a MinMax table.

When the MinMax table check box was selected along with the Delta rule for this particular problem shown in figure A3, NeuralWorks automatically created a minmax table from the data files set the network input and output ranges accordingly. for example in figure A3. The Network Ranges is shown in this figure, according to the NeuralWorks, the selection of Network Ranges is directly connected :

- The number of inputs
- The type of transfer function in operation
- The initial weight values in the network

Indeed, the basic idea is to find a range that gives summation which will not initially saturate the transfer function. The pre-processing facility can be found without the Back Propagation command through two dialogue boxes :

- The I/O/Parameters dialogue box to set network ranges.
- The I/O/MinMax tables dialogue box to build and edit MinMax tables.

A.6. Changing learning parameters and momentum

It is important to remember that different learning coefficients for each layer in a multi-layer network can improve the learning time considerably. So far the default values have been taken along with the learning rates and momentum terms for each of the layers.

This is done by using the tool palette to build new schedules assigned to the output and hidden layers. To create a schedule for the output layer the object palette on the left is clicked and held. This is shown in figure A7

From this menu, the LRS tool is selected. A message will prompt the user to click on the layer that will be linked with the L/R Schedule. Changes can be made to the L/R Schedule dialogue by clicking on any processing element in the output. This is shown in figure A8.

From figure A8, out is the name assigned to this schedule. In column one the learn count is 10000, this means that the Coefficients in column one will be valid until the Learn Count increases beyond 10000. Column 2 shows that it is 30000, this means that when the learn count is greater than the value specified in column one, but not greater than the count in column two, the coefficients in column two will be used. This technique also applies to the changes to Momentum and Learning Rate.

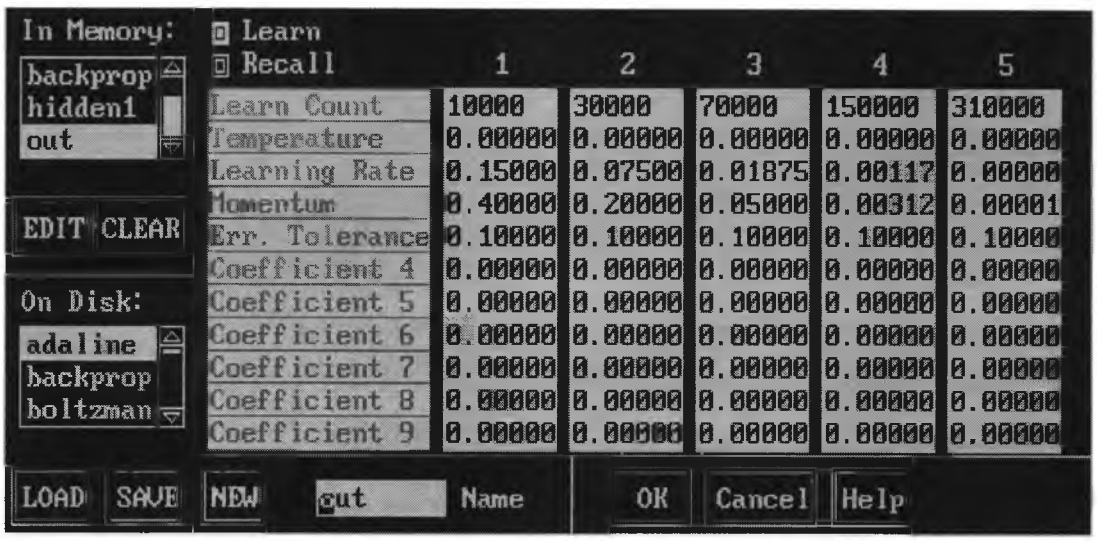


Figure A8: The LRS tool dialogue box

A.7. Training the network

To start the training Learn is selected from the Run menu, For from the radio button is highlighted, then a figure of 50,000 is selected for iteration. This informs NeuralWorks to present 50,000 training examples to the network in the Learn mode. The training can be interrupted with the escape key. Figure A10 below, shows the network after it has been trained. A counter in the message box indicates the number of examples that have presented to the network.

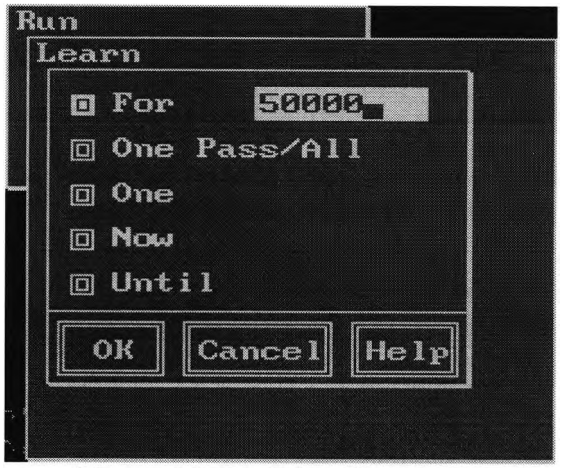


Figure A9: Training the network

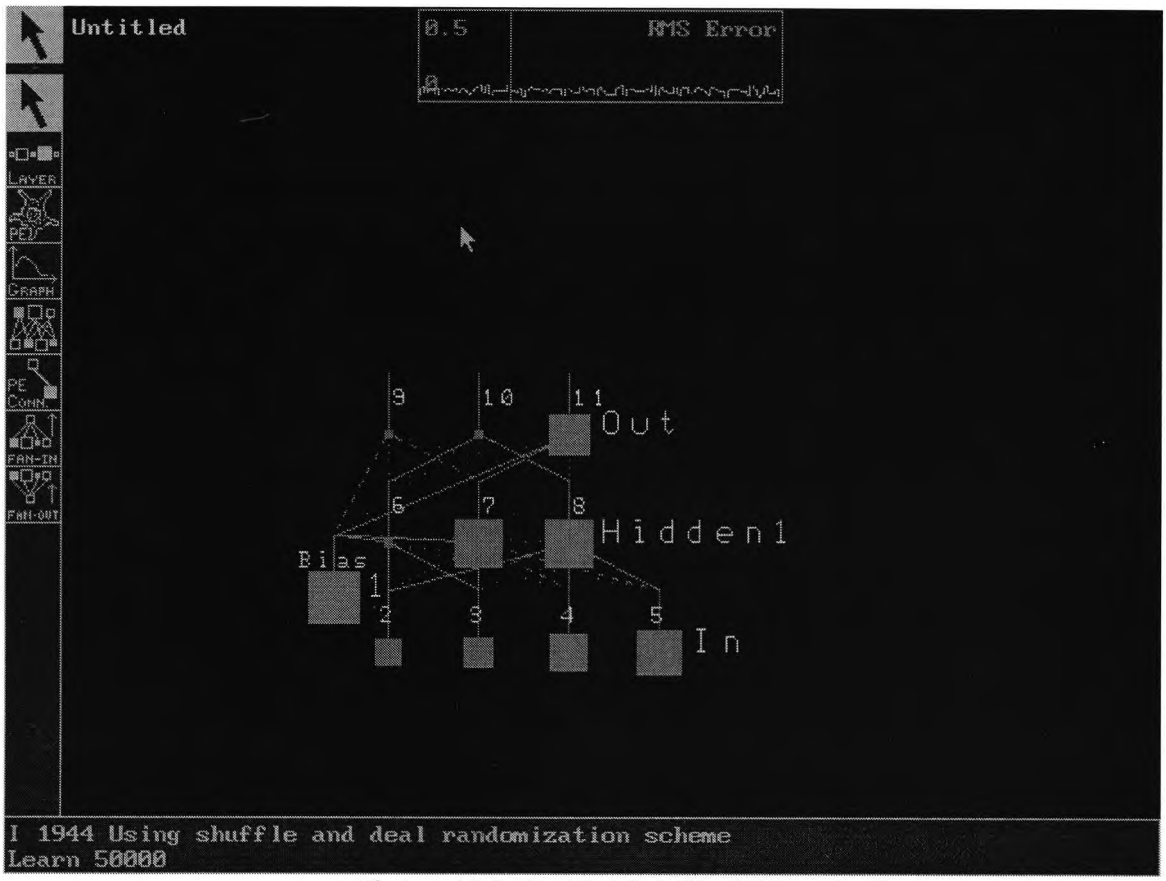


Figure A10: A trained network

A.8. Testing the network

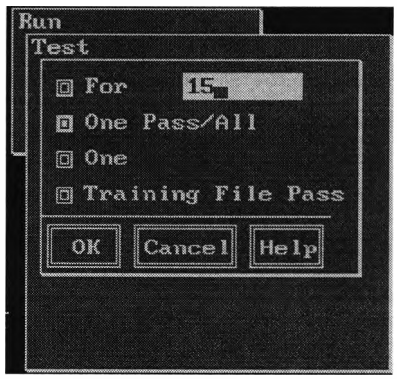


Figure A11: Testing the network

Now that the neural network is trained, it is ready to be tested , this is done by selecting the Test command from the Run menu, then clicking One Pass/All in the dialogue box. This is shown in figure A10. The software will now read the whole test file and present all the examples to the trained network. The number of examples given will be periodically shown in the message box. The desired outputs and the actual network results will be written to the result file.

A.9. Saving the network

The network can be saved in two ways :

- Save - This is used to save as to update a network file which is previously saved.
- Save as - This is used to initially save a network, or save the current version of the network to a new file.

The network is saved in binary format because it takes the least amount of room, and contains all the information needed to recreate that particular network. To save the network, the Save as is selected from the File menu as shown in figure A12.

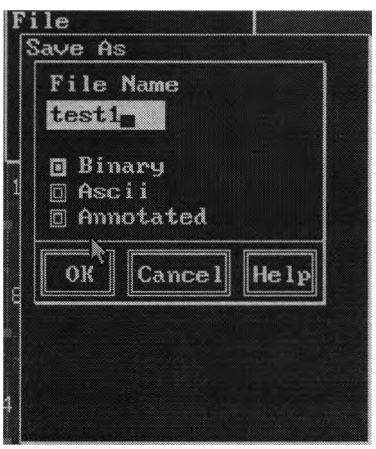


Figure A12: Saving the network

A.10. Results

The results of this example is shown in table A.2, the first three columns are the actual data set by the user, and the last three columns are the results the neural network has achieved.

Actual			Neural Network		
1	0	0	0.971079	0.041252	-0.0119
0	1	0	-0.00839	0.988939	0.014991
0	0	1	0.000101	0.02864	0.968763
1	0	0	0.989782	0.013271	-0.00397
0	1	0	-0.01141	0.942758	0.06781
0	0	1	-0.00164	0.084684	0.914046

Appendix A

Actual			Neural Network		
1	0	0	0.994701	-0.00281	0.005613
0	1	0	-0.00086	0.939724	0.057019
0	0	1	0.001091	-0.05469	1.051583
1	0	0	0.998421	-0.00149	0.001513
0	1	0	0.092834	1.032084	-0.10466
0	0	1	-0.00437	0.075363	0.926866
1	0	0	0.9891	0.016078	-0.00575
0	1	0	0.016086	0.882588	0.09395
0	0	1	0.006952	-0.0908	1.082713
1	0	0	1.002406	-0.00728	0.003309
0	1	0	-0.00083	0.930663	0.065463
0	0	1	-0.00366	0.067931	0.933707
1	0	0	1.01212	-0.0132	0.000556
0	1	0	-0.00998	0.900726	0.106769
0	0	1	-0.00028	-0.02864	1.027182
1	0	0	1.021731	-0.03966	0.016712
0	1	0	-0.01448	1.079745	-0.07141
0	0	1	0.004884	-0.05476	1.047802
1	0	0	0.996783	-0.00086	0.002282
0	1	0	0.018485	1.101882	-0.11756
0	0	1	0.004626	-0.03198	1.024319
1	0	0	1.012904	-0.02478	0.01025
0	1	0	0.008313	1.083525	-0.09231
0	0	1	-0.01612	0.360357	0.662949
1	0	0	0.997804	-0.00524	0.005241
0	1	0	-0.00463	1.078111	-0.07748
0	0	1	0.003321	-0.00238	0.996487
1	0	0	0.920628	0.099936	-0.01883
0	1	0	-0.01048	1.026066	-0.01958
0	0	1	-0.00758	0.252475	0.754813
1	0	0	0.952528	0.065964	-0.01685
0	1	0	-0.01564	1.062202	-0.05214
0	0	1	-0.00524	0.213682	0.789001
1	0	0	0.998639	-0.0026	0.00228
0	1	0	-0.01256	0.595119	0.419053
0	0	1	-0.00687	0.721476	0.282874
1	0	0	0.990862	0.010255	-0.00234
0	1	0	0.043362	1.104204	-0.13524
0	0	1	-0.01385	0.516514	0.500303
1	0	0	0.950913	0.060219	-0.01158
0	1	0	0.020975	1.100225	-0.11702
0	0	1	-0.0026	0.196634	0.801442
1	0	0	1.029562	-0.04778	0.017766
0	1	0	-0.00751	0.353696	0.653711
0	0	1	-0.01303	0.639155	0.376103
1	0	0	0.993882	0.007982	-0.00285
0	1	0	0.005414	0.864837	0.122388
0	0	1	0.001422	-0.0597	1.056324
1	0	0	1.022295	-0.03395	0.010786
0	1	0	-0.02774	0.998931	0.028656
0	0	1	-0.00219	0.180579	0.818028
1	0	0	0.994787	0.003461	0.000204
0	1	0	-0.00251	0.9711	0.028416
0	0	1	-0.00079	0.055089	0.942444
1	0	0	0.871473	0.196253	-0.04758

0	1	0	-0.00697	0.944916	0.058266
0	0	1	0.001238	-0.00338	0.999227
1	0	0	0.945115	0.063573	-0.00989
0	1	0	0.063685	1.061741	-0.11044
0	0	1	0.004827	-0.07783	1.071469
1	0	0	0.962232	0.051651	-0.01327
0	1	0	0.024773	1.046638	-0.0711
0	0	1	0.00223	-0.04171	1.037213
1	0	0	1.001637	-0.00364	0.000798
0	1	0	-0.00742	1.062809	-0.05992
0	0	1	-0.00125	0.083794	0.914247
1	0	0	0.994474	0.005333	-0.00111
0	1	0	0.005749	1.017628	-0.02673
0	0	1	0.000509	0.171585	0.823581

Table A.2: Neural network results

A.11. Summary

The use of the Proff 11/Plus neural network software was demonstrated by a flower classification problem. Building a back propagation network with min/max tables has been shown. Learning and testing a network has been presented.

Appendix B

FINITE ELEMENT ANALYSIS USING ANSYS

Preparation of data and Interpretation of results

B.1. Introduction

The purpose of this appendix is to illustrate the preparation of input for finite element analysis and illustrate the processing of the results following finite element analysis. In the former we have to consider the preparation of the mesh, and the means of representing the impact pulse. The purpose of the latter is to obtain a spectrum analysis of the reflected signal, the time-displacement response of the receiver node.

B.2. Finite element software

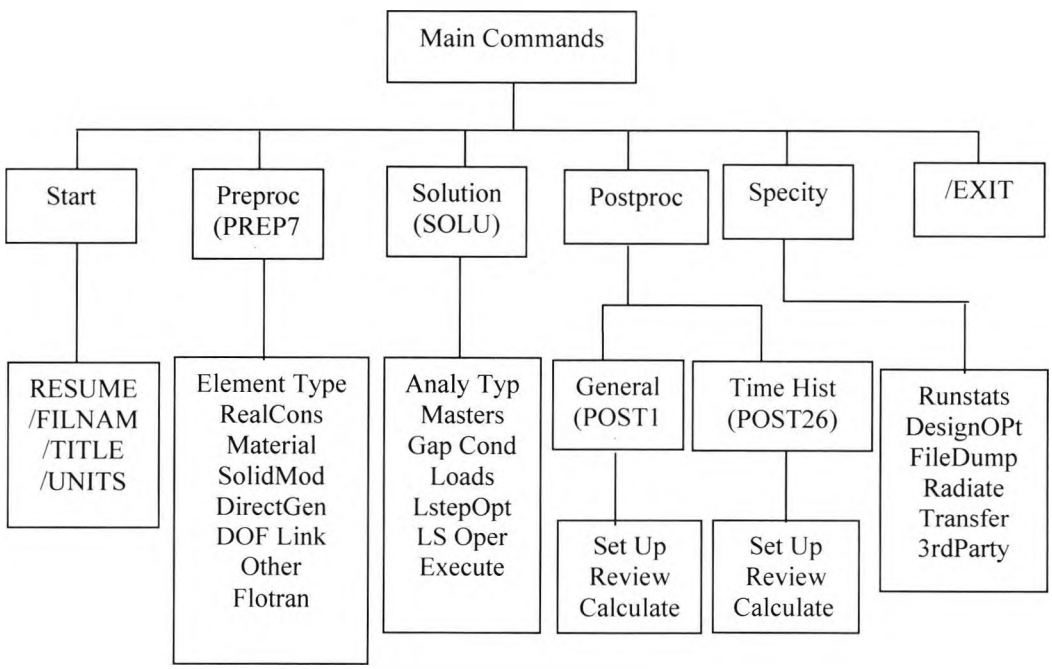


Figure B1: Structure of ANSYS

In the finite element modelling, ANSYS was the software package used. The general structure of ANSYS is shown in figure B1. The three main stages are the pre-processing, the solution, and the post-processing stage. In the next section a simple example is used to illustrate the finite element package.

B2.1. Data input

There are two ways to import the data and commands which are to either to use the interactive mode or the batch mode. Both methods must be entered at one stage. This is done by clicking on the 'Run Interactive Now' from the ANSYS menu in windows. Once the program has been entered, the layout is shown in figure B2.

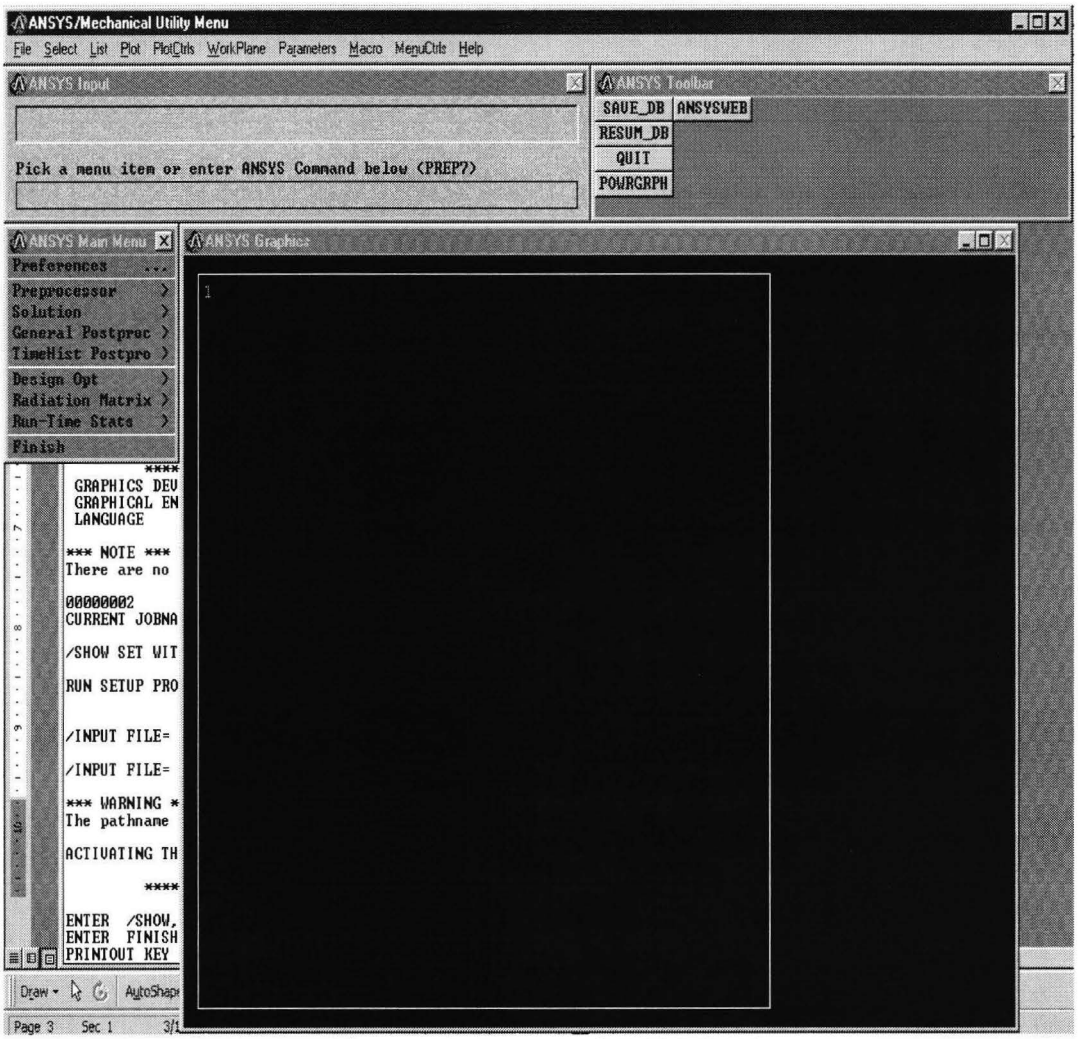


Figure B2: Layout of ANSYS 5.4

The layout contains five main parts which are:

- ANSYS/Mechanical Utility Menu
- Input
- Main menu
- Graphics
- Toolbox

B2.1.1. Interactive mode

The commands in the interactive mode are written in the ANSYS program by use of the mouse, or keyboard to go from one command in the menu bars to another. Each command entered is written to a default file(file.log), which can be specified.

B2.1.2. Batch mode

In the batch mode, a file in the format of the file.log is written in note pad or MS-DOS editor. The complete file can be resumed into the ANSYS program.

B.3. Example

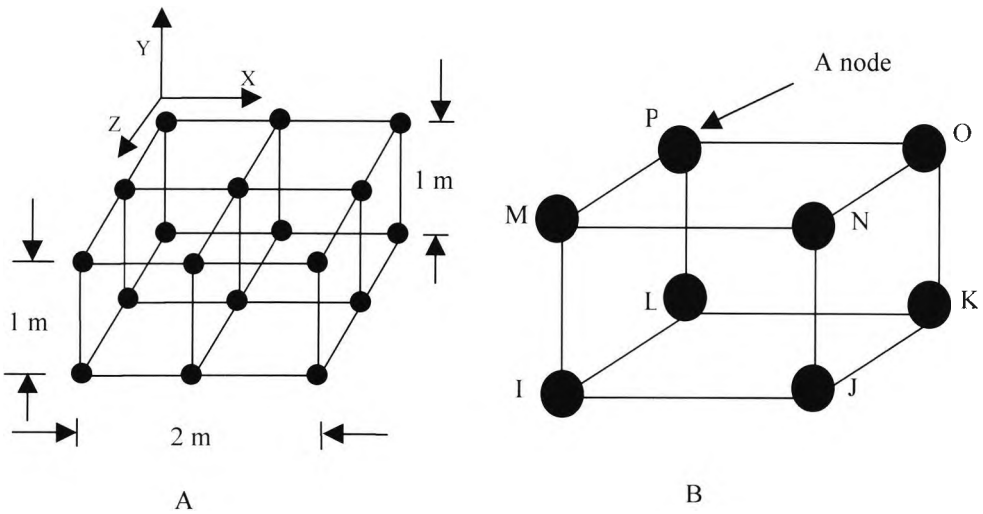


Figure B3: Example model

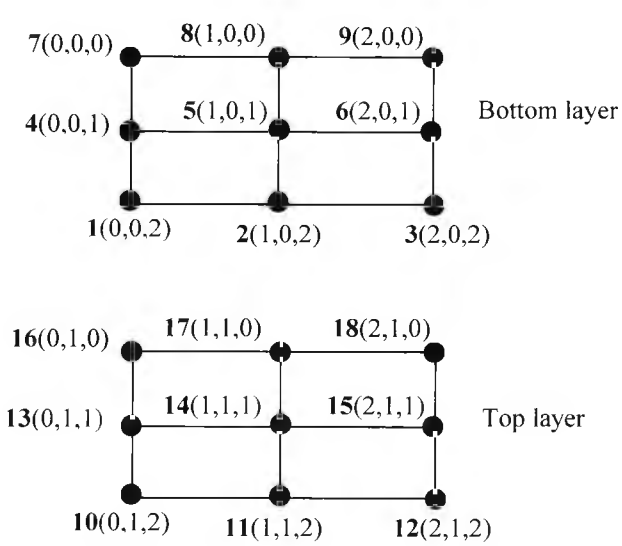


Figure B4: Layers

An example model of 2m x 1m x 1m shown in figure B3 (A). There are 4 elements, each having 8 nodes, as shown in figure B3 (B). These nodes must be defined in the order of I, J, K, L, M, N, O, and P. At each node there are 3 DOF, the translations Ux, Uy and Uz.

With the software, there are two types of modelling, solid and direct. Solid modelling is automatic generation, where the shape to be modelled is defined, the software building the model. The second method needs the user to build the model. In this example, direct modelling is used. For this method, it is ideal to plot the nodes and elements on paper making it clearer to visualise. Figure B4 shows the top and bottom layer with their nodes and co-ordinates.

B.4. Pre-processing

The pre-processing stage is called Prep7. At this stage of the program, the model is constructed. This begins with defining the element to be used in the modelling. Figure B5 shows solid 45, a 3D brick element selected from the library elements. Element type for this example is 1, which identifies the element type. Next, the material properties for concrete is chosen which is shown in figure B6. Once this is completed, the nodes need to be defined, referring to figure B4, node 7 is the first node defined with co-ordinates (0,0,0). This is shown in figure B7.

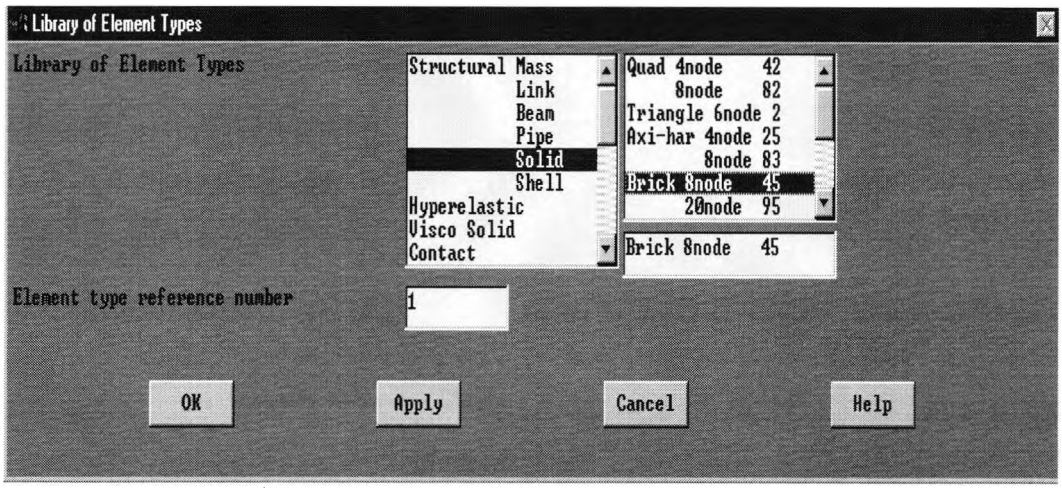


Figure B5: Defining the type of element

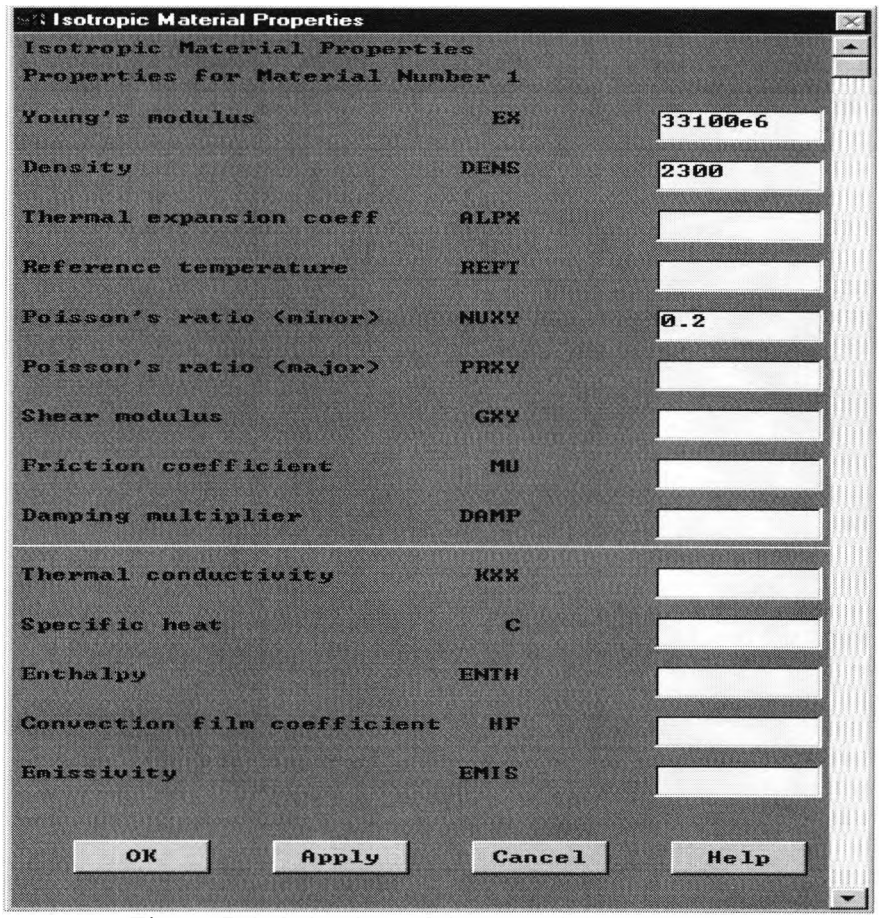


Figure B6: Selection of the material properties

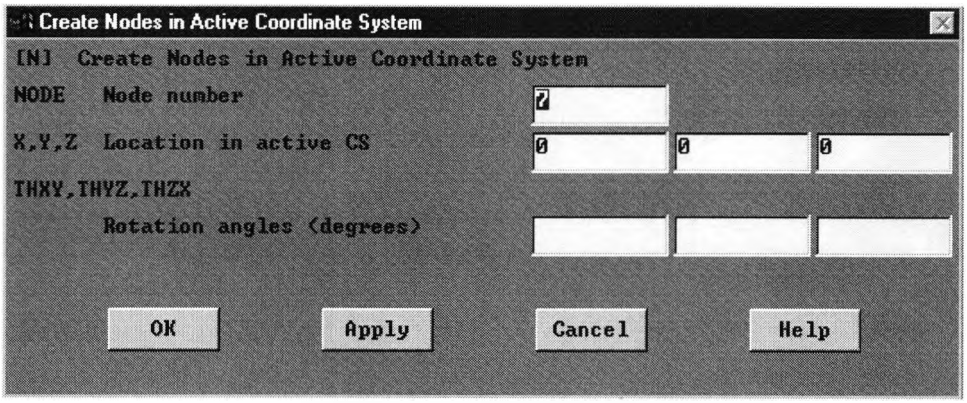


Figure B7: Defining nodes

This is repeated for node 9, then node 8 is created using the FILL command shown in figure B8.

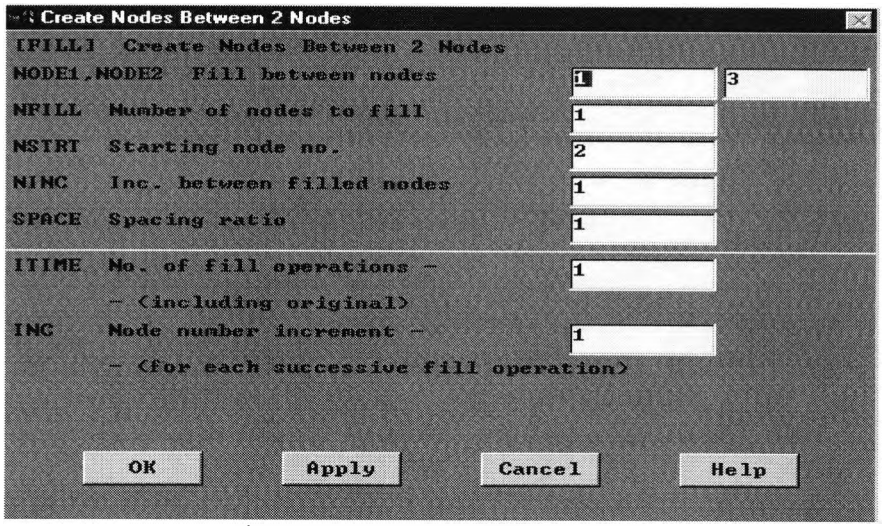


Figure B8: Filling between nodes

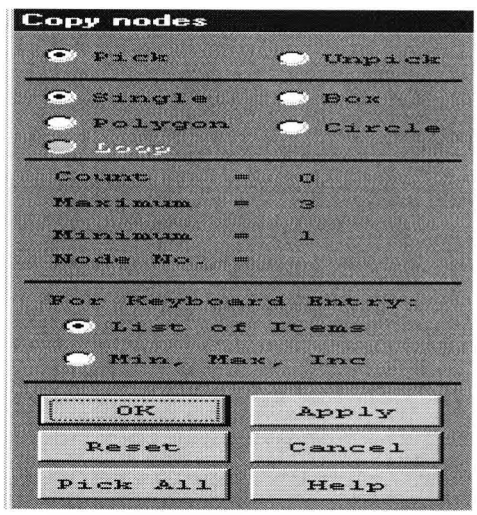


Figure B9: Picking nodes to copy

The three nodes can copied to create the bottom layer, this is done by 'picking' these nodes, shown in figure B9. Once the nodes are selected for copy, the NGEN command appears, this is shown in figure B10. The bottom layer is now generated, this can be seen in figure B11.

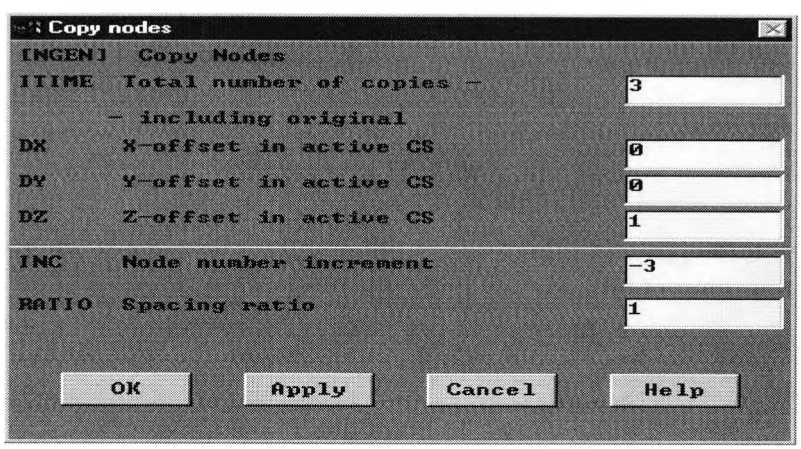


Figure B10: Copying nodes

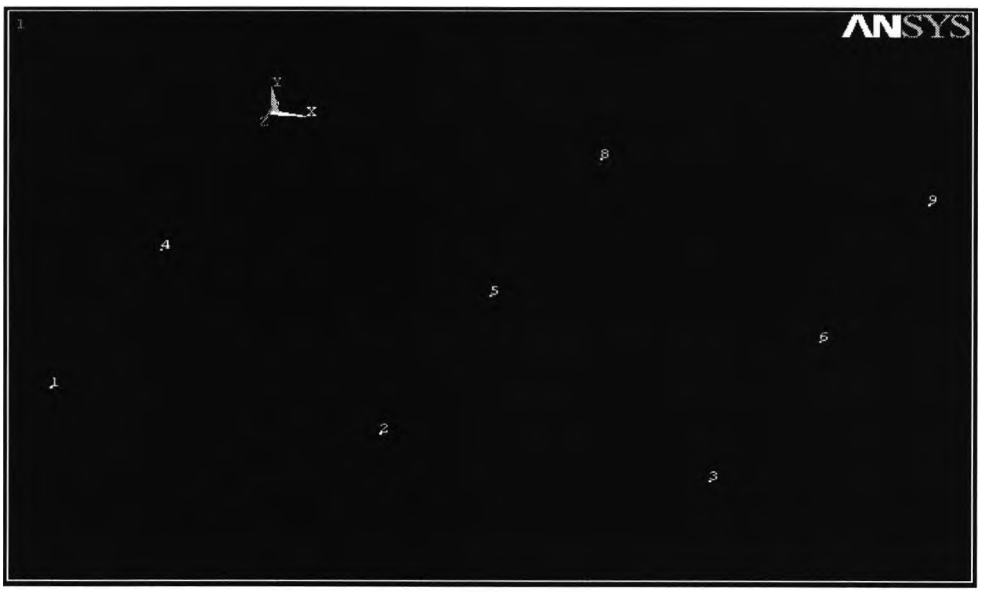


Figure B11: Building of bottom layer

To check that the nodes have been plotted with the correct co-ordinates, it is best to list all the nodes, this is accomplished by selecting NODE from the LIST menu under the mechanical utility menu. The list of these nodes are shown in figure B12.

```
LIST ALL SELECTED NODES.   DSYS=  0
SORT TABLE ON  NODE  NODE  NODE

  NODE        X           Y           Z          THXY     THYZ     THZX
     1     .00000      .00000     2.0000          .00      .00      .00
     2     1.0000      .00000     2.0000          .00      .00      .00
     3     2.0000      .00000     2.0000          .00      .00      .00
     4     .00000      .00000     1.0000          .00      .00      .00
     5     1.0000      .00000     1.0000          .00      .00      .00
     6     2.0000      .00000     1.0000          .00      .00      .00
     7     .00000      .00000     .00000          .00      .00      .00
     8     1.0000      .00000     .00000          .00      .00      .00
     9     2.0000      .00000     .00000          .00      .00      .00
```

Figure B12: List of nodes created

The view in the graphics window can be changed by the PAN, ZOOM, ROTATE command under the PLOTCTRLS in the MECHANICAL UTILITY MENU. This is shown in figure B13.

Once the bottom layer is established, this can be copied again using the NGEN command. Then the elements can be created. The method for building nodes exactly the same, so to create the first elements, the nodes attached to this element need to defined. Referring to figure B3 and B4, the nodes for the first element is 4, 5, 8, 7, 13, 14, 17, and 16. The nodes need to 'picked' which is shown in figure B13. Once the nodes has been selected for the element specified, the element is created, this is shown in figure B14.

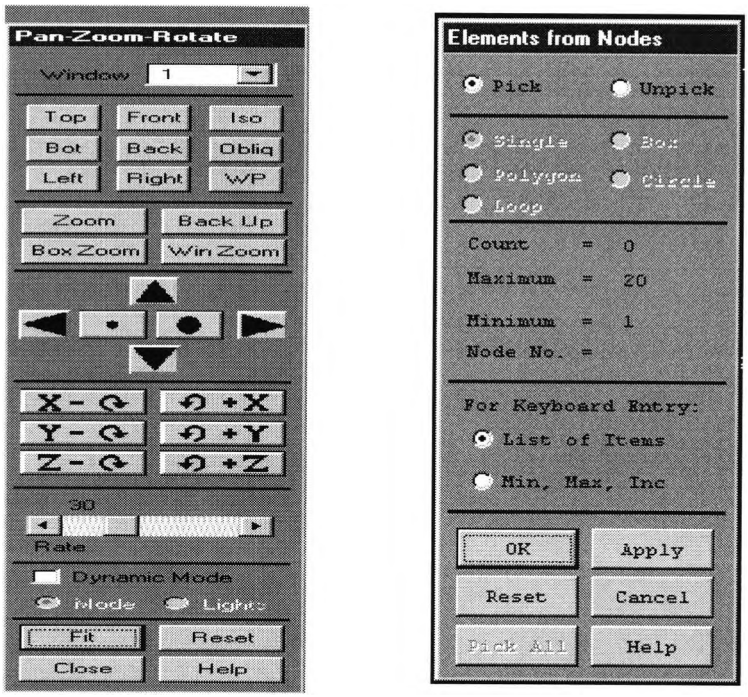


Figure B13: Changing view and picking the elements

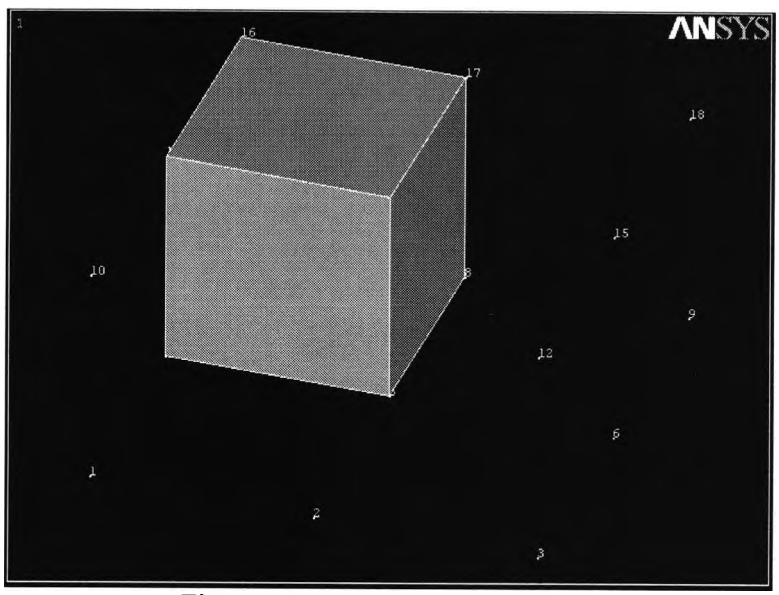


Figure B14: Creating an element

The element created can be copied to create the next element in it's row using the EGEN command. This is shown in figure B15. The result of this is shown in figure B16.

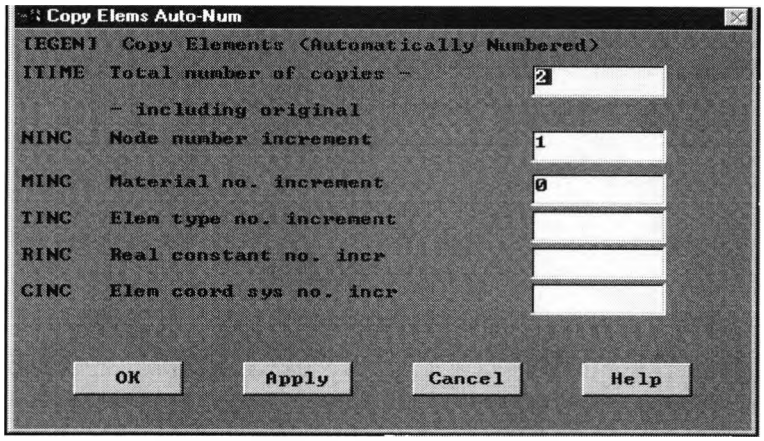


Figure B15 : Copying elements

This can then be copied again by 'picking' the elements to be copied, and using the EGEN command. It is important to define the same material number under MINC which in this case is 0. This is the default value which would refer to the material number stated earlier under material properties. The final model is now built, this is shown in figure B17. Using the PAN, ZOOM, ROTATE command, the view of the model can be altered in several direction to see that the model is accurately produced.

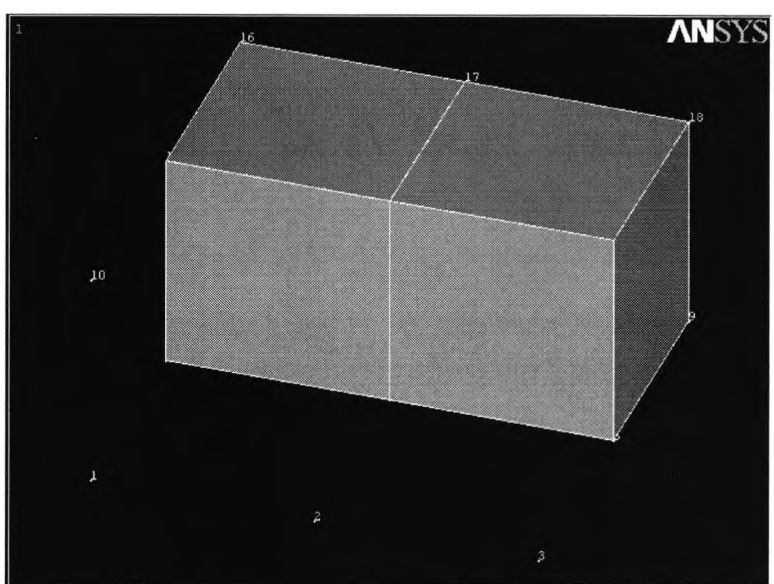


Figure B16: Copy of 2nd element

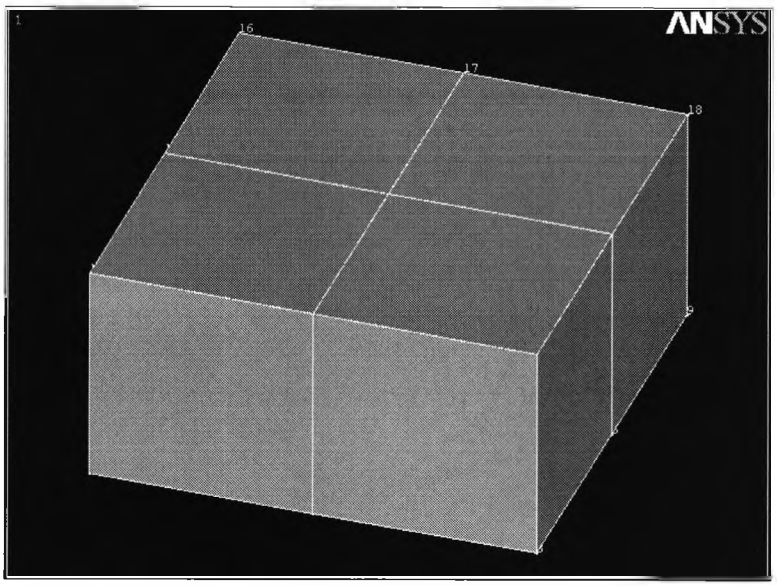


Figure B17: Copy of Remaining elements

Boundary conditions now need to be applied. The four nodes at the corners of the top and bottom of the model is selected by 'pick'. The displacement in the x, y, and z is set to 0 from the menu shown in figure B18. Once this is selected, the constraints on the model are shown in figure B19.

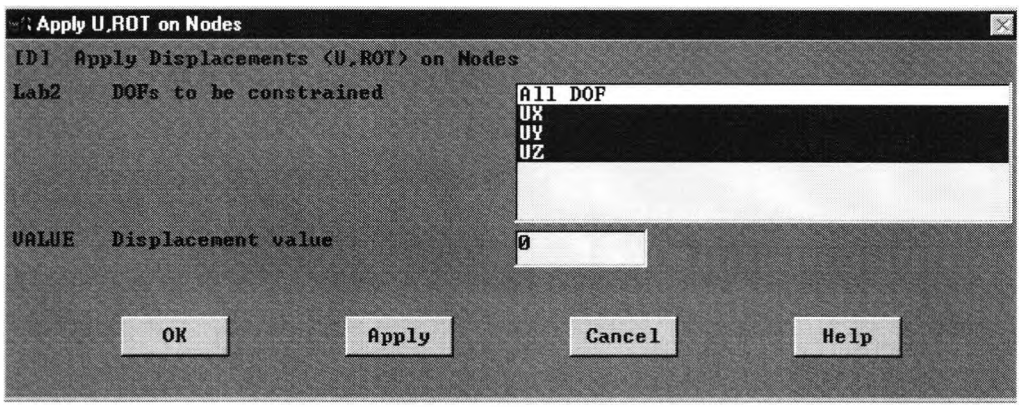


Figure B18: Adding the constraints

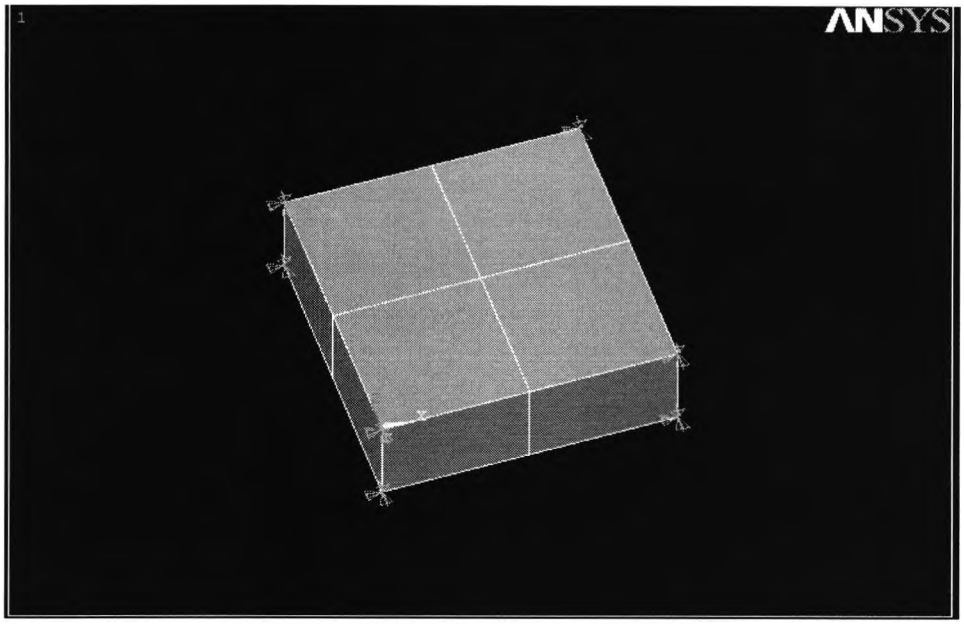


Figure B19: Model with constraints

The model is now complete, by selecting FINISH allows the user to return to the ANSYS MAIN MENU. Now the next stage need to be entered which is SOLUTION.

