Vowel Synthesis Using Feed-Forward Neural Networks

Stephen Malcolm Conway

Thesis submitted for the degree of PhD
University of Edinburgh
1994

Abstract

This thesis is an investigation into the ability of artificial neural networks to learn to map from a symbolic representation of CVC triphones to a continuous representation of vowel formant tracks, and the influence of a number of factors on that ability. This mapping is interesting because, apart from being a necessary part of any text to speech system and not having any accepted definitive solution, it is from a discrete symbolic representation to a continuous non-symbolic representation. Neural networks provide one method of automatically learning such mappings and prove to be capable of doing so in this particular case.

The input representation used appears to have little effect on the performance of the neural networks. A feature based representation does no better than a 1-of-n coding of the phonemes. The representation of the vowel formant tracks, produced as output of the neural networks, has a far greater effect on performance. Simple representations consisting of the initial, central and final frequencies of the formant tracks out-perform polynomial and Fourier coefficient representations which encode more information about the shape of the formant tracks.

The back-propagation and conjugate gradient neural network training algorithms produced neural networks with similar performance, and the use of cross-validation made no difference in generalisation (although the cross-validation data set was far too small). Interestingly, neural networks with no hidden layer proved to be as capable of learning the mapping as those with a hidden layer.

I have also derived a relationship predicting the result of a modified rhyme test from the root-mean-square error of the F1 vowel formant tracks of a set of utterances, and confidence bounds on that prediction. However, it would be unwise to apply this result to speech produced by other means than my neural networks.

Declaration

This thesis has been composed by myself and the work reported within was executed by myself.

January 1994

Acknowledgments

I would like to acknowledge the many contributions of others that made this thesis possible. In particular I would like to thank:

Steve Isard for being my supervisor. He has been an easy-going and fruitful source of guidance and information.

Jim Hurford and Bob Ladd for being my second and third supervisors, and for being ready to listen even when I wandered far from their own interests.

The members of the Linguistics Department in general, for being friendly, interesting and interested despite my residence in exile at CSTR. In particular, thanks to my fellow post-graduates, especially the other inmates of the E-Room.

The members of CSTR, for help, support, friendship and distractions, and CSTR itself for allowing me the resources necessary to carry out my research. Also the computing officers, Andie, Bob, Tim and Alistair for the necessary magic to make things work.

My parents, for continual support of all kinds, and for putting up with my lack of communication and endless inaccurate estimates of when I would finish.

SERC, for funding me for the first three years.

My many friends, Livingston and District Amateur Athletic Club (especially the Green Bogey Bing Bashers), the Uni Hare and Hounds and Interløpers, for making life interesting. Special thanks to Martin and Margaret Hyman for organisation and support beyond belief, and to Colin for always knowing what is going on and for being Minister for Transport.

Finally, special thanks and love to Sue, for making everything worthwhile.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Aims of the Thesis

The purpose of this thesis is to investigate the ability of artificial neural networks (ANNs) to learn to produce the F1, F2 and F3 formant tracks of the vowel in CVC triphones, and to examine the effects of different input and output representations, ANN architectures, ANN training algorithms and training methodologies.

The mapping from a phonemic or phonetic representation of speech to a representation of the acoustic structure of the speech which is suitable for driving a speech synthesiser is a necessary part of any text-to-speech system. It is also the point at which a discrete symbolic representation must be mapped to a continuous non-symbolic representation. ANNs are a useful tool in mappings of this kind. ANNs implement a very general class of models. That is, they make very few assumptions about the nature of the mapping between the input and output data. This contrasts with most other approaches to the problem which assume that the mapping takes a particular form. In essence, the ANNs' implementation of the mapping is based purely on the data used to train the net, and incorporates no phonetic knowledge, except for that used to construct the input and output representations.

I have trained a large number of ANNs to map from an input representation of CVC triphones to an output representation of the F1, F2 and F3 vowel formant tracks. I have used a variety of input and output representations. The ANNs also differed in the training algorithm used (back-propagation or conjugate gradient) and on whether cross-validation was used or not. Most ANNs trained had a hidden layer, but a small number did not. When a hidden layer was used, the number of nodes in the layer was varied. For each combination of the above, a number of different initial states of the ANN weights were used, to increase the chance of finding the best solution.

The questions which I have sought to answer are:

- **Question 1.** Are feed-forward ANNs capable of learning to map from descriptions of CVC triphones to descriptions of the F1, F2 and F3 formant tracks of the vowel? See Section 9.1 for evaluation of this question.
- **Question 2.** What is the effect of different representations of the input phonemes on the ability of the ANNs to learn the mapping? The input representations are described in Section 6.4 and evaluated in Section 9.2.
- **Question 3.** What is the effect of different output representations in representing formant tracks? That is, how good are these representations when extracted from real speech data, not as produced by an ANN? The output representations are described in Section 6.5 and this question is discussed in Section 6.5.4. Section 9.3 answers this question.
- **Question 4.** What is the effect of different representations of the output formant tracks on the ability of the ANNs to learn the mapping? The output representations are discussed in Section 6.5 and evaluated in Section 9.4.
- **Question 5.** What are the effects of different ANN training algorithms, namely a conjugate gradient method and the back-propagation algorithm? This is discussed in Section 6.6.1 and evaluated in Section 9.5.1.
- **Question 6.** How effective is cross-validation in preventing overtraining

of the ANNs and hence increasing generalisation to previously unseen triphones? This is discussed in Section 6.6.2 and evaluated in Section 9.5.2.

- **Question 7.** What are the differences in performance between two and three layer ANNs? The former are less powerful than the latter. This is discussed in Section 6.6.3 and evaluated in Section 9.5.3.

- **Question 8.** What are the effects of the input and output representations on the number of hidden layer nodes needed for best performance? This gives a crude measure of the difficulty of the task. This is discussed in Section 6.6.4 and evaluated in Sections 9.2.3 and 9.4.5.

For each combination of representations, training algorithms, methodologies and size of hidden layer I trained 5 ANNs, each with a different initial state. For the perceptrons, with no hidden nodes, I trained 10 ANNs with different initial states for each combination of representations and methodologies used. In all, I trained 800 different ANNs for these experiments, plus a number more in initial work and other explorations. As each ANN took between 20 minutes and several hours to train, depending on the training algorithm and size of hidden layer, this represents a large amount of computer time.

The need for a large amount of computer time restricted the total number of combinations which could be trained. I have not trained every possible combination. Instead I have trained three main sets. Firstly, I have trained ANNs which vary only in the input representation used and the number of hidden nodes. They all use the same single choice of other experimental variables. Similarly, I have trained ANNs which only vary in the output representation. Finally I have trained a number of combinations which differ from some ANNs in the first two sets only on one dimension, such as training algorithm or the use of two layers instead of three.

The training data was extracted from a database of single word utterances recorded by a single speaker. The CVC triphones used were restricted to those containing stop consonants /p, t, k, b, d, g/ and monophthongs /ɪ, i, ɛ, a, ɑ, ɒ, ɔ, ʊ, u, ɜ, ə, ʌ/, giving a total of 554 triphones in all.

1.2 Thesis Structure

The thesis begins with a review of approaches to speech synthesis. Chapter 2 begins by examining the stages involved in text to speech systems. Spectrum and waveform types of synthesis are discussed, as is articulatory synthesis, synthesis by rule and concatenative synthesis. I give a characterisation of the contrasting natures of the phonemic and formant spaces. The mapping between these two spaces, using ANNs, and the effects of representations within these spaces on learning the mapping, is the main focus of this thesis.

Chapter 3 presents an overview of learning systems in general and ANNs in particular. The idea of a model is outlined, and the choices to be made in defining a model are discussed. The distinction between training and testing sets of data is explained. Generalisation, the ability to perform correctly on a previously unseen test set, is the most important ability of a learning system. It is desirable to prevent overtraining, where high performance on the training data is at the expense of poor generalisation. Cross-validation is advanced as a precaution against overtraining.

After a brief discussion of neural networks in general, feed-forward and simple recurrent artificial neural networks are discussed in more detail. In my work I have used only two and three layer feed-forward ANNs, but some related approaches use recurrent ANNs in which the output or hidden node activations are fed back to extra input nodes (called context nodes).

Chapter 4 is a review of applications of ANNs to language. It begins by reviewing the use of ANNs to model phonological processes and in speech synthesis. The use of ANNs in speech recognition, parsing, syntax, morphology, comprehension, grammar learning and in very simple models of language users is covered in less detail.

Chapter 5 examines a number of differing models of formant tracks. The simplest models are those which represent a formant track as a single value, either the central or most extreme frequency, or an average frequency. These may be of use in classification but are not useful in synthesis, except in conjunction with some method to convert them to a full trajectory that can incorporate transitions with the neighbouring segments, such as the simple step function models [63]. The Holmes, Mattingly and Shearme model [41] is an example of a more elaborate model typical of those used in synthesis by rule systems.

The Broad and Fertig model [12] of vowel formant tracks for CɪC triphones sums a vowel target value and two functions, one for each of the context consonants. Each function is a set of points for the consonant, derived from speech data. Broad and Clermont elaborate this model [11], with a number of models, each more abstracted than the first. The final model is a sum of vowel target values and exponential curves whose parameters are determined by the consonantal context. Imaizuma and Kiritani present a different model [45] based on second-order delay functions.

I end this chapter by discussing the use of formant models in training ANNs. In my work, the ANNs and formant models used are together equivalent to the models discussed previously. They all map from a phonemic or phonetic description to a vowel formant trajectory. These other models can be seen as rival to my work. However, direct comparisons are difficult, and would be best achieved by deriving the models from the speech data I have used and then comparing the accuracy on the training and test sets.

The different input and output representations used are described in detail in Chapter 6. The different training regimes are explained — different ANN training algorithms, the use of cross-validation, the differences between two and three layer ANNs and the variation in hidden layer size. The creation of the data used in training, testing and cross-validation and the composition of the data sets are described. Section 6.8 explains the scheme I have used to name ANN training trials to reflect the particular combinations of input and output representations, training algorithms and methodology and ANN architecture used. I then list the particular combinations used.

I have used a form of modified rhyme test [25, 7] in evaluating the performance of the ANNs, and the adequacy of the formant track representations used. I discuss this and sources of intelligibility error, together with a formant track error which does not require the use of subjects in evaluation, but whose relationship to a perceptual intelligibility error needs to be determined. Chapter 6 concludes with a description of the process used to turn a representation of vowel formant tracks into CVC utterances recorded on cassette tape.

Chapter 7 describes Experiment I, in which I determine a linear relationship between the root-mean-square error on the F1 vowel formant tracks and the intelligibility error as measured by a modified rhyme test presented to subjects. The experiment and derivation of the relationship are described in detail. Confidence bounds on the regression line are also found, making the relationship a useful tool for choosing between ANNs when a perceptual test is not possible. The chapter concludes with a detour to examine the significant differences between the responses of the subjects with English accents and those with Scottish accents.

Chapter 8 describes Experiment II, a second modified rhyme test which was designed to answer the questions set out earlier. The experimental setup is described and the raw results are presented. Chapter 9 is the heart of the thesis. Here the results of Experiment II are used to answer the questions raised at the beginning of this introduction. The performance of ANNs which differ on only the dimension in question are compared, allowing the best input and output representations, ANN architecture, ANN training algorithm, and training methodology to be determined.

Chapter 10 takes a look at some of the formant tracks produced by the best overall ANN, comparing them with the original data. Comparisons are made for the complete tracks of all instances of the triphone /pɔt/ and for the vowel formant track initial, central and final frequencies across all instances and in particular contexts. This chapter illustrates some interesting features of the ANN output compared with the original data.

Chapter 11 concludes the thesis. It discusses the work presented in the previous chapters, the limitations of the work and suggests possible further work.

Appendix A describes a flexible formant synthesiser program used in the experiments. Written in the object-oriented language C++ this synthesiser allows the user to arrange the digital components in any desired configuration and makes adding new components a simple task. The current components implement the equations used in the Klatt formant synthesiser [51].

Appendix B contains some of the materials used in the modified rhyme tests to test the intelligibility of the ANN produced formant tracks and the various formant track representations.

Appendix C contains a subclass of the Machine Readable Phonetic Alphabet (MRPA) [61] which is used in some of the figures in the thesis. The MRPA uses only alphabetic characters, which makes it useful when IPA characters are not available, such as in the data analysis (Splus [10]) and drawing package (xfig) used in the production of this thesis.

Chapter 2

Speech Synthesis

2.1 The Stages Involved in Text to Speech Systems

Text to speech (TTS) systems are composed of a number of stages. The overall aim of such a system is to take a string of words written using ordinary alphabetic (graphemic) characters and to produce the spoken utterance equivalent to the words, just as if a human being had read out the words. The stages typically involved are:

1. *Text to Phonemic Level.* The graphemic representation is mapped to a phonemic representation of the utterance. This holds information on the phonemes making up the words, and probably some prosodic information such as word and sentence stress.

2. *Phonemic Level to Phonetic Level.* The phonemic representation is converted to a phonetic representation. The phonetic level carries information about the actual realisation of the phonemes in the phonemic level. Effects such as coarticulation and reduction must be taken into account. The correct allophones for the context must be found. The prosodic information may be used to create pitch contours and may alter durations of segments.

3. *Synthesis.* The phonetic representation is mapped to a representation to be used for input to the production stage.

4. *Production.* The representation produced by the synthesis stage is used to drive some process which in turn produces the waveform of the utterance.

The different approaches to TTS may not all follow these stages. In articulatory synthesis, the phonemic to phonetic stage may be reduced in complexity as effects such as coarticulation should occur naturally as a result of the articulatory model of the human speech production system used. In the work presented in this thesis, the input for the artificial neural networks (ANNs) is close to the phonemic level of representation. Many of the processes normally included in the phonemic to phonetic stage are carried out by the ANNs, along with the synthesis stage.

2.2 Models of Speech Production

Linggard [64] divides models of speech production into three main types. These are articulatory synthesis (discussed in Section 2.3), spectrum synthesis and time domain synthesis. However, features of the latter two can often be related to articulatory models. For example, formants are an acoustic phenomenon which spectrum synthesis aims to produce, but they are the result of resonances in cavities of the vocal tract, modelled in articulatory synthesis, and the frequency of the formant is the resonant frequency of the matching cavity (determined in turn by the length of the cavity, modified by complex interactions with the cavity wall and articulators).

2.2.1 Spectrum Synthesis

In spectrum methods of synthesis, the aim is to directly model the speech signal itself, working in the frequency domain. An example of this approach is formant synthesis [1, 27, 41, 51]. A source signal (which can incorporate voicing, aspiration and frication) is modified by resonators (producing poles, or peaks in the signal at a given frequency) and antiresonators (producing zeros, or troughs in the signal at a given frequency) to produce a signal containing formants similar to that of natural speech. The resonators and antiresonators have variable frequency and bandwidth. The components can be analogue electronic devices [41], or can be simulated in a software program on a digital computer.

In a serial or cascade synthesiser the resonators are connected in series, which automatically results in correct amplitudes for the formants, as shown by Fant [27]. This arrangement can be regarded as a simple model of the vocal tract, without nasal coupling. Only a small number of resonators (3-5) are necessary to produce acceptable quality speech. This arrangement performs well for non-nasal voiced speech.

A parallel synthesiser, where the resonators are in parallel with each having a gain control setting the amplitude, performs better on stops, nasals and fricatives. The Klatt synthesiser [51] incorporates both serial and parallel pathways, producing a synthesiser capable of production of the full range of speech sounds. The synthesiser I have written for my own use is based on the Klatt synthesiser but has greater flexibility in the configuration of components (see Appendix A). Since I only synthesise vowels I only use the serial or cascade mode of operation.

To control a formant synthesiser it is necessary to provide the frequency and bandwidths for the resonators. In the parallel case it is necessary to also provide the amplitudes. Bandwidths and amplitudes may be set to constant values for given phonemes, in which case all that is needed is to specify the formant trajectories.

2.2.2 Waveform Synthesis

Formant synthesis and other spectrum methods attempt to match the spectrum of a natural utterance. That is, they attempt to match the speech in the frequency domain. Some methods of synthesis attempt to match the waveform of a natural utterance. That is, they operate in the time domain. The best known of these techniques is Linear Predictive Coding (LPC) [47, 48, 67].

In LPC synthesis predictor coefficients are calculated for a linear weighted sum whose input is a series of samples of a signal and whose output is the next sample in the signal. Some minimisation procedure is used to find the coefficients which minimise the mean-squared prediction error. The prediction error is the difference between the actual signal and the predicted signal. New coefficients are calculated periodically. This technique is a very effective way of producing low bit-rate codings of speech. That is, predictor coefficients can be extracted from speech then resynthesised to produce a highly accurate recreation of the original speech. Resynthesis requires an error signal, which may be random noise for unvoiced speech and an impulse train for voiced speech. The predictor coefficients are a much more compact representation of the speech than the speech signal itself.

There are a number of variants on the LPC theme, including the PARCOR method of Itakura and Saiko [48, 49]. Other time-domain codings include Pitch-Synchronous Overlap-Add (PSOLA), which multiplies the speech signal by a moving Hamming window to produce the coding [39]. PSOLA allows manipulation of pitch periods and duration of segments.

2.3 Articulatory Synthesis

Articulatory synthesis is carried out by creating quantitative models of the human speech production process, that is, by simulating the operation of the processes leading up to the acoustic signal. The full set of processes that could be modelled are neuromuscular, articulatory, aerodynamic and acoustic source generation processes [92]. A very early articulatory model was the ingenious mechanical device constructed by von Kempelen in the late eighteenth century [102]. An early digital simulation was developed by Flanagan and colleagues [30].

Given the complexity of the structures involved, simulations are based on greatly simplified models of the articulatory system. Typical simplifications include regarding the vocal tract as a series of linked tubes of either constant cross-sectional area or constant cross-sectional shape. Cross-modes and reflections are often ignored, and the simpler wave equations used. A full model would have to include the effects of smooth and discontinuous changes in pipe cross-sectional area and shape, yielding pipe walls, the effects of turbulence and possibly even changes in the speed of sound, which depends on temperature, humidity, gas density and the diameter of the pipe. A general discussion of models can be found in [64, 92].

Typically, the number of parameters to be controlled is about 7-9, if lip and tongue parameters are included. The values of the parameters change slowly, relative to the granularity of the simulations, but the accuracy of their trajectories is important in producing high quality speech output. Various theories have been advanced about how these trajectories should be constructed. Gesture theories construct the trajectories from a concatenated series of articulatory gestures (ie. segments of trajectories) which must be smoothed together at the transitions [14, 31]).

Alternatively, the trajectories could be the result of a number of conflicting effects. The formant target theory says that when a vowel is produced the articulators attempt to reach their ideal *target* values for that vowel. Their initial positions are determined by the previous segment and they must reach the initial positions for the following segment. They have only a limited time in which to achieve this and are subject to limits on their maximum speed and acceleration due to the physical effect of inertia, the physiological limits on the muscles and the limitations of the neuromuscular control system. This theory therefore sees the trajectories as an attempt to reach goal states under a set of constraints.

The creation of the trajectories of the articulatory parameters corresponds to the synthesis stage of the TTS process, as described in Section 2.1. In a complete system, these trajectories might be calculated as in a synthesis by rule system (based on one of the theories discussed above). It might eventually be possible to build a complete system that begins with a cognitive model of phonological processes, proceeds to a neuromuscular control system and then to an articulatory synthesis system.

At present, articulatory synthesis is a research field, not a route to commercially viable speech synthesis.

2.4 Synthesis By Rule

A number of successful systems have used the synthesis by rule approach to mapping from a phonetic level description to the control parameters for the speech production system. Synthesis by rule systems use a set of rules (possibly supplemented by a look-up table) to perform the mapping. Liberman and colleagues [62] specified a set of rules associated with subphonemic features which were used to produce formant tracks. There were rules for manner, place, voicing and position. Later examples include the work of Holmes, Mattingley and Shearme [41], the MITalk system [1], and the work of Klatt [51, 52]. The speech production module has generally been an acoustic domain method such as a formant synthesiser or an LPC synthesiser.

The formant synthesis method requires formant trajectories to be produced. This can be based on the formant target approach. Each vowel has an associated target value for each formant. This value is the "ideal" frequency for the formant — the value it will reach given sufficient duration. The phonemes before and after the vowel establish start and end points for the formant tracks (loci). The formant track begins at the set start point (or points at it), moves towards the target vowel frequency, possibly not reaching it, then moves to the final frequency.

The rate of change of the formant track frequency has some maximum value, and the initial and final values must be reached. This means that the central target frequency may not be reached. The synthesis by rule program must calculate trajectories for the formant tracks which meet these constraints. The trajectories may be altered by the addition of durations for steady state segments and transition segments within the vowel formant track. The process creating the trajectories must be able to combine the various targets and durations in some manner. Priorities may be associated with the elements to be combined. For instance, the model may require the formant tracks to reach the consonant loci, but only to move towards the vowel target. The various elements may be found by examining real speech, but typically some hand adjustment takes place.

The values for the formant targets and consonant loci may be derived purely from rules based on phonemic features as in Liberman's work [62], or may be found from a look-up table, or may result from a mix of these two approaches as in the Klatt system [51]. The speech produced by all of these methods is intelligible, if machine-like.

2.5 Concatenative Synthesis

Concatenating single phonemes is a poor method of synthesis, due to the inability to cope with coarticulation. Researchers have tried using larger units of speech, such as words, syllables, half-syllables and diphones [13, 26, 66, 73, 74, 75, 89]. The process involves concatenating segments of speech waveform, or some transform of the speech waveform, to produce continuous speech. Linear predictive coding is commonly used in concatenative synthesis. Formant tracks, PSOLA and other codings have also been used.

In diphone synthesis segments of speech are taken between the centres of adjacent phonemes so that the important transition information between phonemes is captured. All possible transitions between all of the allophones in the target language must be included in the database of diphones. In cases where coarticulation effects occur beyond adjacent phonemes it may be advantageous to add segments covering three phonemes. Depending on the type of coding used, techniques may be available to alter the duration and pitch of segments so that natural sounding speech with stress and prosody are produced. PSOLA allows this type of manipulation [39].

2.6 Mapping From the Phoneme Space to the Vowel Formant Space

The lowest level of *discrete symbolic* description of speech is the phonetic description. The levels above this (phonemic, syntactic) are also discrete symbolic descriptions. The next level down is a time series of control parameters for some synthesis method, such as formant, LPC or articulatory synthesis. This level is not symbolic, but a *continuous, non-symbolic* description. Hence in any speech production system, whether it is a TTS system or a human speaker (assuming that humans really do use discrete symbols in cognition), there is some process mapping from a symbolic description to a set of non-symbolic, continuous values.

The process of producing vowel formant tracks given a phonemic or phonetic description of the CVC triphone containing the vowel can be seen in terms of a mapping from one mathematical space (the phoneme space) to another space (the formant space). The task of the researcher is to find this mapping, or an adequate approximation to it, using some set of tools.

In my experiments using ANNs, the inputs to the ANNs are symbolic descriptions of CVC triphones and the outputs are continuous descriptions of vowel formant tracks.

2.6.1 Characteristics of the Phoneme Space

The phoneme space is discrete on most dimensions. Typical dimensions are height, voicing and roundedness. These are all expressed as either binary values or as a small number of values implying an ordering. Most systems of phonemic features have imposed no more structure beyond specifying the existence of a number of features which are present in each segment of a string of concatenated segments. Some approaches, such as *autosegmental phonology* [38], have used more complex structures. Autosegmental phonology employs a series of *tiers*, or ordered sequences of objects. Relationships are established between the tiers by association lines. One aspect of this is that the segmentations on different tiers are no longer necessarily at the same instants in time.

In *generative phonology*, strings of phonemes (with associated features) are rewritten to phonetic descriptions. It is at this stage that processes such as coarticulation have their effect. The rules employed may be complex and somewhat ad hoc and may be applied multiple times. It is this type of mapping which is generally used in TTS systems. Local [65] has called for the use of phonological models which allow a simpler and more coherent mapping to the phonetic level. I have ignored this question entirely by using ANNs to map directly from a broad phonetic level representation to some representation of the vowel formant tracks[1]. Any necessary transformations of the this level of description have been carried out by the "black box" of the ANN, as well as the mapping to the formant track description.

2.6.2 Characteristics of the Formant Space

The formant space is some representation of the actual vowel formant tracks. The thing being represented is a physical object, not a mental object, being composed of sound waves in a medium. Most representations will be comprised of continuous valued numbers. The most "direct" representation used here, after the digitised waveform itself (composed in fact of integers in a large range, but approximating a continuous object), is the vowel formant tracks extracted by a formant tracker. This representation is a series of numbers in a computer of type real that in fact take a large but finite number of values but which are in essence continuous when compared to the small set of values phonemic features take.

A number of more compact representations are used in this work; using a small number of points on the formant tracks, or using the coefficients of some mathematical function (polynomials or Fourier transforms) applied to the tracks.

[1] That is the transcribed phonemes in the description extracted from the corpus of segmented speech which provided the experimental material were those *observed* by the transcriber. These may well not have been the phonemes which would have been present in a citation form transcription of the words contained in the utterance. That is, the observed phonemes may have been the result of some transformations.

It would be possible to represent the vowel formant tracks in a discrete manner however, and it might have been useful to have done so. One possibility would be to divide the duration of the track up into a number of segments, find the mean frequency of a formant track in that segment and divide these frequencies up into a number of bins. The natural choice would be frequency bins one Bark unit apart, as the Bark unit is based on the width of the critical band for frequency resolution in humans, as derived from psychophysical experiments [107]. Another possibility would be to describe formant tracks in terms of the closest fit to a number of standard shapes, although this would risk losing entirely any connection with the continuous nature of the physical formants.

2.6.3 The Mapping

The mapping can be summed up as an informal theorem — there exists some mathematical function which maps the discrete phonemic value of a speech utterance to a continuous acoustic value of that speech utterance. This function should have a number of properties. In particular, it should have some form of continuity, so that phonemes with similar features should produce similar acoustic values. This does of course assume that the features used to describe phonemes have some correspondence to features of speech. Strictly speaking, they correspond to aspects of the mental representation of speech, at least in theory. Phonetic features correspond to physical aspects of speech. These are frequently confused or conflated. It is possible to argue that the whole of the above would be better stated as a discussion of the mapping between the phonetic level of representation and the formant (acoustic) level.

2.7 Learning the Function

To learn the function mapping a CVC triphone to the vowel formant tracks there are two requirements:

1. A class of models. This determines the nature of the function carrying out the mapping. This can be a detailed model motivated by phonetic or physiological knowledge, or a more general model. These include the general linear model (GLIM) [15] which is the linear sum of a series of terms involving the input values, various non-linear extensions to the GLIM, or models involving other combinations of functions. Artificial Neural Networks (ANNs) provide a very general model, being capable of approximating a very wide class of functions [19, 34].

2. A method of determining the parameters of the chosen model. A wide range of methods are available, either in the field of statistics or the related field of learning systems. The method must automatically adjust the parameters in order to reproduce a training data set. Hopefully the final model will then successfully work with new data not included in the training data set.

The various types of synthesis by rule can be seen as implementing particular models of the vowel synthesis function. This is discussed further in Chapter 5.

Chapter 3

Learning Systems and Artificial Neural Networks

3.1 General Issues

The term *learning system* can be applied to a wide range of techniques throughout traditional statistics and less traditional approaches such as artificial neural networks (ANNs) in their various forms and inductive learning systems. Although it generally appears attached to techniques outside mainstream statistics, this is a reflection of a separation in the jargon used rather than a true distinction. What they all have in common is that they can all be seen as automatically adjusting parameters of some more or less general model to *fit* some training data set. The training set may be presented as pairs of input and output values, or as independent and dependent variables. The user of the system may wish to use it to predict output or dependent variables given an input not in the training set. The ability to make predictions (correct or otherwise) given new input values is called *generalisation*.

Learning systems are used in two main ways. The first is *classification*, where the input data is assigned to a limited number of disjoint sets. This can be seen as a *partition* of the input space. An example of this is deciding what the species of a flower is from information such as the length and width of petals and sepals, as in the well known *iris* data set [6, 29]. The second is as a mapping from the input space to the output space. An example is predicting someone's height given their age and weight. One particular type of prediction that has its own set of techniques is the prediction of the next element in a *time-series* given the history of the series.

3.1.1 Models

Any particular instance of using a learning system embodies a particular *model* of the process that generated the training data. Linear regression assumes that the dependent variable is linearly related to the independent variables and hence can be predicted by a linear combination of them. Any transformations applied to variables before applying the regression procedure are also part of the model. The commonly used least-squares method of fitting the regression line is based on assumptions about the nature of the distribution of the samples about the population mean and of the distribution of measuring errors. Even the choice of what variables to use is a part of the model.

In classification problems, the fundamental difference between different methods is in the nature of the partition of the input space. Methods which construct decision trees correspond to dividing the space up with planes parallel to the axes. Other methods correspond to planes in any orientation, or to curved surfaces or to volumes of various shapes.

3.1.2 Training and Testing Sets

The training set is the set of data used in constructing or training the model. While the errors (that is, difference between the output values in the data and those produced by the model) on the training set can give some indication of how well the model matches the underlying, real physical process which generated the data, they give a biased estimate of how close that match is, underestimating the true probable error. For this reason it is necessary to use a test data set, composed of data not used in training, to give a measure of how successfully the model fits the process. The one exception to this is if the training data is exhaustive. That is, if the training data includes all possible cases. In this case the training set error is obviously the true error.

3.1.3 Generalisation and Overfitting

In most cases, we are interested in using the model to predict the output matching a previously unseen input — we are interested in generalisation. We can never guarantee to have produced the best possible model of the process just using the data set, unless it covers all possibilities. All we can do is produce the best model possible in the class of models the learning system is capable of, under the set of assumptions we have made about the process (often implicit and unstated). Of course, if we know exactly how the process being modelled works, then we can build an accurate mathematical model, but then we wouldn't need a learning system.

One example of the problems of generalisation and choosing model parameters is fitting a polynomial curve of order n to a set of n points. If no further restrictions are made, there is an infinite set of curves that can be fitted through the set of points with zero error. However, if we are using the fitted curve to predict what other points "belong" to the set of which the training points are a sample, then we get a different answer for each curve.

A different type of problem occurs when there is some degree of "noise" in the data. This can be due to the probabilistic nature of the process. One example of this is in speech. For the same vowel in the same context, the acoustic structure of the vowel will be different each time it is uttered. The utterances will form a distribution about the average vowel utterance. Another source of noise is measurement error. In the case of a process that produces "noisy" data, a full model would consist of some predictor of the average case and a description of the distribution of points around that average.

If the sample data is taken from a process which produces a distribution of points around some average case, then it may be a mistake to attempt to minimise the error as much as possible. We are not interested in fitting a model to the actual points we have, but in fitting a model to the underlying process, to the average[1] of the distribution. If we fit too closely to the training data we may in fact do worse at generalisation. This type of problem is known as *overfitting*. The general solution for this is to pick the simplest class of models you think can capture the nature of the underlying process and to keep the power of the model as restricted as possible. A number of other techniques are also used to guard against overfitting, including a number based around cross-validation and some that introduce complexity costs. However, all strategies for avoiding overfitting (other than selecting models based on a real understanding of the process being modeled) are necessarily ad hoc and do not guarantee success. These issues are discussed further in [87, 106].

3.1.4 Cross-Validation

Cross-validation is the use of a third data set (along with the training and test sets) to aid in selection of the best model. If we have a number of competing models, we can use a set of data not used for training to measure the performance of each of the models and pick the best. Because it was used as the selection criterion, the performance on this new data set is not an unbiased estimator of the performance of the model. For this reason, we must not use the test set for selection, but use a third set, the cross-validation set.

In learning systems that iteratively improve the model's fit to the data, each iteration can be seen as producing a separate model. Cross-validation can be used to select the iteration at which the model gives the best performance on the previously unseen cross-validation data. It is assumed that this will be the iteration which gives the best performance on all data. The training proceeds

[1] I am being deliberately vague about what I mean by average, as it depends on what assumptions you make about the nature of the distribution and the best way of fitting a model to it, which may also depend on your intended use of the model!

as normal, until either convergence is reached (there is no further improvement), or some other stopping criterion, such as a maximum number of iterations, is reached. On each training iteration, the error on the cross-validation set is determined. The state of the model at the iteration with the best cross-validation error is taken as the output of the training. Normally, the state of the model at the last iteration is taken as the output of the training.

3.1.5 Size of Training Set

We want to produce a model that reflects the operation of the underlying process creating the data — we want to minimise error over all the possible data that the process can generate. For this, the more data we have, the better. If we have all the possible data, we don't need a model for prediction, only a lookup table (although we may be interested in the "why" as well as the "what" and still want to create a simpler model). If we don't have all the data, more data results in a lower estimated prediction error. That is, we expect to get better predictions if we have more training data.

However, generating data is generally not free, and there may be practical restrictions on the amount of data that can be processed while producing the model. We have to come to a compromise. At an absolute minimum, we must have more training examples than number of free parameters in the fitted model. That is, if we have found the values of N parameters controlling the model, we had better have at least $N + 1$ training examples, and hopefully many more. Some learning systems can automatically adjust the number of free parameters depending on the performance on the training data, or a researcher may adjust things by hand.

3.2 Neural Networks

The term *neural networks* is applied to a class of computational devices which consist of a large number of connected simple elements or *nodes*. Some researchers are interested in producing detailed models of biological neural networks. Most work is unconnected to biological neural networks, and I have used the term *artificial neural network* (ANN) to refer to the type of neural network I have been using. ANNs work on numerical data only, not on symbols. They process input vectors and produce output vectors. Some are *static*, producing one output vector for each input vector. Some are *dynamic*, producing a succession of output vectors for a constant or changing input vector.

The connections of an ANN are weighted and the output, or *activation* of the node is the result of some function applied to the sum of the weighted inputs. In some cases the weighted inputs are multiplied not summed. A single node is shown in Figure 3.1.

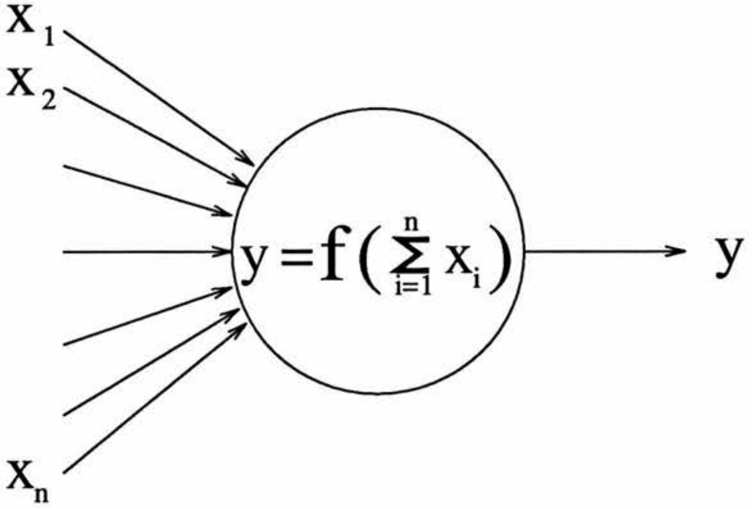

Figure 3.1. A single node in an artificial neural network.

General discussions of the types of ANNs and of the uses to which they have been put are available in [40, 69, 86] and many other books.

3.2.1 Feed-Forward Artificial Neural Networks

The simplest form of ANN has a layer of input nodes and a layer of output nodes, with the weighted links projecting only from the input layer to the output layer. This type of network is called a *perceptron* and has a simple learning rule (the algorithm used to adjust the weights) called the *delta rule*. The perceptron has limited computational capacities. In classification it can only produce linear separations [71]. It is trained by repeatedly presenting each of the input/output training patterns. The activations of the input layer nodes are set to the input pattern. The input layer activations then determine the weighted inputs of the nodes on the output layer. These weighted inputs are acted on by each output node's *activation function* to determine the node's activation. The values of the weights are then altered by the training algorithm (either after each training pattern, or after all patterns have been presented), to reduce the error between the output layer activations and the output training pattern.

Generally, the error measure to be minimised is the least-squares error $E = \sum_p (t^p - y^p)^2$ where E is the error, p is the input/output pair, t^p is the target output for p and y^p is the actual output produced by the ANN for p. Any *activation function* (the function mapping the input to a node to the output of the node) can be used, but the non-linear logistic function shown in Equation 3.1 and plotted in Figure 3.2 is the most common choice. This function keeps values in the range (0,1) and acts as a smoothed threshold (which is useful in classification but not necessary here). If all nodes were linear, then the ANN would only be able to carry out linear mappings, as the sum of linear transformations is a linear transformation. The back-propagation algorithm uses the derivative of the activation function. In the case of the logistic function this derivative is very simple.

$$l(z) = \frac{e^z}{1 + e^z} \tag{3.1}$$

A more complex ANN is the *multi-layer feed-forward* ANN (also called a *multi-layer perceptron*. This has one or more *hidden* layers of nodes between

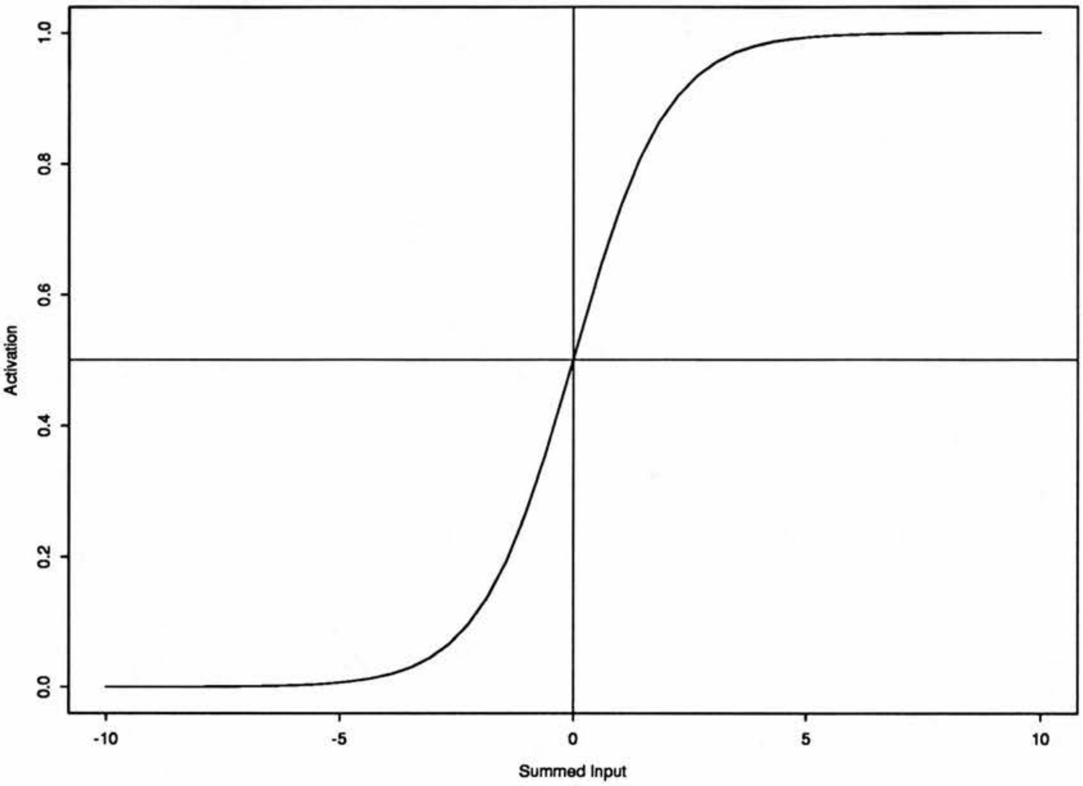

Figure 3.2. The logistic function.

the input and the output layers, as shown in Figure 3.3. The weighted links project only from one layer to the next, never across layers or back to other layers. These types of ANNs can be trained using the *generalised delta rule* [84], or using any of a number of error minimisation techniques such as conjugate-gradient minimisation [77]. As in the case of the perceptron, each input and output pattern is treated separately. Activations propagate forward through the ANN which then reaches its final state.

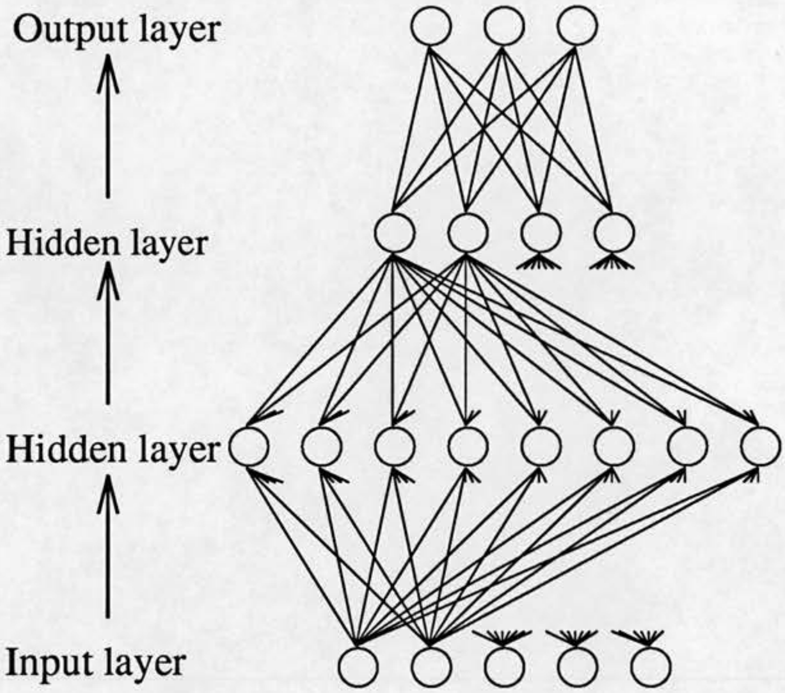

Figure 3.3. The architecture of a multi-layer, feedforward ANN.

Feed-forward, three-layer (ie. one hidden layer) ANNs can learn to successfully classify any set of data, given enough nodes in the hidden layer, as demonstrated by Baum [8]. Baum and Haussler have also investigated the ability of such ANNs to generalise to new examples drawn from the same distribution as the training data [9].

These types of ANNs can be related to methods in classification and approximation. ANNs make no prior assumptions about the process being modelled.

This makes ANNs a good choice for situations when you either have no knowledge of the mapping being used, or when you wish to act as if you have no such knowledge. However, if knowledge about the mapping is available, more mainstream techniques that can incorporate this knowledge should do as well or better than ANNs.

3.2.2 Recurrent Neural Networks

Recurrent ANNs are those whose connections do not just feed forward but which form recurrent connections. This allows the networks to evolve with time — to be *dynamic*. Even with a constant input vector, the output can change at each time-step of the network. This makes recurrent ANNs capable of modelling time-series in a natural fashion, and gives them a memory of previous events. However, training general recurrent ANNs is far more computationally expensive than training feed-forward ANNs. Williams and Zipser [105] present a training algorithm whose storage requirements are $O(n^3)$ on the number of nodes and whose computational requirements are $O(n^4)$ on the number of nodes.

There are a number of simple recurrent ANNs that can use the training algorithms used for feed-forward ANNs. Jordan [50] experimented with networks in which the output unit activations were copied back to an extra set of input units. Elman [22] used a similar architecture in which the hidden layer was copied back to the input layer. In both cases, the activations of the extra *context* nodes depended on both the copied activation and their own previous activations. That is $a(t) = \mu a(t-1) + x(t-1)$ where $a(t)$ is the context node's activation at time t and $x(t)$ is the activation of its associated hidden or output node at time t. This scheme gives the ANN a memory. The degree of influence of the past can be adjusted by changing the value of μ. Figure 3.4 shows the architecture of Jordan and Elman ANNs.

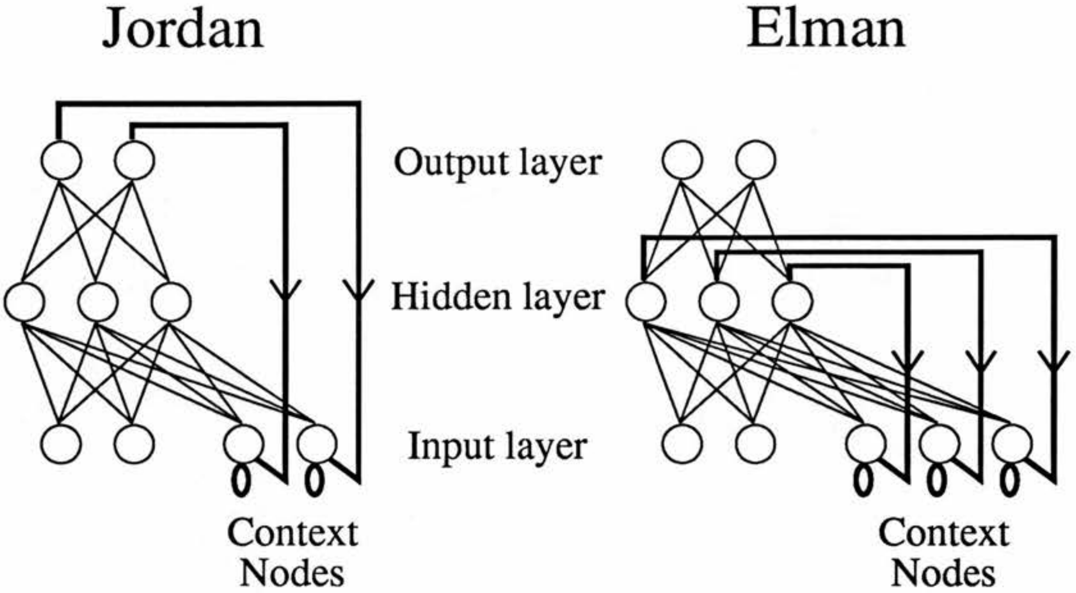

Figure 3.4. The architecture of Jordan and Elman ANNS.

Chapter 4

Artificial Neural Networks and Language

Artificial neural networks (ANNs) have been used in a number of different ways by linguists and other researchers working with language and speech. ANNs have been used in phonology [36, 50, 80, 93, 98], in text-to-speech systems [46, 56, 90, 91, 99, 100, 101], in speech recognition [23, 32, 33, 68, 81, 82, 83] and as parsers [2, 4, 5, 22, 28, 57, 95, 88, 97]. They have also been used in systems modelling language use, such as question-answering and paraphrasing of script-based stories [3, 5, 70, 96].

4.1 Phonological Processes

Traditional generative phonology [16] sees phonological processes as consisting of a set of rules applied in a sequential order to strings of symbols, often cyclically. These are both awkward to apply, relying on ordering to select the next rule, and unconvincing as accounts of psychological processes (although in general linguists claim to not care about this point). A number of researchers have argued for other accounts of these processes. Some of the newer approaches to phonology, collectively known as non-linear phonologies, such as autosegmental phonology, may offer simpler mappings between levels of representation, due to richer structures within those representations. Most connectionist implementations of phonological processes and speech synthesis have mapped in single steps between layers of representation, but have generally stuck to linear representations, as has the author of this thesis in his experiments.

Coleman, Local and others have developed a text-to-speech system (YorkTalk) which uses a non-segmental phonemic representation, with a richer structure than string-based phonologies [17]. There is no rewriting within the phonological level, although constraint processes do operate during the construction of the phonological representation by a phonotactic parser operating on the input text string. A strict distinction is drawn between the phonemic level, composed of a structured symbolic representation, and the phonetic level, composed of numeric parameter values suitable for controlling a speech synthesiser (in this case, a version of the Klatt formant synthesiser). A mapping from the phonemic to phonetic levels is accomplished by a single application of a set of *exponency* rules which set parameter values, followed by a single application of a set of *interpolation* rules. After this it is possible to assign values for all necessary parameters at 5ms intervals. ANNs would be a possible replacement for the single, parallel application of rules between the layers of representation, although ANNs are effectively black boxes and would hence not embody any phonetic knowledge.

In 1988 George Lakoff advanced an outline of a connectionist phonology [60], drawing on his theory of cognitive linguistics as outlined in [59] which some researchers, such as Touretzky [98] and Gasser [36] have taken as inspiration. Lakoff's cognitive phonology uses parallel rules and constraints applying everywhere simultaneously, which corresponds naturally to using ANNs to map between layers of representation.

Touretzky has attempted to implement Lakoff's theories, constructing a series of connectionist systems that map between various levels of representation [98]. He aims to ensure that his mapping architecture can only accomplish the types of transformations observed in phonology, and requires that observed phenomena should be explicable in term of the computational architecture. This differs significantly from much of the ANN work in phonological processes, including most of that discussed below. In most work, ANNs may learn to produce the types of mapping described by linguists, but are not constrained to do so by their architecture. As Pinker and Prince point out [76], they are capable of transformations never seen in language, such as reversing all the phonemes in a word.

NETtalk is probably the best known application of ANNs to language. Sejnowski and Rosenberg [93] trained a feedforward ANN with one hidden layer to map a graphemic representation of a word to its phonemes, represented as phonemic features. The input layer held seven letters in all, and the phoneme for the central letter was produced. Each letter was coded by turning on one of 29 input nodes, each representing a possible letter of the alphabet or a punctuation mark such as a word boundary. The output phoneme was coded in terms of 21 phonemic features (which the authors describe as articulatory features) and 5 other features coding for such things as stress and syllable boundaries.

The NETtalk ANN was trained on two different texts. The first was a transcription of 1024 words of informal speech by a child, which included the elisions and modifications to be expected in this kind of speech. After 50 passes through the corpus, 95% of the phonemes were produced correctly. Stress was assigned almost totally correctly after only 5 passes. Phonemes in a test set of 439 words were produced with 78% accuracy, indicating that the network generalised well to new words. The authors do not say how the results per phoneme translated into results per word. Assuming 5 letter words and equal distribution of errors on phonemes, 78% accuracy on phonemes would translate into 29% accuracy on words. In a similar experiment, Dietterich et al [20, 21] achieved a performance of 80.8% correct per phoneme on a 1000 word test set, with only 13.6% correct per word.

On a corpus of the 1000 most commonly occurring words (as listed in the Brown corpus [55]), the network achieved a phoneme performance of 98%, with a generalisation performance of 77% phoneme accuracy on 20012 test words. With the same assumptions as above, this translates into a word accuracy of 27%.

The learning curves for NETtalk follow the same power laws that characterise human learning. However, the type of ANN used is capable of arbitrary mappings, and therefore drawing parallels with human learning is somewhat suspect. Overall, NETtalk is a good demonstration of the possibilities of ANNs. To be used successfully as a replacement for a conventional text-to-phoneme module of a TTS system its performance would have to be much improved, which might happen with a much larger training corpus. Many TTS systems, such as DECtalk (whose phonemic-to-phonetic and speech synthesis modules NETtalk borrowed to produce actual speech), use a large look-up table of phonemic transcriptions of words with a rule-based module to produce transcriptions of words not in the look-up table. An improved NETtalk might make a replacement for the rule-based module.

Dietterich et al [20, 21] compared the performance of a NETtalk style ANN with the simple decision-tree learning algorithm ID3 [78, 79]. They found that if they used the same procedures as with NETtalk then the ANN performed better than ID3. If the ANN output was thresholded to produce only zeros and ones, as ID3 is constrained to do, the ANN did worse than ID3.

However, if the method of decoding the output (in terms of features) was altered to only consider those phoneme/stress pairs occurring in the training data, then the ID3 performance increased to the same level as that of the ANN. If the decoding only considered sequences of phoneme/stress pairs seen in the training data then performance of both the ANN and ID3 increased, and were equal. This underlines the importance of representations and the output decoding method used in systems containing an ANN or other learning system.

Tuerk et al [101] improved NETtalk performance by not using a distributed output coding, and by correcting an error in coding in the original work, improving phoneme recognition to 99.4% correct on the 1000 most frequent words. On a test set of new words, 73% of words were judged to be pronounced acceptably by a synthesiser when the Dietterich block coding was used.

Reggia et al [80] have constructed a connectionist model of grapheme-to-phoneme mapping based around an "indirectly interactive dual-route hypothesis of reading aloud" that accounts for observed psychological data and is also influenced by neurophysiological evidence. The ANN uses a form of spreading activation. The model is intended as a test (or instantiation) of a psychological hypothesis and is very dissimilar in form and intent from the NETtalk style ANN.

Jordan [50] trained a recurrent ANN to produce a single sequence of phonemic feature vectors. The feature values took values between 0.1 and 0.9, and were left unspecified for some phonemes. In training the target values were set every four network iterations, and the weights updated. Once trained the ANN successfully produced the targets at the specified instants and interpolated in the intermediate steps and for the unspecified values. Jordan interprets the trajectories of the output node values in terms of theories and observations about articulation. However, an ANN that can only produce one output sequence is inadequate as a model of coarticulation. While the output values may follow paths which correspond to coarticulation, it is probably best to take this as an illustration that ANNs may be capable of modelling these processes.

4.2 Speech Synthesis

ANNs have been used surprisingly little in *speech synthesis* — the production of control parameters for a synthesiser from a phonemic representation of an utterance. This mapping is, in general, between two different types of representation — from a discrete symbolic representation to a continuous real-valued representation. It is precisely in this type of mapping that the strengths of ANNs lie.

Tuerk et al [100, 101] have addressed the problem of producing synthesiser parameters from a phonemic representation. Kumar et al [56] have used ANNs to produce formant tracks in a flawed set of experiments. Ishikawa and Nakajima [46] used ANNs to interpolate sampled spectra between CV syllables in concatenative synthesis. Scordilis and Gowdy [90, 91] have used ANNs to produce fundamental frequency (F_0) contours, as has Traber [99].

Tuerk, Monaco and Robinson [101] trained a set of 61 ANNs (one for each allophone) to produce LPC parameters. The ANNs were feed-forward nets with the previous output copied to the input. The rest of the input consisted of a feature-based representation of the adjacent two phonemes, a description of the speaker's dialect, sex, height and age, an indicator of the number of frames to be output and an indicator of the current frame. The training data was taken from the TIMIT database [35]. The authors were interested in the effects of setting different speaker characteristics. Only the sex setting had any effect. However, the use of such settings did allow the researchers to use more training data than might otherwise have been available, as the usual need for single-speaker data can be restricting. The researchers do not give the number of tokens of speech used, or the number of different speakers. The speech generated was poor, but the researchers blame the LPC representation for this. No evaluation scores (either distance metrics on the produced parameters or listening tests) are given, and it is not clear if the ANNs were tested on previously unseen inputs.

In subsequent work, Tuerk and Robinson [100] concentrated on single speaker synthesis of LPC coefficients. They trained 50 ANNs (one for each phoneme) to produce coefficients suitable for use in synthesising continuous speech, using the feed-back mechanism of the Jordan style ANNs to give smooth transitions between adjacent segments. The inputs consisted of the previous output, a feature-based representation of the adjacent two phonemes, an indicator of the number of frames to be output and an indicator of the current frame. The researchers have not conducted listening tests, but are aware of the difficulties of evaluating the quality of the synthesis without them. They do examine the speech in the light of phonetic knowledge and conclude that it behaves as it should. Generalisation to new sequences of phonemes seems to be good.

Overall, this seems to be a successful method of using ANNs in synthesis. By going directly for synthesis involving all phonemes, they have made it easier to acquire sufficient training data. The use of recurrent ANNs removes the problem of choosing a static model of formants and other linguistic features, as each successive time frame is represented in turn.

Kumar et al [56] taught an Elman style recurrent ANN (with extra input units based on previous activations of hidden units) to map from phonemes to formants. A separate ANN was trained for each of the first three formants. The input training data consisted of a total of nine CVC triphones. In one set of tests the triphones were of fixed duration, in a second they were of varying duration, but the researchers do not indicate if there were multiple instances of the same triphone with different durations. The 6 input nodes represented the phonemes /b, d, g, a, ɛ, ɪ/. The appropriate vowel was activated for the duration of the utterance, while the first and then the second consonant were turned on in a pair of square-wave pulses. This results in there being no information about the final consonant until halfway through the vowel, which is too late. The output data was composed of artificial formant tracks created using the duration-dependent exponential model of CVC formants due to Broad and Clermont [11]. The researchers also trained a set of feed-forward ANNS, with the square pulses of the consonant inputs replaced with half-gaussian pulses, in order to impart temporal information and a steady and smooth change in output.

The results give root mean square errors for the Elman net of 47 Hz for F1, 119 Hz for F2 and 47 Hz for F3. These are then expressed as percentages of the formant frequency value (as far as I can tell, it is not stated), which is misleading. A percentage in terms of the possible range of values would give a better indication of the ANN performance in learning the trajectory. Bark scaled figures would give a better comparison between the formants.

The results for previously unseen test triphones give similar results to the training triphones. This seems surprising given the apparent small size of the training set, which I would have expected to have led to overtraining on the training set and a resultant poor generalisation to the test set. However, this may be explained by examining the nature of the model formants used.

All CV and VC transitions were included in the training data. In the model (as derived by Broad and Clermont [11], see Section 5.2.4), the tracks are composed of two consonantal contours, one determined by the initial consonant, the other by the final consonant, added to the vowel target. The contours are exponential curves, and as expected in a model of CVC triphones, the influence of the initial consonant falls off rapidly into the second half of the vowel, and similarly the final consonant has little influence in the first half. Therefore, the test data was almost a reprise of the training data, with the triphone halves joined up in a different order. The final consonant cannot influence the first half of the formant tracks, because the ANN does not know what it will be. The initial consonant can affect the second half of the formant tracks as it helps determine the state of the hidden units (copied back to the input layer) at the centre of the triphone.

As no listening tests were carried out, it is difficult to gauge the performance of the ANNs, and as I have shown above, there was no real test of generalisation. The question of performance depends on the accuracy of the artificial formant tracks, and on the adequacy of the effective division of the vowel into two almost independent halves.

Ishikawa and Nakajima [46] used ANNs to interpolate between the spectrograms of concatenated CV syllables. CV syllables are an effective unit for concatenative synthesis in Japanese, but some interpolation is necessary. They trained a feed-forward ANN to produce a phonetic feature vector when presented with a frame of a spectrogram. At transitions between phonemes, the classification ANN produced feature vectors that lay between those for the two phonemes.

They then trained a set of ANNs (each trained only on a sub-set of consonants) to produce frames of spectrograms, taking as input the phonetic classification output from the classification ANN. For those spectrogram frames from the centre of a phoneme the phonetic feature vector produced by the classifier would be similar to the prototypical vector for that phoneme. For frames taken from the transition between phonemes, the feature vectors produced by the classifier would lie somewhere between the prototypical vectors of the two phonemes.

In synthesis the researchers used the prototypical phonetic feature vectors assigned to each phoneme. To produce transitions between phonemes, the production networks were given inputs linearly interpolating between the phonemes, and they produced output that smoothly interpolated between phonemes, but not necessarily in a linear fashion. The researchers say that this ingenious scheme gave good coarticulation and natural sounding speech, although no formal evaluation is included in the paper.

Scordilis and Gowdy [90, 91] used feed-forward ANNs to generate an F_0 value for a phoneme given the phoneme, 10 previous phonemes and one future phoneme ("macrophonemic" network) , and a Jordan style recurrent ANN to generate the F_0 values frame by frame within the phonemes ("microphonemic" network). The only results given are for training on 20 isolated words. The ANNs get within a few Hz of the contour, suggesting that the small training set and large number of hidden units (20 or 30) has led to overtraining. There are no results for previously unseen test words, and I would expect generalisation to be poor.

Traber [99] makes a far more solid attempt at using ANNs to produce F_0 contours over whole sentences. An Elman style net with activations from the hidden layer (second of two hidden layers in this case) was fed back to the input. The researchers felt that the feedback was necessary to allow declination to be learnt. The input represented a wide window over accent, boundary and phrase information and a narrower window over segmental syllable information. The output was eight values corresponding to four straight lines which represented the F_0 contour within the syllable at the centre of the input window.

The researchers discovered that an ANN trained on a small number of contours learnt them by heart but generalised poorly. Networks with large hidden layers and large windows also generalised poorly even with more training data. They present results for a network which performed well on test data. The larger input window covered 13 syllables in all, and the smaller covered 3 syllables. The total number of nodes was 95, with 2180 weights. There were 186 training sentences, with a total of 6584 syllables. This ANN had a root-mean-squared error of 7.02 Hz on the training data and 9.38 Hz on test data. The researchers actually prefer to use a smaller ANN in their TTS system as it copes better with previously unseen types of pattern not contained in the training corpus, despite doing less well on the test data. They have not carried out formal listening tests but are aware of the need for them and the weaknesses of evaluating by using distance metrics between the target and produced trajectories.

4.3 Speech Recognition

Automatic speech recognition (ASR) consists of a number of stages. The speech signal (usually after some processing to produce a spectrogram or LPC coefficients, for example) is mapped to a phonetic level of description. This is then transformed to a phonemic level description, taking into account effects such as coarticulation, and then the words matching the phoneme string must be found. Generally, the descriptions will be probabilistic, expressing the probability of a slice of speech corresponding to a particular phone or word. ANNs have been used both as modules in complete ASR systems, and in a more cognitive framework as potential models of human performance at specific levels.

McClelland and Elman [68] constructed two connectionist models of speech perception. Both used feedforward ANNs with inhibitory and excitatory connections within layers. TRACE I mapped from a set of input features extracted from real speech, covering 100 time slices, to a phonemic level. The features used are not specified. TRACE II mapped from artificial parameters to phonemes and on to words. The authors claimed to observe various effects matching those in real speech perception, including the ability to cope with coarticulation. In general, they are more interested in providing a model of human performance than in creating a system useful for ASR, and they do not provide tables of the performance of the systems on the training data or on test data.

Elman and Zipser [23] explored the ability of feed-forward ANNs to learn the mapping from spectrograms (divided into frequency bins) to phonemes. In the same paper they showed that if an ANN was taught to reproduce an input waveform spectrum on its output, through a hidden layer with fewer nodes than the input and output (an *encoder net*), then a feature representation was learnt, with many features corresponding to those commonly identified by phoneticians. There was also a segmenting effect.

Other researchers have applied ANNs within an ASR framework. Renals, Rohwer et al explored the use of ANNs to label spectrograms with words [83] and with phonetic labels [81], using a variety of networks such as feed-forward ANNs trained using back-propagation and ANNs based on radial basis functions. They have also looked at the effects of different input representations, such as LPC cepstral coefficients, quantised FFT spectrograms and an auditory model [82]. Franzini et al [32, 33] integrated a feed-forward ANN which assigned phone labels to frames of LPC cepstral coefficients with a Hidden Markov Model based Viterbi recogniser, achieving a 98.5% test word accuracy on a digit recognition task. More recently, ANNs have become common tools in ASR research, with many papers published, and ASR forms the major application of ANNs in language related research.

4.4 Parsing, Syntax, Morphology and Comprehension

ANNs have been applied in a number of different ways at the syntactic and semantic levels of language — as parsers, to form past tenses, to disambiguate referents and to create conceptual representations of sentences.

4.4.1 Learning Grammars

A number of researchers have explored the ability of various types of ANNs to learn grammars. Generally the grammars used are regular expressions and context-free grammars (of relevance to natural language). Formal grammars are characterised by the sets of symbols used, a set of production rules and start symbol. For each grammar there exists a *language* (the set of strings generated by the grammar) and an *automaton* (a machine which recognizes the strings belonging to the grammar). For more information about types of grammars and automata, see [42].

A number of researchers have used simplified recurrent networks of the types suggested by Jordan [50] and by Elman [22]. Both are feed-forward networks with either the output layer (Jordan) or the hidden-unit layer (Elman) connected back to an extra set of input units. The activation of the input units depends on a a weighted sum of the previous output activations, thereby providing a memory beyond the previous iteration.

Elman [22] showed that his recurrent networks could learn to predict the next letter in a sequence of letters, formed into words. The result was that the prediction error was high at the start of a word and decreased within the word. If letters were coded by features, features such as *consonant* had low error rates (the position of consonants being fairly predictable) which other features such as *high* were less predictable. Elman also showed that when sequences of words (represented by random vectors) were presented, ordered into simple sentences such as *monster eat mouse*, then the ANN could learn to predict the next word, though with low accuracy (the next word not being very predictable). However, the internal representation (as found using hierarchical clustering on the activation patterns of the hidden units) showed that the ANN had discovered the grammatical categories that the words fell into, such as nouns, transitive verbs and intransitive verbs. If a new word was added to the sentences fed a trained-up network, the word would be quickly added to the appropriate category.

In the same paper, Elman demonstrates an ANN learning pronominal reference. The inputs are simple sentences made up of words represented by random vectors, corresponding to one of a number of templates with a pronoun present. The output was a vector coding for the structural position of the referent (including outside the input sentence). The network was able to learn the task and generalised fairly well (61% correct) to test sentences including new arrangements of the structures. Allen and Riecken also demonstrated pronoun reference in their experiments with ANNs answering questions about microworlds [5].

Allen used Elman type nets in experiments in which he showed ANNs capable of both recognising and generating strings drawn from simple context-free grammars and regular expressions [2, 4]. Giles et al used similar nets to learn regular grammars and then extracted the equivalent Deterministic Finite Automaton (DFA) from the trained network [37]. Earlier they had used ANNs with second-order (multiplicative) connections and an external, continuous valued stack to learn context-free grammars [97].

Servan-Schreiber et al [95] explored the conditions under which an Elman style ANN can carry information about long-distance dependencies across intervening elements to distant elements when trained to predict the next element in strings produced by a finite-state grammar. They showed that the embeddings (the intermediate steps) must have some dependency on the information to be maintained, otherwise it was liable to be lost. However, the information was maintained even if only subtle statistical properties of the embedded strings depended on the early information, allowing the ANN to deal with the types of dependencies found in natural language.

Kwasny took another approach in using an ANN as an alternative to rules in a "Wait and See" Parser operating with a stack which stored structures [57]. The aim was to create a deterministic parser that would learn its rules and handle ill-formed inputs.

Another approach, completely ignoring the ability of ANNs to learn and to operate in a subsymbolic fashion, is to create an algorithm which will convert a grammar into an ANN. Fanty's algorithm [28] converts a context-free grammar into a structure comprised of a large number of units, operating in parallel, that represent terminals and nonterminals in the grammar. The final pattern of activation represents the parse tree. Schnelle and Doust take a similar approach to create network forms of an Earley chart-parser [88].

ANNs have been used for morphology, in particular to learn the past tenses of English verbs. Rumelhart and McClelland [85] trained a feed-forward ANN to produce the past tense of the input word. The input word was coded in terms of its phonemes. This coding was converted by a hard-wired layer of the ANN into what the authors named a *Wickelfeature* representation as it was based on an idea originally suggested by Wickelgren [104]. Wickelfeatures form a distributed representation based on phonemic features taken from three successive phonemes (including word boundaries). The input Wickelfeature coding was converted to the output Wickelfeature coding by the learning portion of the ANN, a linear pattern associator (since there were no hidden layers) and decoded to a phonemic representation by another set of hard-wired weights. The input values were present tenses of verbs and the output was trained to be the corresponding past tense.

The training took part in three stages. Initially the 10 most frequent verbs were presented for 10 training epochs, then the 410 medium frequency verbs were added and another 190 training epochs run. Finally the 86 low frequency verbs were added, but not trained over. The high frequency verbs were learned accurately at first, with no difference between the regular and irregular verbs. The addition of the medium frequency verbs led to an initial drop in the performance on the high frequency irregular verbs but not on the high frequency regular verbs. The irregular verbs were treated as if they were regular verbs. After more training the accuracy increased on the irregular verbs increased again. It is claimed that the pattern observed is similar to that found in children acquiring the same skill. However, Pinker and Prince [76] have criticised on a number of grounds the notion that this ANN represents a model of human language acquisition. Mozer [72] has investigated the same learning problem using recurrent ANNs with the input presented sequentially.

4.5 Language Users

Allen has explored the abilities of recurrent ANNs to answer questions about microworlds [3, 5]. The ANNs used were modified versions of the Elman type network. The microworlds consisted of objects with attributes. The inputs to the ANNs consisted of a representation of a microworld and a coding representing a question, such as *What color is the car?* The output represented the response, such as *blue*. The ANNs also showed the ability to cope with pronouns. If the network was asked *Is the apple on the left?* and then asked *Is it red?*, it would answer correctly.

St. John and McClelland [96] taught recurrent ANNs to accept a sentence as input and to produce a conceptual representation of the event the sentence describes, with any unspecified slots filled on the basis of expectations learnt by the ANN. Miikulainen and Dyer [70] took this further by applying multiple recurrent ANNs to the task of paraphrasing script-based stories. The input is a representation of a story, presented sequentially. This is generally only part of the full script about, for example, eating in a restaurant. A series of four ANNs processes this to finally produce an output sequence corresponding to the full script for the particular story that was input, with the slots for *customer*, for example, filled in correctly. The intermediate representations include conceptual representations as in the St. John and McClelland paper.

Chapter 5

Formant Models

I am interested in investigating the ability of ANNs to map from phonemic descriptions of CVC triphones to the associated F1, F2 and F3 vowel formants. For this it is necessary to have some representation or model of the vowel formant tracks. The representation extracted from the speech data is a series of frequency values at successive time frames, a description consisting of anything up to 60 numbers per formant track, or 180 numbers per vowel, with the length depending on the duration of the vowel and the frame-shift. This is not a useful representation for use with a static ANN, although it would be usable with a recurrent ANN evolving through time, such as a Jordan or Elman style ANN.

It is necessary to reduce the formants to a compact representation which captures the phonetically relevant information, allowing the reproduction of recognisable and natural sounding vowels. The representation should also be learnable by a ANN, and should be robust. That is, small errors in the representation of a formant track should lead to only small changes in the vowel reproduced from it. I am restricting the discussion to the formant frequencies only. This ignores the formant bandwidths and intensities, which also form part of a full model of vowel formants.

A model of a formant track comprises two parts. Firstly, a set of parameters which in some way capture the important information underlying the segments involved. Secondly, some way of combining these values to produce a full formant track. The model includes some assumptions about what is important about vowel formants.

Vowel formants are affected by their context. In my experiments this context is provided by adjacent consonants, and many other studies also use adjacent phonemes only, although it is known that phonemes more distant from the vowel can also have an effect. Broad and Fertig [12] showed that the effects of an adjacent consonant extended across the whole vowel. The general effect of adjacent consonants is to alter the initial and final positions of a formant track (some theories assign a *locus* to each consonant — a characteristic frequency from which the formant emerges, or which the formant approaches) and to move the central portion of the vowel formant away from the characteristic frequency found in isolated instances of the vowel, in the direction of the consonantal loci, a phenomenon referred to as *undershoot*. The general effect in speech is a *reduction* towards the central schwa vowel. The duration of the vowel, affected by speaking rate and intonation, also affects the degree of reduction. Any adequate model which produces vowel formants from phonemic or phonetic specifications must produce formant tracks incorporating these phenomena.

5.1 Single Point Models

Much of the work on representing vowel formant tracks has been from the point of view of classification of the vowel, not production of the vowel. These efforts have often used a single value to represent each formant track. Typically the representation of a formant track is either a) the frequency at the centre of the vowel (that is, after half the vowel duration), b) the maximum or minimum frequency (extremum) for the formant (which may be at different times for different formants in the same vowel), or c) the mean value of the formant track, averaged over the duration of the vowel (or some subset of the vowel's duration), as illustrated in Figure 5.1. Different representations may be used for different formants within a single vowel. The mean value of the formant track is generally used only for F1. Huang discusses these representations in [43].

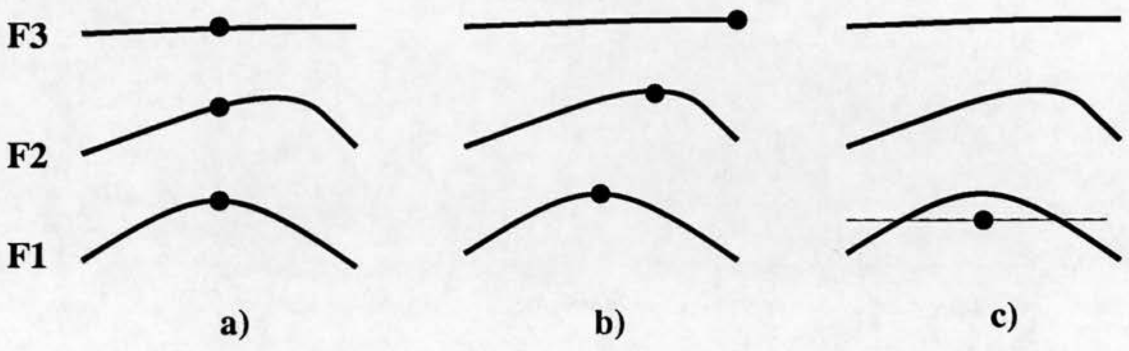

Figure 5.1. Single point models of vowel formants. a) Centre frequencies. b) Extrema. c) Mean value (F1 only).

These representations may be adequate for classification (or they may need further information), but they are not adequate in themselves for producing vowel formants where the entire track must be constructed. The points identified in this manner may form part of a method for constructing a formant track.

5.2 Vowel Trajectory Models

For production of vowels it is necessary to produce full formant tracks spanning the duration of the vowel. Usually, varying the first three formant tracks F1, F2 and F3 is seen as sufficient to produce natural sounding vowels, with any higher formants being kept constant across all vowels. The synthesis methods discussed below can be seen as consisting of a set of parameters and a method of constructing a full formant track from those parameters. A full formant track is taken to be a specification that allows the determination of formant frequencies for each time frame used to drive a synthesiser.

5.2.1 Step Function Models

These are the simplest and crudest form of production model. Each vowel and consonant is assigned a set of parameters which are the frequencies for the formant tracks within that phoneme. These match the consonantal formant loci and vowel formant targets. Formant tracks are constructed from step functions which are then smoothed. That is, a formant track is set to its associated value for the duration of a segment (this may be modified in some variants of the model), giving discontinuous changes at the boundaries of segments. These tracks are then smoothed. An example of this kind of model is given in [63]. Figure 5.2 shows the construction of a formant track using this model. The major disadvantage of this model is that there is no mechanism to incorporate vowel target undershoot.

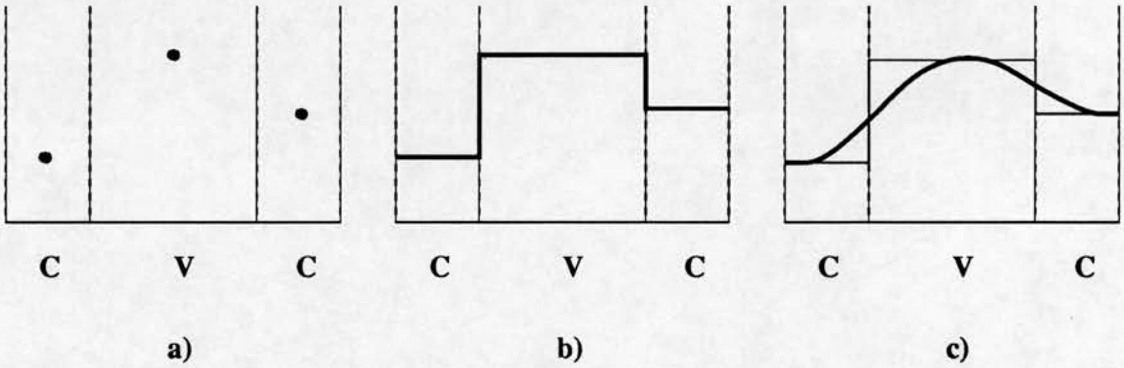

Figure 5.2. A very simple step model of a single formant track. a) Parameters for each segment. b) The step function set by the parameters. c) The smoothed step model.

5.2.2 The Holmes, Mattingly and Shearme Model

The model of vowel formants used by Holmes et al [41] uses a larger set of parameters and a more complex combination algorithm. "Phonetic elements" (the elements making up the string of symbols which is transformed to synthesiser parameters) have up to 25 associated parameters, of which 13 play a role in determining vowel formant trajectories:

- *Rank.* The relative rank of two adjacent elements determines which set of parameters determine the trajectory of the transition between the elements. Stop consonants rank high, vowels low and others between these extremes.

- *Standard duration.*

- *Unstressed duration.* Vowels only

- *F1, F2 and F3 frequencies.*

- *Fixed contribution for F1, F2 and F3.*

- *Proportion of steady state added to fixed contribution for F1, F2 and F3.* A proportion of the steady state frequency of the adjacent element is added to the fixed contribution to determine the formant frequency at the boundary between the elements.

- *External transition durations for F1, F2 and F3.* The duration of the transition in the adjacent element.

- *Internal transition durations for F1, F2 and F3.* The duration of the transition in the controlling element.

If two adjacent segments have equal ranking, then the first has priority. The frequency at the segment boundary is calculated as the fixed contribution plus the steady state frequency of the adjacent element multiplied by the given proportion. The transition is then linearly interpolated from the boundary to reach the steady states at the limits of the transition durations. If a segment contains two transitions whose summed durations do not fill the entire duration of the segment, then the remaining duration is at the steady state frequency for the segment. Otherwise, the transitions are terminated at the point at which they meet. Figures 5.3 and 5.4 illustrate these processes.

The Klatt [52] and MITalk [1] systems and many other synthesis by rule systems use similar models. In some cases some parameter values are calculated using rules triggered by phonetic features instead of being obtained from a look-up table. In the MITalk system, Klatt uses a locus model in which the formant

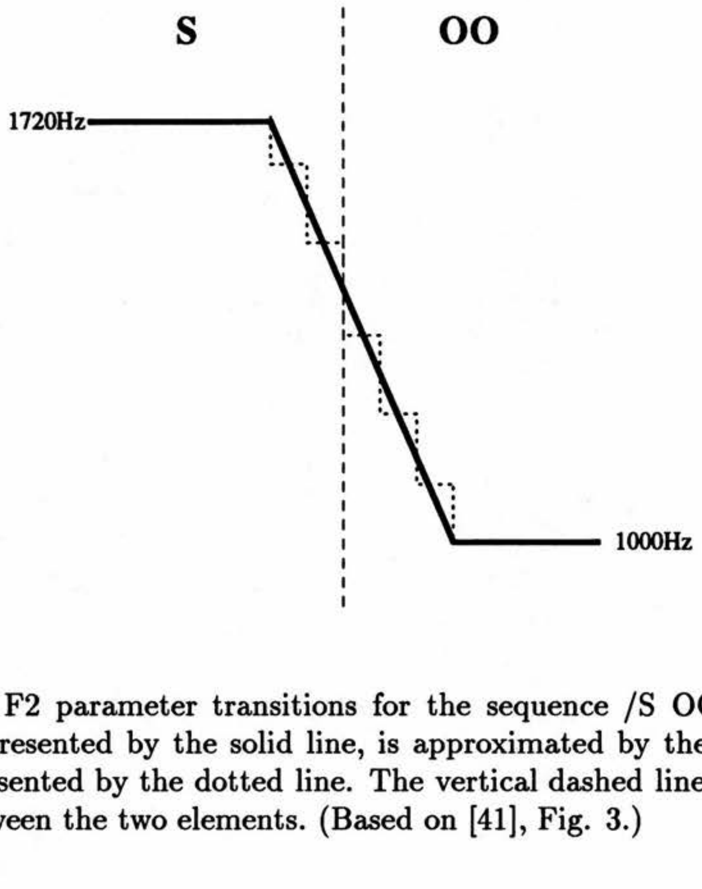

Figure 5.3. F2 parameter transitions for the sequence /S OO/. The ideal transition, represented by the solid line, is approximated by the series of time samples, represented by the dotted line. The vertical dashed line represents the boundary between the two elements. (Based on [41], Fig. 3.)

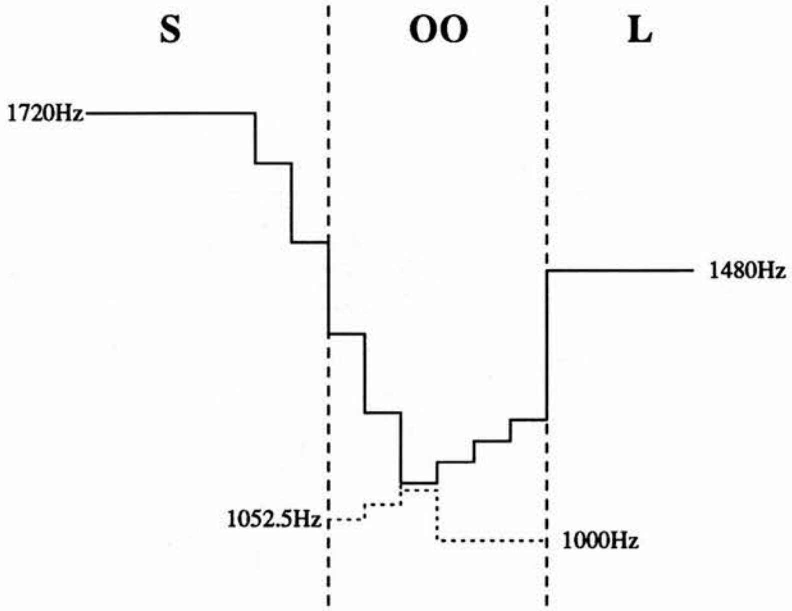

Figure 5.4. F2 parameter transitions for the sequence /S OO L/ showing the intersection of the transitions for /S OO/ and for /OO L/. the solid line represents the transitions actually used; the dotted lines represent values calculated but later discarded. The vertical dashed lines represent boundaries between adjacent elements. ([41], Fig. 4.)

transitions into the vowel are determined in part by consonantal loci. The York-Talk system [17] uses Klatt's locus model, but the parameters are calculated using rules mapping directly from a richly structured phonological representation.

5.2.3 The Broad and Fertig Model

Broad and Fertig [12] analysed the influences of the initial and final consonants on the vowel formant tracks in CVC triphones containing the vowel /ɪ/. They found that, while the nonlinear interaction between the consonants was statistically significant, its contribution to the formant tracks was low and it was feasible to ignore it in constructing a model of the influence of the consonants on the formant tracks. They proposed the model

$$f_{i,j}^{(n)}(t) = m^{(n)}(t) + \sigma_i^{(n)}(t) + \tau_j^{(n)}(t), \qquad t = 1, 2, \ldots, 11 \tag{5.1}$$

where $f_{i,j}^{(n)}(t)$ is the expected value of the nth formant frequency at time t when the initial consonant is C_i and the final consonant is C_j, $m^{(n)}(t)$ is the mean over all utterances of the nth formant at time t, $\sigma_i^{(n)}(t)$ is an additive initial consonant influence on the nth formant determined by C_i and $\tau_j^{(n)}(t)$ is a final consonant influence determined by C_j.

As the mean for a formant track at a particular instant, $m^{(n)}(t)$, does not seem very informative they recast the equation as

$$f_{i,j}^{(n)}(t) = V^{(n)} + \sigma_i'^{(n)}(t) + \tau_j'^{(n)}(t), \tag{5.2}$$

where $V^{(n)}$ is a constant for each n and can be interpreted as vowel target values.

From a speech database of 1728 CVC triphones consisting of three repetitions of 24 × 24 CVC triphones (the 24 consonants included silence), the researchers traced the vowel formant tracks and derived the values of the formant targets and the consonantal influences (I am ignoring the detail of this derivation). The final model consists of the vowel targets and curves for each initial and final consonantal context. The vowel formant track for the consonantal context is derived by adding the formant target, the initial consonantal curve and the final consonantal curve. This process is illustrated in Figure 5.5.

Each curve is in fact composed of values found at each of 11 time points throughout the vowel. The vowels are normalised in time, so duration effects are ignored. The model cannot incorporate changes in vowel formant trajectory due to different rates of speech. A statistical analysis showed that the errors in predicting instances of the formants were consistent with the natural variation in formant tracks for the same triphones, suggesting that the model was a good fit to the data.

5.2.4 The Broad and Clermont Models

Broad and Clermont [11] built on the earlier work by Broad and Fertig [12] to produce a series of increasingly abstracted models of vowel formant tracks in CVC' context (they used C' to indicate the final consonant). They used a smaller set of consonants (/b,d,g/) with a larger set of ten vowels. The contexts used were 30 VC and 30 CVd sequences, recorded three times each.

Model I: Additivity of CV and VC' Transitions

Model I was built on the assumption of the additivity of CV and VC transitions, as used in the Broad and Fertig model. They restated the Broad and Fertig model as

$$F_{CVC'}(n) = f_{CV}(n) + T_V + g_{VC'}(n), \tag{5.3}$$

where F is the vowel formant track in context CVC', n is the discrete time in frames, f and g are the initial and final consonant transition functions and T is the vowel target for vowel V. They calculated the values of the elements of equation 5.3 using the following (where a dot "." indicates averaging over a subscript or argument) :

$$T_V = F_{V..}(0), \tag{5.4}$$

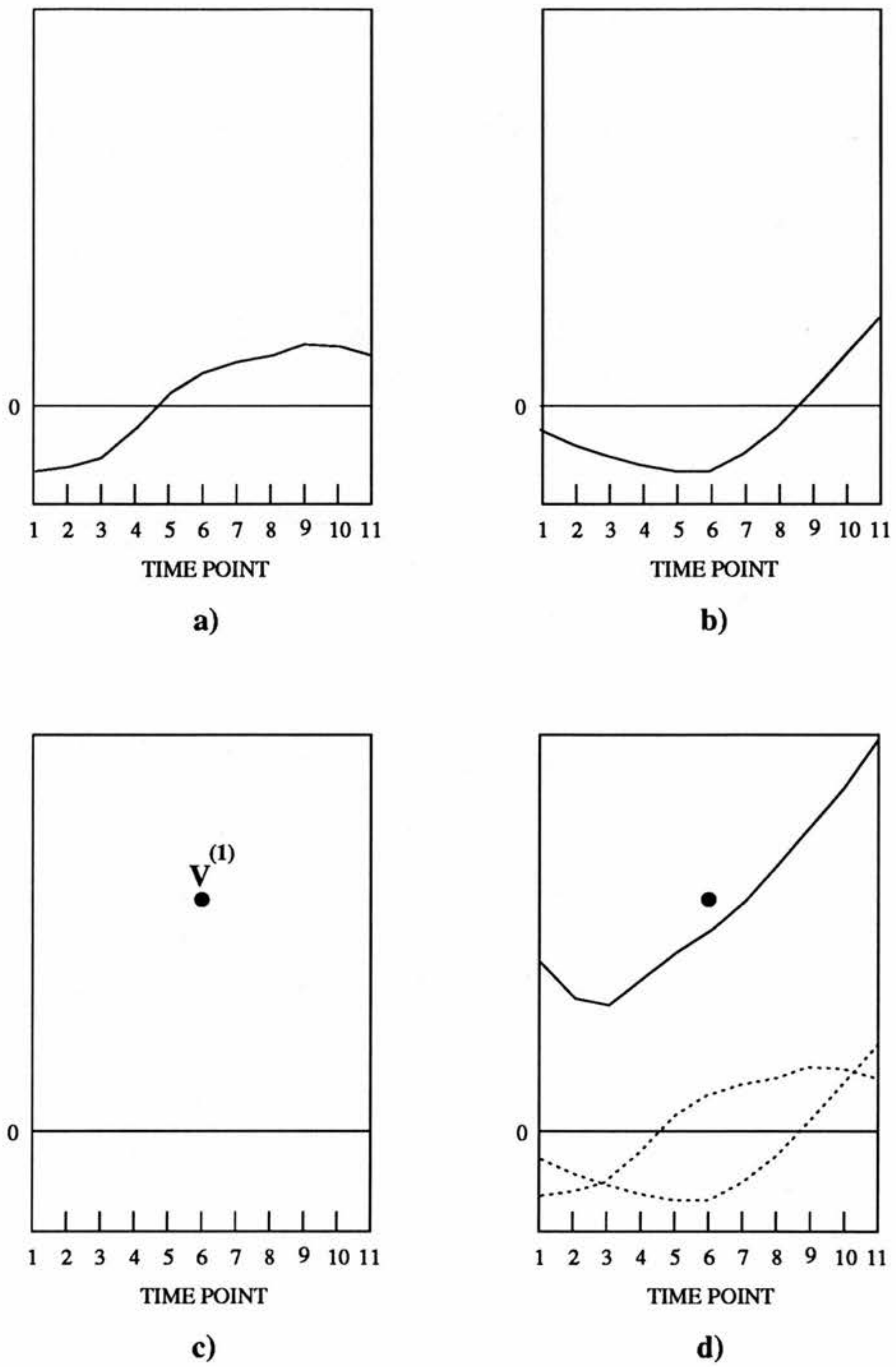

Figure 5.5. The Broad and Fertig vowel formant model. a) Curve for first initial consonant. b) Curve for final consonant. c) $V^{(1)}$ for vowel. d) Summation of consonant curves and $V^{(1)}$.

$$\begin{aligned} g_{VC'}(n) &= F_{VC'.}(n) - T_V, && (5.5) \\ f_{CV}(n) &= F_{CVd.}(n) - T_V - g_{Vd}(n). && (5.6) \end{aligned}$$

averaging over instances where appropriate. In Equation 5.5 the vowel formant targets are taken to be the mean values of the onset of the formant tracks in the VC' utterances. This model is very similar to that of Broad and Fertig, but applied to a number of vowels. The consonantal curves are still composed of a sequence of values, one for each time slice, and the vowels are all normalised to have the same number of time slices.

Model II: Per-Consonant Similarity

The next assumption incorporated was that initial consonant formant contours had the same underlying shape over all the vowels, and similarly for final consonant formant contours. Model I had a separate contour for each consonant-vowel combination. This new assumption can be expressed as

$$F_{VC'.}(n) = L'_{C'} + k'_{VC'} g^*_{C'}(n), \qquad (5.7)$$

for the final consonant contours. $F_{VC'.}$ is the average VC' contour, L' is the consonant locus, $g^*_{C'}$ is the common contour shape for final consonant C' and k' is a scale factor. By a complex process, the researchers derived the values of L', g^* and k' for each final consonant, and similarly for L, f^* and k for each initial consonant. The transition functions now become

$$\begin{aligned} f_{CV}(n) &= L_C - T_V + k_{CV} f^*_C(n), && (5.8) \\ g_{VC'} &= L'_{C'} - T_V + k'_{VC'} g^*_{C'}(n). && (5.9) \end{aligned}$$

Model III: Target-Locus Scaling

In Model III, the assumption is that scale factors k_{CV} and $k'_{VC'}$ found for Model II are proportional to the distance between the vowel formant targets T_V and the consonant loci L_C and $L'_{C'}$. This can be expressed as

$$\begin{aligned} k_{CV} &= \mu_C(T_V - L_C), && (5.10)\\ k'_{VC'} &= \mu'_{C'}(T_V - L'_{C'}). && (5.11) \end{aligned}$$

where μ_C and $\mu'_{C'}$ are consonant dependent constants which the researchers derived from the data. The transition functions now become

$$\begin{aligned} f_{CV}(n) &= (T_C - L_V)[\mu_C f^*_C(n) - 1], && (5.12)\\ g_{VC'} &= (T_V - L'_{C'})[\mu'_{C'} g^*_{C'}(n) - 1]. && (5.13) \end{aligned}$$

Model IVa: Duration-Independent Exponentiality

The researchers say that the consonant contours look suggestively exponential, although the example they give doesn't really convince me over other alternatives. However, exponentials have the useful property of tending to an asymptote, which is important to the model. They also assume that the vowel targets should match the asymptotes of the exponential curves used as consonantal contours. The derivation is described as "cumbersome" by the researchers. In the process they derive new values for the formant targets T_V and the consonant loci $L'_{C'}$ as well as the scale factors $k'_{C'}$ and $b'_{C'}$, and presumably for L_C, k_C and b_C, although these are not discussed. The transition functions now become

$$\begin{aligned} f_{CV}(n) &= k_C(T_C - L_V)\exp(b_C n), && (5.14)\\ g_{VC'} &= k'_{C'}(T_V - L'_{C'})\exp(b'_{C'} n). && (5.15) \end{aligned}$$

Model IVb: Duration-Dependent Exponentiality

The time scales used in the previous models were all normalised for vowel duration, that is, all vowels were divided into the same number of time frames (11). The researchers construct a model of type IVa to unnormalised formant tracks.

They then suggest that formant tracks of differing lengths can be obtained by using segments of the exponential curves truncated at durations matching the vowel duration, measuring from the consonant end of the curve. This is illustrated in Figure 5.6.

The models are all evaluated thoroughly as to how well they explain the training data. They provide good fits to the data (taking into account the natural variability of speech data), with the accuracy not decreasing too much as the models become more abstracted. However, the researchers do not examine if the models provide a good fit to vowels in CVC triphones not used to create the models.

5.2.5 The Imaizumi and Kiritani Model

Imaizumi and Kiritani [45] propose a model of CVC formant transitions, where the CVC triphone is embedded in a VCVCV utterance. They use delay functions to represent the vowel-to-vowel, consonant-to-vowel and vowel-to-consonant influences, combined in an additive fashion as for Broad and Clermont. They are particularly interested in incorporating the ability to represent different speech rates into their model. The trajectory of the nth formant in a vowel segment $F_n(t)$ is expressed as

$$F_n(t) = U_n(t) - C_{np}(t) - C_{nf}(t) \qquad (5.16)$$

where $U_n(t)$ is the step response of a second order delay function (described below) which represents the vowel-to-vowel effects and $C_{np}(t)$ and $C_{nf}(t)$ are first order delay functions which represent the effects of preceding and following consonants. $R_{i,j}$ are taken as fixed vowel target frequencies of each vowel in the sequence $V_1C_pV_2C_fV_3$. The index i selects the vowel. The index j represents the formants. For back vowels, the numbering of formants goes 1, 2, 3 from F1 to F3, but for front vowels it goes 1, 3, 2 to account for continuity in formants.