B.5. Solution

This stage defines the loading on the model and the type of analysis. The analysis is chosen to be transient which is appropriate for this case. Selection of this is shown in Figure B20.

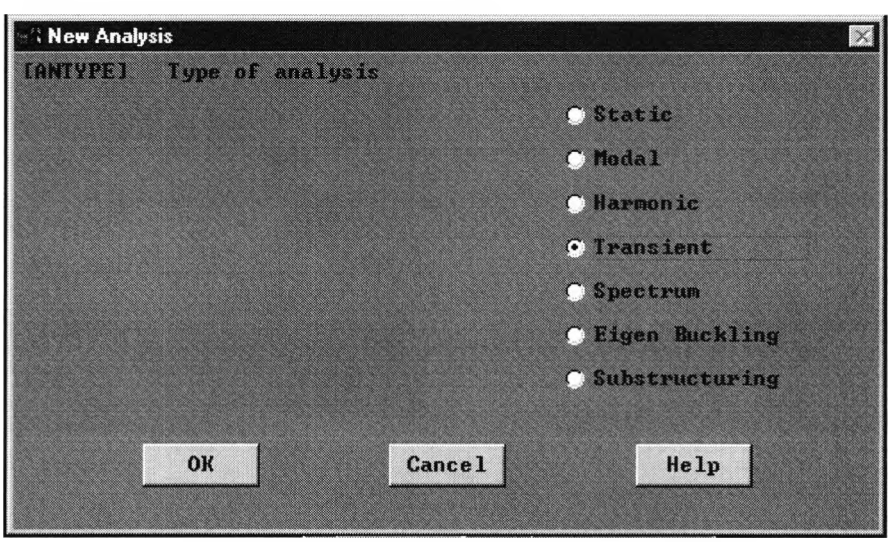


Figure B20: Selection of transient analysis

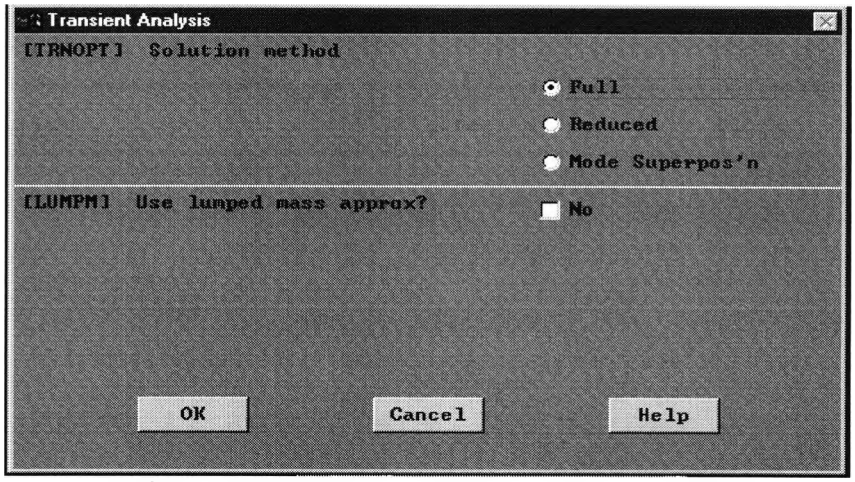


Figure B21: Selection of full transient analysis

There are several types of transient analysis available, in this example, a full transient analysis is selected. This is shown in figure B21.

The program is halted at this stage the procedures are all saved and program is exited. The file.log is open which has recorded all the commands entered. The commands of ANSYS 5.0 is added to the file and saved as a new file. This new file is then entered into ANSYS through the READ command from the MECHANICAL UTILITY MENU. This method is simpler and quicker, since the commands, TIME, FORCE, and LSWRITE have to be repeated 110 times. The program is now complete and ready to be solved, the force applied to the model is shown in figure B22.

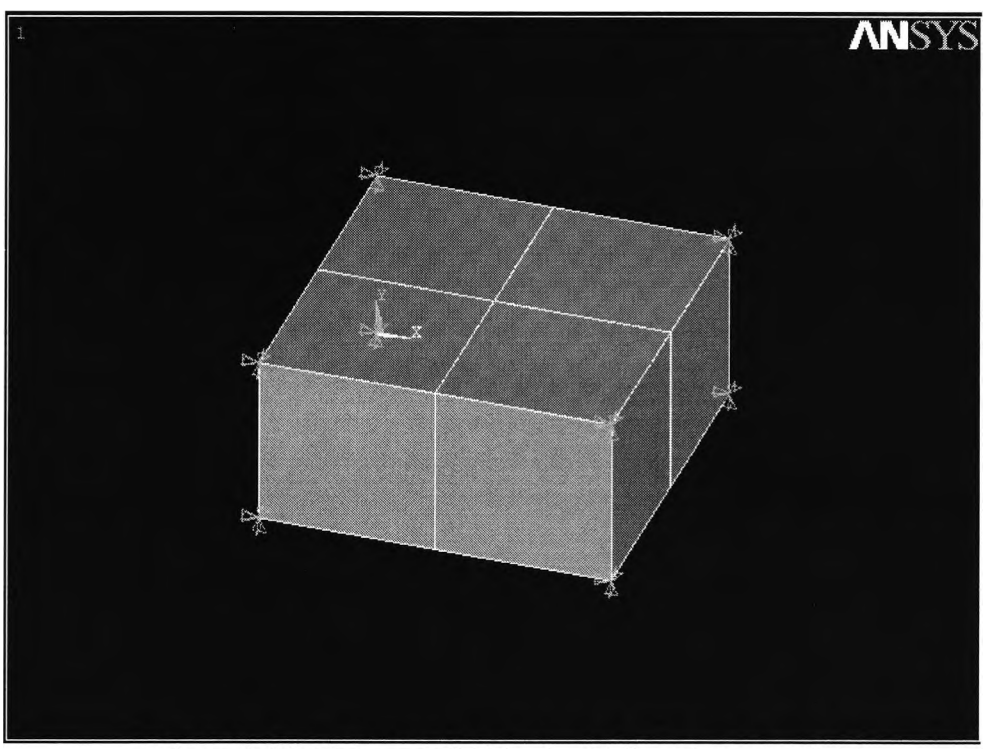


Figure B22: Force applied in the Y direction at node14

To solve the program, the SOLVE command is selected, and the LSWRITE files are selected from 1 to 110. This shown in figure B23. This will take about 30 minutes to complete. Then the results are ready to be examined. The results for node 15 will be investigated.

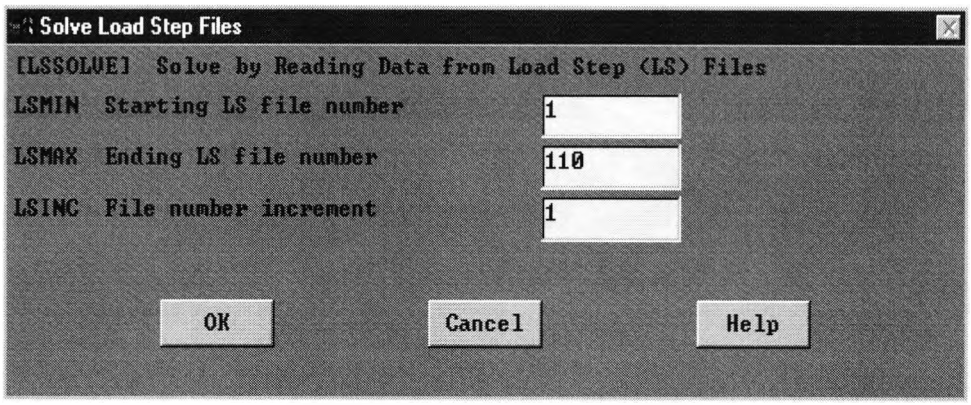


Figure B23: Selection of the files

B.6. Post-processing

The post-processing stage consist of two options. The first part is called POST1 which allows the user to examine the results at a specified time, this is useful for static analysis. The second option is the POST26, which allows a time history of the results to be investigated. This option is most appropriate in this example.

Although, it is possible to plot results for the node specified, the displacement for node 15 are not given at a constant time interval. So, a program is written in Fortran to provide this. This program is entered into ANSYS when the model is solved. A fresh start to ANSYS is made, the relevant data base file called file.db of the model is RESUMED. This file contains all the information about the nodes and elements only however. This is shown in figure B24

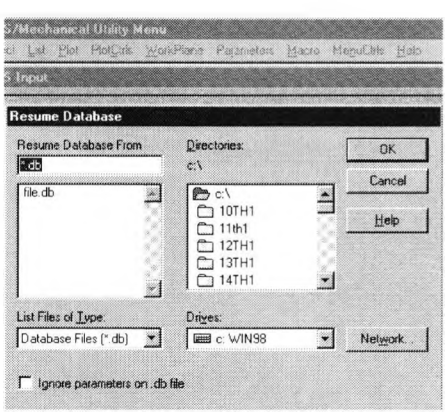


Figure B24 : Resuming the database file

Next, the file to convert data at constant time interval is entered, this is done by using the READ command.

Once this is completed, the results for node 15 is given in a form which can be requires editing. A typical output file produced by ANSYS is shown below:

*** These column contains 1024 points of time increments of 10 microseconds ***

	LOCATION		VALUE
1	1	1	1.000000000E-05
2	1	1	2.000000000E-05
3	1	1	3.000000000E-05
4	1	1	4.000000000E-05
5	1	1	5.000000000E-05
		to	
1005	1	1	1.005000000E-02
1006	1	1	1.006000000E-02
1007	1	1	1.007000000E-02
1008	1	1	1.008000000E-02

*** These column contains 1024 displacement values for the corresponding time** *

1	2	1	5.865070286E-13
2	2	1	3.942119502E-12
3	2	1	9.987967229E-12
		to	
1021	2	1	-1.337658889E-11
1022	2	1	-2.072736914E-11
1023	2	1	-2.735328574E-11
1024	2	1	-3.314114102E-11

The last column on the right are the time, displacement. This is edited using excel, so that single columns of time and displacement is obtained. The results are plotted which is shown in figure B25.

The displacement column is copied to a separate file and saved as a text file, so that FFT is processed in Matlab. The amplitude spectrum of this example is shown in the figure B26 below.

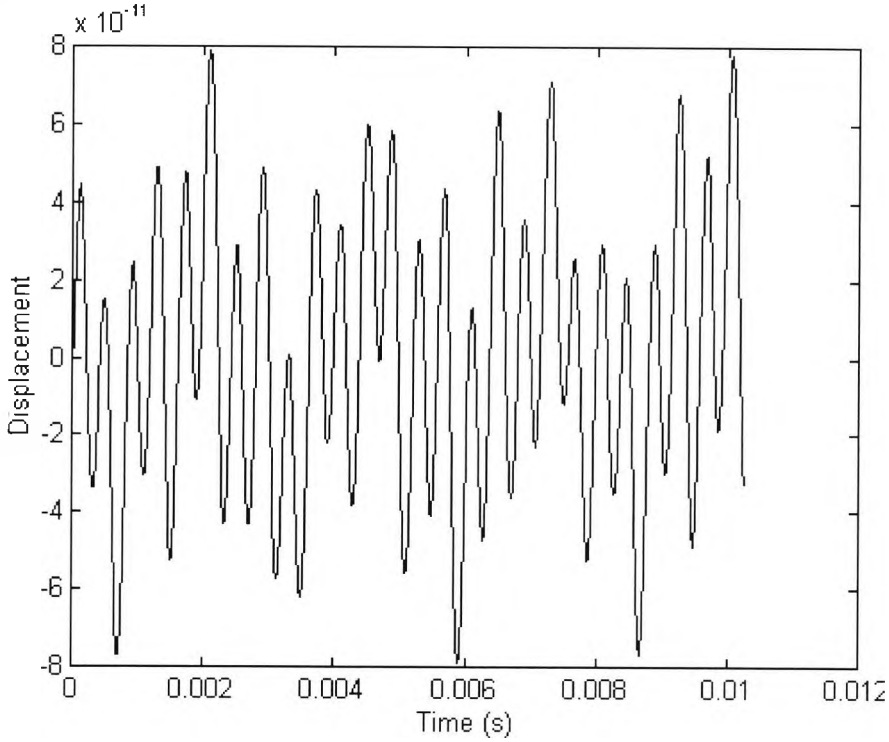


Figure B25: Displacement vs Time

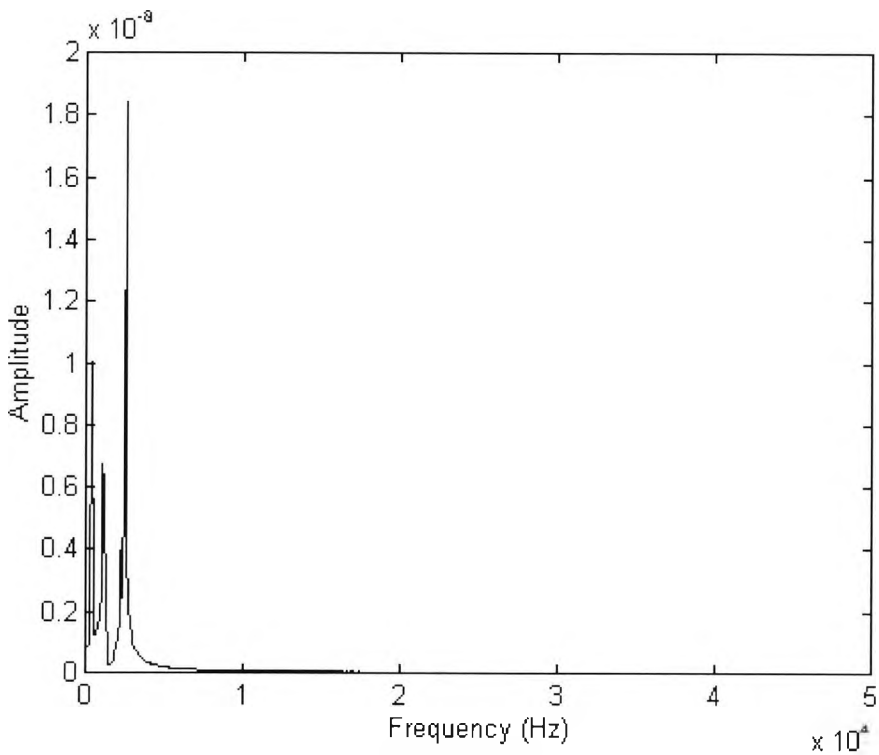


Figure B26: Amplitude vs Frequency

B.7. Summary

An example of a 3D model has been used to show the operation of the ANSYS finite element program. At the pre-processing stage, nodes and elements are defined with the material properties for concrete. The boundary conditions, constraints and force are applied at the solution phase, and finally once the model is solved, the results are view under the post-processing stage.

There are two stages of data input in ANSYS, one is the interactive mode, the other the batch mode, where a program which has been created beforehand can be 'read' into the software. Both of these examples have been demonstrated in this appendix.

Appendix C

FFT Method

C.1. Introduction

This appendix explains the process of FFT. It is a mathematical process which transform data from time domain to frequency domain. The FFT as discussed, has been used on the data received from the impact echo data and results from the impact echo simulation. A simple example to illustrate the method is shown below.

C.2. Theory

The method used to change an analogue signal x(t) into frequency domain is the fourier transform which is defined by :

$$F(t)\ ) = a_0/2 + \sum_{n=1}^{\infty} (a_n \cos n\omega_T + b_n \sin n\omega_T) \qquad \text{Equation C1}$$

Where ω_T is $2\pi/T$

$$a_0 = 2/T \int_0^T F(t)\, dt \qquad \text{Equation C2}$$

$$a_n = 2/T \int_0^T F(t)\, dt \cos n\omega_T\, dt \qquad \text{Equation C3}$$

where n = 1, 2,3, ...

$$b_n = 2/T \int_0^T F(t)\, dt \sin n\omega_T\, dt \qquad \text{Equation C4}$$

where n = 1, 2,3, ...

A periodic time signal of period T can be represented by Fourier series in time of the form given by equation C1 with Fourier coefficients defined by equations C2 to C4. The Fourier coefficients represent frequency domain information about a given signal. Also, the Fourier coefficients a_n and b_n given in equations C3 and C4 illustrate the connection between Fourier analysis and vibration experiments. The analogue output signals of a

transducer, x(t), are the inputs to the analyser (signal processor). The analyser calculates the spectral coefficients of these signals. The analogue to digital (A/D) converter processes the following :

- The analyser converts the analogue signals into digital records.
- It samples the signal x(t) at many different equally spaced values and a version of the signal in the form of a set of numbers $\{x(t_k)\}$, where k = 1,2, ...N. N = Number of samples, and t_k = Discrete value of the time.

When the digital record of the signal is ready, the discrete version of the Fourier transform is processed. The digital Fourier transform is defined by :

$$x_k = x(t_k) = a_0/2 + \sum_{i=1}^{N/2} (a_i \cos 2\pi i t_k / T + b_i \sin 2\pi i t_k / T) \qquad \text{Equation C5}$$

where k = 1,2,....N

The digital spectral coefficients which are the digital versions of equations C2, C3 and C4:

$$a_0 = 1/N \sum_{k=1}^{N} x_k \qquad \text{Equation C6}$$

$$a_i = 1/N \sum_{k=1}^{N} x_k \cos 2\pi i k / N \qquad \text{Equation C7}$$

$$b_i = 1/N \sum_{k=1}^{N} x_k \sin 2\pi i k / N \qquad \text{Equation C8}$$

The function of the analyser is to calculate a_0, a_i and b_i given the digital record $x(t_k)$ for the measured signals. The number of samples, N, is usually a set for a given analyser and is the power of two, such common sizes are 512 and 1024.

For each sample, writing out equations C6 to C8 results in N linear equations in the N Fourier coefficients (a_0, ..., $a_{N/2}$, b_0, ... , $b_{N/2}$) which can be written in the form of matrix equations. In matrix form they are:

$$x = C\, a \qquad \text{Equation C9}$$

where x is the vector of samples with elements x_k

a is the vector of Fourier coefficients (a_0,a_i and b_i)

C is the elements containing the coefficients cos ($2\pi it_k/T$) and sin ($2\pi it_k/T$)

It can be noted that x represents the digital version, the Fourier coefficients a represent the frequency content of the input signal. The solution of equation C9 for the Fourier coefficients is given by :

$$a = C^{-1} x \qquad \text{Equation C10}$$

The analyser is employed to compute the matrix C^{-1} and hence and coefficient a. The FFT is the most widely used method of computing the inverse of this matrix C. It was developed by Cooley & Tukey (1965).

C.3. Example of an amplitude spectrum

A waveform can be represented as a sum of sin and cos curves, each with an individual amplitude, frequency, and phase shift. This transformation is carried out using FFT. One important advantage of Fast Fourier series is that it can represent a function containing discontinuities, whereas other methods such as Maclaurin's and Taylor's series require the function to be continuous.

The FFT is a collective term for a number of efficient algorithms developed to compute the discrete fourier transform (DFT) and the inverse discrete fourier transform(IDFT) (Lockhart & Cheetham, 1989). The FFT technique can be used for spectral analysis, applied mechanics, acoustics, medical imaging, numerical analysis, instrumentation and telecommunications.

It has been shown that with the impact echo technique, interpretation of the recorded waveforms in the time domain, while being feasible was time consuming (Sansalone & Carino, 1986). This involved calculating the time between the start of the impact and the arrival of the first P wave reflection (refer to Chapter 3). It can be illustrated that data interpretation is much simpler in the frequency domain than in the time domain. If a digital time domain waveform function is :

$$g(t) = \sin 2\pi(20)t + 2*\sin 2\pi(40)t + 3*\sin 2\pi(60)t \qquad \text{Equation C11}$$

Where t is time in seconds

It can be seen that this function is composed of three sin curves of different amplitudes with frequencies 20, 40, and 60 Hz. This function can be seen in figure C1.

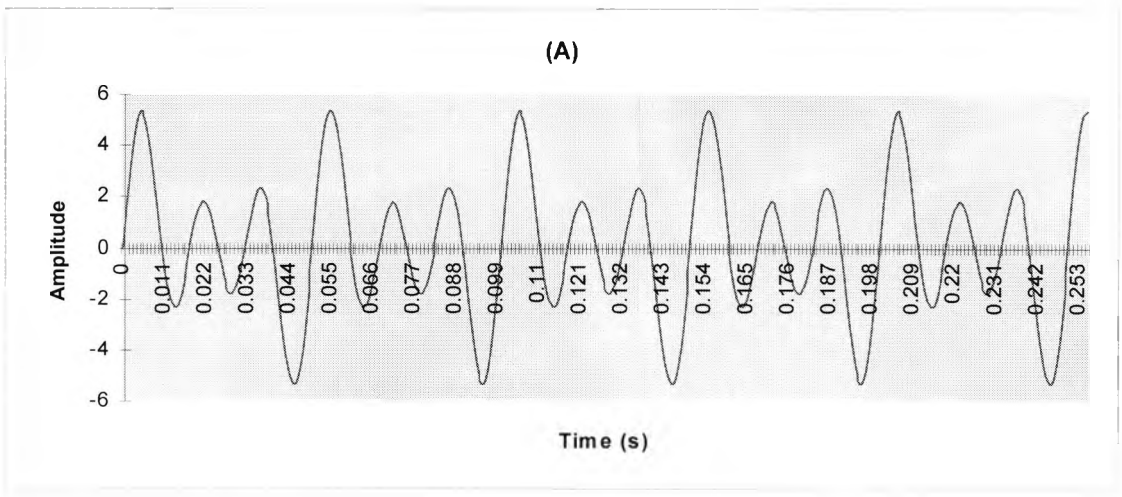


Figure C1: Data in time domain

The function g(t) is composed of three sin curves of different amplitudes having frequencies of 20, 40, and 60 Hz. This can be seen in figure C2 of the frequency spectrum. The peaks occur at these particular frequencies and corresponds to one of the component sin curves in equation C11. This spectrum was obtained from the digital time domain waveform using the FFT technique from Microsoft Excel. The FFT program for Matlab is included in Appendix F.

The graph of figure C1 has :

- Sampling points, $S_p = 256$
- Time interval between points or sampling period, $T = 0.001$ seconds
- Sampling frequency or sampling rate, $F_s = 1 / \mathbf{T} = 1/ 0.001 = 1000$ Hz

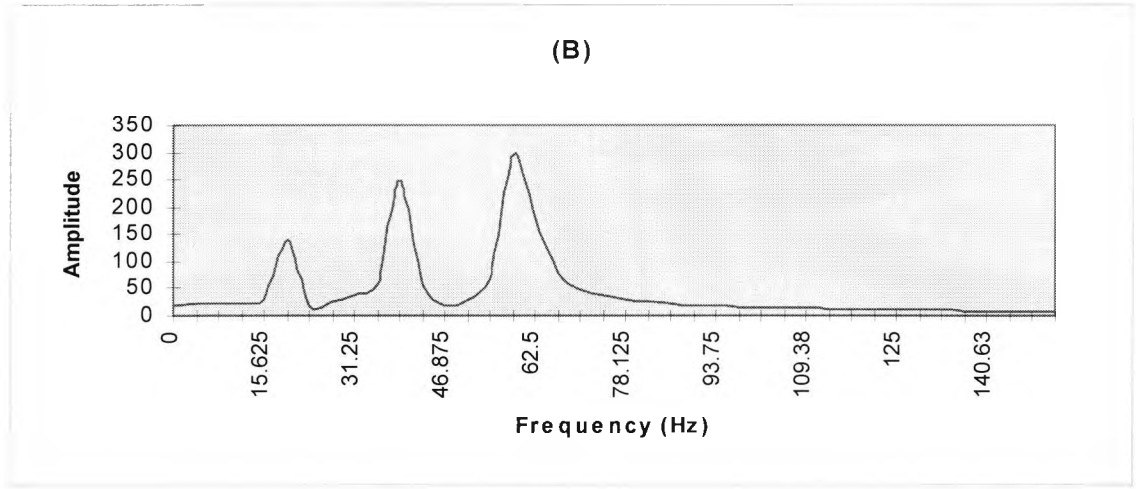


Figure C2: Data in frequency spectrum

The frequency spectrum was calculated by the FFT technique. Hence the graph of figure C2 has :

- Maximum frequency in graph (B), F_{max}(since frequency spectrum from FFT method contains half as many points as the time domain waveform) = 1/2 * F_s = 1/2 * 1000 = 500 Hz
- Frequency interval = F_{max} / 1/2* S_p (No. of points in the spectrum i.e 128) = 500 / 128 = 3.9 Hz

C.4. Selecting a sampling rate

It is clear that the frequency interval is inversely proportional to the sampling rate, this means that a slower sampling rate enhances resolution in the frequency domain. However to avoid errors in frequency analysis:

> To sample a signal properly, the sampling frequency must be at least twice as high as the highest frequency component in the signal

For example, if the sample rate is less than twice the highest frequency, this results in an inaccurate representation of the input. So, new frequencies are generated and added to the sampled signal. This phenomenon is called aliasing.

Aliasing can be avoided in signals containing many frequencies by providing anti-aliasing filter. This only allows low frequencies through, and cuts off frequencies higher than about half the maximum frequency (nyquist frequency).

There is an important relationship between the sample rate and the frequency content. To illustrate this point we sample a signal that contains a single frequency.

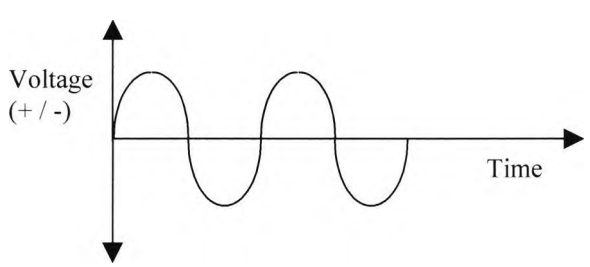


Figure C3: Sampled frequency

From figure C3 , the signal is very regular, it has a crest and trough. So, it is necessary to sample a sine wave twice during each cycle to capture the whole signal.

Another problem encountered in FFT analysis, is leakage. To ensure FFT analysis feasible, the periodic signal must be sampled over a finite time i.e. N must be finite. If the signal is to be finite, the signal could be cut off at any integral multiple of its period. Unfortunately, there is no simple solution to this because complicated signals containing many different frequencies. So the signal may be cut off at midpoint, this is leakage. This means erroneous frequencies appear in the FFT illustration, the digital fourier transform of the finite length signal assumes that the signal is periodic within the sample record length. Hence, the actual frequency will leak into a number of fictitious frequencies.

Leakage can be corrected more or less by the use of a window function. This involves multiplying the original analogue signal by a weighting function which alters the signal to zero outside the sampling period. Hamming window is a widely used function.

C.5. Summary

The theory of the FFT has been presented and an example of this method used demonstrated. The importance of sampling at the correct frequency and windowing effects have been explained.

Appendix D

Neural Network Data

The neural network data with effect of adding noise is illustrated in the following section, the first few graphs show the data with a standard deviation of 0.02, then the remaining graphs show data with a standard deviation 0.025.

The investigations to find the optimum parameters discussed in Chapter 7 is included in this appendix.

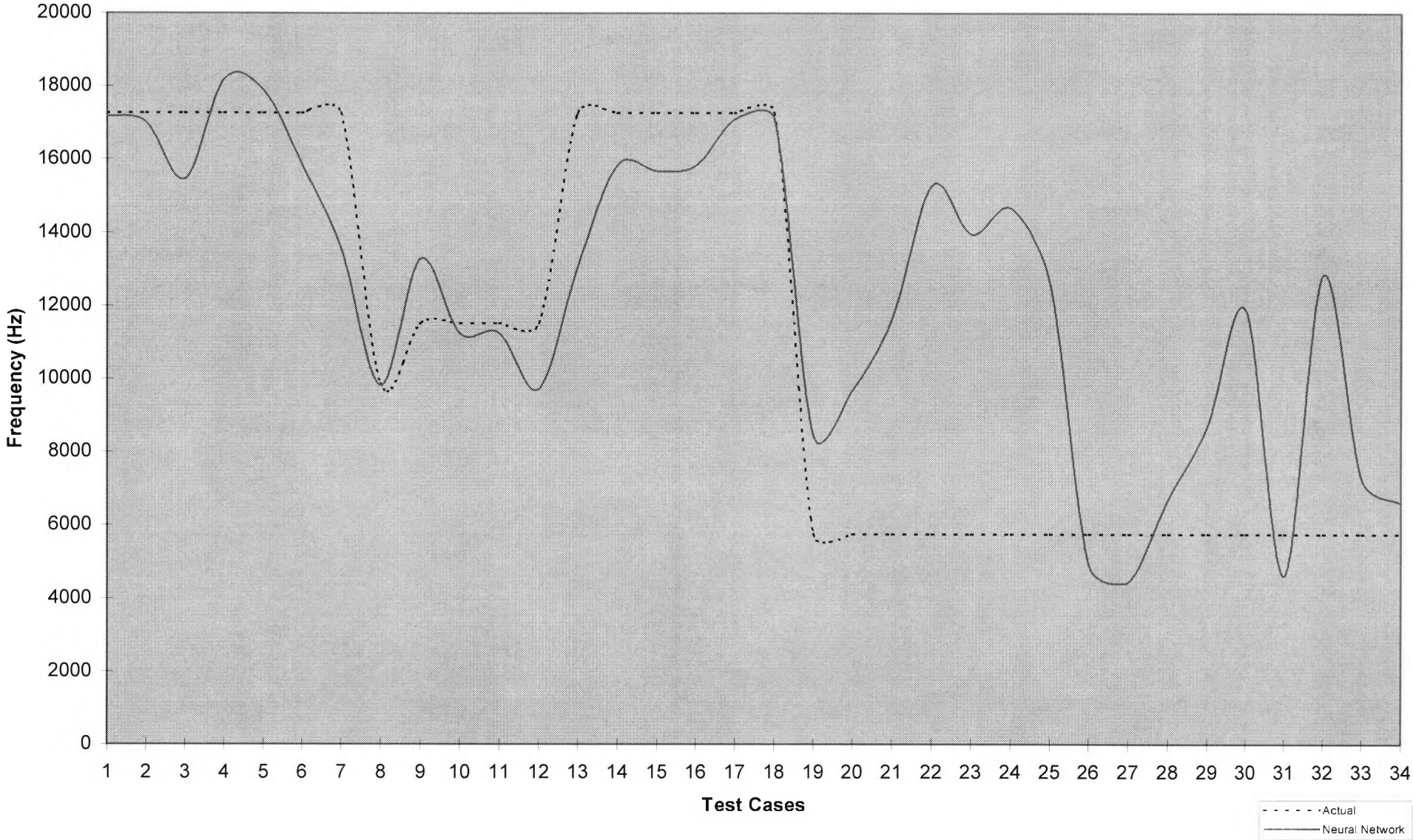


Figure D1: 1st set of noise added, training patterns = 396, test patterns = 34, hidden nodes = 20, standard deviation = 0.02

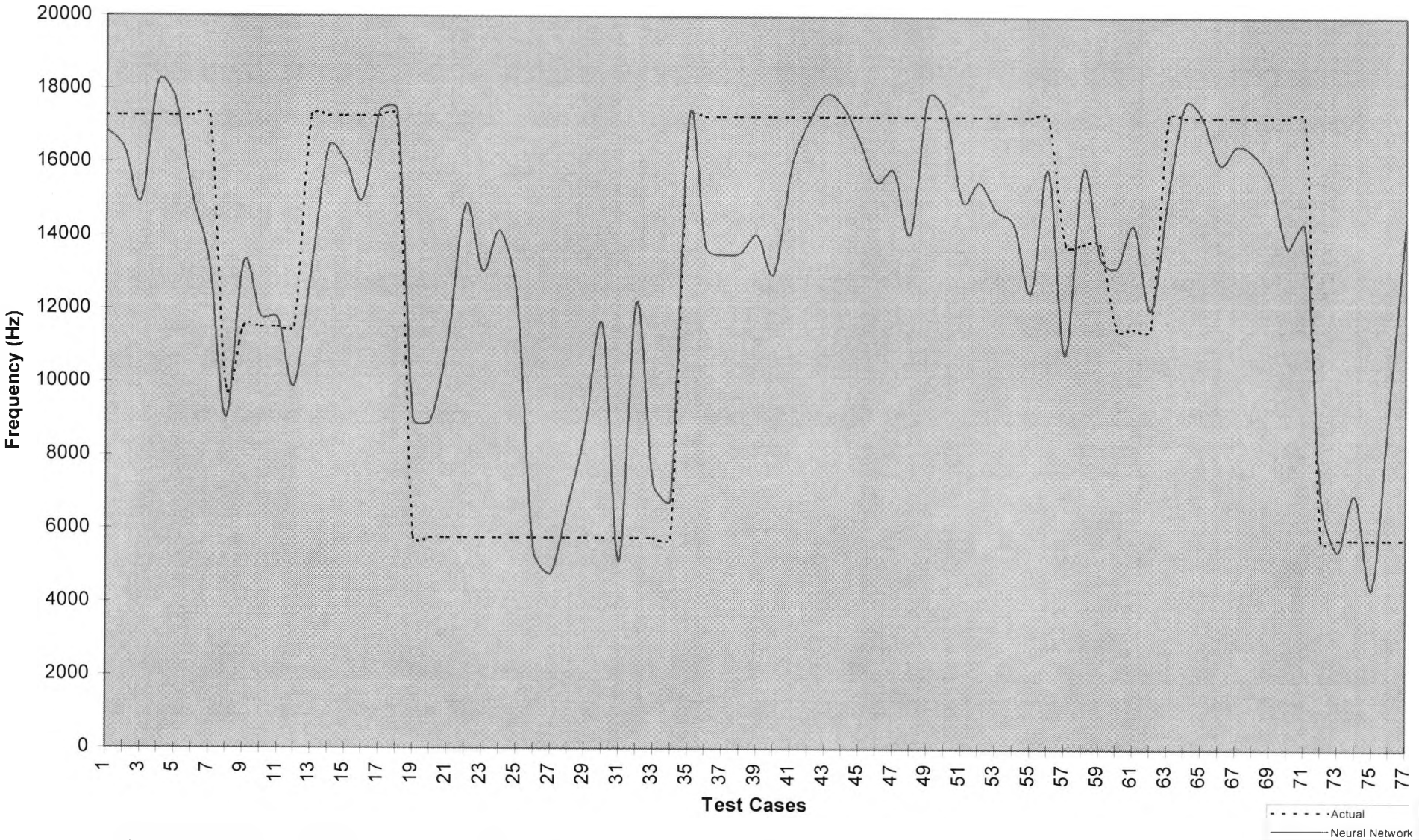


Figure D2: 2nd set of noise added, training patterns = 577, test patterns = 77, hidden nodes = 24, standard deviation = 0.02

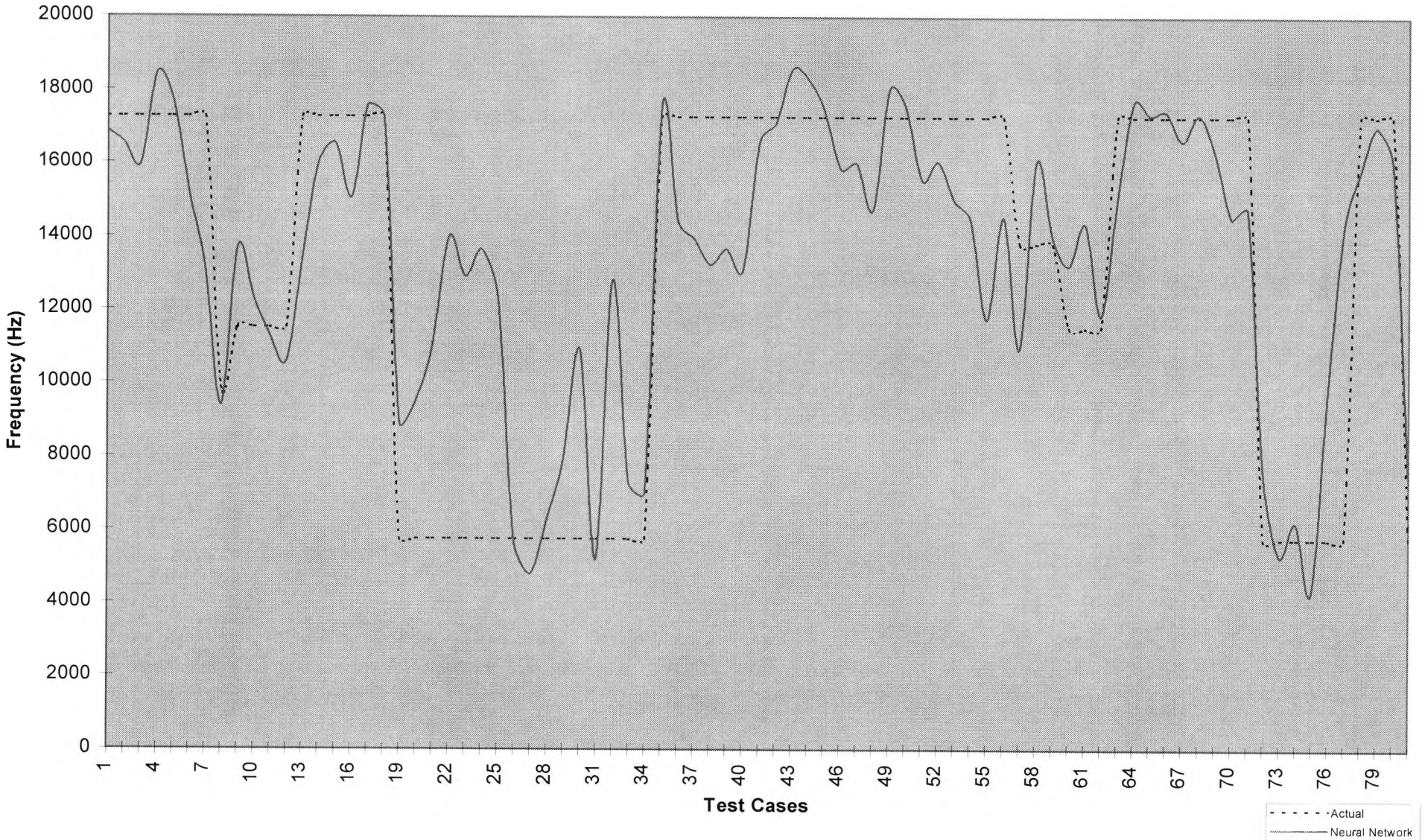


Figure D3: 3rd set of noise added, training patterns = 795, test patterns = 81, hidden nodes = 28, standard deviation = 0.025

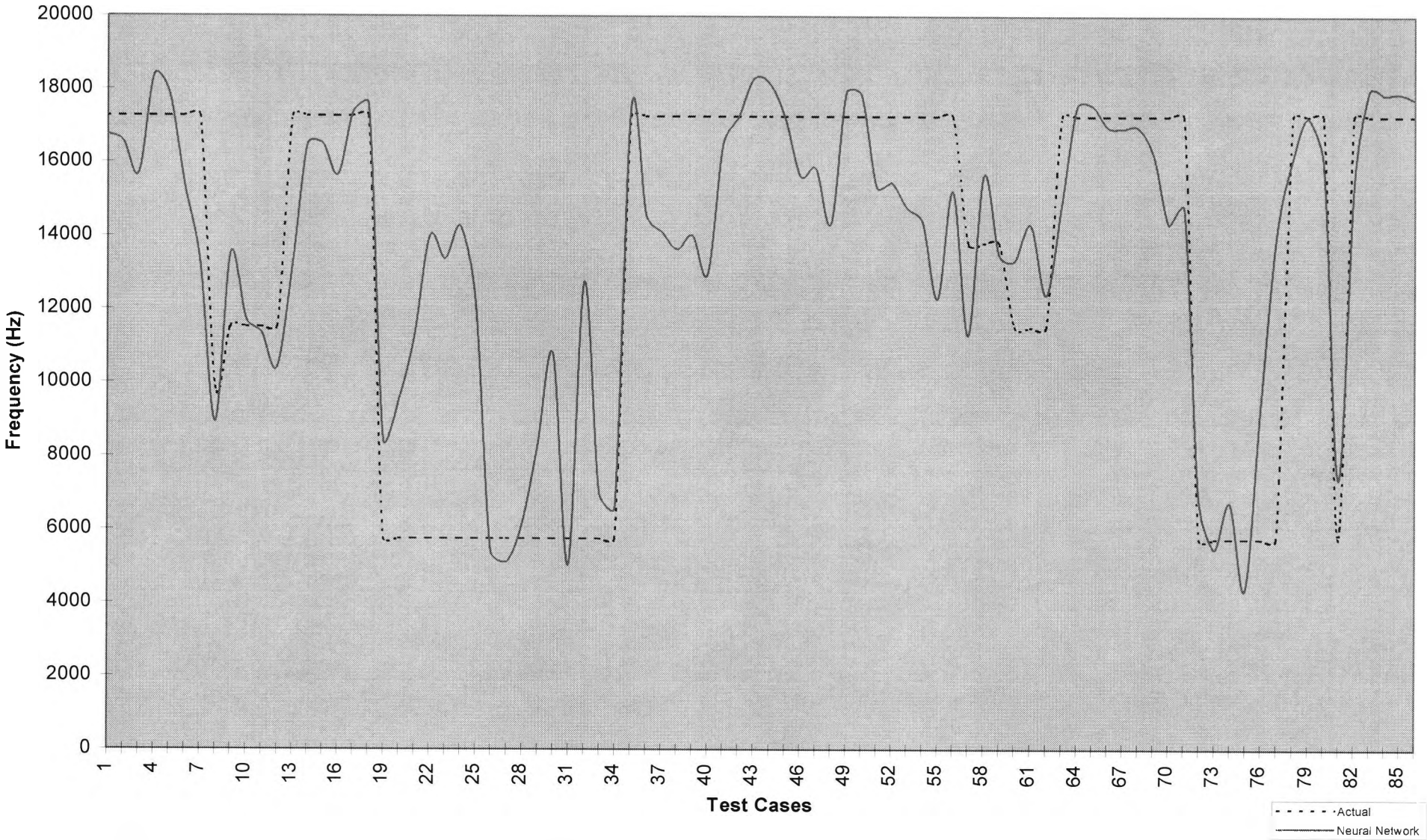


Figure D4: 4th set of noise added, training patterns =1015, test patterns = 86, hidden nodes = 32, standard deviation = 0.025

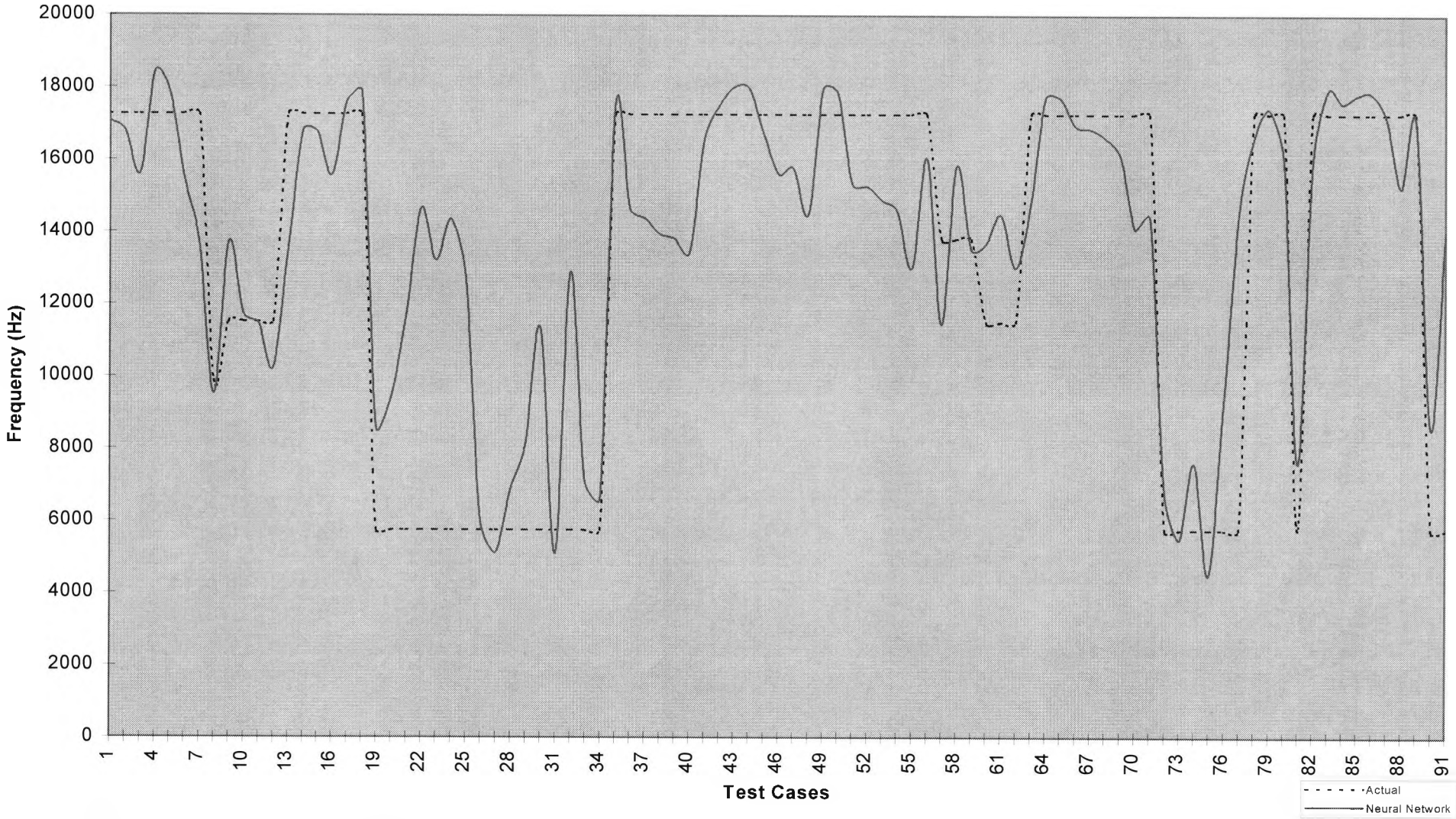


Figure D5: 5th set of noise added, training examples =1232, testing examples =91, hidden nodes = 35, standard deviation = 0.025