The functions $U_n(t)$ are composed of the simpler second order delay functions

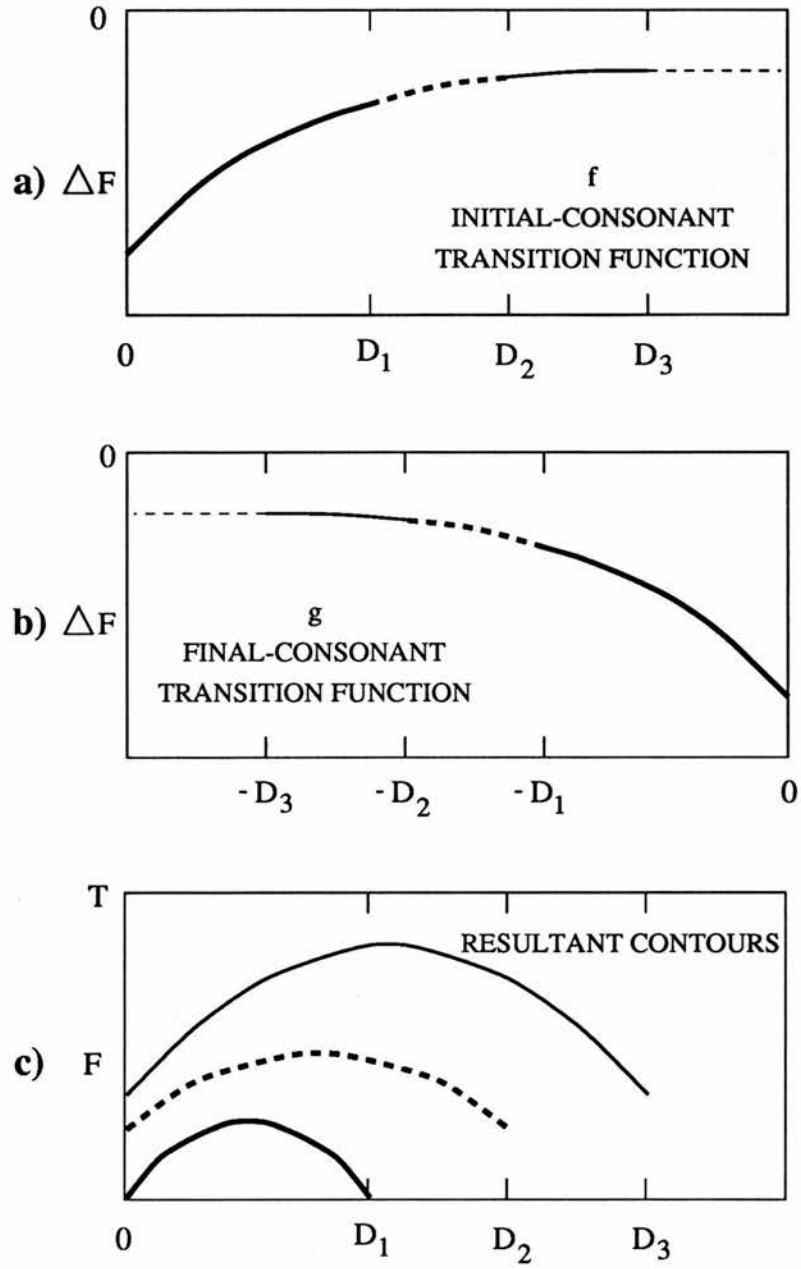

Figure 5.6. Modeling CVC' contours on a real-time scale. (a) Transition function $f_{CV}(t)$ is defined forward from $t = 0$. (b) Transition function $g_{VC'}(t')$ is defined backwards from time $t' = 0$. We truncate f and g to durations D_1, D_2 and D_3. (c) Superposition of the f-g pairs from (a) and (b) with vowel target T. (Based on [11], Fig. 10)

$W_j(t)$, expressed as

$$\begin{aligned}
W_j(t) &= R_{1,j} + a_i(t)(R_{i,j} - R_{i-1,j}), & (5.17)\\
a_i(t) &= 1 - 1 + b_j(t)\exp(-b_j(t))u(t - t_i), & (5.18)\\
b_j(t) &= (t - t_i)/g_j, & (5.19)\\
u(t - t_i) &= \begin{cases} 1 & \text{if } t > t_i \\ 0 & \text{if } t < t_i \end{cases} & (5.20)
\end{aligned}$$

where g_j is a time constant representing transition speed.

These functions are combined to produce the vowel-to-vowel functions in such a way as to account for coupling between resonance frequencies (for more explanation, see [45])

$$\begin{aligned}
U_1 &= W1, & (5.21)\\
U_2 &= h\sqrt{W_2 W_3}, & (5.22)\\
U_3 &= \sqrt{W_2 W_3}/h, & (5.23)\\
q &= (W_2^2 + W_3^2)/W_2 W_3, & (5.24)\\
h &= q - \sqrt{q^2 - 4(1 - k^2)}/(2(1 - k^2)), \qquad k = 0.2. & (5.25)
\end{aligned}$$

The functions representing the effects of the adjacent consonants are defined as follows.

$$\begin{aligned}
C_{np,i}(t) &= c_{np,i}\exp(-(t - t_{p,i})/g_p), \qquad \text{for } t_{p,i} < t < t_{f,i}, & (5.26)\\
C_{nf,i}(t) &= c_{nf,i}\exp(-(t_{f,i} - t)/g_f), \qquad \text{for } t_{p,i} < t < t_{f,i}. & (5.27)
\end{aligned}$$

where t_{pi} is the initial time of vowel V_i, $t_{f,i}$ is the final time of V_i and g_p and g_f are time constants representing the decay speed.

The researchers then proceeded to derive the necessary values from a set of specially recorded speech and explored the effects of changes in speech rate on the intelligibility of the recorded speech and of speech produced via the model described above. They found that at a slow speaking rate vowel intelligibility was

100% and consonant intelligibility was 83%. At a fast speaking rate (twice that of the slow rate), vowel intelligibility was 83% and consonant intelligibility was 63%. For the slow rate, both vowel and consonants synthesised using the model were more intelligible than those synthesised using the formants extracted from the training data. At the fast rate, the vowels constructed using the model were more intelligible than those constructed using the original formants.

5.3 ANNs and Formant Models

The models of vowels used in my experiments differ from those described above in that those above are intended to produce vowel formant tracks for the given contexts with no other mechanism adjusting parameters for them. That is, the models above embody a full method of production from strings of phonetic or phonemic segments to full formant tracks. The parameters, once found, are kept constant.

In my work an ANN creates the parameters controlling the model, standing between it and the input string. The ANN plus the formant model is equivalent to the full models described above, and any comparisons should be made on that basis.

The models used in my work are simple compact descriptions of the formant tracks. In some ways they are more like the single point descriptions used for classification, discussed in Section 5.1, although they do allow the recreation of formant tracks. However, the more complex models used in production may suggest useful representations for my work. Conversely, ANNs may have a possible role as parts of modified versions of the above models. For instance, an ANN could adjust the scale factors of the fixed shapes of consonantal effect contours in the Broad and Clermont models, possibly taking into account wider context than the adjacent vowels.

Chapter 6

Experimental Methods

6.1 Aims of the Experiments

The aim of my experiments is to select a good form of model for mapping from a broad phonetic description of CVC triphones to vowel formant tracks. An ANN is to be used for the mapping, but choices must be made between different possible input and output representations, training methods, ANN architectures and ANN training algorithms. These choices may be independent or may interact. The questions to be answered are listed in Chapter 1. This chapter concerns itself with explaining the representations, training algorithms and methodology. The questions themselves are answered in Chapter 9.

6.2 Representations

In order to train the neural networks it is necessary to represent the input data (phonemes and durations) and the output data (vowel formants) in some fashion which is appropriate for the network. The input data must be coded as a vector of real numbers and the output data must be coded as a vector of real numbers in the range [0, 1] [1].

The form of the representations will have a bearing on how easily (in terms of number of hidden nodes required and number of learning iterations) and how well (in terms of accuracy on both the training and test sets) the network learns the required mapping. A good representation should encode useful information about the domain in a way which aids the learning process. Similar items of data should have similar representations, and distinguishing information should be easily computed. However, it is one of the strengths of neural networks that they can cope with less than optimal representations, which is why they are often used in situations where the underlying process is poorly understood.

The expectation is that input representations which make explicit the phonetic information that determines the shape of the formants will give better performance than other representations. Similarly, an output representation which both represents what is important about the shape of the vowel formants, and is easy to learn should result in good performance.

It should be noted that performance should be measured by how intelligible the resultant speech is, not by how close the produced output values are to the target values, although the two should be related. However, the ANN is trained to produce output values as close as possible to the given target values. I have attempted to find a function that predicts performance on a perceptual test from the root-mean-squared error between original and ANN produced vowel formant tracks (see Chapter 7).

[1] In fact, due to the difficulty in pushing the values of the network output to 0 or 1 (a very large magnitude of input activations into the output nodes being required), outputs are usually coded into the range [0.1, 0.9], as I have done.

6.3 Question 1: Can ANNs Learn The Mapping?

The first question (see Chapter 1) was "Are feed-forward ANNs capable of learning to map from descriptions of CVC triphones to descriptions of the F1, F2 and F3 formant tracks of the vowel?". This question underlies all of the other questions. If an ANN using any of the combinations of input and output representations, training algorithms and methodologies discussed in this chapter produces reasonably intelligible vowels, then the answer to this question is yes.

6.4 Question 2: Input Representations

The input divides into two parts — the phonemic data and the duration data. The phonemic data or representation of the input triphone is comprised of entities which take on a limited number of values, while the duration data is intrinsically real-valued.

6.4.1 Representing the Input Phonemes

The input triphone consists of an initial stop consonant, a vowel and a final stop consonant. The input vectors are constructed by concatenating the representations for the three phonemes. The two consonants use the same coding scheme, the vowel a separate coding scheme. The representations I have used give each input node either a binary value (0 or 1) or a value taken from a small range of values. I have therefore kept each set of node values between 0 and 1, although this is not necessary (unlike the case of the output values).

I have called the three kinds of representations used *Traditional*, *Continuous* and *Symbolic* (see below). The first two are feature representations; the first codes the consonants using binary values only, the second codes place of articulation using one node taking a range of values; and the third is a simple one-of-n coding that does not incorporate any linguistic knowledge.

We would expect the *Traditional* and *Continuous* coding schemes to result in better performance then the *Symbolic* scheme, as they are based on our knowledge of phonetic processes, and so, in theory, require less processing on the part of the ANN than the less rich *Symbolic* scheme. The *Symbolic* scheme is as simple as it can be made.

6.4.2 The *Traditional* Input Representation

The *Traditional* input representation is a representation based on that of Ladefoged [58]. Place of articulation of the consonants has been coded using a node for each position, ie. a node each for *labial*, *alveolar*, and *velar* features. Since I have only used stop consonants, the only other necessary feature is *voicing*, a binary feature. (See Table 6.1).

The *backness* and *height* features of the vowels have been coded using one node each, with them having a range of possible values. *Roundness* was a binary feature. (See Table 6.2).

The multi-valued features contain a notion of ordering and a progression along a continuous dimension that may be useful to the ANN in trying to produce the output representation, especially if the continuous progression along the set of feature values is matched by a progression along a set of matching "features" of the output formants.

6.4.3 The *Continuous* Input Representation

The *Continuous* input representation coded features as positions on a continuum wherever possible. It used the same vowel representation as the *Traditional* representation (see Table 6.2), and represented the consonants using the two features *place* and *voicing* (see Table 6.3).

phoneme	*features*			
	labial	*alveolar*	*velar*	*voicing*
p	1.0	0.0	0.0	0.0
t	0.0	1.0	0.0	0.0
k	0.0	0.0	1.0	0.0
b	1.0	0.0	0.0	1.0
d	0.0	1.0	0.0	1.0
g	0.0	0.0	1.0	1.0

Table 6.1. The coding of consonants in the *Traditional* input representation.

phoneme	*features*		
	back	*height*	*round*
ɪ	0.0	0.67	0.0
i	0.0	1.00	0.0
ɛ	0.0	0.33	0.0
a	0.0	0.00	0.0
ɑ	1.0	0.00	0.0
ɒ	1.0	0.00	1.0
ɔ	1.0	0.33	1.0
ʊ	1.0	0.67	1.0
u	1.0	1.00	1.0
ɜ	0.5	0.50	0.0
ə	0.5	0.50	0.0
ʌ	0.5	0.33	0.0

Table 6.2. The coding of vowels in the *Traditional* and *Continuous* input representations.

phoneme	*features*	
	place	*voicing*
p	0.0	0.0
t	0.5	0.0
k	1.0	0.0
b	0.0	1.0
d	0.5	1.0
g	1.0	1.0

Table 6.3. The coding of consonants in the *Continuous* input representation.

6.4.4 The *Symbolic* Input Representation

The *Symbolic* input representation represented the phonemes using a one-of-n coding, ie. it used as many nodes as there were possible phonemes and represented each by turning on one node and turning the rest off. This obviously contains no useful phonetic knowledge of any kind. See Tables 6.4 and 6.5.

phoneme	*features*					
	p	t	k	b	d	g
p	1.0	0.0	0.0	0.0	0.0	0.0
t	0.0	1.0	0.0	0.0	0.0	0.0
k	0.0	0.0	1.0	0.0	0.0	0.0
b	0.0	0.0	0.0	1.0	0.0	0.0
d	0.0	0.0	0.0	0.0	1.0	0.0
g	0.0	0.0	0.0	0.0	0.0	1.0

Table 6.4. The coding of consonants in the *Symbolic* input representation.

6.4.5 Representing the Duration Information

Some information on the duration of the phonemes making up the triphone was required in the input representation. A triphone was represented as the total duration, the relative start time of the vowel (taking the start of the triphone as time 0) and the relative end time of the vowel (see Figure 6.1).

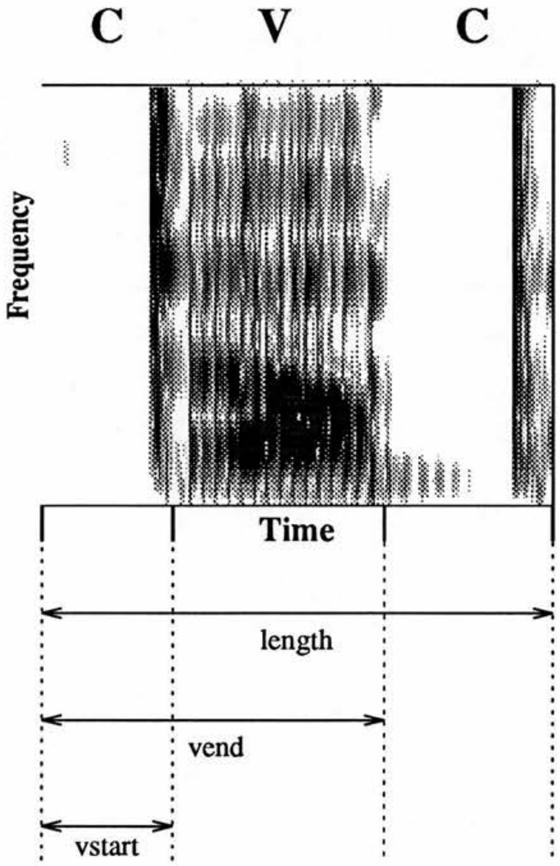

Figure 6.1. The durational information about each triphone was coded by the triple **(length, vstart, vend)**, with each number transformed as in Equation 6.1 ([0, 1] mapped to [0.1, 0.9]).

phoneme	*features*											
	ɪ	i	ɛ	a	ɑ	ɒ	ɔ	ʊ	u	ɜ	ə	ʌ
ɪ	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
i	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ɛ	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
a	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ɑ	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ɒ	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
ɔ	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
ʊ	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
u	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
ɜ	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
ə	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
ʌ	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

Table 6.5. The coding of vowels in the *Symbolic* input representation.

All the triphones were of duration of less than one second, so there was no need to squash the durations onto the range [0, 1]. However, in some previous experiments I had used time information in the outputs, not the inputs, and for this reason I had the times squashed onto the range [0.1, 0.9] by the transformation in Equation 6.1.

$$F(time) = (0.9 - 0.1)time + 0.1 \tag{6.1}$$

This transformation was not necessary when using the durational information as an input, but since I already had the data coded in this way, and it should create no extra work for the ANN, I used it for these experiments.

6.4.6 Representing the Stress Information

Each phoneme representation (see section 6.4.1) had added to it a node representing the stress placed on that phoneme. Stress had been assigned to phonemes by the transcribers of the speech data used for these experiments, and this was used with no modifications. The criteria are discussed in [61]. The nodes took values as follows:

Primary Stress The value 1.0

Secondary Stress The value 0.5

No Stress The value 0.0

6.4.7 The Complete Input Representation

The complete input representation consisted of a vector of numbers in the range [0, 1] composed of the three phoneme representations, plus their stress representations, and the duration representation (see Figure 6.2).

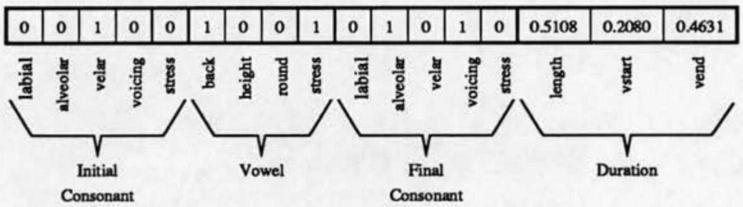

Figure 6.2. The complete input representation for the triphone /k'ɑd/.

6.5 Questions 3 and 4: Output Representations

The output of the ANNs was some representation of the first three formants of the vowel. Three main types of representation were used: a) the frequencies at the start, centre and end of the vowel, b) a set of polynomial coefficients representing a curve fitted to the vowel formants, and c) a set of Fourier coefficients fitted to the vowel formants. The first type (*Tri* representation) was further divided into representations where the second and third formant frequencies were either directly represented or coded as ratios or differences from the first formant.

All formant frequencies were Bark scaled before any representation was extracted. Bark scaling [107] maps the measured frequencies to a scale which more accurately reflects human perception. The scaling is composed of two linear portions between 0 and 500Hz and between 500Hz and 1220Hz and logarithmic above this frequency. The definition (taken from [94]) are shown in Equation 6.2 and plotted in Figure 6.3. This has the desirable property of reducing the penalty for a given error in the formant tracks at higher frequencies, compared with those at lower frequencies. One Bark corresponds to the critical bandwidth. Two formants that are within the critical bandwidth of each other should, in theory, be perceived identically.

$$B(f) = \begin{cases} 0.01f & 0 \leq f < 500 \\ 0.007f + 1.5 & 500 \leq f < 1220 \\ 6 \ln f - 32.6 & 1220 \leq f \end{cases} \tag{6.2}$$

6.5.1 The *Tri* Output Representations

This set of representations mapped each vowel formant track to a triple of frequency values - those at the beginning, centre and end of the track.

Correcting the Formant Tracks

The formant tracker used displayed some inaccuracies at the boundaries of the vowel segment. This was due to sharp changes in formant frequency being smoothed by the tracker algorithm, and hence producing a steep slope at the boundary instead of a discontinuity. Where this slope occurred it could be detected by its gradient which was greater than that normally found in formant tracks. Where the gradient was above a set threshold (50Hz difference in a 5ms time) the data extraction code moved inwards along the track until it found a flatter portion, or it moved beyond a set distance (10ms) from the segment boundary. The value at this point was taken as the true boundary value. Inspection of problem cases showed that this method produced good results.

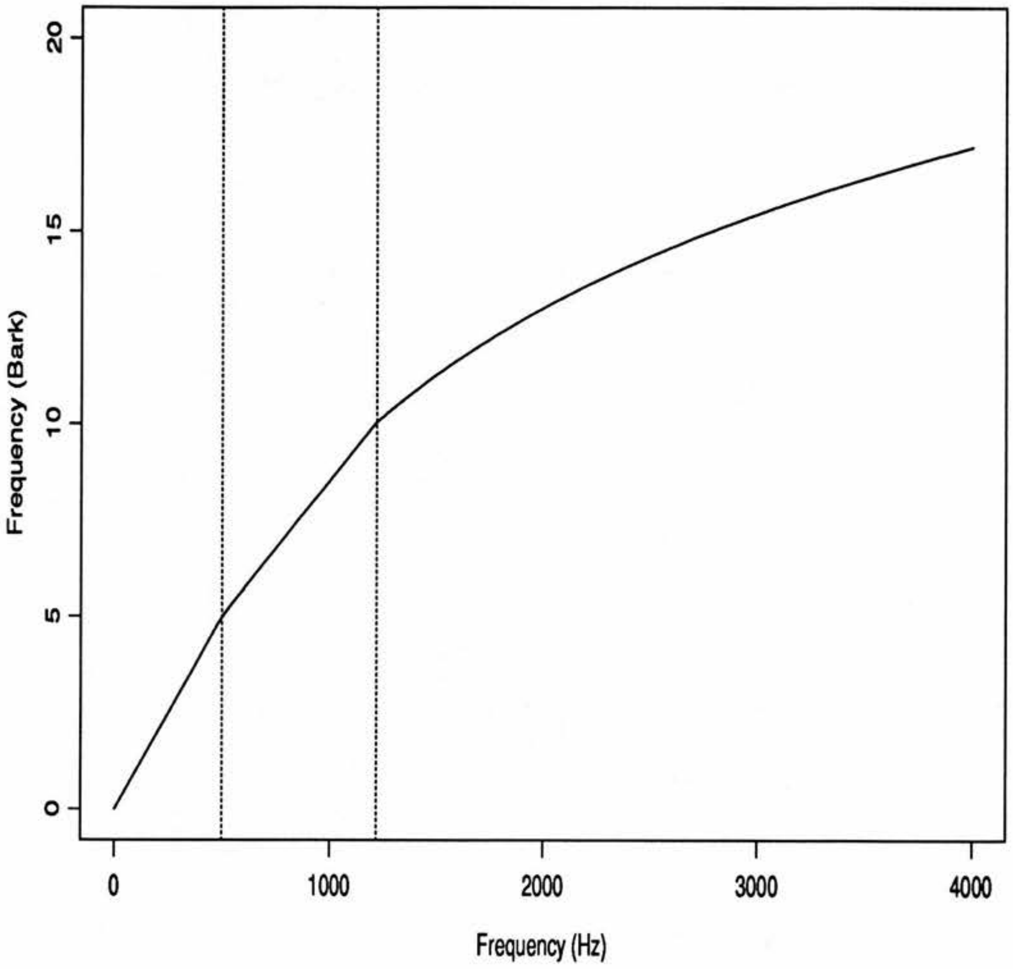

Figure 6.3. The Bark scale.

Coding the First Formant (F1) values in the *Tri* Representations

All of the *Tri* representations coded the F1 values in the same manner. Each triple of F1 values (ie, at the start, centre and end of the vowel segment) was mapped onto the range [0.1, 0.9] by the transformation in Equation 6.3.

$$F(freq) = \frac{(freq - lowfreq)(0.9 - 0.1)}{highfreq - lowfreq} + 0.1 \qquad (6.3)$$

Here $F(freq)$ is the value to be used in training the neural network, corresponding to the F1 Bark frequency freq, *lowfreq* is the bottom limit of the frequency range (set to 2.0 Bark, about 200Hz) and *highfreq* is the top limit of the frequency range (set to 9.9 Bark, about 1200Hz).

The frequency range limits were chosen to ensure that the output numbers filled as much as possible the range [0.1, 0.9], to use the full range of output values available to the neural network and hence use its full power. In theory the network nodes have an output range (0.0, 1.0) but this requires inputs going to $(+\infty, -\infty)$, causing problems if 0.0 or 1.0 are specified as target values.

The *Tri-ratio* Output Representation

In this sub-type of the *Tri* representation the F2 and F3 values were mapped onto the range [0.1, 0.9] by a transformation that expressed them as a ratio to the F1 value (Equation 6.4).

$$F(freq) = \frac{F1freq}{freq} \qquad (6.4)$$

Here $F(freq)$ is the value to be used in training the neural network, *F1freq* is the F1 frequency value, and *freq* is the F2 or F3 frequency value. This transformation produces values in the required range, so no further processing is necessary.

The resultant output values consisted of vectors composed of nine real-valued numbers — the F1, F2 and F3 values for the start, centre and end times of the vowel segment, after the above transforms.

The *Tri-difference* Output Representation

In this representation, the second and third formants are represented as differences from the first formant, which are then scaled onto the interval [0.1, 0.9] (Equation 6.5).

$$F(freq) = \frac{((freq - F1freq) - lowdiff)(0.9 - 0.1)}{highdiff - lowdiff} + 0.1 \qquad (6.5)$$

Here *lowdiff* and *highdiff* are the highest and lowest values for the difference between the first formant and the second or third formant, for each position (start, centre and end). This ensures that the resultant values cover as much of the available range as possible. Table 6.6 shows the values used to code F1 (using Equation 6.3), F2 and F3 (using Equation 6.5).

	F1		F2 – F1		F3 – F1	
	high	low	high	low	high	low
start	7.52	3.18	10.37	2.14	11.65	6.77
centre	11.56	2.85	10.65	2.00	12.00	3.15
end	10.35	2.54	11.07	6.33	12.12	4.30

Table 6.6. The highest and lowest values of F1 and of the differences (F2 – F1) and (F3 – F1), in Bark.

The *Tri-plain* Output Representation

In this representation, the F2 and F3 values are not represented relative to the F1 values, but are merely scaled onto the interval [0.1, 0.9], exactly as for F1. For the second formant, the upper and lower frequency limits are 15.02 Bark and 5.7 Bark, and for the third formant the values are 16.70 Bark and 13.58 Bark. The same limits were used for the initial, central and final points, unlike the method used for the *Tri-diff* output representation.

6.5.2 The *Polynomial* Output Representations

An alternative to the *Tri* input representation was to use a set of polynomial coefficients to represent the formant tracks. Each of the first three vowel formant tracks was represented by the coefficients of a second order polynomial (hence, three values per formant) fitted to the vowel formant tracks. In some trials these tracks were Bark-scaled before the polynomials were found.

The time axis of each track was mapped onto the interval [-1, 1]. This achieved the following:

1. Time 0 was at the centre of each track. This has the effect of making the zeroth coefficient (constant) the frequency of the centre of the track. The order one coefficient (x) measures the slope of the track left to right and the order two coefficient (x^2) measures the quadratic curve of the track, with the maximum or minimum point (after subtracting the slope measured by the order one coefficient) at the centre of the track. This seems to be a reasonable way of describing the characteristics of a formant track.
2. All tracks were mapped to the same time interval. Hence tracks of similar shape, but over different durations, would have a similar polynomial coefficient representation. This may simplify the task of the ANNs learning to produce vowel formant tracks using this representation.

The coefficients were found by general least squares, using the singular value decomposition method [77]. This method ensures that there are none of the problems caused by small or zero pivot values in the standard methods. These can result in very large values for coefficients which cancel each other's effects out in fitting the curve but which would be better replaced by small or zero values. These large values would result in a loss of the continuity of representation required for the neural network to produce good generalisation. However, experimentation showed that the standard methods would have sufficed, producing the same coefficients as the singular value decomposition method. This is to be expected in this case with simple curves and low order polynomials.

The coefficients obtained had to be scaled to the range [0.1, 0.9] to be used in the target output representation of the ANNs. The maximum and minimum values for each of the nine coefficients (three formant tracks each represented by three values) were found and mapped to the extremes of the range, with the other values linearly mapped between them, in a similar way to the *Tri* representation (See Equation 6.3). This still allows the production of coefficient values outside this range with previously unseen triphones, as the ANN can produce output values outside [0.1, 0.9], given large enough activations. This is unlikely to happen except in extreme cases, which is a desirable trait.

I experimented with various orders of polynomials. I evaluated the representations by producing the coefficients for all the formant tracks I had available, reproducing tracks from the coefficients and then measuring the root-mean-squared error between the original and new tracks. Orders of less than two produced poor tracks. Those of order greater than two added little extra accuracy. As discussed above, polynomials of order two seem to be a good representation of vowel formant tracks.

6.5.3 The *Fourier* Coefficient Output Representations

The final type of output representation used was the coefficients of the discrete Fourier transform of the first three vowel formant tracks. This transforms the curve into the frequency domain — representing the curve as a sum of sinusoidal components.

The sinusoid associated with a given coefficient is a function of the interval between the samples in the time domain. In the time domain, the samples are real-valued. In the frequency domain, the discrete Fourier coefficients (samples) are complex-valued, including amplitude and phase information. The zeroth coefficient has a zero complex part. This coefficient represents the steady-state part of the curve. That is, it is the average of the points making up the curve — the baseline from which the sinusoidal components diverge. I used the *realft* function given in [77] to calculate the Fourier coefficients needed.

There were three different representations used :

1. The original vowel formant track is described by 4 discrete Fourier coefficients. This gives 7 real values which describe each vowel formant track (the zeroth coefficient has no imaginary part).
2. The original vowel formant track is Bark scaled before being described by 4 discrete Fourier coefficients. This gives 7 real values which describe each vowel formant track.
3. The original vowel formant track is Bark scaled before being described by 2 discrete Fourier coefficients. This gives 3 real values which describe each vowel formant track.

The resulting values were then scaled to be in the range [0.1, 0.9] (see Section 6.5.2 for explanation).

6.5.4 Question 3: Evaluating the Output Representations

It is desirable to compare the limitations of each output representation. That is, if the representation was produced perfectly, how good would the produced speech be? This can be tested by producing speech from the target representations based on the formant tracks extracted from the original speech data. It is inevitable that some information will have been lost. I wish to know whether this affects the quality of the speech produced. If a representation produces poor speech, then we cannot expect a neural network to magically produce good speech using that representation. However, if the neural networks fail to produce good speech using a representation that is capable of it, then we know there that for some reason the representation is unsuitable for training neural networks, and can explore further.

6.6 Types of Training Regime

A number of combinations of input and output representation, network architecture (two or three layer), training algorithm and training methodology (cross-validation or training to completion) were chosen to allow the comparisons discussed in Section 6.1 to be made. For each combination a number of ANNs were trained. For those combinations using three-layer ANNs, the number of nodes in the hidden layer was varied, with the numbers 1, 2, 4, 6, 8, 10, 12, 14, 16, 18 and 20 being used. For each size of hidden layer, five ANNs were trained, each with different initial states, giving a total of 55 ANNs for each combination. For the combinations using two-layer ANNs, there were no possible variations in network size, as the number of input and output nodes are fixed by the representations used. A total of ten ANNs with different initial states were trained for each of these combinations.

6.6.1 Question 5: Training Methods

Conjugate Gradient Training

Most of the ANNS were trained using a conjugate gradient method. This optimisation technique makes use of more information about the error surface than the back-propagation method. It proceeds by a series of line-searches — looking for the minimum on a particular line on the error surface. The error surface is composed of the error values for each possible combination of parameters (the weights, for ANNs). Details of algorithms for conjugate gradient searches are given in [77]. Researchers have found that conjugate gradient methods usually increase the speed of training ANNs dramatically [54].

In my training trials using conjugate gradient methods, a pair of ANN simulation programs written by Richard Rohwer were used. The program *bp3strict* was used to train three layer ANNs, both with and without cross-validation. The more general program *bp* was used to train two-layer ANNs, but did not implement cross-validation. The training was set to cease after 300 line searches or

700 gradient evaluations, or when the error *converged* (failed to improve).

Back-Propagation

A small number of ANNs were trained using the *back-propagation* algorithm. Back-propagation is an algorithm in which errors on the output nodes of an ANN are propagated backwards through the network and are used in calculating changes to the weights. It uses only the local error gradient in choosing how to update the weights and hence is a simple steepest-descent optimisation algorithm. The amount of change at each update is controlled by a parameter called the *learning rate*.

Back-propagation is usually used with a momentum term which adds some of the previous update to the current update. This can speed up convergence to a minimum in areas where the error gradient is small and fairly constant (the error surface is "flat"), but it can cause overshoot of an actual minimum and so extend training times. While optimisation methods are usually applied after evaluating the error for all the training data (called *batch* updates in the ANN field), back-propagation is often used after each presentation of a training pattern (called *on-line* updates in the ANN field). This does not use the overall error rate, just error rates on single patterns and may not converge in some cases, but in general results in a speed-up in training. Momentum helps avoid problems due to patterns giving contradictory information about the error. Details of the back-propagation algorithm, and a derivation, can be found in [84]. After some experimentation, I used a learning rate of 0.005 and a momentum term of 0.5 in my training trials, with on-line updates. Training ceased either at convergence or after 1000 iterations (determined by experimentation).

6.6.2 Question 6: Cross-Validation Training and Training to Completion

Most of my ANN training trials used cross-validation (as described in Section 3.1.4). As well as the training set of triphones, the error for a cross-validation triphone set was found on each iteration of the training procedure. The weight matrix corresponding to the minimum cross-validation error was stored. The training proceeded until the stopping criterion was reached. The ANN weight matrix corresponding to the minimum cross-validation error was taken as the output of the training process. This is intended to prevent overtraining and hence increase the generalisation of the ANN to previously unseen test data, at the expense of an increase in the training set error. However, due to the paucity of the data available, the cross-validation set was much smaller than it ideally should have been.

A smaller number of training trials took the state of the ANN weight matrix when training ceased as the output of the training process. I refer to this as "training to completion", although strictly speaking, both methodologies are trained to completion.

6.6.3 Question 7: Differences Between Two and Three Layer ANNs

Three layer feed-forward ANNs (ie. those with a single hidden layer) are significantly more powerful than those with no hidden layer. Three layer ANNs can approximate, to any required degree of accuracy, any continuous function uniformly [19, 34] whereas two layer ANNs can only form linear combinations of the activation function used. The mapping from a representation of CVC triphones to a representation of vowel formant tracks may be too complex for a two layer ANN to learn (this was my expectation), but might be learnt by a three layer ANN.

6.6.4 Question 8: Hidden Layer Size

The number of hidden nodes in an multi-layer ANN is seen as a measure of the power of the ANN. That is, the more hidden nodes an ANN has, the more accurately it can learn a given training set. Too few hidden nodes will result in an ANN unable to learn the training set to any degree of accuracy and poor performance on previously unseen inputs (poor generalisation). Too many hidden nodes will allow the ANN to learn the training data to a high degree of accuracy, including the noise present in the data. In effect, instead of finding some underlying relationship between the input and output vectors, the ANN will have learnt each single relationship. This results in poor generalisation to previously unseen input vectors. So, to perform optimally on new test data an ANN should have just enough hidden nodes and no more.

It was with this idea in mind that I trained ANNs with varying numbers of hidden nodes on each pairing of input and output representations of the training data. The number of hidden nodes producing the best performance on the test data can be taken as a measure of how difficult the mapping from the input to the output spaces is for the particular representations used. The error should increase for ANNs using fewer hidden nodes. For ANNs using more hidden nodes, training to completion[2], might be expected to result in poorer error rates in the test data. However, for many of the ANNs I used the cross-validation methodology which should, in theory, prevent overtraining on the training data and poor generalisation. So, we might expect the pattern of errors for the best ANNs with each number of hidden nodes used to be either a V shape or to be decreasing and then flat.

[2] That is, until the error on the training set ceases to decrease or the output is within some specified distance of the target output.

6.7 The Speech Data Used in Training, Cross-validation and Testing

6.7.1 Source of the Speech Data

The training, testing and cross-validation data was extracted from a CSTR speech database produced for ATR [61]. The speech is single-speaker (mgsw), with 5000 single word utterances recorded. The speech files have been hand-labelled at the word, broad phonetic and fine phonetic levels. The speech data was recorded at 20000 Hz in a sound insulated room.

6.7.2 Processing of the Speech Data

Extracting Triphones

All Consonant-Vowel-Consonant (CVC) triphones present in the speech data base were extracted. The speech data contained instances of the consonants /p, t, k, b, d, g, m, n, ŋ, θ, ð, s, z, ʃ, ʒ, ʧ, ʤ, h, j, w, r, l, ʍ, x, f, v, w/ and the vowels /ɪ, i, ɛ, a, ɑ, ɒ, ɔ, ʊ, u, ɜ, ə, ʌ, aɪ, aʊ, ɛə, eɪ, ɪə, ɔɪ, əʊ, ʊə/.

Extracting Vowel Formant Tracks

For each triphone, a generalised centroid formant tracker [18] was run on the speech data to extract three formant tracks. The following summary data was recorded :

- The duration of the triphone.
- The duration of the vowel.
- The start time of the vowel in the triphone.

- The phonemes and stresses. No other label information was preserved. Where a phoneme was segmented internally (eg. into stop and burst), these segments were collapsed together.
- The frequency values of the 1st, 2nd and 3rd formant tracks at the following times :
    - The start of the vowel segment.
    - The centre of the vowel segment.
    - The end of the vowel segment.
- Information identifying the files containing the original speech data and segmentation, and the place within segmentation of the triphone.

Various representations of the vowel formant tracks were created as discussed in Section 6.5.

6.7.3 Partition of the Speech Data into Training, Cross-Validation and Test Sets

I decided to limit my experiments to CVC triphones containing stop consonants /p, t, k, b, d, g/ and monophthongs /ɪ, i, ɛ, a, ɑ, ɒ, ɔ, ʊ, u, ɜ, ə, ʌ/. This gave 554 triphones in all. The 554 triphones used were partitioned into 3 data sets — the training, cross-validation and test sets. A further set of triphones was selected from the training set to act as a further test set.

The Training Speech Data Set

The training set had to be as large as possible. A previous set of experiments using a specially recorded set of triphones had failed because of the lack of training data. The chosen training set contained 512 triphones, and ideally would have been much larger.

Ideally the cross-validation and test sets would have contained a large number of triphones also. However, the need to use as many triphones as possible in the training set was felt to be more important in order to ensure the likelihood of success.

The distribution of triphones in the training set is given in Tables 6.7 and 6.8. The distribution of initial and final consonants, summed over the vowels is given in Table 6.9. It is obvious that the distribution of triphones, vowels, consonants and pairs of initial and final consonants are all uneven. The expected effect of this is to produce better performance on vowels, consonants and triphones which are common in the training data, compared to those which are infrequent.

The Cross-Validation Speech Data Set

The cross-validation set was used in training of ANNs using the cross-validation paradigm. That is, after each iteration of the learning algorithm, the error on the cross-validation set was calculated. After the training was completed, the state of the ANN at the training iteration with the lowest error on the cross-validation set was taken as the output of the training process. This is intended to find the ANN configuration which generalises to new data best. See Section 6.6.2. The cross-validation set contained 20 triphones. 19 of the 20 triphones did not appear in the training set (ie. no triphone with the same three phonemes appeared in the training set), hopefully giving a good measure of generalisation. The triphones used are given in Table 6.10.

The Test Speech Data Set

The test set comprised 20 triphones which did not appear in the training or cross-validation sets (ie. no triphone with the same phonemes appeared in the other sets), providing a strong test of generalisation to previously unseen triphones. The vowel /u/ appeared in the test set (in the triphone /but/) but did not appear at all in the training or cross-validation sets. The triphones used are given in Table 6.11. After much of the ANN training was complete I realised that I

ɪ	p	t	k	b	d	g	
p	-	3	5	-	3	-	11
t	2	3	32	-	81	-	118
k	-	9	-	-	-	-	9
b	-	7	5	-	-	7	19
d	6	4	11	-	30	1	52
g	-	-	-	-	-	-	-
	8	26	53	-	114	8	209

i	p	t	k	b	d	g	
p	2	3	4	-	-	-	9
t	-	-	-	-	-	-	-
k	2	-	-	-	-	-	2
b	-	2	-	-	-	-	2
d	3	3	-	-	2	-	8
g	-	-	-	-	-	-	-
	7	8	4	-	2	-	21

ɛ	p	t	k	b	d	g	
p	-	-	15	-	-	-	15
t	3	-	9	-	3	-	15
k	-	-	-	-	-	-	-
b	-	1	-	-	3	-	4
d	2	-	3	-	-	-	5
g	-	4	-	-	-	-	4
	5	5	27	-	6	-	43

a	p	t	k	b	d	g	
p	-	3	2	-	-	-	5
t	-	-	8	3	-	-	11
k	4	3	-	3	-	-	10
b	-	-	3	-	2	-	5
d	-	-	-	-	-	-	-
g	-	-	-	-	-	-	-
	4	6	13	6	2	-	31

ɑ	p	t	k	b	d	g	
p	-	10	2	-	-	-	12
t	-	5	-	-	-	-	5
k	-	-	-	-	1	-	1
b	-	-	-	-	-	-	-
d	-	-	-	-	-	-	-
g	-	-	-	-	7	-	7
	-	15	2	-	8	-	25

ʌ	p	t	k	b	d	g	
p	-	-	-	3	-	-	3
t	-	-	-	-	2	-	2
k	3	2	-	-	-	-	5
b	-	2	-	-	-	-	2
d	-	-	6	-	-	-	6
g	-	-	-	-	-	-	-
	3	4	6	3	2	-	18

Table 6.7. Distribution of triphones containing ɪ, i, ɛ, a, ɑ and ʌ in the Training set. The initial consonant is to the left, the final consonant above the table.

ɒ	p	t	k	b	d	g	
p	2	2	-	-	-	-	4
t	-	-	2	-	-	-	2
k	2	5	-	-	-	-	7
b	-	2	2	5	-	-	9
d	2	3	-	-	-	2	7
g	-	2	-	-	1	-	3
	6	14	4	5	1	2	32

ɔ	p	t	k	b	d	g	
p	-	18	-	-	-	-	18
t	-	-	4	-	-	-	4
k	2	3	-	-	6	-	11
b	-	-	-	-	5	-	5
d	-	2	-	-	-	-	2
g	-	-	-	-	-	-	-
	2	23	4	-	11	-	40

ʊ	p	t	k	b	d	g	
p	-	4	-	-	-	-	4
t	-	-	-	-	3	-	3
k	-	-	2	-	-	-	2
b	-	-	2	-	-	-	2
d	-	-	2	-	-	-	2
g	-	-	-	-	2	-	2
	-	4	6	-	5	-	15

u	p	t	k	b	d	g	
p	-	-	-	-	-	-	-
t	-	-	-	-	-	-	-
k	-	-	-	-	-	-	-
b	-	-	-	-	-	-	-
d	-	-	-	-	-	-	-
g	-	-	-	-	-	-	-
	-	-	-	-	-	-	-

ə	p	t	k	b	d	g	
p	-	7	-	-	2	-	9
t	-	6	-	9	6	5	26
k	-	2	-	2	-	4	8
b	-	-	-	3	3	3	9
d	3	5	-	2	6	-	16
g	-	-	-	-	-	-	-
	3	20	-	16	17	12	68

ɜ	p	t	k	b	d	g	
p	2	2	-	-	-	-	4
t	-	-	-	-	-	-	-
k	-	3	-	-	-	-	3
b	-	2	-	-	3	-	5
d	-	-	-	-	-	-	-
g	-	-	-	-	-	-	-
	2	7	-	-	3	-	12

Table 6.8. Distribution of triphones containing ɒ, ɔ, ʊ, u, ə and ɜ in the Training set. The initial consonant is to the left, the final consonant above the table.

	p	t	k	b	d	g	
p	6	52	28	3	5	-	94
t	5	14	55	12	95	5	186
k	13	27	2	5	7	4	58
b	-	16	12	8	16	10	62
d	16	17	22	2	38	3	98
g	-	6	-	-	10	-	16
	40	132	119	30	171	22	514

Table 6.9. Summary of distribution of triphones in the Training set, over all the vowels. The initial consonant is to the left, the final consonant above the table.

Initial Consonant	Vowel	Final Consonant	Source
t	ə	p	'ɛntəpraɪz
k	ə	d	akəd'ɛmɪk
d	ə	k	'adəkwət
t	'ɜ	b	dɪst'ɜbd
t	ɜ	d	'ɛntɜd
d	'ɜ	t	d'ɜti
b	'a	g	b'ag
g	a	p	gap
k	'ɑ	d	k'ɑd
b	'ɛ	t	b'ɛtə
b	ɪ	d	bɪd
d	'ɛ	d	d'ɛd
d	ɪ	g	dɪgri
b	i	b	bibis'i
d	i	k	h'andɪkap
p	ɔ	d	pɔd
b	'ɔ	t	b'ɔt
t	'ʊ	k	t'ʊk
b	'ʊ	t	b'ʊtlə
b	'ʌ	k	b'ʌklŋ

Table 6.10. The triphones used for cross-validation.

should have included in the test set triphones which had the same phonemes as some in the training set, but different durations, in order to test how the ANNs performed on triphones with only slight differences to those seen before.

Initial Consonant	Vowel	Final Consonant	Source
p	ə	g	pɒpəg'andə
k	ə	p	kp'asəti
g	ə	t	'ngətɪv
t	ˌɜ	p	ɪntˌɜprɪt'eɪʃən
k	'ɜ	d	ək'ɜd
b	'a	p	b'aptɪzəm
b	'a	t	b'atl
d	'ɑ	k	d'ɑk
k	'ɛ	p	k'ɛpt
t	ɪ	g	ɪnvˌɛstɪg'eɪʃən
p	'i	d	sp'id
b	i	g	big'ɪnz
p	'ɒ	k	p'ɒkɪt
b	ɒ	b	bɒb
t	'ɔ	t	t'ɔt
g	'ɔ	d	g'ɔdən
k	'ʊ	d	k'ʊd
d	'ʊ	g	d'ʊgləs
d	'ʌ	b	d'ʌbəl
b	u	t	buts

Table 6.11. The Test triphones used for testing performance and generalisation.

The TrainTest Speech Data Set

It is impossible to make perceptual evaluations of performance on the entire training set due to its large size. I selected a set of 20 triphones from the training set to form the traintest set. This set was chosen to span the range of phoneme combinations in the training set, and should provide a measure of how well the ANNs learnt the training data. The triphones making up the TrainTest set are listed in Table 6.12.

Initial		Final	
Consonant	Vowel	Consonant	Source
k	'a	t	k'atəgəriz
k	'i	'p	k'i'pɪŋ
t	ɪ	d	əpɔɪntɪd
k	'ɒ	t	sk'ɒtlnd
d	ɒ	g	dɒgz
d	'ɔ	t	d'ɔtəz
t	'a	k	ət'ak
d	ɪ	p	dɪp'ɒzətɪd
d	ʌ	k	pr'ɒdʌkt
t	ɪ	k	ˌɔtəm'atɪk
d	'ɒ	p	əd'ɒpt
d	ɪ	d	h'andɪd
p	'i	k	sp'ikə
p	'ɔ	t	p'ɔt
p	ɛ	k	pr'ɒspɛks
d	i	t	diteɪl
t	'ɛ	k	t'ɛkst
k	ɑ	d	kɑdz
b	ɪ	t	bɪtw 'in
d	ɪ	p	dɪp'ɛnd

Table 6.12. The TrainTest triphones used for testing performance on members of the Training set.

6.8 Naming of ANN Training Trials

Each ANN training trial consisted of a neural network of a given size being trained with a particular set of input and output data, and possibly with a cross-validation set. The input and output representations are described in Chapter 6.2.

I have adopted a uniform naming scheme for these trials, in an attempt to make things clearer. Each trial name is given by concatenating the following:

1. The name of the input representation. See Table 6.13
2. The name of the output representation. See Table 6.14
3. If the ANN weights were set to those found at the completion of training (ie. when no further reduction of error occurred) then the string "end" is included. If the weights were those that gave a minimum error on the cross-validation triphone set training, then the no string is included.
4. If the training was carried out by back-propagation (using my program *ff*) then the string "bp" is included. All the two layer networks (without hidden layers) fall into this category. If the training is carried out by the conjugate-gradient method then the string "bp" is not included.
5. The number of nodes in the hidden layer. In some trials a two layer network was used. In these cases, this number is omitted.
6. The number of the trial. For each setup, a number of trials were run with different initial networks. This number differentiates between the trials and is also the random seed used to set up the initial weights in the ANN. Trials are numbered from zero.

So, for example, the third trial using the *Traditional* input representation, the *Tri-ratio* output representation and having 6 nodes in the hidden layer would be called *Trad.TRat.6.3*. The zeroth trial using the *Continuous* input representation, the *PolyBark* output representation, having no hidden layer and being trained to completion with the back-propagation algorithm, would be called *Cont.PolyBark.bp.end.0*.

Representation	Name
Traditional (Section 6.4.2)	*Trad*
Continuous (Section 6.4.3)	*Cont*
Symbolic (Section 6.4.4)	*Sym*

Table 6.13. The Names Used for Input Representations in Trial Names.

Representation	Name
Tri-ratio (Section 6.5.1)	*TRat*
Tri-difference (Section 6.5.1)	*TDif*
Tri-plain (Section 6.5.1)	*Tri*
Polynomials of Order 2 (Section 6.5.2)	*Poly*
Polynomials of Order 2 of Bark-scaled vowel formant tracks (Section 6.5.2)	*PolyBark*
Four Fourier Coefficients (7 real values) (Section 6.5.3)	*FFT4*
Four Fourier Coefficients of Bark-scaled vowel formant tracks (7 real values) (Section 6.5.3)	*FFTBark4*
Two Fourier Coefficients of Bark-scaled vowel formant tracks (3 real values) (Section 6.5.3)	*FFTBark2*

Table 6.14. The Names Used for Output Representations in Trial Names.

6.9 Combinations of Input, Output and Training Regime Used

The combinations of input representation, output representation, training algorithm and training methodology are shown in Table 6.15.

Name	Input Representation	Output Representation	Training Algorithm	Methodology
Trad.TRat.bp	Traditional	Tri-ratio	bp	Cross-validation
Trad.TDif	Traditional	Tri-difference	cg	Cross-validation
Trad.Tri.bp	Traditional	Tri-plain	bp	Cross-validation
Trad.Tri.bp.end	Traditional	Tri-plain	bp	To completion
Trad.TRat	Traditional	Tri-ratio	cg	Cross-validation
Trad.TRat.end	Traditional	Tri-ratio	cg	To completion
Trad.Tri	Traditional	Tri-plain	cg	Cross-validation
Trad.Tri.end	Traditional	Tri-plain	cg	To completion
Trad.Tri.bp	Traditional	Tri-plain	bp	Cross-validation
Trad.FFT4	Traditional	FFT4	cg	Cross-validation
Trad.FFTBark4	Traditional	FFTBark4	cg	Cross-validation
Trad.FFTBark2	Traditional	FFTBark2	cg	Cross-validation
Trad.Poly	Traditional	Polynomial	cg	Cross-validation
Trad.PolyBark	Traditional	Polynomial Bark	cg	Cross-validation
Trad.Poly.end	Traditional	Polynomial Bark	cg	To completion
Sym.TRat	Symbolic	Tri-ratio	cg	Cross-validation
Cont.TRat	Continuous	Tri-ratio	cg	Cross-validation

Table 6.15. The experimental trials. “bp” stands for back-propagation and “cg” stands for conjugate gradient.

6.10 Evaluation Methods

6.10.1 Introduction

The only real tests of synthesised speech quality are those which involve presenting speech to people. These take a large amount of time and effort, so are infeasible to apply to large numbers of sets of triphones. Hence, I have only used an intelligibility test to a) produce a measure which can be applied to large numbers of vowel formant sets with little cost in time and effort (Experiment I in Chapter 7), and b) as the final test for the triphones produced by the neural networks selected using the measure produced (Experiment II in Chapter 8).

6.10.2 Intelligibility Measures

I have used an intelligibility measure based upon the modified rhyme test as used in [7], which was in turn based on the rhyme test of [25]. Subjects are presented with a series of synthesised utterances and have to select one of a number of responses on a response sheet. Since I am changing the vowels, not the consonants, the test is in fact more of an assonance test than a rhyme test.

I devised words (some nonsense and some real) for all of the vowels /ɪ, i, ɛ, a, ɑ, ɒ, ɔ, ʊ, u, ɜ, ə, ʌ/ and all of the consonant contexts /b_b, b_g, b_p, b_t, d_b, d_d, d_g, d_k, d_p, d_t, g_d, k_d, k_p, k_t, p_d, p_k, p_t, t_d, t_g, t_k, t_p, t_t/ in the test set and traintest set. It became apparent that the test set triphones with the vowel /ə/ could not be used, since I could not create words for the /ə/ case for all the consonant contexts. There was also the risk that any poor quality utterance would be classified as a /ə/ triphone, with no other attempt to identify the nearest vowel. This resulted in there being 37 triphone utterances used per neural network in the intelligibility tests.

The utterances were synthesised as described in Section 6.11. They were recorded on a Marantz CP230 cassette recorder using Dolby B noise reduction onto TDK AR cassette tape. A gap of 3 seconds was left between utterances (using the Unix 'sleep' command, so some of the gaps may have been longer — no unevenness was apparent). An extra 4 second gap was added where the subjects had to turn a page of the response sheet. This spacing gave subjects enough time to respond but gave no time for consideration.

For each utterance the subject had to circle one of six words. The set of responses for a particular triphone remained the same throughout the test. All triphones with the same consonant context had the same six words to choose from. This was so that the subjects could not learn which answer was the correct one for a given pair of consonants. This is not normally a problem with intelligibility tests, but as so many different versions of the same triphone were to be presented in one test (up to 12 times) this was necessary. The possible responses for each consonant context are shown in Table 6.16

6.10.3 Sources of Intelligibility Error

There are three ways in which differences in the errors in the intelligibility of the speech produced by the ANNs may come about:

1. The ANNs may differ in how accurately they learn to produce the output vectors of the training set.
2. They may differ in how well they generalise to produce the output vectors for previously unseen data.
3. For any particular size of error on the output nodes of the ANN, a representation may produce more or less intelligible speech.

6.10.4 The Formant Track Error Measure

It would be desirable to have a measure of how good a synthesised utterance is without the large amount of work involved in an intelligibility test such as the Modified Rhyme Test in which subjects are necessary. This would allow the following:

context	*vowel and matching word*											
b_b	ɒ	bob	a	bab	ɑ	barb	ɪ	bib	ɔ	borb	u	boob
b_g	i	beeg	ɛ	beg	ɪ	big	ʌ	bug	ɜ	berg	a	bag
b_p	a	bap	ɛ	bep	ɪ	bip	i	beep	ɜ	burp	u	boop
b_t	ɪ	bit	a	bat	ɛ	bet	u	boot	ɜ	bert	ɑ	bart
d_b	ʌ	dub	a	dab	ɛ	deb	ɪ	dib	ɜ	durb	ɒ	dob
d_d	ɪ	did	a	dad	ɛ	dead	i	deed	u	dood	ʌ	dud
d_g	ɒ	dog	a	dag	ɪ	dig	ɔ	dawg	ʊ	dug	ɜ	dirg
d_k	ɑ	dark	ɒ	dock	ʌ	duck	ɜ	dirk	u	dook	a	dack
d_p	ɪ	dip	a	dap	ɛ	dep	ɔ	dorp	u	doop	ɒ	dop
d_t	i	deet	ɪ	dit	ɔ	daught	ɜ	dirt	ɑ	dart	u	doot
g_d	ɔ	gored	ɒ	god	ʊ	good	ɜ	gird	ɑ	guard	a	gad
k_d	ʊ	could	ɑ	card	ɔ	cawed	ɒ	cod	ɛ	ked	ɜ	curd
k_p	ɛ	kep	ɪ	kip	i	keep	ʌ	cup	u	coop	ɜ	curp
k_t	ɒ	cot	a	cat	ɑ	cart	ɛ	ket	u	coot	ɜ	curt
p_d	i	peed	ɛ	ped	ɪ	pid	ʊ	pud	u	poohed	ɜ	purred
p_k	i	peak	ɛ	peck	ɪ	pick	ɔ	pork	ɜ	perk	ɒ	pock
p_t	ɔ	port	ɑ	part	ɒ	pot	ʌ	putt	ɜ	pert	a	pat
t_d	ɪ	tid	ɛ	ted	ɑ	tarred	i	teed	ɒ	tod	u	tood
t_g	ɪ	tig	a	tag	ɛ	teg	i	teague	ʌ	tug	ɒ	tog
t_k	a	tack	ɛ	tech	ɪ	tick	i	teak	ɜ	turk	ʌ	tuck
t_p	ɜ	turp	a	tap	ɒ	top	ɔ	torp	ʌ	tup	u	toop
t_t	ɔ	taut	ɑ	tart	ɒ	tot	ʌ	tut	ɜ	turt	u	toot

Table 6.16. The possible response words for each consonant context and choice of vowel.

- "Instant" evaluation of alterations to synthesis methods and comparisons between methods.
- Selection of the best versions of a synthesis method (or of types of input and output representation and numbers of hidden units in ANNs, as in this work), prior to final evaluation using a more costly intelligibility measure.

The example of such a method which I have used is the root-mean-square error of the synthesised vowel formant tracks compared with the original vowel formant tracks for the same utterance, taken from real speech. This is determined by summing the square of the difference between the formants at a number of time intervals (the sample rate is a natural choice for digitised speech), then dividing by the number of intervals before taking the square root:

$$\sqrt{\frac{\sum_{t=1}^{n}(s_t - o_t)}{n}} \tag{6.6}$$

where s_t is the value of the synthesised formant at time t and o_t is the value of the original formant at time t.

A number of other researchers have used this error measure (for instance, [11, 56]) but have used it as the single evaluation method with no further justification or attempt to relate it to other evaluation methods.

6.11 Synthesising Triphones From ANN Output and Vowel Formant Representations

The output from a ANN is a vector of numbers between 0 and 1. From this output we must eventually produce an utterance, a CVC triphone. The stages involved in producing speech from the output of a given ANN were:

1. Reverse any transformations used to produce an output representation with values in the range [0, 1], to create one of the output representations described in Section 6.5.

2. Produce the first three vowel formant tracks from the output representation.

3. Synthesise the vowel from the formants.

4. Concatenate the synthesised vowel and tokens of the initial and final consonants.

5. Play back the resultant CVC triphone using the equipment described in Section 6.10.2.

6.11.1 Synthesising Vowels From Vowel Formants

Synthesis of vowels from the vowel formants produced by the ANNs or from inverting the formant representations was carried out by my own implementation of the formant synthesis algorithm described in [51]. This implementation was designed to be as flexible as possible, allowing the digital components (such as resonators, antiresonators and noise sources) to be connected up as the user wishes, and for it to be a simple matter to add new types of component. It was implemented in C++, as the object-oriented approach suited the type of design I wished to create. More details are given in Appendix A.

6.11.2 Creating the CVC Triphones

The vowel portion of the triphone was synthesised as described in Section 6.11.1. The triphone was created by concatenating the synthesised vowel with tokens of the initial and final consonants. These tokens were extracted from speech from the same speaker as had been used for training and testing the ANNs. The initial and final sets were distinct, but each contained only one example of each consonant. That is, there was no attempt to use initial and final consonants which came from the context in which they were to be used. This will have the effect of reducing the quality of the synthesised triphones, but will mean that correct identification of the vowel rests on the quality of the vowel.

Audio output of the digitised speech produced by the synthesis was via CSTR's audio output system (*ao*, which uses a Macintosh computer with National Instruments NB-AO-6 Rev.C board to carry out the DA conversion which is then amplified by a Yamaha A-09 audio amplifier).

Chapter 7

Experiment I: Relating Formant Track Errors and Intelligibility Errors

7.1 Experimental Setup

12 sets of output data from ANN training trials were chosen randomly from the large number available. The small number of trials which resulted in very extreme formant track errors were omitted from the available choices. The trials chosen gave a reasonable coverage of the range of formant track errors for all three formants. Figure 7.1 shows the distribution of the chosen trials within the full set of trials, for F1 and F2 formant errors.

The 37 triphones for each trial were synthesised as detailed in Section 6.11, resulting in 444 utterances. These were recorded as described in Section 6.10.2, resulting in a tape that was approximately 32 minutes long, plus a two minute break at the half-way point. It quickly became apparent that the break was not needed and it was skipped for all except the first couple of subjects. The front page and first page of the booklet given to the subjects is shown in Appendix B.

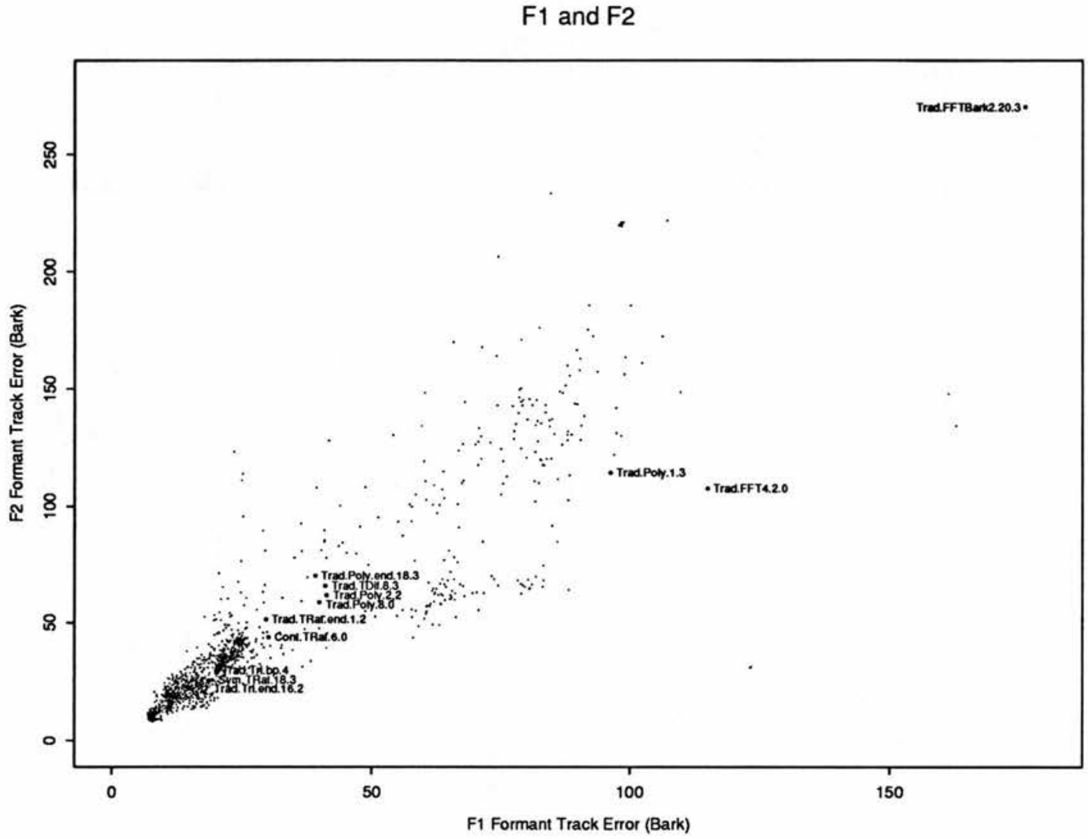

Figure 7.1. The F1 and F2 distributions of the formant track errors for each ANN. The randomly picked trials used in deriving a relationship between formant track errors and intelligibility errors are labeled. The formant track errors shown are the sums of the errors over the 40 utterances used.

The trial utterances were prefaced by 10 example words (all CVC triphones), taken from the training set and synthesised using the original vowel formant tracks. The purpose of this was to familiarise the subjects with the sound of the synthesis and the method of presentation. The recorded utterances were played back through the Marantz CP230 cassette recorder used to record them, amplified by a Revox tape recorder and listened to via Revox 3100 headphones in soundproof booths.

A total of 25 subjects were used. There were 11 Scottish and 12 English speakers, plus two others (American and Irish). I asked for some indication of accent from the subjects because I expected there to be differences between English and Scottish speakers. Scottish accents have preserved post-vocalic /r/ [44, 103]. There is no vowel /ɜ/, and the following contrasts are missing: /a/ v /ɑ/, /ɒ/ v /ɔ/ and /ʊ/ v /u/. Scots will regard a word whose spelling contains a vowel followed by an 'r' as containing the sound /r/, which does not occur in the triphones of the RP speech upon which the synthesis is based. I expected problems with triphones containing the vowels /ɜ, a, ɑ, ɒ, ɔ, ʊ, u/. Two types of error were possible. Firstly, synthesised triphones that contained these triphones were likely to be misclassified as containing other vowels. Secondly, where the triphone did not contain ɜ, the triphone was very unlikely to be mistakenly classified as containing /ɜ/. This increases the chance of a correct classification, even if the vowel is poor, as the subject is choosing between 5 possibilities instead of 6.

7.2 Experimental Results

The percentage errors for the 25 subjects and sets of triphones produced by 12 different ANNs are shown in Figure 7.2. A *boxplot* of these results is shown in Figure 7.3. A boxplot shows the distribution of data within each set. For each set (here the intelligibility errors for each ANN), a box, whiskers (the lines extending from the box) and outliers (the single points) are plotted. The centre of the box is the median. The box shows the inter-quartile range (the central half of the data). The whiskers extend to the nearest point to, but not beyond, the interquartile range multiplied by 1.5. The outlying points are all the points beyond the whiskers.

For example, in the figure, Trad.PolyBark.end.18.3 has a small inter-quartile range and short whiskers, showing that the intelligibility errors for most subjects were fairly evenly spread within a small range. There are two outliers, so two subjects had atypical results. Trad.Poly.8.0 has a large interquartile range, so the main body of subjects produced a wide range of intelligibility errors. The whiskers are short in relation to it, and there are no outliers, so no subjects produced errors far from the normal range.

7.3 Rejecting Intelligibility Errors That May be due to Chance

There is a level of intelligibility error beyond which the measured error might be due to chance. The points in this region cannot be used, since there will be no strong relationship between the formant errors and the intelligibility error. I chose to keep points only where the probability of the error mark being due to chance is less than 15%.

There are 37 responses per experiment, and 25 subjects, giving 925 responses per experiment in all. If responses are randomly picked, then the probability of being correct on one response is 1/6. The distribution is binomial, but can be approximated by the Normal distribution. At the 15% level, this matches 287 correct responses, which gives an Intelligibility Error of 69%.

On this basis I have decided to only use the responses for experiments Trad.-Tri.end.16.2, Trad.Tri.bp.4, Sym.TRat.18.3, Trad.TRat.end.1.22, Cont.TRat.6.0, and Trad.PolyBark.end.18.3.

Looking at the boxplot for the 12 experiments (Figure 7.3), we see that Trad.PolyBark.end.18.3 has a low variability of marks between subjects, whereas

Subject	Trad.Tri.end.16.2	Trad.Tri.bp.4	Sym.TRat.18.3	Trad.TRat.end.1.2	Cont.TRat.6.0	Trad.PolyBark.end.18.3	Trad.Poly.8.0	Trad.FFT4.2.0	Trad.Poly.1.3	Trad.Poly.2.2	Trad.FFTBark2.20.3	Trad.TDif.8.3	Mean
sub1	32.4	27.0	37.8	51.4	59.5	67.6	75.7	73.0	81.1	73.0	81.1	83.8	61.9
sub2	29.7	29.7	32.4	56.8	56.8	62.2	75.7	73.0	62.2	75.7	70.3	83.8	59.0
sub3	24.3	35.1	45.9	67.6	67.6	78.4	81.1	70.3	75.7	83.8	75.7	83.8	65.8
sub4	32.4	27.0	35.1	54.1	70.3	67.6	75.7	81.1	78.4	81.1	89.2	83.8	64.6
sub5	21.6	21.6	27.0	56.8	56.8	67.6	64.9	78.4	81.1	73.0	89.2	73.0	59.2
sub6	24.3	24.3	37.8	64.9	62.2	70.3	78.4	78.4	81.1	81.1	86.5	86.5	64.6
sub7	32.4	27.0	37.8	64.9	62.2	67.6	73.0	70.3	70.3	83.8	83.8	81.1	62.8
sub8	27.0	35.1	18.9	59.5	56.8	70.3	83.8	78.4	81.1	78.4	83.8	81.1	62.8
sub9	21.6	27.0	35.1	56.8	64.9	70.3	81.1	78.4	75.7	81.1	86.5	81.1	63.3
sub10	29.7	24.3	37.8	45.9	70.3	56.8	64.9	62.2	78.4	81.1	73.0	73.0	58.1
sub11	29.7	32.4	21.6	59.5	54.1	67.6	81.1	81.1	78.4	86.5	83.8	86.5	63.5
sub12	32.4	29.7	40.5	56.8	67.6	67.6	81.1	75.7	78.4	81.1	89.2	89.2	65.8
sub13	27.0	37.8	45.9	59.5	73.0	73.0	81.1	78.4	81.1	86.5	83.8	86.5	67.8
sub14	24.3	27.0	24.3	29.7	59.5	67.6	78.4	73.0	78.4	81.1	83.8	83.8	59.2
sub15	27.0	24.3	24.3	56.8	62.2	70.3	81.1	75.7	64.9	83.8	83.8	81.1	61.3
sub16	29.7	24.3	27.0	48.6	59.5	64.9	67.6	73.0	70.3	78.4	83.8	83.8	59.2
sub17	29.7	27.0	29.7	54.1	59.5	64.9	67.6	67.6	73.0	73.0	78.4	75.7	58.3
sub18	29.7	45.9	32.4	59.5	67.6	62.2	70.3	75.7	78.4	78.4	75.7	75.7	62.6
sub19	35.1	32.4	37.8	62.2	62.2	67.6	81.1	78.4	78.4	81.1	75.7	81.1	64.4
sub20	18.9	32.4	21.6	43.2	56.8	70.3	62.2	83.8	73.0	75.7	83.8	75.7	58.1
sub21	29.7	29.7	18.9	56.8	54.1	59.5	81.1	81.1	83.8	75.7	75.7	75.7	60.1
sub22	29.7	27.0	37.8	59.5	64.9	73.0	75.7	78.4	73.0	86.5	78.4	86.5	64.2
sub23	21.6	27.0	37.8	64.9	32.4	62.2	67.6	70.3	73.0	75.7	62.2	83.8	56.5
sub24	27.0	29.7	35.1	67.6	54.1	70.3	64.9	70.3	78.4	81.1	81.1	73.0	61.0
sub25	24.3	29.7	29.7	54.1	62.2	62.2	75.7	73.0	83.8	89.2	89.2	81.1	62.8
Mean	24.4	26.2	29.6	54.3	58.6	65.2	72.9	73.2	74.7	78.6	79.2	79.6	61.9

Figure 7.2. The percentage error for triphones in the intelligibility test used to relate intelligibility error and formant track error.

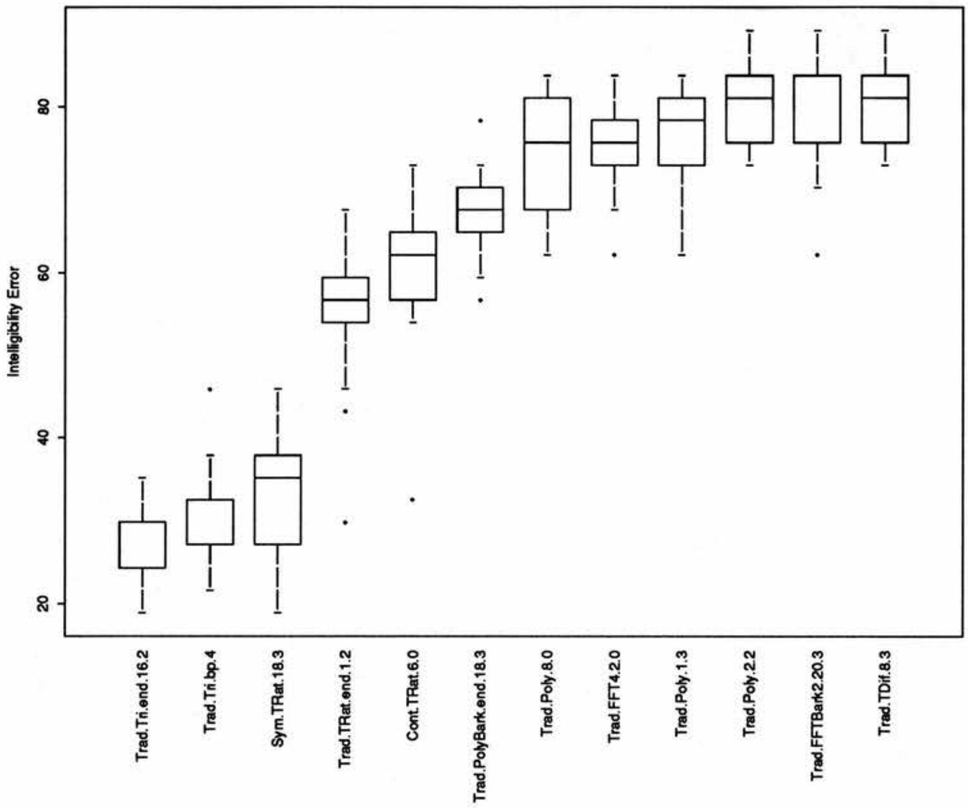

Figure 7.3. The subject intelligibility errors for the 12 ANN triphone sets used in the experiment relating intelligibility error and formant error.

Trad.Poly.8.0 and the worse experiments have higher variabilities. This further suggests that Trad.PolyBark.end.18.3 is less subject to guessing than the 6 worst experiments. However, there is little difference in variability between some of the 6 worst experiments and some of the 6 best experiments. A boxplot for just the 6 best experiments is shown in Figure 7.4.

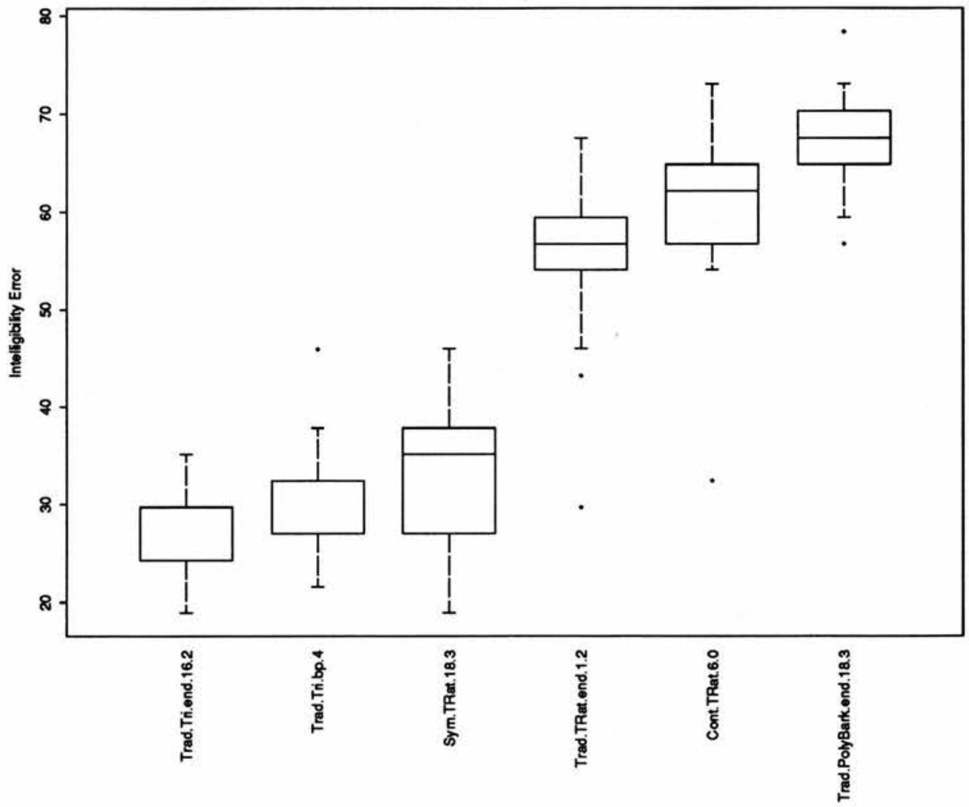

Figure 7.4. Overall intelligibility errors for the best 6 triphone sets used in the experiment relating intelligibility error and formant error.

7.4 Derivation of a Relationship Between Formant Track Errors and Intelligibility Errors

Correlations between the intelligibility error, the mean vowel formant errors per utterance (referred to as "formant errors" henceforth) and a number of transformations of the formant errors are shown in Table 7.1.

	error	F1	F2	F3
error	**1.00**	0.97	0.93	0.49
F1	**0.90**	1.00	0.98	0.25
F2	**0.86**	0.98	1.00	0.17
F3	**0.45**	0.25	0.17	1.00
$F1^2$	**0.87**	0.99	0.99	0.16
$F2^2$	**0.82**	0.97	0.99	0.05
$F3^2$	**0.46**	0.26	0.17	0.99
$\log F1$	**0.91**	1.00	0.98	0.33
$\log F2$	**0.89**	0.98	0.99	0.27
$\log F3$	**0.45**	0.26	0.19	0.99
$\sqrt{F1}$	**0.91**	1.00	0.98	0.29
$\sqrt{F2}$	**0.88**	0.99	1.00	0.22
$\sqrt{F3}$	**0.45**	0.25	0.17	1.00
$1/F1$	**-0.92**	-0.98	-0.96	-0.40
$1/F2$	**-0.90**	-0.97	-0.97	-0.36
$1/F3$	**-0.47**	-0.29	-0.23	-0.96
$1/F1^2$	**-0.92**	-0.97	-0.94	-0.45
$1/F2^2$	**-0.90**	-0.95	-0.94	-0.42
$1/F3^2$	**-0.49**	-0.34	-0.28	-0.93

Table 7.1. The correlations between the intelligibility error and various transformations of the formant errors.

F3 formant track error and transformations of the F3 formant track error were poorly correlated with the intelligibility error. F2 was highly correlated with the intelligibility error at 0.86, but F1 was better at 0.90. The best correlation with the intelligibility error was that for 1/F1 at -0.92. However, with only a small difference in correlation, only 6 groups of samples and no strong reason to suspect that the relationship is anything but linear, I have chosen to fit a linear regression between the mean F1 vowel formant error and the intelligibility error.

An F test of the regression of F1 and the intelligibility error gave an $F_{1,148}$ of 614.3, which is significant at the 0.1% level, so this fit was definitely a significant one. Adding further variables to the regression equation (eg. transformations of F1, F2 and F3) did not significantly improve the fit. The regression obtained is

$$\text{Intelligibility Error} = -8.94 + 83.34(\text{F1 error}) \tag{7.1}$$

The mean of the absolute values of the residuals is 6.15, and the maximum absolute value of a residual is 23.09. It must be remembered that these are the difference between the fit and individual errors per subject, not the mean values over the subjects. The errors in predicting the mean value of intelligibility error for each of the F1 formant errors used are shown in Figure 7.5.

Triphone Set	Mean Value	Predicted Value	Residual
Trad.Tri.end.16.2	27.68	29.85	-2.18
Trad.Tri.bp.4	29.41	32.90	-3.49
Sym.TRat.18.3	32.43	31.53	0.90
Trad.TRat.end.1.22	56.43	52.82	3.61
Cont.TRat.6.0	60.65	54.03	6.62
Trad.PolyBark.end.18.3	67.24	72.70	-5.46

Figure 7.5. The residuals of the predicted mean values of intelligibility error at each value of F1 vowel formant error used.

One reason why the use of only one variable is significant in the regression is that the F1, F2 and F3 formant errors are highly correlated for the formants produced by the neural networks (this may not be the case for speech produced by other methods). This means that the error for F1 gives a good measure of the error for F2 and F3, so the F2 and F3 errors add little extra information.

7.5 Confidence Bands on the Derived Relationship

The relationship I have derived above (Equation 7.1) gives an estimate of what the mean intelligibility error would be for the set of utterances used when presented to a very large number of subjects. We need to determine some error bounds on the predicted values and obtain a measure of how different two predicted values must be before we can be sure that the two sets of utterances being compared really are different.

A confidence interval for the mean value of the intelligibility error $\mu_{Y|X_0}$ (ie. the average intelligibility error over all subjects, which is what we are really interested in) for a given F1 vowel formant error is given by Equation 7.2, where X_0 is the given vowel formant error and $\hat{Y}_{X_0}$ is the predicted mean value of the intelligibility error at X_0. The estimate of the standard deviation of the predicted value $S_{\hat{Y}_{X_0}}$ is given by Equation 7.3. See, for example, [53] for details of the derivation of this confidence interval.

$$\text{Confidence interval for } \mu_{Y|X_0} \text{ is } \hat{Y}_{X_0} \pm t_{n-2,1-\alpha/2} S_{\hat{Y}_{X_0}} \tag{7.2}$$

$$S_{\hat{Y}_{X_0}} = S_{Y|X}\sqrt{\frac{1}{n} + \frac{(X_0 - \bar{X})^2}{(n-1)S_X^2}} \tag{7.3}$$

$$S_{Y|X}^2 = \frac{1}{n-2}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{7.4}$$

$$S_X^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2 \tag{7.5}$$

For the values used to calculate the regression, $S_{Y|X}^2 = 59.61$ and $S_X^2 = 0.018$, with $\bar{X} = 0.655$. Hence, the value of $S_{\hat{Y}_{X_0}}$ for the regression obtained is as given in Equation 7.6.

$$S_{\hat{Y}_{X_0}} = 7.72\sqrt{\frac{1}{150} + \frac{(X_0 - 0.655)^2}{2.68}} \tag{7.6}$$

The regression line and the confidence bands are shown in Figure 7.6.

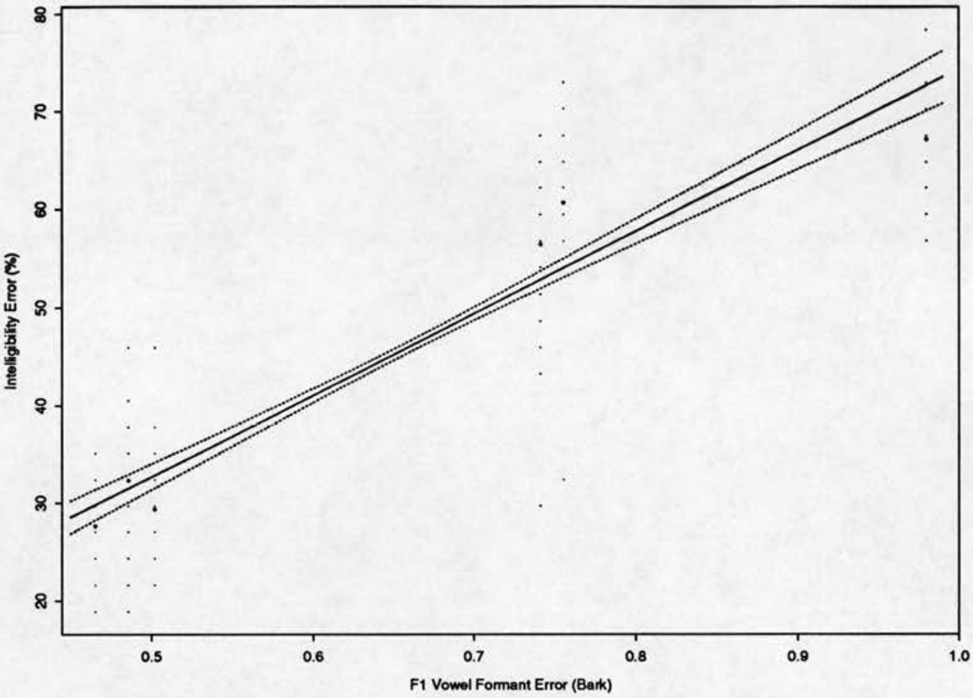

Figure 7.6. The regression line relating F1 formant error and intelligibility error, with 90% confidence bands shown.

Estimates of how far apart two predicted intelligibility values must be in order to be sure (to some confidence level) that they are indeed different is shown in Figure 7.7. These distances are the same as the distance from the regression line to the confidence bands.

7.6 Differences Between the Responses of Scots and English Speakers

The subjects divided roughly into Scots and English groups (with two others). While not connected with the main area of research I was interested in whether

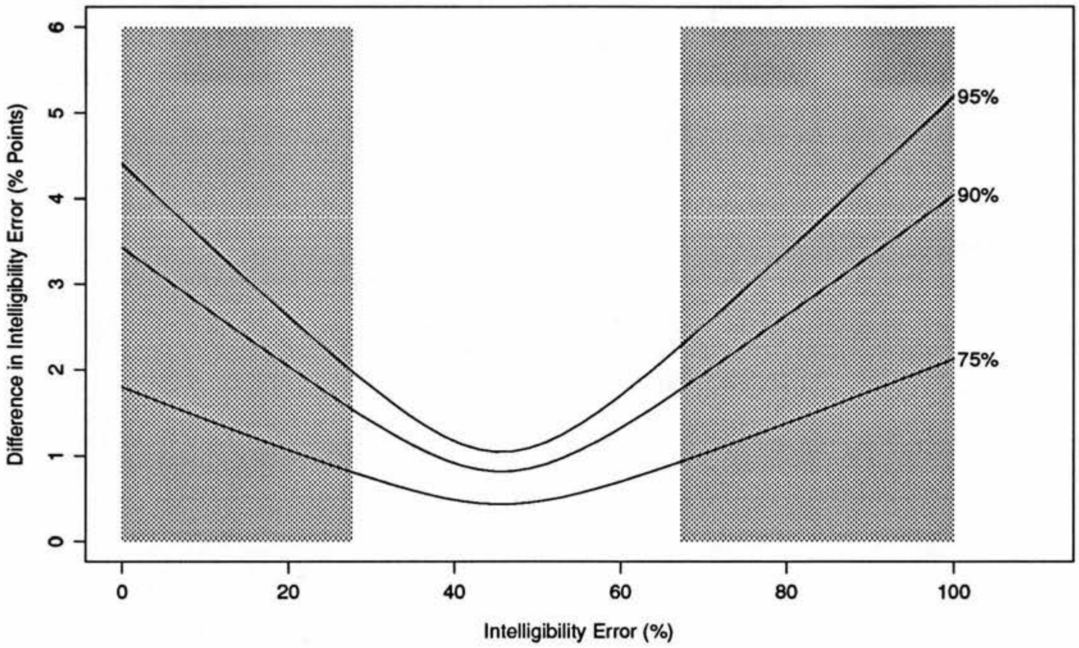

Figure 7.7. The width of the various confidence bands for the predicted intelligibility errors. If two predicted values differ by more than the indicated figure then there is that level of confidence that they are truly different. The shaded areas represent levels of intelligibility error outside the upper and lower limits of the points used in deriving the regression, and so are less firmly based than the unshaded area. The confidence bands are based on a one-sided t-test.

there would be a difference in the responses of these two groups. The overall marks for each of the 11 Scots subjects and 12 English subjects are shown in Table 7.2. Comparing these marks with a T test shows that there is a significant difference at the 0.5% level between the English and Scots subjects.

Scots						Mean 176.5			Sample SD 9.6			
Subject	1	8	14	16	17	18	20	21	23	24	25	
Mark	169	165	181	181	185	166	186	177	193	173	165	
English						**Mean 162.2**			**Sample SD 11.6**			
Subject	2	3	4	5	7	9	11	12	13	15	19	22
Mark	182	152	157	181	165	163	162	152	143	172	158	159

Table 7.2. The overall marks for the 11 Scottish subjects and 12 English subjects.

Comparing the results of each experiment for the Scots and English subjects (Figure 7.3) shows a significant difference (5% level) on six sets of triphones (Trad.TDif.8.3, Sym.TRat.18.3, Cont.TRat.6.0, Trad.Poly.2.2, Trad.Poly.8.0 and Trad.PolyBark.end.18.3) and on the overall marks.

Triphone	English		Scots	
Set	Mean	SE	Mean	SE
Trad.TDif.8.3	7.6	1.6	6.3	1.5
Trad.Tri.bp.4	25.7	2.2	26.2	1.7
Trad.TRat.end.1.2	17.2	3.9	15.1	1.4
Sym.TRat.18.3	26.5	2.6	24.0	2.9
Cont.TRat.6.0	16.1	3.3	13.5	2.1
Trad.Tri.end.16.2	27.1	1.5	26.4	1.6
Trad.FFT4.2.0	9.5	1.8	8.7	1.4
Trad.FFTBark2.20.3	7.5	2.6	6.5	2.3
Trad.Poly.1.3	8.3	1.7	9.3	2.3
Trad.Poly.2.2	8.1	1.7	6.7	1.6
Trad.Poly.8.0	10.3	2.6	8.3	1.9
Trad.PolyBark.end.18.3	12.7	1.4	11.3	1.5

Table 7.3. Mean marks and standard errors for the Scots and English subjects.

There are significant differences between the confusion matrices for Scots and English subjects (see Tables 7.4, 7.5 and 7.6).

Actual	Correct Choice											
Choice	ɜ	a	ɑ	ɛ	ɪ	i	ɒ	ɔ	ʊ	ʌ	u	total
ɜ	11.2	16.6	6.3	10.1	4.2	11.4	7.7	14.7	3.7	5.5	5.5	96.8
a	0.1	14.6	1.9	1.1	1.1	—	1.7	0.2	0.1	1.6	0.9	23.3
ɑ	0.3	2.4	8.0	—	0.1	0.3	4.9	5.7	0.2	0.4	1.1	23.3
ɛ	0.1	5.3	0.3	11.6	7.8	2.6	2.3	—	0.1	0.6	0.4	31.0
ɪ	—	1.4	—	1.6	55.8	15.5	3.3	1.1	2.5	0.6	1.1	82.7
i	—	0.2	—	0.2	0.5	11.0	0.3	0.2	—	—	—	12.4
ɒ	0.5	0.9	0.6	1.4	1.0	1.7	19.7	2.4	2.4	2.0	—	32.6
ɔ	3.3	—	3.1	2.0	—	4.1	12.4	15.1	2.5	—	—	42.4
ʊ	2.7	—	2.3	—	—	1.2	1.8	4.7	12.6	—	—	25.4
ʌ	2.8	1.9	0.6	6.6	10.0	4.3	—	1.2	—	12.1	—	39.5
u	3.0	4.8	0.9	1.6	3.4	8.0	5.9	2.6	—	1.4	2.9	34.5
none	—	—	—	—	0.1	—	—	0.1	—	—	0.1	0.3
total	24	48	24	36	84	60	60	48	24	24	12	444

Table 7.4. Confusion matrix of vowels for the Scots subjects in the first intelligibility test.

If we represent the choices where Scots subjects make a confusion significantly more often than the English subjects by S and the choices where the English subjects make a confusion significantly more than often than the Scots by an E, we have Table 7.7.

As discussed in Section 7.1, the Scots subjects lack the contrasts /a/ v /ɑ/, /ɒ/ v /ɔ/ and /ʊ/ v /u/ and would expect that the words taken as containing an /ɜ/, /ɑ/ or /ɔ/ for the purposes of the experiment would really contain an /r/, and this sound would not be present in any of the presented utterances. As expected from this, the English subjects correctly identified these vowels more often than the Scots subjects.

In fact, we can see that the English subjects generally preferred to respond with /ɜ/ and /ɑ/. It seems that if the vowel was poor it was reported as one of

						Correct Choice						
Actual Choice	ɜ	a	ɑ	ɛ	ɪ	i	ɒ	ɔ	ʊ	ʌ	u	total
ɜ	16.7	20.8	7.8	18.3	6.3	16.7	12.8	19.4	6.8	7.0	6.8	139.3
a	—	13.6	0.8	0.7	1.2	0.2	1.7	0.1	—	1.2	0.3	19.7
ɑ	0.3	2.9	10.3	—	0.3	0.2	8.3	6.3	0.2	0.2	1.1	29.9
ɛ	—	3.3	0.2	8.9	8.3	1.9	2.0	—	0.1	0.5	0.3	25.4
ɪ	—	0.8	—	0.8	52.3	14.5	1.3	0.7	2.8	0.4	0.8	74.3
i	—	—	—	0.7	1.1	11.3	0.3	0.7	—	—	—	14.0
ɒ	0.6	0.7	0.3	0.5	1.0	0.9	11.5	1.3	1.8	1.3	—	19.8
ɔ	2.3	—	1.8	1.4	—	3.9	13.8	13.3	1.4	—	—	38.1
ʊ	0.6	—	1.3	—	—	0.3	1.5	2.7	10.6	—	—	16.9
ʌ	2.1	0.7	0.3	3.3	8.8	3.3	—	0.2	—	11.6	—	30.1
u	1.3	3.6	0.8	0.8	3.7	5.6	3.6	2.5	—	1.4	2.2	25.4
none	0.2	1.8	0.4	0.7	1.3	1.3	3.3	0.9	0.3	0.5	0.6	11.2
total	24	48	24	36	84	60	60	48	24	24	12	444

Table 7.5. Confusion matrix of vowels for the English subjects in the first intelligibility test.

Actual	Correct Choice											
Choice	ɜ	a	ɑ	ɛ	ɪ	i	ɒ	ɔ	ʊ	ʌ	u	total
ɜ	0.4	0.5	9.5	—	5.4	0.6	0.5	2.8	2.9	20.0	2.7	0.1
a	30.7	25.0	3.8	30.4	85.6	17.1	92.8	50.6	30.7	25.0	5.4	11.3
ɑ	76.6	13.8	0.6	—	45.0	55.9	0.1	62.3	92.8	30.0	98.1	0.4
ɛ	30.7	1.1	55.9	3.8	78.7	39.7	61.4	—	95.2	63.0	57.5	18.1
ɪ	—	26.6	—	7.4	11.3	64.5	0.4	6.2	31.6	68.0	28.3	11.5
i	—	13.4	—	11.2	25.7	85.0	92.3	1.8	—	—	—	—
ɒ	92.6	57.3	22.9	1.1	100.0	10.2	—	25.9	27.9	19.0	—	1.0
ɔ	33.4	—	10.5	8.9	—	79.3	37.8	25.8	12.8	—	—	30.0
ʊ	—	—	6.9	—	—	8.4	61.4	2.1	15.8	—	—	—
ʌ	39.0	0.8	16.7	0.1	15.9	15.0	—	1.4	—	69.0	—	1.0
u	3.9	11.1	80.0	9.8	83.0	11.3	2.3	84.8	—	91.0	8.7	6.4
none	17.1	5.9	9.7	15.5	9.6	5.7	5.3	14.5	22.8	5.0	6.7	4.8

Table 7.6. Level of significance (in %) of differences between the confusion matrices of vowels for Scots and English subjects.

these by the English subjects. They did not have /ə/ as a possible response. The Scots subjects, if they had any preferred response, preferred words which would contain the vowels /ɒ/ or /ʊ/ or /u/ in RP, and correctly identified these more often than the English subjects. The distribution of vowels was such that the Scots subjects gave more correct responses than the English subjects. For some reason the Scots subjects seemed better at following the instruction to make some response for each utterance.