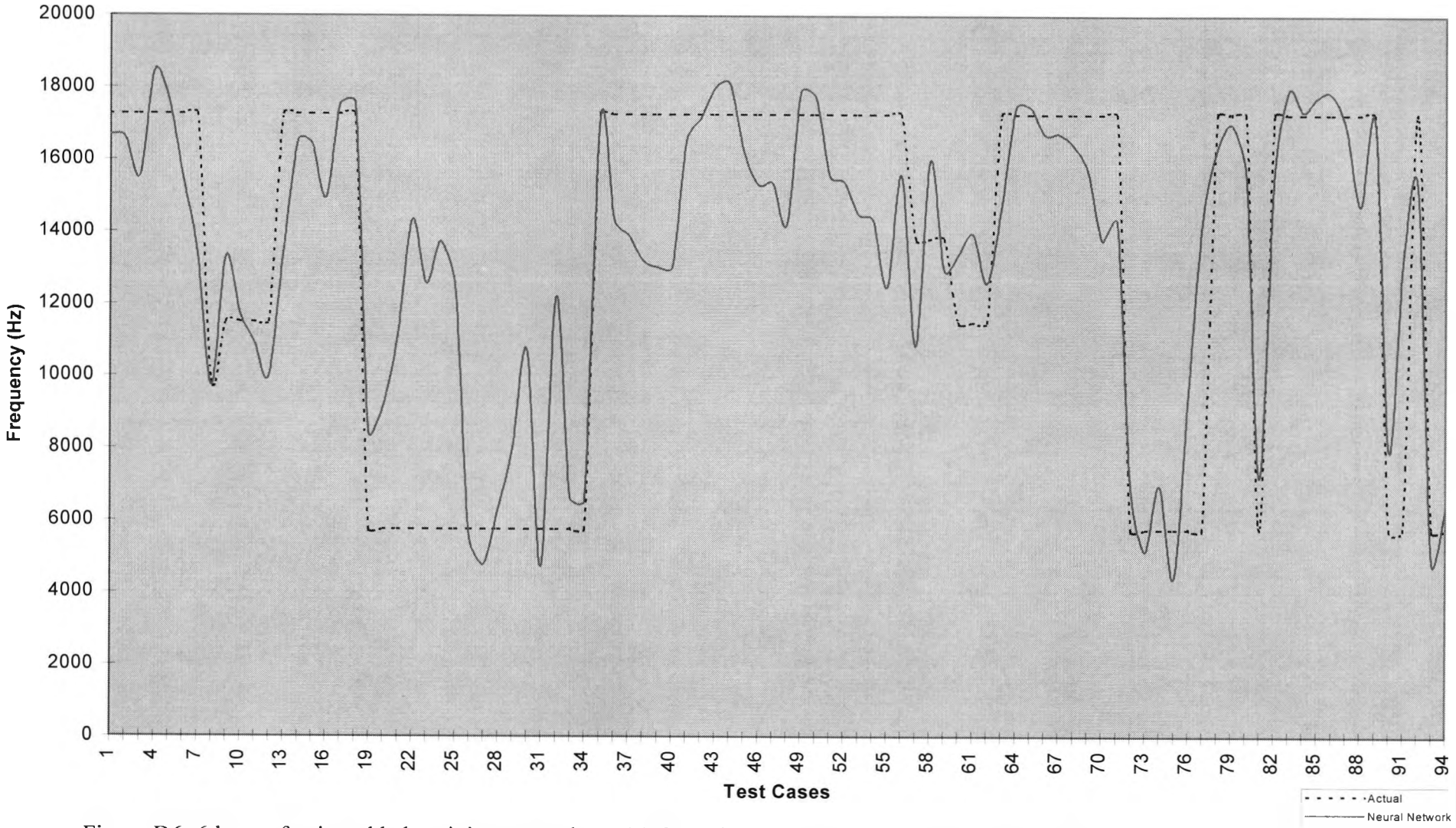


Figure D6: 6th set of noise added, training examples = 1453, testing examples = 94, hidden nodes = 38, standard deviation = 0.025

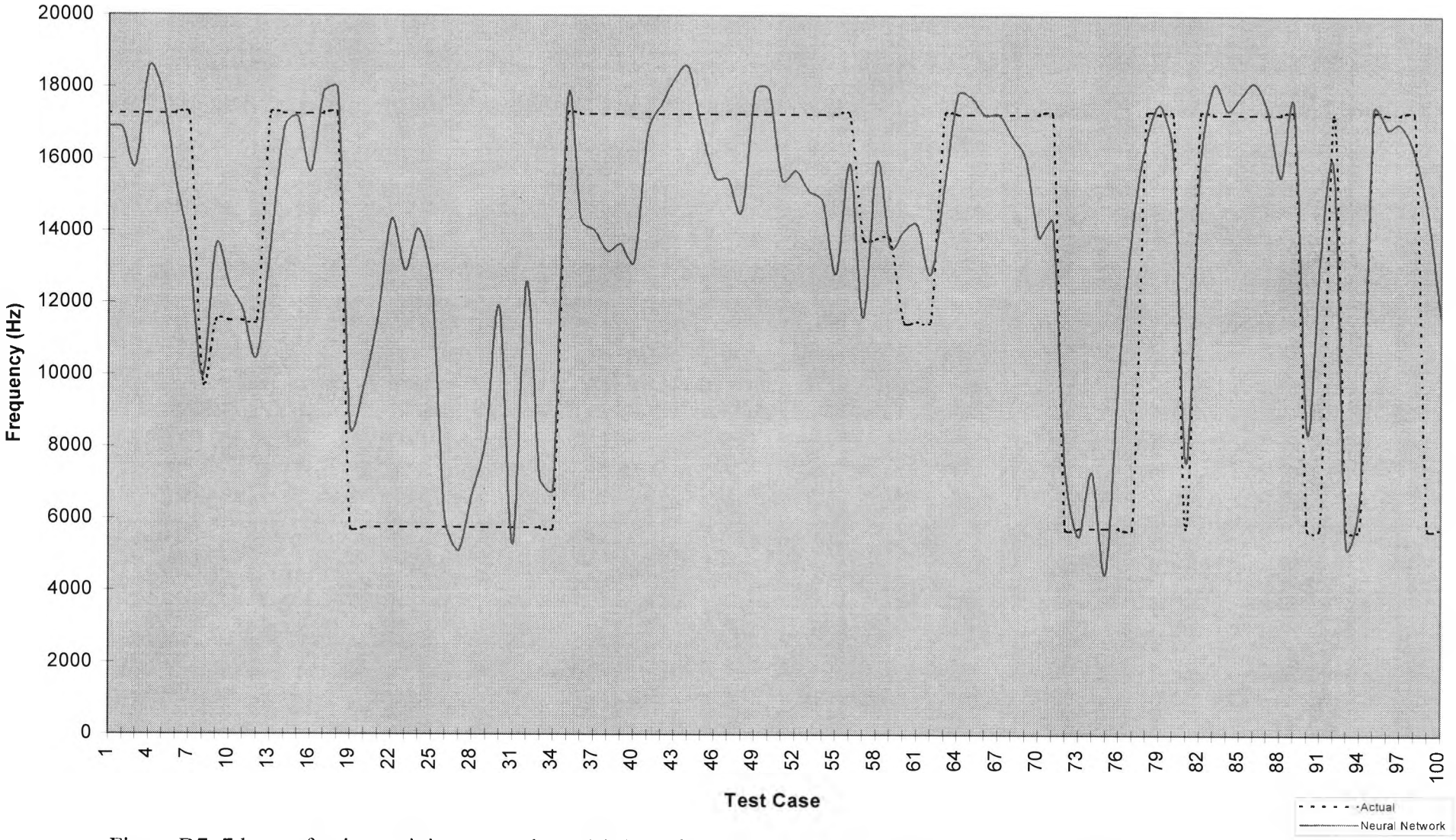


Figure D7: 7th set of noise, training examples = 1671, testing examples = 100, hidden nodes = 41, standard deviation =0.025

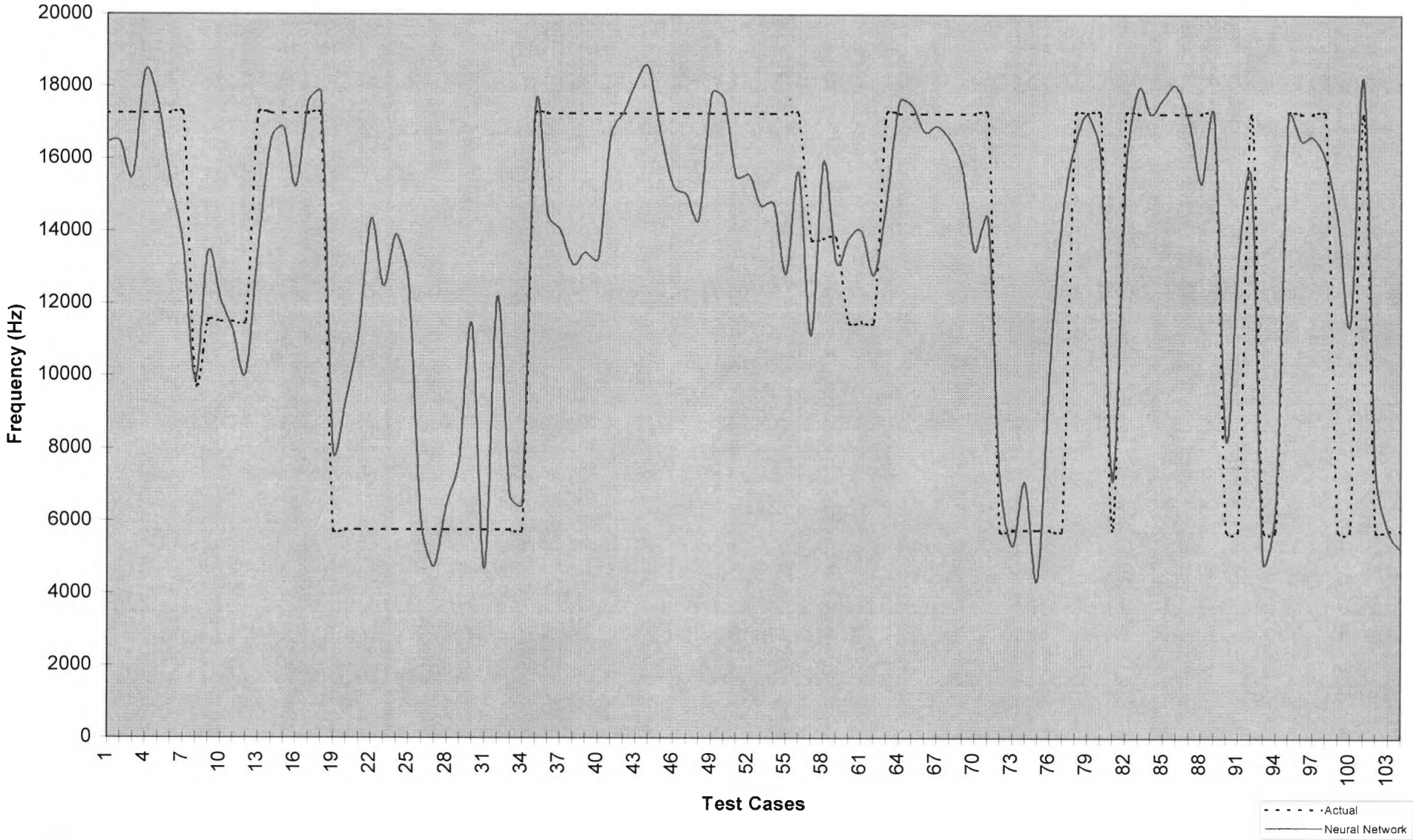


Figure D8: 8th noise set of noise added, training examples =1892, testing examples = 104, hidden nodes = 43, standard deviation = 0.025

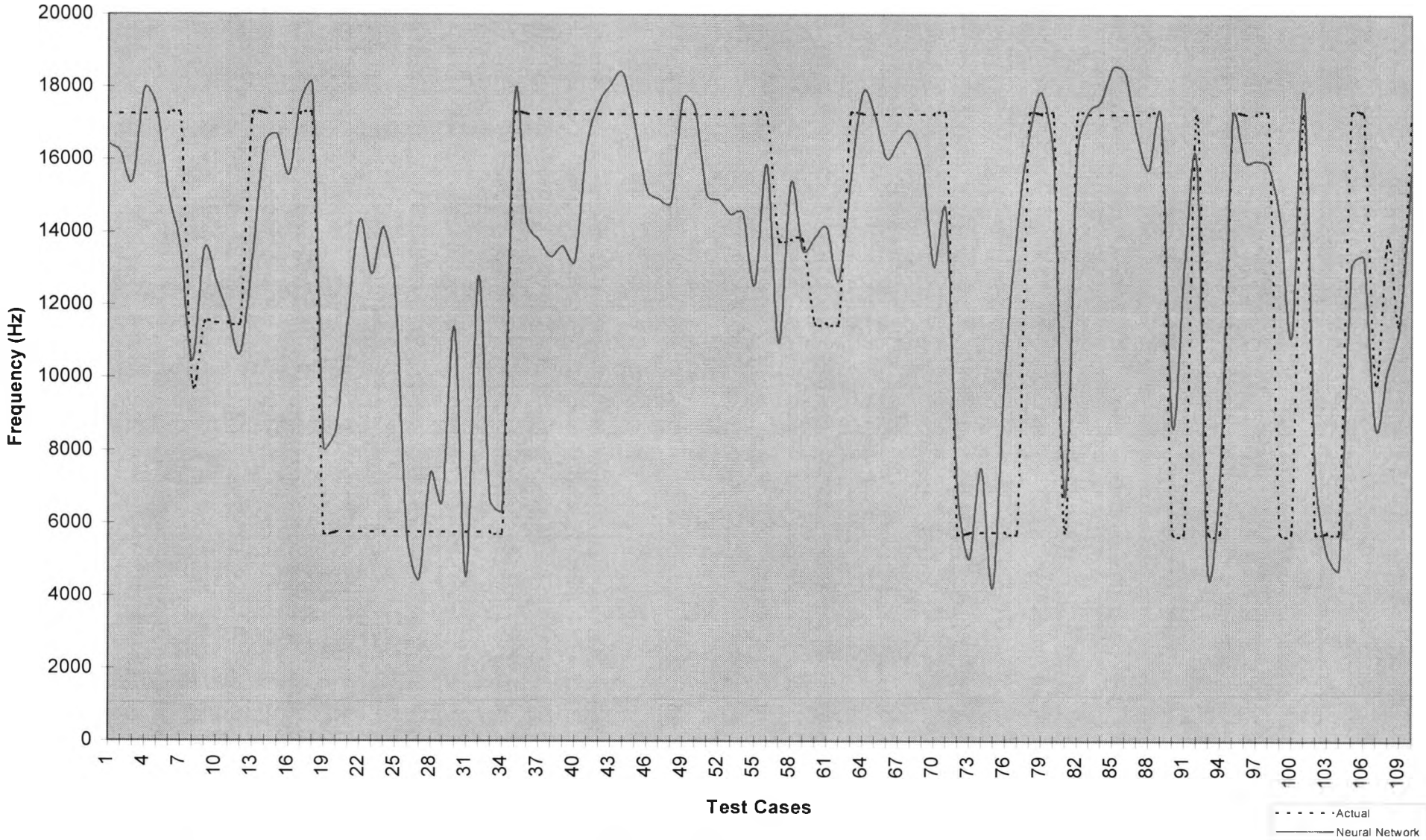


Figure D9: 9th set of noise added, training examples = 2111, testing examples =110, hidden nodes = 46, standard deviation = 0.025

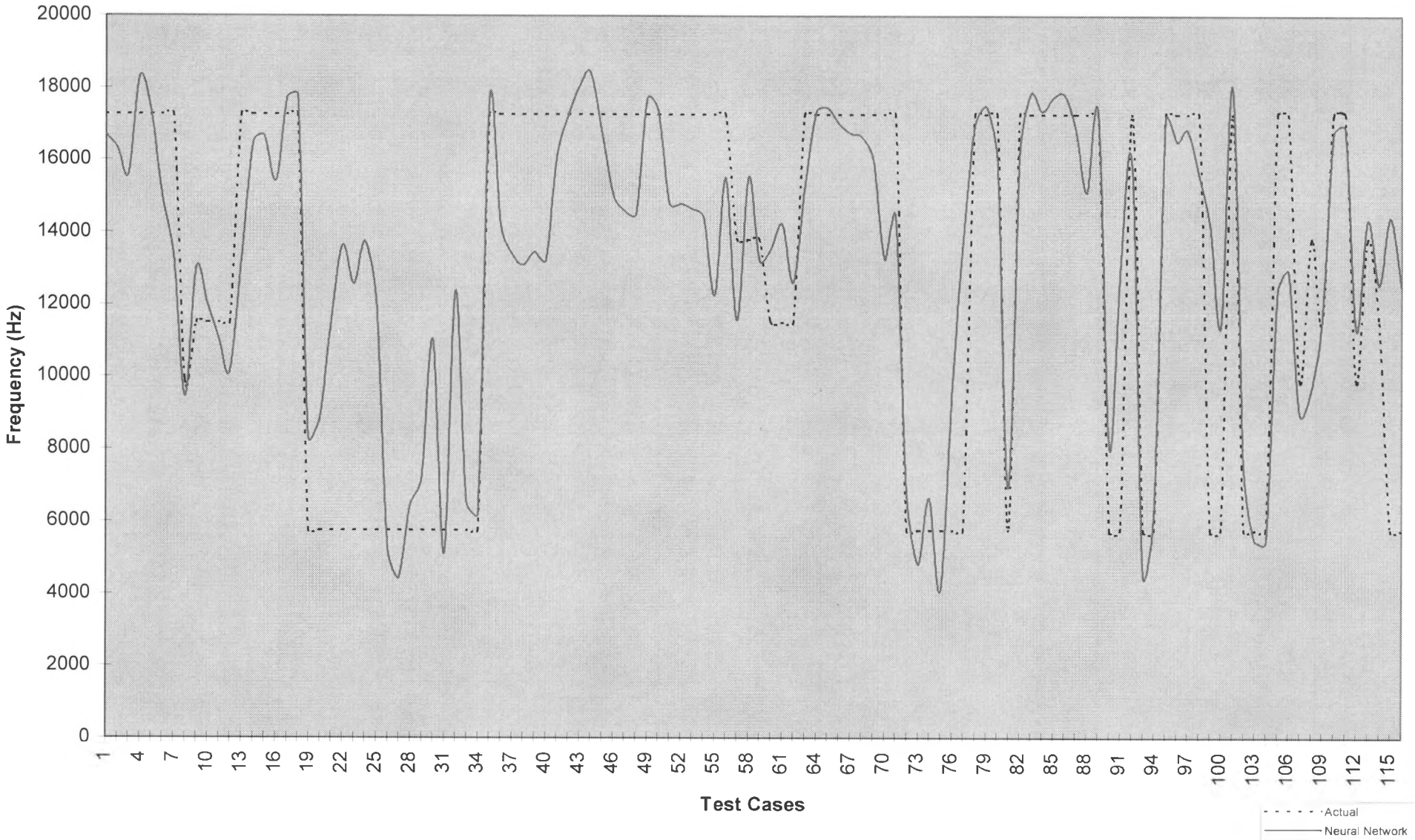


Figure D10: 10th set of noise added, training examples = 2330, testing examples =116, hidden nodes = 48, standard deviation = 0.025

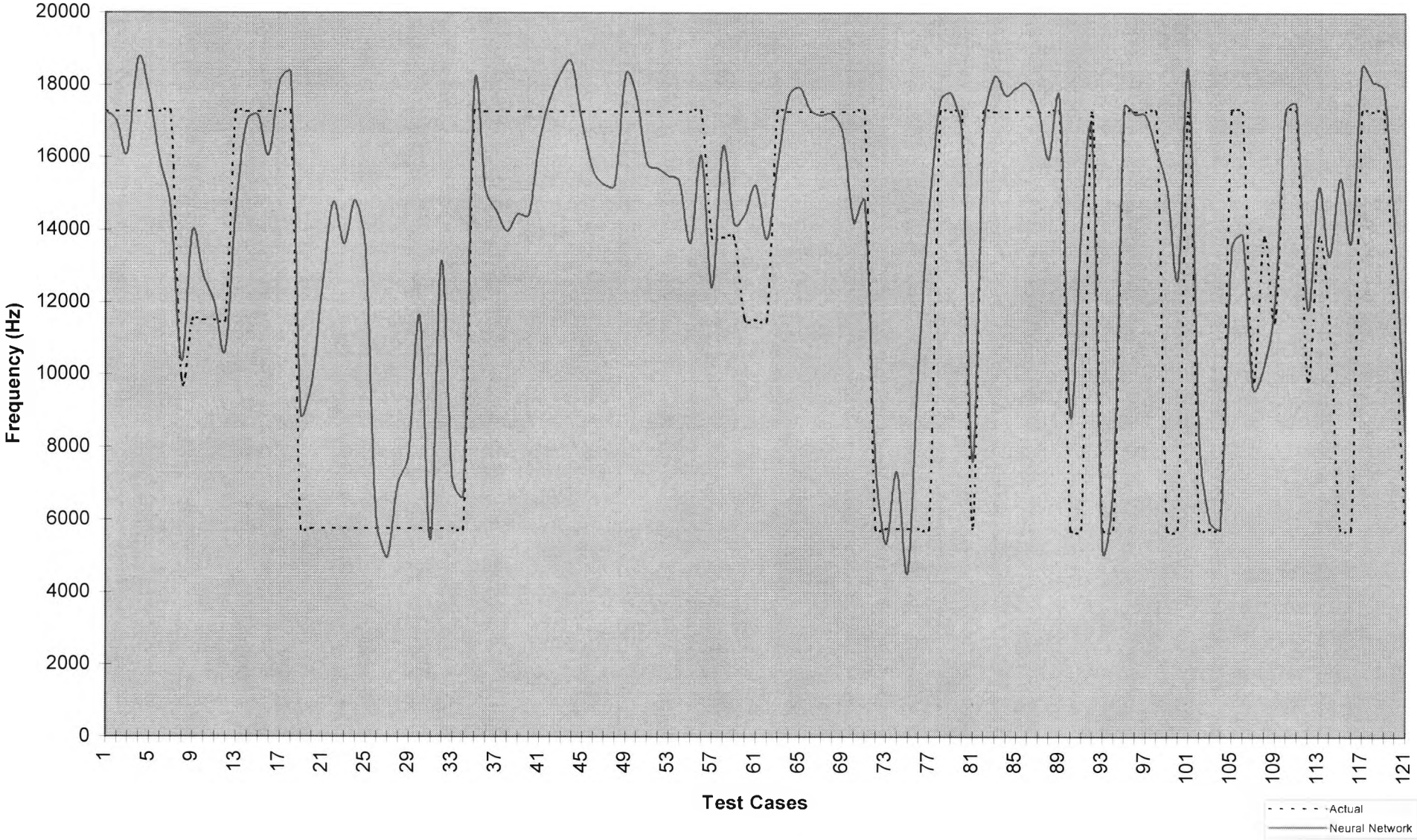


Figure D11: 11th set of noise added, training examples = 2548, testing examples = 121, hidden nodes = 50, standard deviation = 0.025

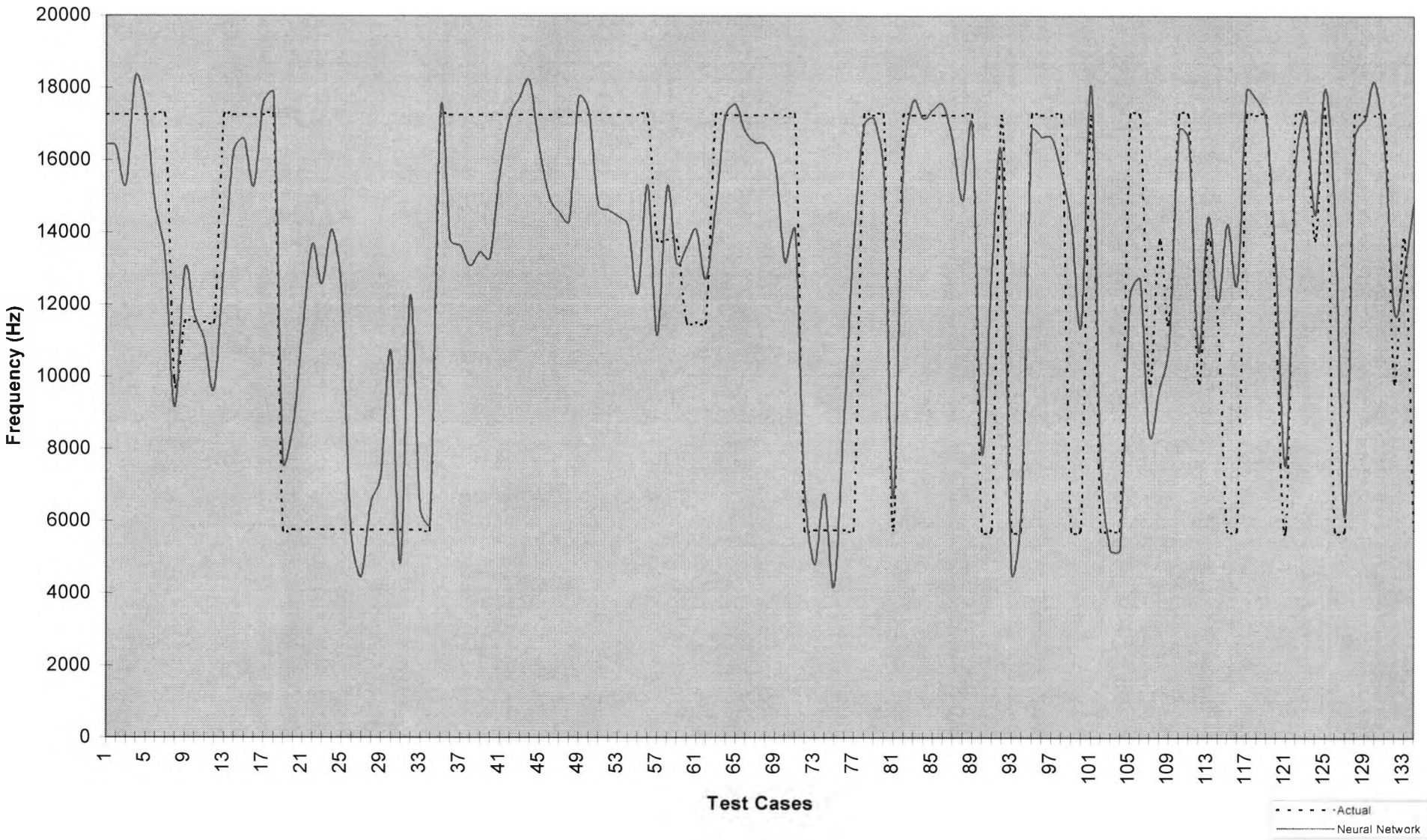


Figure D12: 13th set of noise added, training patterns =2983, test patterns =134, hidden nodes =55, standard deviation = 0.025

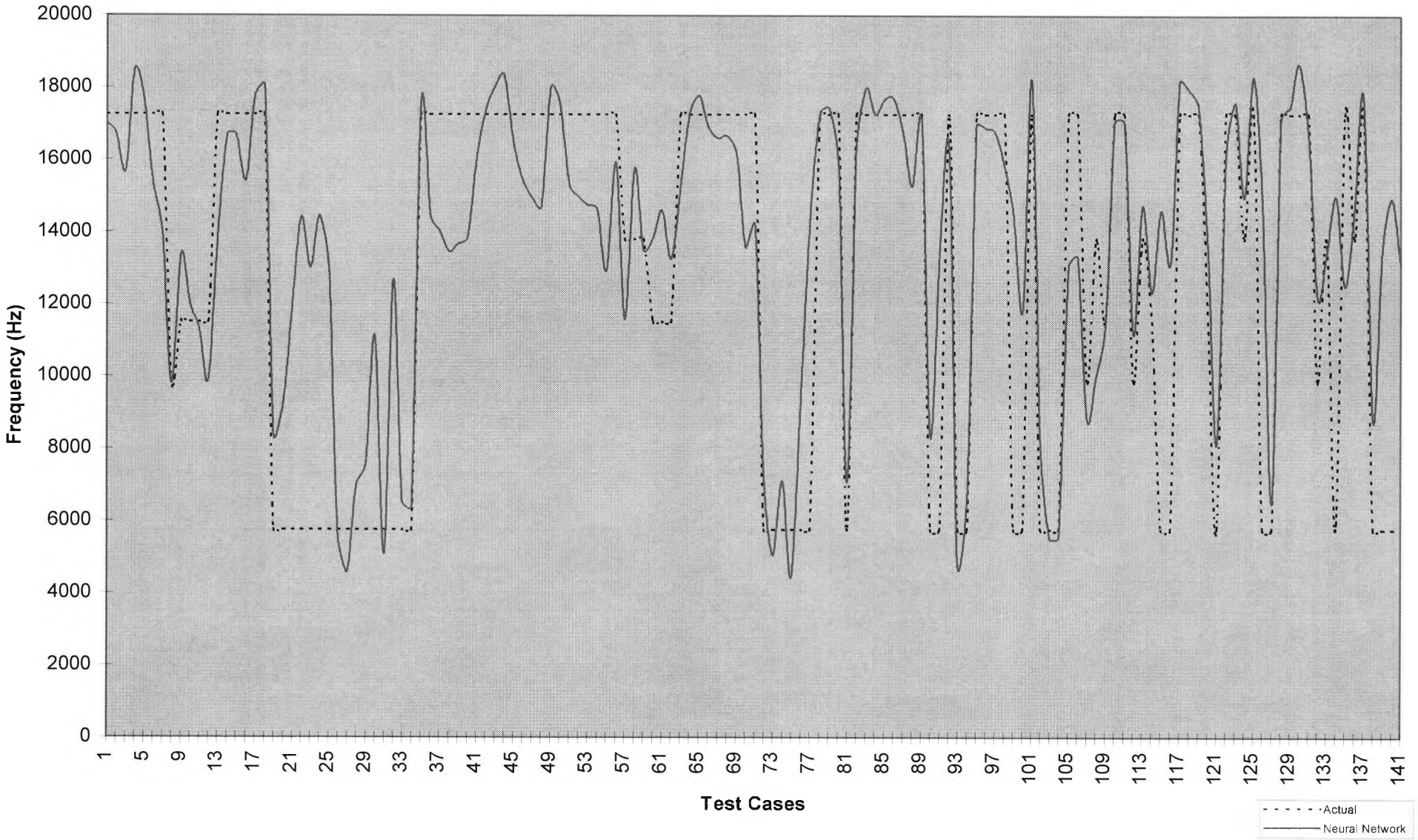


Figure D13 : 14th set of noise added, training patterns =3201, test patterns =141, hidden nodes =57, standard deviation = 0.025

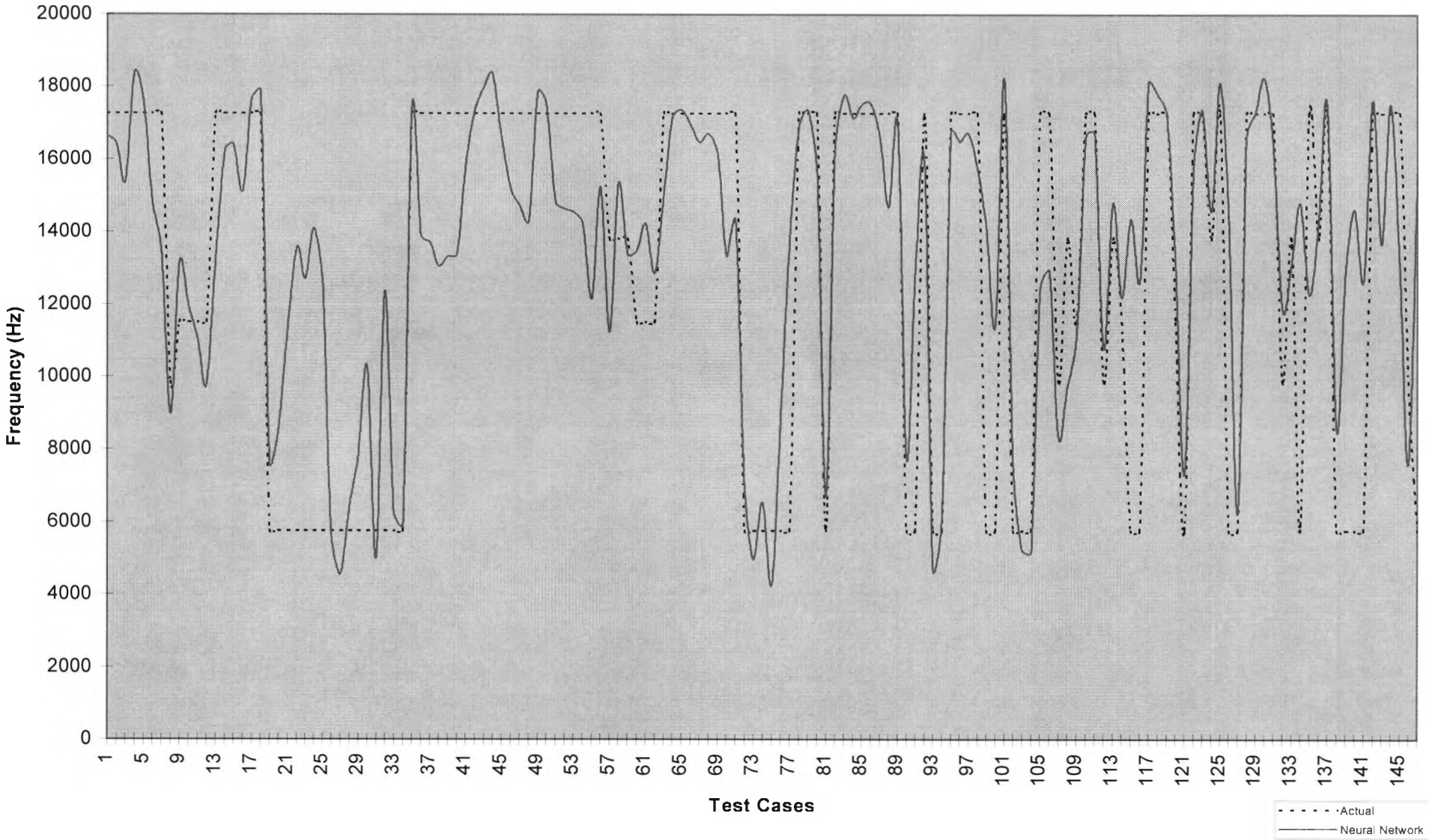


Figure D14: 15th set of noise added, training patterns = 3420, test patterns = 147, hidden nodes = 58, standard deviation = 0.025

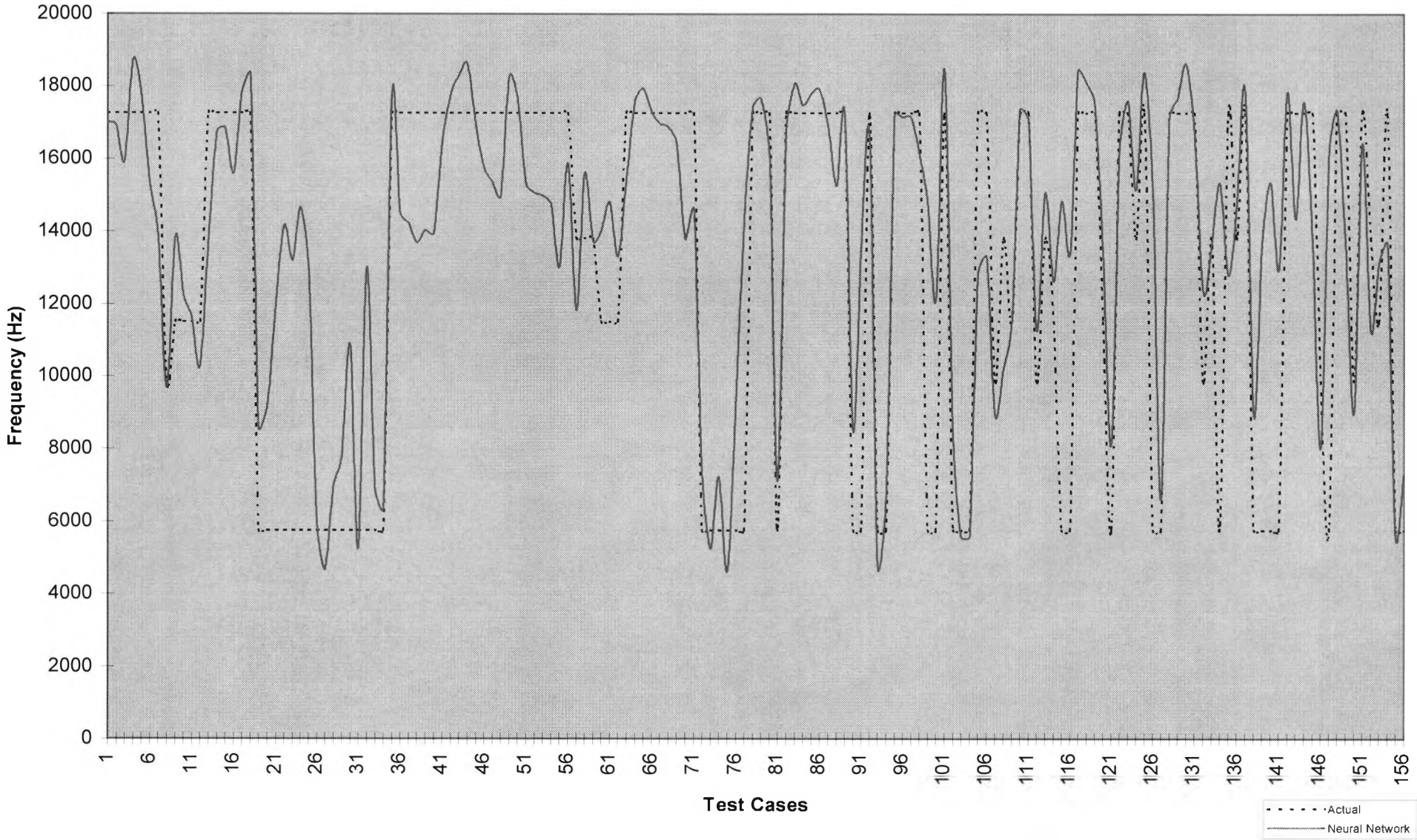


Figure D15: 16th set of noise added, training patterns = 3615, test patterns =156, hidden nodes =60, standard deviation = 0.025

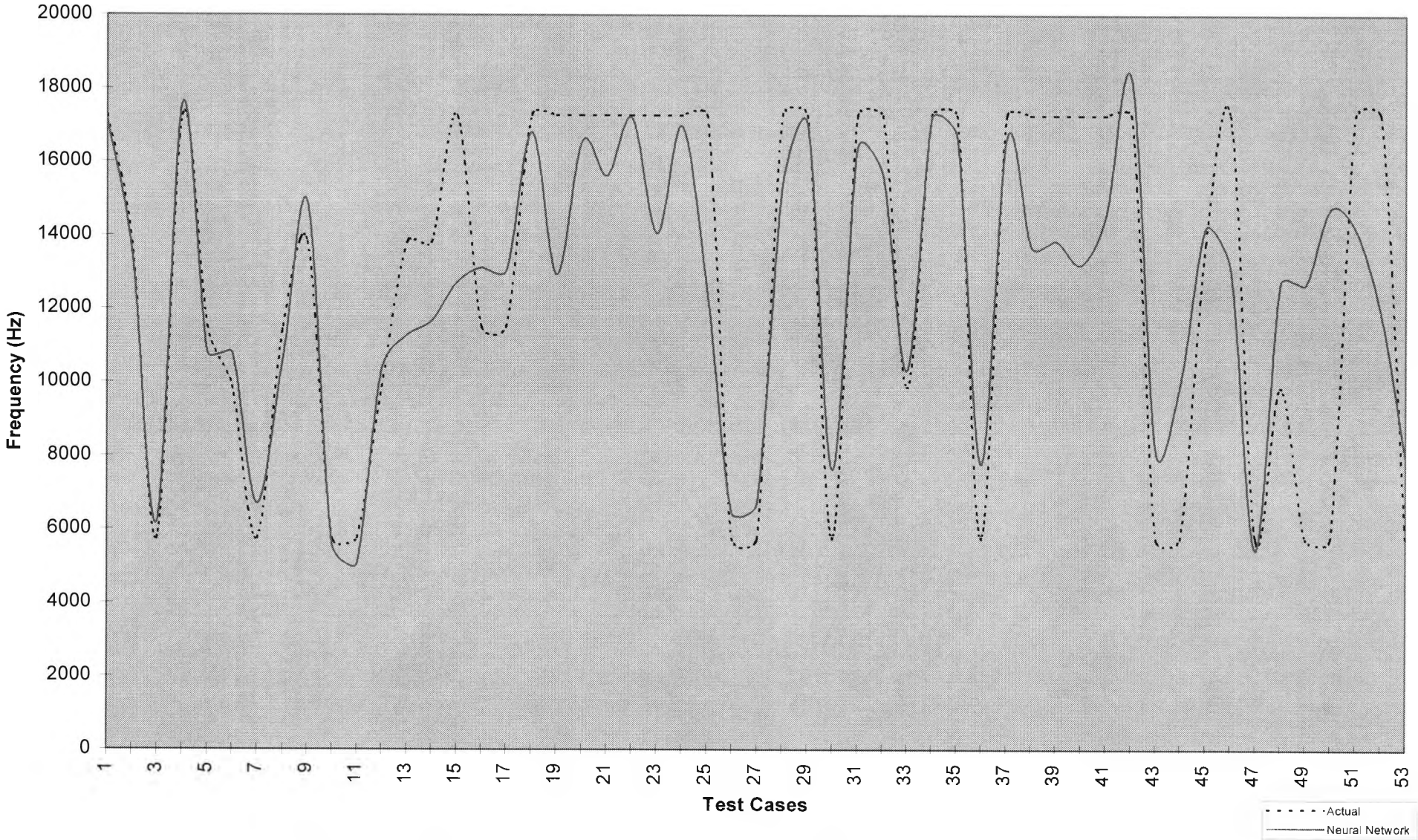


Figure D16: Momentum = 0.5, learning rate(hidden)=0.3, learning rate(output) = 0.15

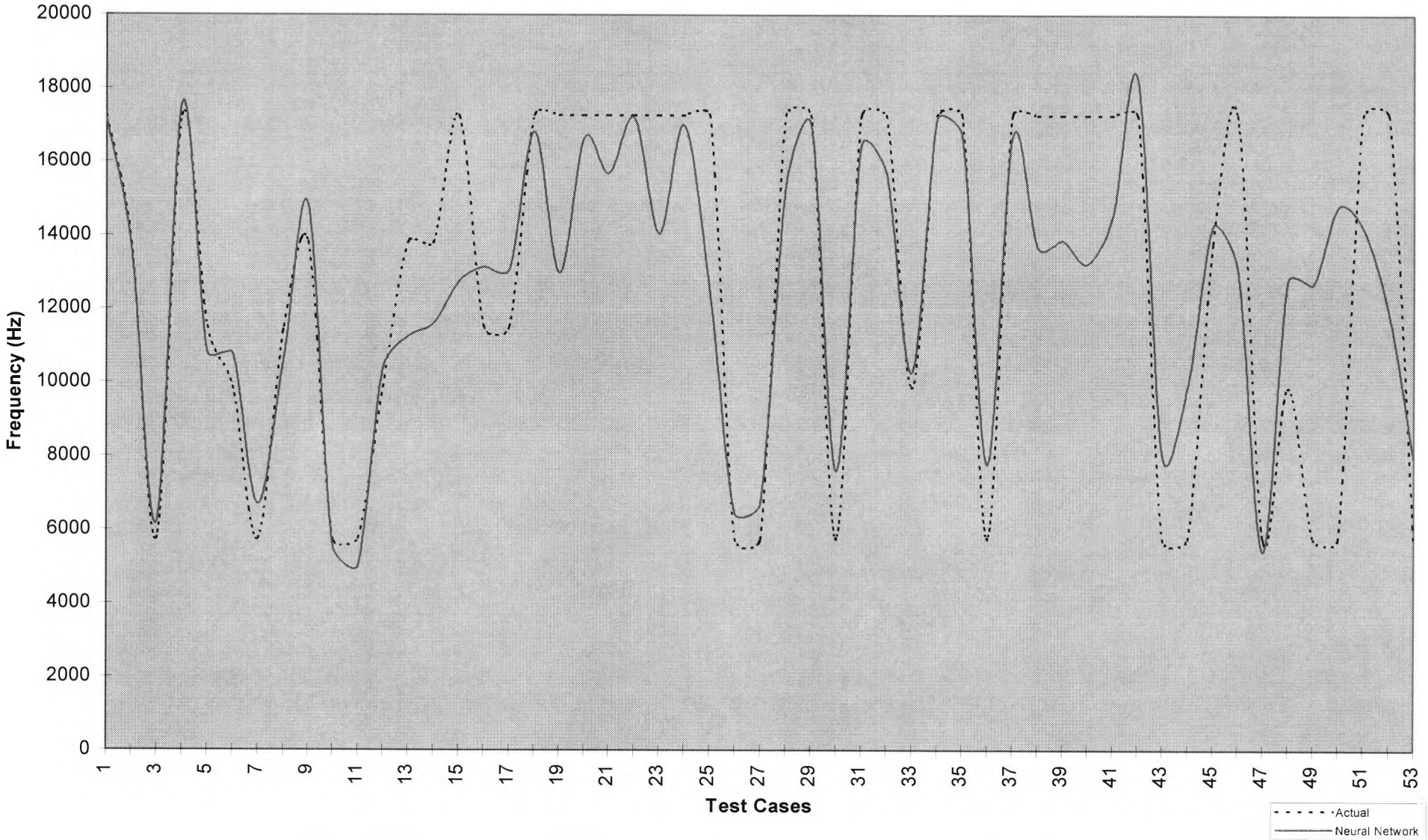


Figure D17: Momentum = 0.6, learning rate(hidden) = 0.25, learning rate(output) = 0.15

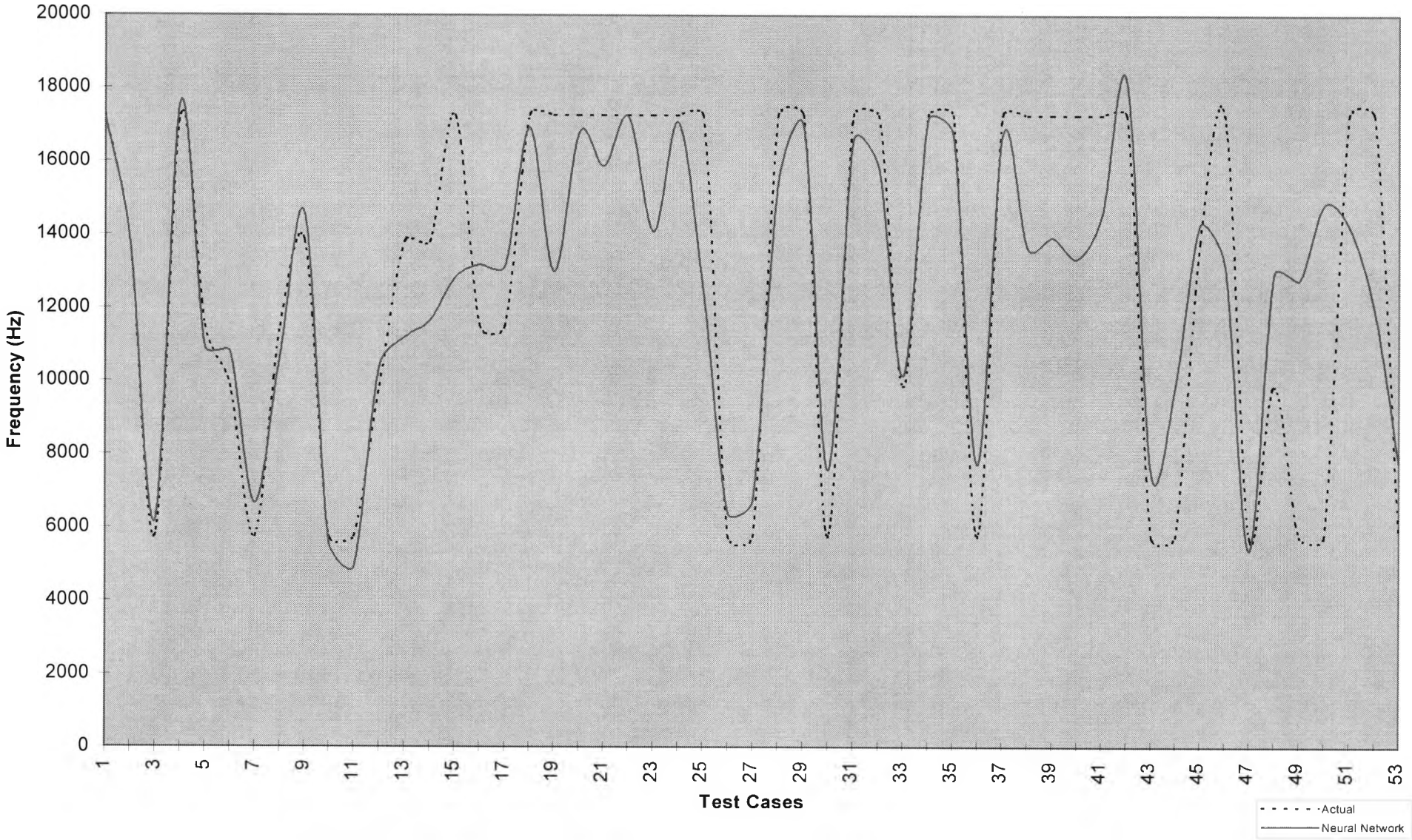


Figure D18: Momentum = 0.7, learning rate(hidden) = 0.25, learning rate(output) = 0.15