						Correct						
Actual						Choice						
Choice	ɜ	a	ɑ	ɛ	ɪ	i	ɒ	ɔ	ʊ	ʌ	u	total
ɜ	E	E		E		E	E	E	E		E	E
ɑ			E				E					E
ɛ		S		S								
ɪ							S					
i								E				
ɒ				S			E					S
ʊ	S							S				S
ʌ		S		S				S				S
u	S											
none												E

Table 7.7. Summary of significant differences between confusions for the English and Scots subjects. Where the English subjects make a confusion significantly more often than the Scots subjects (at a 5% level) there is an E, and for the opposite case there is an S.

Chapter 8

Experiment II: Setup and Raw Results

The Experiment II intelligibility test was designed to enable me to answer the questions raised at the beginning of Chapter 6. This chapter first describes the choice of ANNs whose outputs were used in the intelligibility test and the running of the intelligibility test. The chapter then lists the raw results which are used in Chapter 9 to answer the questions about the best choices of input and output representations, ANN training algorithms, ANN architectures and the training methods. Finally it describes a replication of the derivation of a relationship between the triphone errors and the intelligibility errors, using this new data.

8.1 Choosing the ANN Training Results to Use

In order to compare the effectiveness of the different input and output representations and the variations in training methodology, it was necessary to choose from each experimental setup the "best" performing ANN and then compare these with each other. Given the range of numbers of hidden nodes (typically 1, 2, 4, 6, 8, 10, 12, 14, 16, 18 and 20) and the range of random initial weights (5 different sets of initial weights for each network configuration) this gave up to 55 different sets of speech data to compare to find the best in each setup. Running intelligibility tests was not practical for this purpose, so the F1 formant errors were used to predict the intelligibility errors using Equation 7.1 as derived in Chapter 7 using the results of Experiment I.

The predicted intelligibility errors are subject to error, with confidence bounds as shown in Figure 7.7. For each experimental setup, the best ANN, its associated predicted intelligibility error and the number of ANNs with similar predicted intelligibility errors are shown in Table 8.1. The "best" choices from each setup were then included in the Experiment II intelligibility test. As can be seen from the table, many of the setups did not have one clear best ANN. If the predicted errors are similar then the real errors should also be similar. However, it is possible that if the "best" ANNs from two experimental setups have very similar intelligibility errors in the second test then different choices of the "best" ANNs would have resulted in a different ordering of the performance of the two setups.

The best ANNs for the setups using FFT coefficients as formant representations had very poor predicted intelligibility errors. Trad.FFTBark4 and Trad.-FFTBark2 were omitted from the final intelligibility tests, but the best example from Trad.FFT4 was used in order to give a final comparison for these sets of experiments. Trad.Tri.bp.end was also omitted due to the whole set being very similar to Trad.Tri.bp.

Speech created directly from the various output representations used to train the ANNs was also included in the intelligibility test. This allowed some idea of the limitations of the representations in representing vowel formant tracks to be determined. Speech synthesised from the original vowel formant tracks as extracted from the real speech data was also included in order to give some idea of the limitations of the rather crude speech synthesis method used. This gave a total of 18 sets of triphones to be used in the final synthesis. Each set contained 37 triphones, giving a total of 666 triphones in all.

ANN	Predicted Error	Similar predicted errors		
		5%	10%	25%
Trad.TRat.end.12.1	24.3	0	0	0
Trad.Tri.end.12.1	28.1	5	2	1
Sym.TRat.14.1	28.8	3	2	0
Trad.Tri.bp.10.0	30.9	2	2	2
Trad.Tri.bp.end.7	33.7	9	9	9
Trad.Tri.bp.4	33.7	7	5	4
Trad.TRat.bp.8	33.8	4	2	1
Cont.TRat.20.1	34.0	1	1	0
Trad.Tri.10.0	36.4	0	0	0
Trad.TRat.10.1	38.5	0	0	0
Trad.Poly.end.12.2	65.6	5	4	0
Trad.Poly.12.1	66.5	1	0	0
Trad.PolyBark.14.1	68.4	5	4	2
Trad.TDif.16.4	76.0	27	23	6
Trad.FFT4.20.0	184.4	4	2	1
Trad.FFTBark4.10.0	238.1	5	4	2
Trad.FFTBark2.14.1	243.8	2	1	1

Table 8.1. The ANN with best predicted intelligibility error for each experimental setup. The predicted intelligibility error is shown, as is the number of other ANNs with better than a 5%, 10% and 25% probability of being as good as the predicted best.

8.2 The Experimental Setup

Essentially the same experimental setup was used as in Chapter 7 for Experiment I. The same recording and play back facilities were used. The same choice of triphones were used, with the same possibilities for the responses.

There were 18 sets of 37 triphones, giving 666 in total. At approximately 4 seconds per triphone this would result in an experiment of about 45 minutes duration. This was too long, so the speech utterances were split into two parts, with subjects being given only one part. Each triphone set was split between the two subject groups, with 19 utterances going to one group and 18 to the other, and with the distribution of triphones being balanced between the groups. The distribution of triphones from each experiment into the two groups is shown in Table 8.2.

8.3 Raw Results

The raw intelligibility errors for the two groups of subjects are shown in Tables 8.3 and 8.4. Boxplots of the evaluation errors for each set of subjects are shown in Figures 8.1 and 8.2. The combined means and standard errors are shown in Figure 8.3.

The means and standard errors for the triphones in the test set and those in the traintest set are shown in Figure 8.4. A paired t-test shows that there is a very significant difference overall between the results for the test and traintest triphone sets. Breaking this down to individual ANNs, Trad.TRat.bp.8, Trad.TRat.-end.12.1, Trad.Tri.10.0, Cont.TRat.20.1, Trad.Poly.12.1, Trad.PolyBark.14.1, Trad.Poly.end.12.2 and Trad.FFT4.20.0 had significant differences at the 5% level between intelligibility errors on the test set and the traintest set. One peculiarity is that while the better ANNs did worse on the test set (previously unseen) than on the traintest set (a small subset of the training set), as would be expected, this was reversed for the more poorly performing ANNs.

Triphone	Trad.TRat.bp.8	Trad.TDif.16.4	Trad.Tri.bp.4	Trad.TRat.10.1	Trad.TRat.end.12.1	Sym.TRat.14.1	Cont.TRat.20.1	Trad.Tri.10.0	Trad.Tri.end.12.1	Trad.Tri.bp.10.0	Trad.FFT4.20.0	Trad.Poly.12.1	Trad.PolyBark.14.1	Trad.Poly.end.12.2	trk	tri	poly	fft
w-066_2	2	1	2	1	2	2	1	2	1	1	1	1	1	2	2	1	2	2
w-512_2	1	1	1	2	2	1	1	1	2	1	2	1	2	2	2	2	1	2
w-652_2	1	1	1	2	2	2	1	1	1	2	2	1	2	1	2	1	2	2
w-946_3	2	1	2	1	1	1	1	2	1	1	2	1	2	2	2	2	1	2
w1141_2	2	2	1	2	1	1	2	1	1	2	1	1	2	2	1	2	2	1
w1483_2	1	1	2	1	2	2	1	1	1	1	2	2	2	1	2	1	2	2
w1594_3	1	2	1	2	1	2	2	1	2	1	2	2	1	1	2	1	2	1
w1831_2	2	1	1	1	1	1	2	1	2	2	2	2	2	1	2	1	2	1
w2078_7	1	1	2	2	1	2	2	2	2	2	1	1	1	1	2	1	2	1
w2237_2	1	2	1	1	2	2	2	2	2	2	1	2	1	1	1	1	1	2
w2410_4	2	2	2	2	1	1	2	1	1	1	2	1	2	2	1	1	1	2
w3318_2	2	1	1	2	1	2	1	2	2	2	1	2	2	1	1	1	2	1
w3319_2	1	2	2	2	1	2	2	1	1	1	2	2	1	2	1	1	1	2
w3321_2	2	1	2	1	2	1	1	2	1	2	2	1	2	1	1	2	1	2
w3579_2	2	1	1	2	2	1	1	1	2	2	1	1	1	2	1	2	2	2
w4415_2	1	2	2	1	1	1	1	2	2	1	1	2	2	1	2	2	2	1
w4457_2	1	2	1	2	1	1	1	2	1	2	1	2	2	1	2	2	2	1
w-110_2	2	1	1	1	2	2	1	2	1	2	1	1	1	2	2	2	1	2
w1144_3	1	2	1	1	2	2	2	1	1	1	2	2	2	1	2	1	1	2
w1319_3	2	2	1	2	2	2	2	1	2	1	1	1	2	1	1	1	2	1
w1341_2	2	1	2	1	2	1	1	2	1	2	1	1	2	2	1	1	2	2
w1438_2	1	1	2	1	2	2	2	2	2	1	2	2	1	1	1	2	1	1
w1491_2	2	2	2	2	1	1	1	2	1	1	1	2	1	1	2	2	1	2
w1545_6	1	2	2	1	1	2	2	2	1	2	2	1	1	2	2	1	1	1
w1928_2	2	1	2	1	2	2	2	1	2	2	1	1	1	1	1	2	1	2
w2037_5	1	2	2	1	2	2	2	1	2	1	1	2	1	1	1	2	2	1
w2098_8	1	1	2	1	2	2	2	1	2	1	1	2	1	2	1	2	2	1
w3306_2	2	2	1	2	1	1	1	2	2	1	2	1	1	1	2	2	1	2
w3409_2	2	2	1	1	2	1	2	1	1	2	2	1	1	2	1	1	2	2
w3504_2	1	1	1	2	1	1	2	2	2	1	1	2	1	2	2	2	1	2
w3645_3	1	2	1	1	2	1	1	2	2	1	2	2	1	2	1	2	2	1
w3906_7	2	2	1	1	1	2	2	2	1	1	2	2	2	2	1	1	1	1
w3971_2	2	1	1	2	1	1	2	1	1	2	2	2	2	2	1	2	1	1
w3996_8	1	1	2	1	1	1	2	1	2	2	1	2	1	1	2	2	2	2
w4051_2	1	1	2	2	1	2	1	2	1	2	2	1	2	2	2	1	1	1
w4138_3	2	2	2	2	2	1	1	1	1	1	1	2	2	2	1	2	1	1
w4446_2	1	2	1	2	1	2	1	1	2	2	1	1	2	2	2	1	2	1

Table 8.2. The allocation of triphones from each experimental setup into the two groups of subjects (1 and 2) used in the Experiment II intelligibility test.

Subject	Trad.TRat.bp.8	Trad.TDif.16.4	Trad.Tri.bp.4	Trad.TRat.10.1	Trad.TRat.end.12.1	Sym.TRat.14.1	Cont.TRat.20.1	Trad.Tri.10.0	Trad.Tri.end.12.1	Trad.Tri.bp.10.0	Trad.FFT4.20.0	Trad.Poly.12.1	Trad.PolyBark.14.1	Trad.PolyBark.end.12.2	fft	poly	tri	trk	Mean
1-1	10.5	89.5	5.3	21.1	15.8	27.8	27.8	21.1	10.5	15.8	63.2	38.9	55.6	72.2	16.7	5.6	5.6	22.2	29.2
1-2	10.5	89.5	10.5	31.6	36.8	27.8	27.8	21.1	21.1	15.8	63.2	50.0	50.0	66.7	5.6	11.1	0.0	22.2	31.2
1-3	10.5	78.9	15.8	10.5	15.8	27.8	22.2	10.5	15.8	10.5	57.9	55.6	50.0	66.7	5.6	5.6	11.1	11.1	26.8
1-4	21.1	89.5	21.1	36.8	26.3	55.6	33.3	36.8	21.1	15.8	63.2	50.0	55.6	61.1	16.7	5.6	16.7	27.8	36.3
1-5	21.1	94.7	15.8	21.1	31.6	33.3	16.7	36.8	10.5	42.1	57.9	38.9	50.0	72.2	11.1	11.1	16.7	11.1	32.9
1-6	21.1	78.9	10.5	31.6	26.3	38.9	22.2	26.3	5.3	21.1	78.9	44.4	55.6	72.2	5.6	11.1	11.1	5.6	31.5
1-7	10.5	78.9	10.5	26.3	31.6	22.2	27.8	26.3	15.8	10.5	57.9	50.0	66.7	72.2	11.1	11.1	16.7	22.2	31.6
1-8	36.8	94.7	31.6	57.9	42.1	50.0	44.4	36.8	10.5	31.6	68.4	61.1	61.1	88.9	16.7	22.2	22.2	22.2	44.4
1-9	10.5	89.5	26.3	21.1	26.3	38.9	27.8	21.1	21.1	10.5	68.4	38.9	61.1	66.7	11.1	11.1	11.1	16.7	32.1
1-10	10.5	84.2	5.3	10.5	26.3	38.9	27.8	21.1	15.8	21.1	63.2	50.0	61.1	77.8	11.1	22.2	27.8	11.1	32.5
1-11	26.3	89.5	21.1	42.1	21.1	33.3	27.8	26.3	21.1	42.1	63.2	55.6	55.6	77.8	27.8	16.7	22.2	22.2	38.4
1-12	10.5	84.2	21.1	36.8	26.3	27.8	22.2	21.1	10.5	31.6	63.2	38.9	55.6	61.1	11.1	16.7	11.1	11.1	31.2
1-13	26.3	89.5	42.1	36.8	47.4	38.9	38.9	42.1	36.8	31.6	73.7	50.0	55.6	72.2	16.7	38.9	22.2	16.7	43.1
1-14	5.3	94.7	5.3	15.8	26.3	16.7	22.2	21.1	5.3	15.8	47.4	55.6	61.1	72.2	0.0	5.6	11.1	11.1	27.4
1-15	42.1	94.7	47.4	52.6	47.4	55.6	44.4	36.8	26.3	57.9	73.7	61.1	61.1	77.8	44.4	27.8	33.3	27.8	50.7
1-16	21.1	94.7	15.8	15.8	15.8	38.9	33.3	31.6	21.1	15.8	57.9	44.4	72.2	72.2	11.1	5.6	11.1	27.8	33.7
1-17	15.8	78.9	15.8	31.6	31.6	16.7	22.2	31.6	26.3	21.1	42.1	44.4	44.4	66.7	11.1	22.2	22.2	22.2	31.5
1-18	10.5	84.2	21.1	36.8	31.6	50.0	38.9	31.6	31.6	36.8	73.7	50.0	61.1	72.2	27.8	22.2	27.8	27.8	40.9
1-19	10.5	84.2	10.5	10.5	15.8	33.3	27.8	15.8	21.1	5.3	57.9	44.4	55.6	66.7	0.0	11.1	11.1	16.7	27.7
1-20	31.6	100.0	26.3	57.9	15.8	50.0	38.9	21.1	26.3	36.8	63.2	72.2	66.7	77.8	16.7	11.1	22.2	27.8	42.3
1-21	10.5	78.9	21.1	21.1	15.8	33.3	11.1	10.5	10.5	26.3	57.9	50.0	50.0	61.1	5.6	11.1	5.6	16.7	27.6
1-22	10.5	78.9	10.5	21.1	26.3	38.9	22.2	15.8	15.8	15.8	63.2	50.0	50.0	72.2	0.0	11.1	0.0	11.1	28.5
Mean	17.5	87.3	18.7	29.4	27.3	36.1	28.5	25.6	18.2	24.2	62.7	49.7	57.1	71.2	12.9	14.4	15.4	18.7	34.2

Table 8.3. Intelligibility errors per subject for set one of the Experiment II intelligibility test.

Subject	Trad.TRat.bp.8	Trad.TDif.16.4	Trad.Tri.bp.4	Trad.TRat.10.1	Trad.TRat.end.12.1	Sym.TRat.14.1	Cont.TRat.20.1	Trad.Tri.10.0	Trad.Tri.end.12.1	Trad.Tri.bp.10.0	Trad.FFT4.20.0	Trad.Poly.12.1	Trad.PolyBark.14.1	Trad.PolyBark.end.12.2	fft	poly	tri	trk	Mean
2-1	33.3	83.3	44.4	27.8	22.2	21.1	26.3	38.9	22.2	22.2	83.3	78.9	73.7	63.2	21.1	26.3	31.6	15.8	40.9
2-2	27.8	83.3	38.9	22.2	27.8	5.3	42.1	33.3	11.1	33.3	94.4	78.9	78.9	68.4	21.1	10.5	26.3	5.3	39.4
2-3	27.8	77.8	27.8	38.9	27.8	10.4	21.1	33.3	16.7	27.8	88.9	84.2	84.2	68.4	26.3	10.5	47.4	10.5	40.5
2-4	27.8	83.3	38.9	16.7	16.7	5.3	42.1	33.3	22.2	22.2	88.9	68.4	68.4	63.2	15.8	10.5	21.1	10.5	36.4
2-5	33.3	77.8	33.3	44.4	22.2	31.6	26.3	44.4	27.8	27.8	55.6	84.2	78.9	68.4	21.1	21.1	26.3	10.5	40.8
2-6	27.8	83.3	27.8	33.3	27.8	21.1	36.8	22.2	11.1	16.7	83.3	84.2	73.7	68.4	10.5	15.8	21.1	5.3	37.2
2-7	33.3	66.7	61.1	44.4	38.9	21.1	36.8	33.3	11.1	33.3	94.4	84.2	78.9	68.4	26.3	10.5	31.6	15.8	43.9
2-8	27.8	83.3	50.0	27.8	27.8	21.1	31.6	33.3	22.2	22.2	88.9	68.4	84.2	68.4	21.1	5.3	26.3	10.5	40.0
2-9	16.7	88.9	16.7	27.8	16.7	10.5	36.8	27.8	33.3	33.3	77.8	89.5	84.2	68.4	21.1	15.8	26.3	21.1	39.6
2-10	22.2	83.3	33.3	38.9	22.2	5.3	31.6	16.7	50.0	27.8	83.3	78.9	84.2	63.2	15.8	10.5	21.1	10.5	38.8
2-11	33.3	83.3	44.4	27.8	33.3	26.3	42.1	38.9	11.1	33.3	88.9	84.2	84.2	68.4	21.1	15.8	31.6	10.5	43.3
2-12	27.8	83.3	50.0	27.8	38.9	26.3	31.6	38.9	33.3	33.3	77.8	94.7	73.7	68.4	21.1	15.8	21.1	5.3	42.7
2-13	22.2	72.2	22.2	27.8	27.8	15.8	10.5	27.8	22.2	33.3	72.2	84.2	89.5	57.9	21.1	15.8	21.1	5.3	36.0
2-14	11.1	94.4	38.9	27.8	5.6	26.3	36.8	22.2	38.9	11.1	88.9	73.7	73.7	68.4	21.1	15.8	26.3	10.5	38.4
2-15	22.2	88.9	22.2	38.9	11.1	10.5	31.6	22.2	5.6	33.3	88.9	68.4	84.2	63.2	21.1	5.3	26.3	10.5	36.4
2-16	38.9	83.3	27.8	33.3	11.1	10.5	21.1	27.8	16.7	38.9	72.2	78.9	78.9	68.4	31.6	15.8	21.1	10.5	38.2
2-17	27.8	72.2	44.4	44.4	27.8	21.1	31.6	33.3	16.7	27.8	72.2	78.9	68.4	73.7	31.6	5.3	47.4	10.5	40.8
2-18	33.3	83.3	33.3	22.2	27.8	21.1	26.3	22.2	27.8	22.2	72.2	78.9	78.9	63.2	21.1	15.8	31.6	5.3	38.1
2-19	27.8	77.8	38.9	44.4	22.2	5.3	15.8	33.3	16.7	27.8	88.9	73.7	63.2	68.4	21.1	15.8	36.8	5.3	37.9
2-20	16.7	77.8	27.8	27.8	11.1	10.5	21.1	22.2	5.6	22.2	88.9	84.2	68.4	63.2	15.8	5.3	21.1	5.3	33.0
2-21	22.2	88.9	33.3	22.2	22.2	15.8	36.8	27.8	22.2	33.3	88.9	78.9	78.9	68.4	31.6	10.5	31.6	10.5	40.2
2-22	22.2	77.8	50.0	38.9	5.6	10.5	21.1	33.3	11.1	22.2	83.3	89.5	84.2	63.2	21.1	10.5	21.1	5.3	37.3
Mean	26.5	81.6	36.6	32.1	22.5	16.0	29.9	30.3	20.7	27.5	82.8	80.4	78.0	66.5	21.8	12.9	28.0	9.6	39.1

Table 8.4. Intelligibility errors per subject for set two of the Experiment II intelligibility test.

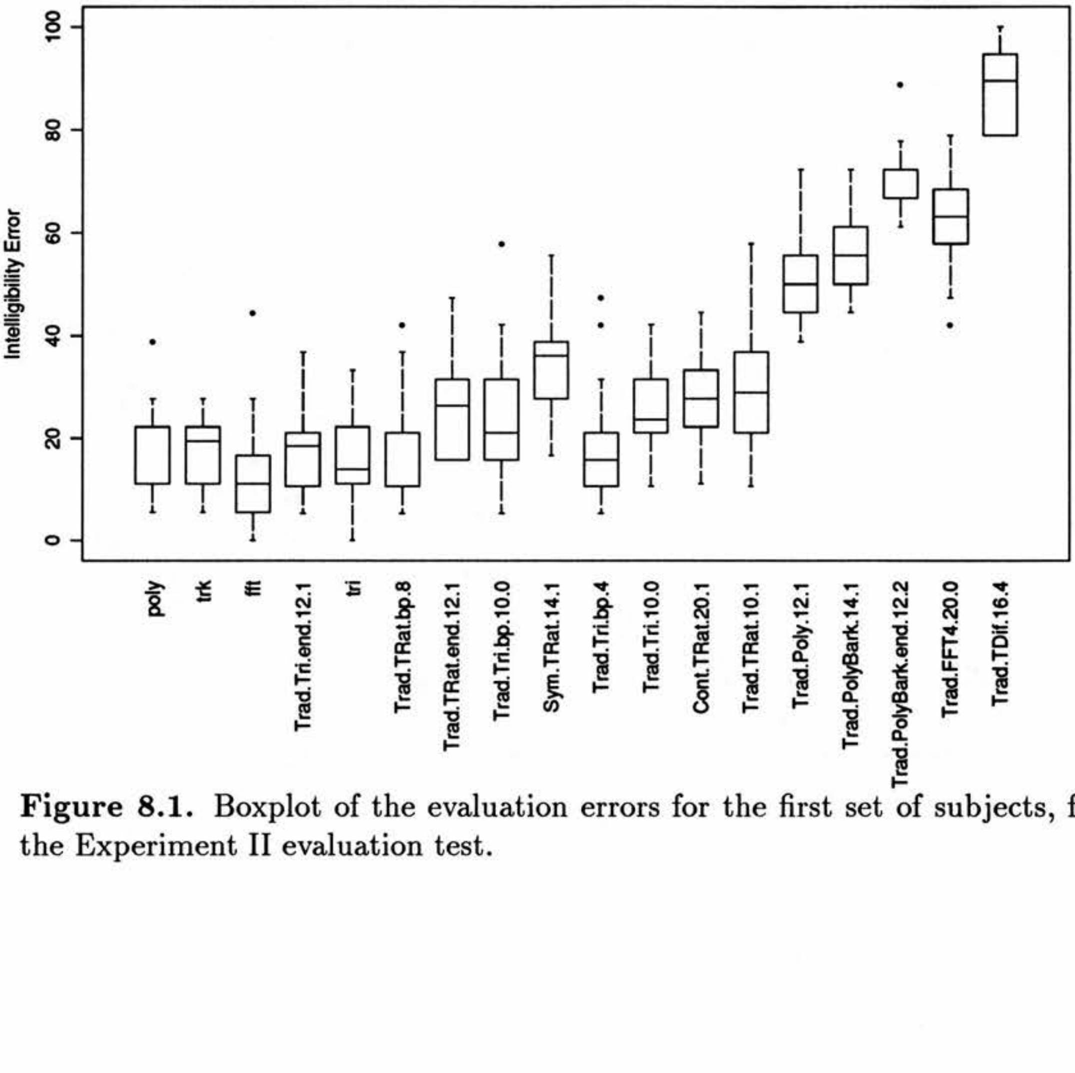

Figure 8.1. Boxplot of the evaluation errors for the first set of subjects, from the Experiment II evaluation test.

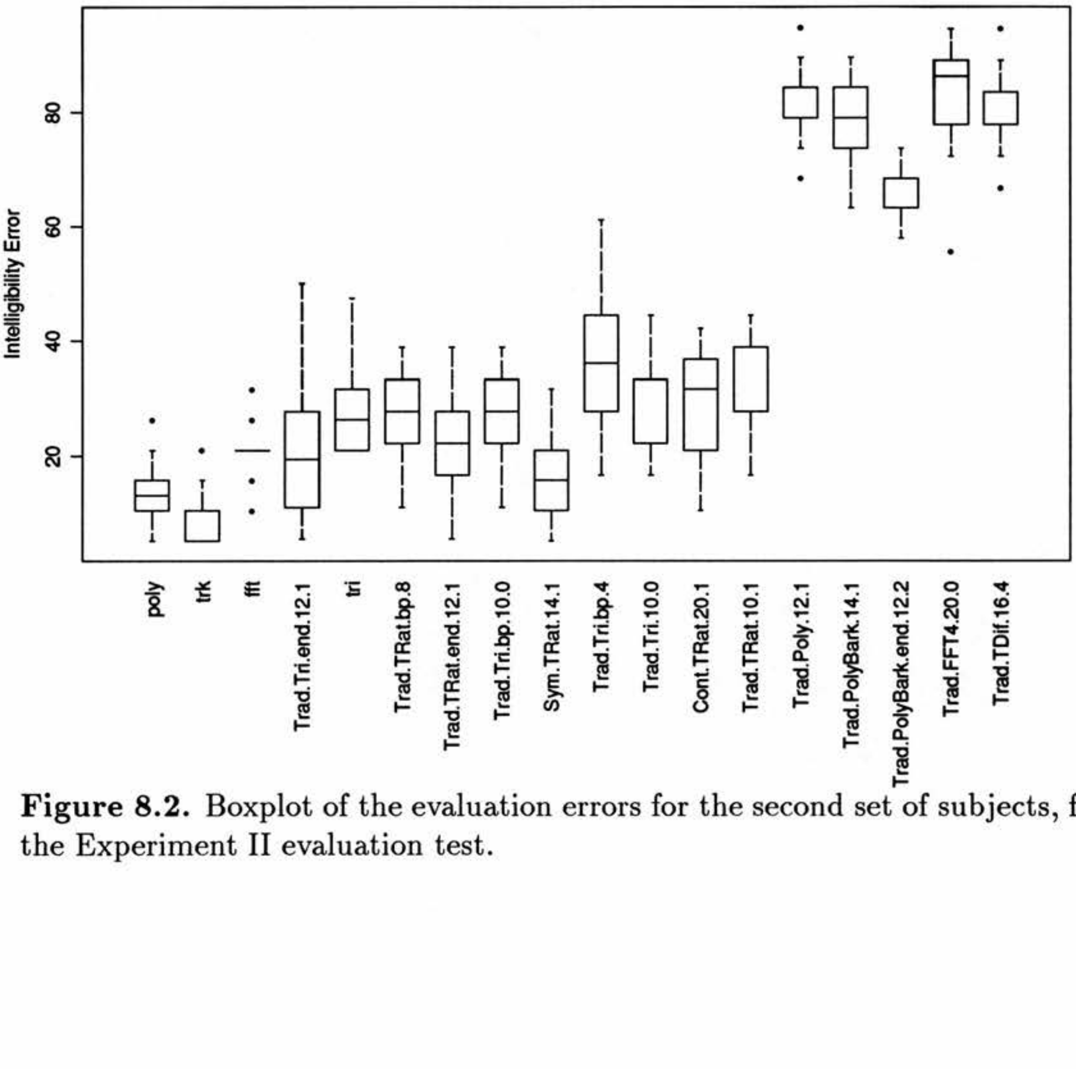

Figure 8.2. Boxplot of the evaluation errors for the second set of subjects, from the Experiment II evaluation test.

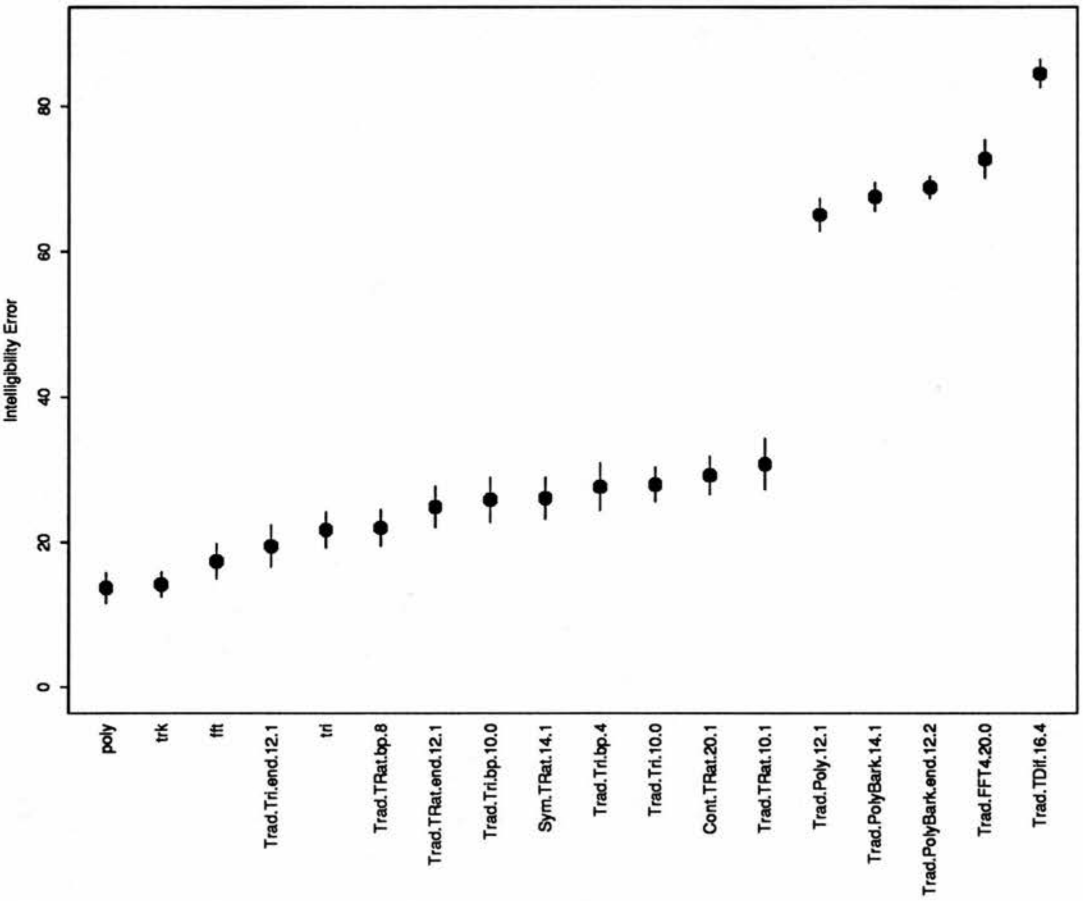

Figure 8.3. Plot of the combined mean evaluation errors over both sets of subjects, from the Experiment II evaluation test. The lines represent one standard error each side of the mean.

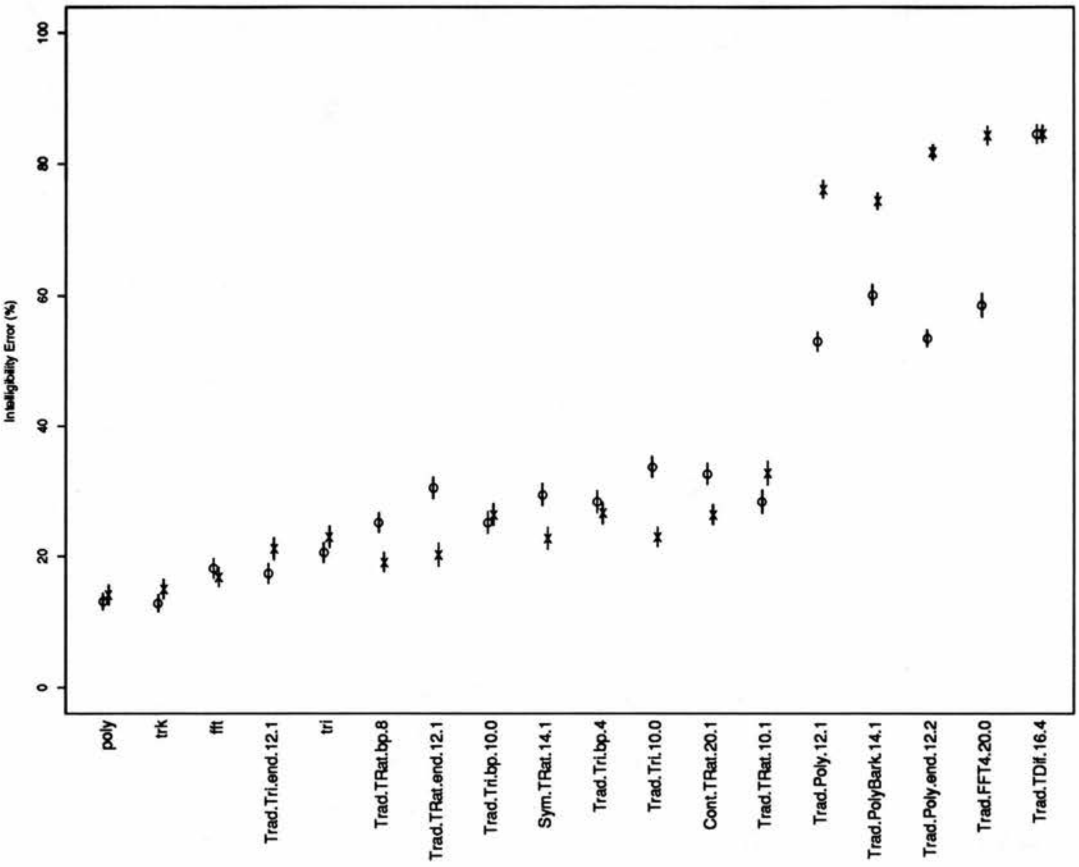

Figure 8.4. Plot of the combined mean evaluation errors over both sets of subjects, showing the test (O) and traintest (X) results separately. From the Experiment II evaluation test. The lines represent one standard error each side of the mean.

8.4 Relating Formant Track Errors and Intelligibility Errors Using the Experiment II Data

Chapter 7 derived a relationship between formant track errors and intelligibility errors, using the Experiment I results. I replicated the derivation using the results from Experiment II. The main difference between the derivations was that with Experiment I it was possible to use the results per subject. The Experiment II data was split into two subject groups and I was forced to use the mean intelligibility errors and estimated variances, resulting in poorer confidence bounds than might have been obtained with a single subject group.

The highest correlation with intelligibility error for untransformed formant errors was for F1, at 0.93, compared with 0.90 with the Experiment I data. This gave the regression equation

$$\text{Intelligibility Error} = -3.09 + 67.3(\text{F1 error}) \tag{8.1}$$

which is a flatter line than that obtained for Experiment I (see Equation 7.1).

F1^2 had a higher correlation with the intelligibility errors of 0.98, and produced a significantly better fit. No other transformations or combinations of transformations made a significant improvement. The fit for F1^2 was

$$\text{Intelligibility Error} = 12.0 + 62.6(\text{F1 error})^2 \tag{8.2}$$

which gave the regression line shown in Figure 8.5 with 90% confidence bands. A plot of confidence bounds against predicted intelligibility error is shown in Figure 8.6. The bands are similar to those in Figure 7.7, but the points used in the derivation span a wider range.

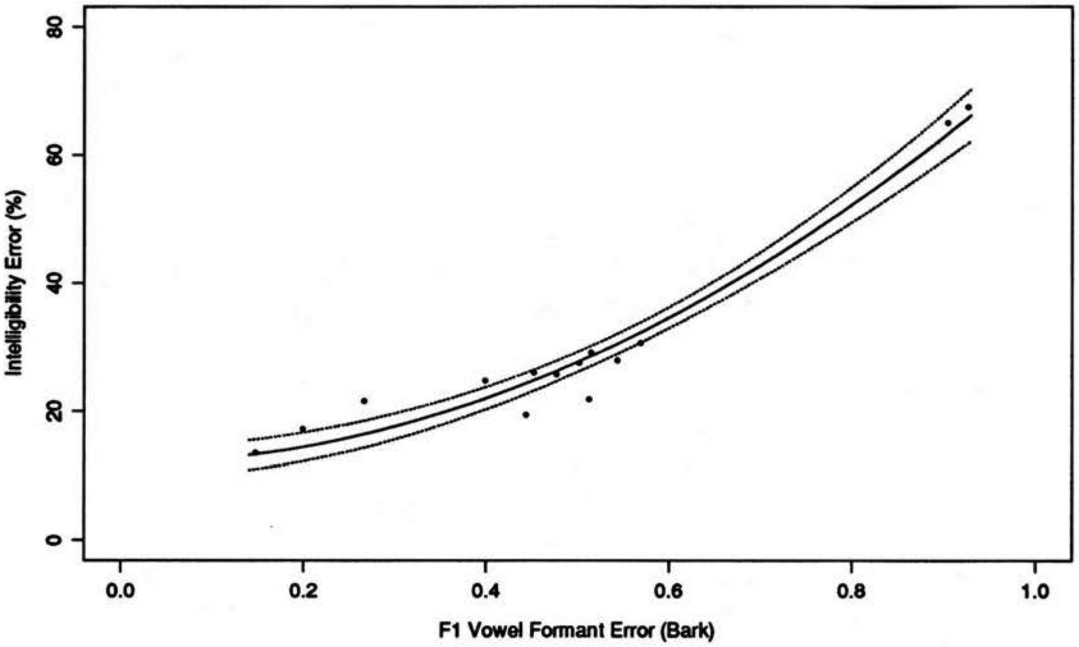

Figure 8.5. The regression line for the square of the F1 formant error, predicting intelligibility error (for Experiment II), with 90% confidence bands shown.

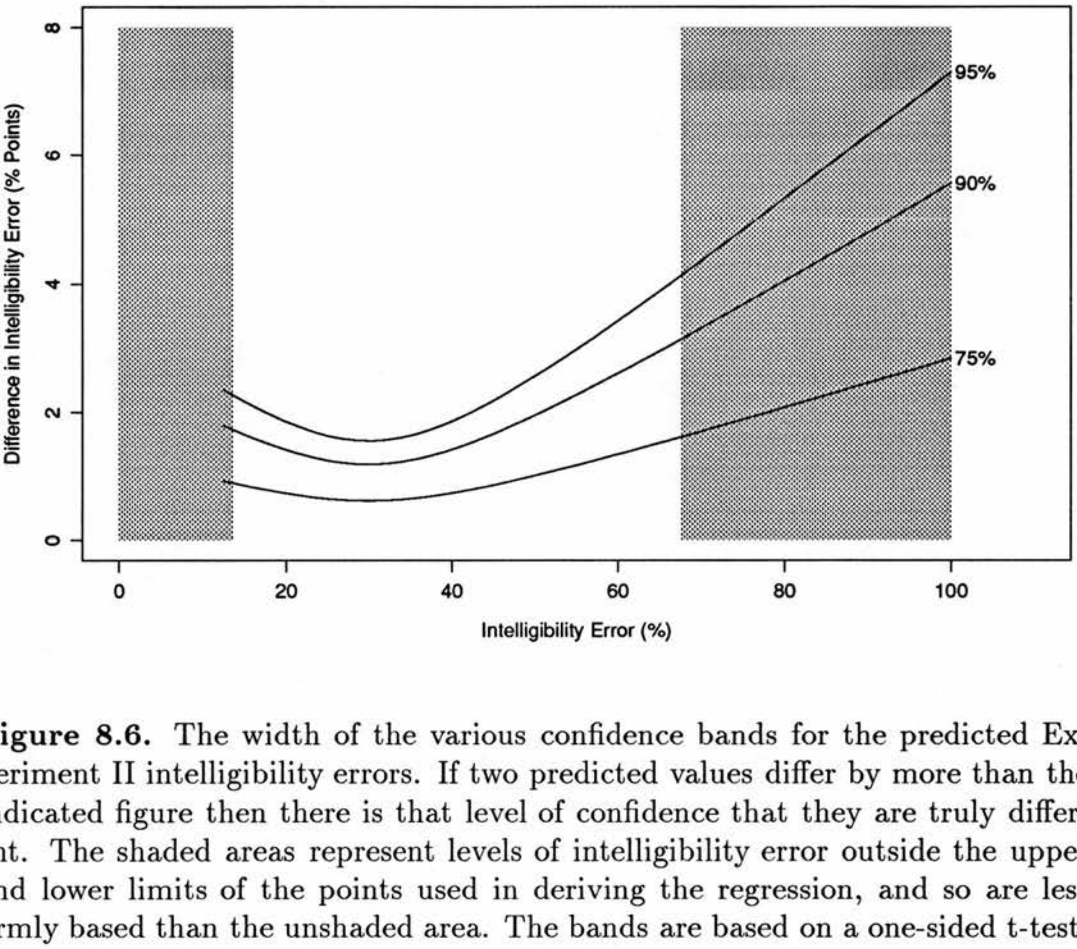

Figure 8.6. The width of the various confidence bands for the predicted Experiment II intelligibility errors. If two predicted values differ by more than the indicated figure then there is that level of confidence that they are truly different. The shaded areas represent levels of intelligibility error outside the upper and lower limits of the points used in deriving the regression, and so are less firmly based than the unshaded area. The bands are based on a one-sided t-test.

Chapter 9

Experiment II: Evaluation of Representations and Methods

9.1 Question 1: Ability of ANNs to Produce Vowel Formant Tracks

It is clear from the raw results of Experiment II (see Figures 8.3 and 8.4) that some of the ANNs learned to produce vowel formant tracks whose intelligibility was only a few percentage points worse than the intelligibility of the original vowel formant tracks used in training. The errors were fairly high, around 20-25%, but so were those of the original tracks when resynthesised. This reflects the inadequacies of the synthesis process used, especially the concatenation of the consonant tokens onto the synthesised vowel token. Interestingly, the best ANN used in the Experiment II intelligibility test (Trad.Tri.end.12.1) had a lower overall intelligibility error than the resynthesis of the *Traditional* output representation (extracted from the original speech data) used in training it. I think that this is due to the regularised nature of the ANN output (see Chapter 10). That is, the original speech contains the natural variation of human utterances, and is affected by context beyond the triphone, whereas the ANN speech is an "average" utterance of the vowel in the context of the triphone alone.

9.2 Question 2: Comparing Different Input Representations

9.2.1 The Performance of Three Input Representations

The three forms of input representation to be compared are described in Section 6.4. They are the *Traditional* representation, the *Continuous* representation (which codes place of articulation, backness and height as continua instead of using binary values) and the *Symbolic* representation which uses a 1-of-n coding which does not incorporate any description of phonetic or phonemic features. The output representation used is that I have called the *Tri-ratio* representation. The Bark scaled F2 and F3 frequencies are represented as ratios to the Bark scaled F1 frequency. The ANNs were trained using the conjugate gradient method and cross-validation was used to determine the end of training. The best ANNs using these three sets of representations were Trad.TRat.10.1, Cont.TRat.20.1 and Sym.TRat.14.1. The test set and traintest set intelligibility errors for these three ANNs and their associated standard errors are shown in Figure 9.1.

The probabilities of the ANNs having the same performance was calculated using a paired t test on the intelligibility errors per subject. The results for the test set are shown in Table 9.1. There is no clear best input representation. The *Continuous* representation seems to be less similar to the other two, but not at a significant level.

The results for the traintest set are shown in Table 9.2. The *Traditional* representation did significantly worse than the *Symbolic* and *Continuous* representations.

From this it seems that the *Traditional* representation may have resulted in slightly poorer performance on the training set, but that the generalisation to new triphones was no worse than the others.

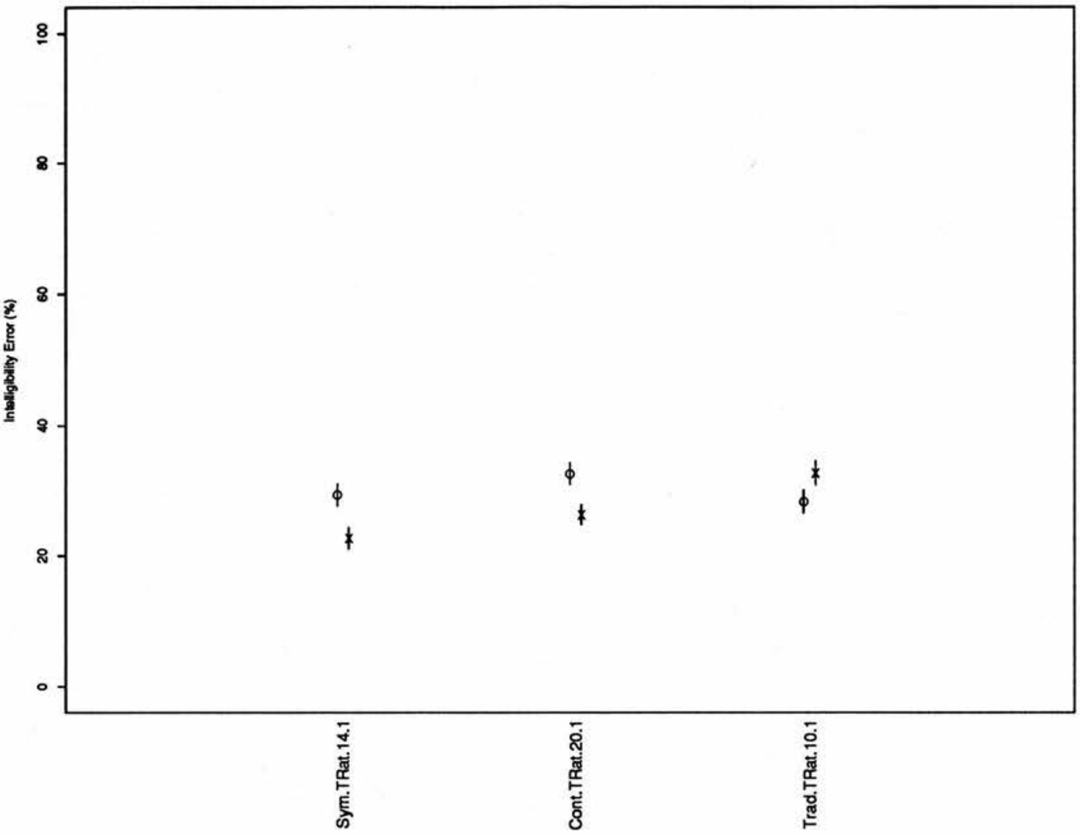

Figure 9.1. The test set (O) and traintest set (X) intelligibility errors of the best ANNs using the *Traditional*, *Continuous* and *Symbolic* input representations and the *Tri-ratio* output representation. The bars show the standard errors.

	Symbolic	Continuous	Traditional
Symbolic	—	0.239	0.945
Continuous	0.239	—	0.172
Traditional	0.945	0.172	—

Table 9.1. The probabilities of the best ANNs using the three types of input representation having the same performance on the test triphone set.

	Symbolic	Continuous	Traditional
Symbolic	—	0.625	0.007
Continuous	0.625	—	0.002
Traditional	0.007	0.002	—

Table 9.2. The probabilities of the best ANNs using the three types of input representation having the same performance on the traintest triphone set.

9.2.2 Generalising to a New Vowel

The test set of triphones contains the vowel /u/ (in the triphone /but/), which does not appear in the training set. This provides an interesting test of the generalisation resulting from the use of features to represent phonemes in the input representations. The *Traditional* and *Continuous* input representations both use the same feature representation of vowels (see Table 6.2). They differ in the feature representations used for consonants (see Tables 6.1 and 6.3). As can be seen in Table 9.3, these representations do result in the vowel being perceived as a /u/. My conclusion is that the ANNs really do use the information encoded in the phonetic representations about the acoustic nature of the vowel and that the ANNs can produce a new vowel on the basis of the phonetic features.

	u	ɜ	ɑ
Traditional	91%	9%	
Continuous	100%		
Symbolic	18%	77%	5%

Table 9.3. The classifications of the vowel in the triphone /but/ synthesised using formant tracks created by ANNs using the *Traditional*, *Continuous* and *Symbolic* input representations. Results are from Experiment II, and are for 22 subjects.

The *Symbolic* input representation uses a one-of-n coding to represent the vowels (see Table 6.5). This does not encode any phonetic information. The input node representing /u/ will not have been set to any value other than zero in training, so the ANN should have totally disregarded that input node. Therefore the /u/ input representation corresponds to turning off all the other nodes in the vowel section of the input. The expected outcome might be a neutral or average vowel (in the context), possibly perceived as a schwa, although the duration of the vowel is longer than might be expected for a schwa. In the intelligibility tests, the possible response words for each utterance did not include schwa, as it was not possible to construct CVC words containing schwa. Table 9.3 shows that 77% of subjects classified the vowel as a /ɜ/ which has a similar vowel quality to schwa, but whose expected duration more closely matches the utterance in question. Most of the remaining subjects classified the vowel as a /u/. I think that this is probably due to an expectation created by hearing a number of other utterances which were clearly /but/ in the same rhyme test. There were three different triphones with a /b_t/ context presented in the rhyme tests — /but/, /bɪt/ and /bat/ . The latter two triphones would be distinctly shorter than /but/, making it the likely choice by subjects who were unsure of an utterance's identity.

The vowel formant tracks for the original speech, the *Tri* representation extracted from the original speech and the tracks produced by ANNs using the *Traditional*, *Continuous* and *Symbolic* input representation are shown in Figure 9.2. The F1 track is reproduced accurately by the ANNs, but the F2 tracks are lower than in the original speech and the F3 tracks are all different in shape. The *Sym* formant errors are larger than those for *Trad* and *Cont* on F1 and F3.

The representation of vowels in the *Traditional* and *Continuous* representations is in terms of three features — backness, height and rounding. The back feature takes the values {0.0, 0.5, 1.0} and the height feature takes the values {0.0, 0.33, 0.67, 1.0}. It is possible that the performance in producing the vowel /u/ might be improved by adjusting the values of the back and height features associated with it (currently both 1.0). It might even be possible to incrementally adjust values of these features to improve performance on all vowels.

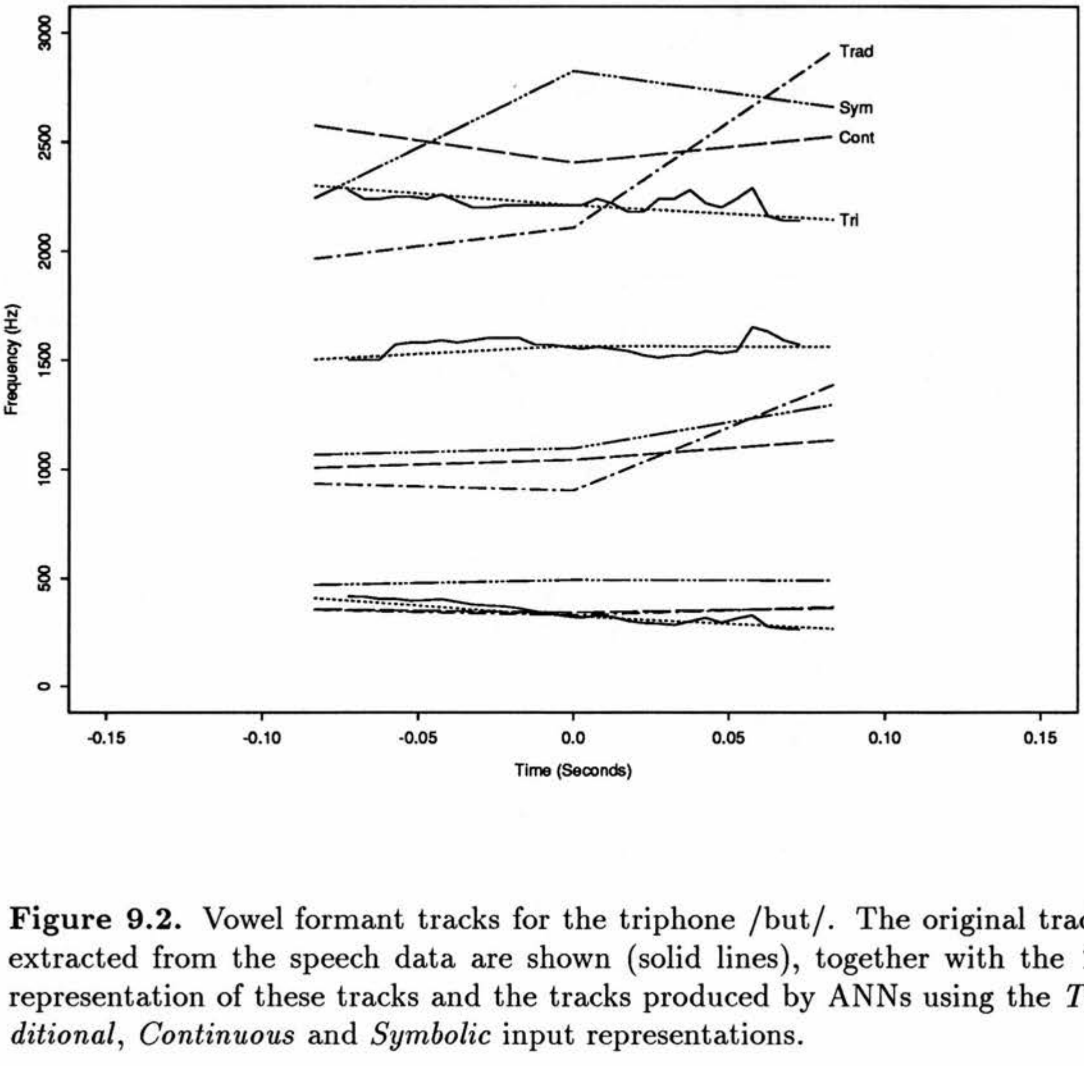

Figure 9.2. Vowel formant tracks for the triphone /but/. The original tracks extracted from the speech data are shown (solid lines), together with the *Tri* representation of these tracks and the tracks produced by ANNs using the *Traditional*, *Continuous* and *Symbolic* input representations.

9.2.3 Questions 2 and 8: Input Representations and Hidden Nodes

The number of hidden nodes necessary to accurately learn the training data and produce good generalisation may provide some insight into the "difficulty" of learning the mapping between input and output data for the representations used. For more discussion, see Section 6.6.4.

Looking at the test set formant track errors plotted against the number of hidden nodes of the ANNs for the three input representations no pattern is apparent. Figure 9.3 shows the errors for the *Symbolic* input representation. The other two ANNs have very similar graphs. The situation is even worse if we look at the patterns of predicted intelligibility error as we often cannot unequivocally choose a best ANN. Figure 9.4 shows the graph for the ANNs using the *Symbolic* input representation[1].

There seems to be no clear relationship between number of hidden nodes and performance for the three sets of ANNs looked at, and no clear differences between the sets. Therefore nothing can be concluded about the power of ANN necessary to use the different representations.

9.3 Question 3: Comparing the Representational Capabilities of the Output Representations

The three output representations used (*Tri, Polynomial* and *FFT*) differ in their ability to represent vowel formant tracks. The Experiment II intelligibility test included triphones produced using the original vowel formant tracks, and the three output representations extracted from these tracks. The results can be seen in Figure 8.3. The best intelligibility came from the *Polynomial* representation, followed by the original tracks, the *FFT* representation and finally the

[1] These two figures are very similar, but the predicted intelligibility error is based only on the F1 errors, not the total errors.

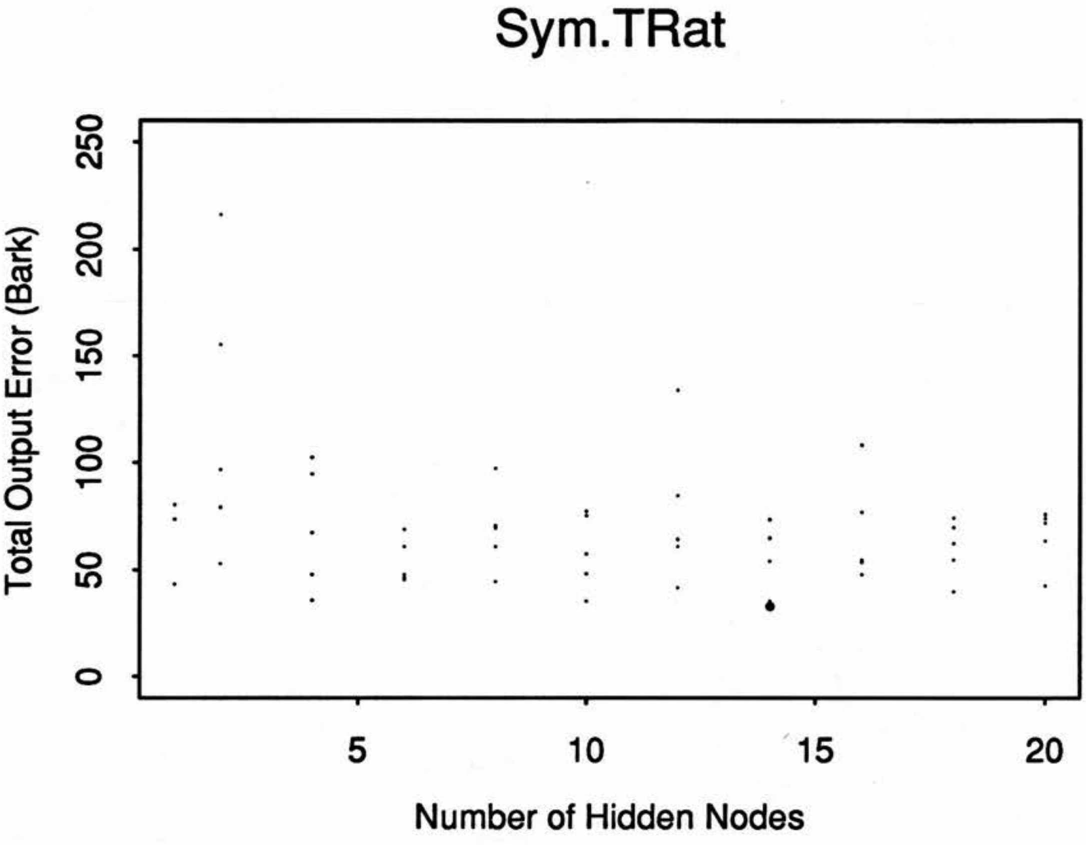

Figure 9.3. The output errors for the ANNs trained using the *Symbolic* input representation and the *Tri-ratio* output representation. The best value is marked with a larger dot.

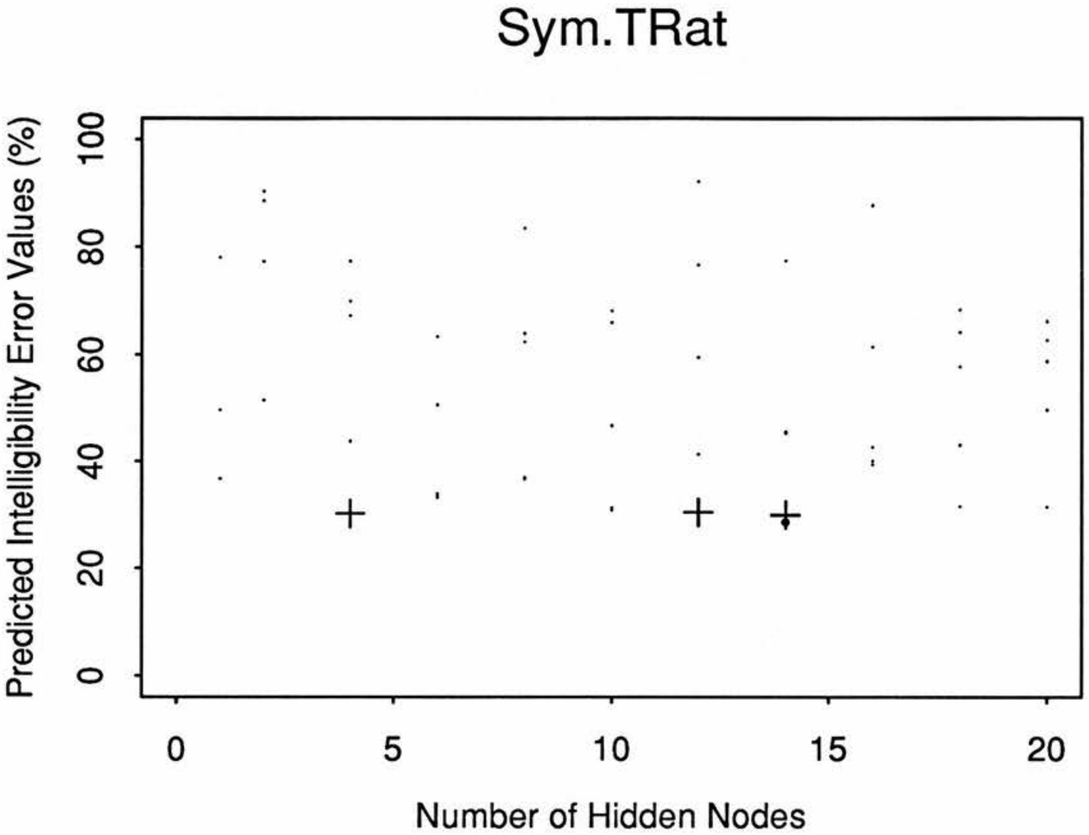

Figure 9.4. The predicted intelligibility errors for the ANNs trained using the *Symbolic* input representation and the *Tri-ratio* output representation. The best predicted value is marked with a larger dot and the values having a greater than 5% chance of actually being better than than that value are marked with a cross.

Tri representation. However, the *Polynomial* representation was not significantly better than the original tracks. The *FFT* representation was significantly worse than the original tracks only at the 10% level. The *Tri* representation was significantly worse than everything else at the 0.1% level. The results of a paired t-test on the intelligibility errors per subject comparing the representations are shown in Table 9.4.

	Polynomial	Original	FFT	Tri
Polynomial	1.00	0.74	0.02	0.00
Original	0.74	1.00	0.07	0.00
FFT	0.02	0.07	1.00	0.00
Tri	0.00	0.00	0.00	1.00

Table 9.4. The probabilities of the *Polynomial* representation, the original formant tracks, the *FFT* representation and the *Tri* representation having the same real intelligibility errors on the test triphone set.

9.4 Question 4: Comparing Different Output Representations

9.4.1 The Different Output Representations

The different output representations to be compared are described in Section 6.5. For each output representation the *Traditional* input representation was used. The conjugate gradient training algorithm was used with the end of training determined by cross-validation.

There are three variants of the *Tri* representation, where each vowel formant track is described by a triple of values — at the beginning, centre and end of the track. In all these cases the formant frequencies were Bark scaled. The *Tri-ratio* representation represented the F2 and F3 values as ratios to the F1 values. The *Tri-difference* representation represented the F2 and F3 formant frequencies as differences from the first formant and the *Tri-plain* representation did no special processing. In all cases where values were not automatically mapped onto [0, 1] by the processing, this was done as a final stage. The best ANNs for these three representations were Trad.TRat.10.1, Trad.TDif.16.4 and Trad.Tri.10.0 respectively.

An alternative representation was to represent each vowel formant as a polynomial curve and to use the polynomial coefficients as the output representation. The two variants of this used were polynomials of order 2 with no Bark scaling of the vowel formant tracks and polynomials of order 2 with Bark scaling. The best ANNs for these two representations were Trad.Poly.12.1 and Trad.PolyBark.14.1 respectively.

The final type of representation used were Fourier transforms of the vowel formant tracks. The Fourier coefficients were used as the representations, with the complex values being represented by two real numbers. The three types of representation used were four Fourier coefficients, four Fourier coefficients of Bark scaled tracks and two Fourier coefficients of Bark scaled tracks. The best ANNs for these three representations were Trad.FFT4.20.0, Trad.FFTBark4.10.0 and Trad.FFTBark2.14.1. respectively. However, since the predicted intelligibility errors for these three ANNs were so poor, only Trad.FFT4.20.0 was used in Experiment II.

The intelligibility of the basic representations used was also tested. That is, utterances were synthesised from the representations used in training the ANNs and tested. The vowel formant tracks extracted from the original recorded speech, upon which everything else was based, were also used to create utterances which were tested. This should give some idea of the limits of performance possible using these representations and how near the ANNs get to reaching the best possible performance. The basic representations are named *Tri*, *Poly* and *FFT*, with the original vowel formant tracks named *Trk*.

9.4.2 Accuracy of Learning the Training Set

I first looked at how well the three output representations *Tri-plain*, *Poly* and *FFT4* are learnt by the ANNs, examining the errors for the output node corresponding to the frequency value of the centre of the first formant. However, the magnitude of the errors must be judged relative to the distribution of the target values for the output nodes. An error of 0.1 is far more serious if all the output values for the training set are in the region [0.49, 0.51] than if they cover the region [0.1, 0.9].

Given the shape of the error distributions and the output value distributions it seems reasonable to scale the output errors by the standard deviations of the output values per node. This should give a clearer view of how well the training set output values were learnt. Histograms of the scaled values for the nodes determining the frequency value of the centre of the first formant are shown in Figure 9.5.

9.4.3 Accuracy of Production of the Traintest and Test Triphones

The intelligibility scores are based not on the whole corpus of speech used in training, but on 20 triphones chosen from the training set (called the traintest set) and on 17 triphones not included in the training set comprising a test set. The intelligibility errors for these two sets are shown in Figure 9.6. While the intelligibility errors for the representations themselves are similar for both the traintest and test sets, there are significant differences between the intelligibility errors for the speech produced by the ANNs. In particular, Trad.Tri.10.0, Trad.Poly.12.1, Trad.PolyBark.14.1, Trad.FFT4.10.0 and Trad.Poly.12.1 had significantly different results for the test and traintest sets.

The probabilities of the best ANNs for each output representation and the original output representations having the same performance, calculated using a paired t-test on the intelligibility errors per subject is shown in Table 9.5 for the

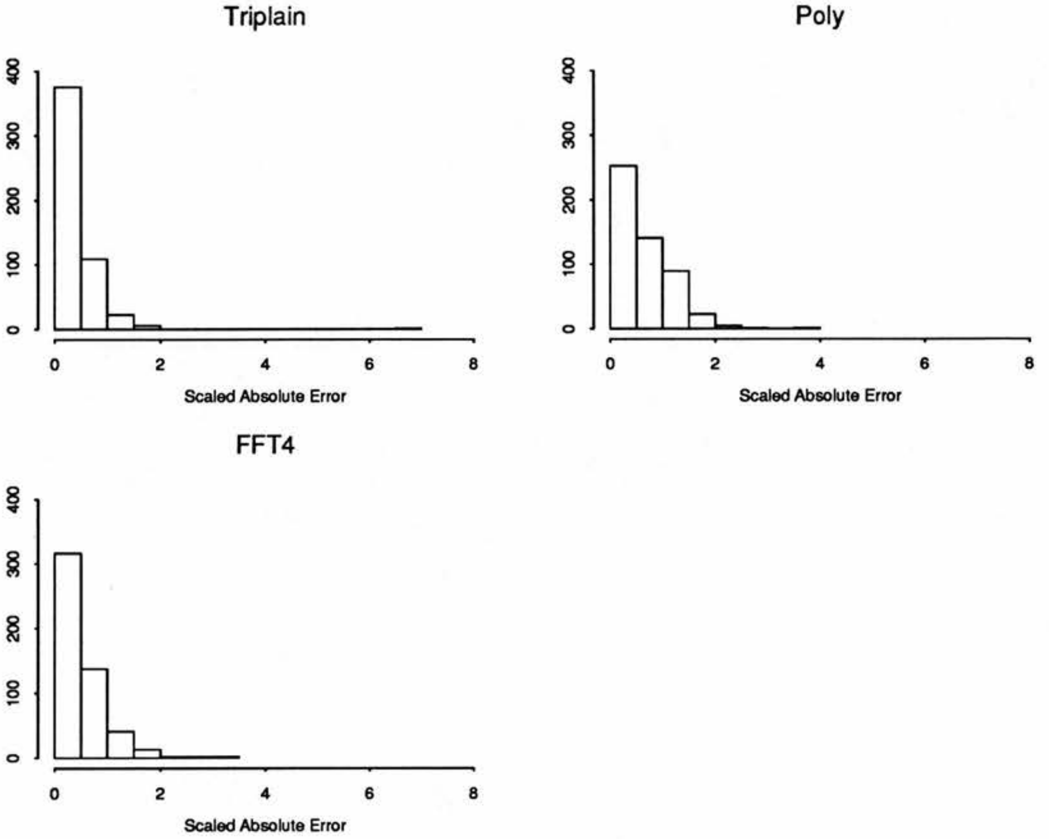

Figure 9.5. The scaled errors on the output node determining the frequency of the centre of the first formant of the best ANN trained to produce the *Triplain*, *Polynomial* and *FFT4* representations of the training data. The errors are scaled by dividing by the standard deviation of the output node values for the representation.

test set and Table 9.6 for the traintest set.

For the test set there is no significant difference in the intelligibility errors for the original vowel formant tracks and the *Polynomial* and *FFT* representations. The *Tri* representation does slightly worse, being significantly different from the tracks and the *FFT* representation at the 5% level and different from the *Polynomial* representation at the 10% level. It may seem odd that the 10% significant difference here is for the two representations with the most difference between the mean intelligibility errors, but the errors per subject must have followed more similar patterns than the others.

The next best performers on the test set are the ANNs using the *Tri-ratio* and *Tri-plain* output representations. Trad.TRat.10.1 and Trad.Tri.10.0 are significantly different from all other ANNs and the representations at the 1% level, but are not significantly different from each other.

The ANNs using *Polynomial*, *PolyBark* and *FFT4* output representations are significantly worse. The *Polynomial* and *PolyBark* ANNs are significantly different at the 10% level, with Bark scaling resulting in worse performance. The *Tri-difference* representation leads to the worst performing ANN of all.

Looking at the traintest set, the intelligibility errors for the original vowel formant tracks, the *Polynomial* and the *FFT* representations are even more similar than for the test set. The ANN trained using the *Tri-plain* output representation produced speech as intelligible as the *Tri* representation itself, but significantly worse than the other representations (5% level). The *Tri-ratio* trained ANN was the next best performer, clearly different from all others. Much worse were the *Polynomial* and *PolyBark* trained ANNs, with the Bark scaling the worst of the pair at 10% level significance. The *FFT4* and *Tri-difference* trained ANNs had the same level of performance, significantly worse than anything else.

Histograms of scaled error values for the nodes corresponding to the centre of the first formant are shown in Figures 9.7 and 9.8.

It seems clear that the error values for the output nodes representing the frequency of the centre of the first formant are worse where the intelligibility

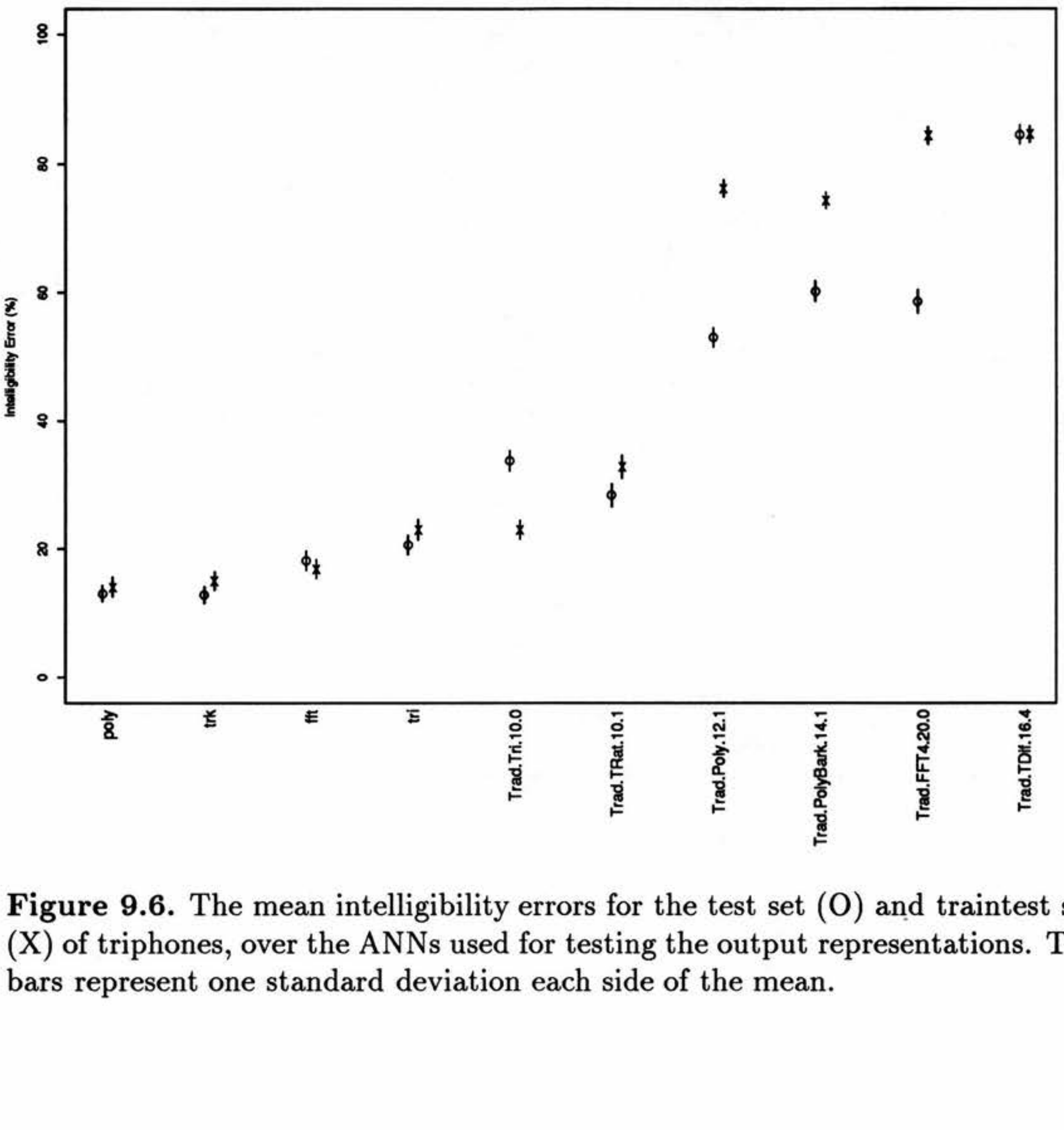

Figure 9.6. The mean intelligibility errors for the test set (O) and traintest set (X) of triphones, over the ANNs used for testing the output representations. The bars represent one standard deviation each side of the mean.

	poly	trk	fft	tri	Trad.Tri.10.0	Trad.TRat.10.1	Trad.Poly.12.1	Trad.PolyBark.14.1	Trad.FFT4.20.0	Trad.TDif.16.4
poly	—	0.44	0.79	0.06	0.00	0.00	0.00	0.00	0.00	0.00
trk	0.44	—	0.55	0.04	0.00	0.00	0.00	0.00	0.00	0.00
fft	0.79	0.55	—	0.01	0.00	0.00	0.00	0.00	0.00	0.00
tri	0.06	0.04	0.01	—	0.00	0.01	0.00	0.00	0.00	0.00
Trad.Tri.10.0	0.00	0.00	0.00	0.00	—	0.13	0.00	0.00	0.00	0.00
Trad.TRat.10.1	0.00	0.00	0.00	0.01	0.13	—	0.00	0.00	0.00	0.00
Trad.Poly.12.1	0.00	0.00	0.00	0.00	0.00	0.00	—	0.07	0.26	000
Trad.PolyBark.14.1	0.00	0.00	0.00	0.00	0.00	0.00	0.07	—	0.94	0.00
Trad.FFT4.20.0	0.00	0.00	0.00	0.00	0.00	0.00	0.26	0.94	—	0.00
Trad.TDif.16.4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	—

Table 9.5. The probabilities of the best ANNs using the different types of input representation having the same real intelligibility errors on the test triphone set.

	poly	trk	fft	tri	Trad.Tri.10.0	Trad.TRat.10.1	Trad.Poly.12.1	Trad.PolyBark.14.1	Trad.FFT4.20.0	Trad.TDif.16.4
poly	—	0.89	0.41	0.00	0.00	0.00	0.00	0.00	0.00	0.00
trk	0.89	—	0.17	0.01	0.00	0.00	0.00	0.00	0.00	0.00
fft	0.41	0.17	—	0.01	0.00	0.00	0.00	0.00	0.00	0.00
tri	0.00	0.01	0.01	—	0.53	0.00	0.00	0.00	0.00	0.00
Trad.Tri.10.0	0.00	0.00	0.00	0.53	—	0.00	0.00	0.00	0.00	0.00
Trad.TRat.10.1	0.00	0.00	0.00	0.00	0.00	—	0.00	0.00	0.00	0.00
Trad.Poly.12.1	0.00	0.00	0.00	0.00	0.00	0.00	—	0.07	0.00	0.00
Trad.PolyBark.14.1	0.00	0.00	0.00	0.00	0.00	0.00	0.07	—	0.00	0.00
Trad.FFT4.20.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	—	0.52
Trad.TDif.16.4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.52	—

Table 9.6. The probabilities of the best ANNs using the different types of input representation having the same real intelligibility errors on the traintest triphone set.

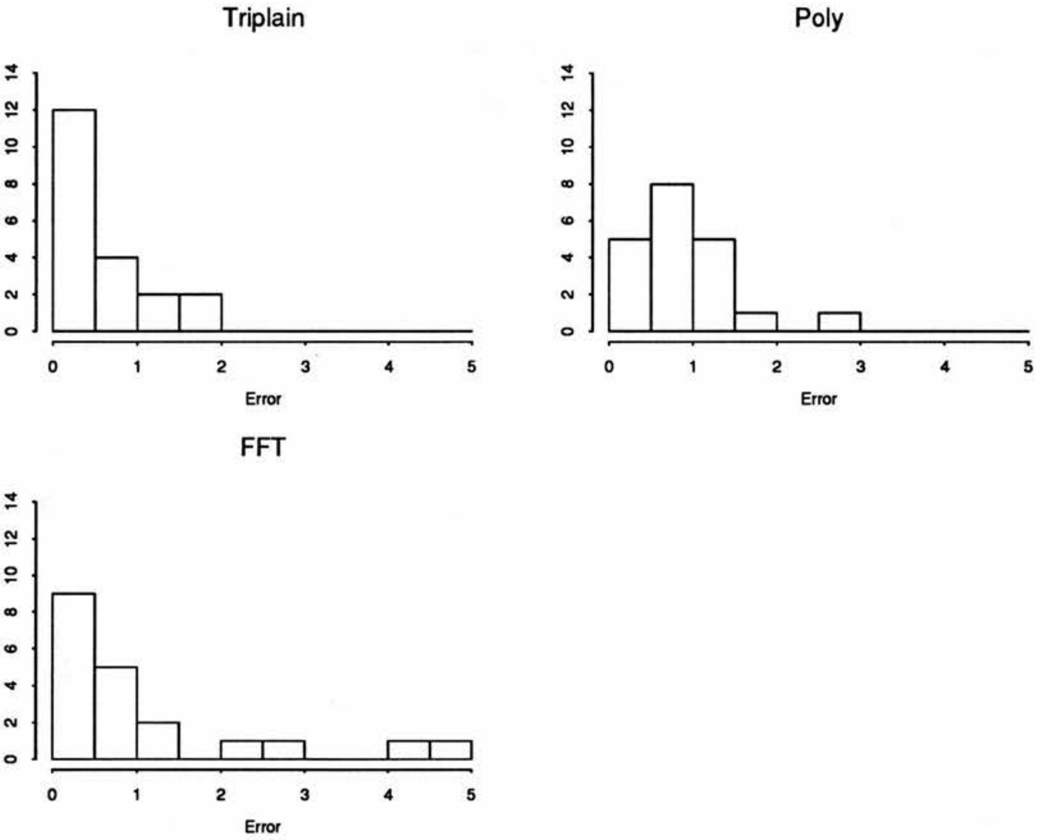

Figure 9.7. The scaled absolute error values for the test set of triphones on the output node determining the frequency of the centre of the first formant of the best ANN trained to produce the *Tri-plain*, *Polynomial* and *FFT4* representations. The scaling was by the standard deviation of the error values of the representation on the training set.

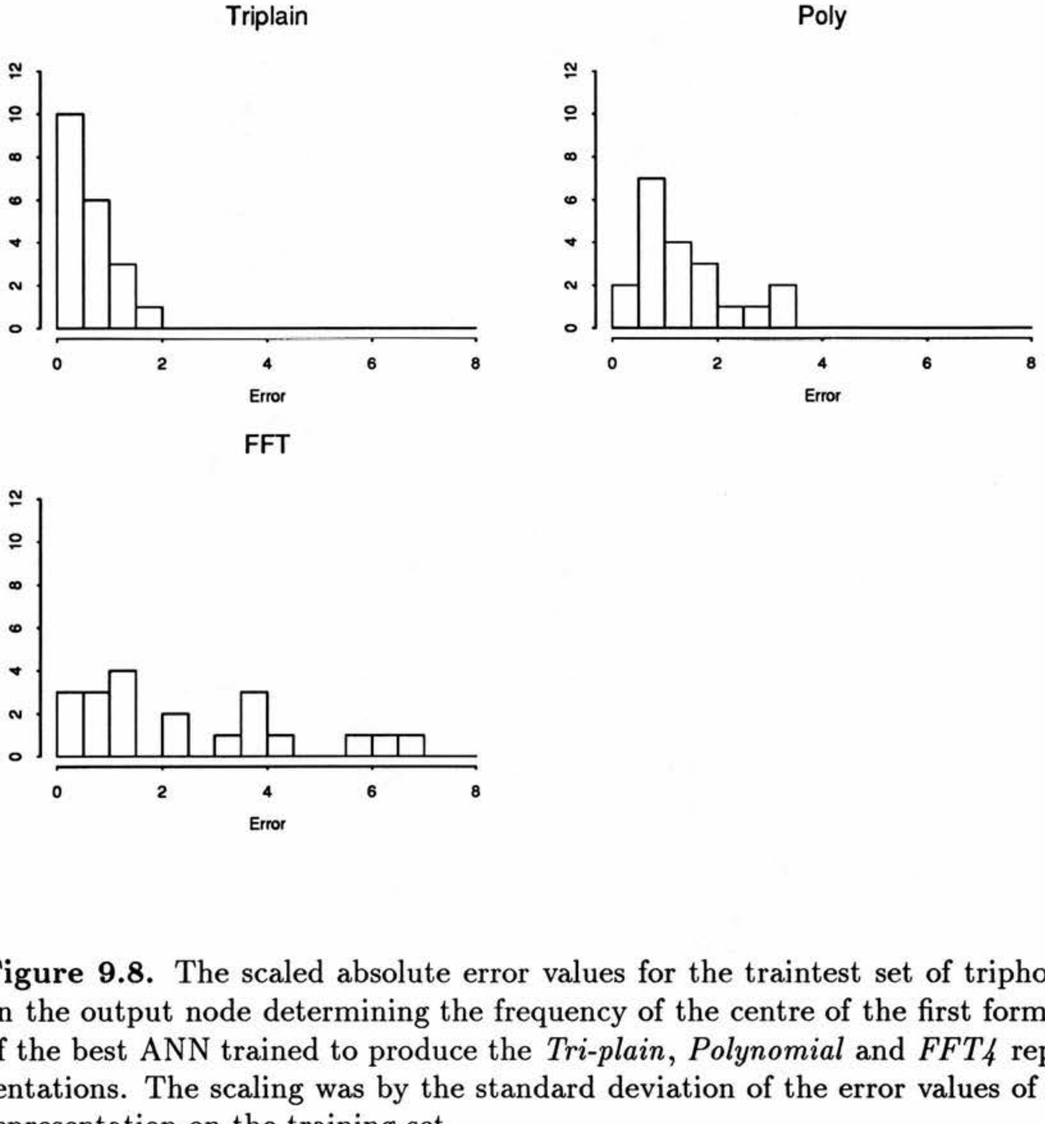

Figure 9.8. The scaled absolute error values for the traintest set of triphones on the output node determining the frequency of the centre of the first formant of the best ANN trained to produce the *Tri-plain*, *Polynomial* and *FFT4* representations. The scaling was by the standard deviation of the error values of the representation on the training set.

errors are worse, both for the traintest and test sets of triphones. This pattern is repeated if the errors across all of the nodes are examined. The output errors for the training set did not follow this clear pattern (compare, for instance, Figures 9.5 against 9.7 and 9.8). The traintest set, being a subset of the training set, should show the same pattern as the training set as a whole, but is clearly not very representative.

Comparing the test set errors with the training set errors shows that while the training set output values for the *Tri-plain*, *Poly* and *FFT4* representations were learnt with similar accuracy, generalisation varied, with the *Tri-plain* representation leading to better generalisation, the *Poly* representation being significantly worse and the *FFT4* representation being worst of all.

9.4.4 Effect of Errors in Output Values on Intelligibility Errors

Once the output error values have been scaled to account for the varying spreads of output values in the training data there is a clear relationship between the output errors and the intelligibility errors. Hence, there is no evidence from the above results for differing sensitivities of the vowel formant representations to errors in the output values. Any such effect is buried in the differences between the test set output error patterns for the three representations examined.

9.4.5 Questions 4 and 8: Output Representations and Hidden Nodes

As for the input representations, the number of hidden nodes necessary to accurately learn the training data and produce good generalisation may provide some insight into the "difficulty" of learning the mapping between input and output data for the representations used. For more discussion, see Section 6.6.4.

Examining the total output errors for all ANNs on any one of the output representations shows no obvious patterns, except that the more successful representations tend to have lower numbers of hidden nodes in their most successful ANN. Figure 9.9 shows the ANN output errors for the ANNs using the *PolyBark* output representation. Looking at the predicted intelligibility errors (which are proportional to the output errors for F1 only) there does seem to be worse performance for lower numbers of hidden units, and possibly a worsening performance with more hidden units than the best for some representations. However, this is very inconclusive and should be regarded with some scepticism. Figure 9.10 shows the predicted values for the ANNs using the *PolyBark* output representation.

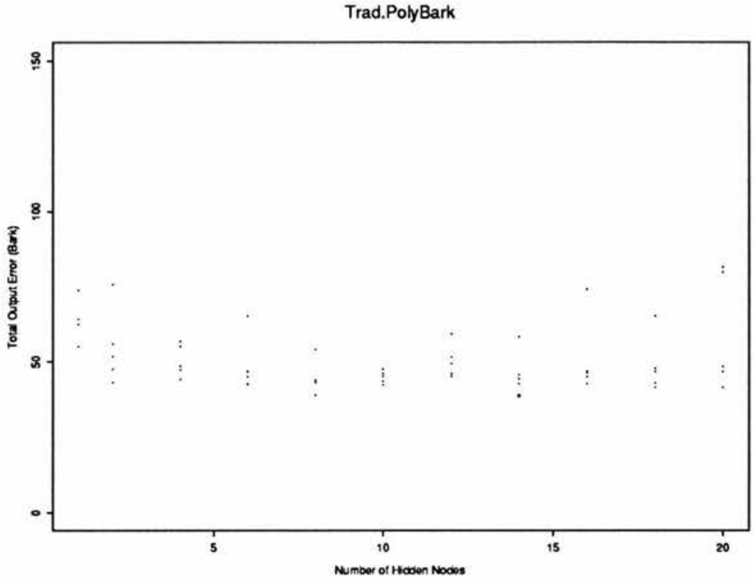

Figure 9.9. The total output errors for all ANNs trained on the *PolyBark* output representation. The best performance is marked with a larger dot. The *Traditional* input representation was used.

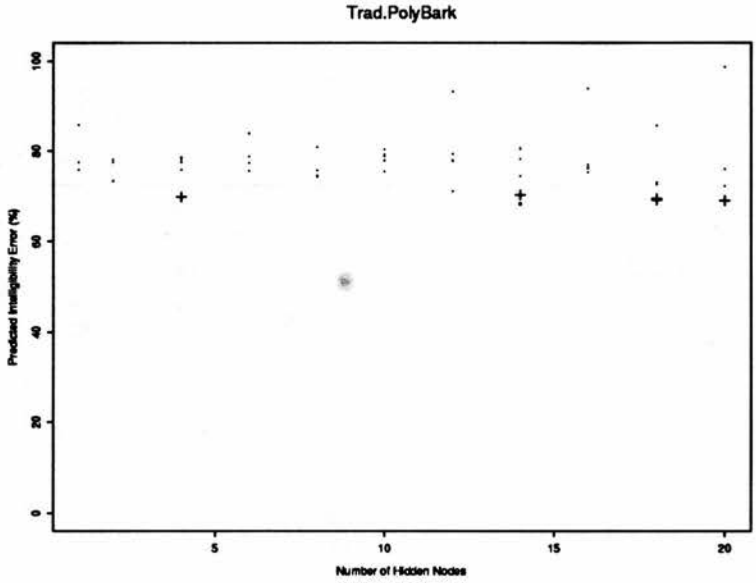

Figure 9.10. The predicted intelligibility errors for all ANNs trained on the *PolyBark* output representation. The best performance is marked with a larger dot. Errors with a more than 5% chance of being better than the best choice are marked with a cross. The *Traditional* input representation was used.

9.5 Comparing Different Training Methodologies

9.5.1 Question 5: Comparing Back-propagation to Conjugate Gradient ANN Training Algorithms

Most of the ANNs in my experiments were trained with an artificial neural network simulator using the conjugate gradient algorithm [54, 77] to minimise the output errors on the training set. A few ANNs were trained with a different simulator using the more well-known back-propagation algorithm [84]. The conjugate gradient method was preferred due to the great speed-up in training times that resulted, typically of the order of 5-10 times. Comparisons can be made between the intelligibility errors for the speech produced on the test and traintest sets by two pairs of ANNs. Trad.Tri.10.0 can be compared with Trad.Tri.bp.10.0 and Trad.TRat.10.1 can be compared with Trad.TRat.bp.8, although for the latter pair there is a further difference in that Trad.TRat.10.0 has no hidden layer. Figure 9.11 shows the mean intelligibility errors for these ANNs and the associated standard errors for the test and traintest triphone sets. Tables 9.7 and 9.8 show the results of a pairwise t-test on the mean marks per subject for the ANNs for the test and traintest triphone sets.

There is no significant difference in intelligibility error between Trad.Tri.10.0 and Trad.Tri.bp.10.0 for the traintest set. There is a significant difference for the test set, with the back-propagation method doing better than the conjugate gradient method.

There is a significant difference (at the 5% level) between the intelligibility errors on the traintest set for Trad.TRat.10.0 and Trad.TRat.bp.8, with the latter, using back-propagation, performing better. This time there is no significant difference on the test set. There is a second difference between these ANNs however, in that Trad.TRat.bp.8 has no hidden layer. It seems that having no hidden layer had no negative effects on performance (see Section 9.5.3). So, in one case the back-propagation algorithm seems to produce better generalisation to the test

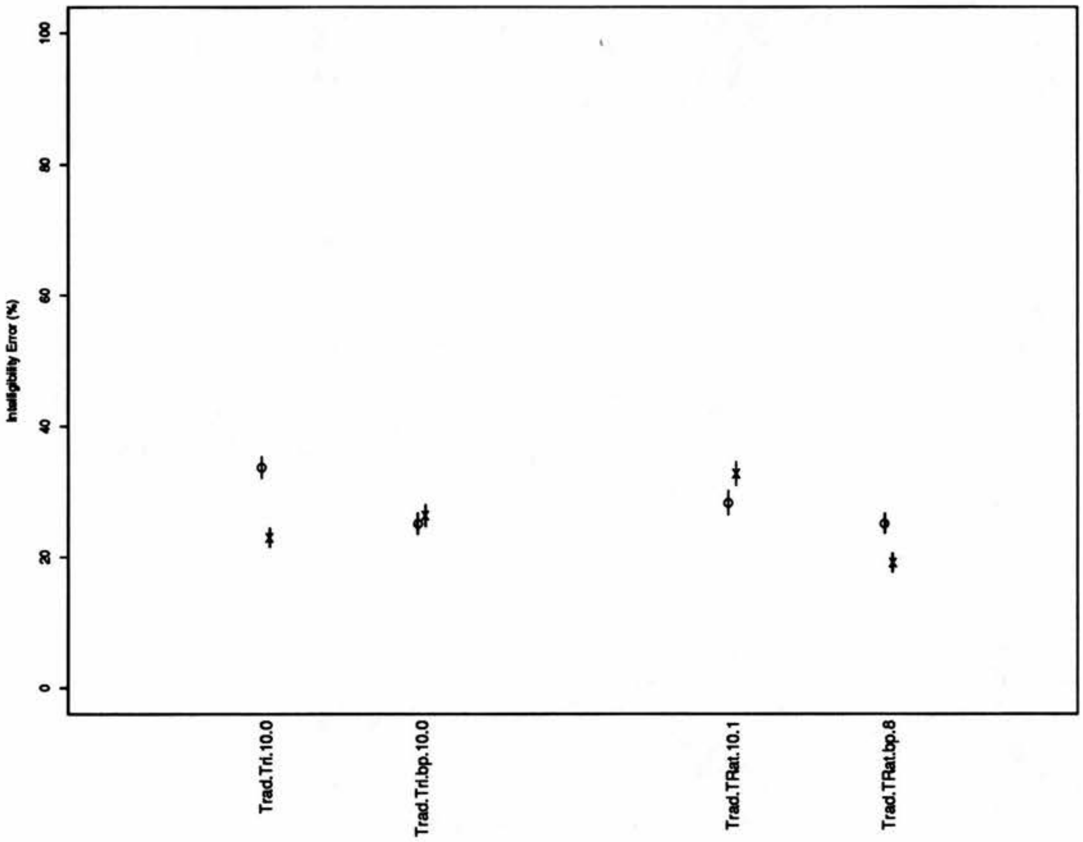

Figure 9.11. The mean test set (O) and traintest set (X) intelligibility errors for the ANNs which differ on the minimisation algorithm, using either conjugate gradient or back-propagation methods. The bars show one standard error each side of the means.