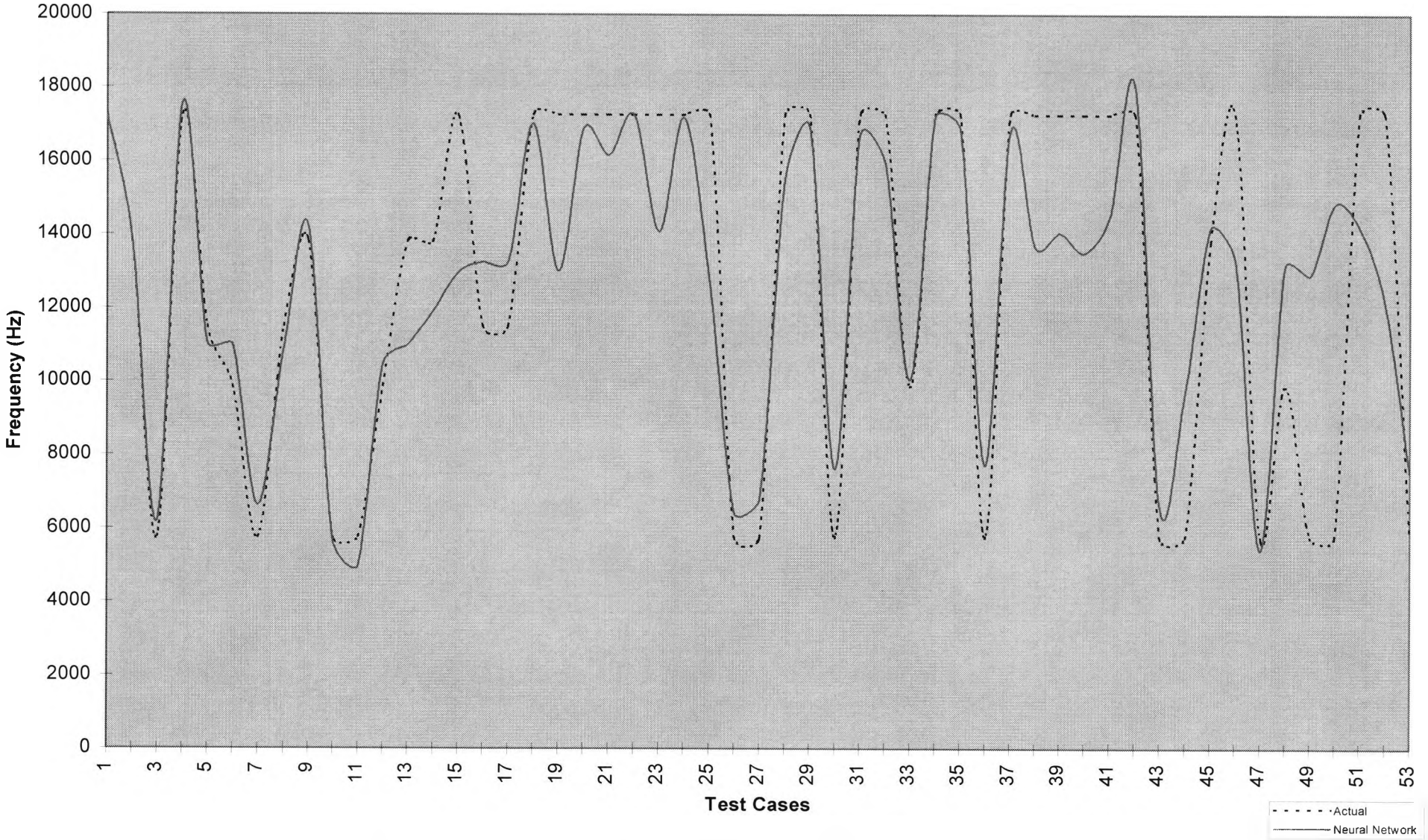


Figure D19: Momentum = 0.8, learning rate(hidden) = 0.25, learning rate(output) = 0.15

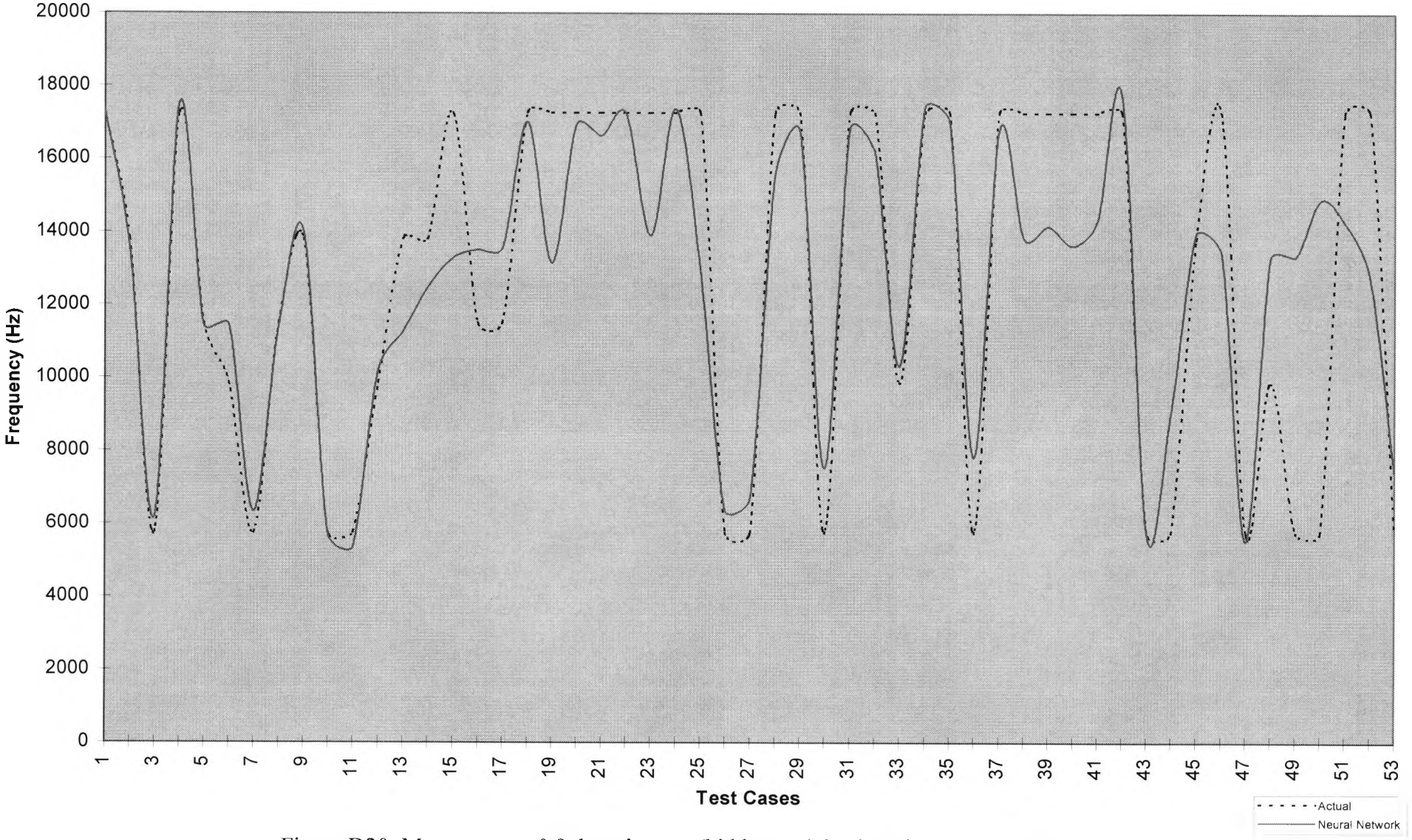


Figure D20: Momentum = 0.9, learning rate(hidden) = 0.25, learning rate(output) = 0.15

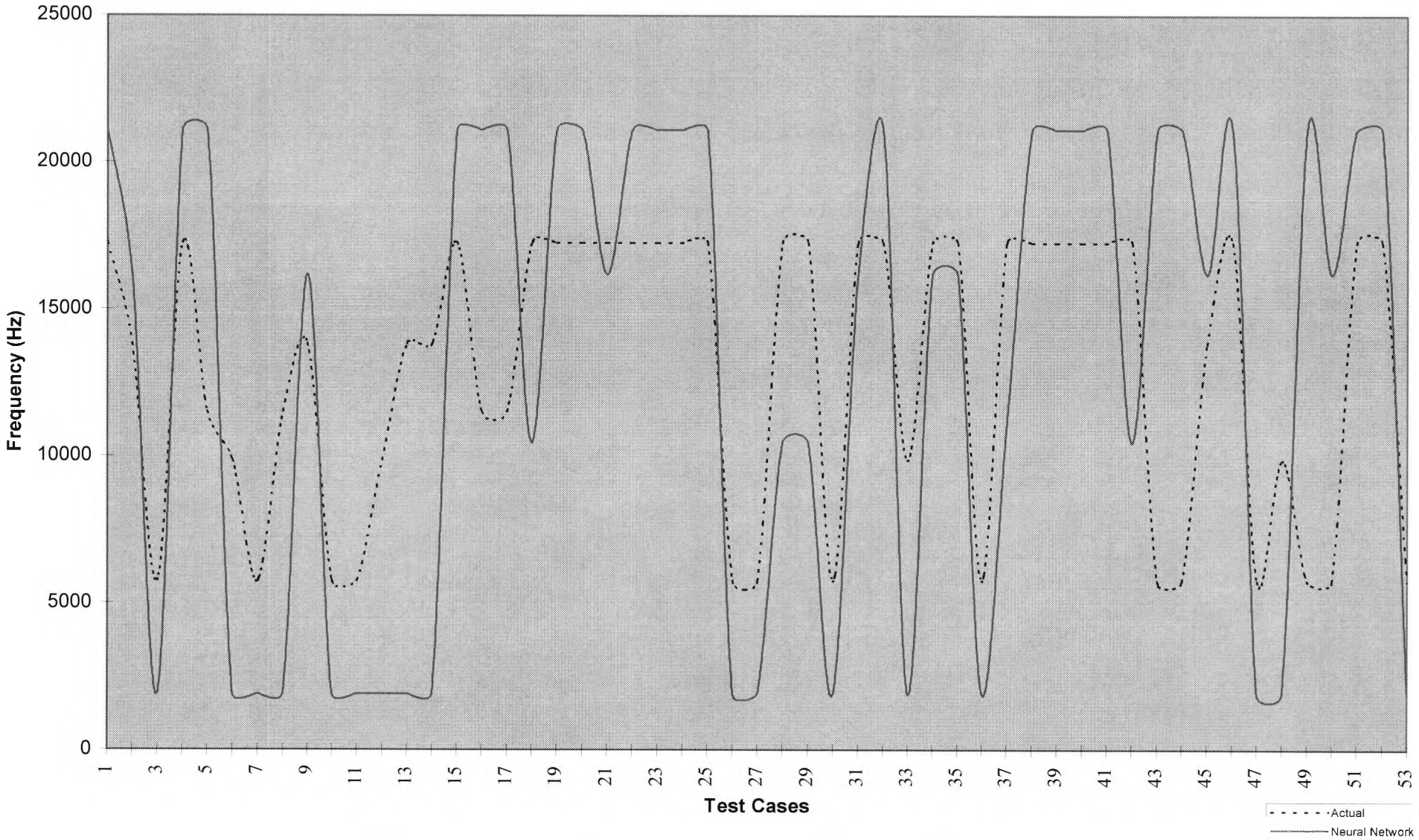


Figure 21: Momentum = 1.0, learning rate(hidden) = 0.25, learning rate(output) = 0.15

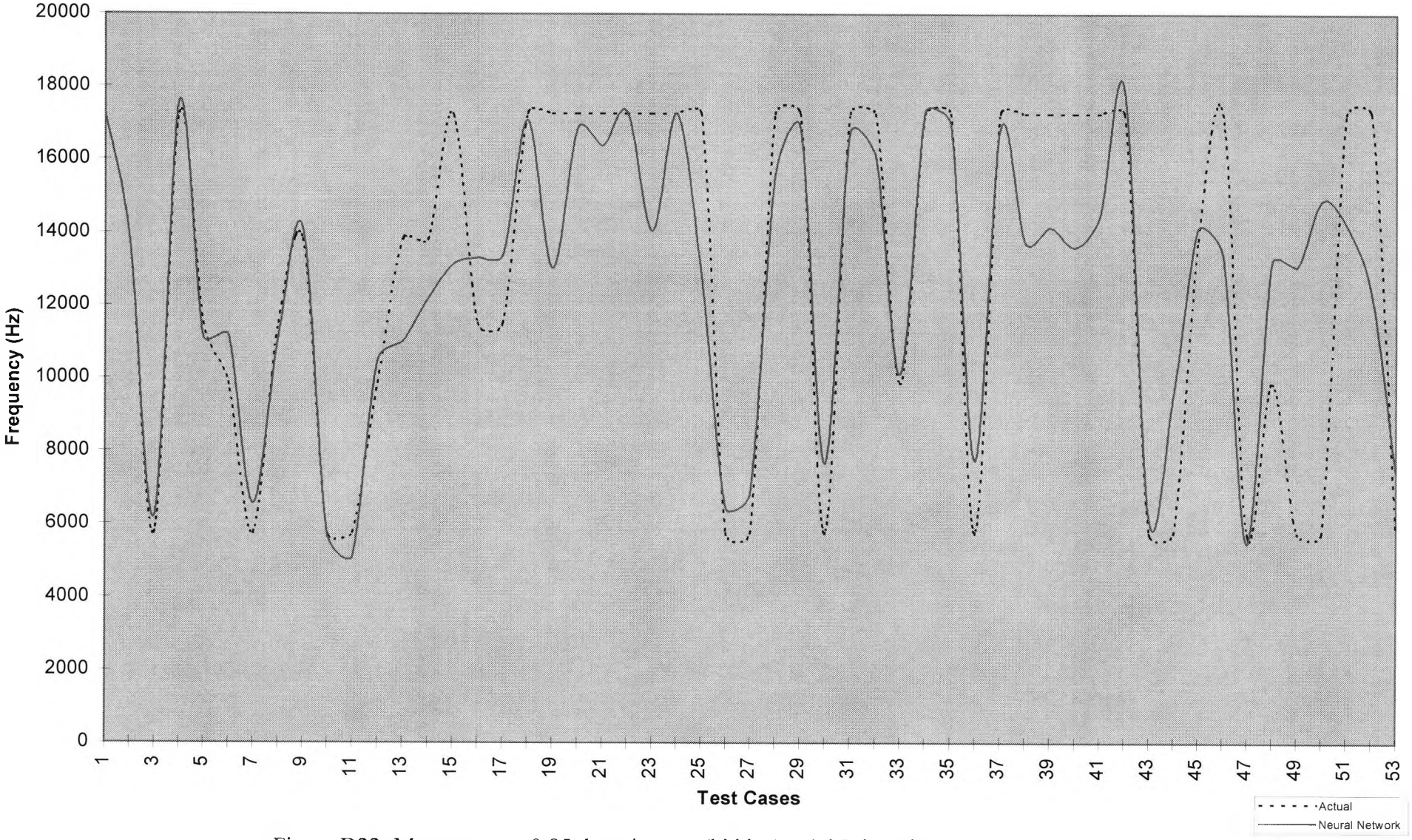


Figure D22: Momentum = 0.85, learning rate(hidden) = 0.25, learning rate(output) = 0.15

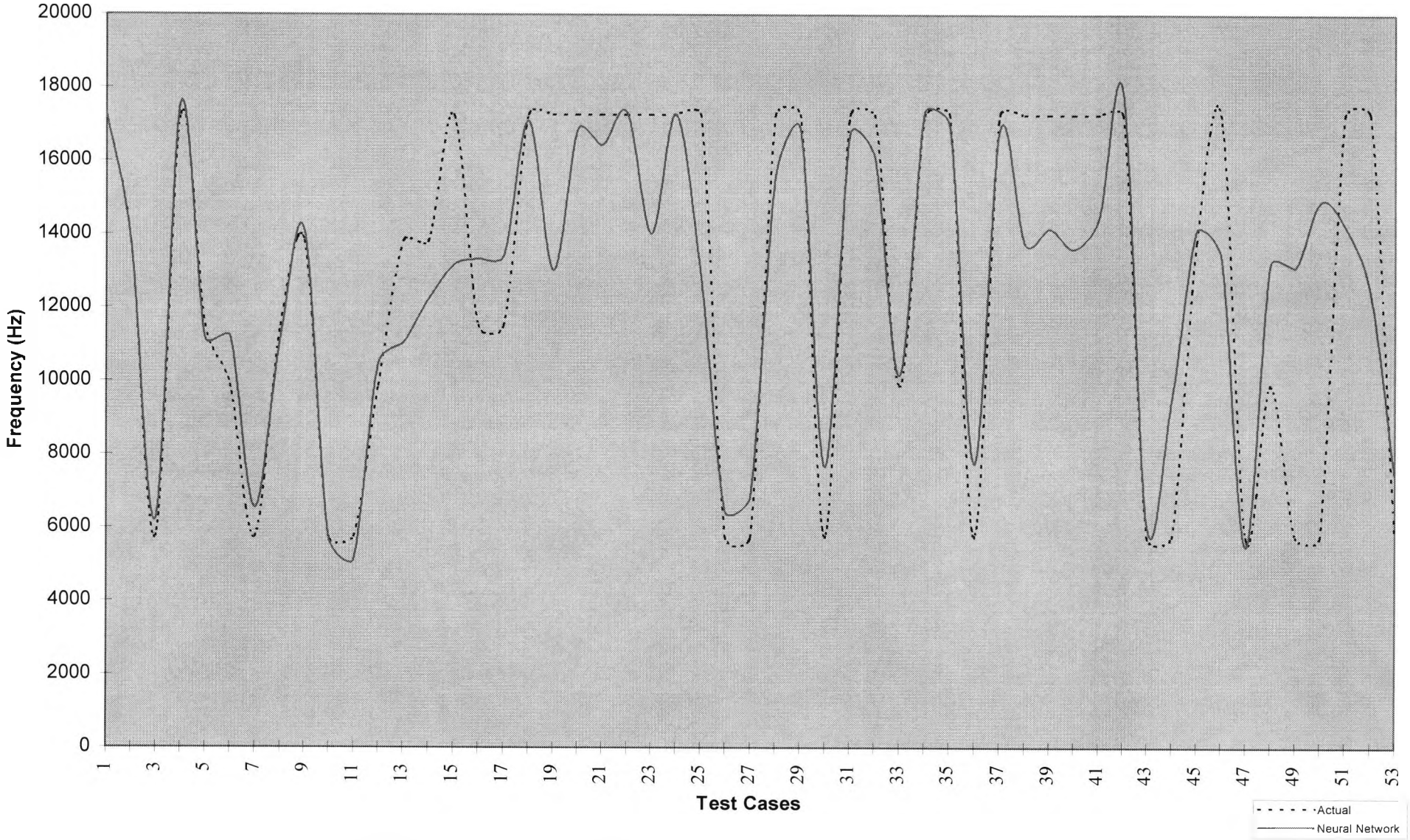


Figure D23: Momentum = 0.86, learning rate(hidden) = 0.25, learning rate(output) = 0.15

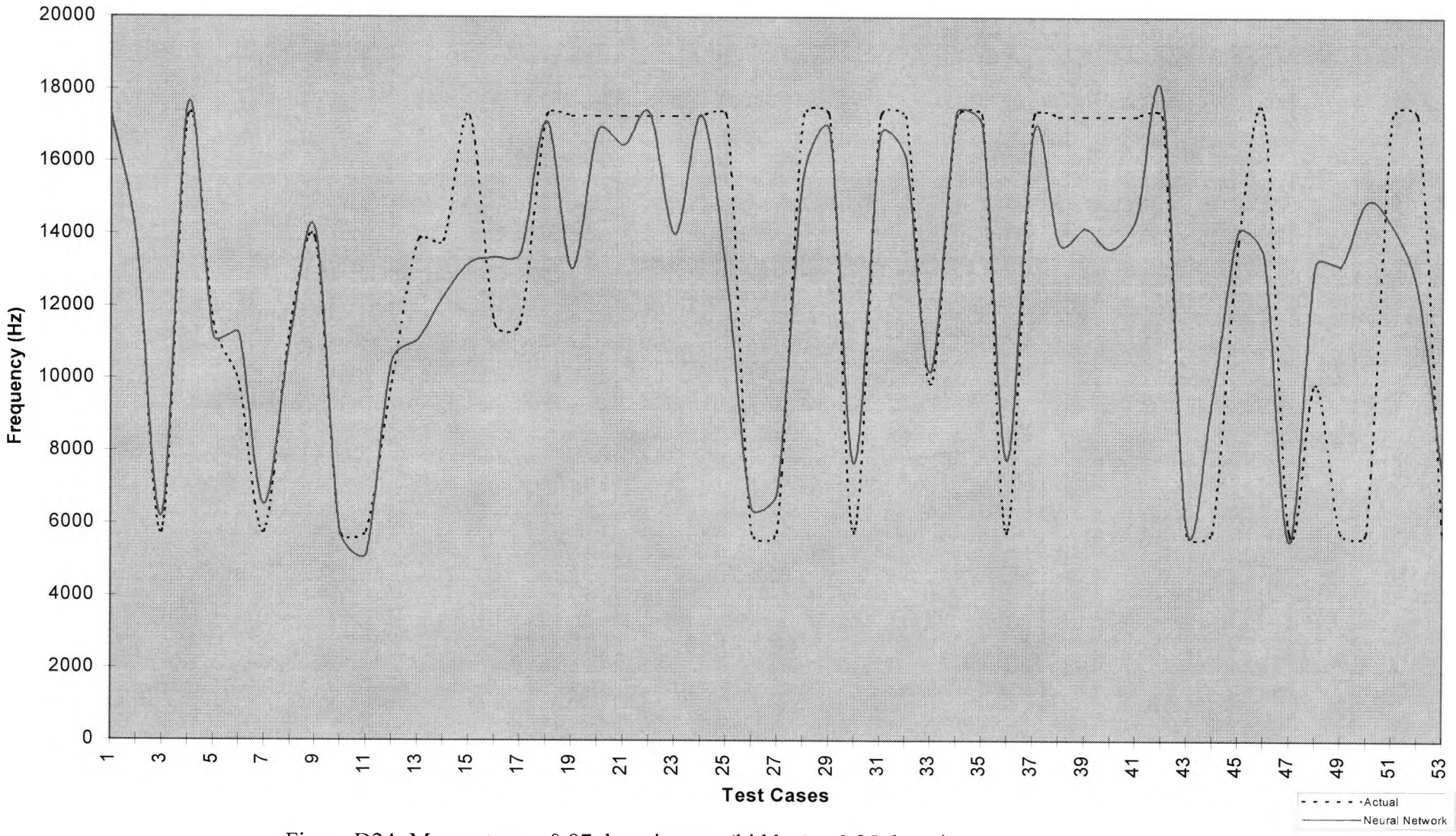


Figure D24: Momentum = 0.87, learning rate(hidden) = 0.25, learning rate(output) = 0.15

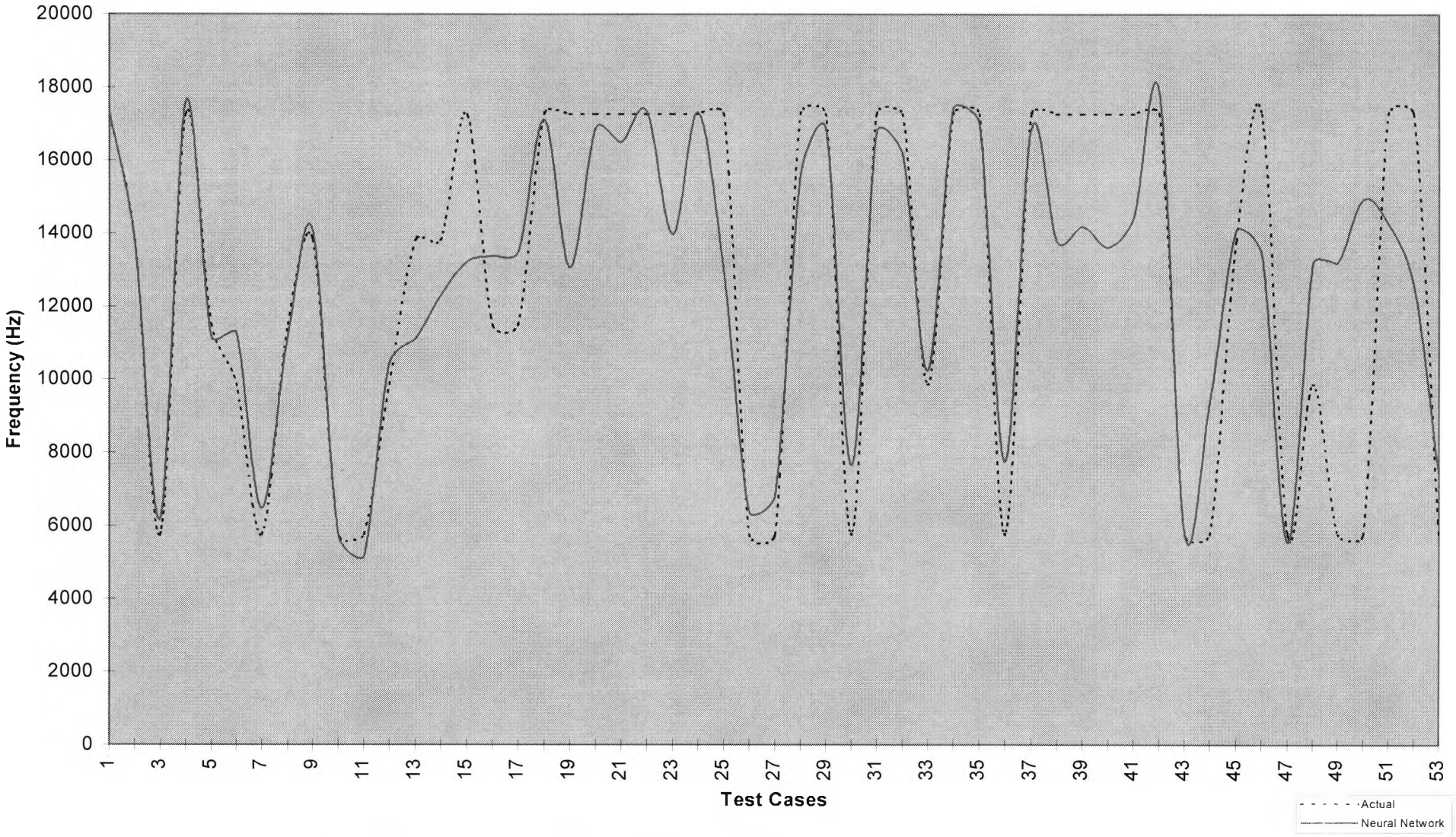


Figure D25: Momentum = 0.88, learning rate(hidden) = 0.25, learning rate(output) = 0.15

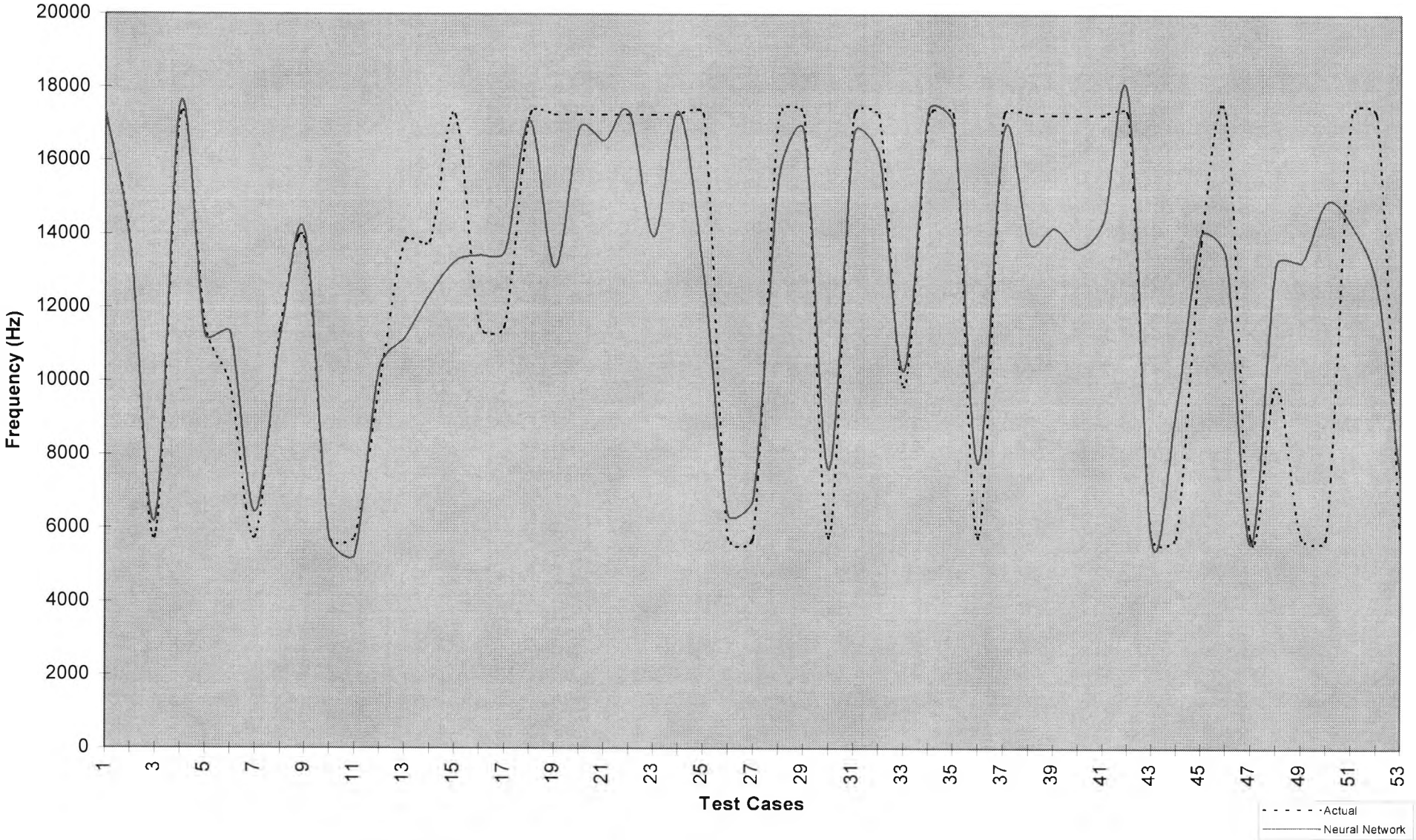


Figure D26: Momentum = 0.89, learning rate = 0.25, learning rate = 0.15

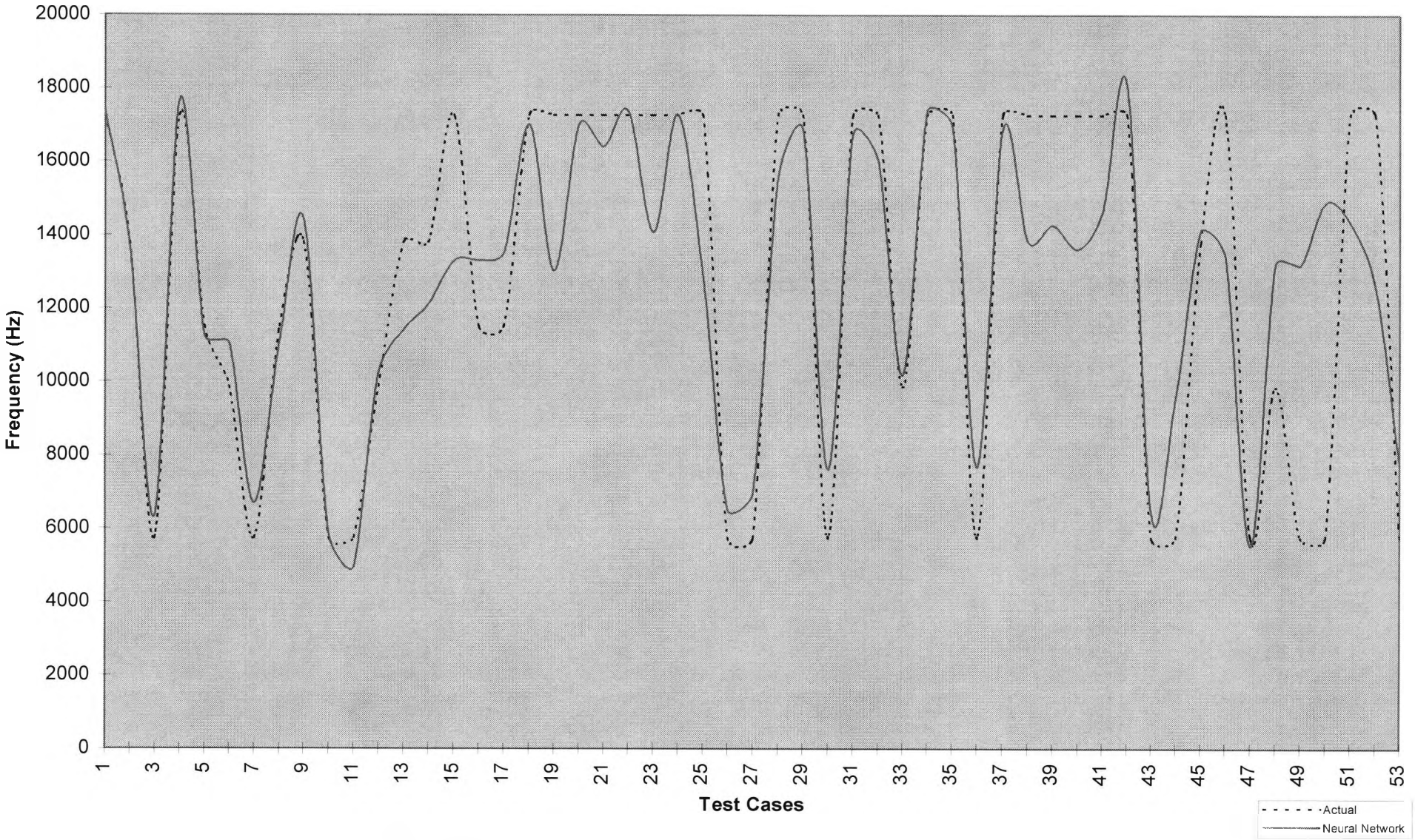


Figure D27: Momentum = 0.89, learning rate = 0.25, learning rate(output) = 0.05

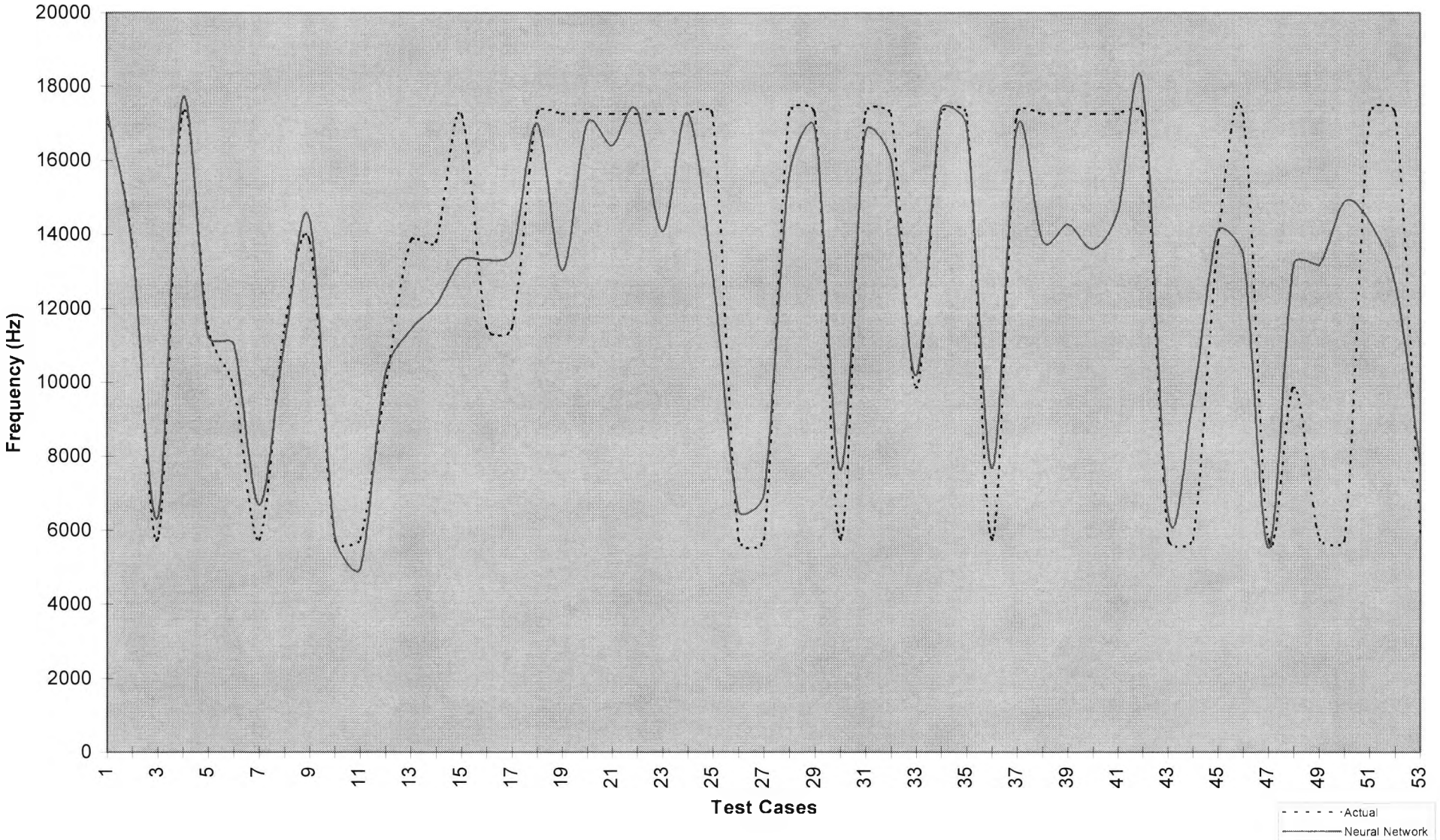


Figure D28: Momentum = 0.89, learning rate(hidden) = 0.25, learning rate(output) = 0.1

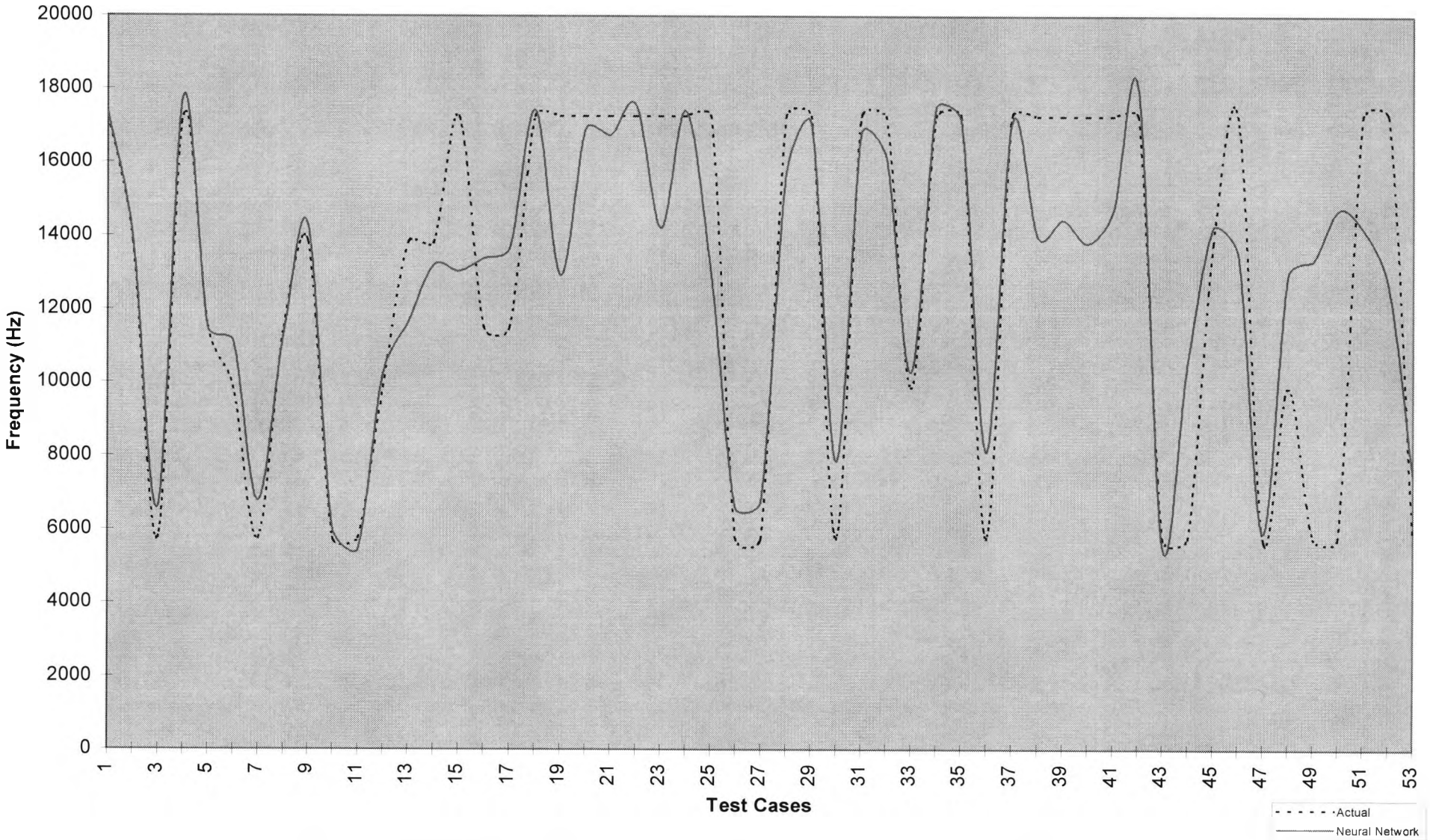


Figure D29: Momentum = 0.89, learning rate(hidden) = 0.25, learning rate(output) = 0.11

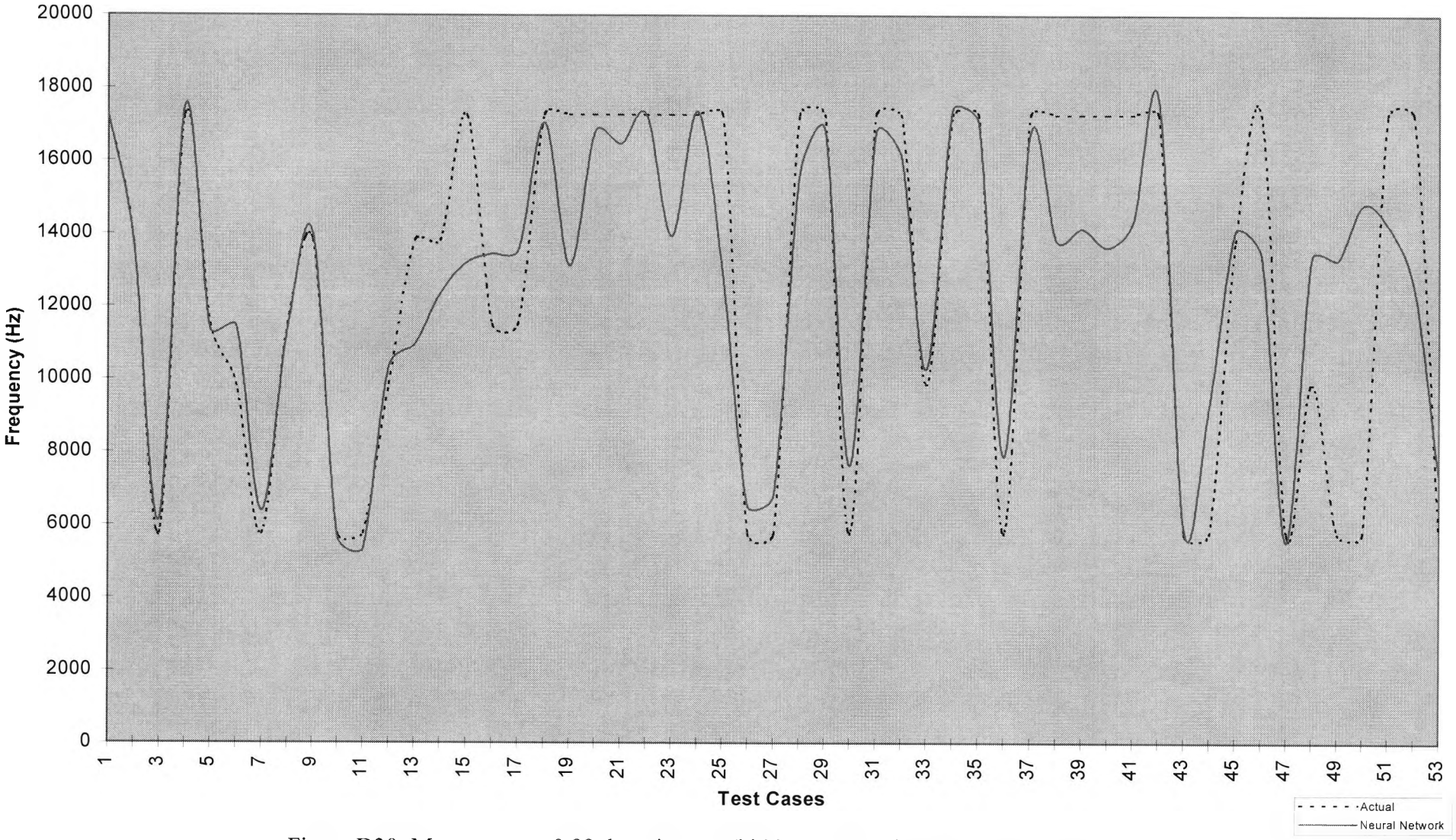


Figure D30: Momentum = 0.89, learning rate(hidden) = 0.25, learning rate(output) = 0.2

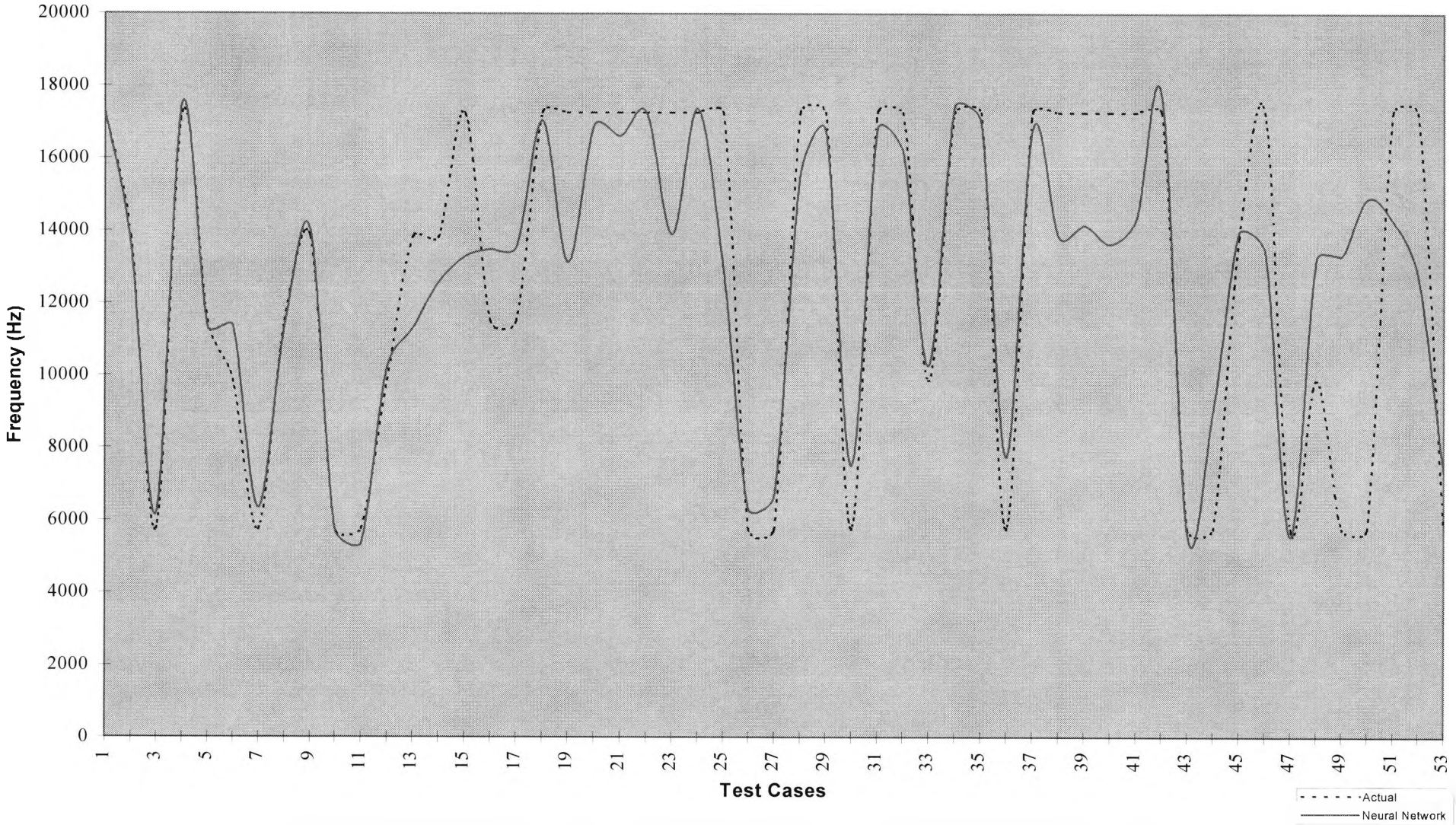


Figure D31: Momentum = 0.89, learning rate(hidden) = 0.3, learning rate(output) = 0.15

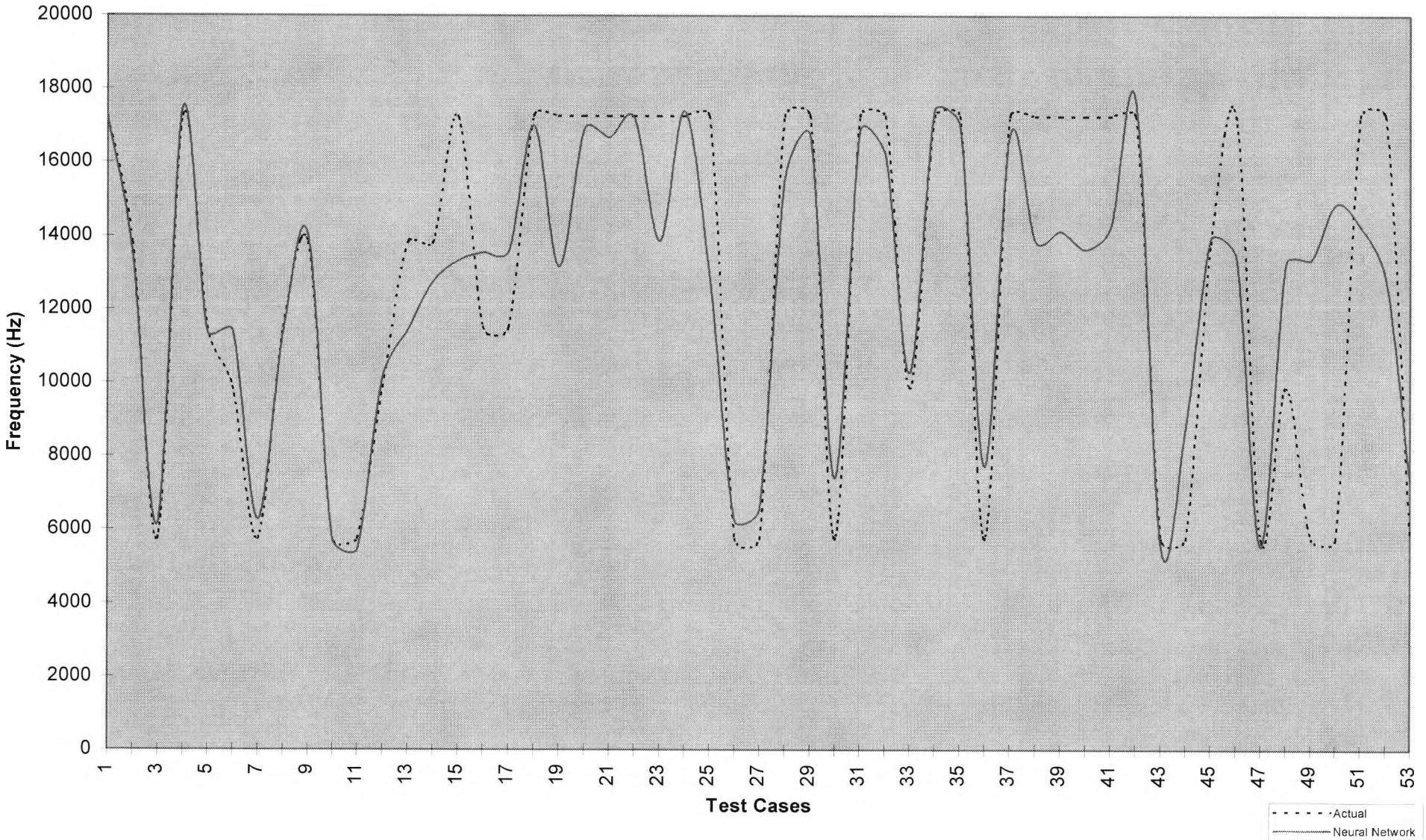


Figure D32: Momentum = 0.89, learning rate(hidden) =0.35, learning rate(output) = 0.15

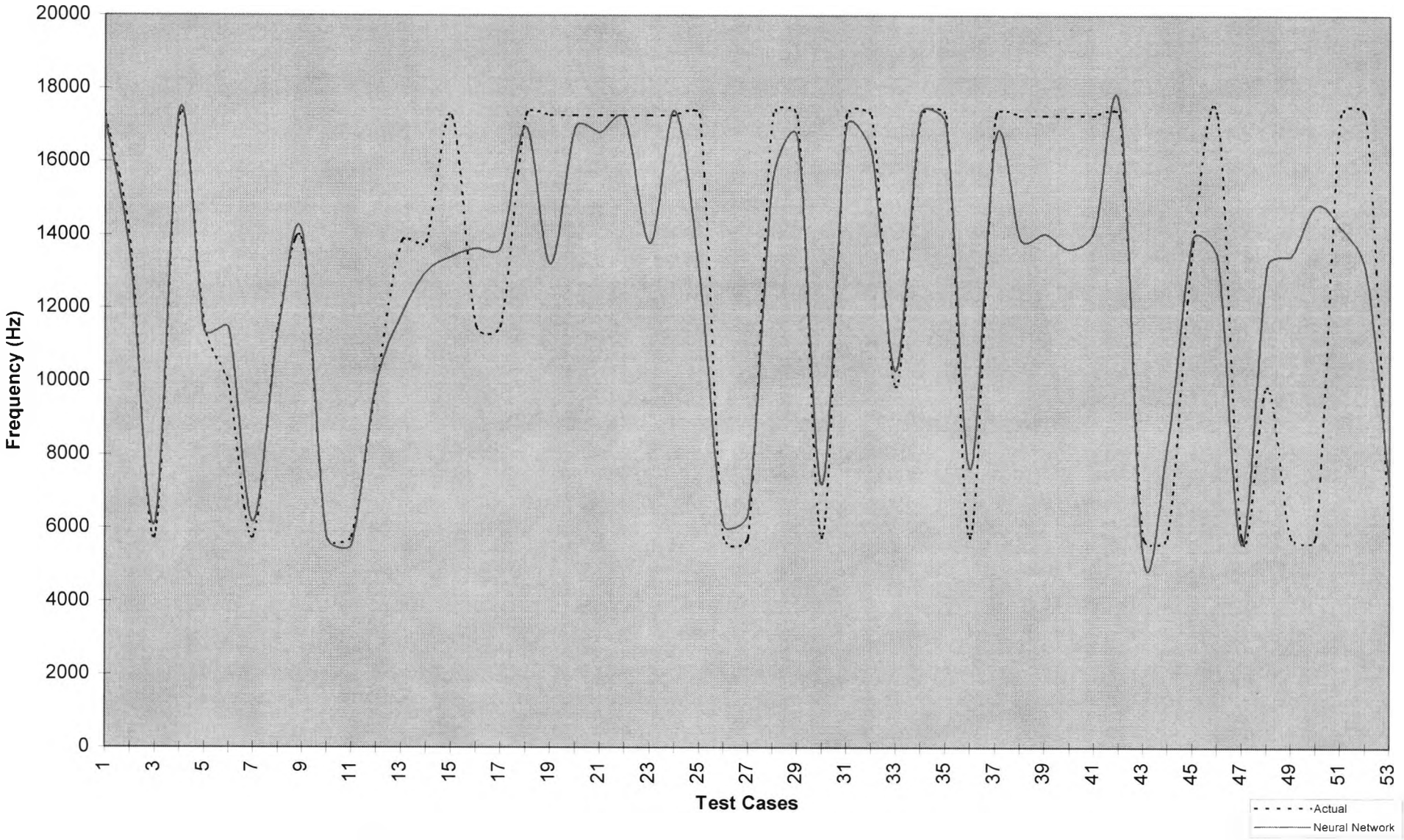


Figure D33: Momentum = 0.89, learning rate(hidden) = 0.5, learning rate(output) = 0.15

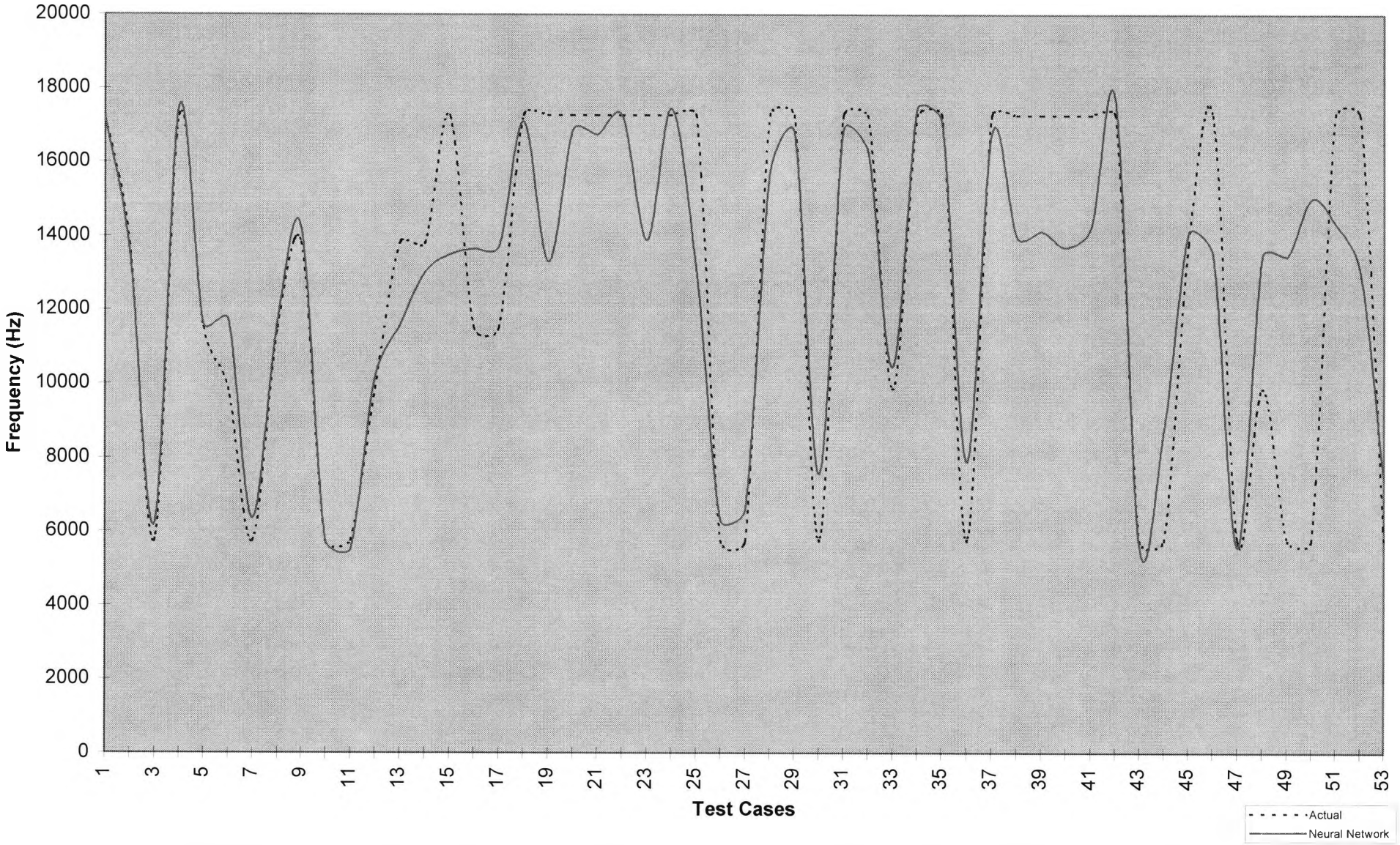


Figure D34: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.15, hidden nodes = 62

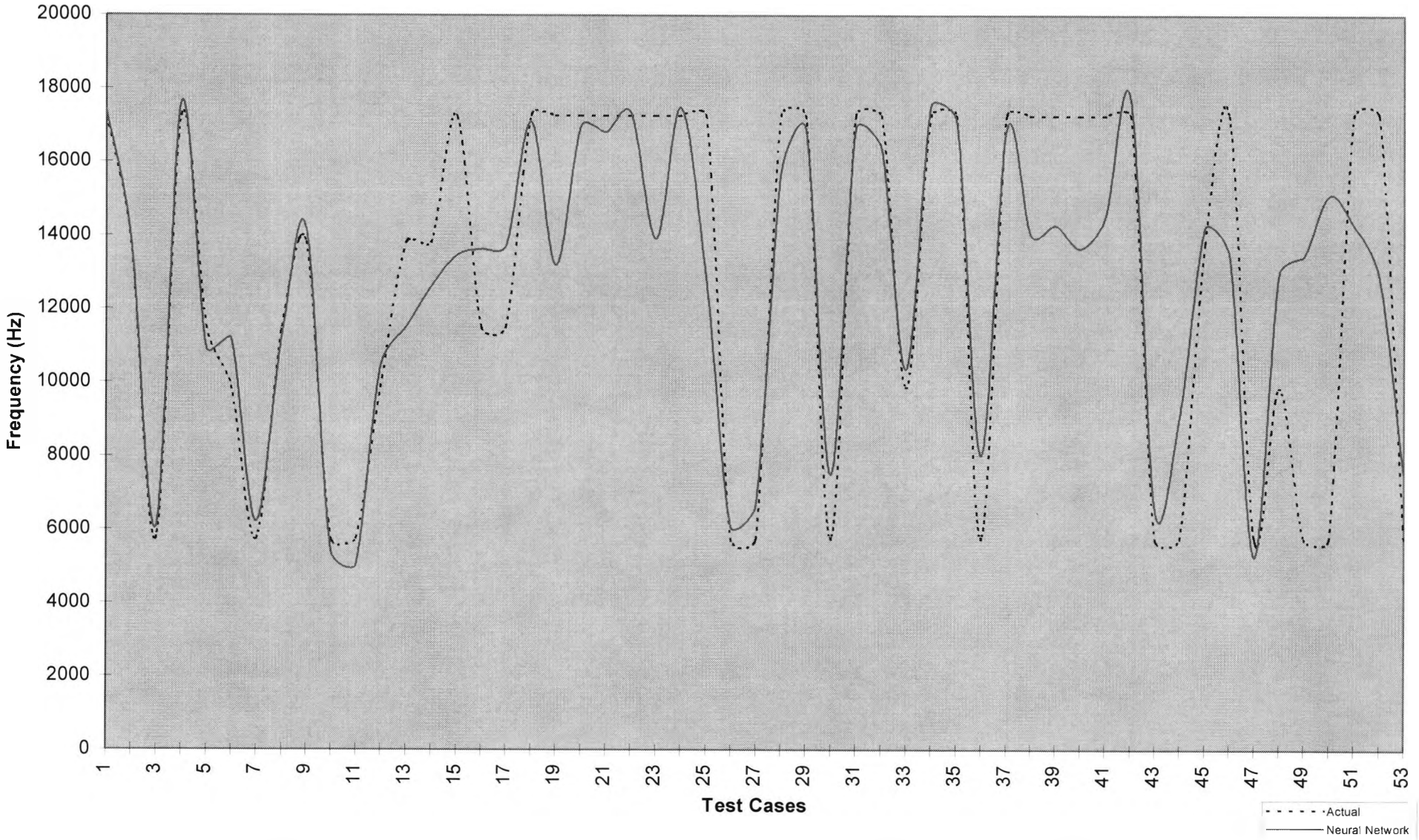


Figure D35: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.15, hidden nodes = 60

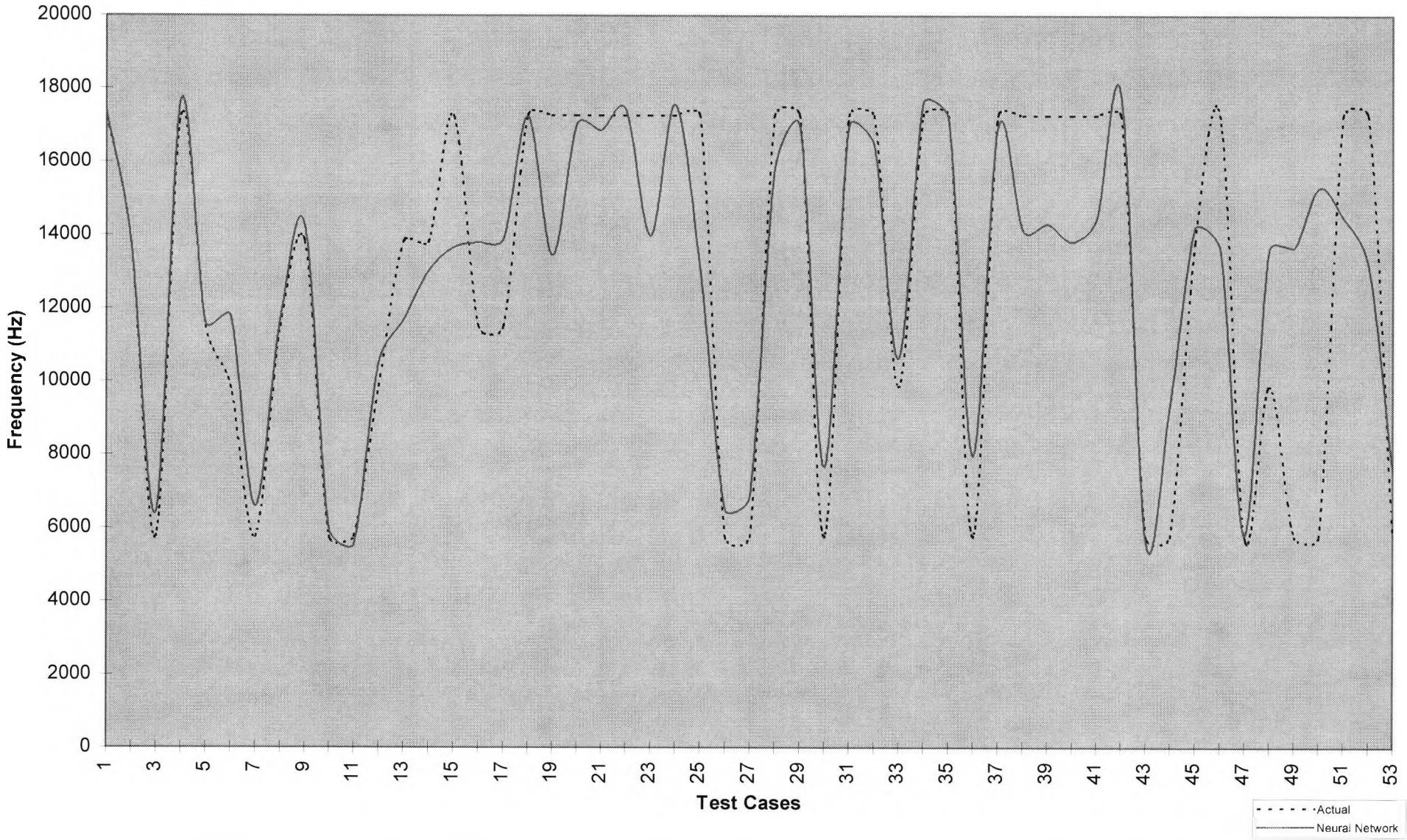


Figure D36: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.15, hidden nodes = 59

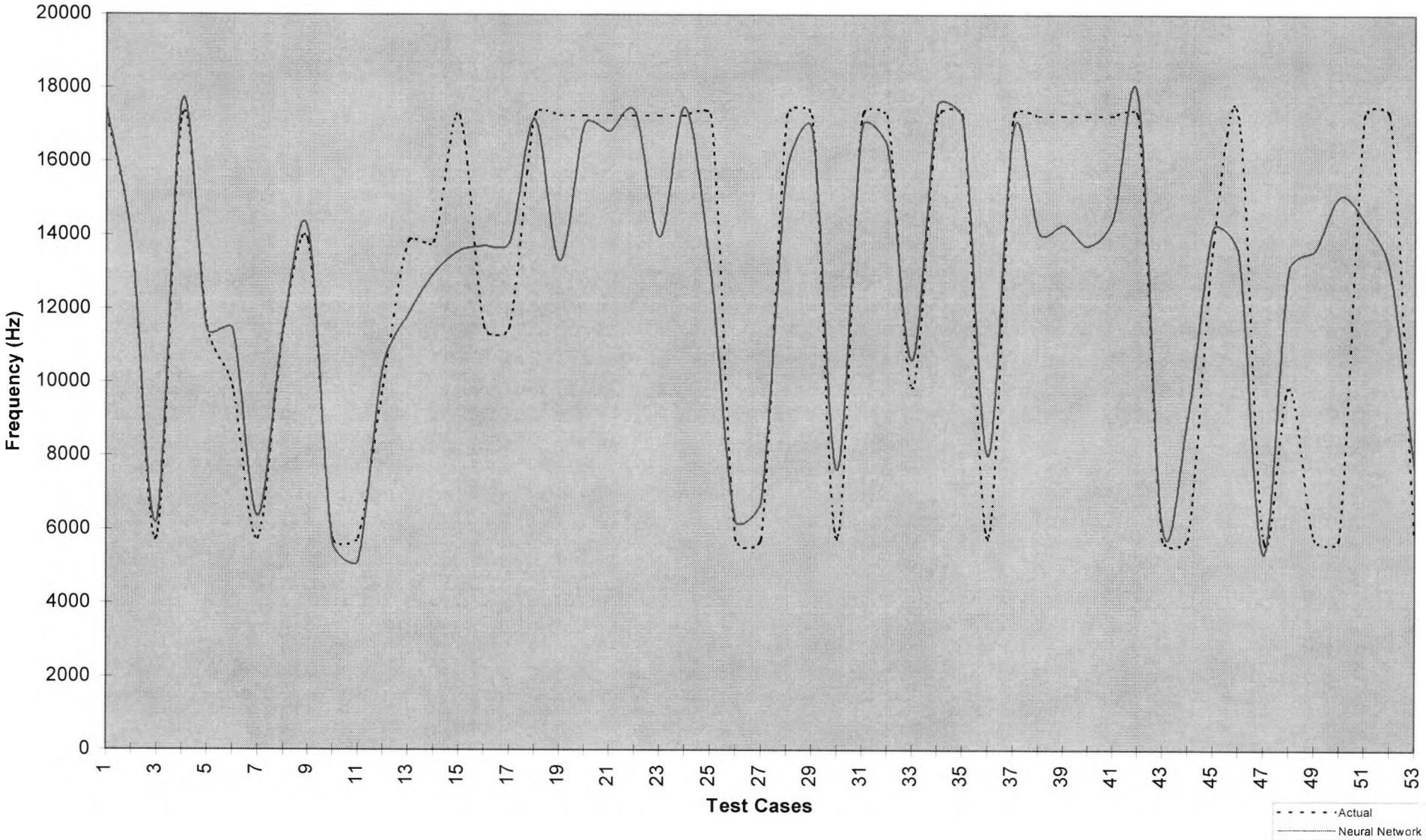


Figure D37: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.15, hidden nodes = 63

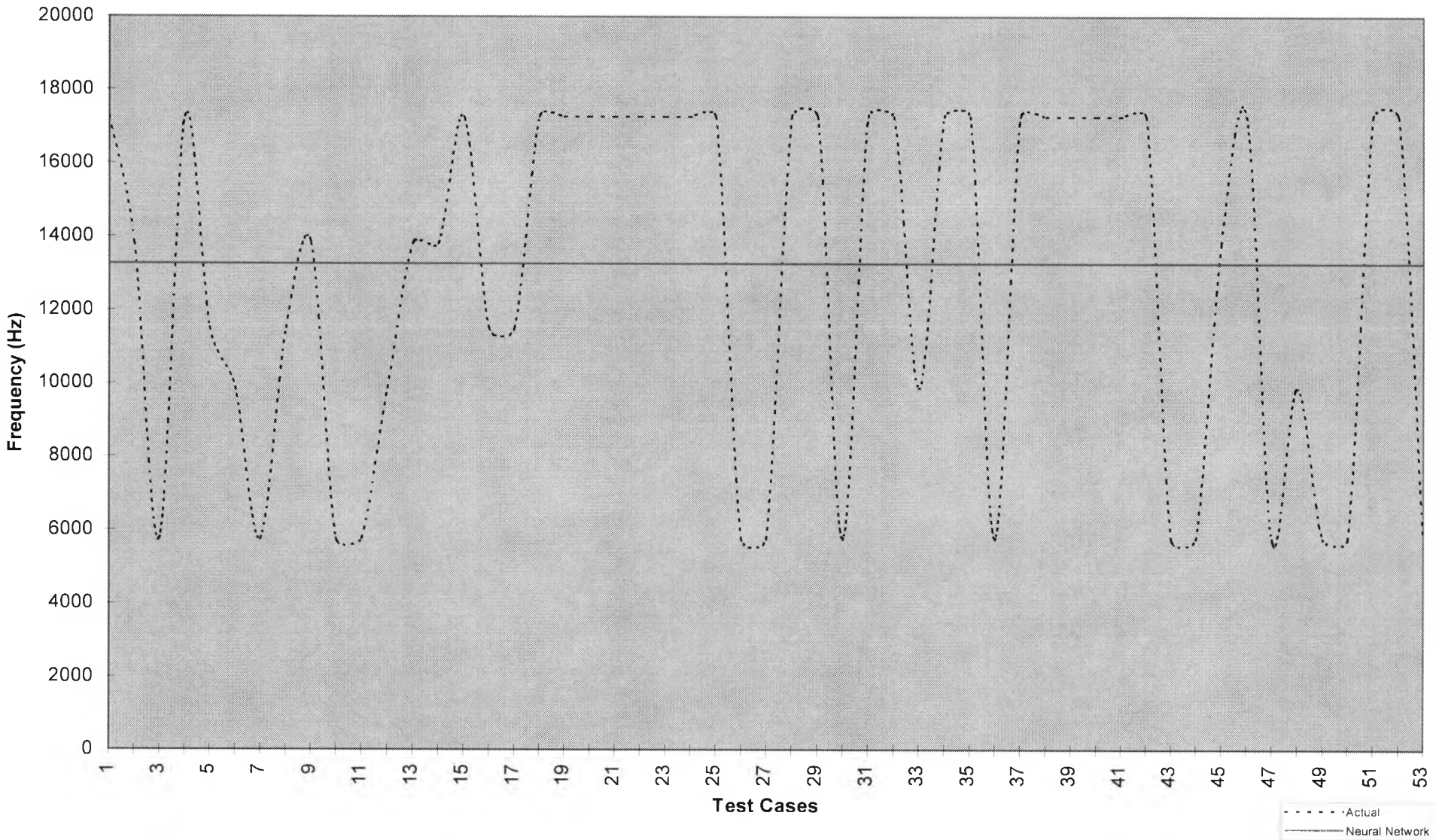


Figure D38: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.15, hidden nodes = 61, transfer function (output) = linear, transfer function(hidden) = sigmoid, transfer function(input) = sigmoid

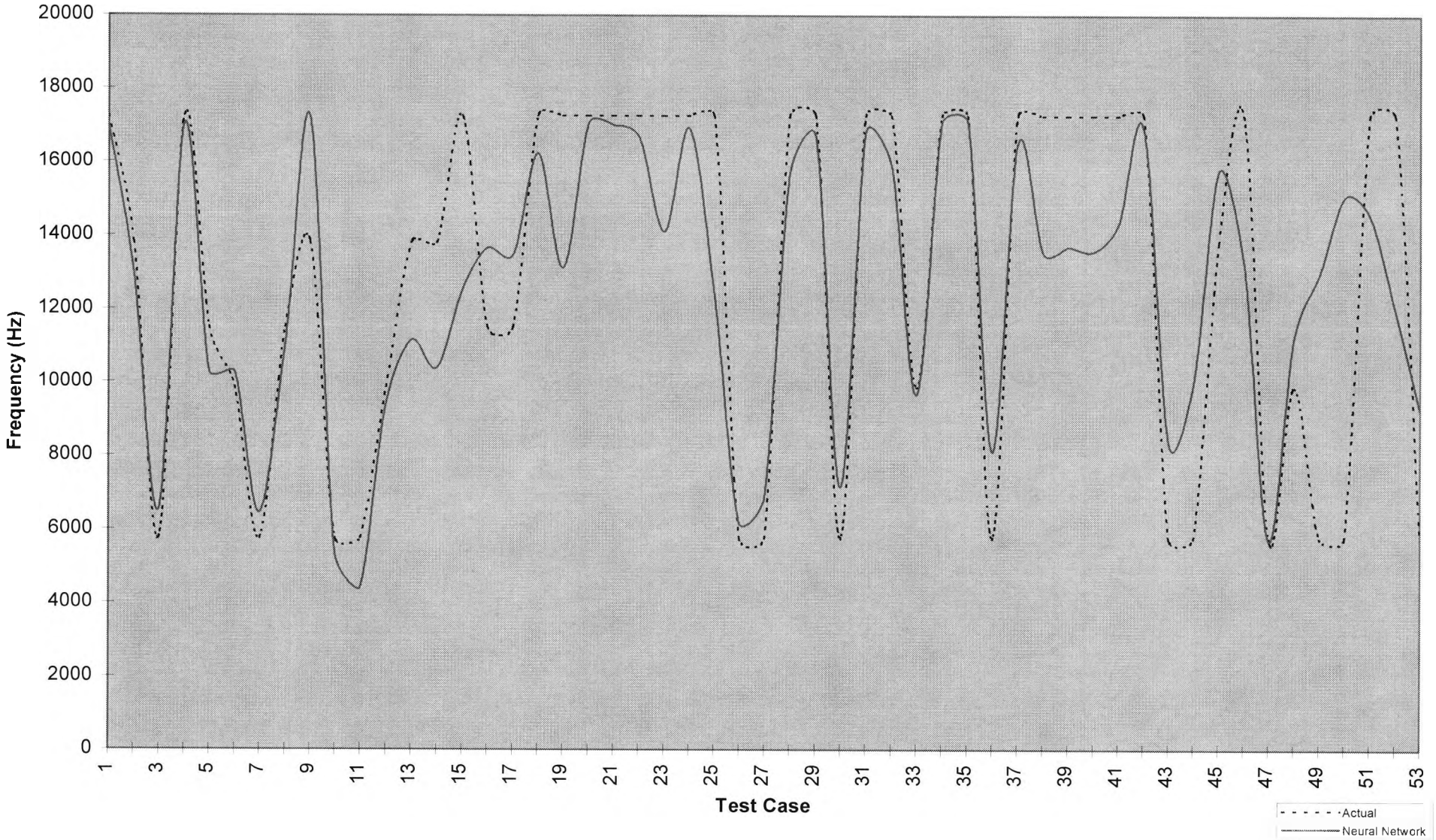


Figure D39: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.15, transfer function (output) = tanH, transfer function(hidden) = sigmoid, transfer function(input) = linear

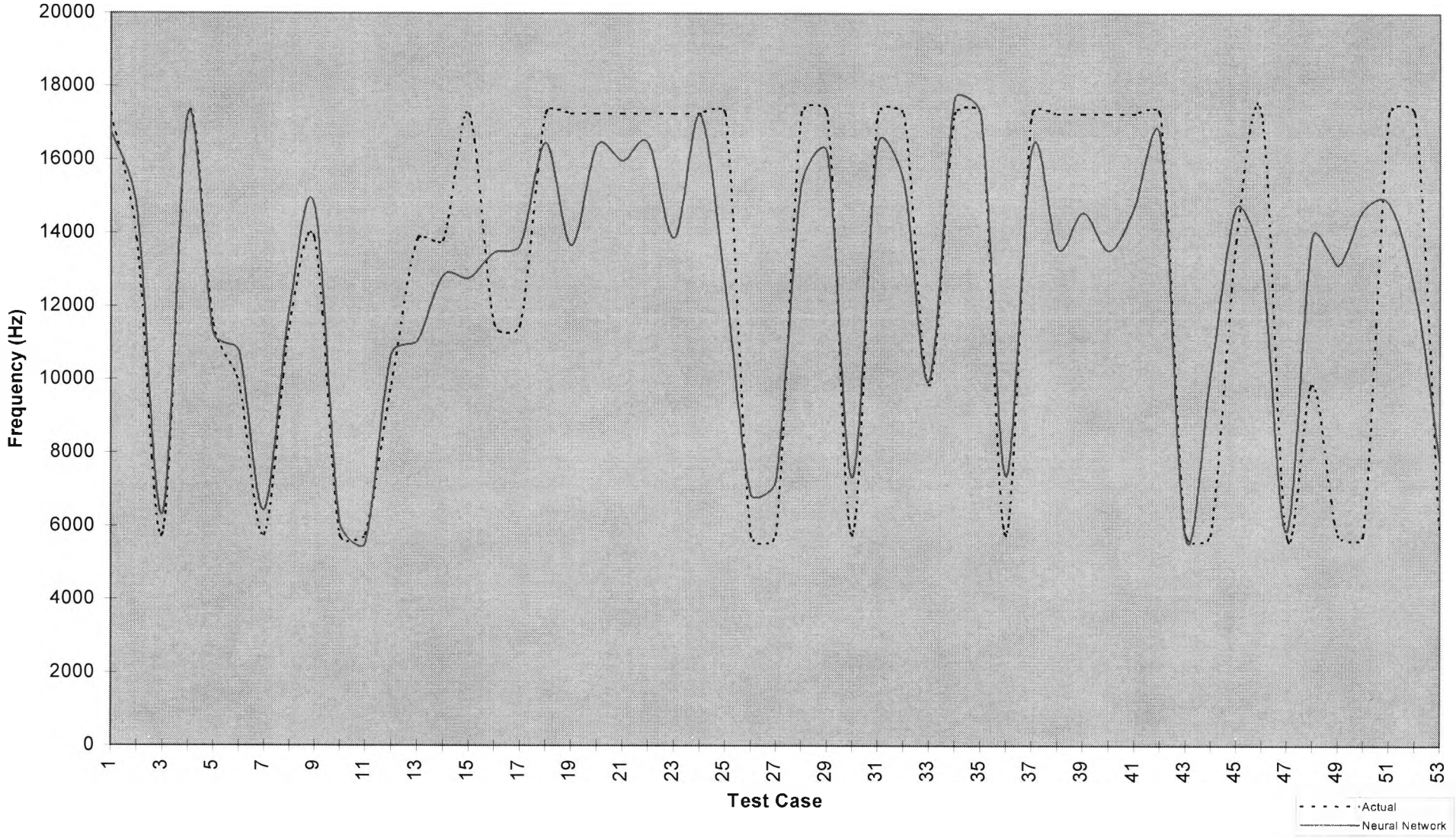


Figure D40: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.15, transfer function (output) = linear, transfer function(hidden) = sigmoid, transfer function(input) = linear

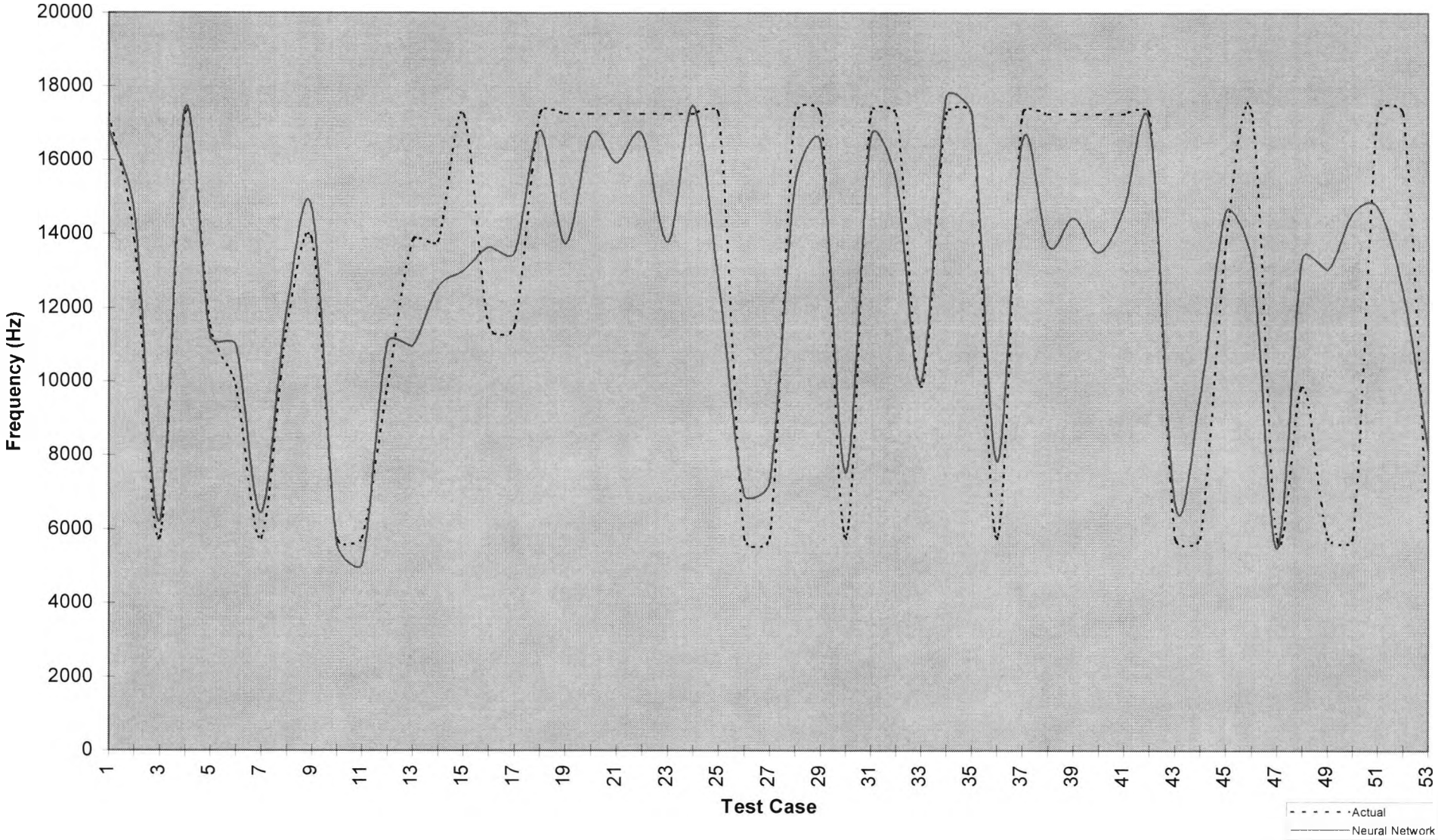


Figure D41: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.25, transfer function (output) = linear, transfer function(hidden) = sigmoid, transfer function(input) = linear

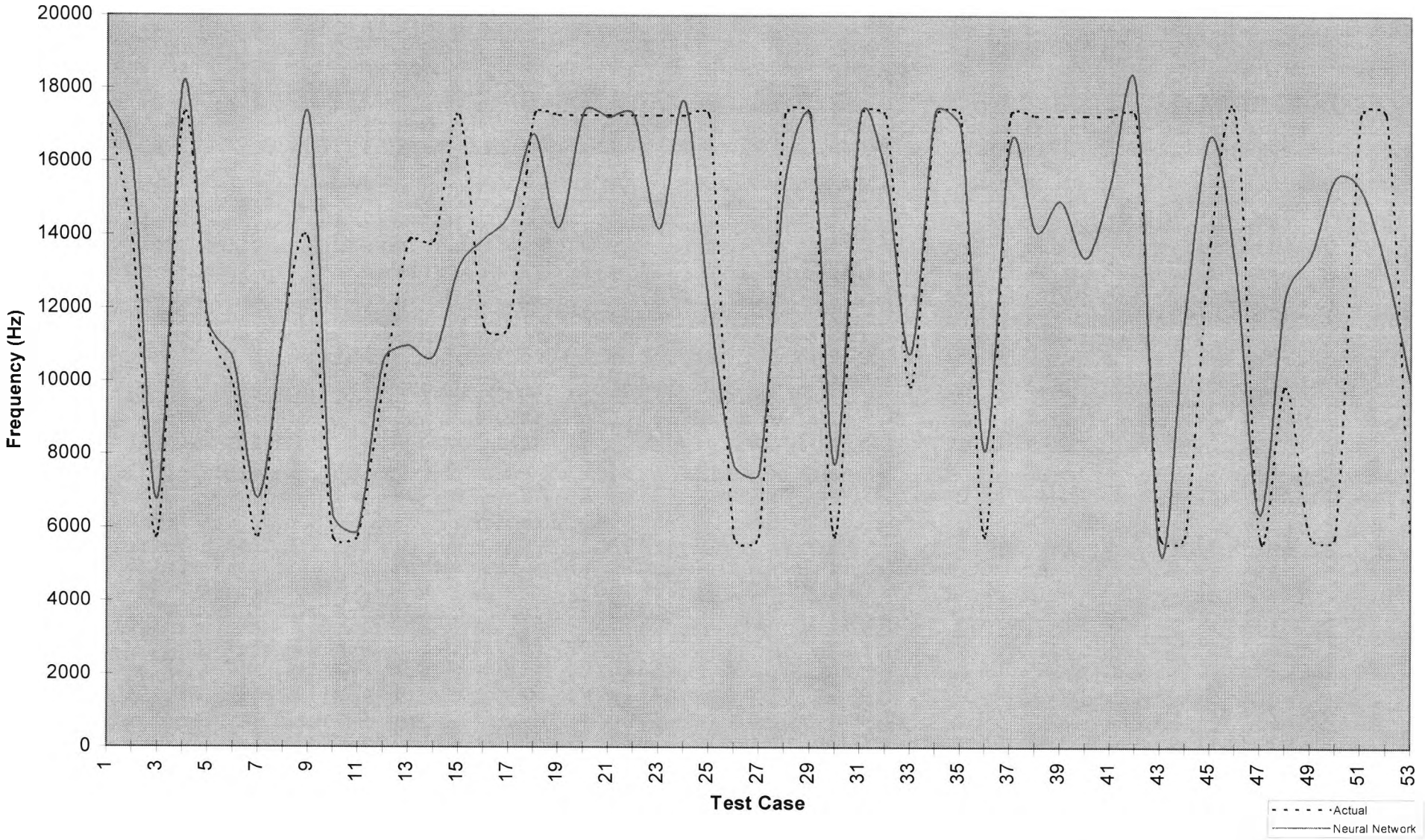


Figure D42: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.25, transfer function (output) = linear, transfer function(hidden) = sigmoid, transfer function(input) = linear, hidden nodes = 62

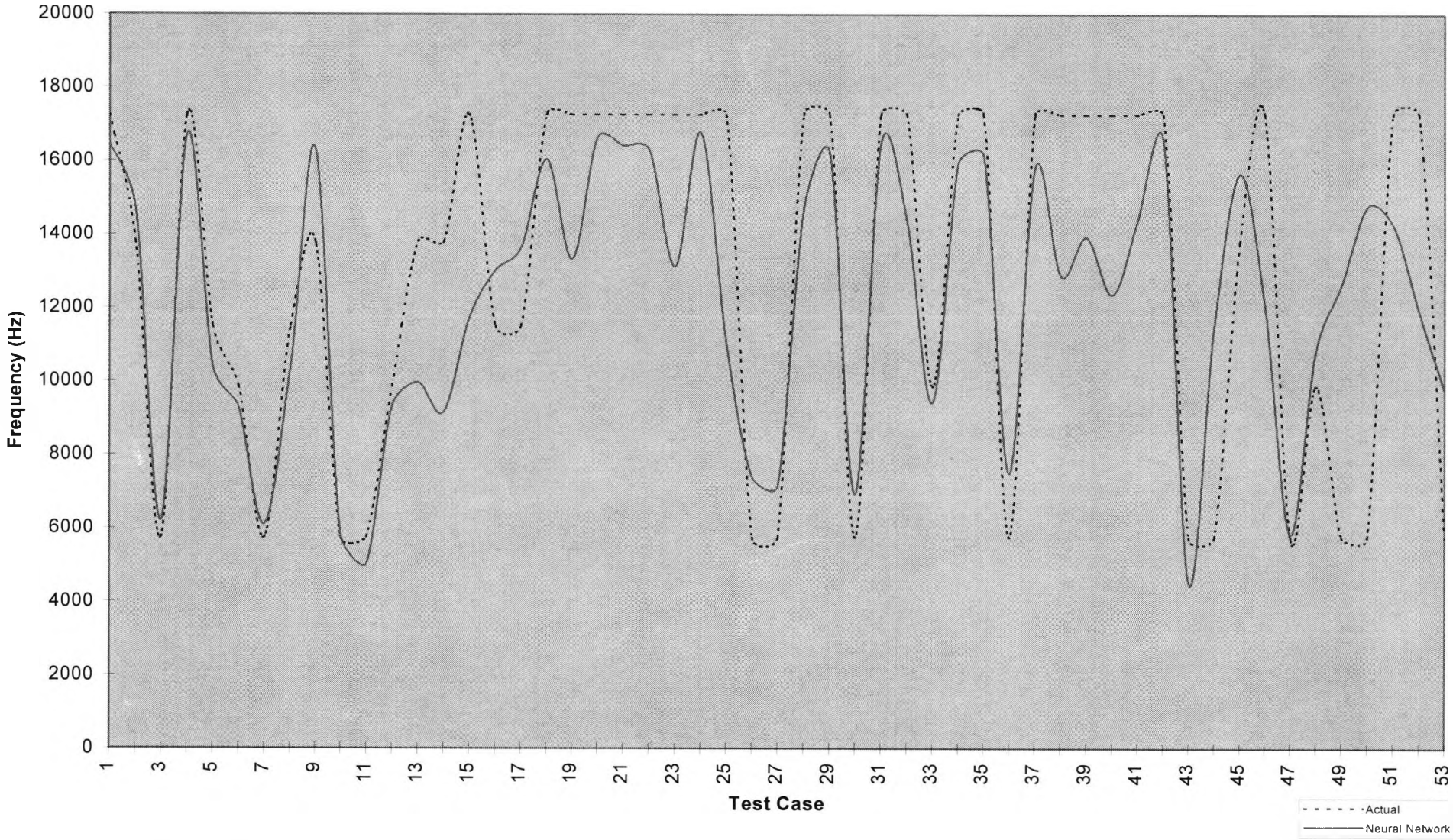


Figure D43: Momentum = 0.89, learning rate(hidden) = 0.4, learning rate(output) = 0.25, transfer function(output) = linear, transfer function(hidden) = sigmoid, transfer function(input) = linear, hidden nodes = 59