	Trad.Tri.10.0	Trad.Tri.bp.10.0	Trad.TRat.10.1	Trad.TRat.bp.8
Trad.Tri.10.0	—	0.00		
Trad.Tri.bp.10.0	0.00	—		
Trad.TRat.10.1			—	0.16
Trad.TRat.bp.8			0.16	—

Table 9.7. The probabilities of the ANNs which differ in minimisation method not having the apparent ordering of intelligibility error for the test triphone set.

	Trad.Tri.10.0	Trad.Tri.bp.10.0	Trad.TRat.10.1	Trad.TRat.bp.8
Trad.Tri.10.0	—	0.10		
Trad.Tri.bp.10.0	0.10	—		
Trad.TRat.10.1			—	0.00
Trad.TRat.bp.8			0.00	—

Table 9.8. The probabilities of the ANNs which differ in minimisation method not having the apparent ordering of intelligibility error for the traintest triphone set.

set without boosting training set performance, and in the other case the training set performance is improved but does not produce better generalisation.

Overall, I think little can be confidently stated about the relative merits of the back-propagation and conjugate gradient minimisation algorithms when applied to this particular set of problems, although the results do suggest that using the back-propagation algorithm may give better results. Against this must be balanced the increased training times for back-propagation compared with the conjugate gradient method. The relative benefits may well depend on the individual case, being determined by the error landscape the learning algorithm must cross (which depends on the nature of the mapping problem), and the starting point in that landscape (which depends on the initial state of the ANN).

9.5.2 Question 6: Comparing Cross-validation to Training to Completion Methodologies

Most ANNs were trained using the cross-validation methodology. That is, after each update of the ANN connection weights in training, the performance on a cross-validation set (not part of the training set or the final test set) was determined. The ANN state producing the best performance on the cross-validation set was taken to be the result of the training procedure. This methodology aims to prevent the effects of overtraining. Some of the ANNs were trained to completion without using a cross-validation set. The state of the ANN when the training procedure reached a steady state (ie. there was no further improvement possible from the current state) was taken to be the final outcome of the training procedure. Trad.TRat.10.1, trained using cross-validation can be compared to Trad,TRat.end.12.1, trained to completion. Trad.Tri.10.0, trained using cross-validation, can be compared to Trad.Tri.12.1, trained to completion. Figure 9.12 shows the mean intelligibility errors of these ANNs and the associated standard errors, for the test and traintest triphone sets. Table 9.9 and 9.10 show the results of a pairwise t-test on the mean marks per subject of the ANNs, for the test and traintest sets.

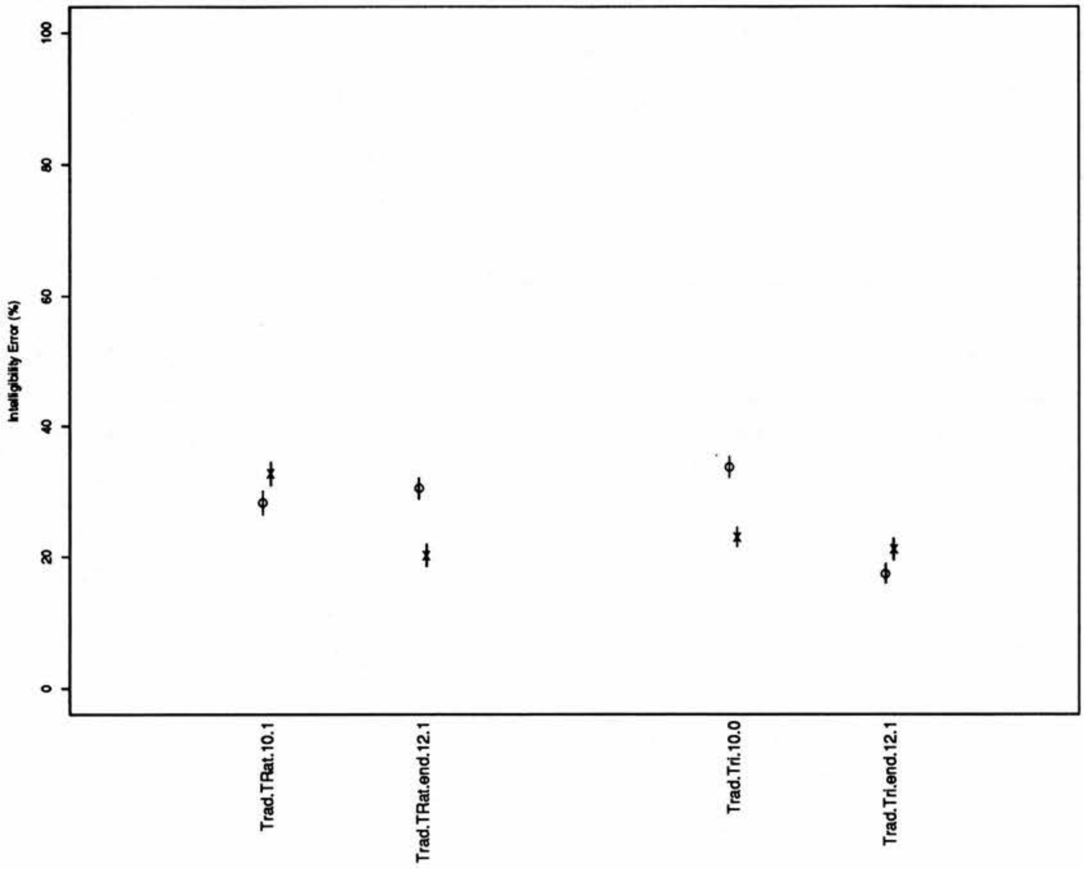

Figure 9.12. The mean test set (O) and traintest set (X) intelligibility errors for the ANNs which differ in the training methodology using either cross-validation or training to completion methods. The bars show one standard error each side of the means.

	Trad.TRat.10.1	Trad.TRat.end.12.1	Trad.Tri.10.0	Trad.Tri.end.12.1
Trad.TRat.10.1	—	0.65		
Trad.TRat.end.12.1	0.65	—		
Trad.Tri.10.0			—	0.00
Trad.Tri.end.12.1			0.00	—

Table 9.9. The probabilities of the ANNs which differ in training methodology (using either cross-validation or training to completion methods) not having the apparent ordering of intelligibility error for the test triphone set.

	Trad.TRat.10.1	Trad.TRat.end.12.1	Trad.Tri.10.0	Trad.Tri.end.12.1
Trad.TRat.10.1	—	0.00		
Trad.TRat.end.12.1	0.00	—		
Trad.Tri.10.0			—	0.43
Trad.Tri.end.12.1			0.43	—

Table 9.10. The probabilities of the ANNs which differ in training methodology (using either cross-validation or training to completion methods) not having the apparent ordering of intelligibility error for the traintest triphone set.

We have the same odd pattern that occurred in Section 9.5.1. Trad.TRat.-end.12.1 performs significantly better than Trad.TRat.10.0 on the traintest set but not on the test set, while Trad.Tri.end.12.1 performs significantly better than Trad.Tri.10.0 on the test set but not on the traintest set. Again, it seems that training to completion may have been advantageous in this case, but the nature of the improvement is not clear cut.

The use of the cross-validation methodology was inadequate in these experiments anyway. The complete set of CVC triphones available was smaller than would have been ideal. This necessitated using most of the triphones (514) in the training set, with the cross-validation and test sets being very small (20 triphones each, originally). Ideally, these three sets would have been of comparable sizes, all larger than the actual training set. This would have required a substantially larger amount of phonemically labelled speech from a single speaker, recorded under a single set of conditions. This was not available.

Using a larger set of consonants, instead of just the stop consonants used here, would have much increased the number of triphones available, while at the same time making the mapping from phonemic to acoustic descriptions more complex. The complexity may well rise more slowly than the increasing number of triphones. The idea of phonemes being compositions of features supports this, as the ANNs would be learning the effects of single features and small groups of features instead of each phoneme being totally distinct in its effects from all others. If learning the effects of one consonant on the adjacent vowel formants did not carry over to other consonants with similar features, then the notion of phonemic features would be weakened. Therefore, it seems that increasing the number of triphones used in training, cross-validation and testing may well have lead to better generalisation performance by the ANNs, and more satisfactory measurement of that performance, despite the increased complexity of the problem.

9.5.3 Question 7: Comparing ANNs Using a Hidden Layer to ANNs With No Hidden Layer

Most of the ANNs trained had a single hidden layer. This allows the ANN to approximate to any required accuracy a large class of continuous functions, given enough hidden nodes and a training algorithm capable of learning the mapping. The task of mapping phonemic descriptions of CVC triphones to the first three formants of the vowel almost certainly falls within the class of functions that an ANN can approximate.

A small number of ANNs were trained using no hidden layer. That is, the input units were directly connected to the output units by weighted links. This architecture limits the class of functions that the ANN can approximate to linear functions. This is a severe restriction and my expectation was that these perceptron ANNs would not be capable of learning the required mapping to the same accuracy as the three-layer ANNs.

The three layer ANN Trad.Tri.bp.10.0 (with 10 hidden nodes) was compared to the two-layer ANN Trad.Tri.bp.4. Both of these ANNs used the back-propagation algorithm in training. The three layer ANN Trad.TRat.10.1 (with 10 hidden nodes) was compared with the two-layer ANN Trad.TRat.bp.8. The first of these was trained using the conjugate gradient algorithm and the second was trained using the back-propagation algorithm. Figure 9.13 shows the mean intelligibility errors of these ANNs and the associated standard errors, for the test and traintest triphone sets. On the traintest set, the first pair had almost identical results, while for the second set Trad.TRat.bp.8 had significantly better performance. However, it is possible this was entirely due to the use of the back-propagation algorithm. On the test set, there was no significant difference within the pairs.

These results suggest that the ANNs with no hidden layers performed as well as the ANNs with a hidden layer. For the second pair, the ANN with no hidden layer performed better, but this may have been due to the difference in training algorithm. Contrary to my expectations, it would seem that the mapping from the phonemic description of CVC triphones (using only stop consonants and monophthongs) to a description of the F1, F2 and F3 formants can be approximated as successfully by a combination of linear functions as by a non-linear function. One caveat is that the inadequacies of the training, cross-validation and test sets may have obscured a real difference between the potential performance of ANNs on this task.

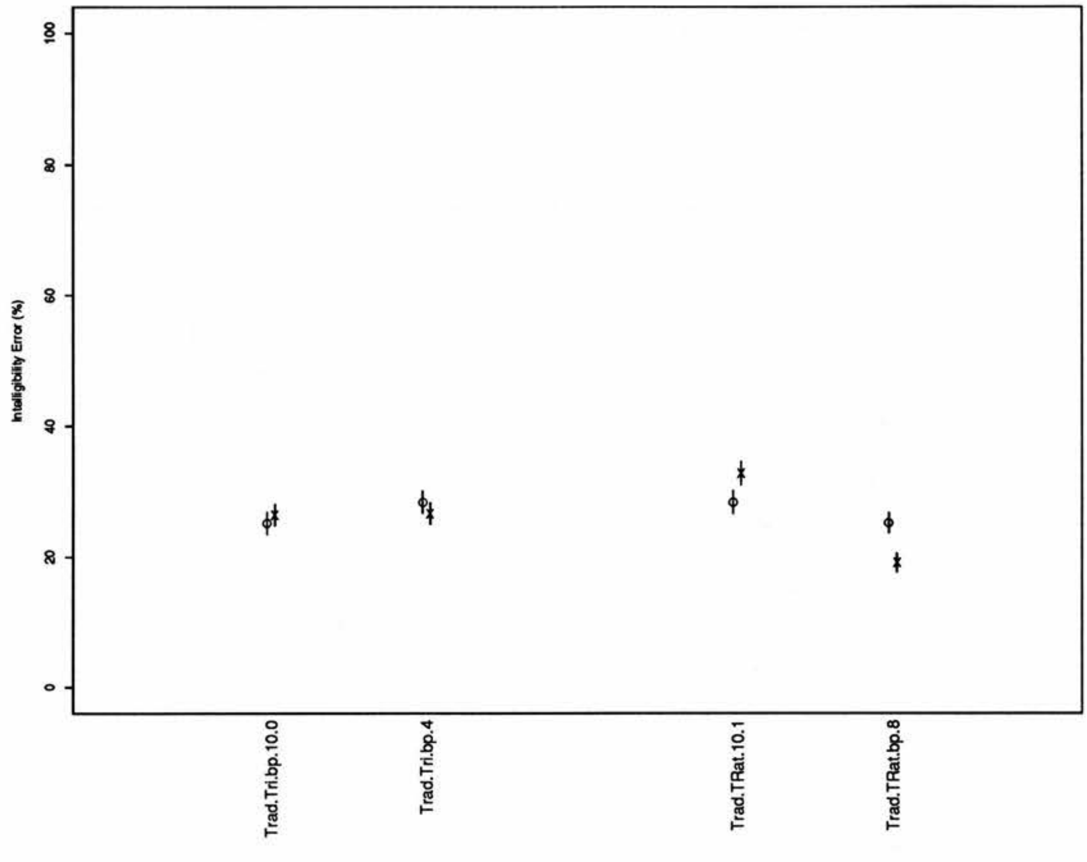

Figure 9.13. The mean test set (O) and traintest set (X) intelligibility errors for the ANNs which differ on whether they have a hidden layer or not. The bars show one standard error each side of the means.

	Trad.Tri.bp.10.0	Trad.Tri.bp.4	Trad.TRat.10.1	Trad.TRat.bp.8
Trad.Tri.bp.10.0	—	0.20		
Trad.Tri.bp.4	0.20	—		
Trad.TRat.10.1			—	0.16
Trad.TRat.bp.8			0.16	—

Table 9.11. The probabilities of the ANNs which differ in whether they have a hidden layer or no hidden layer not having the apparent ordering of intelligibility error for the test triphone set.

	Trad.Tri.bp.10.0	Trad.Tri.bp.4	Trad.TRat.10.1	Trad.TRat.bp.8
Trad.Tri.bp.10.0	—	0.93		
Trad.Tri.bp.4	0.93	—		
Trad.TRat.10.1			—	0.00
Trad.TRat.bp.8			0.00	—

Table 9.12. The probabilities of the ANNs which differ in whether they have a hidden layer or no hidden layer not having the apparent ordering of intelligibility error for the traintest triphone set.

9.6 The Best Combinations of Representations and Methods

The best representations and methods are summarised in Table 9.13.

	Test set	Traintest set
Input Representation	Any	*Symbolic* or *Continuous*
Output Representation	*Tri-plain* or *Tri-ratio*	*Tri-plain*
Back-propagation or conjugate gradient	Unclear	Unclear
Cross-validation or to completion	Completion? Unclear	Completion? Unclear
3-layer or 2-layer	No difference	No difference

Table 9.13. The best representations and methodologies, for the test and traintest sets of triphones.

Of the trained ANNs used in the Experiment II intelligibility test (see Chapter 8), Trad.Tri.end.12.1 performed best on the test set. On the traintest set, Trad.Tri.bp.8 performed best, closely followed by Trad.TRat.end.12.1 and Trad.-Tri.end.12.1. On combined scores, Trad.Tri.end.12.1 came out best. This is consistent with the best choices determined above, except for the *Traditional* input representation. However, there were no examples of ANNs evaluated in the Experiment II intelligibility test with *Symbolic* or *Continuous* input representations and the best choices of other values. That the best overall combination is compatible with the individual best choices suggests that they are independent, or that any interaction is too small to show in the restricted number of combinations examined here.

Chapter 10

Comparison of Original and ANN Produced Vowel Formant Tracks

This chapter looks at some of the vowel formant tracks produced by an ANN, comparing them to the original tracks used in training. I examine some interesting features of the ANN output and the original data.

I have used the ANN Trad.Tri.end.12.1, which had the best overall performance on the test and traintest triphone sets in the final intelligibility test. The effects of duration and contexts are illustrated. The comparisons are made using the *Tri* representation, where each formant track is represented by a triple of values — the initial, central and final frequency values of the formant. In the figures, phonemes are written in the Machine Readable Phonetic Alphabet (MRPA), described in Appendix C.

10.1 Formant Tracks, Durations and Vowels

Figure 10.1 shows the F1, F2 and F3 vowel formant tracks for all instances of /pɔt/ in the original training data. Figure 10.2 shows the equivalent vowel formant tracks produced by the ANN Trad.Tri.end.12.1. The tracks for the original data show the variability of the training data. The tracks produced by the ANN follow a regular pattern, with the effect of duration clearly apparent. For instance, for F2, longer durations lead to a lowering of the central frequency, which could be interpreted as a closer approach to the vowel target. The onset and final frequencies are also lowered with increased duration, but not to the same extent. Figure 10.3 shows how the central vowel formant frequencies vary with vowel duration. The points form straight lines with no flattening off. If the vowel centres are tending towards a vowel target then the target does not appear to be reached in any of the utterances.

Figure 10.4 shows F1, F2 and F3 vowel formant tracks from the original data and from the ANN for an instance of each vowel in the context /p_t/. The instance chosen is that with the median duration over all instances of the triphone. If there are an even number of instances of a triphone, the instance with duration just greater than the median was chosen. The tracks produced by the ANN follow the same general pattern as those taken from the original data, but with numerous small differences. These differences may be as much due to the random variability in the original natural speech as to a failure of the ANNs to produce intelligible vowel formants. For instance, the /pat/ vowel formants produced by the ANN differ in shape from the original vowel formants for the utterance shown, but are in fact much closer to the vowel formant shapes of all the other original /pat/ triphones.

10.2 The Distributions of the Frequencies of the Centre Points of the Vowel Formant Tracks

Figure 10.5 shows the distributions of the frequencies of the centre points for each vowel, for the original data and for the tracks produced by the ANN Trad.-Tri.end.12.1. This follows the expected pattern, with the ANN produced tracks having similar medians but smaller variances than the original data.

Figure 10.6 shows the same comparison for the context /p_t/ only. Both the original and ANN tracks show a general reduction in variance, except for /pat/

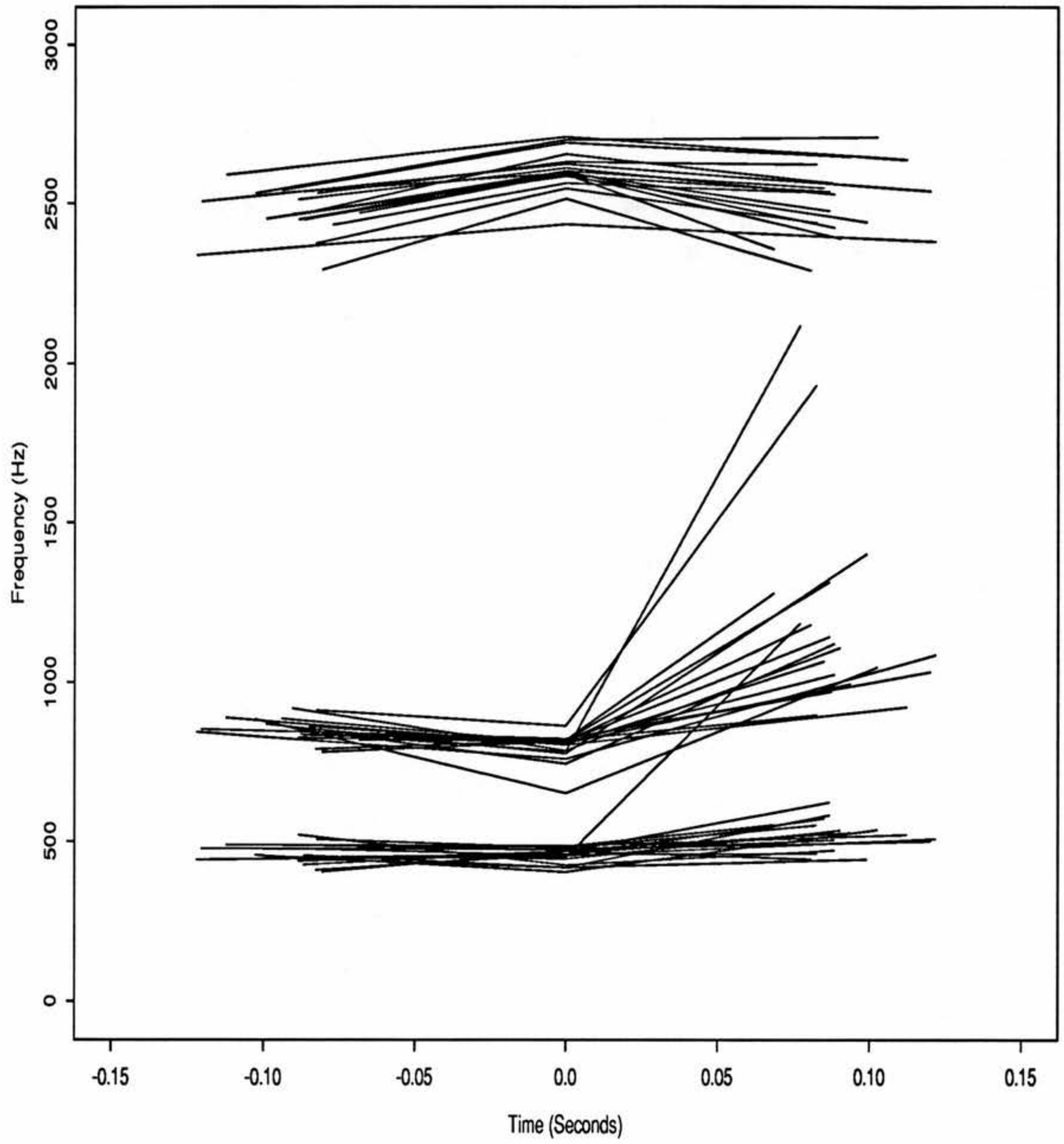

Figure 10.1. The original vowel formant tracks for all instances of the triphone /pɔt/ in the training data. The formant tracks have been summarized as the initial, central and final frequencies.

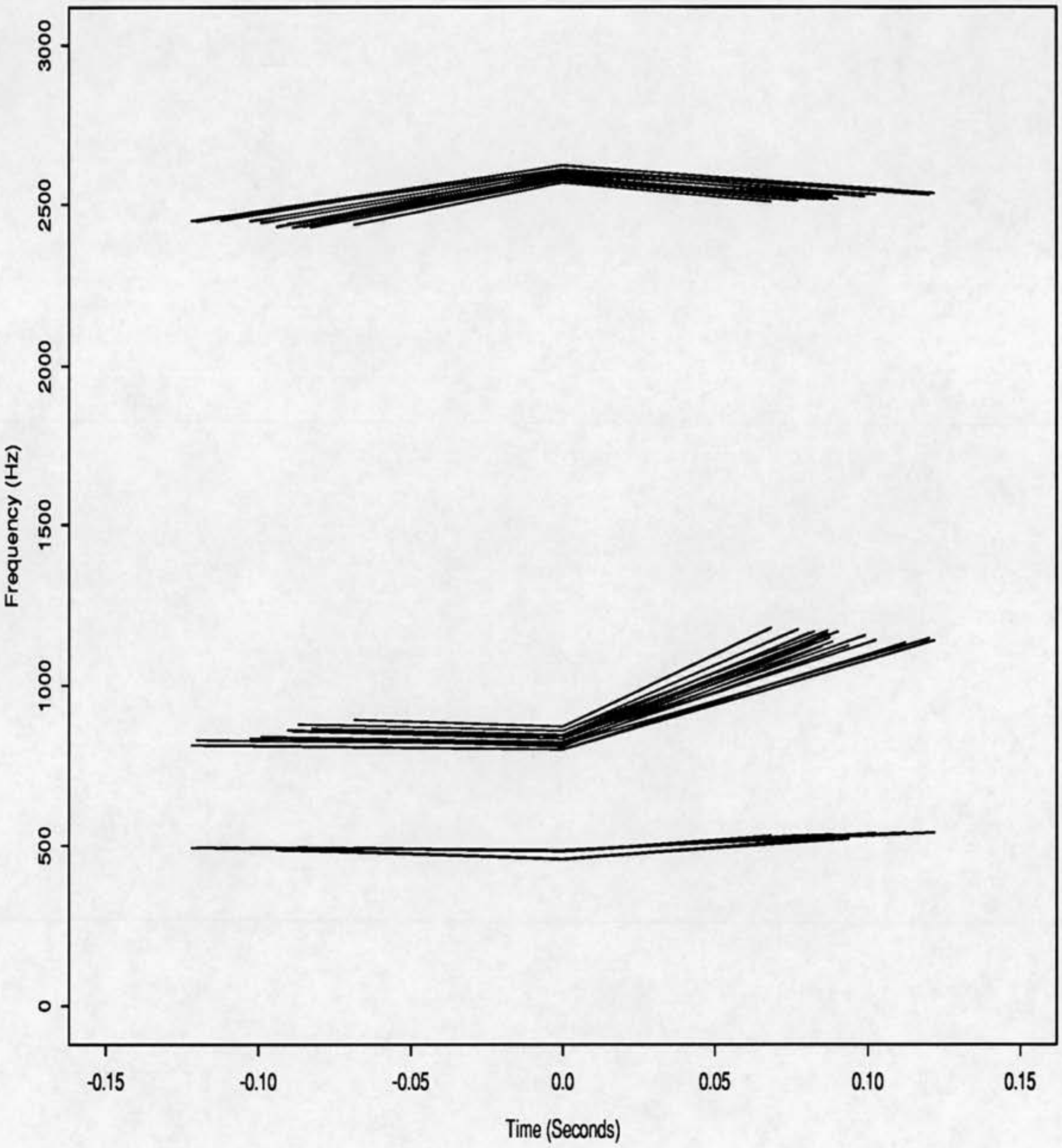

Figure 10.2. The vowel formant tracks produced by the ANN Trad.Tri.end.12.1 for all instances of the triphone /pɔt/ in the training data. The formant tracks have been summarized as the initial, central and final frequencies.

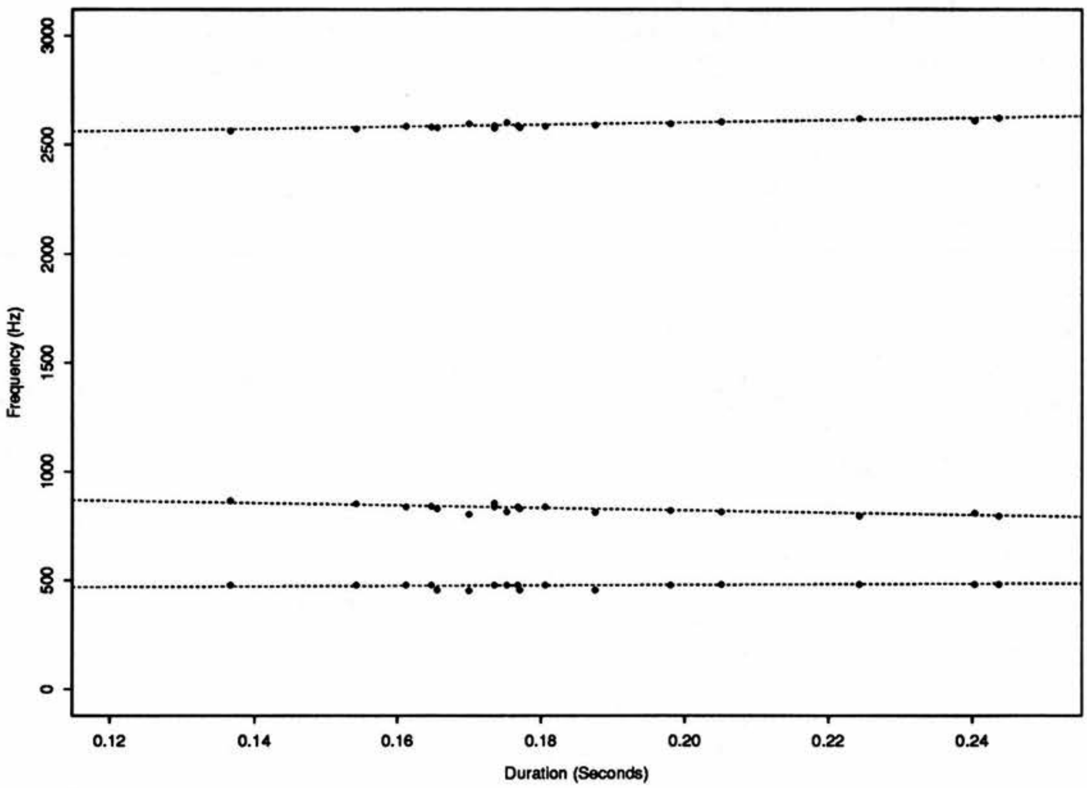

Figure 10.3. The frequencies of the centre of the F1, F2 and F3 vowel formants produced by the best overall ANN for all instances of the triphone /pɔt/ in the training data, plotted against the vowel duration.

and /pɪt/. The ANN F1 tracks show almost no variance, showing that duration has had little effect in determining the F1 tracks in this context (and probably in all contexts). This can be seen in Figure 10.2, where the F1 tracks have a very narrow spread.

10.3 The Distributions of the Frequencies of the End Points of the Vowel Formant Tracks

Figure 10.7 shows the distribution of the frequencies of the initial points of the vowel formant tracks, for both the original data and the tracks produced by the ANN Trad.Tri.end.12.1. Figure 10.8 shows the distribution for triphones with an initial /p/. Figure 10.9 shows the distribution in the context /p_t/. In all cases the medians are similar for the original and ANN produced tracks, with smaller variances for the ANN produced tracks. Adding more context reduces the variances. For instances of the same triphone, the variance is in general small for the original speech and very small for the ANN produced speech. It is clear that the final phoneme does have an effect on the frequency of the start point of the vowel formants. The frequencies of the final points of the vowel formant tracks behave in the same fashion.

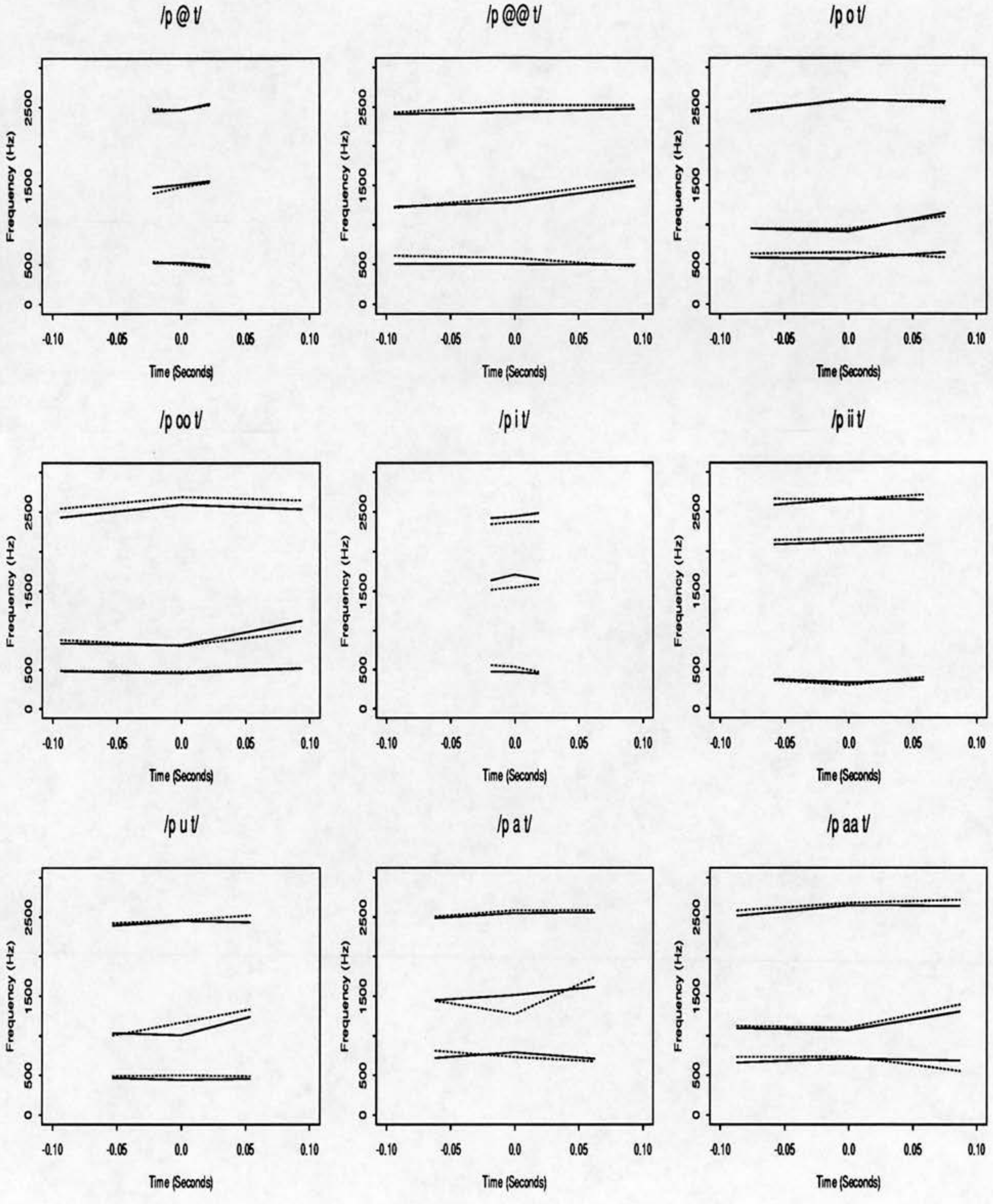

Figure 10.4. Formant tracks for context /p_t/ for each vowel, for the original training data (dashed line) and the ANN Trad.Tri.end.12.1 (solid line). The instance with the median duration is plotted.

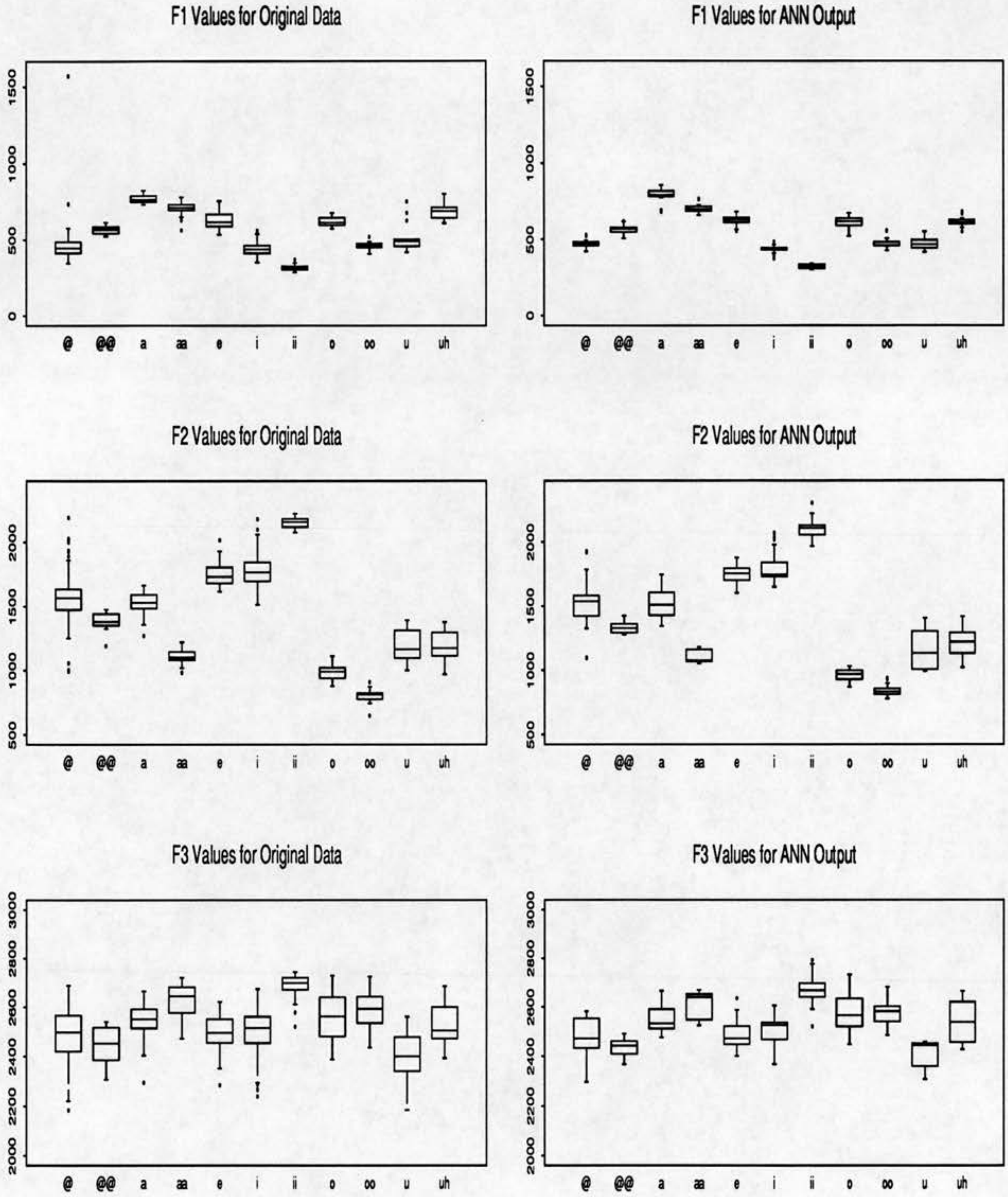

Figure 10.5. Distributions of the vowel centre frequencies for the original formant tracks and the formant tracks produced by the ANN Trad.Tri.end.12.1.

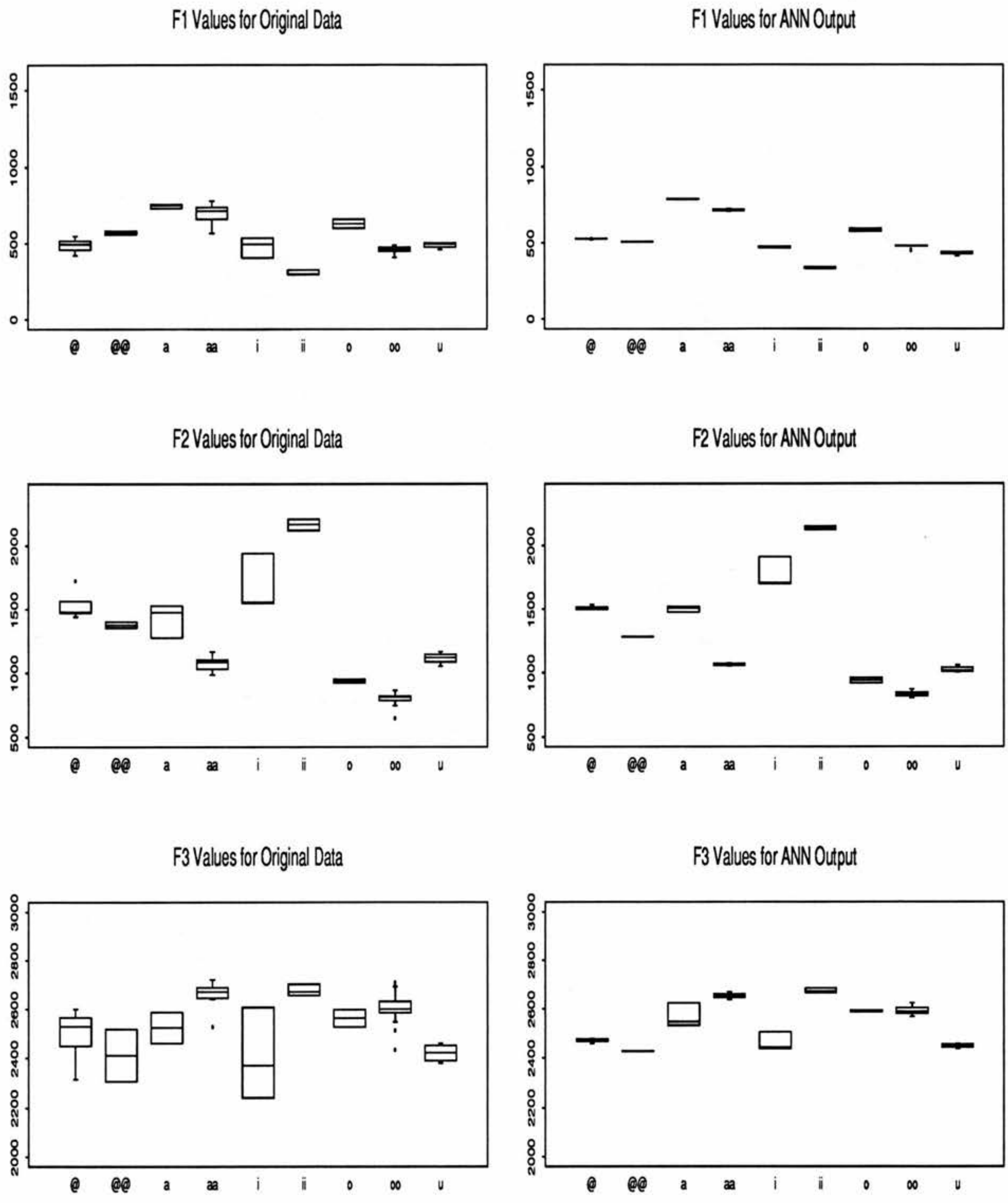

Figure 10.6. Distributions of the vowel centre frequencies for the original formant tracks and the formant tracks produced by the ANN TradTri.end.12.1, in context /p_t/.

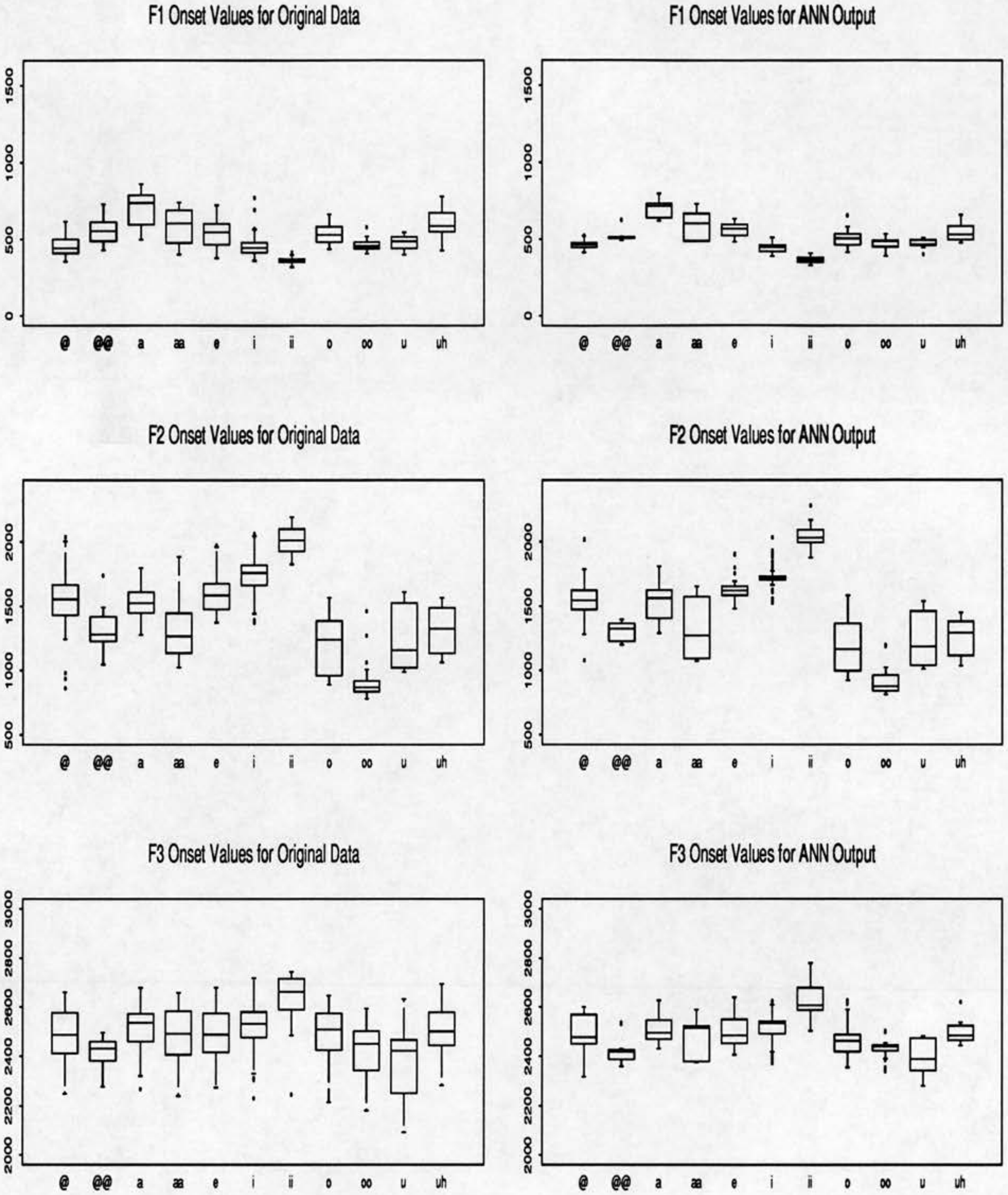

Figure 10.7. Distributions of the vowel onset frequencies for the original formant tracks and the formant tracks produced by the ANN Trad.Tri.end.12.1, over all contexts.

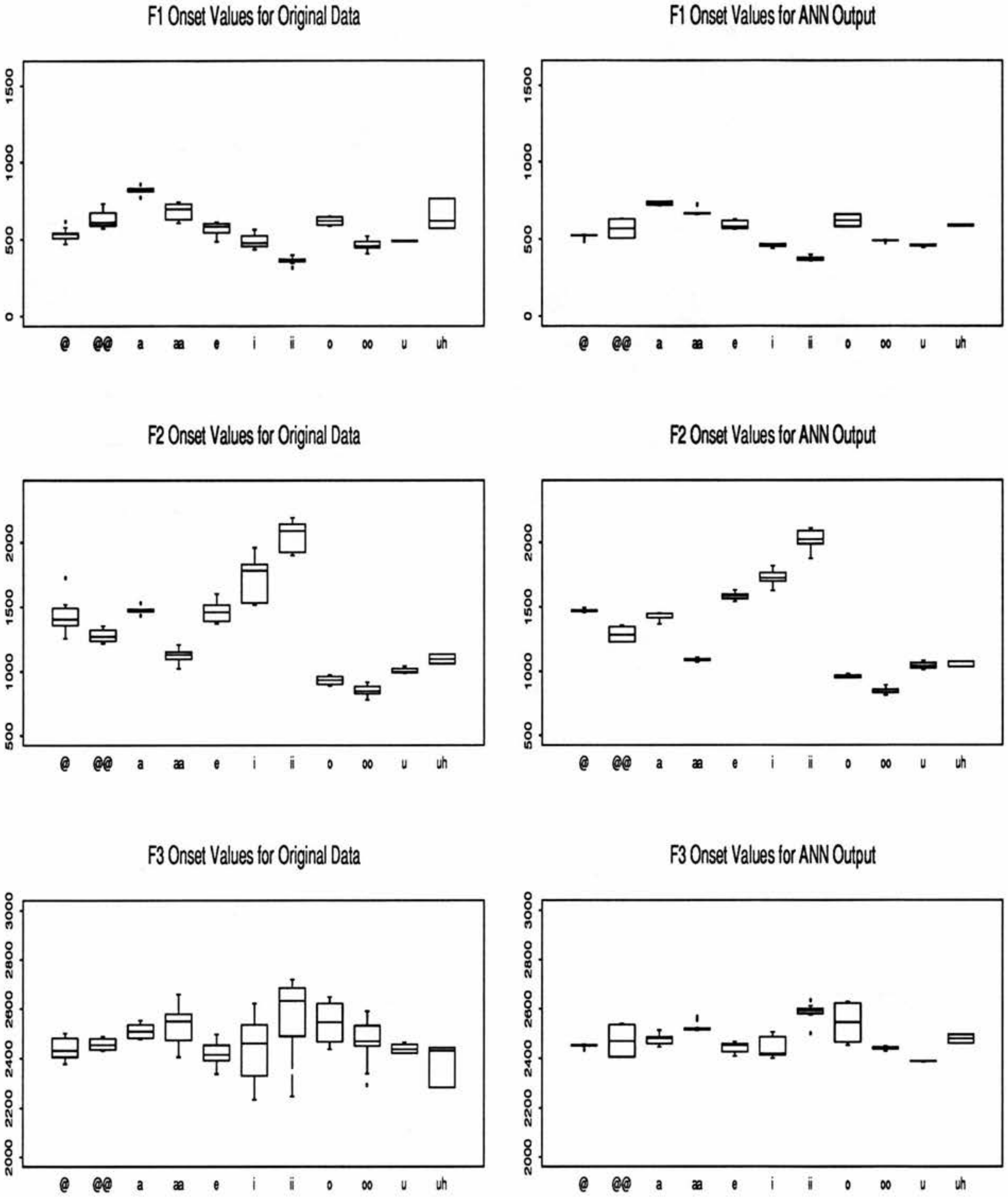

Figure 10.8. Distributions of the vowel onset frequencies for the original formant tracks and the formant tracks produced by the ANN Trad.Tri.end.12.1, in context /p--/.

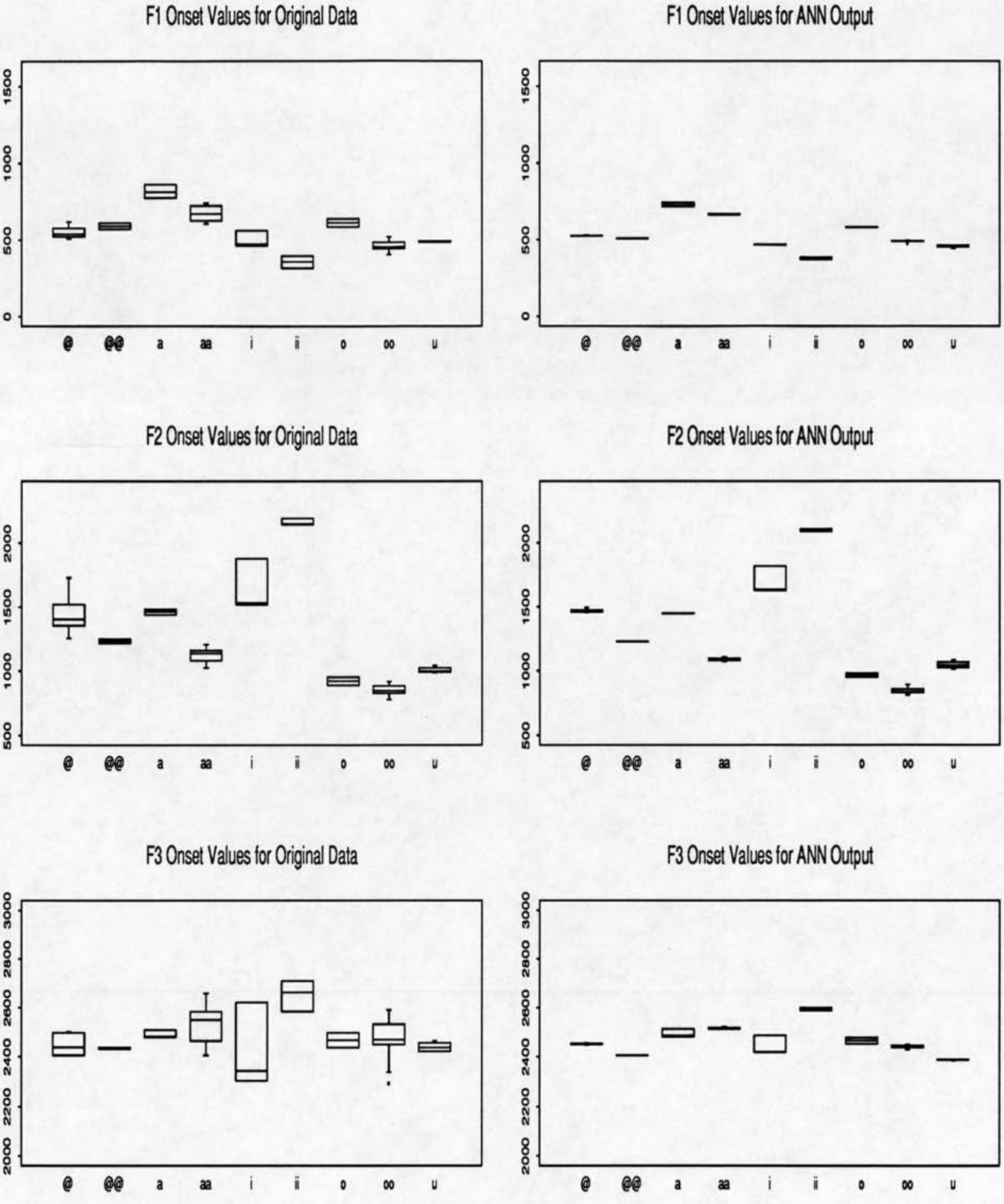

Figure 10.9. Distributions of the vowel onset frequencies for the original formant tracks and the formant tracks produced by the ANN Trad.Tri.end.12.1, in context /p_t/.

Chapter 11

Conclusion

11.1 Summary

The aim of this thesis was to investigate the ability of ANNs to learn the mapping from CVC triphones to F1, F2 and F3 vowel formant tracks and to investigate the influence upon learning this mapping of a number of factors. The form of the output representation was found to be the most important factor, with the simple *Tri-plain* representation, consisting of the initial, central and final frequencies of the tracks, being the most successful. The input representation had less influence, with there being no difference in generalisation between the representations used, and only slight differences on the learning of the training data.

The influence of the ANN training algorithm (either back-propagation or a conjugate-gradient method) was unclear. The effects of using cross-validation were also unclear, although this was possibly due to the inadequate size of the cross-validation data set. Surprisingly, two-layer ANNs (with no hidden layer and hence capable of a more restricted class of functions than those with a hidden layer) appeared to perform as well as three-layer ANNs. If this is so, and not merely a result of other limitations in the experiments, such as the rather crude formation of the triphone utterances, then we can conclude that the function mapping from CVC triphones to the vowel formant tracks is essentially linear.

I was unable to come to any conclusions about the influence of the number of hidden nodes in the three-layer ANNs on learning the mapping.

11.2 Discussion

I have successfully demonstrated the ability of feed-forward ANNs to learn the mapping between a broad phonetic representation of CVC triphones and a representation of F1, F2 and F3 vowel formant tracks. I have also had some success in investigating the influence of input and output representations on the success of that learning process.

Input representations have less of an impact than I would have expected. I now think that I should have used a purely binary representation, such as that used by Chomsky and Halle in The Sound Pattern of English [16], although I suspect that this would have differed little from the other representations in performance. Another possibility would be to use a richer, structured representation, from one of the non-linear phonologies, although the transformation of this kind of representation to a form suitable for input to an ANN would not be straightforward.

It may be that a better method of determining the number of hidden nodes required in adequately learning the mapping would have revealed some differences between the input relationships. The use of an ANN training algorithm that automatically adjusts the number of hidden nodes, such as the Cascade-Correlation algorithm [24] might have been more successful than attempting to select between ANNs trained with different sizes of hidden layer.

There are obvious differences between the performance using the different output representations. Fourier coefficients performed very poorly. Polynomial coefficients performed better, but not greatly. Two of the three representations based on taking three points per formant track (initial, central and final) performed well, producing the most intelligible speech. This was despite the representation, as extracted from the original speech, producing the least intelligible utterances of the three types. This demonstrates that these *Tri* representations were by far the easiest for the ANNs to learn, far outweighing their lack of accuracy in reproducing the original formant tracks.

Interestingly, the ANN produced utterances, using the *Tri-plain* representation, were more intelligible than the utterances produced from the *Tri-plain* representation as extracted from the original speech data. I ascribe this to two factors. Firstly, the lack of random variance in the ANN produced speech, compared with the original, natural speech may increase intelligibility. Secondly, the original vowels were taken from triphones within isolated words, so they may have been influenced by context beyond the CVC triphone which is obviously lacking in the resynthesised utterance.

I found no difference between the performance of ANNs trained using the back-propagation algorithm and those trained using the conjugate gradient algorithm. However, the conjugate gradient algorithm is to be preferred for practical reasons, as training with it takes much less time than with the traditional back-propagation algorithm.

Cross-validation proved to be ineffective, probably due to the very inadequate size of the cross-validation data set. A much larger set of single speaker speech data would have improved all aspects of this work.

It is interesting that ANNs without a hidden layer performed as well as those with a hidden layer. This indicates that the mapping is essentially linear, at least as far as can be determined within the constraints imposed by the limitations of my work. The linearity of the mapping is consistent with the success of the various formant models discussed in Chapter 5. Only the Imaizumi and Kiritani model [45] is truly non-linear. The Broad and Clermont models [11] use linear combinations of functions which are either unique to each consonantal context, or exponential functions whose parameters are determined by the consonantal context.

One major disadvantage of using ANNs to learn a mapping is their black-box nature. It is difficult to get any understanding of how an ANN is carrying out a mapping. I have shown in this thesis how the input and output representations and a number of other factors can influence the ability of ANNs to learn to produce a representation of vowel formant tracks, but I have not been able to give any account of the way in which the ANNs perform the mapping. I have made an attempt to investigate the internal processes of the ANN that had the best overall performance, but have made no progress. The commonly used techniques, such as principal components analysis, apply more naturally to ANNs used for classification than to my ANNs which are mapping to a continuous space.

11.2.1 Discussion of The Relationships Derived Between Vowel Formant Error and Intelligibility Error

The equations predicting modified rhyme test intelligibility errors from formant track errors, derived using data from Experiments I and II, are an interesting facet of my work. The Experiment I derived equation was found in order to help me select ANNs to use in Experiment II. The results support the common practise of using root-mean-square formant track errors as a measure of the adequacy of synthesised speech, and have the added bonus of giving some idea of the likely intelligibility error, and confidence bounds on that error. This allows the experimenter to determine how sure he is that some process which decreases the formant track error really does increase the intelligibility. However, applying these results to synthetic formant tracks outside the work presented in this thesis is rather suspect.

These results are based on a particular set of test triphones, and most of the triphone sets were created by synthesising the formant descriptions output by neural networks trained to produce these when given descriptions of CVC triphones as input. The choice of the test triphones is somewhat arbitrary and is restricted to a subset of stop consonants and vowels. The neural network output may be biased in certain ways and will not cover all possible patterns of errors. In particular, the F1 and F2 formant errors are highly correlated, resulting in there being no extra predictive power in using the F2 errors once the F1 errors have been included in the regression. The F3 errors are less correlated with the

F1 errors but do not quite reach a significant level of predictive power once the square of the F1 formant errors has been used, but with a larger amount of data I would expect this to play a role.

The technique used to produce the triphone given the vowel is crude and there may well be an end effect restricting the minimum intelligibility error obtained.

The relationships found seem reasonable. In both cases the F1 formant error was the most important and the intelligibility error was a monotonically increasing function of the F1 error. The confidence intervals are rather larger than I would like and make the relations a rather blunt instrument in deciding if one predicted intelligibility error is really better than another.

A fundamental restriction is the three basic independent variables used – the F1, F2 and F3 vowel formant errors. These root-mean-square errors probably do not capture all the important features of the shape of vowel formants. They also assume that the model formant tracks (those extracted from the original speech data) are perfect, and are the only correct realisation of the vowel formant in that context. Other aspects of the shape may be important. Two formant tracks with the same root-mean-square formant errors but different shapes (especially at the initial and final points) may have significantly different intelligibility errors. A synthetic formant track which exactly parallels the model track at a small enough distance may have a low intelligibility error but have a high formant track error.

The shape of the formant tracks is not the only determiner of intelligibility. The bandwidth and intensity also matter and were kept at constant rates in the synthesis of the utterances used in the intelligibility tests. If an experimenter is also trying to produce the correct bandwidths and intensities for his synthetic vowels he will need to include some measure of their performance. Another effect of using constant bandwidths and intensities for all the synthesised vowels is that the values used are likely to be more suited to some vowels than others, leading to raised intelligibility error rates for some vowels.

The intelligibility errors and vowel formant errors used are averages over a number of utterances, so the regression equations may be poor predictors of performance on any particular vowel in any particular triphone context. The effects of bandwidth and intensity mentioned above will also have an effect. A third problem is the crude method used to produce the triphones. The consonants were added by concatenating tokens extracted from the original speech to the synthesised vowel. The same tokens were used for all of the triphones. This should reduce the role of particular allophones of the consonants in cueing for the presence of a particular vowel, but will also result in increased error rates for some vowels due to the allophones being the wrong ones for the context.

The regression equations produced above have some utility in helping me to select the best neural networks from a host of candidates (see Section 8.1) but their use for other purposes would be a little suspect. To produce better predictors of intelligibility error based on formant track errors and other measures would require a much larger amount of work. If the formant tracks used for synthesis were produced by one particular method the resulting regression equation would only be useful for prediction of intelligibility errors for vowels produced using that method. The alternative would be to systematically vary the original formant tracks extracted from speech. Phonetic knowledge of the effects of the shapes of vowel formants could guide the variations. The effects of varying F1, F2 and F3 could be isolated. Greater amounts of data would lead to regression equations which were more firmly based and which would have smaller confidence bounds, giving greater discriminatory power.

11.3 Further Work

It would be interesting to find the performance of ANNs trained on CVC triphones containing the full set of consonants and monophthongs available. The complexity of the mapping to be learnt would increase, but the amount of data available would increase greatly. The current work shows that there is generalisation from the effects of a consonant in a particular triphone to the effects in other triphones, so the increase in complexity of the mapping may well be roughly linear, whereas the number of usable triphones in the speech database will increase by roughly the proportionate increase in the number of consonants squared multiplied by the proportionate increase in the number of vowels (assuming a uniform distribution of phonemes, and random ordering, both of which are not realistic).

Re-running the experiments described in the thesis with more data might help determine the questions which were not satisfactorily answered, although it would require much more of computer time than has already been used. It might be preferable to use just use the best combinations as determined here in training on an increased set of data.

It would be interesting to compare the performance of feed-forward ANNs, trained either on the triphones used in this work or on a larger data set, against other vowel synthesis methods. A set of simple recurrent ANNs, as used in the experiments by Tuerk and Robinson [100], would be a good candidate. Another option would be to choose a model such as those discussed in Chapter 5 and determine the model parameters either by some statistical method or using the methods previously used with those models. I would be particularly interested in the performance of the Broad and Clermont [11] models.

The attempt to study the effects of using various numbers of hidden nodes failed. Using a method which automatically adjusts the number of hidden nodes, such as the cascade-correlation algorithm [24] might shed more light on this matter.

Section 11.2.1 discusses the limitations of my derivation of an equation which predicts a modified rhyme test intelligibility error from formant track errors. A systematic investigation of this relationship, looking at each formant track independently and using a variety of different formant track error measures, on a carefully selected set of utterances, might prove interesting and might yield a useful tool. However, it would involve a large amount of work and require a large amount of time to be spent running rhyme tests (or other perceptual tests) on subjects.

Appendix A

The *fsynth* Formant Synthesiser

The synthesised vowels used in evaluating the performance of the ANNs in producing vowel formant tracks were created using a formant synthesiser program called *fsynth*. This was written by the author of this thesis, in C++, with the intention of creating a formant synthesiser that could be configured however the user wished and which could be easily modified. C++ is an object-oriented programming language. The programmer creates *classes*, which are groups of associated variables and functions. When the program is run, *objects* are created, each of which is a member of a particular class. Different classes can have function or *methods* that have the same names but whose actions depend on the class. For instance, a time-step in the synthesiser is achieved by calling the *Step* method of each synthesiser component object, each of which does the correct thing for that type of component.

The *fsynth* synthesiser program is based on the Klatt formant synthesiser [51], using the same digital components. Each type of component has an associated C++ class. The user can specify which components to use, and how to connect them together, so that many different architectures of synthesiser are possible. It is a simple process to create new types of component, due to the object-oriented nature of the design. These can be incorporated into the existing program with minimal changes to the existing code, provided the new classes of component use the specified set of class methods to interact with the rest of the program.

A.1 Component Class Requirements

All component classes must be a daughter class of the *synth_object* class and must implement the following methods:

- *Set.* This method must set the internal parameters which govern the operation of the object. For instance, *resonators* set the frequency, bandwidth and interval between time frames. Values are read from the *controller* object as necessary.
- *Step.* This method runs the component object for one time-step. The new *output* value must be set, and must also be returned from the object.
- *Component-type.* This method returns a value of enumeration type *component-type*. This should be unique to the class, and serves to identify the type of class. This is not currently used, but may be useful in any further development.

A.2 Component Classes

A.2.1 Resonator

The *resonator* class implements the digital resonator, as defined in [51]. This is used to produce poles (such as formants) in the speech signal. The *step* method returns output value $y(nT)$, computed by

$$y(nT) = Ax(nT) + By(nT - T) + Cy(nT - 2T) \qquad \text{(A.1)}$$

where n is the output frame number, T is the time between frames, and $x(nt)$ is the current input. Values $y(nT-T)$ and $y(nT-2T)$ (the two previous outputs) are stored internally within the object. The *set* method reads the formant frequency F and the bandwidth W from the *controller* object and sets the constants A, B

and C. These are calculated as

$$\begin{aligned} C &= -\exp(-2\pi WT), & \text{(A.2)} \\ B &= 2\exp(-\pi WT) * \cos(2\pi FT), & \text{(A.3)} \\ A &= 1 - B - C. & \text{(A.4)} \end{aligned}$$

A.2.2 Antiresonator

The *antiresonator* class implements the antiresonator, used to create zeros in the speech signal. The *step* method returns output value $y(nT)$, computed by

$$y(nT) = A'x(nT) + B'y(nT - T) + C'y(nT - 2T) \qquad \text{(A.5)}$$

where the *step* method sets constants A', B' and C', calculated as

$$\begin{aligned} A' &= 1/A, & \text{(A.6)} \\ B' &= -B/A, & \text{(A.7)} \\ C' &= -CA. & \text{(A.8)} \end{aligned}$$

A.2.3 Impulse Generator

The *impulse_generator* class is used to create output impulses of value 1 at a frequency set by the *set* method. The frequency is used to derive the *period* of the impulses, rounded to the nearest whole number of time frames. The *step* method returns the value 1 every *period* number of time frames, otherwise it returns the value 0.

A.2.4 Differencer

The *differencer* class returns the difference between the current and previous inputs:

$$y(nT) = x(nT) - x(nT - T). \tag{A.9}$$

A.2.5 Amplitude Control

The *ampl_control* class acts as a gain control. The *set* method sets a gain *setting* which the *step* method multiplies by the input to give the output value:

$$y(nT) = \text{setting} * x(nT). \tag{A.10}$$

A.2.6 Noise Generator

The *noise_generator* class generates a noise signal. This is a pseudo-random number with a pseudo-gaussian distribution in the range (-1, 1) (created by summing 16 pseudo-random numbers).

A.2.7 Modulator

The *modulator* class applies a square-wave to its input. The *set* method sets the frequency of the square-wave. The *step* method returns the input value for the first half of the wave, and 0 for the second half of the wave.

A.2.8 Low-Pass Filter

The *low-pass-filter* class acts as a first-order low-pass digital filter. The *step* method returns output values given by

$$y(nT) = x(nT) + y(nT - T). \tag{A.11}$$

A.3 Using the *fsynth* Program

```
Usage :  fsynth <spec file> <parameter file> <output file>
```

A.3.1 The Specification File

The *specification* file specifies what components are to be used and how they are to be connected. The format of the file is as follows:

```
<int number of component objects>

<int component number> <char * component name> <char * component
type> <int number of predecessors> <int predecessor number>
<int predecessor number> ...
```

The first integer is the number of component objects. Each component is then specified. The first entry in the component field is a unique integer, greater than zero, which is used in referring to the component. These should run from one to the number of components. The highest numbered component is the output component. The next entry is a unique name to use for the component (this is not currently used). The third entry is the name of the component class. The fourth entry is the number of predecessor components. That is, the number of components whose outputs provide the input for this component. Finally there is a list of the component numbers of the predecessor components. There should be no loops in the synthesiser. An example of a specification file for a parallel formant synthesiser is:

```
16
1 impulse_generator impulse_generator 0
2 glottal_resonator_1 resonator 1 1
3 glottal_zero antiresonator 1 2
4 voicing_amplitude ampl_control 1 3
```

```
5 preformant_diff differencer 1 4
6 f1_amplitude ampl_control 1 4
7 nasal_amplitude ampl_control 1 5
8 f2_amplitude ampl_control 1 5
9 f3_amplitude ampl_control 1 5
10 f4_amplitude ampl_control 1 5
11 nasal_resonator resonator 1 7
12 f1_resonator resonator 1 6
13 f2_resonator resonator 1 8
14 f3_resonator resonator 1 9
15 f4_resonator resonator 1 10
16 radiation_characteristic differencer 5 11 12 13 14 15
```

A.3.2 The Parameter File

The *parameter* file controls the operation of the formant synthesiser. It is laid out as follows:

```
<float sample freq>
<float end time>

<int component number> <setting 1> <setting 2> ...
<int component number> <setting 1> ...
...
0

<float time>
<int component number> <setting 1> <setting 2> ...
<int component number> <setting 1> ...
...
0
```

```
...

0
<float time>
0
-999
```

The first entry is the sample frequency. The second entry is the time to run the synthesiser for. The next entries are initial settings for the components. Each field consists of a component number followed by the settings, as appropriate, for that component. A zero ends the initial settings.

Following the initial settings is a series of updates of the component settings, separated by zeros. Each update consists of the time at which to make the update, and a set of component numbers with their new settings. The final time given should match the end time value. A negative time (for example, -999) stops the synthesiser.

An example of a control file for a parallel formant synthesiser, matching the specification file above, is given below. The synthesiser runs for 0.5 seconds and only the F1 formant varies.

```
20000
0.5

1 100
2 0 100
3 1500 6000
4 60
6 80
7 80
8 80
9 80
```

```
10 80
11 250 100
12 450 50
13 1450 70
14 2450 110
15 4000 250
0
0.1
12 460 50
0
0.2
12 470 50
0
0.3
12 480 50
0
0.4
12 490 50
0
0.5
0
-999
```

A.3.3 The Output File

The output file is currently a speech data file using CSTR's *vox* format. A header gives details of the file contents. This is followed by the output values of the synthesiser, in *short int* format.

Appendix B

Intelligibility Test Materials

The following two pages show the cover page and first page of the response booklet given to subjects in Experiment I. See Chapter 7 for more details.

Instructions

You will hear a series of utterances consisting of an initial consonant, a vowel and a final consonant (a triphone). The utterances are in an English RP accent. For each utterance you will have a choice of six possible words. Ring the word that is closest to what you heard.

For example, if you heard the word "tot", and were given the choices

123.   a) tart   b) tut   c) tot   d) turt   e) taut   f) toot

you would ring the word "tot":

123.   a) tart   b) tut   [✓ c) tot]   d) turt   e) taut   f) toot

Ring a word for each utterance. Even if none matches what you heard, ring the closest word. Note that it is the intelligibility of the utterances which is being tested, not you! Some of the words on the response sheet, and some of the utterances, may be nonsense words.

The sets of choices on the response sheet are numbered, but the tape has only the utterances, so you will have to make sure you retain your place on the page.

The utterances will be played continuously, with a three second gap between them. There will be a pause at the end of each page, and there will a short break halfway through the words. Before the test words you will hear 10 example words: bed, bird, deep, talk, cut, got, park, big, bad and book. Please indicate below whether you have an English, Scottish or other accent :

Scottish □   English □   Other (specify) □

SECTION ONE

1.	a) dart	b) doot	c) dit	d) dirt	e) daught	f) deet
2.	a) bart	b) bit	c) bet	d) bat	e) boot	f) bert
3.	a) perk	b) peck	c) peak	d) pock	e) pork	f) pick
4.	a) dub	b) dab	c) dib	d) durb	e) dob	f) deb
5.	a) tuck	b) tack	c) turk	d) tick	e) tech	f) teak
6.	a) bag	b) berg	c) bug	d) beeg	e) big	f) beg
7.	a) dob	b) dab	c) dub	d) deb	e) dib	f) durb
8.	a) dorp	b) dep	c) dip	d) doop	e) dap	f) dop
9.	a) dag	b) dig	c) dawg	d) dug	e) dog	f) dirg
10.	a) teg	b) tag	c) tog	d) tug	e) teague	f) tig
11.	a) cup	b) coop	c) kep	d) kip	e) curp	f) keep
12.	a) peak	b) pork	c) perk	d) pock	e) peck	f) pick
13.	a) bart	b) bert	c) bat	d) bit	e) bet	f) boot
14.	a) peak	b) perk	c) pork	d) pick	e) pock	f) peck
15.	a) bib	b) bab	c) boob	d) bob	e) barb	f) borb
16.	a) dug	b) dirg	c) dig	d) dag	e) dog	f) dawg
17.	a) cart	b) cot	c) ket	d) cat	e) curt	f) coot
18.	a) tug	b) tog	c) tig	d) teague	e) tag	f) teg
19.	a) tack	b) tuck	c) tech	d) tick	e) turk	f) teak
20.	a) beg	b) bag	c) bug	d) beeg	e) big	f) berg
21.	a) ked	b) curd	c) could	d) cawed	e) cod	f) card
22.	a) boot	b) bart	c) bert	d) bat	e) bit	f) bet
23.	a) durb	b) deb	c) dob	d) dab	e) dub	f) dib
24.	a) dip	b) doop	c) dep	d) dap	e) dop	f) dorp
25.	a) tag	b) tig	c) teague	d) teg	e) tog	f) tug
26.	a) dag	b) dirg	c) dig	d) dawg	e) dug	f) dog
27.	a) berg	b) bag	c) beeg	d) beg	e) bug	f) big
28.	a) peak	b) pick	c) pock	d) pork	e) peck	f) perk
29.	a) kip	b) coop	c) keep	d) cup	e) curp	f) kep
30.	a) cod	b) could	c) cawed	d) card	e) curd	f) ked
31.	a) bab	b) barb	c) borb	d) boob	e) bib	f) bob
32.	a) dap	b) doop	c) dep	d) dorp	e) dip	f) dop
33.	a) coot	b) curt	c) cat	d) cot	e) cart	f) ket

Appendix C

The Machine Readable Phonetic Alphabet (MRPA)

The main text of this thesis uses IPA notation. However, in the figures I have had to use an ASCII based notation called the Machine Readable Phonetic Alphabet (MRPA), developed by the Centre for Speech Technology Research (CSTR), Edinburgh University [61]. The relevant MRPA symbols are listed below with their IPA equivalents and example words.

IPA	MRPA	Example	IPA	MRPA	Example
p	p	*p*ea	a	a	b*a*d
t	t	*t*ea	ɑ	aa	b*ar*d
k	k	*k*ey	ʌ	uh	b*u*d
b	b	*b*ee	ɜ	@@	b*ir*d
d	d	*d*ye	ə	@	*a*bout
g	g	*g*uy	ɒ	o	p*o*t
ɪ	i	b*i*d	ɔ	oo	p*or*t
i	ii	b*ea*d	ʊ	u	p*u*t
ɛ	e	b*e*d	u	uu	b*oo*t

Bibliography

[1] Jonathon Allen, M. Sharon Hunnicutt, and Dennis Klatt. *From Text to Speech: the MITalk System.* Cambridge University Press, 1987.

[2] Robert B. Allen. Conncctionist state machines, November 1988.

[3] Robert B. Allen. Sequential connectionist networks for answering simple questions about a microworld. In *Proceedings of the Cognitive Science Society*, pages 489–495, August 1988.

[4] Robert B. Allen. Adaptive training for connectionist state machines. In *ACM Computer Science Conference*, page 428, February 1989.

[5] Robert B. Allen and Mark E. Riecken. Reference in connectionist language users. In *Connectionism in Perspective*, October 1988.

[6] Edgar Anderson. The irises of the Gaspe Peninsula. *Bulletin of the American Iris Society*, 59:2–5, 1935.

[7] Michael H. L. Hecker Arthur S. House, Carl E. Williams and K. D. Kryter. Articulation-testing methods : Consonantal differentiation with a closed-response set. *Journal of the Acoustical Society of America*, 37:158–166, January 1965.

[8] Eric B. Baum. On the capabilities of multilayer perceptrons. *Journal Of Complexity*, 4:193–215, 1988.

[9] Eric B. Baum and David Haussler. What size net gives valid generalization? *Neural Computation*, 1:151–160, 1989.

[10] Richard A. Becker, John M. Chambers, and Allan R. Wilks. *The New S Language.* Computer Science Series. Wadsworth and Brooks/Cole, California, 1988.

[11] David J. Broad and Frantz Clermont. A methodology for modeling vowel formant contours in CVC context. *Journal of the Acoustical Society of America*, 81(1):155–165, January 1987.

[12] David J. Broad and Ralph H. Fertig. Formant-frequency trajectories in selected CVC-syllable nuclei. *Journal of the Acoustical Society of America*, 47(6):1572–1582, 1970.

[13] C.P. Browman. Rules for demisyllable synthesis using LINGUA, a language interpreter. In *ICASSP-80*, pages 561–564. Institute of Electrical and Electronic Engineers, 1980.

[14] C.P. Browman and L.M. Goldstein. Towards an articulatory phonology. *Phonology Yearbook*, 3:219–252, 1986.

[15] Christopher Chatfield and Alexander J. Collins. *Introduction to Multivariate Analysis.* Chapman and Hall, London, 1980.

[16] N. Chomsky and M. Halle. *The Sound Pattern of English.* Harper and Row, New York, 1968.

[17] John Coleman. "Synthesis-by-rule" without segments or rewrite-rules. In G. Bailly, C. Benoit, and T.R. Sawallis, editors, *Talking Machines: Theories, Models and Designs*, pages 43–60. Elsevier Science Publishers B.V., 1991.

[18] A. Crowe and M.A. Jack. A globally optimising formant tracker using generalised centroids. *Electronics Letters*, 23:1019–1020, 1987.

[19] G. Cybenko. Approximation by superposition of a sigmoidal function. *Math. Control Systems Signals*, 2:303–314, 1989.

[20] T.G. Dietterich, H. Hild, and G. Bakiri. A comparason of ID3 and backpropagation for English text-to-speech mapping. Technical Report OR 97331-3102, Oregan State University, 1990.

[21] T.G. Dietterich, H. Hild, and G. Bakiri. A comparative study of ID3 and backpropagation for English text-to-speech mapping. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 24–31. Morgan Kaufmann, 1990.

[22] Jeffrey L. Elman. Finding structure in time. CRL 8801, Centre for Research In Language, UCSD., April 1988.

[23] J.L. Elman and D. Zipser. Learning the hidden structure of speech. Technical Report ICS Report 8701, UCSD, February 1987.

[24] S.E. Fahlman and C. Lebiere. The cascade-correlation learning architecture. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*. Morgan Kaufmann, 1988.

[25] G. Fairbanks. Test of phonemic differentiation : The rhyme test. *Journal of the Acoustical Society of America*, 30:596–601, July 1958.

[26] F. Fallside and S.J. Young. Speech output from a computer-controlled water supply network. volume 125, pages 157–161. Institute of Electrical and Electronic Engineers, 1978.

[27] C.G.M. Fant. On the predictability of formant levels and spectrum envelopes from formant frequencies. In *For Roman Jakobson*, pages 109–120. Mouton, 1956.

[28] Mark Fanty. Context-free parsing in connectionist networks. Technical Report CS TR174, Rochester, November 1985.

[29] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[30] J.L. Flanagan, K. Ishizaka, and K.L. Shipley. Synthesis of speech from a dynamic model of the vocal cords and vocal tract. *Bell System Technical Journal*, 54:485–506, 1975.

[31] C. Fowler. Coarticulation and theories of extrinsic timing. *J. Phonetics*, 8:113–133, 1980.

[32] M. A. Franzini, K. Lee, and Waibel. Connectionist Viterbi training: A new hybrid method for continuous speech recognition. In *ICASSP 90*, pages 425–428, 1990.

[33] M. A. Franzini, M. J. Witbrock, and K. Lee. A connectionist approach to continuous speech recognition. In *ICASSP 89*, pages 425–428, 1989.

[34] K. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2:183–192, 1989.

[35] John S. Garofolo. *Getting Started with the DARPA TIMIT CD-ROM: An Acoustic Phonetic Continuous Speech Database.* National Institute of Standards and Technology (NIST), Gaithersburgh, MD, 1988.

[36] Michael Gasser. Networks that learn phonology. Technical report, Indiana University, 1990.

[37] C.L. Giles, C.B. Millar, D. Chen G.Z. Sun, H.H. Chen, and Y.C. Lee. Extracting and learning an unknown grammar with recurrent neural networks. In J.E. Moody, S.J. Hanson, and R.P. Lippmann, editors, *Advances in Neural information Processing Systems*, volume 4. Morgan Kaufmann, San Mateo, Ca., 1992.

[38] J.A. Goldsmith. *Autosegmental Phonology.* Indiana University Linguistics Club, Bloomington, 1976.

[39] C. Hamon, E. Moulines, and F Charpeitier. A diphone synthesis system based on time-domain modifications of speech. In *ICASSP-89*, 1989.

[40] J. Hertz, A. Krogh, and R.G. Palmer. *Introduction to the Theory of Neural Computation.* Addison-Wesley, 1991.

[41] J.N. Holmes, Ignatius G. Mattingly, and J.N. Shearme. Speech synthesis by rule. *Language and Speech*, 7:127–143, 1964.

[42] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to automata theory, languages and computation.* Addison-Wesley Series in Computer Science. Addison-Wesley Publishing Company, Inc., 1979.

[43] Caroline B. Huang. Modelling human vowel formant trajectory and context. In Y. Tohkura, E. Vatikiotis-Bateson, and Y. Sagisaka, editors, *Speech Perception, Production and Linguistic Structure*, pages 43–61. IOS Press, 1992.

[44] Arthur Hughes and Peter Trudgill. *English Accents and Dialects: An Introduction to Social and Regional Varieties of British English.* Edward Arnold (Publishers) Ltd, London, 1979.

[45] Satoshi Imaizumi and Shigeru Kiritani. A generation model of formant trajectories at variable speaking rates. In G. Bailly, C. Benoit, and T.R. Sawallis, editors, *Talking Machines: Theories, Models and Designs*, pages 61–75. Elsevier Science Publishers B.V., 1992.

[46] Yasushi Ishikawa and Kunio Nakajima. Neural network based concatenation method of synthesis units for synthesis by rule. In *Proceedings ICSLP 90*, volume 2, pages 793–796, 1990.

[47] F. Itakura and S. Saito. Analysis synthesis telephony based upon the maximum likelihood method. In Y. Kohasi, editor, *Reports of 6th International Congress of Acoustics.* Tokyo, 1968.

[48] F. Itakura and S. Saito. A statistical method for estimation of speech spectral density and formant frequencies. *Electronics and Communication*, 53-A:36–43, 1970.

[49] F. Itakura and S. Saito. On the optimal quantization of feature parameters in the parcor speech synthesiser. In *Conf. Speech Commun. and Process.*, pages 434–437, New York, 1972.

[50] Michael I. Jordan. Serial order: A parallel distributed approach. Technical Report ICS Report 8604, UCSD, May 1986.

[51] Dennis H. Klatt. Software for a cascade/parallel formant synthesizer. *J. Acoust. Soc. Am.*, 67(3):971–995, 1980.

[52] D.H. Klatt. The Klattalk text-to-speech system. In *ICASSP-82*, pages 1589–1592, 1982.

[53] David G. Kleinbaum, Lawrence L. Kupper, and Keith E. Muller. *Applied Regression Analysis and Other Multivariate Methods.* PWS-KENT Publishing Company, Boston, second edition, 1988.

[54] A.H. Kramer and A. Sangiovanni-Vincentelli. Efficient parallel learning algorithms for neural networks. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume I, pages 40–49. Morgan Kaufmann, 1989.

[55] H. Kuchera and W.N. Francis. *Computational Analysis of Modern-Day American English.* Brown University Press, Providence, Rhode Island, 1967.

[56] Vinod V. Kumar, Stanley C. Ahalt, and Ashok K. Krishnamurthy. Phonetic to acoustic mapping using neural networks. In *International Conference on Acoustics, Speech and Signal Processing*, pages 753–756. IEEE, 1991.

[57] Stan C. Kwasny. A parallel distributed approach to parsing natural language deterministically. Technical Report CS WUCS-88-21, Washington University, August 1988.

[58] Peter Ladefoged. *A Course in Phonetics.* Harcourt Brace Jovanovich, Inc., second edition, 1982.

[59] G. Lakoff. *Women, Fire and Dangerous Things: What Categories Reveal About the Mind.* University of Chicago Press, 1987.

[60] G. Lakoff. A suggestion for a linguistics with connectionist foundations. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988*

Connectionist Models Summer School, pages 301–314. Morgan Kaufmann, 1988.

[61] J. Laver, C. Bennett, I. Cohan, J. Dalby, D. Davies, and M. McAllister. ATR/CSTR speech database project. Status Report 1, Centre for Speech Technology Research, University of Edinburgh, U.K., 1988.

[62] A. Liberman, F. Ingermann, L. Lisker, P. Delattre, and F. Cooper. Minimal rules for synthesising speech. *Journal of the Acoustical Society of America*, 31:1490–1499, 1959.

[63] J. Liljencrants. Speech synthesizer control by smoothed step functions. Speech Transmission Laboratory Quarterly Progress and Status Report QPSR-4/1969, Royal Institute of Technology (KTH), Stockholm, 1970.

[64] R. Linggard. *Electronic synthesis of speech.* Cambridge University Press, 1985.

[65] J. Local. Modelling assimilation in non-segmental rule-free synthesis. Technical report, Experimental Phonetics Laboratory, Department of Language and Linguistic Science, University of York, 1989.

[66] J.B. Lovins, M.J. Macchi, and O. Fujimura. A demisyllable inventory for speech synthesis. *Journal of the Acoustical Society of America*, 65:5130–31, 1979.

[67] J.D. Markel and Jr. A.H. Gray. *Linear Prediction of Speech*, volume 12 of *Communication and Cybernetics.* Springer-Verlag, Berlin, Heidelberg, New York, 1976.

[68] J.L. McClelland and J.L. Elman. Interactive processes in speech perception: The trace model. In J.L. McClelland, D.E. Rumelhart, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 2, chapter 15, pages 58–121. MIT Press, 1986.

[69] J.L. McClelland, D.E. Rumelhart, and the PDP Research Group. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 2: Psychological and Biological Models. MIT Press, 1986.

[70] Risto Miikkulainen and Michael G. Dyer. A modular neural network architecture for sequential paraphrasing of script-based stories. Technical Report UCLA-AI-89-02, UCLA, February 1989.

[71] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry.* MIT Press, 1969.

[72] Michael C. Mozer. A focused back-propagation algorithm for temporal pattern recognition. Technical report, University of Toronto Department of Psychology and Computer Science, June 1988.

[73] J.P. Olive and L.H. Nakatina. Rule-synthesis of speech by word concatenation: A first step. *Journal of the Acoustical Society of America*, 55:660–666, 1974.

[74] J.P. Olive and N. Spickenagel. Speech resynthesis from phoneme-related parameters. *Journal of the Acoustical Society of America*, 59:993, 1976.

[75] Peterson, Wang, and Sivertson. *J. Acoust. Soc. Am.*, 30:739–742, 1958.

[76] S. Pinker and A. Prince. On language and connectionism: analysis of a parallel distributed processing model of language acquisition. In S. Pinker and J. Mehler, editors, *Connections and Symbols.* MIT Press, 1988.

[77] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T Vetterling. *Numerical Recipes in C.* Cambridge University Press, 1988.

[78] J.R. Quinlan. Learning efficient classification proceedures and their application to chess endgames. In R.S. Michalski, J. Carbonell, and T.M. Mitchell, editors, *Machine learning: An artificial intelligence approach.*, volume I, pages 463–482. Tioga Press, Palo Alto, 1983.

[79] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 86.

[80] James A. Reggia, Patricia M. Marsland, and Rita Sloan Berndt. Competitive dynamics in a dual-route connectionist model of print-to-sound transformation. *Complex Systems*, 2:509–547, 1988.

[81] Steve Renals and Richard Rohwer. Learning phoneme recognition using neural networks. In *ICASSP-89*, 1989.

[82] Steve Renals, Richard Rohwer, and Mark Terry. A comparison of speech recognition front ends using a connectionist classifier. *Proc. Speech '88*, 4, August 1988.

[83] Richard Rohwer. Connectionist methods in speech recognition. *Proc. Speech '88*, 3, August 1988.

[84] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, chapter 8, pages 318–364. MIT Press, 1986.

[85] D.E. Rumelhart and J.L. McClelland. On learning the past tenses of English verbs. In James L. McClelland, David E. Rumelhart, and the PDP Research Group, editors, *Parallel Distributed Processing*, volume 2, chapter 18, pages 216–271. MIT Press, 1986.

[86] D.E. Rumelhart, J.L. McClelland, and the PDP Research Group. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1: Foundations. MIT Press, 1986.

[87] Cullen Schaffer. Overfitting avoidance as bias. *Machine Learning*, 10:153–178, 1993.

[88] H. Schnelle and R. Doust. A net-linguistic "Earley" chart-parser. In R. Reilly and N.E. Sharkey, editors, *Connectionist Approaches to Languages*, volume I. North-Holland, Amsterdam, 1989.

[89] R. Schwartz, J. Klovstad, J. Makhoul, D. Klatt, and V. Zue. Diphone synthesis for phonetic vocoding. In *ICASSP-79*, pages 891–894. IEEE, 1979.

[90] M. S. Scordilis and J. N. Gowdy. Neural network control for a cascade/parallel formant synthesiser. *ICASSP 90*, 1:297–300, 1990.

[91] Michael S. Scordilis and John N. Gowdy. Neural network based generation of fundamental frequency contours. In *International Conference on Acoustics, Speech and Signal Processing*, pages 219–222. IEEE, 1989.

[92] C. Scully. Articulatory synthesis. In W.J. Hardcastle and A. Marchal, editors, *Speech Production and Speech Modelling*, volume 55 of *NATO ASI Series D*, pages 151–186. Kluwer Academic Publishers, 1990.

[93] Terence J. Sejnowski and Charles R. Rosenberg. Parallel networks that learn to pronounce English text. *Complex Systems*, 1:145–168, 1987.

[94] S. Seneff. A computational model for the peripheral auditory system: application to speech recognition research. In *ICASSP 86*, pages 1983–1986. IEEE, Tokyo, 1986.

[95] David Servan-Schreider, Axel Cleeremans, and James L McClelland. Encoding sequential structure in simple recurrent networks. Technical Report CMU-CS-88-183, CMU, November 1988.

[96] Mark F. St.John and James L. McClelland. Applying contextual constraints in sentence comprehension. Technical report, Connectionist Models Summer School, 1988.

[97] G.Z. Sun, H.H. Chen, C.L. Giles, Y.C. Lee, and D. Chen. Connectionist pushdown automata that learn context-free grammars. In *Proceedings of the International Joint Conference on Neural Networks*, volume I, page 577. Lawrence Erlbaum, 1990.

[98] David S. Touretzky. BoltzCONS: Reconciling connectionism with the recursive nature of stacks and trees. In *Proc. 8th Conference of the Cognitive Science Society*, pages 522–530, 1986.

[99] C. Traber. F_0 generation with a database of natural F_0 patterns and with a neural network. In G. Bailly, C. Benoit, and T.R. Sawallis, editors, *Talking Machines: Theories, Models and Designs*, pages 287–304. Elsevier Science Publishers B.V., 1992.

[100] C. Tuerk and T. Robinson. Speech synthesis using artificial neural networks trained on cepstral coefficients. In *Eurospeech 93*, pages 1713–1716, 1993.

[101] Christine Tuerk, Peter Monaco, and Tony Robinson. The development of a connectionist multiple-voice text-to-speech system. In *International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 749–752. IEEE, 1991.

[102] W. von Kempelen. *Le Méchanisme de la Parole, Suivi de la Description d'une Machine Parlante.* J.V. Degan, Vienna, 1791.

[103] J.C. Wells. *Accents of English*, volume 2. Cambridge University Press, Cambridge, 1982.

[104] W.A. Wickelgren. Context-sensitive coding, associative memory, and serial order in (speech) behavior. *Psychological Review*, 76:1–15, 1969.

[105] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. Technical Report ICS Report 8805, Institute of Cog.Sci, UCSD., October 1988.

[106] David H. Wolpert. On overfitting avoidance as bias. Technical report, The Santa Fe Institute, 1993.

[107] E. Zwicker. Subdivision of the audible frequency range into critical bands (frequenzgruppen). *Journal of the Acoustical Society of America*, 33:248, 1961.