Appendix E

Finite element programs used in ANSYS

Input file for axisymmetric model (defect free solid)

```
ET,1,plane82,,,1
MP,EX,1,33100e6
MP,NUXY,1,0.2
MP,DENS,1,2300
N,1,0,0
N,61,0.9,0
FILL,1,61,59,2,0.015
NPLOT,1
N,62,0,0.015
N,122,0.9,0.015
FILL,62,122,59,63,0.015
N,123,0,0.03
N,183,0.9,0.03
FILL,123,183,59,124,0.015
N,184,0,0.045
N,244,0.9,0.045
FILL,184,244,59,185,0.015
N,245,0,0.06
N,305,0.9,0.06
FILL,245,305,59,246,0.015
N,306,0,0.075
N,366,0.9,0.075
FILL,306,366,59,307,0.015
N,367,0,0.09
N,427,0.9,0.09
FILL,367,427,59,368,0.015
N,428,0,0.105
N,488,0.9,0.105
FILL,428,488,59,429,0.015
N,489,0,0.12
N,549,0.9,0.12
FILL,489,549,59,490,0.015
N,550,0,0.135
N,610,0.9,0.135
FILL,550,610,59,551,0.015
N,611,0,0.15
N,671,0.9,0.15
FILL,611,671,59,612,0.015
N,672,0,0.165
N,732,0.9,0.165
FILL,672,732,59,673,0.015
N,733,0,0.18
N,793,0.9,0.18
FILL,733,793,59,734,0.015
N,794,0,0.195
N,854,0.9,0.195
FILL,794,854,59,795,0.015
N,855,0,0.21
N,915,0.9,0.21
FILL,855,915,59,856,0.015
N,916,0,0.225
N,976,0.9,0.225
FILL,916,976,59,917,0.015
N,977,0,0.24
N,1037,0.9,0.24
FILL,977,1037,59,978,0.015
N,1038,0,0.255
N,1098,0.9,0.255
FILL,1038,1098,59,1039,0.015
N,1099,0,0.27
N,1159,0.9,0.27
FILL,1099,1159,59,1100,0.015
N,1160,0,0.285
```

```
N,1220,0.9,0.285
FILL,1160,1220,59,1161,0.015
N,1221,0,0.3
N,1281,0.9,0.3
FILL,1221,1281,59,1222,0.015
EN,1,1,2,63,62
ENGEN,1,60,1,1,1,0
EN,61,62,63,124,123
ENGEN,1,60,1,61,61,1
EN,121,123,124,185,184
ENGEN,1,60,1,121,121,2
EN,181,184,185,246,245
ENGEN,1,60,1,181,181,3
EN,241,245,246,307,306
ENGEN,1,60,1,241,241,4
EN,301,306,307,368,367
ENGEN,1,60,1,301,301,5
EN,361,367,368,429,428
ENGEN,1,60,1,361,361,6
EN,421,428,429,490,489
ENGEN,1,60,1,421,421,7
EN,481,489,490,551,550
ENGEN,1,60,1,481,481,8
EN,541,550,551,612,611
ENGEN,1,60,1,541,541,9
EN,601,611,612,673,672
ENGEN,1,60,1,601,601,10
EN,661,672,673,734,733
ENGEN,1,60,1,661,661,11
EN,721,733,734,795,794
ENGEN,1,60,1,721,721,12
EN,781,794,795,856,855
ENGEN,1,60,1,781,781,13
EN,841,855,856,917,916
ENGEN,1,60,1,841,841,14
EN,901,916,917,978,977
ENGEN,1,60,1,901,901,15
EN,961,977,978,1039,1038
ENGEN,1,60,1,961,961,16
EN,1021,1038,1039,1100,1099
ENGEN,1,60,1,1021,1021,17
EN,1081,1099,1100,1161,1160
ENGEN,1,60,1,1081,1081,18
EN,1141,1160,1161,1222,1221
ENGEN,1,60,1,1141,1141,19
EMID,0
EPLOT
FINISH
/SOLU
ANTYPE,TRAN,NEW
TRNOPT,FULL
ITS=0.00000064
DELTIME,ITS
KBC,1
OUTRES,NSOL,ALL
D,P50X,UX,0
1
D,P50X,UX,0
62
D,P50X,UX,0
123
D,P50X,UX,0
184
D,P50X,UX,0
245
D,P50X,UX,0
306
D,P50X,UX,0
367
D,P50X,UX,0
428
D,P50X,UX,0
489
D,P50X,UX,0
550
D,P50X,UX,0
611
D,P50X,UX,0
672
D,P50X,UX,0
733
```

```
D,P50X,UX,0
794
D,P50X,UX,0
855
D,P50X,UX,0
916
D,P50X,UX,0
977
D,P50X,UX,0
1038
D,P50X,UX,0
1099
D,P50X,UX,0
1160
D,P50X,UX,0
1221
D,P50X,ALL,0
61
D,P50X,ALL,0
122
D,P50X,ALL,0
183
D,P50X,ALL,0
244
D,P50X,ALL,0
305
D,P50X,ALL,0
366
D,P50X,ALL,0
427
D,P50X,ALL,0
488
D,P50X,ALL,0
549
D,P50X,ALL,0
610
D,P50X,ALL,0
671
D,P50X,ALL,0
732
D,P50X,ALL,0
793
D,P50X,ALL,0
854
D,P50X,ALL,0
915
D,P50X,ALL,0
976
D,P50X,ALL,0
1037
D,P50X,ALL,0
1098
D,P50X,ALL,0
1159
D,P50X,ALL,0
1220
D,P50X,ALL,0
1281
TIME,1e-6
F,1221,FY,-1417
LSWRITE,1
TIME,2e-6
F,1221,FY,-2819
LSWRITE,2
TIME,3e-6
F,1221,FY,-4190
LSWRITE,3
TIME,4e-6
F,1221,FY,-5515
LSWRITE,4
TIME,5e-6
F,1221,FY,-6780
LSWRITE,5
TIME,6e-6
F,1221,FY,-7970
LSWRITE,6
TIME,7e-6
F,1221,FY,-9073
LSWRITE,7
TIME,8e-6
F,1221,FY,-10077
```

```
LSWRITE,8
TIME,9e-6
F,1221,FY,-10970
LSWRITE,9
TIME,10e-6
F,1221,FY,-11743
LSWRITE,10
TIME,11e-6
F,1221,FY,-12388
LSWRITE,11
TIME,12e-6
F,1221,FY,-12896
LSWRITE,12
TIME,13e-6
F,1221,FY,-13264
LSWRITE,13
TIME,14e-6
F,1221,FY,-13486
LSWRITE,14
TIME,15e-6
F,1221,FY,-13560
LSWRITE,15
TIME,16e-6
F,1221,FY,-13486
LSWRITE,16
TIME,17e-6
F,1221,FY,-13264
LSWRITE,17
TIME,18e-6
F,1221,FY,-12896
LSWRITE,18
TIME,19e-6
F,1221,FY,-12388
LSWRITE,19
TIME,20e-6
F,1221,FY,-11743
LSWRITE,20
TIME,21e-6
F,1221,FY,-10970
LSWRITE,21
TIME,22e-6
F,1221,FY,-10077
LSWRITE,22
TIME,23e-6
F,1221,FY,-9073
LSWRITE,23
TIME,24e-6
F,1221,FY,-7970
LSWRITE,24
TIME,25e-6
F,1221,FY,-6780
LSWRITE,25
TIME,26e-6
F,1221,FY,-5515
LSWRITE,26
TIME,27e-6
F,1221,FY,-4190
LSWRITE,27
TIME,28e-6
F,1221,FY,-2819
LSWRITE,28
TIME,29e-6
F,1221,FY,-1417
LSWRITE,29
TIME,30e-6
F,1221,FY,-0.00000001
LSWRITE,30
**This is repeated to loadstep 100**
TIME,1140e-6
F,1221,FY,-0.00000001
LSWRITE,100
```

Input file for defect model with contact element

```
/BATCH
! /COM,ANSYS RELEASE  5.4    UP19970828        15:02:22    07/06/1999
/input,start,ans    ,C:\ANSYS54\docu\,,,,,,,,,,,,,,,,,,1
/NOPR
/PMETH,OFF
KEYW,PR_SET,1
KEYW,PR_STRUC,1
```

Appendix E

```
KEYW,PR_THERM,0
KEYW,PR_FLUID,0
KEYW,PR_MULTI,0
/GO
/PREP7
ET,1,PLANE82
ET,2,CONTAC12
KEYOPT,1,3,2
KEYOPT,1,5,0
KEYOPT,1,6,0
KEYOPT,2,1,0
KEYOPT,2,2,0
KEYOPT,2,4,0
KEYOPT,2,7,0
R,2,180.8275000,-0.0001,3.827500,
UIMP,1,EX,,,33100000,
UIMP,1,DENS,,,2.3,
UIMP,1,ALPX,,,,
UIMP,1,REFT,,,,
UIMP,1,NUXY,,,,
UIMP,1,PRXY,,,0.2,
UIMP,1,GXY,,,,
UIMP,1,MU,,,,
UIMP,1,DAMP,,,,
UIMP,1,KXX,,,,
UIMP,1,C,,,,
UIMP,1,ENTH,,,,
UIMP,1,HF,,,,
UIMP,1,EMIS,,,,
UIMP,1,QRATE,,,,
UIMP,1,RSVX,,,,
UIMP,1,PERX,,,,
UIMP,1,VISC,,,,
UIMP,1,SONC,,,,
UIMP,2,EX,,,,
UIMP,2,DENS,,,,
UIMP,2,ALPX,,,,
UIMP,2,REFT,,,,
UIMP,2,NUXY,,,,
UIMP,2,PRXY,,,,
UIMP,2,GXY,,,,
UIMP,2,MU,,,0.1,
UIMP,2,DAMP,,,,
UIMP,2,KXX,,,,
UIMP,2,C,,,,
UIMP,2,ENTH,,,,
UIMP,2,HF,,,,
UIMP,2,EMIS,,,,
UIMP,2,QRATE,,,,
UIMP,2,RSVX,,,,
UIMP,2,PERX,,,,
UIMP,2,VISC,,,,
UIMP,2,SONC,,,,
K,1,,,,
K,2,1,,,
K,3,1,.3,,
K,4,0,0.3,,
K,4,0,0.3,,
K,5,.7,.15/2,,
K,6,1,.15/2,,
K,7,1,.15,,
K,8,.7,0.15,,
K,8,0.7,0.15,,
LSTR,    4,    1
LSTR,    1,    2
LSTR,    2,    3
LSTR,    3,    4
LSTR,    8,    5
LSTR,    5,    6
LSTR,    7,    8
FLST,2,4,4
FITEM,2,1
FITEM,2,2
FITEM,2,4
FITEM,2,3
AL,P51X
FLST,2,4,4
FITEM,2,5
FITEM,2,6
FITEM,2,7
FITEM,2,3
```

Appendix E

```
AL,P51X
LSTR,     6,     7
FLST,2,4,4
FITEM,2,5
FITEM,2,6
FITEM,2,7
FITEM,2,8
AL,P51X
ASBA,     1,     2
TYPE,1
MAT,1
REAL,1
ESYS,0
TSHAP,LINE
MSHAPE,0,2D
MSHKEY,0
CM,_Y,AREA
ASEL, , , ,      3
CM,_Y1,AREA
CHKMSH,'AREA'
CMSEL,S,_Y
AMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
ACLEAR,      3
ESIZE,.15/4,0,
CM,_Y,AREA
ASEL, , , ,      3
CM,_Y1,AREA
CHKMSH,'AREA'
CMSEL,S,_Y
AMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
/UI,MESH,OFF
N,766,.7000,.11250,,,,,
N,767,1.00,.1500,,,,,
N,768,.9625,0.15,,,,,
N,768,0.9625,0.15,,,,,
! NLIST,ALL, , , ,NODE,NODE,NODE
N,769,.92500,0.15,,,,,
! NLIST,ALL, , , ,NODE,NODE,NODE
N,770,.88750,0.15,,,,,
N,771,.8500,0.15,,,,,
N,772,.8125,0.15,,,,,
N,773,.7750,0.15,,,,,
! NLIST,ALL, , , ,NODE,NODE,NODE
N,774,.7375,0.15,,,,,
N,775,0.7375,0.1125,,,,,
N,776,0.7375+0.0375,0.1125,,,,,
N,777,0.7375+2*0.0375,0.1125,,,,,
N,778,0.7375+3*0.0375,0.1125,,,,,
N,779,0.7375+4*0.0375,0.1125,,,,,
N,780,0.7375+5*0.0375,0.1125,,,,,
N,781,0.7375+6*0.0375,0.1125,,,,,
N,782,0.7375+7*0.0375,0.1125,,,,,
N,782,0.7375+7*0.0375,0.1125,,,,,
NDELE,    766
TYPE,1,
MAT,1,
REAL,1,
ESYS,0,
FLST,2,4,1
FITEM,2,94
FITEM,2,775
FITEM,2,774
FITEM,2,92
E,P51X
FLST,2,4,1
FITEM,2,775
FITEM,2,776
FITEM,2,773
FITEM,2,774
E,P51X
FLST,2,4,1
FITEM,2,776
FITEM,2,777
FITEM,2,772
FITEM,2,773
```

```
E,P51X
FLST,2,4,1
FITEM,2,777
FITEM,2,778
FITEM,2,771
FITEM,2,772
E,P51X
FLST,2,4,1
FITEM,2,778
FITEM,2,779
FITEM,2,770
FITEM,2,771
E,P51X
FLST,2,4,1
FITEM,2,779
FITEM,2,780
FITEM,2,769
FITEM,2,770
E,P51X
FLST,2,4,1
FITEM,2,780
FITEM,2,781
FITEM,2,768
FITEM,2,769
E,P51X
FLST,2,4,1
FITEM,2,781
FITEM,2,782
FITEM,2,767
FITEM,2,768
E,P51X
FLST,2,4,1
FITEM,2,76
FITEM,2,78
FITEM,2,775
FITEM,2,94
E,P51X
FLST,2,4,1
FITEM,2,78
FITEM,2,80
FITEM,2,776
FITEM,2,775
E,P51X
FLST,2,4,1
FITEM,2,80
FITEM,2,82
FITEM,2,777
FITEM,2,776
E,P51X
FLST,2,4,1
FITEM,2,82
FITEM,2,84
FITEM,2,778
FITEM,2,777
E,P51X
FLST,2,4,1
FITEM,2,84
FITEM,2,86
FITEM,2,779
FITEM,2,778
E,P51X
FLST,2,4,1
FITEM,2,86
FITEM,2,88
FITEM,2,780
FITEM,2,779
E,P51X
FLST,2,4,1
FITEM,2,88
FITEM,2,90
FITEM,2,781
FITEM,2,780
E,P51X
FLST,2,4,1
FITEM,2,90
FITEM,2,72
FITEM,2,782
FITEM,2,781
E,P51X
TYPE,2,
MAT,2,
```

```
REAL,2,
ESYS,0,
EINTF,0.0001,
FINISH
/SOLU
FLST,2,55,1,ORDE,3
FITEM,2,2
FITEM,2,18
FITEM,2,-71
D,1,ALL
D,2,ALL
D,3,ALL
D,4,ALL
D,5,ALL
D,6,ALL
D,7,ALL
D,8,ALL
D,9,ALL
D,10,ALL
D,11,ALL
D,12,ALL
D,13,ALL
D,14,ALL
D,15,ALL
D,16,ALL
D,17,ALL
FINISH
/SOLU
ANTYPE,TRAN,NEW
TRNOPT,FULL
ITS=0.00000134
DELTIME,ITS
KBC,1
OUTRES,NSOL,ALL
TIME,1e-6
F,126,FY,-1417
LSWRITE,1
TIME,2e-6
F,126,FY,-2819
LSWRITE,2
TIME,3e-6
F,126,FY,-4190
LSWRITE,3
TIME,4e-6
F,126,FY,-5515
LSWRITE,4
TIME,5e-6
F,126,FY,-6780
LSWRITE,5
TIME,6e-6
F,126,FY,-7970
LSWRITE,6
TIME,7e-6
F,126,FY,-9073
LSWRITE,7
TIME,8e-6
F,126,FY,-10077
LSWRITE,8
TIME,9e-6
F,126,FY,-10970
LSWRITE,9
TIME,10e-6
F,126,FY,-11743
LSWRITE,10
TIME,11e-6
F,126,FY,-12388
LSWRITE,11
TIME,12e-6
F,126,FY,-12894
LSWRITE,12
TIME,13e-6
F,126,FY,-13244
LSWRITE,13
TIME,14e-6
F,126,FY,-13484
LSWRITE,14
TIME,15e-6
F,126,FY,-13540
LSWRITE,15
TIME,16e-6
F,126,FY,-13484
```

```
LSWRITE,16
TIME,17e-6
F,126,FY,-13244
LSWRITE,17
TIME,18e-6
F,126,FY,-12894
LSWRITE,18
TIME,19e-6
F,126,FY,-12388
LSWRITE,19
TIME,20e-6
F,126,FY,-11743
LSWRITE,20
TIME,21e-6
F,126,FY,-10970
LSWRITE,21
TIME,22e-6
F,126,FY,-10077
LSWRITE,22
TIME,23e-6
F,126,FY,-9073
LSWRITE,23
TIME,24e-6
F,126,FY,-7970
LSWRITE,24
TIME,25e-6
F,126,FY,-6780
LSWRITE,25
TIME,26e-6
F,126,FY,-5515
LSWRITE,26
TIME,27e-6
F,126,FY,-4190
LSWRITE,27
TIME,28e-6
F,126,FY,-2819
LSWRITE,28
TIME,29e-6
F,126,FY,-1417
LSWRITE,29
TIME,30e-6
F,126,FY,0
LSWRITE,30
**This is repeated to loadstep 110**
TIME,10240e-6
F,126,FY,0
LSWRITE,110
```

Input file for 2D defect model without contact element

```
/BATCH
! /COM,ANSYS RELEASE  5.4    UP19970828       15:02:22    07/06/1999
/input,start,ans    ,C:\ANSYS54\docu\,,,,,,,,,,,,,,,,1
/NOPR
/PMETH,OFF
KEYW,PR_SET,1
KEYW,PR_STRUC,1
KEYW,PR_THERM,0
KEYW,PR_FLUID,0
KEYW,PR_MULTI,0
/GO
/PREP7
ET,1,PLANE82
KEYOPT,1,3,2
KEYOPT,1,5,0
KEYOPT,1,6,0
KEYOPT,2,1,0
KEYOPT,2,2,0
KEYOPT,2,4,0
KEYOPT,2,7,0
UIMP,1,EX, , ,33100000,
UIMP,1,DENS, , ,2.3,
UIMP,1,ALPX, , , ,
UIMP,1,REFT, , , ,
UIMP,1,NUXY, , , ,
UIMP,1,PRXY, , ,0.2,
UIMP,1,GXY, , , ,
UIMP,1,MU, , , ,
UIMP,1,DAMP, , , ,
UIMP,1,KXX, , , ,
UIMP,1,C, , , ,
```

```
UIMP,1,ENTH, , , ,
UIMP,1,HF, , , ,
UIMP,1,EMIS, , , ,
UIMP,1,QRATE, , , ,
UIMP,1,RSVX, , , ,
UIMP,1,PERX, , , ,
UIMP,1,VISC, , , ,
UIMP,1,SONC, , , ,
K,1,,,,
K,2,1,,,
K,3,1,.3,,
K,4,0.0.3,,
K,4.0,0.3,,
K,5,.7,.15/2,,
K,6,1,.15/2,,
K.7,1,.15,,
K,8,.7,0.15,,
K,8,0.7,0.15,,
LSTR,    4,    1
LSTR,    1,    2
LSTR,    2,    3
LSTR,    3,    4
LSTR,    8,    5
LSTR,    5,    6
LSTR,    7,    8
FLST,2,4,4
FITEM,2,1
FITEM,2,2
FITEM,2,4
FITEM,2,3
AL,P51X
FLST,2,4,4
FITEM,2,5
FITEM,2,6
FITEM,2,7
FITEM,2,3
AL,P51X
LSTR,    6,    7
FLST,2,4,4
FITEM,2,5
FITEM,2,6
FITEM,2,7
FITEM,2,8
AL,P51X
ASBA,    1,    2
TYPE,1
MAT,1
REAL,1
ESYS,0
TSHAP,LINE
MSHAPE,0,2D
MSHKEY,0
CM,_Y,AREA
ASEL, , , ,    3
CM,_Y1,AREA
CHKMSH,'AREA'
CMSEL,S,_Y
AMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
ACLEAR,    3
ESIZE,.15/4,0,
CM,_Y,AREA
ASEL, , , ,    3
CM,_Y1,AREA
CHKMSH,'AREA'
CMSEL,S,_Y
AMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
/UI,MESH,OFF
N,766,.7000,.11250,,,,,
N,767,1.00,.1500,,,,,
N,768,.9625,0.15,,,,,
N,768,0.9625,0.15,,,,,
N,769,.92500,0.15,,,,,
N,770,.88750,0.15,,,,,
N,771,.8500,0.15,,,,,
N,772,.8125,0.15,,,,,
```

```
N,773,.7750,0.15,,,,,
N,774,.7375,0.15,,,,,
N,775,0.7375,0.1125,,,,,
N,776,0.7375+0.0375,0.1125,,,,,
N,777,0.7375+2*0.0375,0.1125,,,,,
N,778,0.7375+3*0.0375,0.1125,,,,,
N,779,0.7375+4*0.0375,0.1125,,,,,
N,780,0.7375+5*0.0375,0.1125,,,,,
N,781,0.7375+6*0.0375,0.1125,,,,,
N,782,0.7375+7*0.0375,0.1125,,,,,
N,782,0.7375+7*0.0375,0.1125,,,,,
 NDELE,    766
TYPE,1,
MAT,1,
REAL,1,
ESYS,0,
FLST,2,4,1
FITEM,2,94
FITEM,2,775
FITEM,2,774
FITEM,2,92
E,P51X
FLST,2,4,1
FITEM,2,775
FITEM,2,776
FITEM,2,773
FITEM,2,774
E,P51X
FLST,2,4,1
FITEM,2,776
FITEM,2,777
FITEM,2,772
FITEM,2,773
E,P51X
FLST,2,4,1
FITEM,2,777
FITEM,2,778
FITEM,2,771
FITEM,2,772
E,P51X
FLST,2,4,1
FITEM,2,778
FITEM,2,779
FITEM,2,770
FITEM,2,771
E,P51X
FLST,2,4,1
FITEM,2,779
FITEM,2,780
FITEM,2,769
FITEM,2,770
E,P51X
FLST,2,4,1
FITEM,2,780
FITEM,2,781
FITEM,2,768
FITEM,2,769
E,P51X
FLST,2,4,1
FITEM,2,781
FITEM,2,782
FITEM,2,767
FITEM,2,768
E,P51X
FLST,2,4,1
FITEM,2,76
FITEM,2,78
FITEM,2,775
FITEM,2,94
E,P51X
FLST,2,4,1
FITEM,2,78
FITEM,2,80
FITEM,2,776
FITEM,2,775
E,P51X
FLST,2,4,1
FITEM,2,80
FITEM,2,82
FITEM,2,777
FITEM,2,776
```

```
E,P51X
FLST,2,4,1
FITEM,2,82
FITEM,2,84
FITEM,2,778
FITEM,2,777
E,P51X
FLST,2,4,1
FITEM,2,84
FITEM,2,86
FITEM,2,779
FITEM,2,778
E,P51X
FLST,2,4,1
FITEM,2,86
FITEM,2,88
FITEM,2,780
FITEM,2,779
E,P51X
FLST,2,4,1
FITEM,2,88
FITEM,2,90
FITEM,2,781
FITEM,2,780
E,P51X
FLST,2,4,1
FITEM,2,90
FITEM,2,72
FITEM,2,782
FITEM,2,781
E,P51X
FINISH
```

The remaining part of the input file is identical to the input file for 2D defect model using contact element

Input file for 3D model of Wall A

```
/BATCH
! /COM,ANSYS RELEASE  5.3    UP071096        13:06:42    05/26/1997
/input,start,ans    ,C:\ANSYS53\docu\,,,,,,,,,,,,,,,,1
/FILNAM,mai
KEYW,PR_SET,1
KEYW,PR_STRUC,1
KEYW,PR_THERM,0
KEYW,PR_ELMAG,0
KEYW,PR_FLUID,0
KEYW,PR_MULTI,0
KEYW,PR_CFD,0
KEYW,LSDYNA,0
/PMETH,OFF
/TITLE,wall with a voids
/PREP7
ET,1,SOLID95
KEYOPT,1,5,0
KEYOPT,1,6,0
UIMP,1,EX, , ,3.31e7,
UIMP,1,DENS, , ,2.3,
UIMP,1,ALPX, , , ,
UIMP,1,REFT, , , ,
UIMP,1,NUXY, , , ,
UIMP,1,PRXY, , ,.2,
UIMP,1,GXY, , ,12961538,
UIMP,1,MU, , , ,
UIMP,1,DAMP, , , ,
UIMP,1,KXX, , , ,
UIMP,1,C, , , ,
UIMP,1,ENTH, , , ,
UIMP,1,HF, , , ,
UIMP,1,EMIS, , , ,
UIMP,1,QRATE, , , ,
UIMP,1,MURX, , , ,
UIMP,1,MGXX, , , ,
UIMP,1,RSVX, , , ,
UIMP,1,PERX, , , ,
UIMP,1,VISC, , , ,
UIMP,1,SONC, , , ,
FLST,2,8,4,ORDE,2
FITEM,2,73
FITEM,2,-80
LDELE,P51X, , ,1
FLST,2,4,4,ORDE,3
FITEM,2,89
```

```
FITEM,2,92
FITEM,2,-94
LDELE,P51X, , ,1
LDELE,     91, , ,1
FLST,2,10,4,ORDE,7
FITEM,2,81
FITEM,2,-85
FITEM,2,87
FITEM,2,-88
FITEM,2,90
FITEM,2,95
FITEM,2,-96
LDELE,P51X, , ,1
LDELE,     86, , ,1
/PREP7
VDELE,     3, , ,
FLST,2,4,3,ORDE,4
FITEM,2,18
FITEM,2,27
FITEM,2,36
FITEM,2,55
KDELE,P51X
/PREP7
BLOCK,0,1.219,0,1.219,0,.3048,
K,1,,,,
K,15,,,,
K,15,,,,
/DEVICE,VECTOR,0
/DEVICE,DITHER,1
VDELE,     1, , ,1
! KPLOT
KDELE,     15
CSYS,0
WPAVE,0,0,0
CSYS,0
CSYS,0
WPAVE,0,0,0
CSYS,0
BLOCK,0,1.219,0,1.219,0,0.3048,
K,9,.254,.254,.1016,
KWPAVE,     9
BLC4,0,0,.1524,.1524,.1016
VSBV,     1,     2
K,18,.9652,0.254,0.1016,
KWPAVE,     18
BLC4,0,0,.1524,.1524,.0254
VDELE,     1, , ,1
KDELE,     18
CSYS,0
WPAVE,0,0,0
CSYS,0
K,18,0.8128,0.254,0.1016,
KWPAVE,     18
BLC4,0,0,.1524,.1524,.0254
VSBV,     3,     1
K,27,.254,.8128,.127,
VDELE,     2
FLST,2,18,5,ORDE,2
FITEM,2,1
FITEM,2,-18
ADELE,P51X, , ,1
CSYS,0
WPAVE,0,0,0
CSYS,0
FLST,2,2,3,ORDE,2
FITEM,2,18
FITEM,2,27
KDELE,P51X
BLC4,0,0,1.219,1.219,.3048
KWPAVE,     9
BLC4,0,0,.1524,.1524,.1016
VSBV,     1,     2
K,18,.8128,.254,.1778,
KWPAVE,     18
BLC4,0,0,.1524,.1524,.0254
VSBV,     3,     1
K,27,.254,.8128,.127,
KWPAVE,     27
BLC4,0,0,.1524,.1524,.0762
VSBV,     2,     1
K,36,.8128,0.8128,.1524,
```

```
KWPAVE,     36
BLC4,0,0,.1524,.1524,.0508
VSBV,     3,     1
FINISH
/PREP7
K,45,.5334,.4572,.127,
KWPAVE,     45
BLC4,0,0,.1524,.3048,.0762
VSBV,     2,     1
TYPE,1,
MAT,1,
REAL,1,
ESYS,0,
ESHAPE,1,0
SMRTSIZE,6
MOPT,TETEXPND,0.8,
FLST,2,10,4,ORDE,4
FITEM,2,5
FITEM,2,-6
FITEM,2,25
FITEM,2,-32
LESIZE,P51X,.05, , ,1,
CM,_Y,VOLU
VSEL, , , ,     3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
FLST,2,2662,2,ORDE,2
FITEM,2,1
FITEM,2,-2662
EDELE,P51X
VCLEAR,     3
ESIZE,0,4,
FLST,2,10,4,ORDE,4
FITEM,2,5
FITEM,2,-6
FITEM,2,25
FITEM,2,-32
LESIZE,P51X, , , ,1,
FLST,2,72,4,ORDE,2
FITEM,2,1
FITEM,2,-72
LCLEAR,P51X
FLST,2,72,4,ORDE,2
FITEM,2,1
FITEM,2,-72
LCLEAR,P51X
SMRTSIZE,6
ESIZE,0,4,
ESHAPE,1,0
SMRTSIZE,6
ESIZE,0,0,
CM,_Y,VOLU
VSEL, , , ,     3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
VCLEAR,     3
FLST,2,2,4,ORDE,2
FITEM,2,5
FITEM,2,-6
LCLEAR,P51X
FLST,2,14,4,ORDE,4
FITEM,2,4
FITEM,2,6
FITEM,2,25
FITEM,2,-36
LCLEAR,P51X
ESHAPE,1,0
SMRTSIZE,6
SMRTSIZE, ,1,1,2,15,30,1.5,1,1,4
ESIZE,0,0,
FLST,2,2,4,ORDE,2
```

Appendix E

```
FITEM,2,5
FITEM,2,-6
LESIZE,P51X,2, ,0.0,
FLST,2,2,4,ORDE,2
FITEM,2,5
FITEM,2,-6
LESIZE,P51X,.1, ,0.0,
FLST,2,2,4,ORDE,2
FITEM,2,5
FITEM,2,-6
LESIZE,P51X,0.13, ,0.0,
FLST,2,8,4,ORDE,2
FITEM,2,25
FITEM,2,-32
LESIZE,P51X,0.5, ,0.0,
FLST,2,8,4,ORDE,2
FITEM,2,25
FITEM,2,-32
LESIZE,P51X,0.05, ,0.0,
FLST,2,8,4,ORDE,2
FITEM,2,25
FITEM,2,-32
LESIZE,P51X,0.06, ,0.0,
FLST,2,2,4,ORDE,2
FITEM,2,3
FITEM,2,-4
LESIZE,P51X,0.13, ,0.0,
CM,_Y,VOLU
VSEL, , , ,      3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
VCLEAR,      3
SMRTSIZE,7
CM,_Y,VOLU
VSEL, , , ,      3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
 CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
VCLEAR,      3
K,54,0.6095,0,0.3048,
K,55,0.6095,0,0,
K,56,1.219,1.219,0,
KDELE,      56
K,56,1.219,.6095,0,
K,57,1.219,0.6095,.3048,
LSTR,      54,      55
LSTR,      57,      56
LSBL,      5,      73
LSBL,      4,      4
! LPLOT
LSBL,      6,      74
LSTR,      57,      21
LDELE,      6, , ,1
K,58,1.219,0.6095,0,
K,59,1.219/2,0,0,
LSTR,      54,      59
LSTR,      57,      58
LSBL,      4,      6
LSBL,      3,      74
SMRTSIZE,8
LDELE,      5, , ,1
SMRTSIZE,8
FLST,2,5,4,ORDE,5
FITEM,2,5
FITEM,2,-6
FITEM,2,10
FITEM,2,76
FITEM,2,78
LESIZE,P51X,0.13, ,0.0,
CM,_Y,VOLU
VSEL, , , ,      3
CM,_Y1,VOLU
```

```
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
VCLEAR,      3
SMRTSIZE,8
ESIZE,0,0,
FLST,2,5,4,ORDE,5
FITEM,2,5
FITEM,2,-6
FITEM,2,10
FITEM,2,76
FITEM,2,78
LESIZE,P51X,0.10, ,0,0,
CM,_Y,VOLU
VSEL, , , ,      3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
NSEL,S,LOC,Z,.3048
VSEL,ALL
ASEL,ALL
LSEL,ALL
KSEL,ALL
ESEL,ALL
NSEL,ALL
FLST,2,8,4,ORDE,2
FITEM,2,49
FITEM,2,-56
LESIZE,P51X,0.07, ,0,0,
FLST,2,5,4,ORDE,5
FITEM,2,5
FITEM,2,-6
FITEM,2,10
FITEM,2,76
FITEM,2,78
LESIZE,P51X,0.06, ,0,0,
CM,_Y,VOLU
VSEL, , , ,      3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
NSEL,S,LOC,Z,.3048
VSEL,ALL
ASEL,ALL
LSEL,ALL
KSEL,ALL
ESEL,ALL
NSEL,ALL
NSEL,S,LOC,X,0
FINISH
/SOLU
FLST,2,4,1,ORDE,4
FITEM,2,1
FITEM,2,-2
FITEM,2,9
FITEM,2,-10
D,P51X, ,0, , , ,ALL
FLST,2,39,1,ORDE,12
FITEM,2,1
FITEM,2,-2
FITEM,2,9
FITEM,2,-10
FITEM,2,53
FITEM,2,-63
FITEM,2,95
FITEM,2,-105
FITEM,2,635
FITEM,2,732
FITEM,2,744
FITEM,2,-754
```

```
D,P51X, ,0, , , ,UX,UZ
VSEL,ALL
ASEL,ALL
LSEL,ALL
KSEL,ALL
ESEL,ALL
NSEL,ALL
FINISH
/PREP7
VCLEAR,      3
SMRTSIZE,8
ESIZE,0,0,
FLST,2,5,4,ORDE,5
FITEM,2,5
FITEM,2,-6
FITEM,2,10
FITEM,2,76
FITEM,2,78
LESIZE,P51X,.06, , ,1,
FLST,2,5,4,ORDE,5
FITEM,2,5
FITEM,2,-6
FITEM,2,10
FITEM,2,76
FITEM,2,78
LESIZE,P51X,0.1, , ,1,
FLST,2,8,4,ORDE,2
FITEM,2,25
FITEM,2,-32
LESIZE,P51X,0.08, , ,1,
CM,_Y,VOLU
VSEL, , , ,      3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
VCLEAR,      3
FLST,2,8,4,ORDE,2
FITEM,2,49
FITEM,2,-56
LESIZE,P51X,0.15, , ,1,
FLST,2,16,4,ORDE,4
FITEM,2,13
FITEM,2,-20
FITEM,2,37
FITEM,2,-44
LESIZE,P51X,0.15, , ,1,
CM,_Y,VOLU
VSEL, , , ,      3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
NSEL,S,LOC,X,0
FINISH
/SOLU
FLST,2,4,1,ORDE,4
FITEM,2,1
FITEM,2,-2
FITEM,2,9
FITEM,2,-10
D,P51X, ,0, , , ,ALL
FLST,2,39,1,ORDE,12
FITEM,2,1
FITEM,2,-2
FITEM,2,9
FITEM,2,-10
FITEM,2,53
FITEM,2,-63
FITEM,2,93
FITEM,2,-103
FITEM,2,423
FITEM,2,468
FITEM,2,480
FITEM,2,-490
```

```
D,P51X, ,0, , , ,UX,UZ
FINISH
/SOLU
OUTRES,NSOL,LAST,
/PREP7
VCLEAR,      3
TYPE,1,
MAT,1,
REAL,1,
ESYS,0,
SMRTSIZE,8
ESIZE,0,0,
FLST,2,1,4,ORDE,1
FITEM,2,11
LESIZE,P51X, , ,4,1,
FLST,2,1,4,ORDE,1
FITEM,2,11
LESIZE,P51X, , ,2,1,
FLST,2,8,4,ORDE,2
FITEM,2,13
FITEM,2,-20
LESIZE,P51X, , ,1,1,
FLST,2,8,4,ORDE,2
FITEM,2,37
FITEM,2,-44
LESIZE,P51X, , ,1,1,
FLST,2,8,4,ORDE,2
FITEM,2,61
FITEM,2,-68
LESIZE,P51X,2, ,1,1,
FLST,2,8,4,ORDE,2
FITEM,2,49
FITEM,2,-56
LESIZE,P51X,1, ,1,1,
CM,_Y,VOLU
VSEL, , , ,      3
CM,_Y1,VOLU
CHKMSH,'VOLU'
CMSEL,S,_Y
VMESH,_Y1
CMDEL,_Y
CMDEL,_Y1
CMDEL,_Y2
NSEL,S,LOC,Z,.3048
VSEL,ALL
ASEL,ALL
LSEL,ALL
KSEL,ALL
ESEL,ALL
NSEL,ALL
FINISH
/SOLU
FLST,2,4,1,ORDE,4
FITEM,2,1
FITEM,2,-2
FITEM,2,9
FITEM,2,-10
D,P51X, ,0, , , ,ALL
FLST,2,33,1,ORDE,12
FITEM,2,1
FITEM,2,-2
FITEM,2,9
FITEM,2,-10
FITEM,2,53
FITEM,2,-61
FITEM,2,87
FITEM,2,-95
FITEM,2,363
FITEM,2,410
FITEM,2,421
FITEM,2,-429
D,P51X, ,0, , , ,UX,UZ
ANTYPE,TRAN,NEW
TRNOPT,FULL
ITS=0.00000134
DELTIME,ITS
KBC,1
OUTRES,ALL,NONE
OUTRES,NSOL,ALL
TIME,1e-6
F,276,FZ,-1253.3
```

```
LSWRITE,1
TIME,2e-6
F,276,FZ,-2486.9
LSWRITE,2
TIME,3e-6
F,276,FZ,-3681.2
LSWRITE,3
TIME,4e-6
 F,276,FZ,-4817.5
LSWRITE,4
TIME,5e-6
F,276,FZ,-5877.9
LSWRITE,5
TIME,6e-6
 F,276,FZ,-6845.5
LSWRITE,6
TIME,7e-6
 F,276,FZ,-7705.1
LSWRITE,7
TIME,8e-6
F,276,FZ,-8443.3
LSWRITE,8
TIME,9e-6
F,276,FZ,-9048.3
LSWRITE,9
TIME,10e-6
F,276,FZ,-9510.6
LSWRITE,10
TIME,11e-6
F,276,FZ,-9822.9
LSWRITE,11
TIME,12e-6
F,276,FZ,-9980.3
LSWRITE,12
TIME,13e-6
F,276,FZ,-9980.3
LSWRITE,13
TIME,14e-6
F,276,FZ,-9822.9
LSWRITE,14
TIME,15e-6
F,276,FZ,-9510.6
LSWRITE,15
TIME,16e-6
F,276,FZ,-9048.3
LSWRITE,16
TIME,17e-6
F,276,FZ,-8443.3
LSWRITE,17
TIME,18e-6
F,276,FZ,-7705.1
LSWRITE,18
TIME,19e-6
F,276,FZ,-6845.5
LSWRITE,19
TIME,20e-6
F,276,FZ,-5877.8
LSWRITE,20
TIME,21e-6
F,276,FZ,-4817.5
LSWRITE,21
TIME,22e-6
F,276,FZ,-3681.2
LSWRITE,22
TIME,23e-6
 F,276,FZ,-2486.9
LSWRITE,23
TIME,24e-6
 F,276,FZ,-1253.3
LSWRITE,24
TIME,25e-6
F,276,FZ,0
LSWRITE,25
**This is repeated to loadstep 110**
TIME,10240e-6
F,276,FZ,0
LSWRITE,110
```

Appendix F

Graphs and programs using Matlab 4.0

Selection of impact echo data with waveform and FFT processed graphs

As discussed in chapter 3, there were a total of 225 impact echo data, however, only a selection of the best graphs for each test point are illustrated here. Also, for each location, the raw and FFT processed are shown. The code names referring to these locations given in chapter 3 are indicated for each graphs. The essential statistics for the impact echo testing are as follows:

- Sampling period (time interval between points) = 10 μs
- Sampling points for FFT spectrum = 1024
- Total duration (time span) = 10240 μs
- Impact duration (contact time of the impact on the surface) = 25 μs

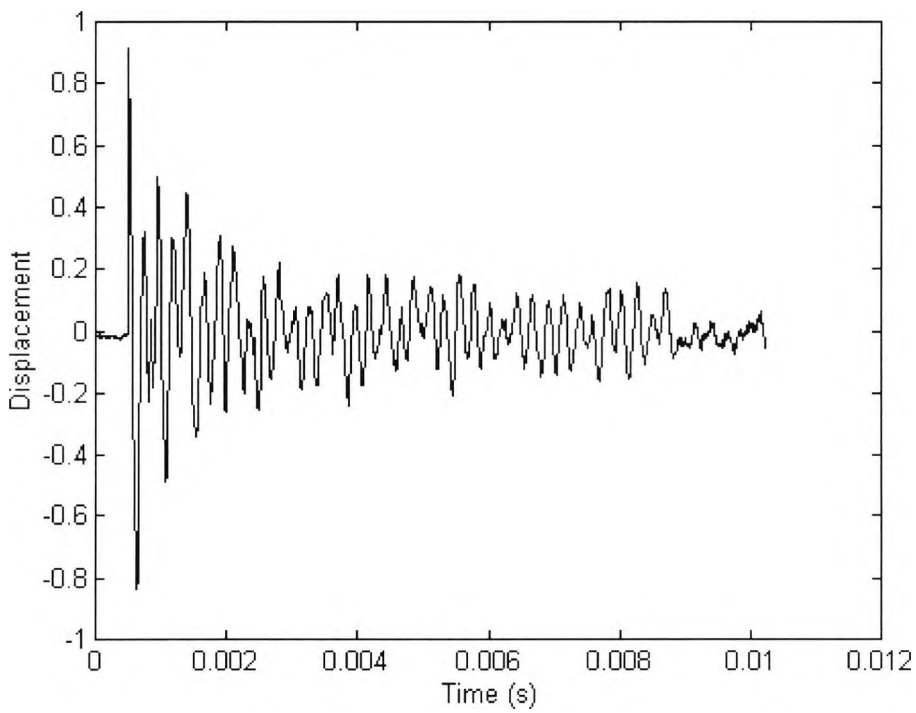


Figure F1: Waveform of Wav1e

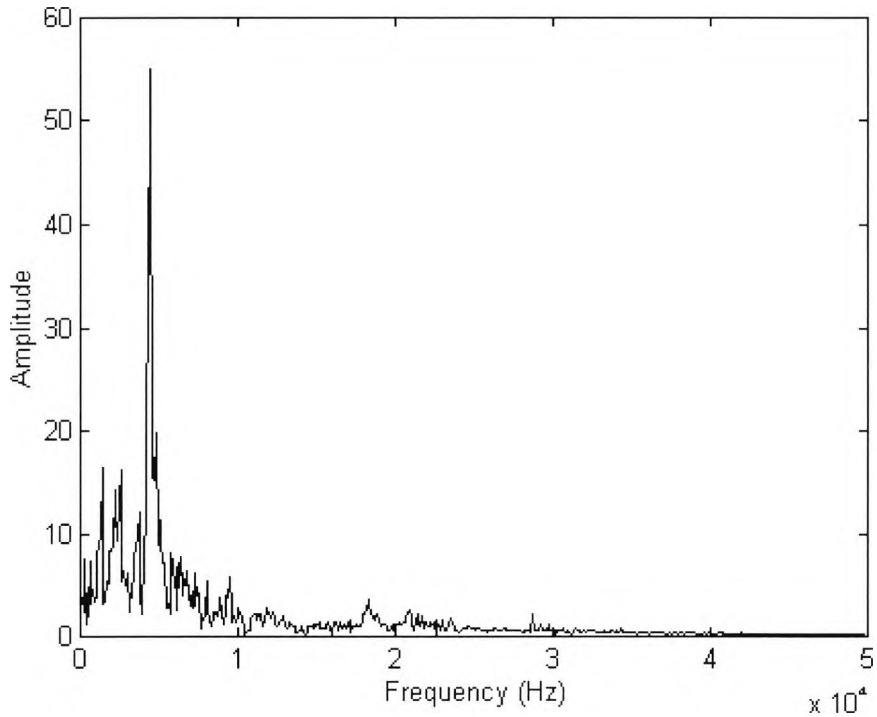


Figure F2: Amplitude spectrum of Wav1e

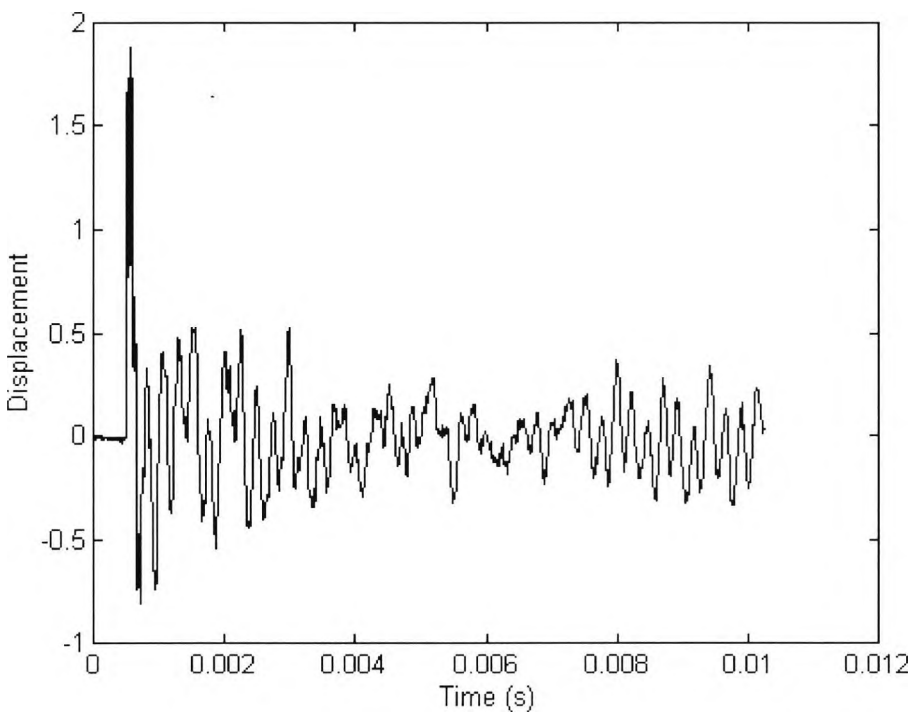


Figure F3: Waveform of Wav2e

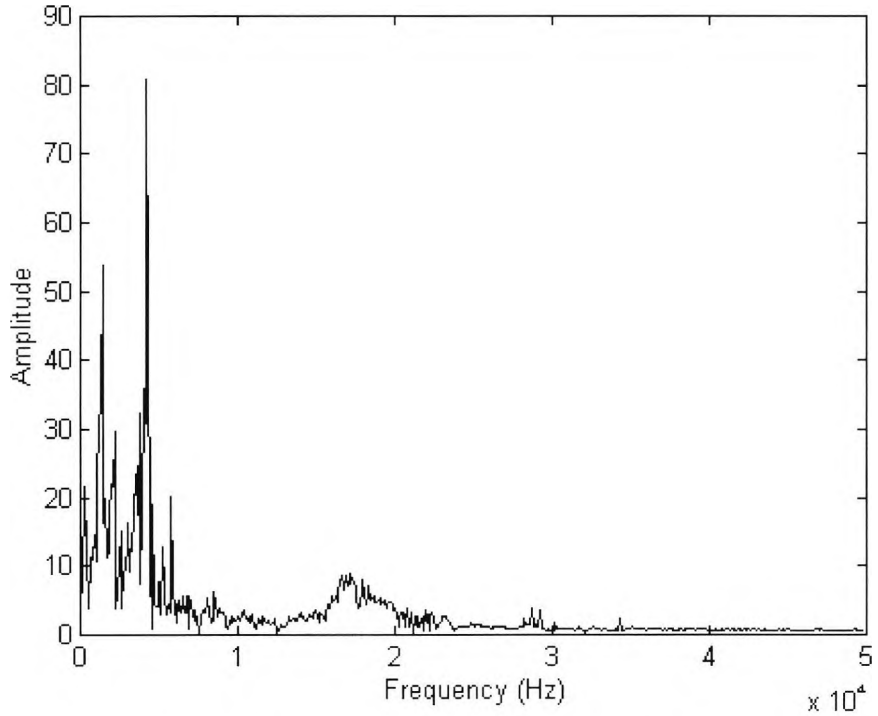


Figure F4: Amplitude spectrum of Wav2e

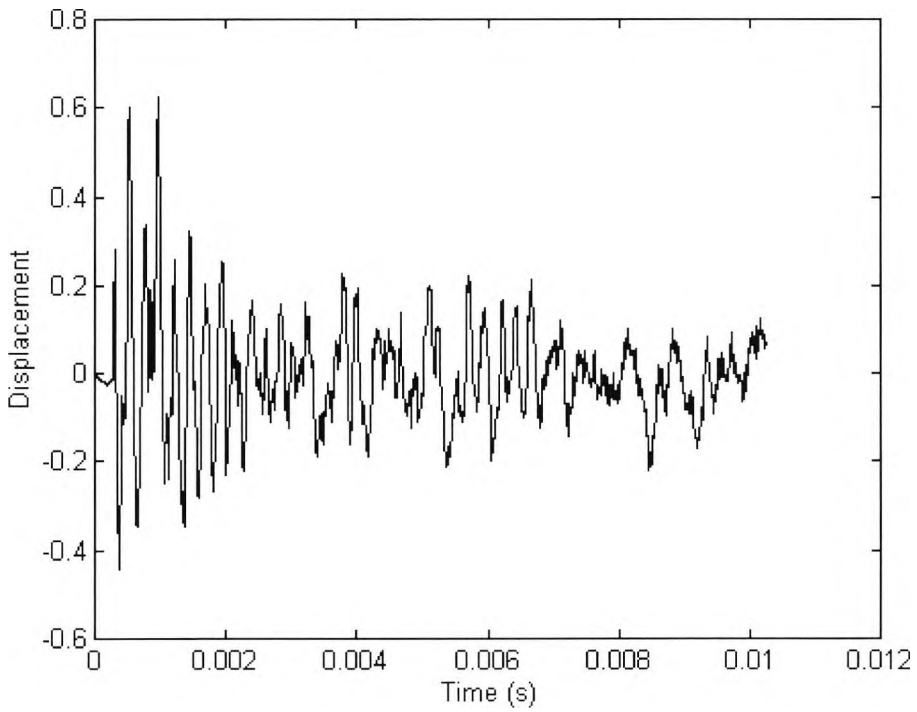


Figure F5: Waveform of Wav3e

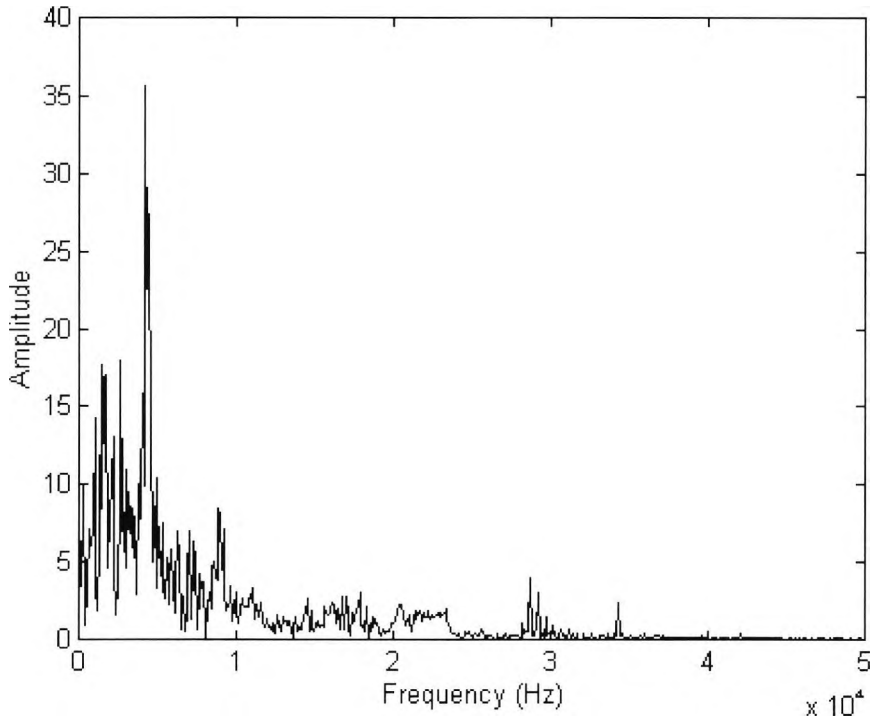


Figure F6: Amplitude spectrum of Wav3e

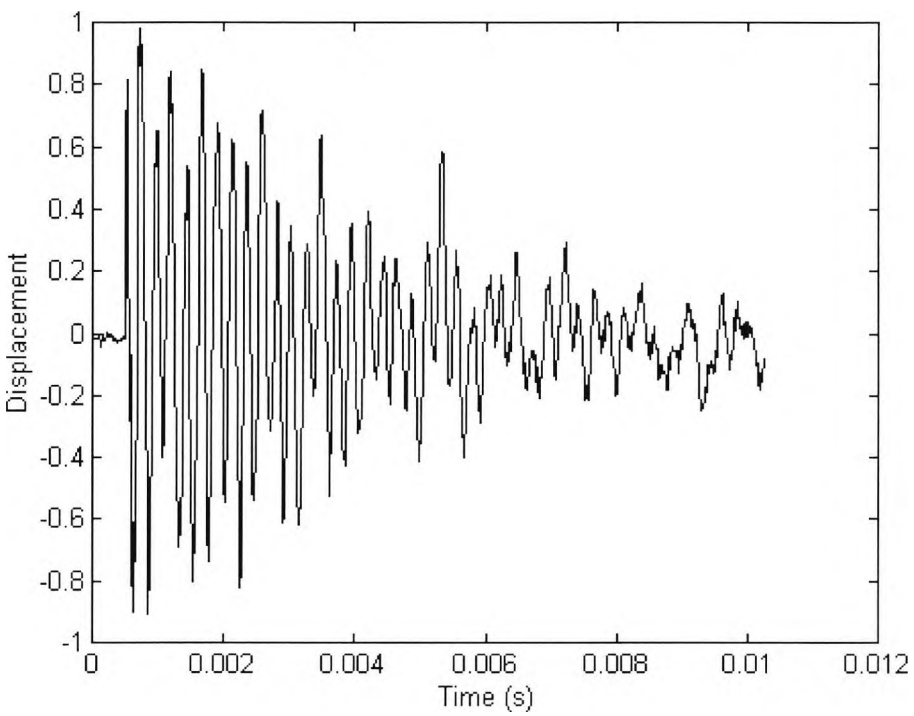


Figure F7: Waveform of Wav4e

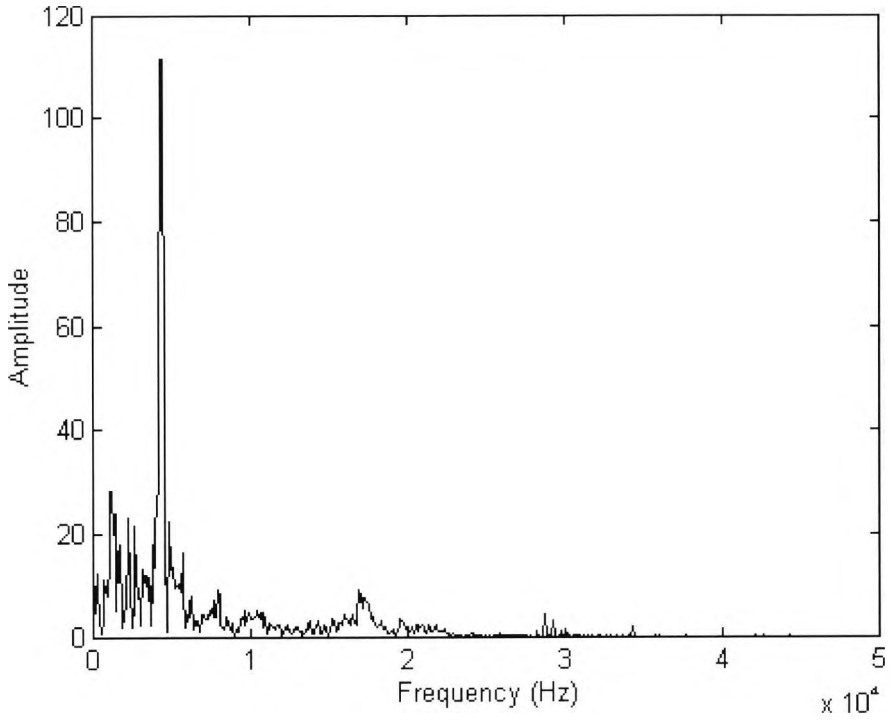


Figure F8: Amplitude spectrum of Wav4e

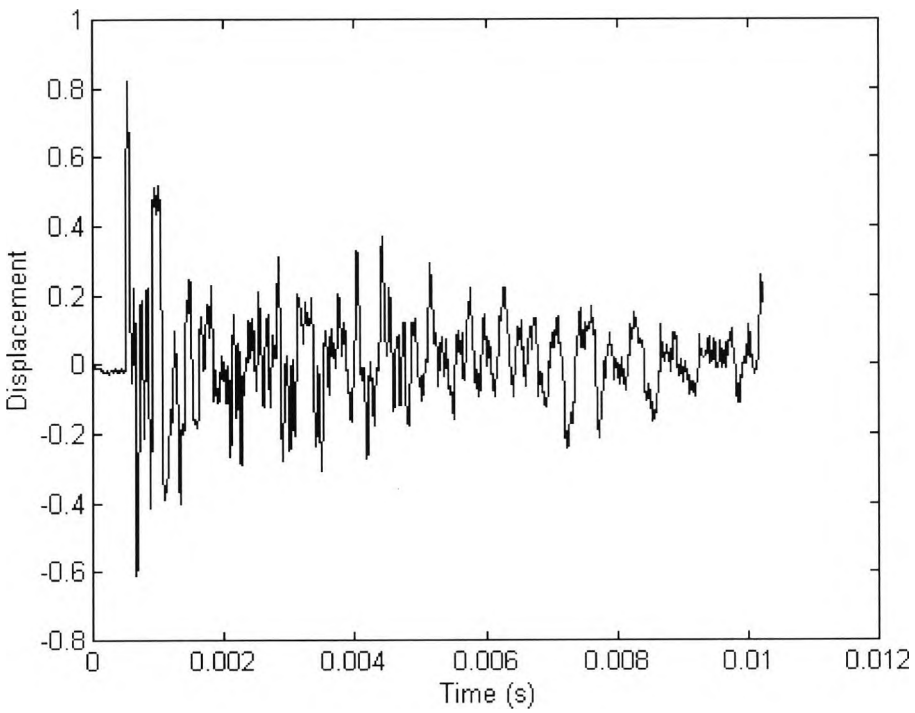


Figure F9: Waveform of Wav1w

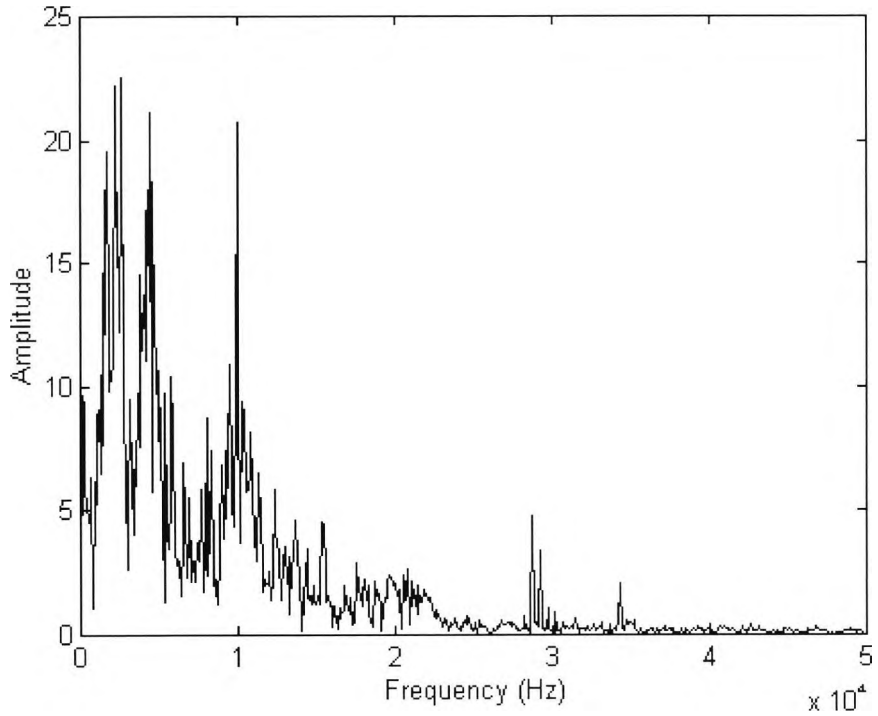


Figure F10: Amplitude spectrum of Wav1w

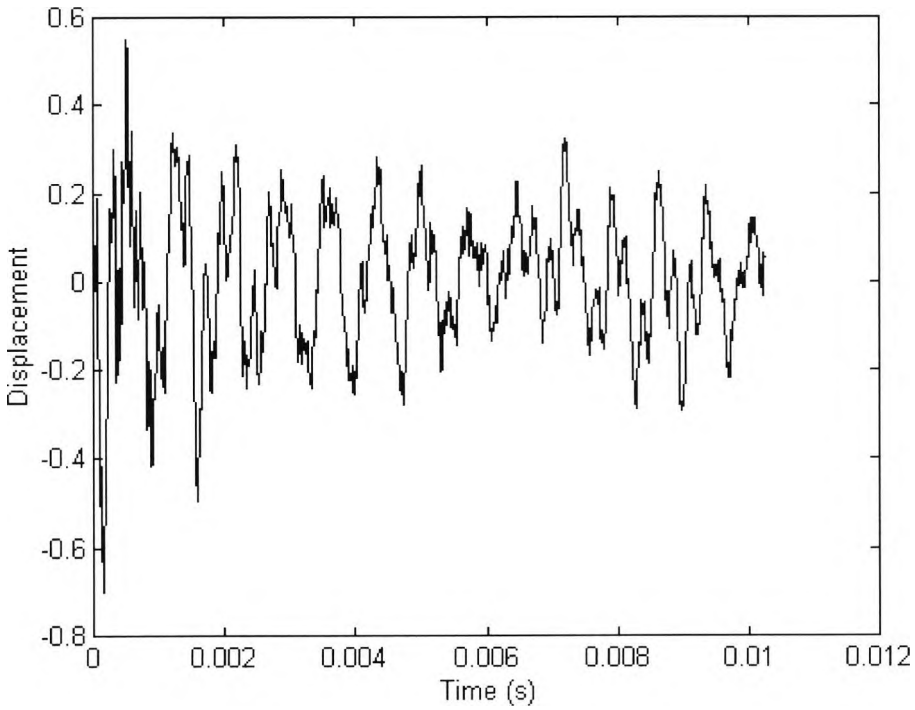


Figure F11: Waveform of Wav2w

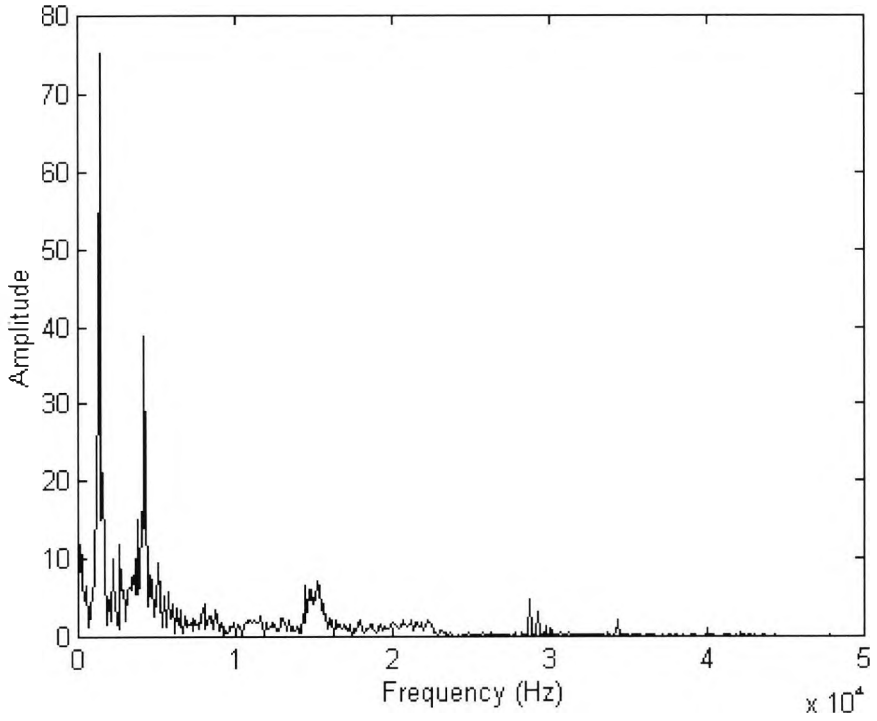


Figure F12: Amplitude spectrum of Wav2w

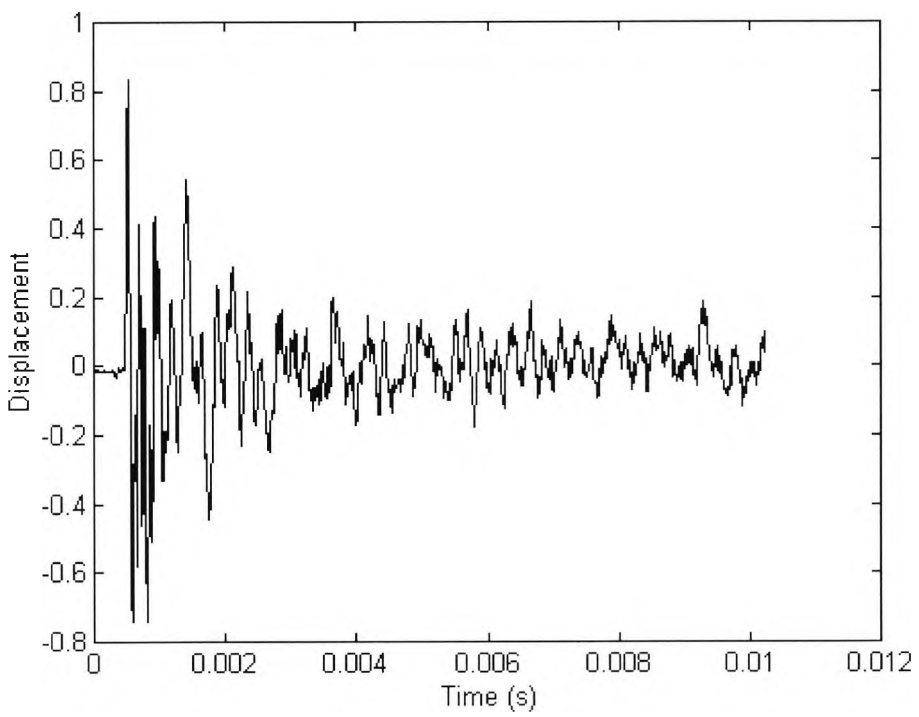


Figure F13: Waveform of Wav3w

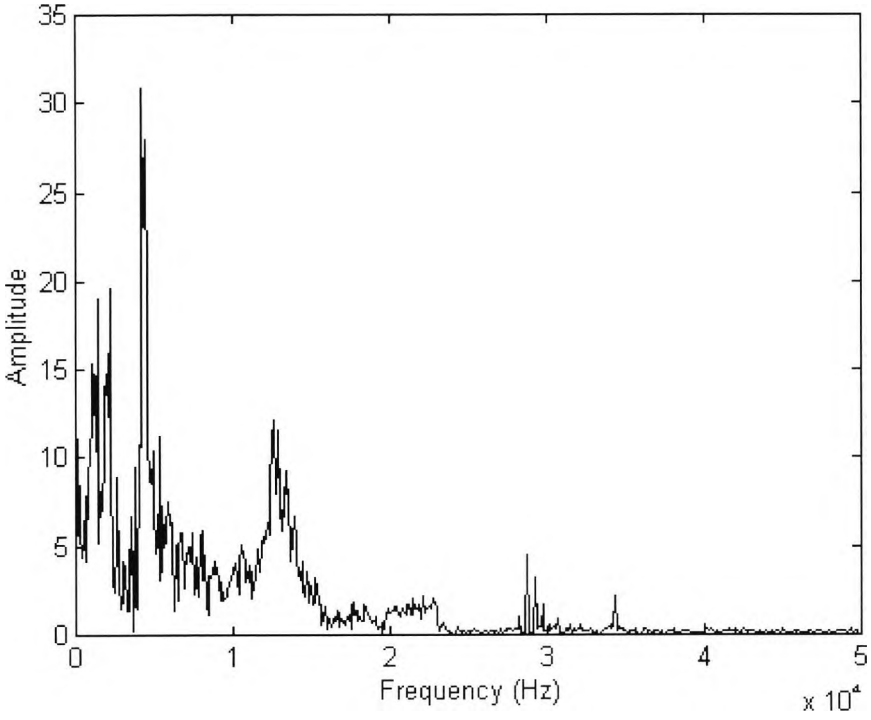


Figure F14: Amplitude spectrum of Wav3w

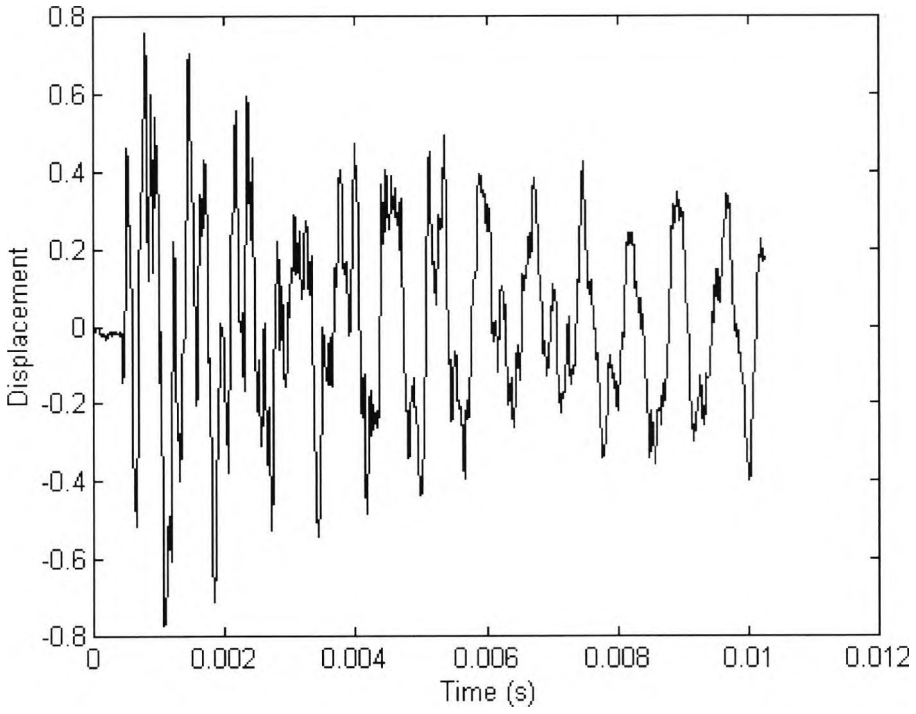


Figure F15: Waveform of Wav4w

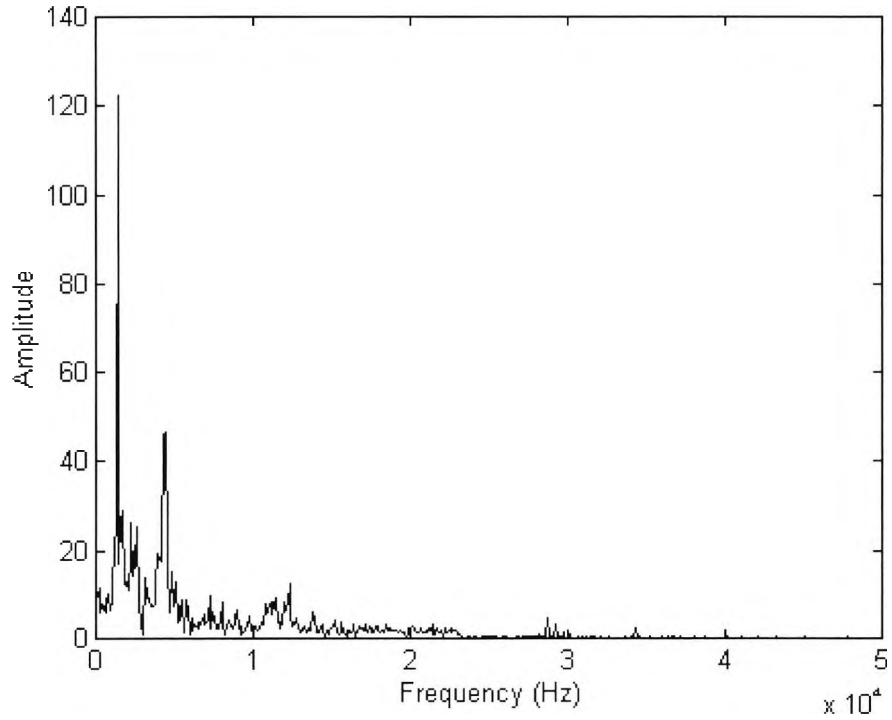


Figure F16: Amplitude spectrum of Wav4w

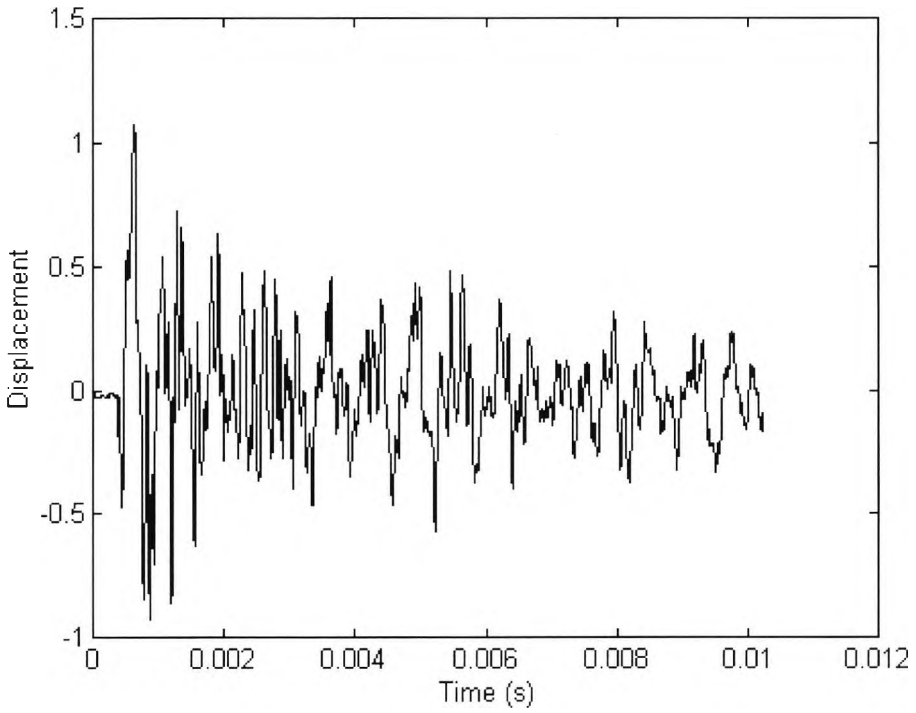


Figure F17: Waveform of Wdv5e

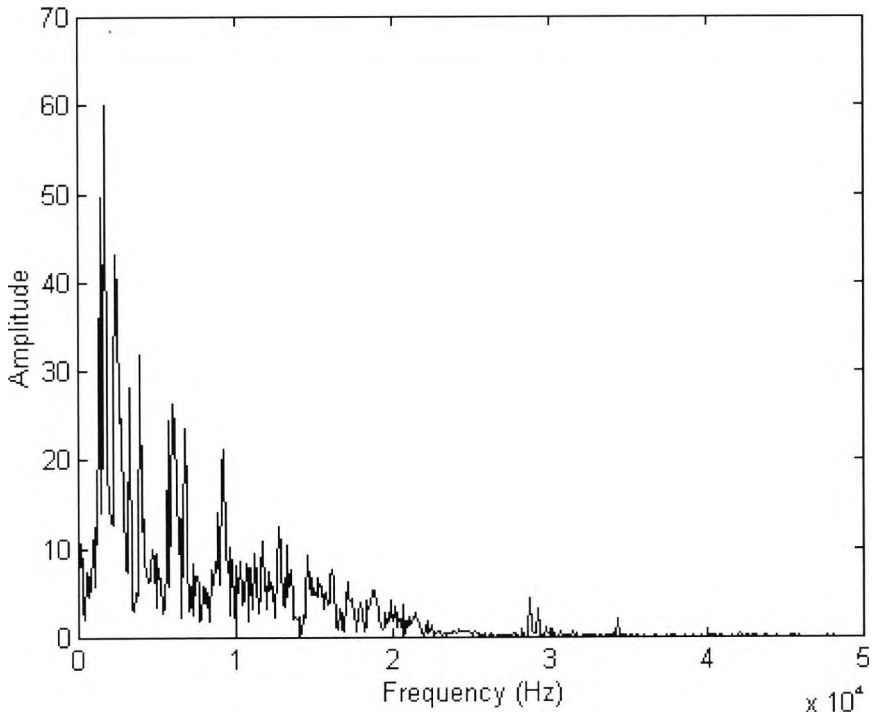


Figure F18: Amplitude spectrum of Wdv5e

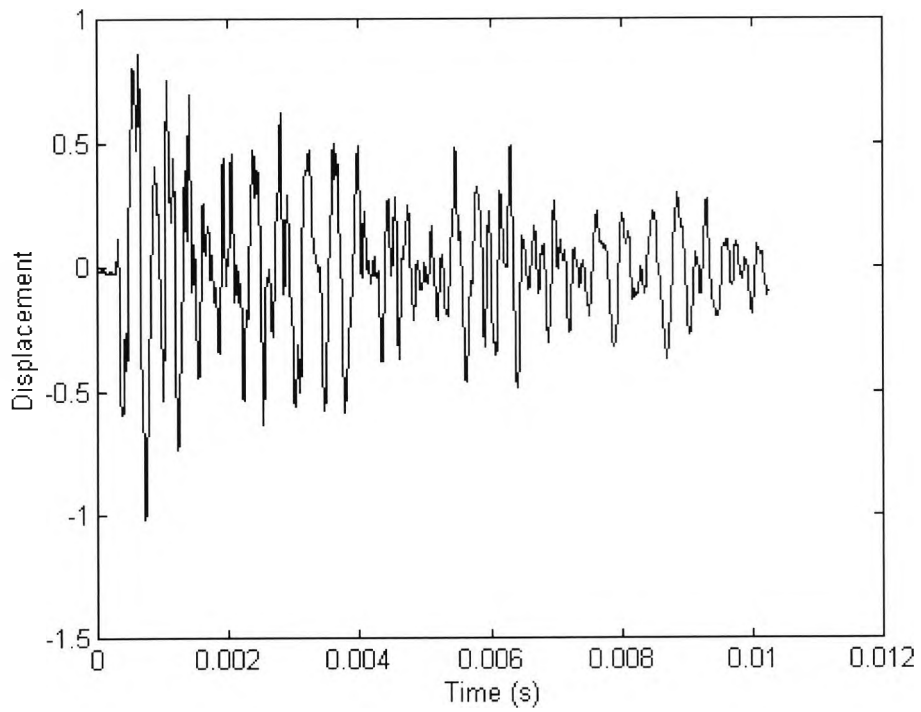


Figure F19: Waveform of Wdv6e

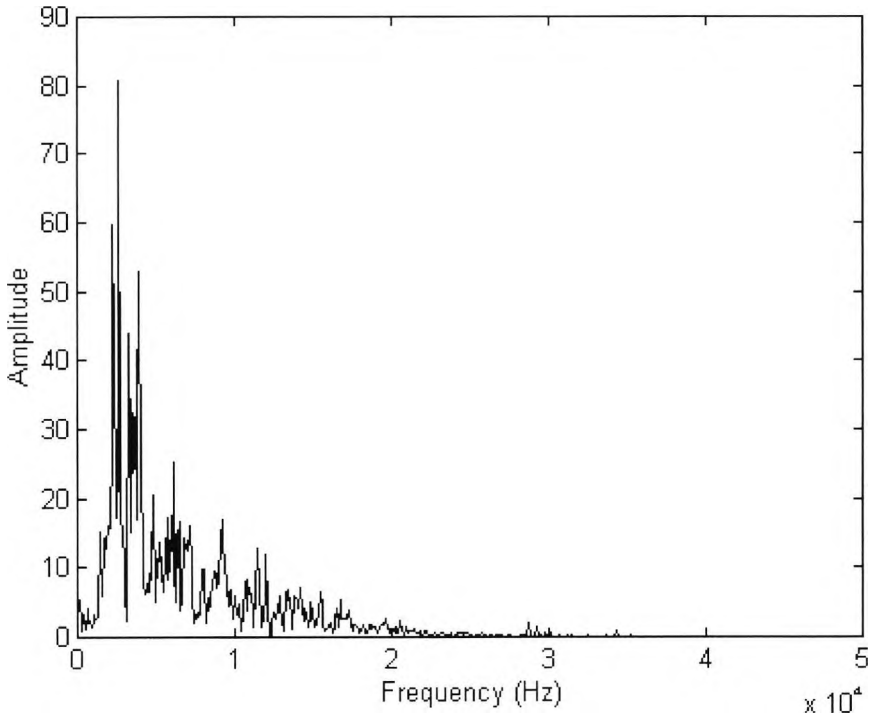


Figure F20: Amplitude spectrum of wdv6e

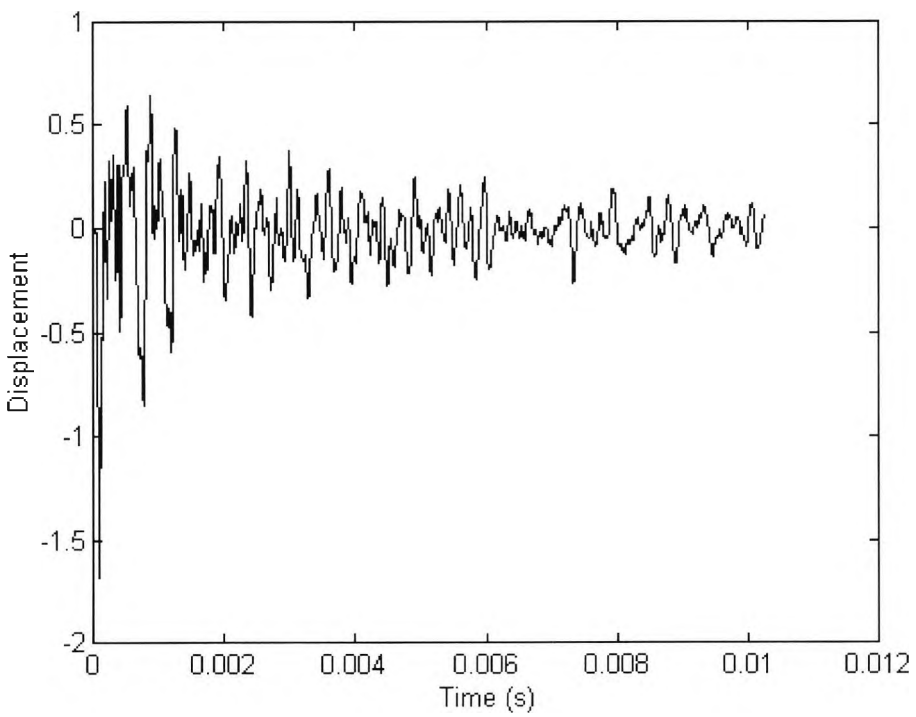


Figure F21: Waveform of Wdv7e

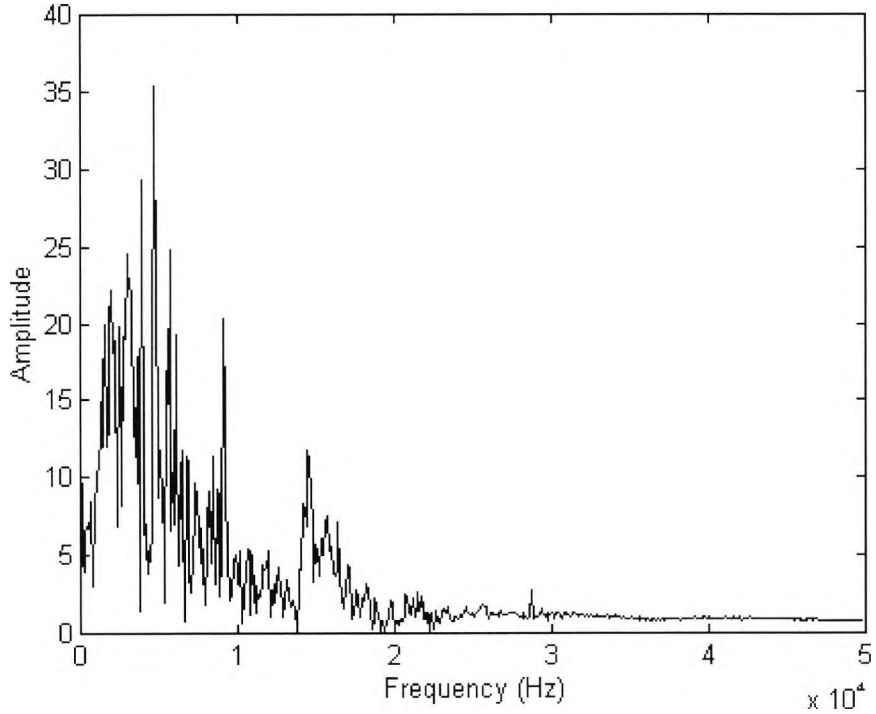


Figure F22: Amplitude of Wdv7e

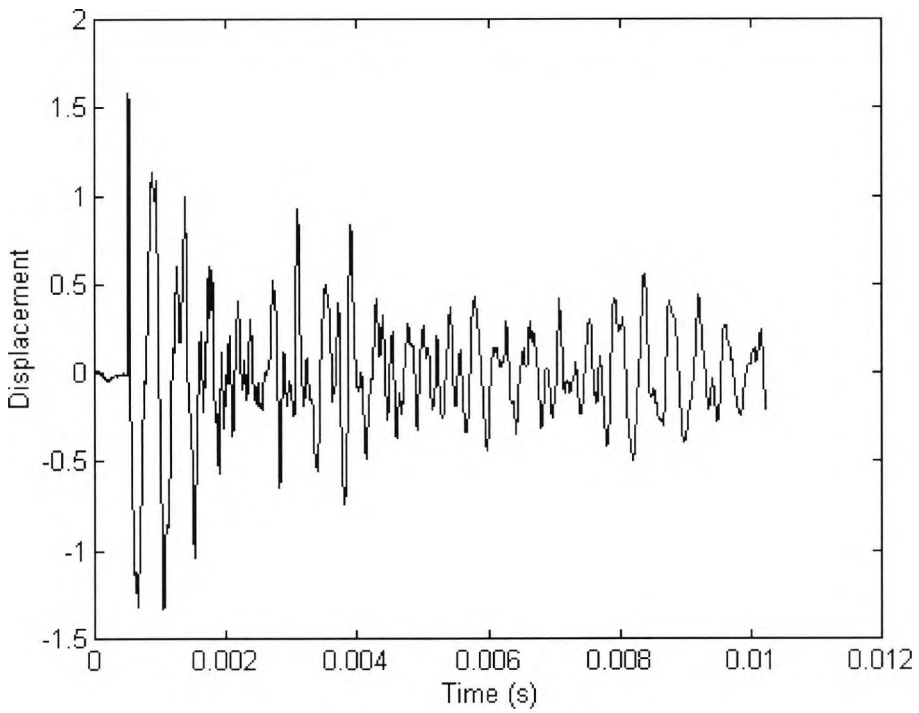


Figure F23: Waveform of Wdv8e

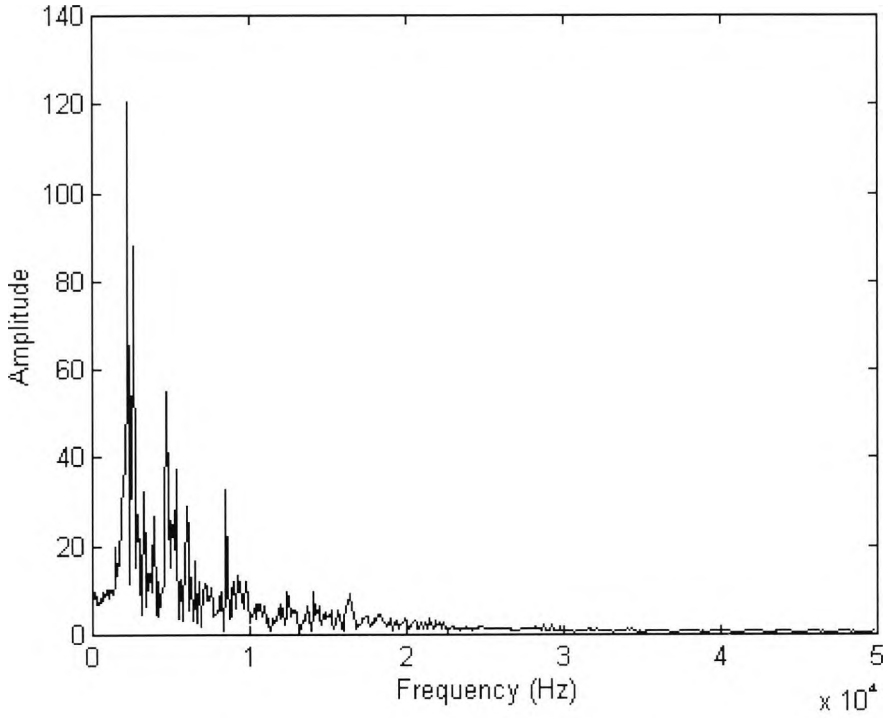


Figure F24: Amplitude of Wdv8e

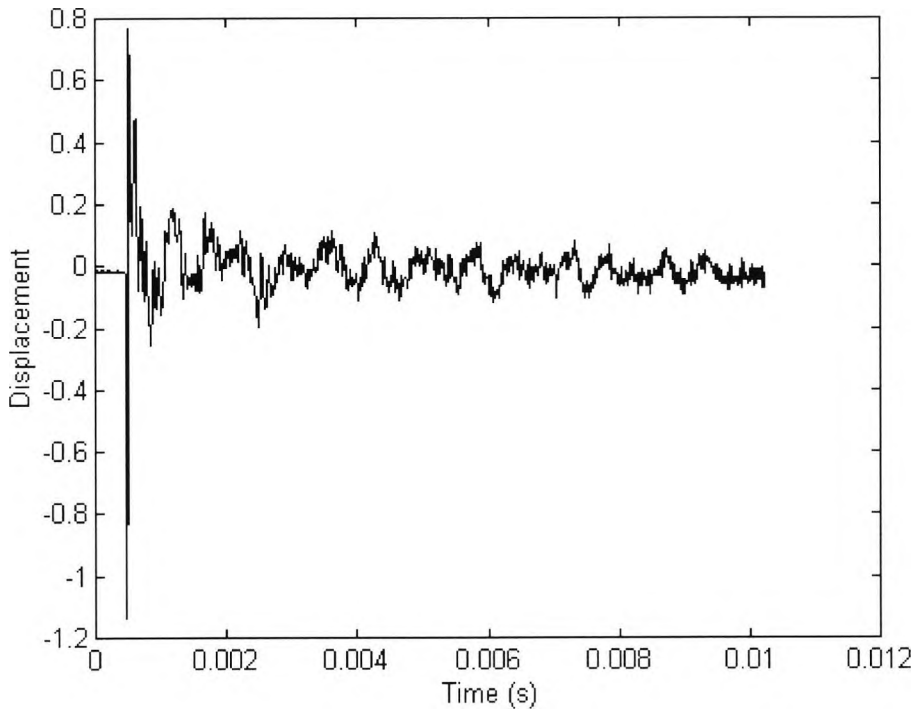


Figure F25: Waveform of Wacl1

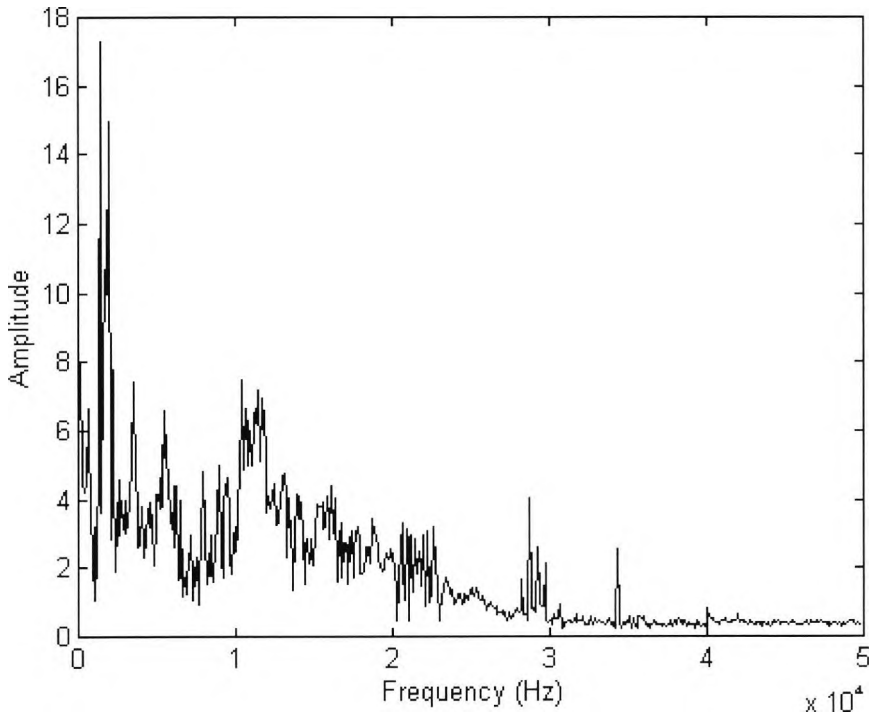


Figure F26: Amplitude of Wacl1

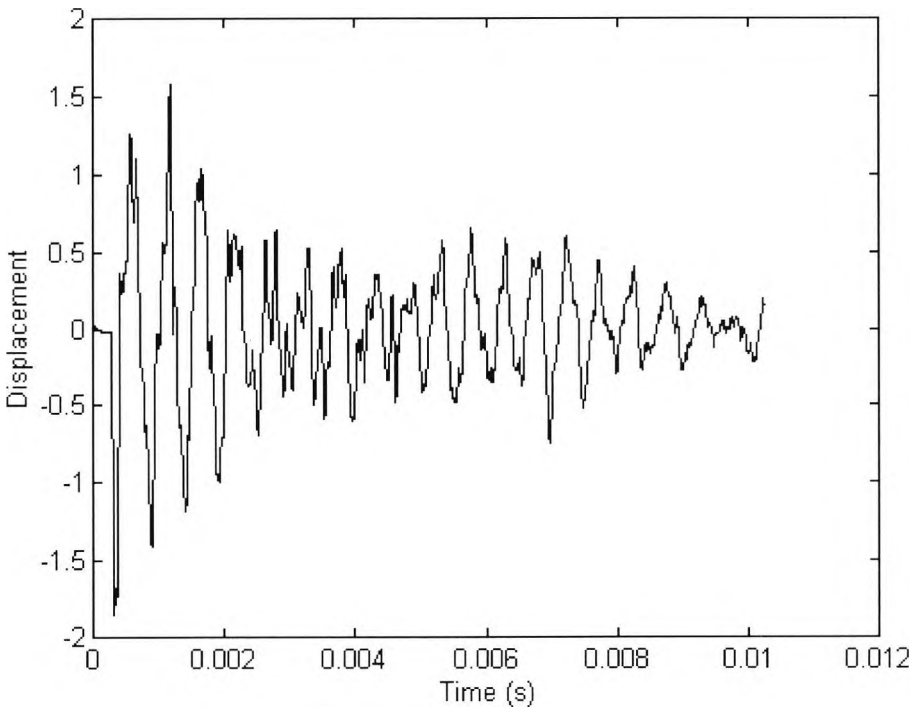


Figure F27: Waveform of Wacl2

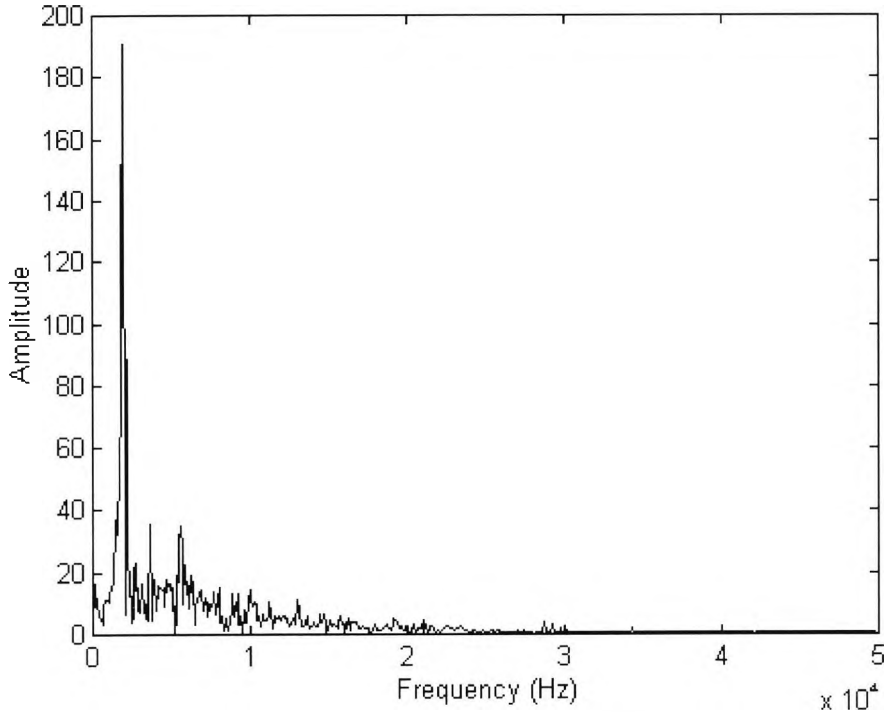


Figure F28: Amplitude spectrum of Wacl2

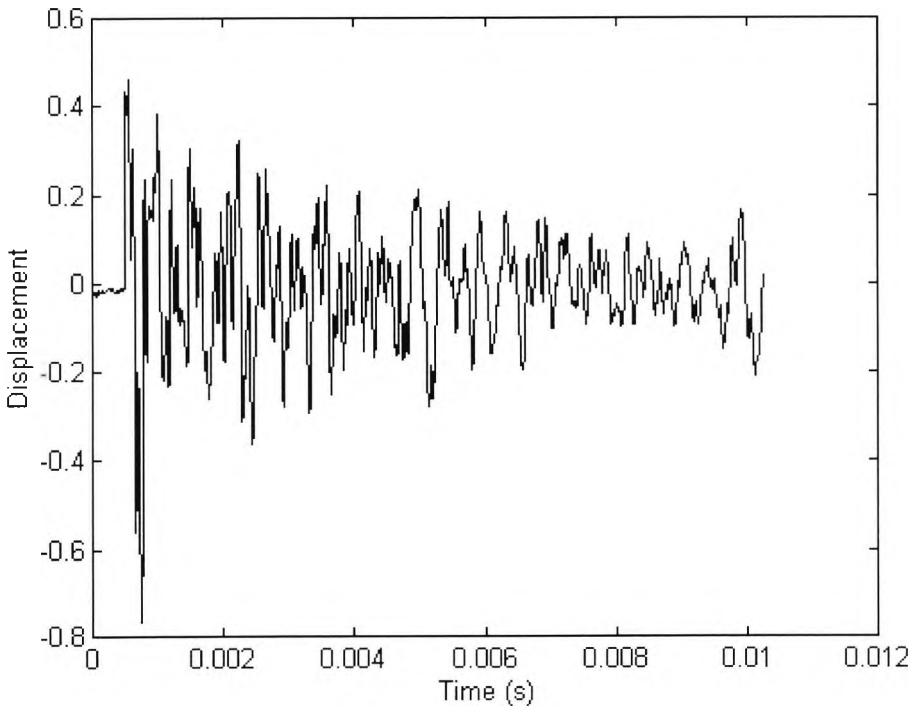


Figure F29: Waveform of Wdcle

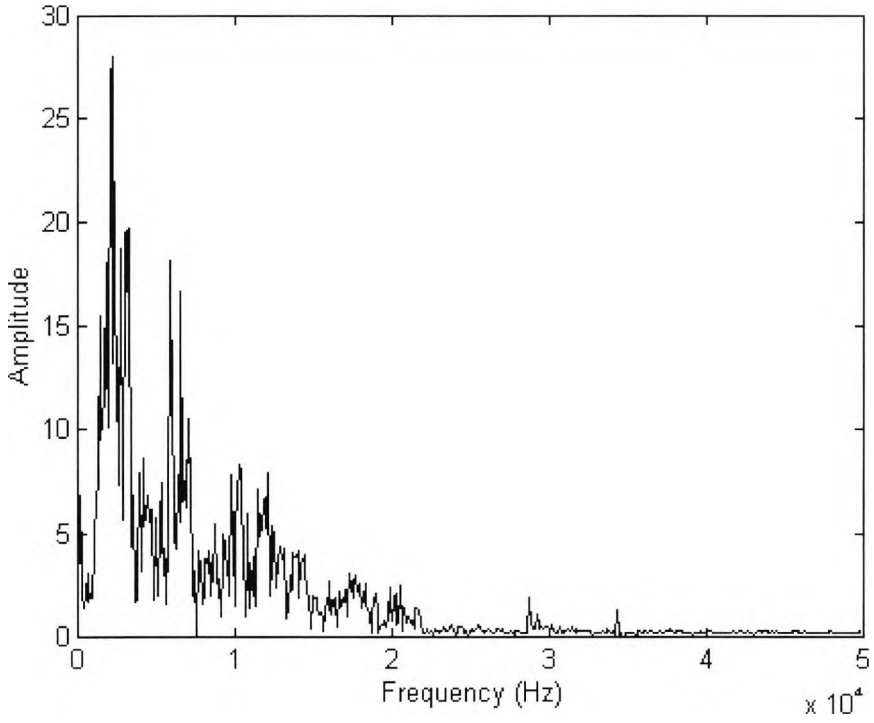


Figure F30: Amplitude spectrum of Wdcle

Program to determine FFT and plot a 2D graph

```
Load idata.txt **** load text file idata ****
idata **** idata is displayed ****
t=10e-6:10e-6:10240e-6; *** defining time increment of 10e-6
plot(t,idata) *** plot waveform ***
xlabel('Time (s)') *** labelling the xaxis ***
ylabel('Displacement') *** labelling the yaxis ***
Y=fft(idata,1024); **** do fft on the idata which is a one dimensional array of 1024 points ****
N=1024; *** sampling points defined ***
f=(0:511)*100000/N; *** calculation of actual frequency ***
plot(f,abs(Y(1:512))), title('Amplitude Spectrum') *** plotting the amplitude graph ***
xlabel('Frequency (Hz)') *** labelling the xaxis ***
ylabel('Amplitude') *** labelling the yaxis ***
Amp=abs(Y(1:512)); *** defining the new array of fft values ***
fprintf('mori.txt','%f\n',Amp) *** printing the fft values to a file ***
```

Addition of noise to the original data

The following graphs illustrate the effect of noise with various standard deviation discussed in chapter 7. Record 8 of Wav1e has been used to show the addition of noise.

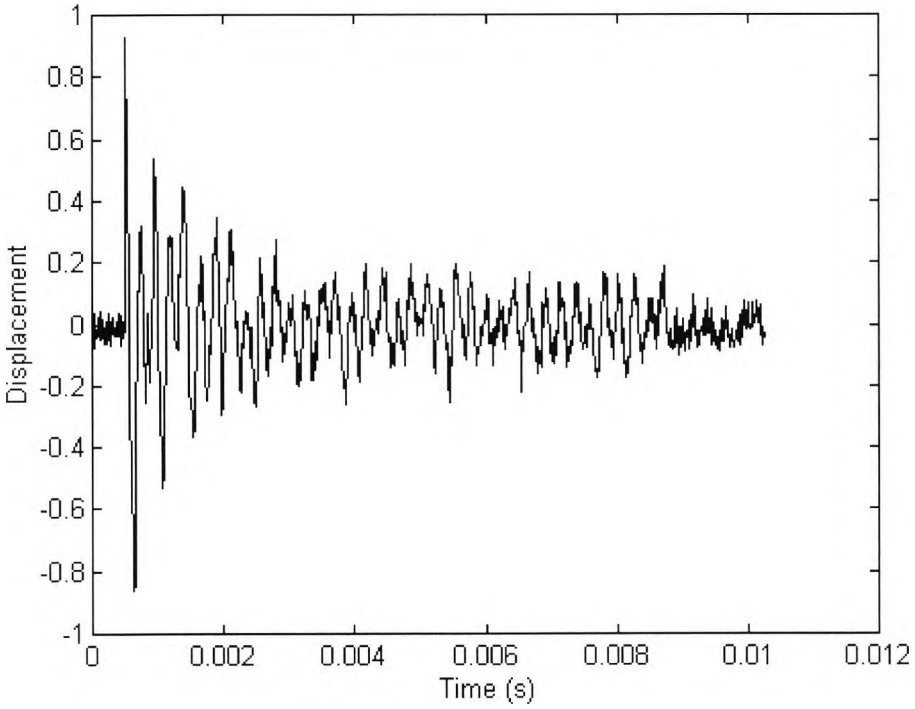


Figure F31: Addition of noise, standard deviation = 0.03

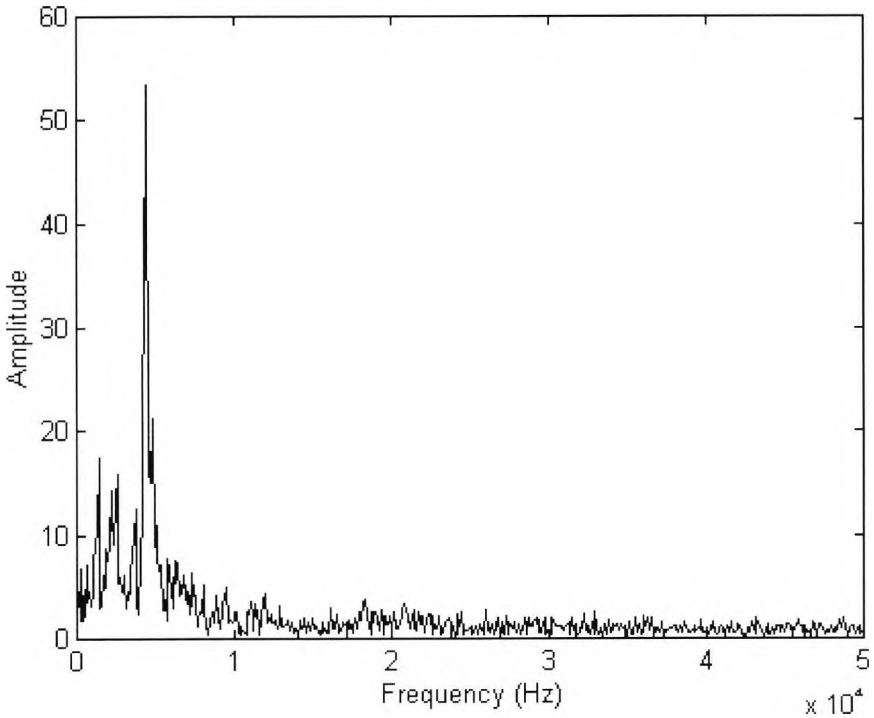


Figure F32: Amplitude spectrum , standard deviation = 0.03

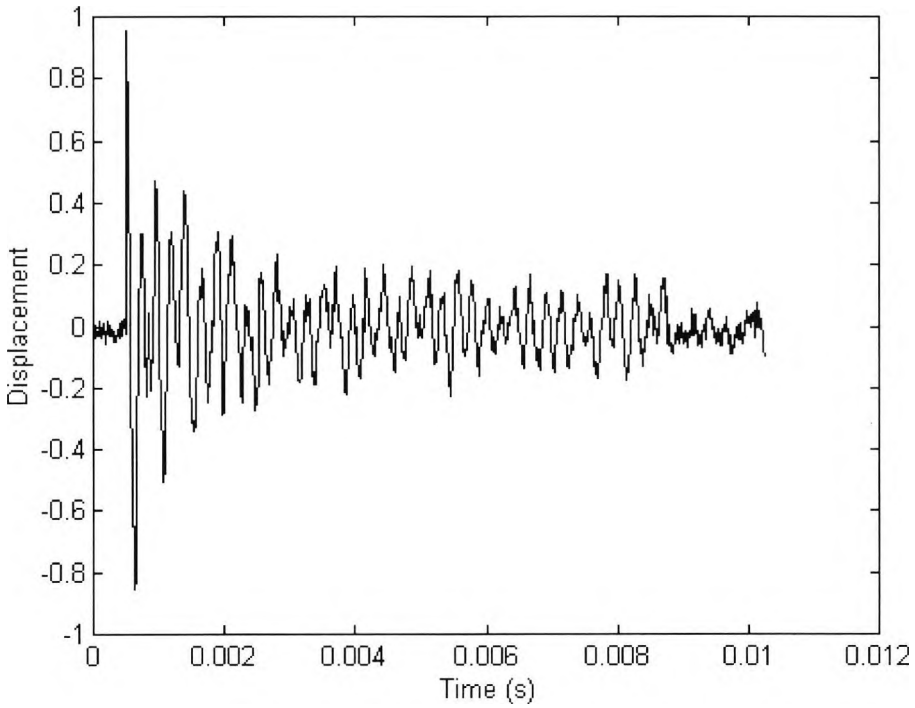


Figure F33: Addition of noise, standard deviation = 0.02

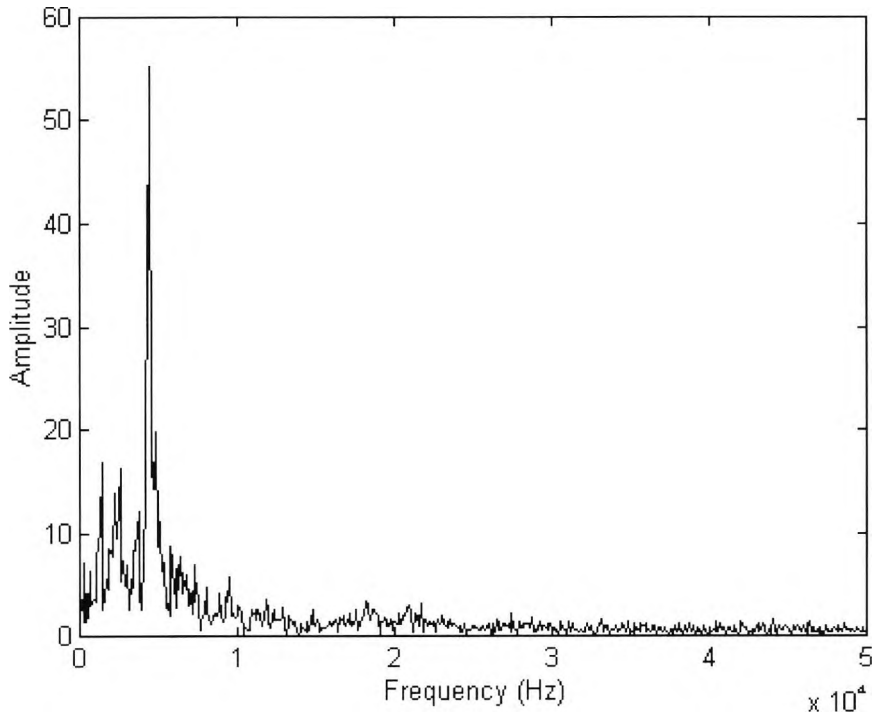


Figure F34: Amplitude spectrum, standard deviation = 0.02

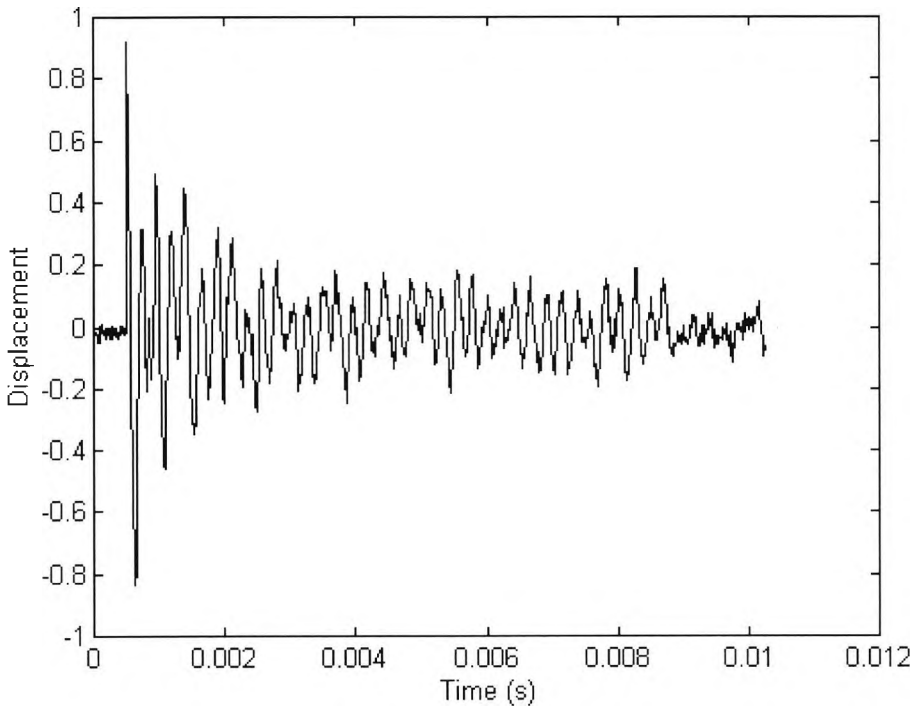


Figure F35: Addition of noise, standard deviation = 0.015

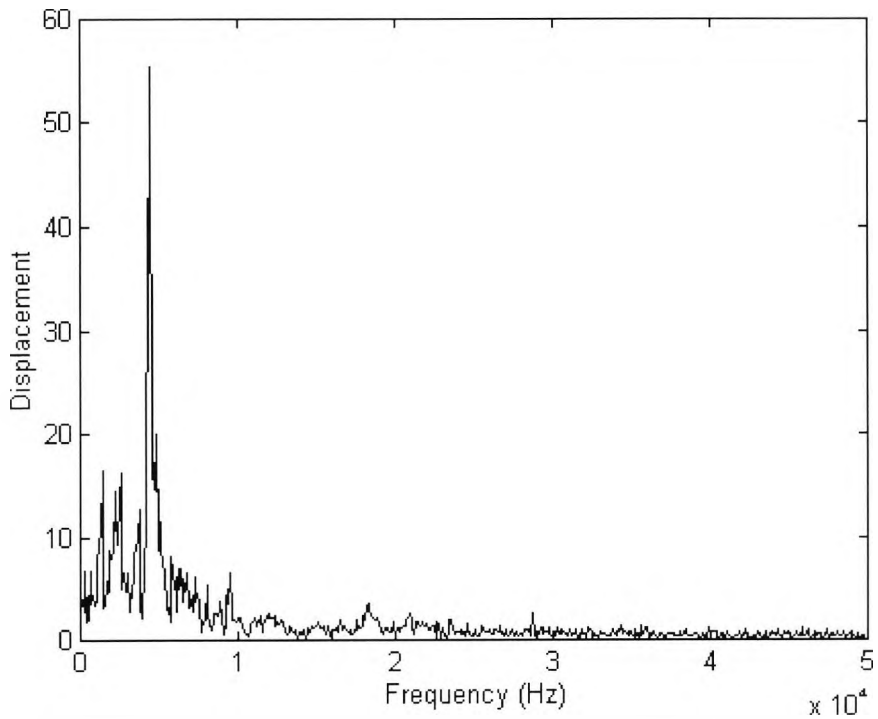


Figure F36: Amplitude spectrum, standard deviation = 0.015

Labview program which creates the addition of noise

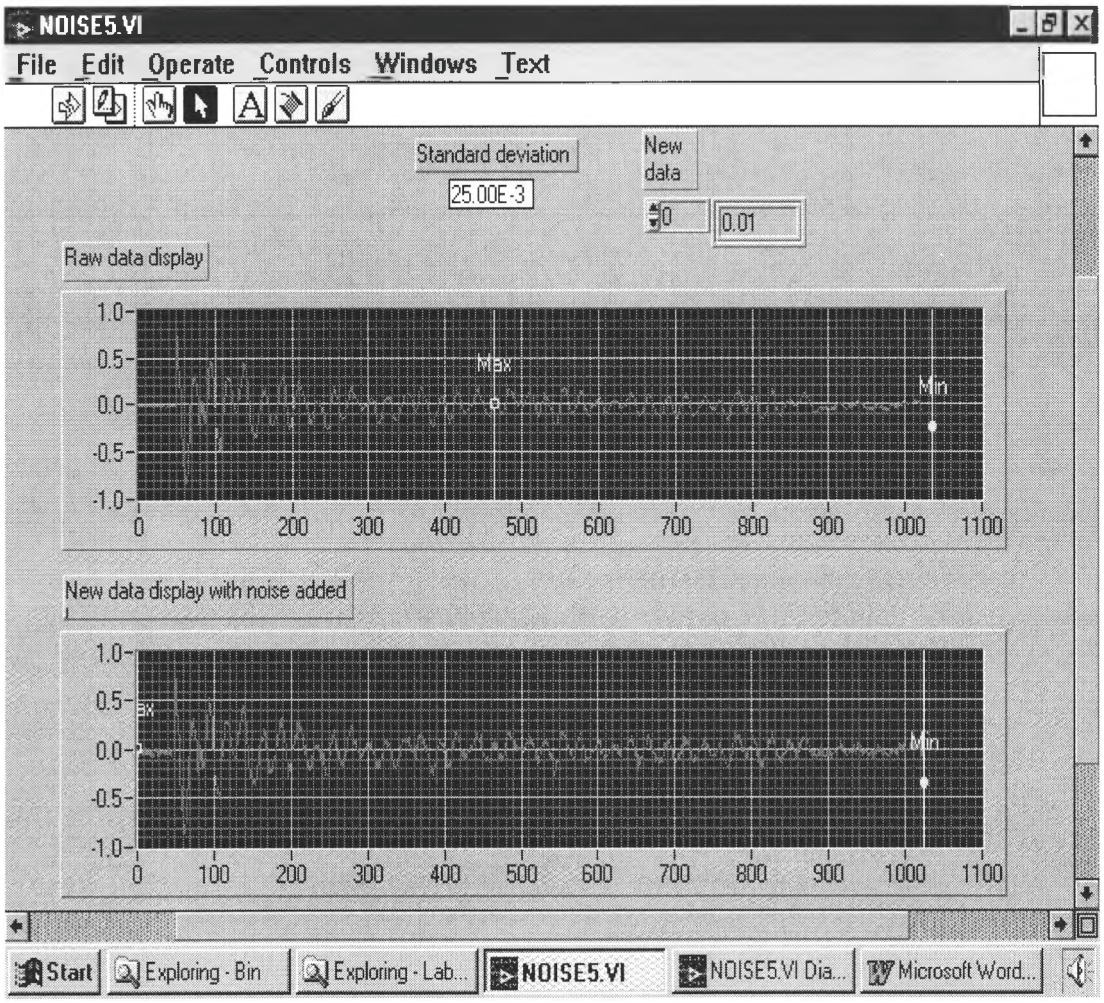


Figure F37: Front panel of the Labview program

The Labview program consist of two parts, one is the 'front panel' indicated in figure F37, the other is the 'diagram panel' shown in figure F38. As shown in figure F37, there are two graphs, the first displays the original one dimensional array of the actual data before the addition of noise. The second shows the effect of the noise added which is controlled by the measure of standard deviation. The 'new data' is just an indicator to show the new one dimensional array with the addition of noise.

Figure F38: Gaussian noise VI

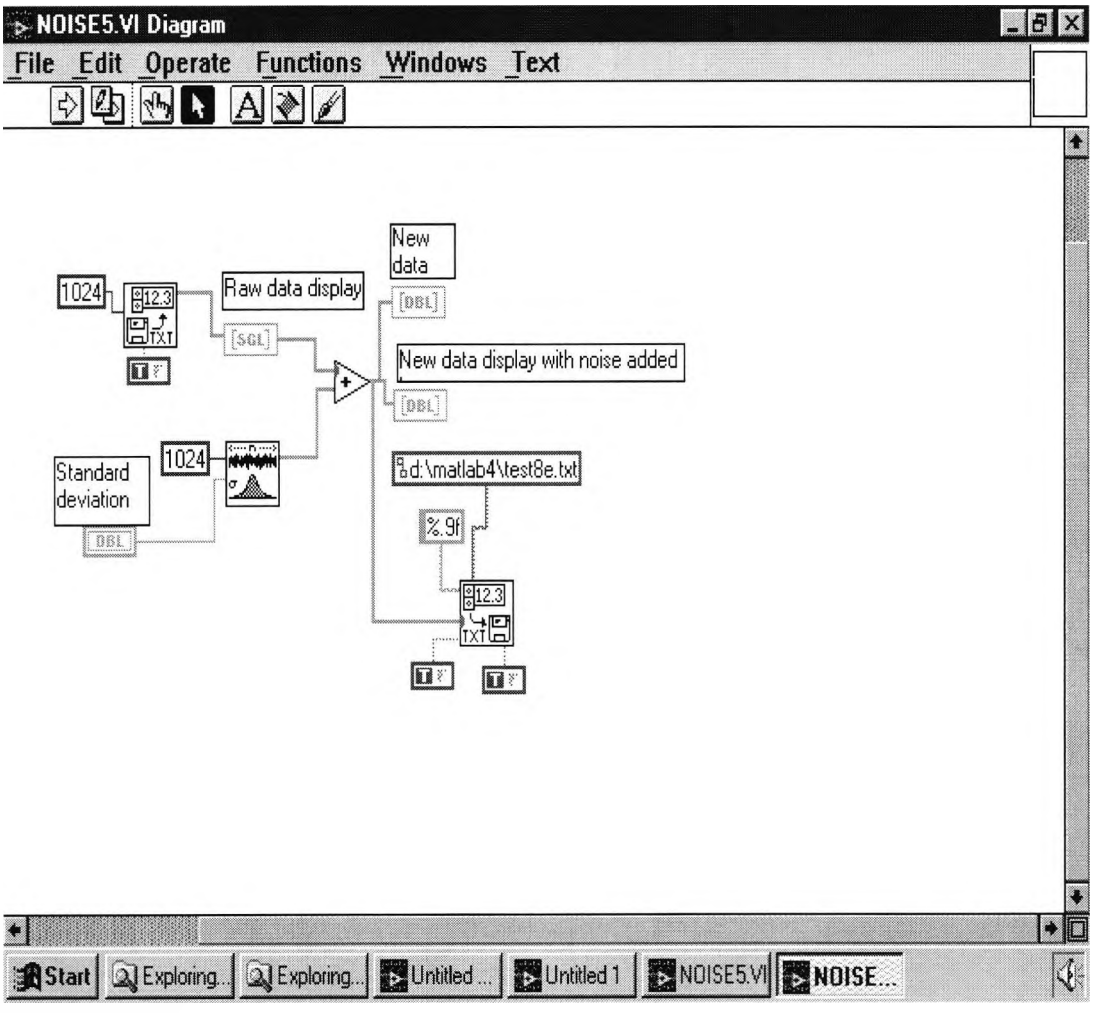


Figure F39: Diagram panel

Figure F39 illustrates many 'icons' which are called VI, these VI perform certain functions, figure F38 shows the VI which creates a level of noise depending on the size of samples and standard deviation for example.

In figure F39, a file is open, the data is plotted, then addition of noise is added to the original data, with the selected value of standard deviation. Now, this new data is plotted (shown in figure F37) and written to a new file.

Interpolation of graphs

As discussed in chapter 7, the variety of sampling points are shown in figure F50 to F56.

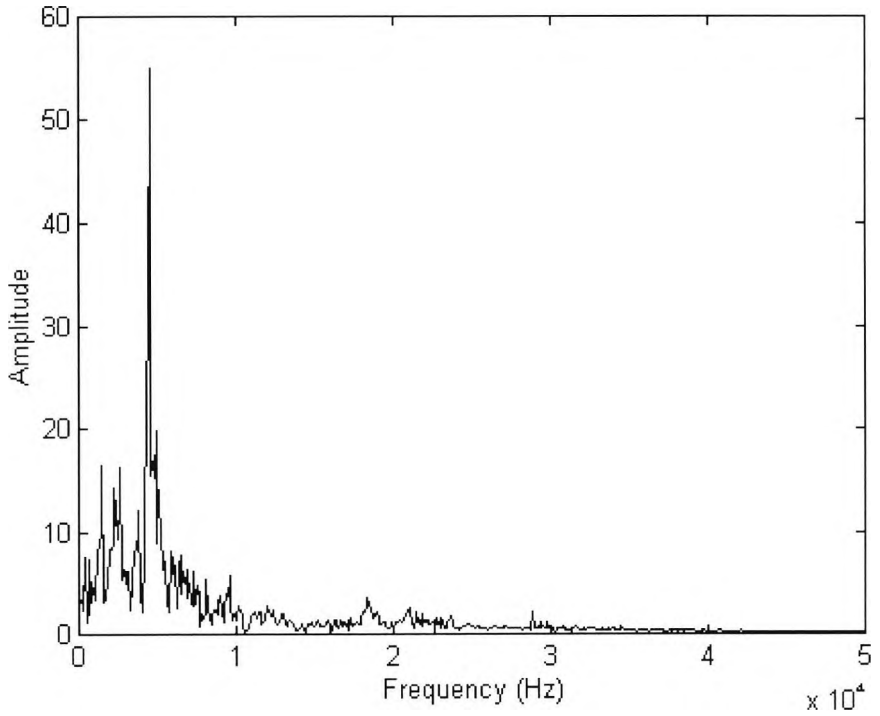


Figure F40: Amplitude graph with 512 points (97.65625 Hz frequency increment)

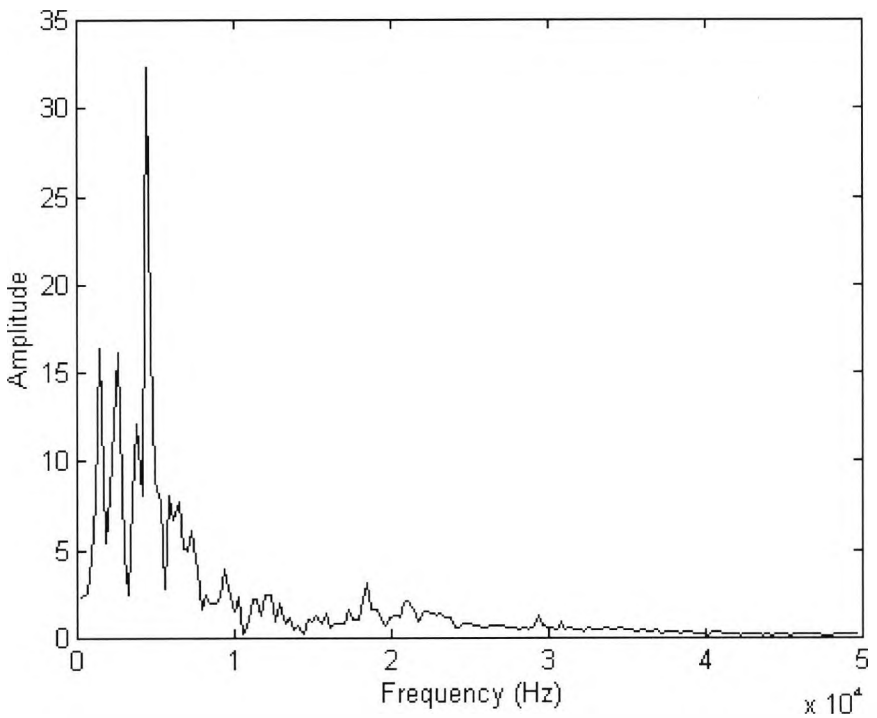


Figure F41: Amplitude spectrum with 256 points (195.3125 Hz frequency increment)

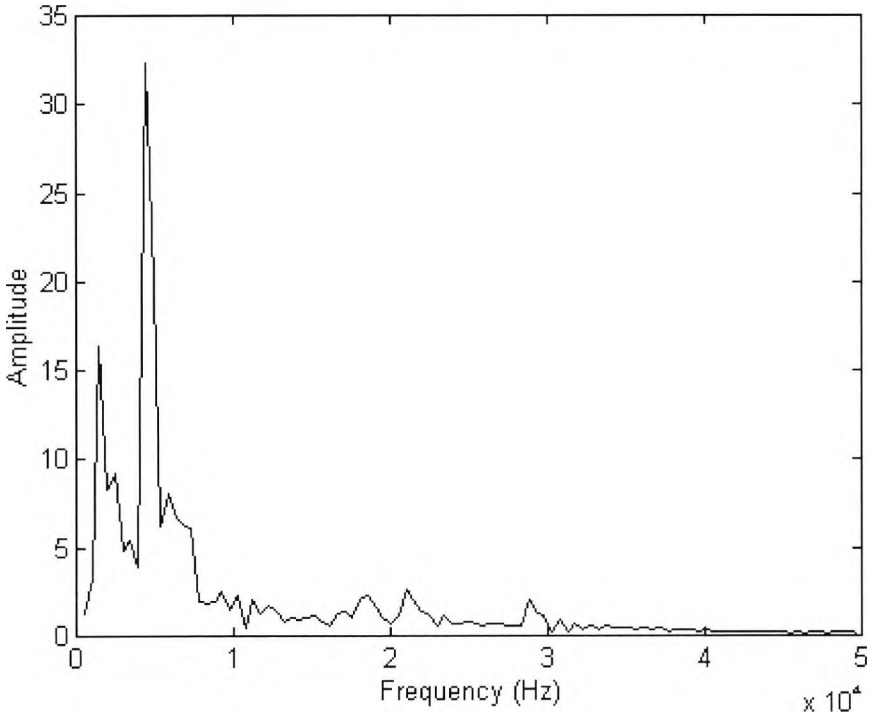


Figure F42: Amplitude spectrum with 128 points (390.625 Hz frequency increment)

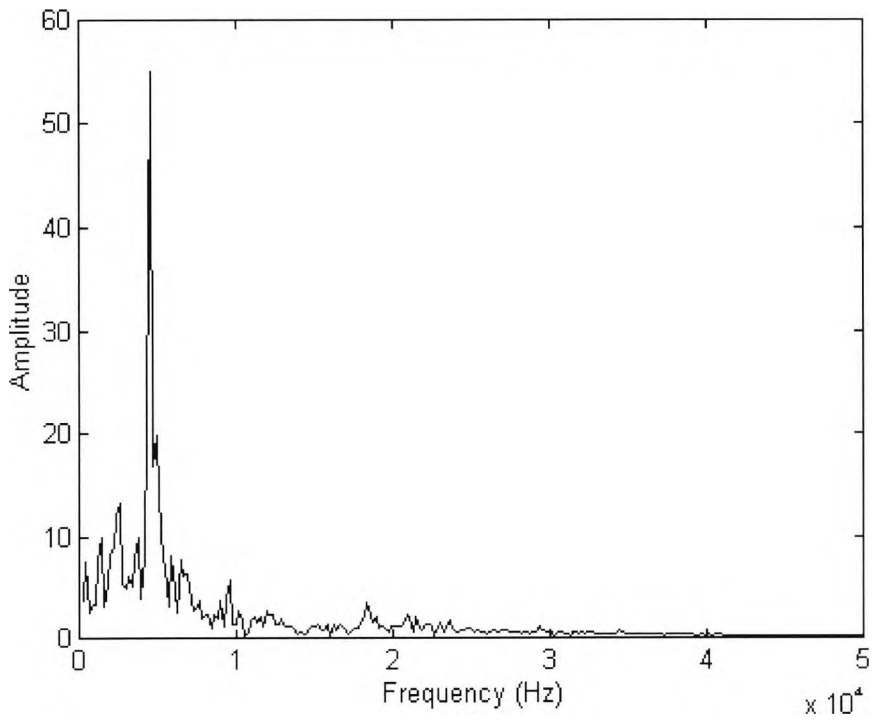


Figure F43: Amplitude spectrum with 102 points (488.28125 Hz frequency increment)

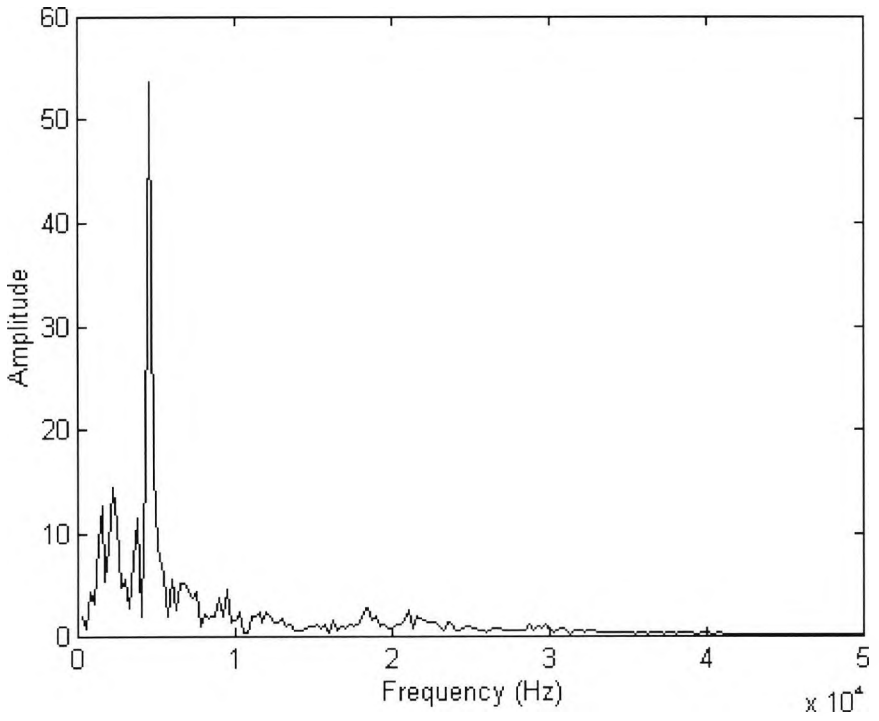


Figure F44: Amplitude spectrum with 200 points (250 Hz frequency increment)

Program for interpolation

load proc1.txt ***load the file***
proc1 ***Type in the name of file as shown***
proc1 =

5.6214
8.3275
4.6177
6.2319
5.7764
etc....

f1=97.65625:97.65625:50000; ***This is the original increment***
f2=292.96875:292.96875:50000; ***This is the new increment****
Y=interp1(f1,proc1,f2,'spline'); ***Find the new amplitude value for the new frequency incement as shown above using cubic spline interpolation(1D interpolation)***
fprintf('pol1.txt','%f\n',Y) ***print this to a file***
plot(f2,Y) *** plot the new interpolated data***
xlabel('Frequency (Hz)') *** label x axis ***
ylabel('Amplitude') *** label y axis ***

Labview program to filter out unwanted frequencies

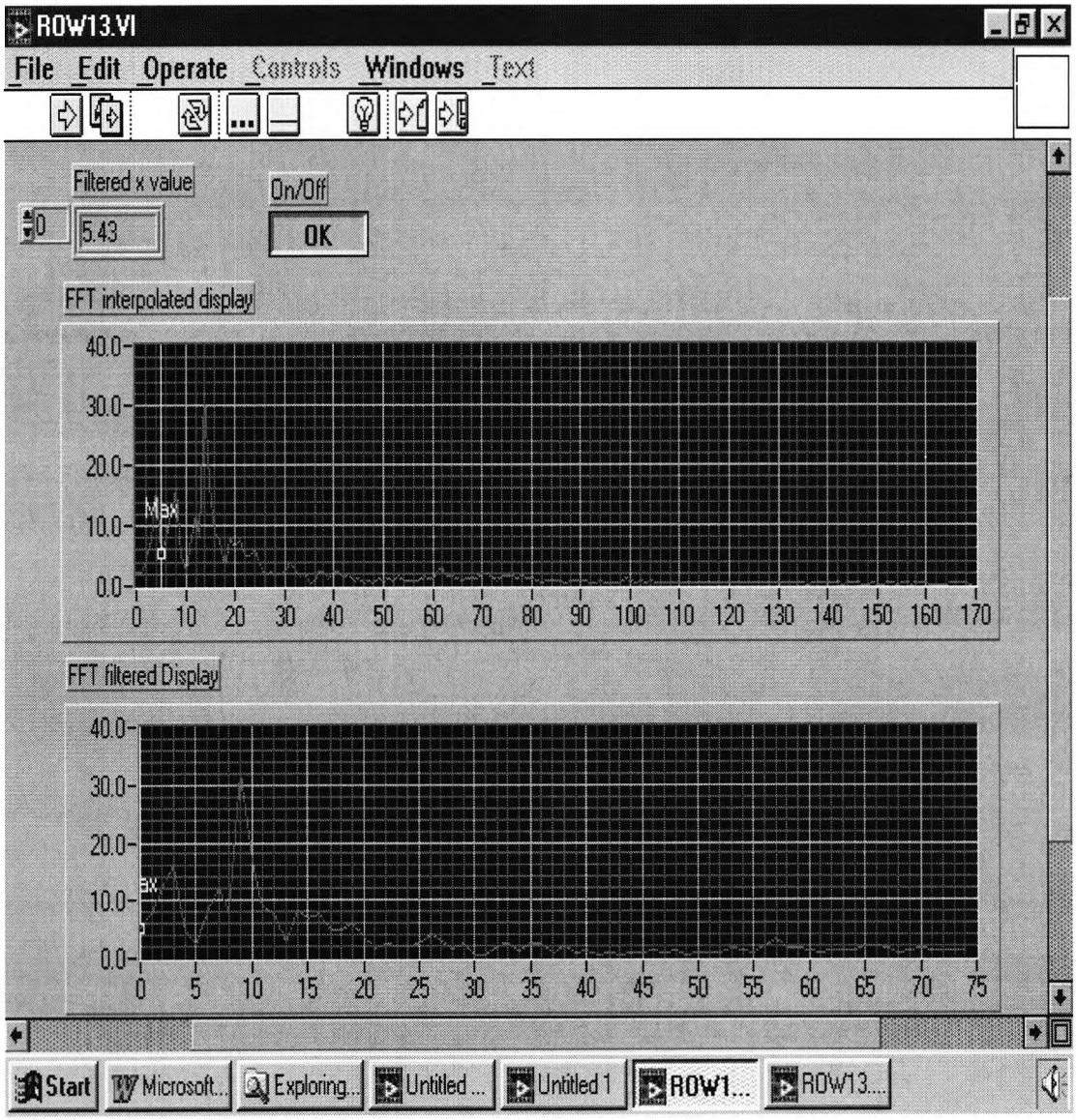


Figure F45: Front panel

Figure F45 shows the interpolated FFT graph with 170 points and the removal of unwanted frequencies. The diagram in figure F46 shows a selected file opened and plotted, then the first five points are removed from the graph, this is then displayed in the next graph. Also, the last 80 points are removed. This is then written to a new file called 'Rose1.xls', this will have 75 points of FFT data.

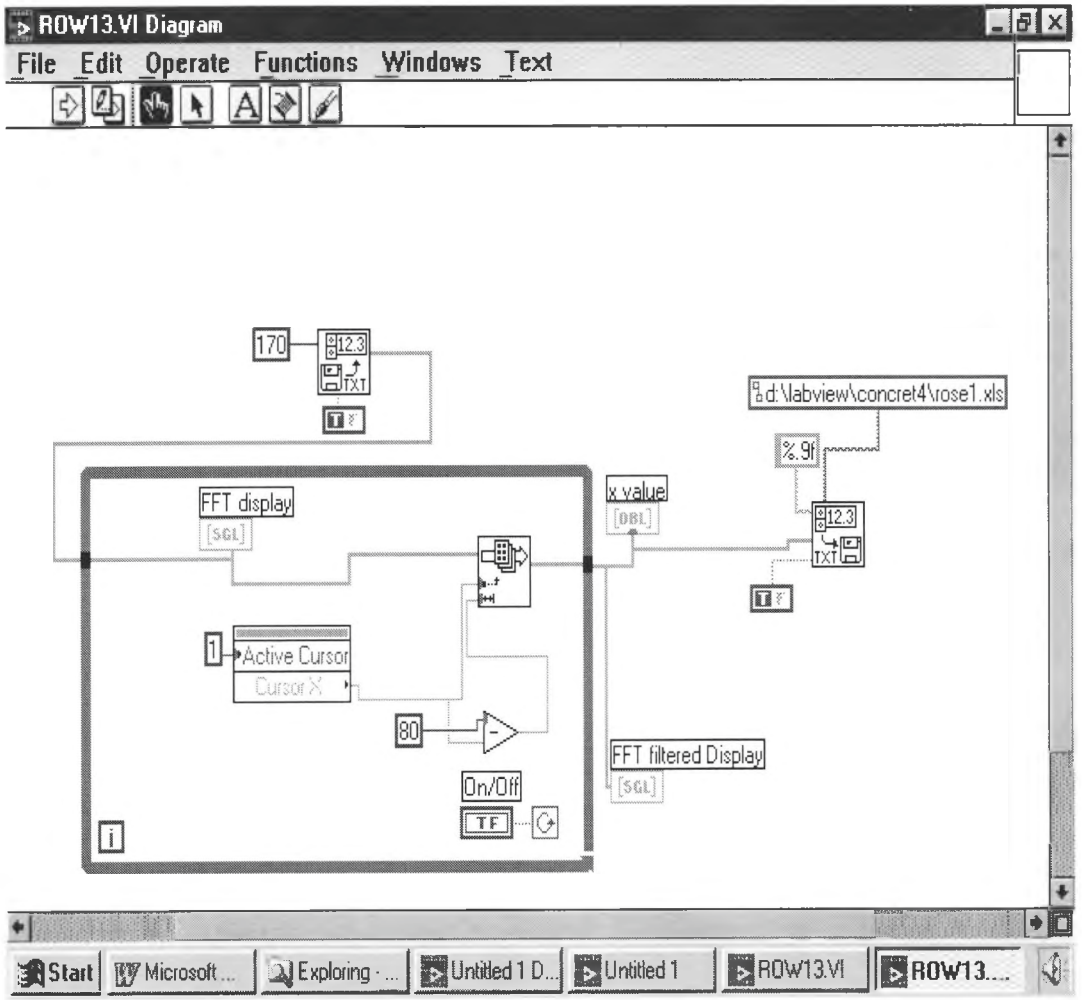


Figure F46: Diagram panel